



INTERNATIONAL TELECOMMUNICATION UNION

**ITU-T**

TELECOMMUNICATION  
STANDARDIZATION SECTOR  
OF ITU

**M.3100**

(07/95)

**MAINTENANCE**

**TELECOMMUNICATIONS MANAGEMENT NETWORK**

---

**GENERIC NETWORK INFORMATION MODEL**

**ITU-T Recommendation M.3100**

(Previously "CCITT Recommendation")

---

## FOREWORD

The ITU-T (Telecommunication Standardization Sector) is a permanent organ of the International Telecommunication Union (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 (Helsinki, March 1-12, 1993).

ITU-T Recommendation M.3100 was revised by ITU-T Study Group 4 (1993-1996) and was approved under the WTSC Resolution No. 1 procedure on the 27th of July 1995.

---

## NOTE

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

© ITU 1996

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the ITU.

## CONTENTS

	<i>Page</i>
1 Scope, purpose, and field of application .....	1
1.1 Scope .....	1
1.2 Purpose .....	1
1.3 Field of application .....	1
1.4 Structure of this Recommendation.....	2
2 Overview of the network model.....	2
3 Object classes .....	3
3.1 Network fragment.....	12
3.2 Managed element fragment.....	12
3.3 Termination point fragment.....	17
3.4 Switching and transmission fragment.....	21
3.5 Cross-connection fragment.....	24
3.6 Functional area fragment .....	28
4 Packages.....	30
4.1 Administrative Operational States .....	30
4.2 Affected Object List .....	30
4.3 Alarm Severity Assignment Pointer .....	30
4.4 Attribute Value Change Notification.....	30
4.5 Audible Visual Local Alarm.....	30
4.6 Channel Number.....	30
4.7 Characteristic Information .....	31
4.8 Client Connection List.....	31
4.9 Client Trail.....	31
4.10 Create Delete Notifications.....	31
4.11 Cross-Connection Pointer .....	31
4.12 CTP Instance.....	31
4.13 Current Problem List .....	31
4.14 Environmental Alarm .....	31
4.15 Environmental AlarmR1 .....	32
4.16 Equipment Alarm Effect on Service.....	32
4.17 Equipments Equipment Alarm.....	32
4.18 Equipments Equipment AlarmR1 .....	32
4.19 External Time .....	32
4.20 Location Name.....	32
4.21 Named Cross-Connection .....	32
4.22 Network Level .....	33
4.23 Operational State .....	33
4.24 Object Management Notifications .....	33
4.25 Processing Error Alarm .....	33
4.26 Processing Error AlarmR1 .....	33
4.27 Protected .....	33
4.28 Reset Audible Alarm .....	33
4.29 Server Connection List .....	34
4.30 Server Trail List.....	34
4.31 Software Processing Error Alarm .....	34
4.32 Software Processing Error AlarmR1 .....	34
4.33 Subordinate Circuit Pack .....	34
4.34 Supportable Client List.....	34
4.35 State Change Notification.....	34

	<i>Page</i>
4.36 System Timing Source.....	35
4.37 TMN Communications Alarm Information.....	35
4.38 TTP Instance.....	35
4.39 User Label.....	35
4.40 Vendor Name.....	35
4.41 Version.....	35
5 Attributes.....	36
5.1 A-Termination Point Instance.....	36
5.2 Acceptable Circuit Pack Type List .....	36
5.3 Administrative State .....	36
5.4 Affected Object List .....	36
5.5 Alarm Severity Assignment List.....	36
5.6 Alarm Severity Assignment Profile Id.....	37
5.7 Alarm Severity Assignment Profile Pointer.....	37
5.8 Alarm Status .....	37
5.9 Channel Number .....	37
5.10 Characteristic Information .....	38
5.11 Circuit Directionality .....	38
5.12 Circuit End Point Subgroup Id .....	38
5.13 Circuit Pack Type .....	38
5.14 Client Connection List.....	38
5.15 Client Trail.....	39
5.16 Connected Termination Point Count .....	39
5.17 Connection Id .....	39
5.18 Connection Termination Point Id .....	39
5.19 Cross-Connection Id .....	40
5.20 Cross-Connection Name .....	40
5.21 Cross-Connection Object Pointer .....	40
5.22 Current Problem List .....	40
5.23 Directionality .....	40
5.24 Downstream Connectivity Pointer .....	41
5.25 Equipment Holder Address.....	41
5.26 Equipment Holder Type.....	41
5.27 Equipment Id .....	42
5.28 External Time .....	42
5.29 Fabric Id.....	42
5.30 From Termination.....	42
5.31 Group Termination Point Id.....	42
5.32 Holder Status .....	43
5.33 Idle TP Count.....	43
5.34 Information Transfer Capabilities.....	43
5.35 Label of Far End Exchange .....	43
5.36 List of Characteristic Info .....	44
5.37 Location Name.....	44
5.38 Managed Element Id.....	44
5.39 Managed Element Complex Id .....	44
5.40 Multi-Point Cross-Connection Id.....	44
5.41 Network Id.....	45
5.42 Network Level Pointer.....	45
5.43 Number of Circuits .....	45
5.44 Operational State .....	45
5.45 Protected .....	45
5.46 Redline.....	46

	<i>Page</i>
5.47 Replaceable.....	46
5.48 Serial Number.....	46
5.49 Server Connection List .....	46
5.50 Server Trail List .....	46
5.51 Signalling Capabilities.....	47
5.52 Signal Type.....	47
5.53 Software Id .....	47
5.54 Subordinate Circuit Pack Software Load.....	47
5.55 Supportable Client List .....	48
5.56 Supported By Object List .....	48
5.57 System Timing Source.....	48
5.58 System Title .....	48
5.59 Total TP Count .....	48
5.60 To Termination .....	48
5.61 TP Pool Id .....	49
5.62 TPs In GTP List .....	49
5.63 TPs In TP Pool List.....	49
5.64 Trail Id .....	49
5.65 Trail Termination Point Id .....	50
5.66 Transmission Characteristics .....	50
5.67 Upstream Connectivity Pointer.....	50
5.68 Usage State .....	50
5.69 User Label.....	50
5.70 Vendor Name.....	50
5.71 Version.....	51
5.72 Z-Termination Point Instance .....	51
6 Name Bindings.....	51
6.1 Alarm Record.....	51
6.2 Alarm Severity Assignment Profile .....	51
6.3 Circuit Pack .....	53
6.4 ConnectionR1 .....	53
6.5 Connection Termination Point Source.....	53
6.6 Connection Termination Point Sink .....	54
6.7 Cross-Connection .....	55
6.8 Equipment.....	55
6.9 Equipment Holder.....	56
6.10 Event Forwarding Discriminator .....	56
6.11 Fabric .....	57
6.12 GTP.....	57
6.13 Log.....	57
6.14 Managed Element .....	57
6.15 Managed Element Complex.....	58
6.16 Multi-Point Cross-Connection .....	59
6.17 Network .....	59
6.18 Software.....	59
6.19 TP Pool .....	60
6.20 TrailR1 .....	60
6.21 Trail Termination Point Source .....	61
6.22 Trail Termination Point Sink .....	61
7 Actions .....	61
7.1 Add TPs To GTP .....	61
7.2 Add TPs To TP Pool .....	61
7.3 Allow Audible Visual Local Alarm.....	62

	<i>Page</i>
7.4 Connect .....	62
7.5 Disconnect .....	63
7.6 Inhibit Audible Visual Local Alarm .....	63
7.7 Remove TPs From GTP.....	63
7.8 Remove TPs From TP Pool .....	64
7.9 Reset Audible Alarm .....	64
7.10 Switch Over .....	64
8 Notifications .....	64
8.1 Attribute Value Change .....	64
8.2 Communications Alarm .....	65
8.3 Environmental Alarm .....	65
8.4 Equipment Alarm.....	65
8.5 Object Creation .....	65
8.6 Object Deletion .....	65
8.7 Processing Error Alarm .....	65
8.8 State Change .....	65
9 Parameters .....	65
9.1 Alarm Effect On Service Parameter .....	65
9.2 Create Error .....	65
10 ASN.1 defined types module .....	66
10.1 Rules of Extensibility.....	66
10.2 ASN.1 Module .....	66
11 TMN application context .....	74
12 Entity-Relationship diagrams .....	74
Annex A – Index .....	74
A.1 Managed objects .....	74
A.2 Packages .....	75
A.3 Attributes .....	76
A.4 Name bindings .....	77
A.5 Actions .....	78
A.6 Notifications .....	78
A.7 Parameters.....	78
Appendix I – User guidelines .....	78
I.1 Introduction .....	78
I.2 Use of supported by object list .....	78
I.3 Use of upstream and downstream connectivity pointers .....	78
I.4 Use of cross-connection objects .....	79
I.5 Cross-connection use examples .....	79
I.6 Object classes and logical layering .....	84
I.7 Mandatory naming attribute.....	85
I.8 Interoperability between Recommendation M.3100 (1992) and this Recommendation.....	87
I.9 Support for multi-point trails .....	87
Appendix II – Deprecated/obsolete management information from Recommendation M.3100 (1992) .....	87
II.1 Object classes.....	87
II.2 Packages .....	89
II.3 Attributes .....	89
II.4 Name binding.....	89
Appendix III – Differences from Recommendation M.3100 (1992).....	90
References .....	90

## **ABSTRACT**

This Recommendation provides a generic network information model. The model describes managed object classes and their properties that are generic and useful to describe information exchanged across all interfaces defined in M.3010 TMN architecture. These generic managed object classes are intended to be applicable across different technologies, architectures and services. The managed object classes in this Recommendation may be specialized to support the management of various telecommunications networks.

## **KEYWORDS**

Actions; ASN.1; Attributes; Generic Network Information Model; Managed Object Class; Notifications.





## **GENERIC NETWORK INFORMATION MODEL**

*(revised 1995)*

### **1 Scope, purpose, and field of application**

#### **1.1 Scope**

This Recommendation provides a generic network information model. It identifies TMN object classes that are common to managed telecommunications networks; or are of a generic type that can be used to manage a network at a technology-independent level; or are super-classes of technology-specific managed objects in a telecommunications network; or management support objects that are required for the management of the telecommunications network. These objects are relevant to information exchanged across standardized interfaces defined in the Recommendation M.3010 TMN architecture[1].

This Recommendation addresses generically the abstractions of those aspects of telecommunication resources (e.g. equipments, telecommunication services) required to manage the network. It also includes the abstractions related to the management services. Recommendation G.803 on the architecture of the transport network is used as the basis in developing the transport aspects of this model.

This Recommendation does not address abstractions relevant to technology-specific areas or implementation-specific details.

#### **1.2 Purpose**

##### **1.2.1 Interoperability**

There will be a variety of TMN conformant management systems and managed systems concerning many technology-specific areas, such as switching and transmission. One purpose of this Recommendation is to provide a vehicle for management interoperability between such systems.

##### **1.2.2 Technology independent management**

By introducing the concept of technology-independent management, it is possible to perform management of diverse equipment using common communications interfaces. In this manner, an “abstract” view over a set of network-elements can be achieved.

##### **1.2.3 Facilitating information model development**

This Recommendation also provides a framework from which technology-specific information models may be developed using the modelling principles defined in Recommendation X.720[2].

### **1.3 Field of application**

This Recommendation captures the generally applicable requirements of the technology-independent and technology-specific information models as well as information relating to TMN management services.

Through specialization, this Recommendation is applicable to technology-specific TMN information models. The mechanism for specialization is inheritance.

Even though technology-specific models may be derived from this Recommendation, some of the generic managed object classes in this Recommendation are instantiable in order to provide interoperability between equipment supporting information models derived from this Recommendation and equipment that only supports the information model in this Recommendation.

## 1.4 Structure of this Recommendation

Clause 2 provides an overview of the generic network model in this Recommendation. The definition of management information in clauses 3 to 8, describing information model is documented using the notational mechanisms defined in Recommendation X.722[3]. The relationships between the managed object classes for the different fragments of the model in clause 3 are depicted using entity relationship diagrams. Clause 9 contains the syntax definitions of the information carried in the protocol. The notation used is Abstract Syntax Notation One (ASN.1) defined in Recommendation X.208 | X.680-3 [4].

When referencing the definitions for the templates in this Recommendation by other documents, the prefix “Recommendation M.3100” should be used to identify the source for the definitions.

## 2 Overview of the network model

A generic network information model is essential to the generation of uniform fault, configuration, performance, security, and accounting management standards. A common network model, identifying the generic resources that exist in a network and their associated attribute types, events, actions, and behaviours, provides a foundation for understanding the interrelationships between these resources and attributes, and may, in turn, promote uniformity in dealing with the various aspects of managing these resources and attributes.

Network resources may be customer- or provider-owned; the latter includes portions that may be assigned for exclusive use by specific customers. Resources may be physical or logical in nature. Physical resources include customer (e.g. PBXs) or provider (e.g. digital cross-connect systems) systems, their associated subsystems (e.g. a line card within a PBX) and also the links that interconnect these systems. Such systems are generally known as Network Elements (NEs). Logical resources include communication protocols, application programs, logs, and network services.

There may also exist (separate or integrated) Telecommunications Management Network (TMN) resources involved in operating a telecommunications network. These resources include the Operations Systems (OSs) closely associated with managing specific NEs, and OSs that have network-wide responsibilities.

Resources have attributes which allow the user to control and/or observe the behaviour of the resource. Attributes may also allow the user to control and/or observe the relationships between resources.

There is a need to represent the way resources, or entities can be combined and interrelated (relationships). In this version, Entity-Relationship (E-R) diagram techniques have been used to represent inter-object relationships. As these tools are improved, newer ones may be used in future issues.

These E-R diagrams result in a high-level view (schema) of the Generic Network Information Model. This view can be used to derive information related to naming, to verifying consistency, and to ensuring completeness. For example, it ensures that sufficient information (i.e. relationships) is provided from a physical resource to identify the services that are dependent on that resource.

The information exchanged at the management interface is modelled using design principles outlined in Recommendation X.720[2], Management Information Model. Resources are modelled as objects, and the management view of a resource is called a managed object. Additional objects, called support managed objects, are defined to support the functions of managing a telecommunications network.

Objects with similar attributes and behaviours may be grouped into object classes. An object is characterized by its object class and object instance, and may possess multiple attribute types and associated values. Similarly, the terms managed object class and managed object instance apply specifically to objects that are being managed. This Recommendation specifies the properties of the resource (i.e. managed object) visible for management.

An object class may be a subclass of another object class. A subclass inherits attribute types and behaviours of its super-class, in addition to possessing its own specific attributes and properties.

Object classes and attribute types are defined only for the purpose of communicating network management messages between systems, and need not be related to the structuring of data within these systems. Some object classes defined in these issues (and future issues) of the model apply to many management functional areas, while others support specific functional areas.

This version of the Generic Network Information Model contains common object classes and attribute types as well as those particular to Alarm Surveillance. Subsequent issues of this Recommendation will augment the list of objects classes, attribute types, and operations in order to further accommodate other functional areas.

Annex A contains an index of managed object classes, packages, attributes, notifications and actions defined in this Recommendation.

There are several different viewpoints of management information which may be defined for management purposes, with the Network Element level viewpoint, the Network level viewpoint and the Service level viewpoint defined below. These viewpoints are not restrictive but define the levels of abstraction of particular types of interfaces. That is, object class definitions are not forced into this categorization but are constructed to meet the needs of exchanging management information across TMN interfaces. Objects defined for a given viewpoint may be used in others, and any object may be used by any interface which requires it. The definition of viewpoint is a means of generating requirements, hence there is no implicit definition of interfaces or storage requirements. This information is defined for the purpose of management via an open interface.

The Network Element level viewpoint is concerned with the information that is required to manage a Network Element (NE). This refers to the information required to manage the Network Element Function (NEF) and the physical aspects of the NE. The information may be derived from open systems other than the NE.

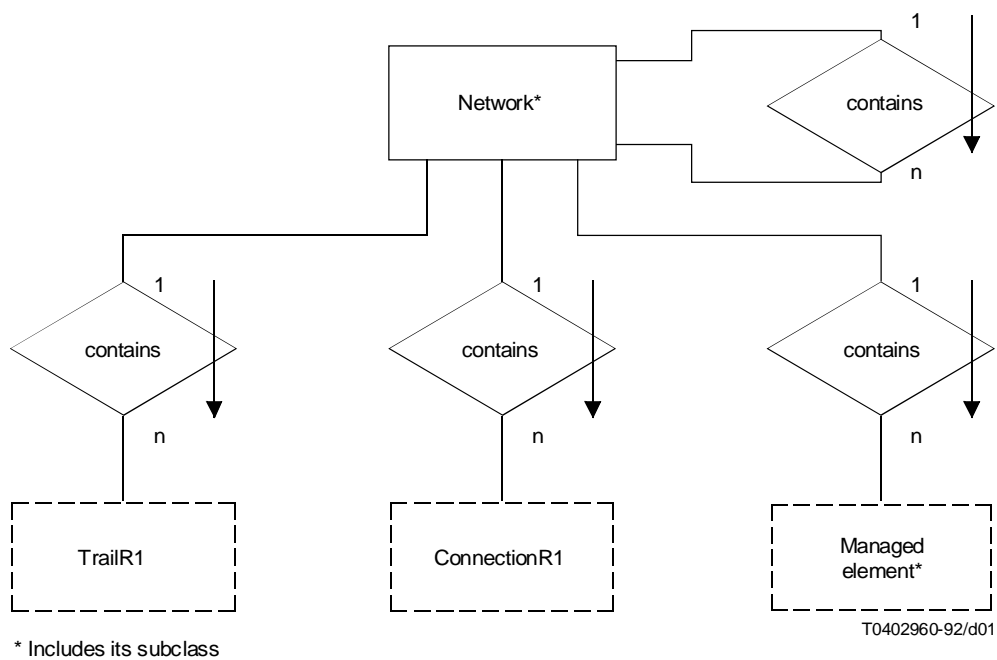
The Network level viewpoint is concerned with the information representing the network, both physically and logically. It is concerned with how network element entities are related, topographically interconnected, and configured to provide and maintain end-to-end connectivity.

The Service level viewpoint is concerned with how Network level aspects (such as an end-to-end path) are utilized to provide a network service, and as such is concerned with the requirements of a network service (e.g. availability, cost, etc.) and how these requirements are met through the use of the network, and all related customer information.

### **3 Object classes**

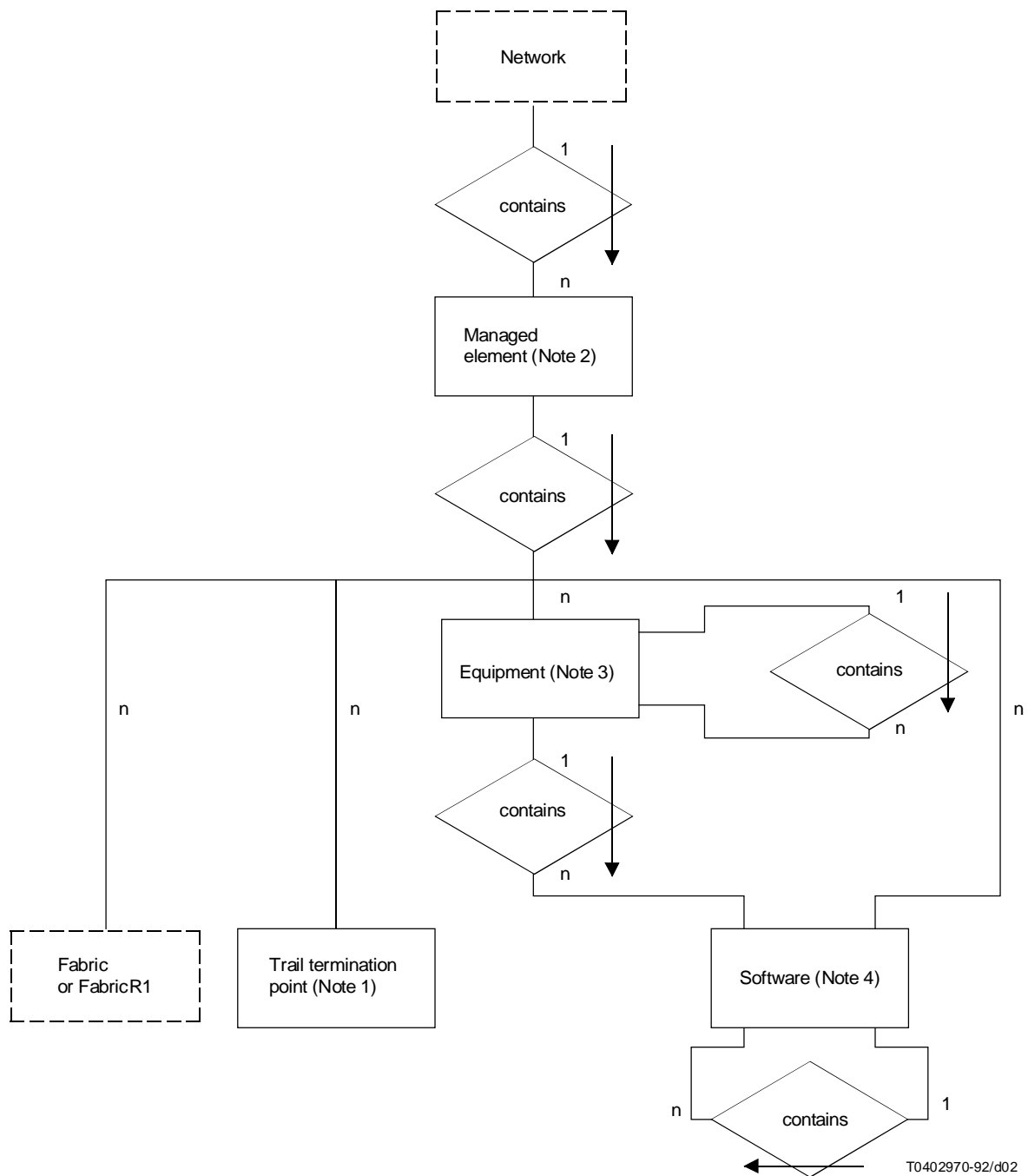
This clause contains the definitions of the object classes that form the basis for the generic network information model. These object classes are grouped into 6 fragments and they are depicted in Figures 1 to 6. These fragments show all related object classes from different perspectives. Additional fragments and object classes for each fragment are for further study. The inheritance hierarchy of this model is presented in Figure 7.

The purpose of defining fragments is only to have a document that is easier to read by grouping a limited number of object class definitions. Each fragment deals with a particular subject (e.g. network, managed element, transmission, support objects) but object classes of each fragment will be usable in various models depending on the functional area managed and/or on the level viewpoint considered.



NOTE – Object classes contained in dotted boxes may be found in other views.

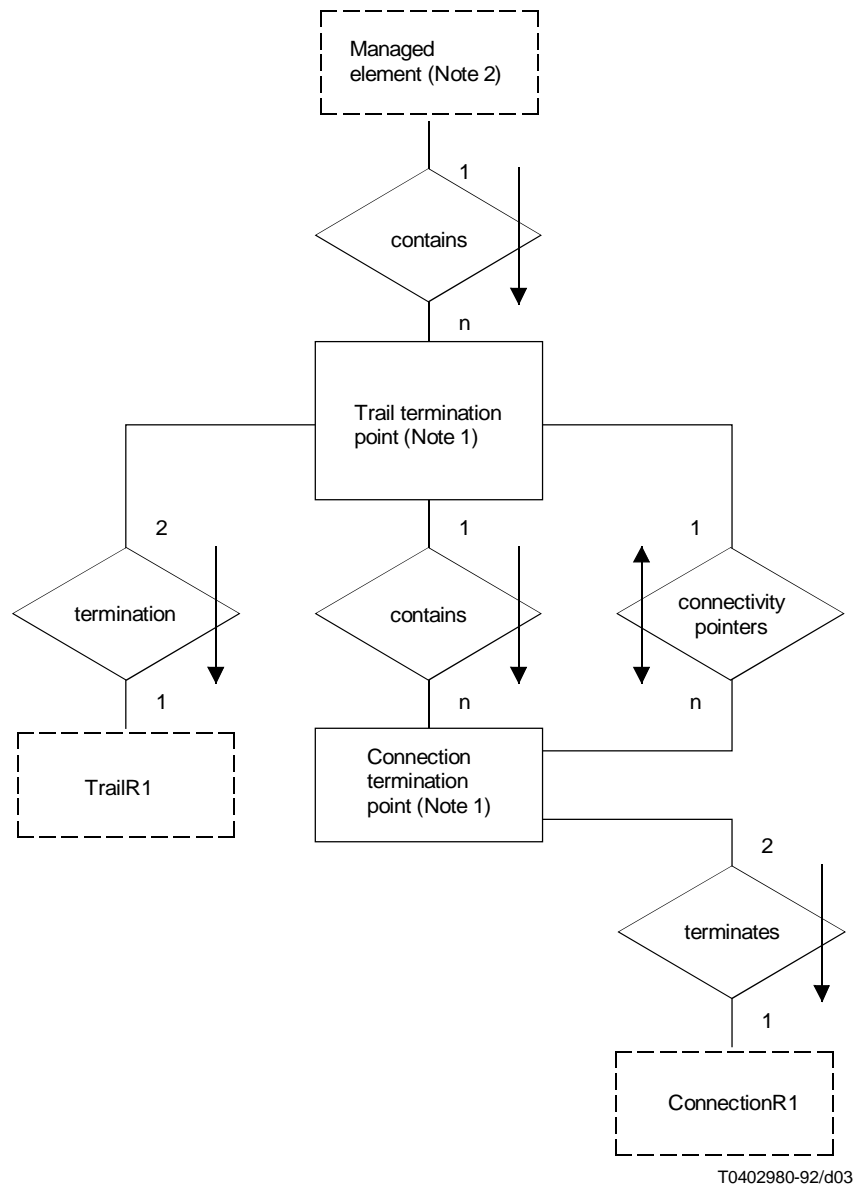
FIGURE 1/M.3100  
Entity-Relationship depiction of the network fragment



#### NOTES

- 1 Represents source, sink and bidirectional object classes.
- 2 Includes the subclass managedElementR1.
- 3 Includes the subclasses equipment R1, equipmentHolder and circuitPack.
- 4 Includes the subclass software R1.
- 5 Object classes contained in dotted boxes may be found in other views.

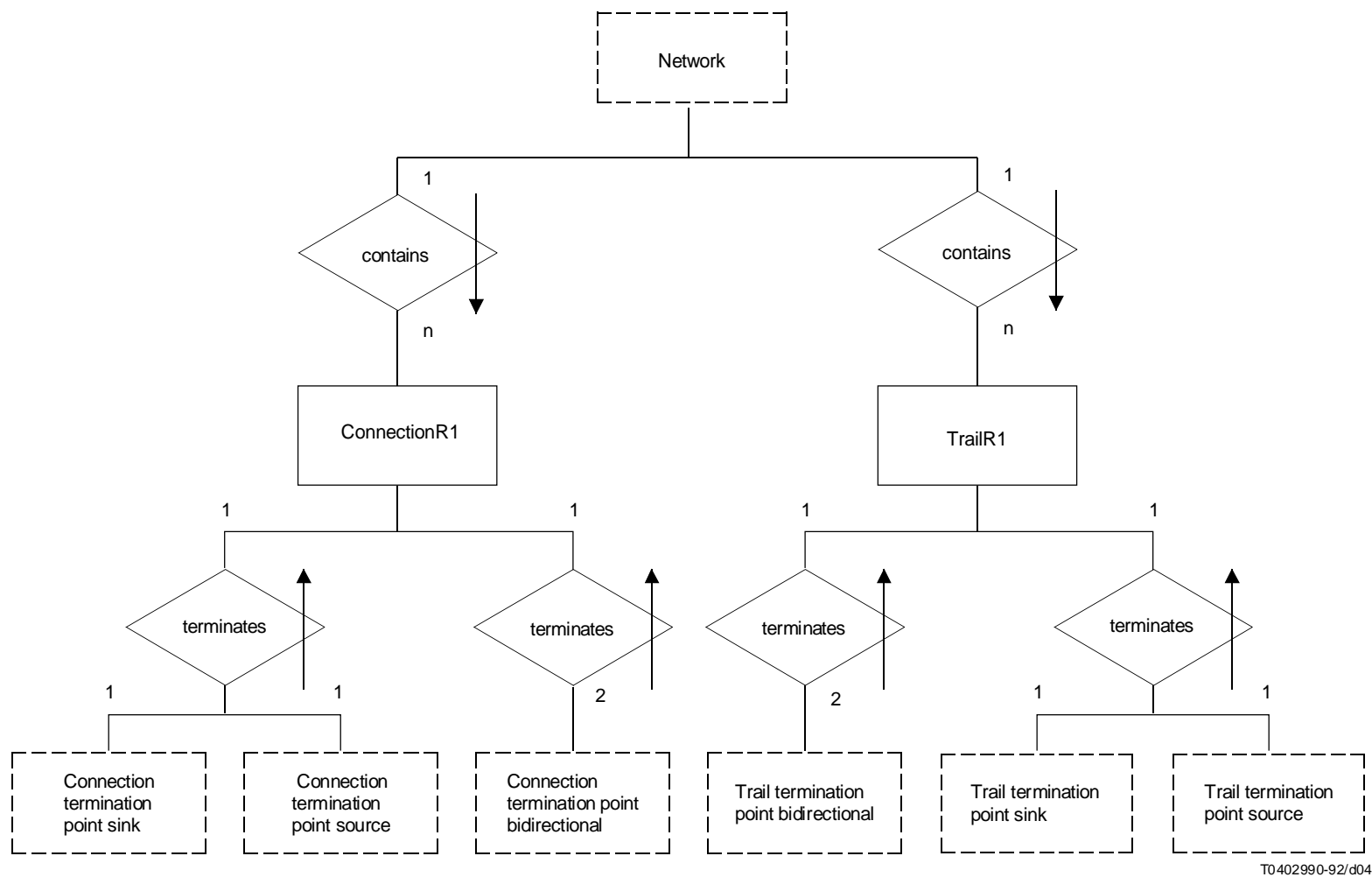
FIGURE 2/M.3100  
Entity-Relationship depiction of the managed element fragment



#### NOTES

- 1 Represents source, sink and bidirectional classes.
- 2 Includes the subclass managedElementR1.
- 3 Object classes contained in dotted boxes may be found in other views.

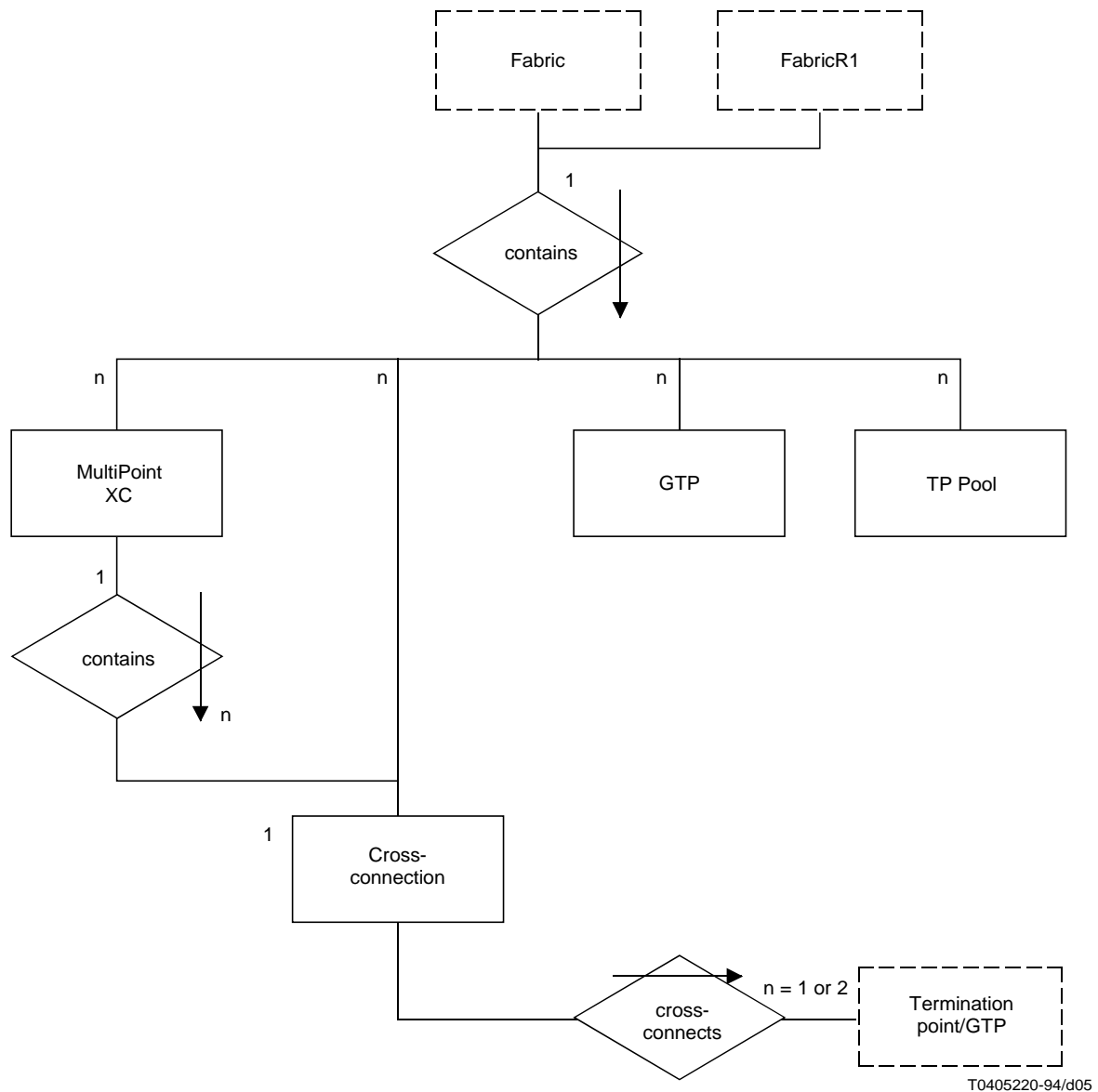
FIGURE 3/M.3100  
Entity-Relationship depiction of termination point fragment



T0402990-92/d04

NOTE – Object classes contained in dotted boxes may be found in other views.

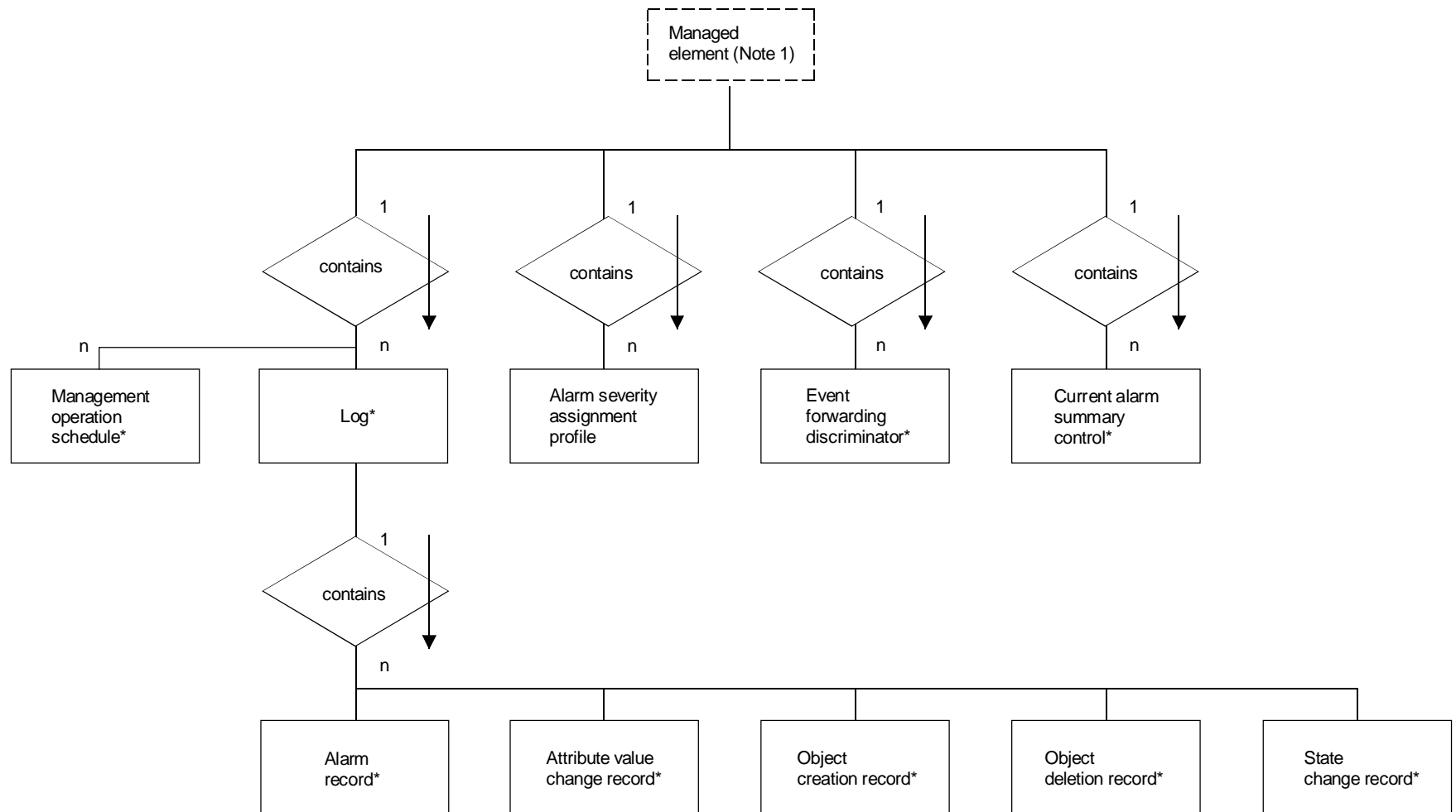
FIGURE 4/M.3100  
Entity-Relationship depiction of transmission view



NOTE – Object classes (termination point subclasses of GTP) contained in dotted boxes may be found in other views.

FIGURE 5/M.3100  
Entity-Relationship depiction of cross-connection fragment



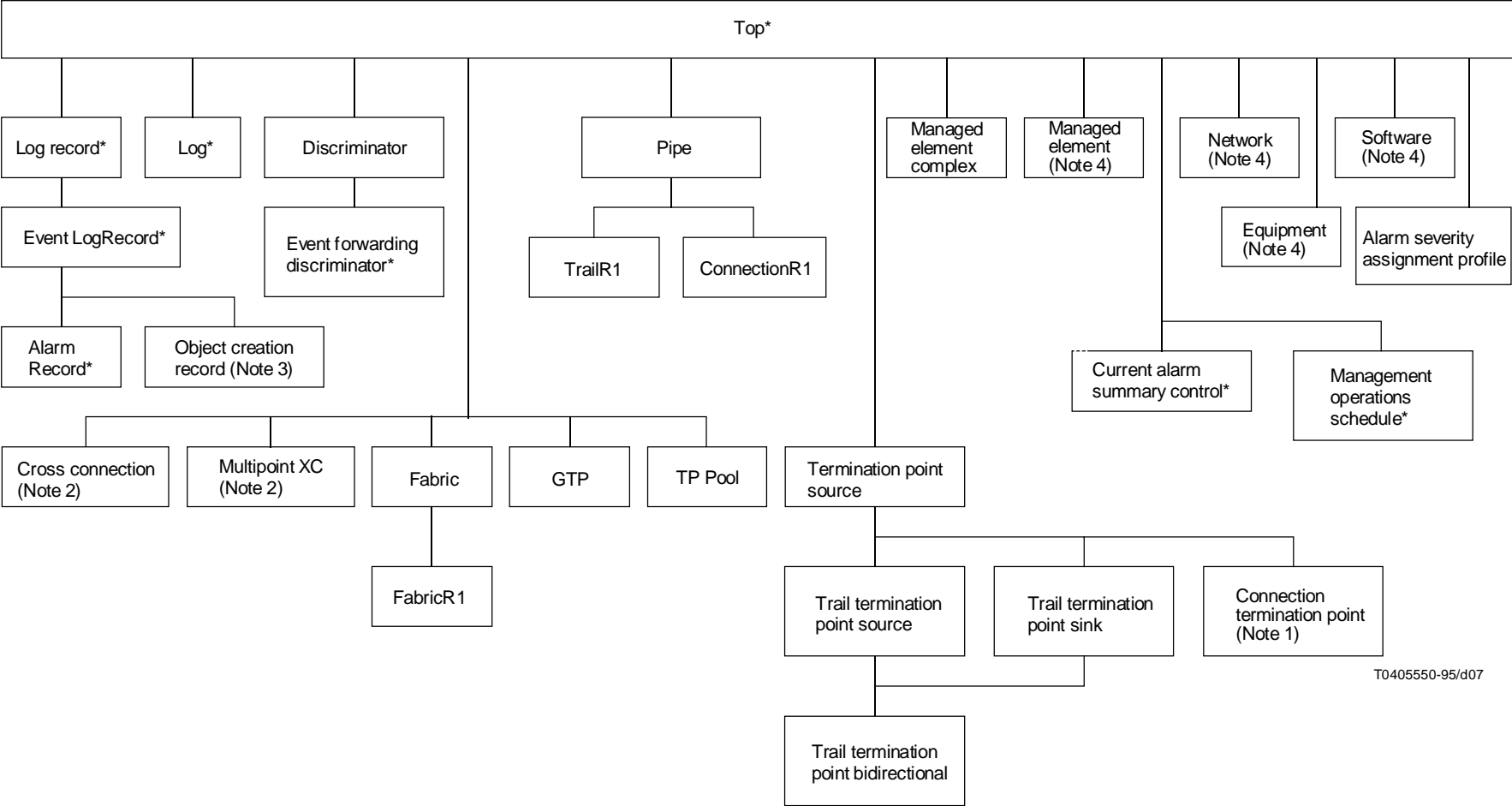


T0403010-92/d06

#### NOTES

- 1 Includes subclass managedElementR 1.
- 2 Object classes contained in dotted boxes may be found in other views.
- 3 Object classes denoted by \* are defined in Recommendations X.721 or Q.821 and referenced in this Recommendation.

FIGURE 6/M.3100  
Entity-Relationship depiction of the functional area fragment



T0405550-95/d07

NOTES

- 1 Represents source, sink and bidirectional object classes as in trail TP.
- 2 Subclasses called named cross-connection not shown.
- 3 Other records not shown.
- 4 Subclasses not shown.
- 5 Object classes denoted by \* are defined in Recommendations X.721 or Q.821 and referenced in this Recommendation.

FIGURE 7/M.3100  
Inheritance hierarchy

Table 1 lists the object classes defined or referred to in this Recommendation.

TABLE 1/M.3100  
**Managed object classes**

Object Class
Alarm Record*
Alarm Severity Assignment Profile
Attribute Value Change Record*
Circuit Pack
Circuit Subgroup
Connection <sup>+</sup>
ConnectionR1
Connection Termination Point Bidirectional
Connection Termination Point Sink
Connection Termination Point Source
Connectivity <sup>+</sup>
Cross-Connection
Current Alarm Summary Control*
Discriminator*
Equipment
Equipment Holder
EquipmentR1
Event Forwarding Discriminator*
Event Log Record*
Fabric
FabricR1
Group Termination Point
Log*
Log Record*
Managed Element
Managed ElementR1
Managed Element Complex
Management Operations Scheduler*
Multipoint Cross-Connection
Named Cross-Connection
Named Multipoint Cross-ConnectionNetwork
Network
Network R1
Object Creation Record*
Object Delection Record*
Pipe
Software
SoftwareR1
State Change Record*
Termination Point
TP Pool
Trail <sup>+</sup>
TrailR1
Trail Termination Point Bidirectional
Trail Termination Point Sink
Trail Termination Point Source
NOTE – Object classes denoted by * are defined in other Recommendations and in this Recommendation. Object classes denoted by + were included in the 1992 Recommendation and are now considered obsolete/deprecated (included in Appendix II).

### 3.1 Network fragment

Managed object classes in network fragment are presented in Figure 1. The definition(s) of the managed object class(es) are specified as follows.

#### 3.1.1 Network

```
network MANAGED OBJECT CLASS
  DERIVED FROM      "Recommendation X.721: 1992":top;
  CHARACTERIZED BY
    networkPackage PACKAGE
      BEHAVIOUR
        networkDefinition;
      ATTRIBUTES
        networkId GET;;;
  CONDITIONAL PACKAGES
    userLabelPackage PRESENT IF "an instance supports it";

REGISTERED AS {m3100ObjectClass 1};
```

```
networkDefinition BEHAVIOUR
  DEFINED AS
```

“The Network object class is a class of managed objects that are collections of interconnected telecommunications and management objects (logical or physical) capable of exchanging information. These objects have one or more common characteristics, for example they may be owned by a single customer or provider, or associated with a specific service network. A network may be nested within another (larger) network, thereby forming a containment relationship. An example of a network that is contained in another network is a transmission sub-network. It is owned by a single Administration and can only perform transmission functions.”;

#### 3.1.2 NetworkR1

```
networkR1 MANAGED OBJECT CLASS
  DERIVED FROM      network;
  CHARACTERIZED BY
    networkR1Package PACKAGE
      ATTRIBUTES
        "Recommendation X.721:1992":systemTitle GET-REPLACE;;;

REGISTERED AS {m3100ObjectClass 33};
```

### 3.2 Managed element fragment

Managed object classes in managed element fragment are presented in Figure 2. The definition(s) of the managed object class(es) are specified as follows.

#### 3.2.1 Circuit Pack

```
circuitPack MANAGED OBJECT CLASS
  DERIVED FROM      equipmentR1;
  CHARACTERIZED BY
    createDeleteNotificationsPackage,
    administrativeOperationalStatesPackage,
    stateChangeNotificationPackage,
    equipmentsEquipmentAlarmR1Package,
    currentProblemListPackage,
    equipmentAlarmEffectOnServicePackage,
    alarmSeverityAssignmentPointerPackage,
    circuitPackPackage PACKAGE
      BEHAVIOUR circuitPackBehaviour;
      ATTRIBUTES
        circuitPackType GET SET-BY-CREATE,
        "Recommendation X.721: 1992": availabilityStatus
        PERMITTED VALUES ASN.CONTDefinedTypesModule.CircuitPackAvailabilityStatus
        GET;;;

REGISTERED AS {m3100ObjectClass 30};
```

**circuitPackBehaviour    BEHAVIOUR  
DEFINED AS**

“The Circuit Pack object class is a class of managed objects that represents a plug-in replaceable unit that can be inserted into or removed from the equipment holder of the Network Element. Examples of plug-in cards include line cards, processors and power supply units.

The attribute availability status is used to indicate whether the correct physical circuit pack is inserted or not. This is a set valued attribute and includes the value notInstalled or empty. If the type of the inserted physical circuit pack matches the value of the circuitPackType attribute (relating to the circuitPack instance) then the value of the availabilityStatus is an empty set. Otherwise, the value of the availabilityStatus attribute is notInstalled even if it is one of the acceptable circuit pack type.”;

**3.2.2    Equipment**

**equipment MANAGED OBJECT CLASS  
DERIVED FROM            "Recommendation X.721: 1992":top;  
CHARACTERIZED BY  
equipmentPackage PACKAGE  
BEHAVIOUR  
    equipmentBehaviour BEHAVIOUR  
    DEFINED AS**

“The equipment object class is a class of managed objects that represents physical components of a managed element, including replaceable components. An instance of this object class is present in a single geographic location. An equipment may be nested within another equipment, thereby creating a containment relationship. The equipment type shall be identified by sub-classing this object class. Either the name of the sub-class or an attribute may be used for identifying the equipment type.

When the attribute value change notification package is present, the attributeValueChange notification defined in Recommendation X.721 shall be emitted when the value of one of the following attribute changes: alarm status, affected object list, user label, version, location name and current problem list. Because the above attributes are all in conditional packages, the behaviour for emitting the attribute value change notification applies only when the corresponding conditional packages are present in the managed object. When the state change notification package is present, the stateChangeNotification defined in Recommendation X.721 shall be emitted if the value of administrative state or operational state changes (when the administrativeOperationalStates conditional package is present).” ;;

**ATTRIBUTES**  
    equipmentId GET SET-BY-CREATE,  
    replaceable GET SET-BY-CREATE;  
;;

**CONDITIONAL PACKAGES**

createDeleteNotificationsPackage PRESENT IF "the objectCreation and objectDeletion notifications defined in Recommendation X.721 are supported by an instance of this class.",  
attributeValueChangeNotificationPackage PRESENT IF "the attributeValueChange notification defined in Recommendation X.721 is supported by an instance of this class.",  
stateChangeNotificationPackage PRESENT IF "the stateChange notification defined in Recommendation X.721 is supported by an instance of this class.",  
administrativeOperationalStatesPackage PRESENT IF "an instance supports it.",  
affectedObjectListPackage PRESENT IF "an instance supports it.",  
equipmentsEquipmentAlarmPackage PRESENT IF "the equipmentAlarm notification defined in Recommendation X.721 is supported by an instance of this class.",  
environmentalAlarmPackage PRESENT IF "the environmentalAlarm notification defined in Recommendation X.721 is supported by an instance of this class.",  
tmnCommunicationsAlarmInformationPackage PRESENT IF "the communicationsAlarm notification defined in Recommendation X.721 is supported by an instance of this class.",  
processingErrorAlarmPackage PRESENT IF "the processingErrorAlarm notification defined in Recommendation X.721 is supported by an instance of this class.",  
userLabelPackage PRESENT IF "an instance supports it",  
vendorNamePackage PRESENT IF "an instance supports it",  
versionPackage PRESENT IF "an instance supports it",  
locationNamePackage PRESENT IF "an instance supports it",  
currentProblemListPackage PRESENT IF "an instance supports it";

**REGISTERED AS {m3100ObjectClass 2};**

### 3.2.3 Equipment Holder

**equipmentHolder** MANAGED OBJECT CLASS

**DERIVED FROM** equipmentR1;

**CHARACTERIZED BY**

**equipmentHolderPackage** PACKAGE

**BEHAVIOUR** equipmentHolderBehaviour;

**ATTRIBUTES**

equipmentHolderType GET SET-BY-CREATE,

equipmentHolderAddress GET SET-BY-CREATE ;;;

**CONDITIONAL PACKAGES**

subordinateCircuitPackPackage

**PRESENT IF** "the resource represented by this equipmentHolder instance is allowed to contain a circuit pack";

**REGISTERED AS** {m3100ObjectClass 32};

**equipmentHolderBehaviour** **BEHAVIOUR**

**DEFINED AS**

"The Equipment Holder object class is a class of managed objects that represents physical resources of a network element that are capable of holding other physical resources. Examples of resources represented by instances of this object class are equipment bay, shelf and slot.

The conditional package subordinateCircuitPackPackage consists of three attributes:

- acceptableCircuitPackTypeList

This attribute specifies the types of circuit packs that are acceptable by the equipment holder. Values may be added, replaced or removed to this set-valued attribute. If the equipmentHolder is currently containing a circuitPack, then the value of corresponding type (of the circuitPack) shall not be replaced or removed from this attribute. The type of the circuitPack contained shall be one of the types specified for this attribute.

- holderStatus

This attribute indicates the status of the equipment holder. The status of the holder may be one of the following.

- empty to indicate that there is no replaceable unit in the holder.
- the holder contains a unit that is one of the types in the acceptableCircuitPackType list.
- the holder contains a unit recognizable by the network element; but not one of the types in the acceptableCircuitPackTypeList.
- unrecognized replaceable unit.

if the holder contains a unit that is acceptable and its type matches the value of the circuitPackType attribute(of the circuitPack object), then the availableStatus of the circuitPack will be an empty set. In all other cases the availabilityStatus will include a notInstalled value.

- subordinateCircuitPackSoftwareLoad

This attribute specifies the software load, if there is any, which is currently designated as the one to be loaded to the subordinate(contained) circuitPack (if it is software loadable) whenever an automatic reload of software is needed.”;

### 3.2.4 EquipmentR1

**equipmentR1** MANAGED OBJECT CLASS

**DERIVED FROM** equipment;

**CHARACTERIZED BY**

**equipmentR1Package** PACKAGE

**ATTRIBUTES**

serialNumber GET,

supportedByObjectList GET-REPLACE ADD-REMOVE;;;

## CONDITIONAL PACKAGES

alarmSeverityAssignmentPointerPackage PRESENT IF "the managed object supports configuration of alarm severities",  
equipmentAlarmR1Package PRESENT IF "the equipmentAlarm notification defined in Recommendation X.721 is supported by an instance of this class.",  
environmentalAlarmR1Package PRESENT IF "the environmentalAlarm notification defined in Recommendation X.721 is supported by an instance of this class.",  
processingErrorAlarmR1Package PRESENT IF "the processingErrorAlarm notification defined in Recommendation X.721 is supported by an instance of this class.";

REGISTERED AS {m3100ObjectClass 28};

### 3.2.5 Managed Element

**managedElement** MANAGED OBJECT CLASS

DERIVED FROM "Recommendation X.721: 1992":top;

CHARACTERIZED BY

managedElementPackage PACKAGE

BEHAVIOUR

managedElementBehaviour BEHAVIOUR

DEFINED AS

"The Managed Element object class is a class of managed objects representing telecommunications equipment or TMN entities (either groups or parts) within the telecommunications network that performs managed element functions, i.e. provides support and/or service to the subscriber. Managed elements may or may not additionally perform mediation/OS functions. A managed element communicates with the manager (directly or indirectly) over one or more standard Q-interfaces for the purpose of being monitored and/or controlled. A managed element contains equipment that may or may not be geographically distributed.

When the attribute value change notification package is present, the attributeValueChange notification defined in Recommendation X.721 shall be emitted when the value of one of the following attribute changes: alarm status, user label, version, location name and current problem list. For the above attributes that are in conditional packages, the behaviour for emitting the attribute value change notification applies only when the corresponding conditional packages are present in the managed object. When the state change notification package is present, the stateChangeNotification defined in Recommendation X.721 shall be emitted if the value of administrative state or operational state or usage state changes."

;;

#### ATTRIBUTES

managedElementId GET,

"Recommendation X.721: 1992":systemTitle GET-REPLACE,

alarmStatus GET,

"Recommendation X.721: 1992":administrativeState GET-REPLACE,

"Recommendation X.721: 1992":operationalState GET,

"Recommendation X.721: 1992":usageState GET;

#### NOTIFICATIONS

"Recommendation X.721: 1992":environmentalAlarm,

"Recommendation X.721: 1992":equipmentAlarm,

"Recommendation X.721: 1992":communicationsAlarm,

"Recommendation X.721: 1992":processingErrorAlarm;;;

## CONDITIONAL PACKAGES

createDeleteNotificationsPackage PRESENT IF "the objectCreation and objectDeletion notifications defined in Recommendation X.721 are supported by an instance of this class.",

attributeValueChangeNotificationPackage PRESENT IF "the attributeValueChange notification defined in Recommendation X.721 is supported by an instance of this class.",

stateChangeNotificationPackage PRESENT IF "the stateChange notification defined in Recommendation X.721 is supported by an instance of this class.",

audibleVisualLocalAlarmPackage PRESENT IF "an instance supports it",

resetAudibleAlarmPackage PRESENT IF "an instance supports it",

userLabelPackage PRESENT IF "an instance supports it",

vendorNamePackage PRESENT IF "an instance supports it",

versionPackage PRESENT IF "an instance supports it",

locationNamePackage PRESENT IF "an instance supports it",

currentProblemListPackage PRESENT IF "an instance supports it",

externalTimePackage PRESENT IF "an instance supports it",

systemTimingSourcePackage PRESENT IF "an instance supports it";

REGISTERED AS {m3100ObjectClass 3};

### 3.2.6 Managed ElementR1

```
managedElementR1 MANAGED OBJECT CLASS
DERIVED FROM      managedElement;
CHARACTERIZED BY
managedElementR1Package PACKAGE
NOTIFICATIONS
"Recommendation X.721: 1992":environmentalAlarm
  "Recommendation Q.821:1992":logRecordIdParameter
  "Recommendation Q.821:1992":correlatedRecordNameParameter
  "Recommendation Q.821:1992":suspectObjectListParameter,
"Recommendation X.721: 1992":equipmentAlarm
  "Recommendation Q.821:1992":logRecordIdParameter
  "Recommendation Q.821:1992":correlatedRecordNameParameter
  "Recommendation Q.821:1992":suspectObjectListParameter,
"Recommendation X.721: 1992":communicationsAlarm
  "Recommendation Q.821:1992":logRecordIdParameter
  "Recommendation Q.821:1992":correlatedRecordNameParameter
  "Recommendation Q.821:1992":suspectObjectListParameter,
"Recommendation X.721: 1992":processingErrorAlarm
  "Recommendation Q.821:1992":logRecordIdParameter
  "Recommendation Q.821:1992":correlatedRecordNameParameter
  "Recommendation Q.821:1992":suspectObjectListParameter;;;
CONDITIONAL PACKAGES
  alarmSeverityAssignmentPointerPackage PRESENT IF
    "the managed object supports configuration of alarm severities";
REGISTERED AS {m3100ObjectClass 27};
```

### 3.2.7 Managed Element Complex

```
managedElementComplex MANAGED OBJECT CLASS
DERIVED FROM      "Recommendation X.721: 1992":top;
CHARACTERIZED BY
managedElementComplexPackage PACKAGE
BEHAVIOUR
  managedElementComplexBehaviour BEHAVIOUR
DEFINED AS

  "The Managed Element Complex object class is a class of managed objects that represents a collection of
  network elements. An OS can reference and manage one or more NEs belonging to the complex represented
  by an instance of this object class. "
  ;
ATTRIBUTES
  managedElementComplexId GET,
  "Recommendation X.721: 1992":systemTitle GET-REPLACE;;;
CONDITIONAL PACKAGES
  createDeleteNotificationsPackage PRESENT IF "the objectCreation and objectDeletion
  notifications defined in Recommendation X.721 are supported by an instance of this class.";
REGISTERED AS {m3100ObjectClass 34};
```

### 3.2.8 Software

```
software MANAGED OBJECT CLASS
DERIVED FROM      "Recommendation X.721: 1992":top;
CHARACTERIZED BY
softwarePackage PACKAGE
BEHAVIOUR
  softwareBehaviour BEHAVIOUR
DEFINED AS

  "The Software object class is a class of managed objects that represent logical information stored in
  equipment, including programs and data tables. Software may be nested within other software, thereby
  creating a containment relationship.
```



When the attribute value change notification package is present, the attributeValueChange notification defined in Recommendation X.721 shall be emitted when the value of one of the following attribute changes: alarm status, affected object list, user label, version, and current problem list. Because the above attributes are all in conditional packages, the behaviour for emitting the attribute value change notification applies only when the corresponding conditional packages are present in the managed object. When the state change notification package is present, the stateChangeNotification defined in Recommendation X.721 shall be emitted if the value of administrative state or operational state changes (when the administrativeOperationalStates conditional package is present)."

;;

#### **ATTRIBUTES**

softwareId GET SET-BY-CREATE;

;;

#### **CONDITIONAL PACKAGES**

createDeleteNotificationsPackage PRESENT IF "the objectCreation and objectDeletion notifications defined in Recommendation X.721 are supported by an instance of this class.",  
attributeValueChangeNotificationPackage PRESENT IF "the attributeValueChange notification defined in Recommendation X.721 is supported by an instance of this class.",  
stateChangeNotificationPackage PRESENT IF "the stateChange notification defined in Recommendation X.721 is supported by an instance of this class.",  
administrativeOperationalStatesPackage PRESENT IF "an instance supports it",  
affectedObjectListPackage PRESENT IF "an instance supports it",  
softwareProcessingErrorAlarmPackage PRESENT IF "an instance supports it",  
userLabelPackage PRESENT IF "an instance supports it",  
vendorNamePackage PRESENT IF "an instance supports it",  
versionPackage PRESENT IF "an instance supports it",  
currentProblemListPackage PRESENT IF "an instance supports it";

REGISTERED AS {m3100ObjectClass 4};

### **3.2.9 SoftwareR1**

#### **softwareR1 MANAGED OBJECT CLASS**

DERIVED FROM software;

#### **CONDITIONAL PACKAGES**

alarmSeverityAssignmentPointerPackage PRESENT IF  
"the managed object supports configuration of alarm severities",  
softwareProcessingErrorAlarmR1Package PRESENT IF "an instance supports it";

REGISTERED AS {m3100ObjectClass 29};

### **3.3 Termination point fragment**

Managed object classes in termination point fragment are presented in Figure 3. The behaviour definition(s) of the managed object class(es) are specified as follows.

#### **3.3.1 Connection Termination Point Bidirectional**

The Connection Termination Point Bidirectional object class is a class of managed objects that originates a link connection and terminates a link connection. Technology-specific connection termination bidirectional subclasses are derived by multiple inheritance from this object class and corresponding technology-specific source and sink object classes.

#### **connectionTerminationPointBidirectional MANAGED OBJECT CLASS**

DERIVED FROM

connectionTerminationPointSource,  
connectionTerminationPointSink;

REGISTERED AS {m3100ObjectClass 5};

### 3.3.2 Connection Termination Point Sink

The Connection Termination Point Sink object class is a class of managed objects that terminates a link connection.

**connectionTerminationPointSink** MANAGED OBJECT CLASS

**DERIVED FROM** terminationPoint;

**CHARACTERIZED BY**

**connectionTerminationPointSinkPackage** PACKAGE

**BEHAVIOUR**

**connectionTerminationPointSinkBehaviour** BEHAVIOUR

**DEFINED AS**

“This managed object terminates a link connection. The downstream connectivity pointer attribute points to the termination point managed object, within the same managed element, that receives information (traffic) from this termination point at the same layer, or is null. The referenced object shall be an instance of one of the following classes or its subclasses: Trail Termination Point Sink, Trail Termination Point Bidirectional, Connection Termination Point Source, Connection Termination Point Bidirectional. The downstream connectivity pointer may identify one or more objects depending on whether the signal is connected to one or more termination point objects.”

;;

**ATTRIBUTES**

**downstreamConnectivityPointer** PERMITTED VALUES

-- The allowed choices for the syntax of this attribute are restricted in the subtype

-- CTPDownstreamPointer

ASN.CONTDefinedTypesModule.CTPDownstreamPointer GET SET-BY-CREATE;;;

**CONDITIONAL PACKAGES**

**ctpInstancePackage** PRESENT IF "the name binding used to create an instance of this object class requires this attribute.",

**channelNumberPackage** PRESENT IF "an instance supports it";

**REGISTERED AS** {m3100ObjectClass 6};

### 3.3.3 Connection Termination Point Source

The Connection Termination Point Source object class is a class of managed objects that originates a link connection.

**connectionTerminationPointSource** MANAGED OBJECT CLASS

**DERIVED FROM** terminationPoint;

**CHARACTERIZED BY**

**connectionTerminationPointSourcePackage** PACKAGE

**BEHAVIOUR**

**connectionTerminationPointSourceBehaviour** BEHAVIOUR

**DEFINED AS**

“This managed object originates a link connection. The upstream connectivity pointer attribute points to the termination point managed object, within the same managed element, that sends information (traffic) to this termination point at the same layer, or is null. The referenced object shall be an instance of one of the following classes or its subclasses: Trail Termination Point Source, Trail Termination Point Bidirectional, Connection Termination Point Sink, Connection Termination Point Bidirectional.”

;;

**ATTRIBUTES**

**upstreamConnectivityPointer** PERMITTED VALUES

-- The allowed choices for the syntax of this attribute are restricted in the subtype

-- CTPUpstreamPointer

ASN.CONTDefinedTypesModule.CTPUpstreamPointer GET SET-BY-CREATE

;;;

**CONDITIONAL PACKAGES**

**ctpInstancePackage** PRESENT IF "the name binding used to create an instance of this object class requires this attribute.",

**channelNumberPackage** PRESENT IF "an instance supports it";

**REGISTERED AS** {m3100ObjectClass 7};

### 3.3.4 Termination Point

The Termination Point object class is a class of managed objects that terminates transport entities, such as trails and connections. This object class is a basic object class from which subclasses, such as trail termination and connection termination point are derived. The use of operational state is further refined in subclasses of this class. This is an uninstantiable managed object class.

#### **terminationPoint MANAGED OBJECT CLASS**

**DERIVED FROM** "Recommendation X.721: 1992":top;

**CHARACTERIZED BY**

**terminationPointPackage PACKAGE**

**BEHAVIOUR**

**terminationPointBehaviour BEHAVIOUR**

**DEFINED AS**

"This managed object represents the termination of a transport entity, such as a trail or a connection. The characteristic information attribute is used to identify equivalence between subclasses of termination points in order to determine whether cross connection or connectivity is possible. The operational state reflects the perceived ability to generate and/or receive a valid signal. Subclasses of termination point shall specify the attributes and states for which attribute value change and state change notifications will be generated."

;;

**ATTRIBUTES**

**supportedByObjectList GET;;**

**CONDITIONAL PACKAGES**

**createDeleteNotificationsPackage PRESENT IF**

"the objectCreation and objectDeletion notifications defined in Recommendation X.721 are supported by an instance of this managed object class",

**attributeValueChangeNotificationPackage PRESENT IF**

"the attributeValueChange notification defined in Recommendation X.721 is supported by an instance of this managed object class",

**stateChangeNotificationPackage PRESENT IF**

"the stateChange notification defined in Recommendation X.721 is supported by an instance of this managed object class",

**operationalStatePackage PRESENT IF**

"the resource represented by this managed object is capable of assessing the ability to generate and/or receive a valid signal.",

**crossConnectionPointerPackage PRESENT IF**

"the termination point can be flexibly assigned, (i.e. cross connected).",

**characteristicInformationPackage PRESENT IF**

"an instance supports it.",

**networkLevelPackage PRESENT IF "an instance supports it",**

**tmnCommunicationsAlarmInformationPackage PRESENT IF**

"the communicationsAlarm notification (as defined in Recommendation X.721) is supported by this managed object",

**alarmSeverityAssignmentPointerPackage PRESENT IF**

"the tmnCommunicationsAlarmInformationPackage package is present AND the managed object supports configuration of alarm severities";

**REGISTERED AS** {m3100ObjectClass 8};

### 3.3.5 Trail Termination Point Bidirectional

The Trail Termination Point Bidirectional object class is a class of managed objects representing a termination point where one trail is originated and another trail is terminated. It represents the access point in a layer network which is a focus for both the trail relationship and the client/server relationship. These concepts are defined in Recommendation G.803 [6]. Subclasses of this generic super-class include, in addition, the overhead interpretation aspects of trail termination function described in G.803 [6]. Technology-specific (e.g. PDH, SDH) trail termination bidirectional object classes may be defined directly as subclasses of this class and corresponding source and sink object classes using multiple inheritance.

#### **trailTerminationPointBidirectional MANAGED OBJECT CLASS**

**DERIVED FROM** trailTerminationPointSource, trailTerminationPointSink;

**CHARACTERIZED BY**

**trailTerminationPointBidirectionalPackage PACKAGE**

**BEHAVIOUR**

**trailTerminationPointBidirectionalBehaviour BEHAVIOUR**

## **DEFINED AS**

"The operational state is disabled if either the sink or source part of the termination point is disabled."

;;;

**REGISTERED AS {m3100ObjectClass 9};**

### **3.3.6 Trail Termination Point Sink**

The Trail Termination Point Sink object class is a class of managed objects representing a termination point where a trail is terminated. It represents the access point in a layer network which is a focus for both the trail relationship and the client/server relationship. These concepts are defined in Recommendation G.803 [6]. Subclasses of this generic super-class include, in addition, the overhead interpretation aspects of trail termination function described in Recommendation G.803 [6].

#### **trailTerminationPointSink MANAGED OBJECT CLASS**

**DERIVED FROM** terminationPoint;

**CHARACTERIZED BY**

operationalStatePackage,

trailTerminationPointSinkPackage **PACKAGE**

**BEHAVIOUR**

trailTerminationPointSinkBehaviour **BEHAVIOUR**

**DEFINED AS**

"This managed object represents a termination point where a trail is terminated. It represents the access point in a layer network which is a focus for both the trail relationship and the client/server relationship.

The operational state reflects the perceived ability to receive a valid signal. If the termination point detects that a signal received has failed or it is unable to process the incoming signal, then the operational state will have the value disabled.

When the administrative state is locked, the termination point is administratively removed from service. When the administrative state is unlocked, the termination point is administratively in service. Changes to administrative state have no effect on the connectivity pointer.

A change in the operational state shall cause a state change notification. If administrative state is present in an instance of trail termination point sink class, it shall not emit a state change notification. However, subclasses of trail termination point sink class may modify this behaviour to require this notification. Subclasses of trail termination point sink shall specify the attributes for which attribute value change notifications should be generated.

The upstream connectivity pointer attribute points to the termination point managed object, within the same managed element, that sends information (traffic) to this termination point at the same layer, or is null. The referenced object shall be an instance of one of the following classes or its subclasses: Connection Termination Point Sink or Bidirectional (single or a concatenated sequence) or Trail Termination Point Source or Bidirectional."

;;

**ATTRIBUTES**

upstreamConnectivityPointer **GET SET-BY-CREATE**;;

**CONDITIONAL PACKAGES**

"Recommendation X.721:1992":administrativeStatePackage **PRESENT IF**

"the resource represented by the managed object is capable of being administratively placed in and out of service",

supportableClientListPackage **PRESENT IF**

"the object class can support more than one type of client",

ttpInstancePackage **PRESENT IF**

"the name binding used to create an instance of this object class requires this attribute.";

**REGISTERED AS {m3100ObjectClass 10};**

### 3.3.7 Trail Termination Point Source

The Trail Termination Point Source object class is a class of managed objects representing a termination point where a trail is originated. It represents the access point in a layer network which is a focus for both the trail relationship and the client/server relationship. These concepts are defined in Recommendation G.803 [6].

#### **trailTerminationPointSource MANAGED OBJECT CLASS**

**DERIVED FROM** terminationPoint;

**CHARACTERIZED BY**

operationalStatePackage,

**trailTerminationPointSourcePackage PACKAGE**

**BEHAVIOUR**

**trailTerminationPointSourceBehaviour BEHAVIOUR**

**DEFINED AS**

“This managed object represents a termination point where a trail is originated. It represents the access point in a layer network which is a focus for both the trail relationship and the client/server relationship.

The operational state reflects the perceived ability to generate a valid signal. If the termination point detects that a valid signal cannot be generated, then the operational state will have the value disabled.

When the administrative state is locked, the termination point is administratively removed from service. When the administrative state is unlocked, the termination point is administratively in service. Changes to administrative state have no effect on the connectivity pointer.

A change in the operational state shall cause a state change notification. If administrative state is present in an instance of trail termination point source class, it shall not emit a state change notification. However, subclasses of trail termination point source class may modify this behaviour to require this notification. Subclasses of trail termination point source shall specify the attributes for which attribute value change notifications should be generated.

The downstream connectivity pointer attribute points to the termination point managed object, within the same managed element, that receives information (traffic) from this termination point at the same layer, or is null. The referenced object shall be an instance of one of the following classes or its subclasses: Connection Termination Point Source or Bidirectional (single or a concatenated sequence or a set if connected to more than one connection termination point source objects) or Trail Termination Point Sink or Bidirectional (single or a set if connected to more than one trail termination point sink objects).”

;;

**ATTRIBUTES**

**downstreamConnectivityPointer** GET SET-BY-CREATE;;;

**CONDITIONAL PACKAGES**

**"Recommendation X.721: 1992":administrativeStatePackage PRESENT IF**

**"the resource represented by the managed object is capable of being administratively placed in and out of service",**

**supportableClientListPackage PRESENT IF**

**"the object class can support more than one type of client",**

**ttpInstancePackage PRESENT IF**

**"the name binding used to create an instance of this object class requires this attribute.";**

**REGISTERED AS {m3100ObjectClass 11};**

## 3.4 Switching and transmission fragment

Object classes in switching and transmission fragment are presented in Figure 4. The behaviour definition(s) of the object class(es) are specified as follows.

### 3.4.1 Circuit End Point Subgroup

**circuitEndPointSubgroup** MANAGED OBJECT CLASS

**DERIVED FROM** "Recommendation X.721 : 1992":top;

**CHARACTERIZED BY**

**circuitEndPointSubgroupPackage** PACKAGE

**BEHAVIOUR**

**circuitSubgroupBehaviour** BEHAVIOUR

**DEFINED AS**

“A set of circuit end points that directly interconnects one exchange with another, having common values for the attributes listed in this package. Note that the term exchange includes PBX where applicable.”

-- Annex A/E.410 defines circuit sub group--

;;

**ATTRIBUTES**

<b>circuitEndPointSubgroupId</b>	<b>GET,</b>
<b>numberOfCircuits</b>	<b>GET,</b>
<b>labelOfFarEndExchange</b>	<b>GET,</b>
<b>signallingCapabilities</b>	<b>GET,</b>
<b>informationTransferCapabilities</b>	<b>GET,</b>
<b>circuitDirectionality</b>	<b>GET,</b>
<b>transmissionCharacteristics</b>	<b>GET,</b>
<b>userLabel</b>	<b>GET-REPLACE;</b>

**NOTIFICATIONS**

"Recommendation X.721:1992": attributeValueChange,

"Recommendation X.721:1992": objectCreation,

"Recommendation X.721:1992": objectDeletion;

;;

**REGISTERED AS** {m3100ObjectClass 31};

### 3.4.2 ConnectionR1

**connectionR1** MANAGED OBJECT CLASS

**DERIVED FROM** pipe;

**CHARACTERIZED BY**

**connectionPackage** PACKAGE

**BEHAVIOUR**

**connectionBehaviour** BEHAVIOUR

**DEFINED AS**

“The Connection object class is a class of managed objects responsible for the transparent transfer of information between connection termination points. A connection is a component of a trail.

Several connections can be bundled into a higher rate trail. A sequence of one or more connections are linked together to form a trail. A connection may be either uni- or bi-directional.”

;;

**ATTRIBUTES**

**connectionId** GET SET-BY-CREATE ;;;

**CONDITIONAL PACKAGES**

**serverTrailListPackage** PRESENT IF "an instance supports it",

**clientTrailPackage** PRESENT IF "an instance supports it";

**REGISTERED AS** {m3100ObjectClass 23};

### 3.4.3 Pipe

#### pipe MANAGED OBJECT CLASS

**DERIVED FROM** "Recommendation X.721: 1992":top;

**CHARACTERIZED BY**

**pipePackage PACKAGE**

**BEHAVIOUR**

**pipeBehaviour BEHAVIOUR**

**DEFINED AS**

“The Pipe object class is a class of managed objects which ensures the transfer of information between two termination points. The pipe object class is not instantiable because the transfer is effected via the client-server relationship of trail and connection. Connectivity direction is determined by the directionality of the a and z termination points.

If an instance of this class is bidirectional, the a- and z-termination points shall also be bidirectional. If an instance of this class is unidirectional, the a-point shall be the source TP and the z-termination point shall be the sink TP.

The operational state indicates the capability to carry a signal.”

;;

**ATTRIBUTES**

**directionality GET,**

**"Recommendation X.721: 1992":administrativeState GET-REPLACE,**

**"Recommendation X.721: 1992":operationalState GET,**

**a-TPInstance GET SET-BY-CREATE ,**

**z-TPInstance GET SET-BY-CREATE**

;;;

**CONDITIONAL PACKAGES**

**createDeleteNotificationsPackage PRESENT IF**

**"the objectCreation and objectDeletion notifications defined in Recommendation X.721 are supported by an instance of this managed object class",**

**attributeValueChangeNotificationPackage PRESENT IF**

**"the attributeValueChange notification defined in Recommendation X.721 is supported by an instance of this managed object class",**

**stateChangeNotificationPackage PRESENT IF**

**"the stateChange notification defined in Recommendation X.721 is supported by an instance of this managed object class",**

**characteristicInformationPackage PRESENT IF**

**"an instance supports it.",**

**protectedPackage PRESENT IF**

**"an instance supports it.",**

**tmnCommunicationsAlarmInformationPackage PRESENT IF**

**"the communicationsAlarm notification (as defined in Recommendation X.721) is supported by this managed object",**

**alarmSeverityAssignmentPointerPackage PRESENT IF**

**"the tmnCommunicationsAlarmInformationPackage package is present AND the managed object supports configuration of alarm severities",**

**userLabelPackage PRESENT IF "an instance supports it";**

*-- the above package may be used for M.1400 type designations.*

**REGISTERED AS {m3100ObjectClass 24};**

### 3.4.4 TrailR1

#### trailR1 MANAGED OBJECT CLASS

**DERIVED FROM** pipe;

**CHARACTERIZED BY**

trailR1Package PACKAGE

BEHAVIOUR

trailBehaviour BEHAVIOUR

**DEFINED AS**

“Trail is a class of managed objects in layer networks which is responsible for the integrity of transfer of characteristic information from one or more other layer networks. A trail is composed of two Trail Termination Points and one or more connection and associated connection termination points.” ;;

#### ATTRIBUTES

trailId GET SET-BY-CREATE ;;;

#### CONDITIONAL PACKAGES

serverConnectionListPackage PRESENT IF "an instance supports it",

clientConnectionListPackage PRESENT IF "an instance supports it";

**REGISTERED AS** {m3100ObjectClass 25};

## 3.5 Cross-connection fragment

### 3.5.1 Cross-Connection

#### crossConnection MANAGED OBJECT CLASS

**DERIVED FROM** "Recommendation X.721: 1992":top;

**CHARACTERIZED BY**

crossConnectionPackage PACKAGE

BEHAVIOUR

crossConnectionBehaviour BEHAVIOUR

**DEFINED AS**

“A managed object of this class represents an assignment relationship between the termination point or GTP object listed in the From Termination attribute and the termination point or GTP objects listed in the To Termination attribute of this managed object.

The To Termination attribute will always be non-NULL. The From Termination attribute will only be NULL in the case of point-to-multipoint configurations. If the From Termination attribute has a value of NULL, the assignment relationship is between the termination point object or the GTP object listed in the From Termination attribute of the containing Multipoint Cross-Connection managed object and the termination point object or GTP object listed in the To Termination attribute of this managed object.

A point to point cross-connection can be established between: one of CTP Sink, CTP Bidirectional, TTP Source, TTP Bidirectional, or GTP; and one of CTP Source, CTP Bidirectional, TTP Sink, TTP Bidirectional, or GTP.

In a unidirectional cross-connection, the termination or GTP object pointed to by the From Termination and the termination point or GTP object pointed to by the To Termination attribute (in this object or the containing mpCrossConnection) are related in such a way that traffic can flow between the termination points represented by these managed objects. In a bidirectional cross-connection, information flows in both directions.

If the objects listed in the From Termination and To Termination attributes are GTPs, the nth element of the From Termination GTP is related to the nth element of the To Termination GTP (for every n).

If the fromTermination attribute has a value of NULL, the directionality attribute must have the value 'unidirectional'.

The total rate of the From Terminations must be equal to the total rate of To Terminations.

The attribute Signal Type describes the signal that is cross-connected. The termination points or GTPs that are cross-connected must have signal types that are compatible.



If an instance of this object class is contained in a multipoint cross-connection and the operational state of the containing multipoint cross-connection is 'disabled', the operational state of this object will also be 'disabled'.

The following are the definitions of the administrative state and the operational state attributes:

Administrative State:

- Unlocked: The Cross-Connection object is administratively unlocked. Traffic is allowed to pass through the connection.
- Locked: No traffic is allowed to pass through the Cross-Connection. The connectivity pointers in the cross-connected termination points is NULL.

Operational State:

- Enabled: The Cross-Connection is performing its normal function.
- Disabled: The Cross-Connection is incapable of performing its normal cross-connection function.”

;;

#### ATTRIBUTES

<b>crossConnectionId</b>	<b>GET,</b>	
<b>"Recommendation X.721 : 1992":administrativeState</b>		<b>GET-REPLACE,</b>
<b>"Recommendation X.721 : 1992":operationalState</b>	<b>GET,</b>	
<b>signalType</b>	<b>GET,</b>	
<b>fromTermination</b>	<b>GET,</b>	
<b>toTermination</b>	<b>GET,</b>	
<b>directionality</b>	<b>GET;</b>	

;;

REGISTERED AS {m3100ObjectClass 15};

### 3.5.2 Fabric

**fabric MANAGED OBJECT CLASS**

**DERIVED FROM "Recommendation X.721 : 1992":top;**

**CHARACTERIZED BY**

**fabricPackage PACKAGE**

**BEHAVIOUR**

**fabricBehaviour BEHAVIOUR**

**DEFINED AS**

“The Fabric object represents the function of managing the establishment and release of cross-connections. It also manages the assignment of termination points to TP Pools and GTPs.

Administrative State:

- Unlocked: The Fabric is allowed to perform its normal functions. ACTIONS will be accepted to setup or remove cross-connections, to rearrange TP Pools, to add/remove termination points to/from GTPs.
- Locked: The Fabric is not allowed to perform its normal functions. No ACTIONS will be accepted. No new cross-connection can be setup or removed, no TP Pool can be rearranged, and no termination points can be added/removed to/from GTPs.

Operational State:

- Enabled: When the Fabric is in the enabled operational state, it may be fully-operational or partially-operational (partially operational is indicated by the availability status attribute).
- Disabled: The Fabric is incapable of performing its normal function. For instance, the managing system will not be able to:
  - 1) setup or remove any cross-connection;
  - 2) rearrange TP Pools; and
  - 3) add/remove termination points to/from GTPs.

Availability Status:

The supported values for this attribute are:

- Degraded: The Fabric is degraded in some respect. For instance, the Fabric cannot perform the function of establishing new cross-connections while it can still accept ACTIONS to re-arrange TP Pools. The Fabric remains available for service (i.e. its operational state is enabled) while it is degraded.
- Empty SET.”

;;

#### ATTRIBUTES

<b>fabricId</b>	<b>GET SET-BY-CREATE,</b>
<b>"Recommendation X.721: 1992":administrativeState</b>	<b>GET-REPLACE,</b>
<b>"Recommendation X.721 : 1992":operationalState</b>	<b>GET,</b>
<b>"Recommendation X.721 : 1992":availabilityStatus</b>	<b>GET,</b>
<b>listOfCharacteristicInfo</b>	<b>GET SET-BY-CREATE,</b>
<b>supportedByObjectList</b>	<b>GET-REPLACE ADD-REMOVE;</b>

#### ACTIONS

**addTpsToGTP,**  
**removeTpsFromGTP,**  
**addTpsToTpPool,**  
**removeTpsFromTpPool,**  
**connect,**  
**disconnect;**

;;

**REGISTERED AS {m3100ObjectClass 16};**

### 3.5.3 FabricR1

**fabricR1 MANAGED OBJECT CLASS**

**DERIVED FROM fabric;**

**CHARACTERIZED BY**

**fabricR1Package PACKAGE**

#### ACTIONS

**switchOver;**

;;

#### CONDITIONAL PACKAGES

**createDeleteNotificationsPackage PRESENT IF "the objectCreation and objectDeletion notifications defined in Recommendation X.721 are supported by an instance of this class.";**

**REGISTERED AS {m3100ObjectClass 26};**

### 3.5.4 Group Termination Point

**gtp MANAGED OBJECT CLASS**

**DERIVED FROM "Recommendation X.721 : 1992":top;**

**CHARACTERIZED BY**

**gtpPackage PACKAGE**

#### BEHAVIOUR

**gtpBehaviour BEHAVIOUR**

**DEFINED AS**

“This object class represents a group of termination points treated as a single unit for management purposes such as cross-connections. The signalType attribute describes the composition of the GTP. When a termination point is involved in a GTP, it cannot be cross-connected independently of that GTP. ”

;;

#### ATTRIBUTES

<b>gtpId</b>	<b>GET,</b>
<b>crossConnectionObjectPointer</b>	<b>GET,</b>
<b>signalType</b>	<b>GET,</b>
<b>tpsInGtpList</b>	<b>GET;</b>

;;

**REGISTERED AS {m3100ObjectClass 17};**

### 3.5.5 Multi-point Cross-Connection

#### **mpCrossConnection MANAGED OBJECT CLASS**

**DERIVED FROM "Recommendation X.721: 1992":top;**

#### **CHARACTERIZED BY**

**mpCrossConnectionPackage PACKAGE**

#### **BEHAVIOUR**

**mpCrossConnectionBehaviour BEHAVIOUR**

#### **DEFINED AS**

“This class represents an assignment relationship between the termination point or GTP object listed in the From Termination attribute and the termination point or GTP objects listed in the To Termination attributes of the contained crossConnection managed objects.

A multipoint cross-connection can be established between one of CTP Sink, CTP Bidirectional, TTP Source, TTP Bidirectional, or GTP; and a set whose members are CTP Source, CTP Bidirectional, TTP Sink, TTP Bidirectional, or GTP.

The fromTermination attribute will always be non-NULL. The termination point or GTP object pointed to by the fromTermination attribute is related to all the termination point or GTP objects pointed to by the toTermination attribute of the contained crossConnection managed objects in such a way that traffic can flow between the termination points represented by these managed objects.

Information flows from the From Termination to the To Termination of the contained cross-connection objects.

If the objects listed in the fromTermination attribute and in the toTermination attribute of the contained crossConnection objects are GTPs, the nth element of the From Termination GTP is related to the nth element of the To Termination GTP (for every n).

The total rate of the From Terminations must be equal to the total rate of To Terminations in each contained crossConnection object.

The attribute Signal Type describes the signal that is cross-connected. The termination points or GTPs that are cross-connected must have signal types that are compatible.

The following are the definitions of the administrative state and the operational state attributes:

Administrative State:

- Unlocked: The mpCrossConnection object is administratively unlocked. It allows traffic to pass through each contained connection depending on its administrative state.
- Locked: No traffic is allowed to pass through the Cross-Connection between the cross-connected termination points. The effect of this value overrides the effect of the administrative state of each contained cross-connection.

Operational State:

The operational state of a Multipoint Cross-Connection object reflects the overall health of the cross- connection including all the Cross-Connection objects contained in the Multipoint Cross-Connection.

- Enabled: The Cross-Connection is performing its normal function. Note that some (but not all) of the cross-connection objects contained in the Multipoint Cross-Connection may be disabled.
- Disabled: The Cross-Connection is incapable of performing its normal cross-connection function. All the cross-connection objects contained in the Multipoint Cross-Connection are disabled.

Availability Status:

The supported values for this attribute are:

- In test
- Degraded: The Multipoint Cross-Connection is degraded in some respect. For instance, if one or more (but not all) Cross-Connection objects contained in the Multipoint Cross-Connection are disabled, the Multipoint Cross-Connection will be considered as degraded. The Multipoint Cross-Connection remains available for service (i.e. its operational state is enabled) while it is degraded.
- Empty SET.”

;;

**ATTRIBUTES**

<b>mpCrossConnectionId</b>	<b>GET,</b>	
<b>"Recommendation X.721 : 1992":administrativeState</b>		<b>GET-REPLACE,</b>
<b>"Recommendation X.721 : 1992":operationalState</b>	<b>GET,</b>	
<b>"Recommendation X.721 : 1992":availabilityStatus</b>	<b>GET,</b>	
<b>signalType</b>	<b>GET,</b>	
<b>fromTermination</b>	<b>GET;</b>	

;;

**REGISTERED AS {m3100ObjectClass 18};**

### **3.5.6 Named Cross-Connection**

**namedCrossConnection MANAGED OBJECT CLASS**

**DERIVED FROM crossConnection;**

**CHARACTERIZED BY**

**namedCrossConnectionPackage;**

**REGISTERED AS {m3100ObjectClass 19};**

### **3.5.7 Named Multi-Point Cross-Connection**

**namedMpCrossConnection MANAGED OBJECT CLASS**

**DERIVED FROM mpCrossConnection;**

**CHARACTERIZED BY**

**namedCrossConnectionPackage;**

**REGISTERED AS {m3100ObjectClass 20};**

### **3.5.8 TP Pool**

**tpPool MANAGED OBJECT CLASS**

**DERIVED FROM "Recommendation X.721 : 1992":top;**

**CHARACTERIZED BY**

**tpPoolPackage PACKAGE**

**BEHAVIOUR**

**tpPoolBehaviour BEHAVIOUR**

**DEFINED AS**

“The tpPool object represents a set of termination points or GTPs that are used for some management purpose, such as routing. A termination point that is a member of a GTP cannot be a member of a tpPool independent of the remainder of the GTP. ”

;;

**ATTRIBUTES**

<b>tpPoolId</b>	<b>GET,</b>
<b>tpsInTpPoolList</b>	<b>GET,</b>
<b>totalTpCount</b>	<b>GET,</b>
<b>connectedTpCount</b>	<b>GET,</b>
<b>idleTpCount</b>	<b>GET;</b>

;;

**REGISTERED AS {m3100ObjectClass 21};**

## **3.6 Functional area fragment**

Object classes in functional area fragment are presented in Figure 6. The references/definitions of the object classes are given below.

### 3.6.1 Alarm Record

The Alarm Record object class is defined in Recommendation X.721[5].

### 3.6.2 Alarm Severity Assignment Profile

**alarmSeverityAssignmentProfile** MANAGED OBJECT CLASS

**DERIVED FROM** "Recommendation X.721 : 1992":top;

**CHARACTERIZED BY**

**alarmSeverityAssignmentProfilePackage** PACKAGE

**BEHAVIOUR**

**alarmSeverityAssignmentProfileBehaviour** BEHAVIOUR

**DEFINED AS**

“The alarm severity assignment profile object class is a class of management support object that specifies the alarm severity assignment for managed objects. Instances of this object are referenced by the alarmSeverityAssignmentProfilePointer attribute in the managed objects.”

;;

**ATTRIBUTES**

**alarmSeverityAssignmentProfileId** GET SET-BY-CREATE,

**alarmSeverityAssignmentList** GET-REPLACE ADD-REMOVE ;

;;

**CONDITIONAL PACKAGES**

**objectManagementNotificationsPackage** PRESENT IF "an instance supports it";

**REGISTERED AS** {m3100ObjectClass 22};

### 3.6.3 Attribute Value Change Record

The Attribute value Change Record object class is defined in Recommendation X.721[5].

### 3.6.4 Current Alarm Summary Control

The Current Alarm Summary Control object class is defined in Recommendation Q.821[7].

### 3.6.5 Discriminator

The Discriminator object class is defined in Recommendation X.721[5].

### 3.6.6 Event Forwarding Discriminator

The Event Forwarding Discriminator object class is defined in Recommendation X.721[5].

### 3.6.7 Event Log Record

The Event Log Record object class is defined in Recommendation X.721[5].

### 3.6.8 Log

The Log object class is defined in Recommendation X.721[5].

### 3.6.9 Log Record

The Log Record object class is defined in Recommendation X.721[5].

### 3.6.10 Management Operations Schedule

The Management Operations Schedule object class is defined in Recommendation Q.821[7].

### 3.6.11 Object Creation Record

The Object Creation Record object class is defined in Recommendation X.721[5].

### 3.6.12 Object Deletion Record

The Object Deletion Record object class is defined in Recommendation X.721[5].

### 3.6.13 State Change Record

The State Change Record object class is defined in Recommendation X.721[5].

## **4 Packages**

### **4.1 Administrative Operational States**

**administrativeOperationalStatesPackage PACKAGE**

**ATTRIBUTES**

"Recommendation X.721:1992":administrativeState GET-REPLACE,  
"Recommendation X.721:1992":operationalState GET;

**REGISTERED AS {m3100Package 1};**

### **4.2 Affected Object List**

**affectedObjectListPackage PACKAGE**

**ATTRIBUTES**

affectedObjectList GET;

**REGISTERED AS {m3100Package 2};**

### **4.3 Alarm Severity Assignment Pointer**

**alarmSeverityAssignmentPointerPackage PACKAGE**

**BEHAVIOUR**

alarmSeverityAssignmentPointerPackageBehaviour BEHAVIOUR  
**DEFINED AS**

"If the alarm severity assignment profile pointer is NULL, then one of the following two choices applies when reporting alarms:

- a) agent assigns the severity; or
  - b) the value 'indeterminate' is used."
- ;;

**ATTRIBUTES**

alarmSeverityAssignmentProfilePointer GET-REPLACE ;

**REGISTERED AS {m3100Package 3};**

### **4.4 Attribute Value Change Notification**

**attributeValueChangeNotificationPackage PACKAGE**

**NOTIFICATIONS**

"Recommendation X.721:1992":attributeValueChange;

**REGISTERED AS {m3100Package 4};**

### **4.5 Audible Visual Local Alarm**

**audibleVisualLocalAlarmPackage PACKAGE**

**ACTIONS**

allowAudibleVisualLocalAlarm,  
inhibitAudibleVisualLocalAlarm;

**REGISTERED AS {m3100Package 5};**

### **4.6 Channel Number**

**channelNumberPackage PACKAGE**

**ATTRIBUTES**

channelNumber GET SET-BY-CREATE;

**REGISTERED AS {m3100Package 6};**

#### 4.7 Characteristic Information

characteristicInformationPackage PACKAGE  
ATTRIBUTES  
characteristicInformation GET SET-BY-CREATE;  
REGISTERED AS {m3100Package 7};

#### 4.8 Client Connection List

clientConnectionListPackage PACKAGE  
ATTRIBUTES  
clientConnectionList GET SET-BY-CREATE;  
REGISTERED AS {m3100Package 35};

#### 4.9 Client Trail

clientTrailPackage PACKAGE  
ATTRIBUTES  
clientTrail GET SET-BY-CREATE;  
REGISTERED AS {m3100Package 9};

#### 4.10 Create Delete Notifications

createDeleteNotificationsPackage PACKAGE  
NOTIFICATIONS  
"Recommendation X.721:1992":objectCreation,  
"Recommendation X.721:1992":objectDeletion;  
REGISTERED AS {m3100Package 10};

#### 4.11 Cross-Connection Pointer

crossConnectionPointerPackage PACKAGE  
ATTRIBUTES  
crossConnectionObjectPointer GET;  
REGISTERED AS {m3100Package 11};

#### 4.12 CTP Instance

ctpInstancePackage PACKAGE  
ATTRIBUTES  
cTPId GET SET-BY-CREATE;  
REGISTERED AS {m3100Package 12};

#### 4.13 Current Problem List

currentProblemListPackage PACKAGE  
ATTRIBUTES  
currentProblemList GET;  
REGISTERED AS {m3100Package 13};

#### 4.14 Environmental Alarm

environmentalAlarmPackage PACKAGE  
NOTIFICATIONS  
"Recommendation X.721:1992":environmentalAlarm;  
REGISTERED AS {m3100Package 14};

#### 4.15 Environmental AlarmR1

environmentalAlarmR1Package PACKAGE

##### NOTIFICATIONS

"Recommendation X.721:1992":environmentalAlarm  
"Recommendation Q.821:1992":logRecordIdParameter  
"Recommendation Q.821:1992":correlatedRecordNameParameter  
"Recommendation Q.821:1992":suspectObjectListParameter;

REGISTERED AS {m3100Package 36};

#### 4.16 Equipment Alarm Effect on Service

equipmentAlarmEffectOnServicePackage PACKAGE

##### NOTIFICATIONS

"Recommendation X.721:1992":equipmentAlarm  
alarmEffectOnServiceParameter;

REGISTERED AS {m3100Package 38};

#### 4.17 Equipments Equipment Alarm

equipmentsEquipmentAlarmPackage PACKAGE

##### ATTRIBUTES

alarmStatus GET;

##### NOTIFICATIONS

"Recommendation X.721:1992":equipmentAlarm;

REGISTERED AS {m3100Package 15};

#### 4.18 Equipments Equipment AlarmR1

equipmentsEquipmentAlarmR1Package PACKAGE

##### ATTRIBUTES

alarmStatus GET;

##### NOTIFICATIONS

"Recommendation X.721:1992":equipmentAlarm  
"Recommendation Q.821:1992":logRecordIdParameter  
"Recommendation Q.821:1992":correlatedRecordNameParameter  
"Recommendation Q.821:1992":suspectObjectListParameter;

REGISTERED AS {m3100Package 37};

#### 4.19 External Time

externalTimePackage PACKAGE

##### ATTRIBUTES

externalTime GET-REPLACE;

REGISTERED AS {m3100Package 16};

#### 4.20 Location Name

locationNamePackage PACKAGE

##### ATTRIBUTES

locationName GET-REPLACE;

REGISTERED AS {m3100Package 17};

#### 4.21 Named Cross-Connection

namedCrossConnectionPackage PACKAGE

##### ATTRIBUTES

redline GET-REPLACE,  
crossConnectionName GET-REPLACE;;

-- The above package is not registered because it is used as a mandatory package in this  
-- Recommendation.



#### 4.22 Network Level

**networkLevelPackage**        **PACKAGE**  
    **BEHAVIOUR**

**networkLevelPackageBehaviour**        **BEHAVIOUR**  
        **DEFINED AS**

            “The network level pointer identifies a network level object. The value of the network level pointer shall only be modified by the managing system.”

        ;;

**ATTRIBUTES**

**networkLevelPointer** **GET-REPLACE;**

**REGISTERED AS {m3100Package 18};**

#### 4.23 Operational State

**operationalStatePackage** **PACKAGE**  
    **ATTRIBUTES**

**"Recommendation X.721:1992":operationalState** **GET;**

**REGISTERED AS {m3100Package 19};**

#### 4.24 Object Management Notifications

**objectManagementNotificationsPackage**        **PACKAGE**  
    **NOTIFICATIONS**

**"Recommendation X.721:1992":objectCreation,**

**"Recommendation X.721:1992":objectDeletion,**

**"Recommendation X.721:1992":attributeValueChange;**

**REGISTERED AS {m3100Package 20};**

#### 4.25 Processing Error Alarm

**processingErrorAlarmPackage** **PACKAGE**  
    **NOTIFICATIONS**

**"Recommendation X.721:1992":processingErrorAlarm;**

**REGISTERED AS {m3100Package 21};**

#### 4.26 Processing Error AlarmR1

**processingErrorAlarmR1Package** **PACKAGE**  
    **NOTIFICATIONS**

**"Recommendation X.721:1992":processingErrorAlarm**

**"Recommendation Q.821:1992":logRecordIdParameter**

**"Recommendation Q.821:1992":correlatedRecordNameParameter**

**"Recommendation Q.821:1992":suspectObjectListParameter;**

**REGISTERED AS {m3100Package 39};**

#### 4.27 Protected

**protectedPackage**        **PACKAGE**  
    **ATTRIBUTES**

**protected**        **GET SET-BY-CREATE;**

**REGISTERED AS {m3100Package 22};**

#### 4.28 Reset Audible Alarm

**resetAudibleAlarmPackage** **PACKAGE**  
    **ACTIONS**

**"Recommendation Q.821:1992":resetAudibleAlarm;**

**REGISTERED AS {m3100Package 23};**

#### 4.29 Server Connection List

serverConnectionListPackage PACKAGE  
ATTRIBUTES  
serverConnectionList GET SET-BY-CREATE;  
REGISTERED AS {m3100Package 24};

#### 4.30 Server Trail List

serverTrailListPackage PACKAGE  
ATTRIBUTES  
serverTrailList GET SET-BY-CREATE;  
REGISTERED AS {m3100Package 25};

#### 4.31 Software Processing Error Alarm

softwareProcessingErrorAlarmPackage PACKAGE  
ATTRIBUTES  
alarmStatus GET;  
NOTIFICATIONS  
"Recommendation X.721:1992":processingErrorAlarm;  
REGISTERED AS {m3100Package 26};

#### 4.32 Software Processing Error AlarmR1

softwareProcessingErrorAlarmR1Package PACKAGE  
ATTRIBUTES  
alarmStatus GET;  
NOTIFICATIONS  
"Recommendation X.721:1992":processingErrorAlarm  
"Recommendation Q.821:1992":logRecordIdParameter  
"Recommendation Q.821:1992":correlatedRecordNameParameter  
"Recommendation Q.821:1992":suspectObjectListParameter;  
REGISTERED AS {m3100Package 40};

#### 4.33 Subordinate Circuit Pack

subordinateCircuitPackPackage PACKAGE  
ATTRIBUTES  
acceptableCircuitPackTypeList GET-REPLACE ADD-REMOVE,  
holderStatus GET,  
subordinateCircuitPackSoftwareLoad GET-REPLACE;  
REGISTERED AS {m3100Package 41};

#### 4.34 Supportable Client List

supportableClientListPackage PACKAGE  
ATTRIBUTES  
supportableClientList GET SET-BY-CREATE;  
REGISTERED AS {m3100Package 27};

#### 4.35 State Change Notification

stateChangeNotificationPackage PACKAGE  
NOTIFICATIONS  
"Recommendation X.721:1992":stateChange;  
REGISTERED AS {m3100Package 28};

#### 4.36 System Timing Source

systemTimingSourcePackage PACKAGE  
ATTRIBUTES  
systemTimingSource GET-REPLACE;  
REGISTERED AS {m3100Package 29};

#### 4.37 TMN Communications Alarm Information

tmnCommunicationsAlarmInformationPackage PACKAGE  
BEHAVIOUR  
tmnCommunicationsAlarmInformationBehaviour;  
ATTRIBUTES  
alarmStatus GET,  
currentProblemList GET;  
NOTIFICATIONS  
"Recommendation X.721:1992":communicationsAlarm  
"Recommendation Q.821:1992":logRecordIdParameter  
"Recommendation Q.821:1992":correlatedRecordNameParameter  
"Recommendation Q.821:1992":suspectObjectListParameter;

REGISTERED AS {m3100Package 30};

tmnCommunicationsAlarmInformationBehaviour BEHAVIOUR  
-- The following behaviour text is taken directly from 5.3.1.1/Q.821  
DEFINED AS

“An alarm report which contains a Perceived Severity parameter with a value of 'cleared' and a Correlated Notifications parameter shall only indicate the clearing of those alarms whose Notification Identifiers are included in the set of Correlated Notifications. An alarm report which contains a Perceived Severity parameter with a value of 'cleared', but no Correlated Notifications parameter, shall indicate the clearing of alarms based on the value of the Alarm Type, Probable Cause, and Specific Problems parameters.

The parameters that are associated with the communications alarm, if present, are placed in individual elements of the SET OF ManagementExtension in the additionalInformation field of the notification.”;

#### 4.38 TTP Instance

ttpInstancePackage PACKAGE  
ATTRIBUTES  
tTPId GET SET-BY-CREATE;  
REGISTERED AS {m3100Package 31};

#### 4.39 User Label

userLabelPackage PACKAGE  
ATTRIBUTES  
userLabel GET-REPLACE;  
REGISTERED AS {m3100Package 32};

#### 4.40 Vendor Name

vendorNamePackage PACKAGE  
ATTRIBUTES  
vendorName GET-REPLACE;  
REGISTERED AS {m3100Package 33};

#### 4.41 Version

versionPackage PACKAGE  
ATTRIBUTES  
version GET-REPLACE;  
REGISTERED AS {m3100Package 34};

## 5 Attributes

### 5.1 A-Termination Point Instance

**a-TPInstance ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.RelatedObjectInstance;**  
**MATCHES FOR EQUALITY;**  
**BEHAVIOUR**  
**aTPInstanceBehaviour BEHAVIOUR**  
**DEFINED AS**

“The A-Termination Point Instance attribute type identifies one of the two termination points of an instance of the connectivity object class or one of its subclasses.”;;

**REGISTERED AS {m3100Attribute 1};**

### 5.2 Acceptable Circuit Pack Type List

**acceptableCircuitPackTypeList ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.AcceptableCircuitPackTypeList;**  
**MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION;**  
**BEHAVIOUR**  
**acceptableCircuitPackTypeListBehaviour BEHAVIOUR**  
**DEFINED AS**

“This attribute indicates the types of the circuit packs that can be contained in an equipment holder object.”;;

**REGISTERED AS {m3100Attribute 58};**

### 5.3 Administrative State

The Administrative State attribute is defined in Recommendation X.721[5].

### 5.4 Affected Object List

**affectedObjectList ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.ObjectList;**  
**MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION;**  
**BEHAVIOUR**  
**affectedObjectListBehaviour BEHAVIOUR**  
**DEFINED AS**

“The Affected Object List attribute type specifies the object instances which can be directly affected by a change in state or deletion of a given managed object. The attribute does not force internal details to be specified, but only the necessary level of detail required for management.”;;

**REGISTERED AS {m3100Attribute 2};**

### 5.5 Alarm Severity Assignment List

**alarmSeverityAssignmentList ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.AlarmSeverityAssignmentList;**  
**BEHAVIOUR**  
**alarmSeverityAssignmentListBehaviour BEHAVIOUR**  
**DEFINED AS**

“The Alarm Severity Assignment List is an attribute type whose value provides a listing of all abnormal conditions that may exist in instances of an object class, and shows the assigned alarm severity information (minor, major, etc.) for each condition.”;;

**REGISTERED AS {m3100Attribute 3};**

## 5.6 Alarm Severity Assignment Profile Id

**alarmSeverityAssignmentProfileId** ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.NameType;

MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;

BEHAVIOUR

"Recommendation X.721 : 1992" : rDNIdBehaviour,

-- The above behaviour is defined as part of discriminatorId in Recommendation X.721

**alarmSeverityAssignmentProfileIdBehaviour** BEHAVIOUR

DEFINED AS

"The Alarm Severity AssignmentProfile Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the Alarm SeverityAssignment Profile object class.";;

REGISTERED AS {m3100Attribute 4};

## 5.7 Alarm Severity Assignment Profile Pointer

**alarmSeverityAssignmentProfilePointer** ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.PointerOrNull;

MATCHES FOR EQUALITY;

BEHAVIOUR

**alarmSeverityAssignmentProfilePointerBehaviour** BEHAVIOUR

DEFINED AS

"This attribute identifies a Alarm Severity Assignment Profile object.";;

REGISTERED AS {m3100Attribute 5};

## 5.8 Alarm Status

**alarmStatus** ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.AlarmStatus;

MATCHES FOR EQUALITY;

BEHAVIOUR

**alarmStatusBehaviour**

BEHAVIOUR

DEFINED AS

"The Alarm Status attribute type indicates the occurrence of an abnormal condition relating to an object. This attribute may also function as a summary indicator of alarm conditions associated with a specific resource. It is used to indicate the existence of an alarm condition, a pending alarm condition such as threshold situations, or (when used as a summary indicator) the highest severity of active alarm conditions. When used as a summary indicator, the order of severity (from highest to lowest) is:

activeReportable-Critical  
activeReportable-Major  
activeReportable-Minor  
activeReportable-Indeterminate  
activeReportable-Warning  
activePending  
cleared.";;

REGISTERED AS {m3100Attribute 6};

## 5.9 Channel Number

**channelNumber** ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.ChannelNumber;

MATCHES FOR EQUALITY, ORDERING;

REGISTERED AS {m3100Attribute 7};

## 5.10 Characteristic Information

**characteristicInformation ATTRIBUTE**  
**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.CharacteristicInformation;**  
**MATCHES FOR EQUALITY;**  
**BEHAVIOUR**  
**characteristicInformationBehaviour BEHAVIOUR**  
**DEFINED AS**

“The value of this attribute is used to verify the connectability of instances of the termination point subclasses.”;;

**REGISTERED AS {m3100Attribute 8};**

## 5.11 Circuit Directionality

**circuitDirectionality ATTRIBUTE**  
**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.CircuitDirectionality;**  
**MATCHES FOR EQUALITY;**  
**BEHAVIOUR**  
**circuitDirectionalityBehaviour BEHAVIOUR**  
**DEFINED AS**

“The attribute type specifies the directionality of the circuits in the circuit subgroup.”;;

**REGISTERED AS {m3100Attribute 66};**

## 5.12 Circuit End Point Subgroup Id

**circuitEndPointSubgroupId ATTRIBUTE**  
**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.NameType;**  
**MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;**  
**BEHAVIOUR**  
**"Recommendation X.721 : 1992" : rDNIdBehaviour,**

*-- The above behaviour is defined as part of discriminatorId in Recommendation X.721*

**circuitEndPointSubgroupIdBehaviour BEHAVIOUR**  
**DEFINED AS**

“The circuitEndPointsubgroup Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the circuitEndPointsubgroup object class.”;;

**REGISTERED AS {m3100Attribute 61};**

## 5.13 Circuit Pack Type

**circuitPackType ATTRIBUTE**  
**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.CircuitPackType;**  
**MATCHES FOR EQUALITY, SUBSTRINGS;**  
**BEHAVIOUR**  
**circuitPackTypeBehaviour BEHAVIOUR**  
**DEFINED AS**

“This attribute indicates the type of the circuit pack.”;;

**REGISTERED AS {m3100Attribute 54};**

## 5.14 Client Connection List

**clientConnectionList ATTRIBUTE**  
**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.ObjectList;**  
**MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION;**  
**BEHAVIOUR**  
**clientConnectionListBehaviourBEHAVIOUR**

## DEFINED AS

“The values of this attribute identify the client connections served by a trail. These client connections may be either of a lower or same rate as the trail. In the latter case, the list consists of only one client connection.”;

REGISTERED AS {m3100Attribute 53};

### 5.15 Client Trail

#### clientTrail ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.ObjectInstance;  
MATCHES FOR EQUALITY;  
BEHAVIOUR  
    clientTrailBehaviour BEHAVIOUR  
DEFINED AS

“The value of this attribute identifies the trail object instance in the same network layer as the connection served by a connection object.”;

REGISTERED AS {m3100Attribute 10};

### 5.16 Connected Termination Point Count

#### connectedTpCount ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.Count;  
MATCHES FOR EQUALITY, ORDERING;  
BEHAVIOUR  
    connectedTpCountBehaviour BEHAVIOUR  
DEFINED AS

“This attribute indicates the total number of termination points associated with a tpPool that have been connected.”;

REGISTERED AS {m3100Attribute 11};

### 5.17 Connection Id

#### connectionId ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.NameType;  
MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;  
BEHAVIOUR  
    "Recommendation X.721 : 1992" : rDNIdBehaviour,  
    -- The above behaviour is defined as part of discriminatorId in Recommendation X.721  
    connectionIdBehaviour BEHAVIOUR  
DEFINED AS

“The Connection Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the Connection object class.”;

REGISTERED AS {m3100Attribute 12};

### 5.18 Connection Termination Point Id

#### cTPIId ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.NameType;  
MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;  
BEHAVIOUR  
    "Recommendation X.721 : 1992" : rDNIdBehaviour,  
    -- The above behaviour is defined as part of discriminatorId in Recommendation X.721

REGISTERED AS {m3100Attribute 13};

## 5.19 Cross-Connection Id

**crossConnectionId ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.NameType;

**MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;**

**BEHAVIOUR**

"Recommendation X.721 : 1992" : rDNIdBehaviour,

-- The above behaviour is defined as part of discriminatorId in Recommendation X.721

**crossConnectionIdBehaviour BEHAVIOUR**

**DEFINED AS**

"The Cross-Connection Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the crossConnection object class.";;

**REGISTERED AS {m3100Attribute 14};**

## 5.20 Cross-Connection Name

**crossConnectionName ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.CrossConnectionName;

**MATCHES FOR EQUALITY, SUBSTRINGS;**

**BEHAVIOUR**

**crossConnectionNameBehaviour BEHAVIOUR**

**DEFINED AS**

"This attribute is a descriptive name for a cross-connection or multipoint cross-connection managed object.";;

**REGISTERED AS {m3100Attribute 15};**

## 5.21 Cross-Connection Object Pointer

**crossConnectionObjectPointer ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX**

ASN1DefinedTypesModule.CrossConnectionObjectPointer;

**MATCHES FOR EQUALITY;**

**BEHAVIOUR**

**crossConnectionObjectPointerBehaviour BEHAVIOUR**

**DEFINED AS**

"This attribute points to a managed object such as a Cross-connection, a GTP or a Fabric. When a termination point is neither connected nor reserved for connection, its crossConnectionObjectPointer points to the Fabric object responsible for its connection.";;

**REGISTERED AS {m3100Attribute 16};**

## 5.22 Current Problem List

**currentProblemList ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.CurrentProblemList;

**BEHAVIOUR**

**currentProblemListBehaviour BEHAVIOUR**

**DEFINED AS**

"The Current Problem List attribute type identifies the current existing problems, with severity, associated with the managed object.";;

**REGISTERED AS {m3100Attribute 17};**

## 5.23 Directionality

**directionality ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.Directionality;

**MATCHES FOR EQUALITY;**

**BEHAVIOUR**

**directionalityBehaviour**

**BEHAVIOUR**



## DEFINED AS

“The Directionality attribute type specifies whether the associated managed object is uni- or bi-directional.”;;

REGISTERED AS {m3100Attribute 18};

### 5.24 Downstream Connectivity Pointer

downstreamConnectivityPointer ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.DownstreamConnectivityPointer;

MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION;

BEHAVIOUR

downstreamConnectivityPointerBehaviour BEHAVIOUR

DEFINED AS

“The matching for equality is applicable for all choices of the syntax. The set operations are permitted only when the choice of the syntax correspond to either broadcast or concatenated broadcast.”;;

REGISTERED AS {m3100Attribute 19};

### 5.25 Equipment Holder Address

equipmentHolderAddress ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.EquipmentHolderAddress;

MATCHES FOR EQUALITY, SUBSTRINGS;

BEHAVIOUR

equipmentHolderAddressBehaviour BEHAVIOUR

DEFINED AS

“This attribute indicates the physical location of the resource represented by the equipmentHolder instance. Depending on the containment hierarchy of the equipmentHolder in the managed system, the value of this attribute may vary. For example, if a system has three levels of equipment holders representing Bay, Shelf and Slot respectively (i.e. the managed Element contains multiple Bay equipment holders, each Bay equipment holder contains multiple Shelf equipment holders and each Shelf equipment holder contains multiple Slot equipment holders), then:

- for the equipmentHolder representing a Bay, the Frame Identification code may be used as the value of this attribute;
- for the equipmentHolder representing a Shelf, the Bay Shelf code may be used as the value of this attribute;
- for the equipmentHolder representing a Slot, the position code may be used as the value of this attribute.

If the system uses only one level of equipment holder, that represents the Shelves (i.e., the managedElement contains multiple Shelf equipment holders, and each Shelf equipment holder contains a circuit pack), then the value of this attribute is a sequence of the Frame Identification code and the Bay Shelf Code.”;;

REGISTERED AS {m3100Attribute 56};

### 5.26 Equipment Holder Type

equipmentHolderType ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.EquipmentHolderType;

MATCHES FOR EQUALITY;

BEHAVIOUR

equipmentHolderTypeBehaviour BEHAVIOUR

DEFINED AS

“The equipment holder type attribute indicates the type of equipment holder using a character string. The possible values for the character string may be bay, shelf, drawer, slot and rack.”;;

REGISTERED AS {m3100Attribute 57};

## 5.27 Equipment Id

**equipmentId ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.NameType;

**MATCHES FOR** EQUALITY, ORDERING, SUBSTRINGS;

**BEHAVIOUR**

"Recommendation X.721 : 1992" : rDNIdBehaviour,

-- The above behaviour is defined as part of discriminatorId in Recommendation X.721

**equipmentIdBehaviour**

**BEHAVIOUR**

**DEFINED AS**

"The Equipment Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the Equipment object class.";;

**REGISTERED AS** {m3100Attribute 20};

## 5.28 External Time

**externalTime ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.ExternalTime;

**MATCHES FOR** EQUALITY;

**BEHAVIOUR**

**externalTimeBehaviour** **BEHAVIOUR**

**DEFINED AS**

"The External time attribute provides time-of-day system time. The attribute functions as a reference for all time stamp activities in the managed element.";;

**REGISTERED AS** {m3100Attribute 21};

## 5.29 Fabric Id

**fabricId ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.NameType;

**MATCHES FOR** EQUALITY, ORDERING, SUBSTRINGS;

**BEHAVIOUR**

"Recommendation X.721 : 1992" : rDNIdBehaviour,

-- The above behaviour is defined as part of discriminatorId in Recommendation X.721

**fabricIdBehaviour** **BEHAVIOUR**

**DEFINED AS**

"The Fabric Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the Fabric object class.";;

**REGISTERED AS** {m3100Attribute 22};

## 5.30 From Termination

**fromTermination ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.PointerOrNull;

**MATCHES FOR** EQUALITY;

**BEHAVIOUR**

**fromTerminationBehaviour** **BEHAVIOUR**

**DEFINED AS**

"This attribute identifies a TTP (source or bidirectional), a CTP (sink or bidirectional) or a GTP composed of members of one of these categories.";;

**REGISTERED AS** {m3100Attribute 23};

## 5.31 Group Termination Point Id

**gtpId ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.NameType;

**MATCHES FOR** EQUALITY, ORDERING, SUBSTRINGS;

## BEHAVIOUR

"Recommendation X.721 : 1992" : rDNIDBehaviour,

-- The above behaviour is defined as part of discriminatorId in Recommendation X.721

gtpIdBehaviour BEHAVIOUR

DEFINED AS

"The gtp Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the gtp object class.";;

REGISTERED AS {m3100Attribute 24};

## 5.32 Holder Status

holderStatus ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.HolderStatus;

MATCHES FOR EQUALITY, SUBSTRINGS;

BEHAVIOUR

holderStatusBehaviour BEHAVIOUR

DEFINED AS

"The holderStatus attribute indicates the status of the physical holder. It specifies if the holder is empty or has a circuitPack of specific type (that may or not be acceptable to the holder) or has an indeterminate type.";;

REGISTERED AS {m3100Attribute59};

## 5.33 Idle TP Count

idleTpCount ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.Count;

MATCHES FOR EQUALITY, ORDERING;

BEHAVIOUR

idleTpCountBehaviour BEHAVIOUR

DEFINED AS

"This attribute indicates the total number of termination points associated with a tpPool that are in an operational state of enabled and that are available for Cross-Connection.";;

REGISTERED AS {m3100Attribute 25};

## 5.34 Information Transfer Capabilities

informationTransferCapabilities ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.InformationTransferCapabilities;

MATCHES FOR EQUALITY;

BEHAVIOUR

informationTransferCapabilitiesBehaviour BEHAVIOUR

DEFINED AS

"The attribute type specifies the different service types such as speech, 64 kbits unrestricted data supported by the circuit subgroup.";;

REGISTERED AS {m3100Attribute 65};

## 5.35 Label of Far End Exchange

labelOfFarEndExchange ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.UserLabel;

MATCHES FOR EQUALITY, SUBSTRINGS;

BEHAVIOUR

labelOfFarEndExchangeBehaviour BEHAVIOUR

DEFINED AS

"This attribute type assigns a user friendly name to the Far End Exchange terminating this circuit subgroup.";;

REGISTERED AS {m3100Attribute 63};

### 5.36 List of Characteristic Info

**listOfCharacteristicInfo ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.ListOfCharacteristicInformation;

**MATCHES FOR EQUALITY;**

**BEHAVIOUR**

**listOfCharacteristicInfoBehaviour BEHAVIOUR**

**DEFINED AS**

“This attribute lists the characteristic information types that can be cross-connected by a Fabric.”;;

**REGISTERED AS {m3100Attribute 26};**

### 5.37 Location Name

**locationName ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.LocationName;

**MATCHES FOR EQUALITY, SUBSTRINGS;**

**BEHAVIOUR**

**locationNameBehaviour BEHAVIOUR**

**DEFINED AS**

“The Location Name attribute type identifies a location.”;;

**REGISTERED AS {m3100Attribute 27};**

### 5.38 Managed Element Id

**managedElementId ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.NameType;

**MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;**

**BEHAVIOUR**

*"Recommendation X.721 : 1992" : rDNIdBehaviour,*

*-- The above behaviour is defined as part of discriminatorId in Recommendation X.721*

**managedElementIdBehaviour BEHAVIOUR**

**DEFINED AS**

“The Managed Element Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the Managed Element object class.”;;

**REGISTERED AS {m3100Attribute 28};**

### 5.39 Managed Element Complex Id

**managedElementComplexId ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.NameType;

**MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;**

**BEHAVIOUR**

*"Recommendation X.721 : 1992" : rDNIdBehaviour,*

*-- The above behaviour is defined as part of discriminatorId in Recommendation X.721*

**managedElementComplexIdBehaviour BEHAVIOUR**

**DEFINED AS**

“The Managed Element Complex Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the Managed Element Complex object class.”;;

**REGISTERED AS {m3100Attribute 68};**

### 5.40 Multi-Point Cross-Connection Id

**mpCrossConnectionId ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.NameType;

**MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;**

## BEHAVIOUR

**"Recommendation X.721 : 1992" : rDNIdBehaviour,**  
*-- The above behaviour is defined as part of discriminatorId in Recommendation X.721*  
**mpCrossConnectionIdBehaviour      BEHAVIOUR**  
**DEFINED AS**

“The mp Cross-Connection Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the mpCrossConnection object class.”;

**REGISTERED AS {m3100Attribute 29};**

## 5.41 Network Id

### networkId ATTRIBUTE

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.NameType;**  
**MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;**  
**BEHAVIOUR**

**"Recommendation X.721 : 1992" : rDNIdBehaviour,**  
*-- The above behaviour is defined as part of discriminatorId in Recommendation X.721*  
**networkIdBehaviour      BEHAVIOUR**  
**DEFINED AS**

“The Network Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the Network object class.”;

**REGISTERED AS {m3100Attribute 30};**

## 5.42 Network Level Pointer

### networkLevelPointer ATTRIBUTE

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.ObjectInstance;**  
**MATCHES FOR EQUALITY;**

**REGISTERED AS {m3100Attribute 31};**

## 5.43 Number of Circuits

### numberOfCircuits ATTRIBUTE

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.NumberOfCircuits;**  
**MATCHES FOR EQUALITY, ORDERING;**  
**BEHAVIOUR**  
**numberOfCircuitsBehaviour      BEHAVIOUR**  
**DEFINED AS**

“The number of circuits in a circuit subgroup.”;

**REGISTERED AS {m3100Attribute 62};**

## 5.44 Operational State

The Operational State attribute is defined in Recommendation X.721[5].

## 5.45 Protected

### protected ATTRIBUTE

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.Boolean;**  
**MATCHES FOR EQUALITY;**  
**BEHAVIOUR**  
**protectedBehaviour BEHAVIOUR**  
**DEFINED AS**

“This attribute identifies whether the associated managed object is protected or not. The value TRUE implies it is protected.”;

**REGISTERED AS {m3100Attribute 32};**

## 5.46 Redline

### redline ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.Boolean;

MATCHES FOR EQUALITY;

#### BEHAVIOUR

redlineBehaviour BEHAVIOUR

DEFINED AS

“This attribute identifies whether the associated managed object is red lined, e.g. identified as being part of a sensitive circuit.”;;

REGISTERED AS {m3100Attribute 33};

## 5.47 Replaceable

### replaceable ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.Replaceable;

MATCHES FOR EQUALITY;

#### BEHAVIOUR

replaceableBehaviour BEHAVIOUR

DEFINED AS

“The Replaceable attribute type indicates whether the associated managed object is replaceable or non-replaceable.”;;

REGISTERED AS {m3100Attribute 34};

## 5.48 Serial Number

### serialNumber ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.SerialNumber;

MATCHES FOR EQUALITY, SUBSTRINGS;

#### BEHAVIOUR

serialNumberBehaviour BEHAVIOUR

DEFINED AS

“The serial number attribute type identifies the serial number of the physical resource.”;;

REGISTERED AS {m3100Attribute69};

## 5.49 Server Connection List

### serverConnectionList ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.SequenceOfObjectInstance;

MATCHES FOR EQUALITY;

#### BEHAVIOUR

serverConnectionListBehaviour BEHAVIOUR

DEFINED AS

“The value of this attribute identifies one or more connection objects within the same network layer as the trail that are connected in series to constitute the trail.”;;

REGISTERED AS {m3100Attribute 35};

## 5.50 Server Trail List

### serverTrailList ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.ObjectList;

MATCHES FOR EQUALITY;

#### BEHAVIOUR

serverTrailListBehaviour BEHAVIOUR

DEFINED AS

“The value of this attribute identifies the trail objects (in most cases one) in a lower order network layer which may be used in parallel to serve a connection object.”;;

REGISTERED AS {m3100Attribute 36};

## 5.51 Signalling Capabilities

**signallingCapabilities ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.SignallingCapabilities;

**MATCHES FOR EQUALITY;**

**BEHAVIOUR**

**signallingCapabilitiesBehaviour      BEHAVIOUR**  
**DEFINED AS**

“The attribute type specifies the signalling types supported by the circuit subgroup.”;;

**REGISTERED AS {m3100Attribute 64};**

## 5.52 Signal Type

**signalType ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.SignalType;

**MATCHES FOR EQUALITY;**

**BEHAVIOUR**

**signalTypeBehaviour BEHAVIOUR**  
**DEFINED AS**

“This attribute uniquely identifies the signal type of a cross-connection, TP pool or GTP. The signal type can either be simple, bundle, or complex. If the signal type is simple, it consists of a single type of characteristic information. If the signal type is bundle, it is made up of a number of signal types all of the same characteristic information. If the signal type is complex, it consists of a sequence of bundle signal type. The order in the complex signal type represents the actual composition of the signal.”;;

**REGISTERED AS {m3100Attribute 37};**

## 5.53 Software Id

**softwareId ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.NameType;

**MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;**

**BEHAVIOUR**

**"Recommendation X.721 : 1992" : rDNIdBehaviour,**

*-- The above behaviour is defined as part of discriminatorId in Recommendation X.721*

**softwareIdBehaviour      BEHAVIOUR**  
**DEFINED AS**

“The Software Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the Software object class.”;;

**REGISTERED AS {m3100Attribute 38};**

## 5.54 Subordinate Circuit Pack Software Load

**subordinateCircuitPackSoftwareLoad      ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.SubordinateCircuitPackSoftwareLoad;

**BEHAVIOUR**

**subordinateCircuitPackSoftwareLoadBehaviour      BEHAVIOUR**  
**DEFINED AS**

“This attribute indicates the software, if any, that is currently designated as the one to be loaded to the containing circuit pack whenever automatic reload of software is needed. The values of this attribute may be one of the following: sequence of printable string, sequence of object instance or NULL. The NULL choice is used when the contained circuit pack is not software loadable, or no software load has been designated. The choice of sequence of ObjectInstance identifies an ordered set of software instances. When the choice of PrintableString is used, the semantics is a local matter.”;;

**REGISTERED AS {m3100Attribute 60};**

## 5.55 Supportable Client List

**supportableClientList ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.SupportableClientList;  
**MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION;**  
**BEHAVIOUR**

**supportableClientListBehaviour BEHAVIOUR**  
**DEFINED AS**

“The value of this attribute is the list of object classes representing the clients which the particular managed object is capable of supporting. This may be a subset of the client layers identified in Recommendation G.803 by the particular server layer managed object.”;

**REGISTERED AS {m3100Attribute 39};**

## 5.56 Supported By Object List

**supportedByObjectList ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.ObjectList;  
**MATCHES FOR EQUALITY, SET-COMPARISON, SET-INTERSECTION;**  
**BEHAVIOUR**

**supportedByObjectListBehaviour BEHAVIOUR**  
**DEFINED AS**

“The Supported By Object List is an attribute type whose value identifies a set of object instances which are capable of directly affecting a given managed object. The object instances include both physical and logical objects. This attribute does not force internal details to be specified, but only the necessary level of detail required for management. If the object instances supporting the managed object are unknown to that object, then this attribute is an empty set.”;

**REGISTERED AS {m3100Attribute 40};**

## 5.57 System Timing Source

**systemTimingSource ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.SystemTimingSource;  
**MATCHES FOR EQUALITY;**  
**BEHAVIOUR**

**systemTimingSourceBehaviour BEHAVIOUR**  
**DEFINED AS**

“The System Timing Source attribute is used to specify the primary and secondary managed element timing source for synchronization.”;

**REGISTERED AS {m3100Attribute 41};**

## 5.58 System Title

This attribute is defined in Recommendation X.721[5].

## 5.59 Total TP Count

**totalTpCount ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.Count;  
**MATCHES FOR EQUALITY, ORDERING;**  
**BEHAVIOUR**

**totalTpCountBehaviour BEHAVIOUR**  
**DEFINED AS**

“This attribute indicates the total number of termination points associated with a tpPool.”;

**REGISTERED AS {m3100Attribute 42};**

## 5.60 To Termination

**toTermination ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX** ASN1DefinedTypesModule.Pointer;  
**MATCHES FOR EQUALITY;**



## **BEHAVIOUR**

**toTerminationBehaviour BEHAVIOUR**  
**DEFINED AS**

“This attribute identifies a CTP (source or bidirectional), a TTP (sink or bidirectional) or a GTP composed of members of one of these categories.”;;

**REGISTERED AS {m3100Attribute 43};**

### **5.61 TP Pool Id**

#### **tpPoolId ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.NameType;**  
**MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;**  
**BEHAVIOUR**

**"Recommendation X.721 : 1992" : rDNIdBehaviour,**

*-- The above behaviour is defined as part of discriminatorId in Recommendation X.721*

**tpPoolIdBehaviour BEHAVIOUR**  
**DEFINED AS**

“The TP Pool Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the tpPool object class.”;;

**REGISTERED AS {m3100Attribute 44};**

### **5.62 TPs In GTP List**

#### **tpsInGtpList ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.TpsInGtpList;**  
**MATCHES FOR EQUALITY;**  
**BEHAVIOUR**

**tpsInGtpListBehaviour BEHAVIOUR**  
**DEFINED AS**

“This attribute lists the termination points that are represented by a GTP.”;;

**REGISTERED AS {m3100Attribute 45};**

### **5.63 TPs In TP Pool List**

#### **tpsInTpPoolList ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.ListOfTPs;**  
**MATCHES FOR EQUALITY;**  
**BEHAVIOUR**

**tpsInTpPoolListBehaviour BEHAVIOUR**  
**DEFINED AS**

“This attribute lists the termination points that are represented by a TP Pool.”;;

**REGISTERED AS {m3100Attribute 46};**

### **5.64 Trail Id**

#### **trailId ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.NameType;**  
**MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;**  
**BEHAVIOUR**

**"Recommendation X.721 : 1992" : rDNIdBehaviour,**

*-- The above behaviour is defined as part of discriminatorId in Recommendation X.721*

**trailIdBehaviour BEHAVIOUR**  
**DEFINED AS**

“The Trail Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the Trail object class.”;;

**REGISTERED AS {m3100Attribute 47};**

## 5.65 Trail Termination Point Id

### †TPId ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.NameType;

MATCHES FOR EQUALITY, ORDERING, SUBSTRINGS;

BEHAVIOUR

"Recommendation X.721 : 1992" : rDNIdBehaviour,

-- The above behaviour is defined as part of discriminatorId in Recommendation X.721

†tpIdBehaviour BEHAVIOUR

DEFINED AS

"The Trail Termination Point Id is an attribute type whose distinguished value can be used as a RDN when naming an instance of the Trail Termination Point object class.";;

REGISTERED AS {m3100Attribute 48};

## 5.66 Transmission Characteristics

### transmissionCharacteristics ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.TransmissionCharacteristics;

MATCHES FOR EQUALITY;

BEHAVIOUR

transmissionCharacteristicsBehaviour BEHAVIOUR

DEFINED AS

"The attribute type specifies the different transmission characteristics such as satellite, echo control supported or not supported by the circuit subgroup. The bit positions are set to indicate if a particular characteristic is supported.";;

REGISTERED AS {m3100Attribute 67};

## 5.67 Upstream Connectivity Pointer

### upstreamConnectivityPointer ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.ConnectivityPointer;

MATCHES FOR EQUALITY;

BEHAVIOUR

upstreamConnectivityPointerBehaviour BEHAVIOUR

DEFINED AS

"The matching for equality is applicable for all the choices of the syntax.";;

REGISTERED AS {m3100Attribute 49};

## 5.68 Usage State

The Usage State attribute is defined in Recommendation X.721[5].

## 5.69 User Label

### userLabel ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.UserLabel;

MATCHES FOR EQUALITY, SUBSTRINGS;

BEHAVIOUR

userLabelBehaviour BEHAVIOUR

DEFINED AS

"The User Label attribute type assigns a user friendly name to the associated object.";;

REGISTERED AS {m3100Attribute 50};

## 5.70 Vendor Name

### vendorName ATTRIBUTE

WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.VendorName;

MATCHES FOR EQUALITY, SUBSTRINGS;

**BEHAVIOUR**  
**vendorNameBehaviour      BEHAVIOUR**  
**DEFINED AS**

“The Vendor Name attribute type identifies the vendor of the associated managed object.”;

**REGISTERED AS {m3100Attribute 51};**

## **5.71      Version**

**version ATTRIBUTE**  
**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.Version;**  
**MATCHES FOR EQUALITY, SUBSTRINGS;**  
**BEHAVIOUR**  
**versionBehaviour      BEHAVIOUR**  
**DEFINED AS**

“The Version attribute type identifies the version of the associated managed object.”;

**REGISTERED AS {m3100Attribute 52};**

## **5.72      Z-Termination Point Instance**

**z-TPIInstance ATTRIBUTE**  
**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.RelatedObjectInstance;**  
**MATCHES FOR EQUALITY;**  
**BEHAVIOUR**  
**z-TPIInstanceBehaviour      BEHAVIOUR**  
**DEFINED AS**

“The Z-Termination Point Instance attribute type identifies one of the two termination points of an instance of the connectivity object class.”;

**REGISTERED AS {m3100Attribute 55};**

# **6          Name Bindings**

The naming hierarchy of the model is presented in Figure 8. The arrows are used to point from the subordinate object classes to the superior object classes. Each arrow represents a name binding defined in this Recommendation.

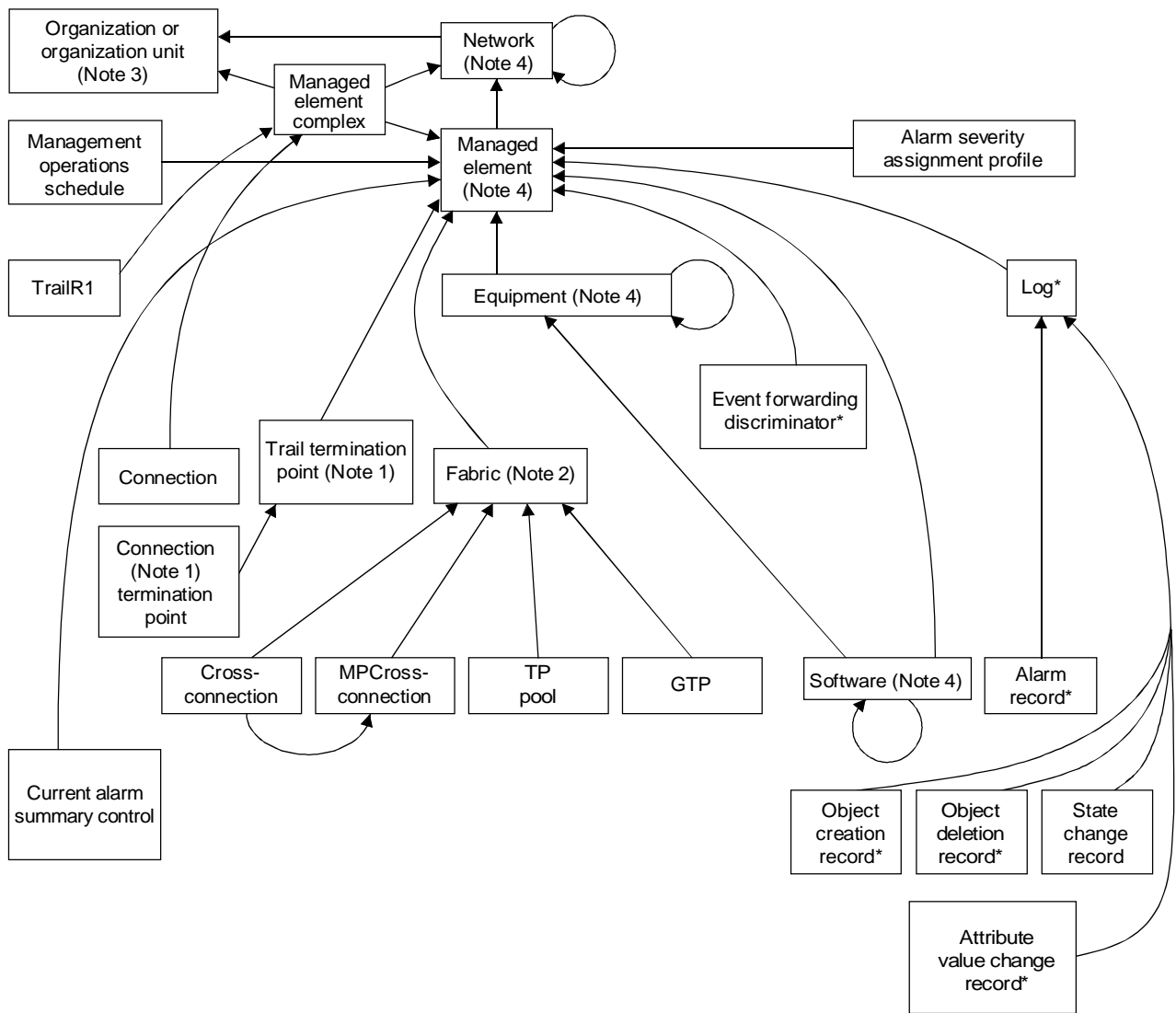
## **6.1      Alarm Record**

The name binding for alarm record is the same as that defined for log record in Recommendation X.721.

## **6.2      Alarm Severity Assignment Profile**

**alarmSeverityAssignment-managedElement NAME BINDING**  
**SUBORDINATE OBJECT CLASS      alarmSeverityAssignmentProfile AND SUBCLASSES;**  
**NAMED BY**  
**SUPERIOR OBJECT CLASS      managedElement AND SUBCLASSES;**  
**WITH ATTRIBUTE      alarmSeverityAssignmentProfileId;**  
**CREATE**  
**WITH-REFERENCE-OBJECT,**  
**WITH-AUTOMATIC-INSTANCE-NAMING;**  
**DELETE**  
**ONLY-IF-NO-CONTAINED-OBJECTS;**

**REGISTERED AS {m3100NameBinding 1};**



T0405560-95/d08

#### NOTES

- 1 Represents source and sink classes.
- 2 Fabric or FabricR1.
- 3 Directory objects in Recommendation X.521.
- 4 Subclasses not shown.
- 5 Object classes denoted by \* are defined in Recommendations X.721/Q.821 and referenced in this Recommendation.
- 6 Arrows are used to point from the subordinated object classes to the superior object classes.

FIGURE 8/M.3100  
Naming hierarchy

### 6.3 Circuit Pack

**circuitPack-equipmentHolder-autoCreated** NAME BINDING  
SUBORDINATE OBJECT CLASS **circuitPack** AND SUBCLASSES;  
NAMED BY SUPERIOR OBJECT CLASS  
**equipmentHolder** AND SUBCLASSES;  
WITH ATTRIBUTE **equipmentId**;  
BEHAVIOUR **circuitPack-equipmentHolder-autoCreateBeh**;

REGISTERED AS {m3100NameBinding 32};

**circuitPack-equipmentHolder-autoCreateBeh** BEHAVIOUR  
DEFINED AS

“This name binding is used to name an instance of an **circuitPack** relative to another **equipmentHolder** instance. The creation of the **circuitPack** object is the result of inserting the physical circuit pack into the resource represented by the superior object.

When the superior object contains a **circuitPack** object, the superior object shall not contain more than one **circuitPack** object, and shall not contain other **equipmentHolder** objects.”;

**circuitPack-equipmentHolder-explicitlyCreated** NAME BINDING  
SUBORDINATE OBJECT CLASS **circuitPack** AND SUBCLASSES;  
NAMED BY SUPERIOR OBJECT CLASS  
**equipmentHolder** AND SUBCLASSES;  
WITH ATTRIBUTE **equipmentId**;  
BEHAVIOUR **circuitPack-equipmentHolder-explicitlyCreateBeh**;  
CREATE  
WITH-REFERENCE-OBJECT,  
WITH-AUTOMATIC-INSTANCE-NAMING  
**createErrorParameter**;  
DELETE  
ONLY-IF-NO-CONTAINED-OBJECTS;

REGISTERED AS {m3100NameBinding 33};

**circuitPack-equipmentHolder-explicitlyCreateBeh** BEHAVIOUR  
DEFINED AS

“This name binding is used to name an instance of a **circuitPack** relative to another **equipmentHolder** instance. The creation of the **circuitPack** object is the result of system management protocol.

When the superior object contains a **circuitPack** object, the superior object shall not contain more than one **circuitPack** object, and shall not contain other **equipmentHolder** objects.”;

### 6.4 ConnectionR1

**connectionR1-network** NAME BINDING  
SUBORDINATE OBJECT CLASS **connectionR1**;  
NAMED BY  
SUPERIOR OBJECT CLASS **network**;  
WITH ATTRIBUTE **connectionId**;  
CREATE  
WITH-REFERENCE-OBJECT,  
WITH-AUTOMATIC-INSTANCE-NAMING;  
DELETE  
ONLY-IF-NO-CONTAINED-OBJECTS;

REGISTERED AS {m3100NameBinding 25};

### 6.5 Connection Termination Point Source

**connectionTerminationPointSource-trailTerminationPointSource** NAME BINDING  
SUBORDINATE OBJECT CLASS  
**connectionTerminationPointSource**;  
NAMED BY  
SUPERIOR OBJECT CLASS **trailTerminationPointSource** AND SUBCLASSES;  
WITH ATTRIBUTE **cTPId**;  
BEHAVIOUR  
**cTPSource-TTPBehaviour**;

```

CREATE
    WITH-REFERENCE-OBJECT,
    WITH-AUTOMATIC-INSTANCE-NAMING;
DELETE
    ONLY-IF-NO-CONTAINED-OBJECTS;

```

REGISTERED AS {m3100NameBinding 3};

connectionTerminationPointSource-trailTerminationPointBidirectional NAME BINDING

```

SUBORDINATE OBJECT CLASS
    connectionTerminationPointSource;
NAMED BY
SUPERIOR OBJECT CLASS trailTerminationPointBidirectional
    AND SUBCLASSES;
WITH ATTRIBUTE cTPId;
BEHAVIOUR
    cTPSource-TTPBehaviour;
CREATE
    WITH-REFERENCE-OBJECT,
    WITH-AUTOMATIC-INSTANCE-NAMING;
DELETE
    ONLY-IF-NO-CONTAINED-OBJECTS;

```

REGISTERED AS {m3100NameBinding 4};

cTPSource-TTPBehaviour BEHAVIOUR  
DEFINED AS

“The name binding represents a relationship in which a TTP receives information (traffic) from a source CTP.

When automatic instance naming is used, the choice of name bindings left as a local matter.”;

## 6.6 Connection Termination Point Sink

connectionTerminationPointSink-trailTerminationPointSink

```

NAME BINDING
SUBORDINATE OBJECT CLASS connectionTerminationPointSink;
NAMED BY
SUPERIOR OBJECT CLASS trailTerminationPointSink AND SUBCLASSES;
WITH ATTRIBUTE cTPId;
BEHAVIOUR
    cTPSink-TTPBehaviour;
CREATE
    WITH-REFERENCE-OBJECT,
    WITH-AUTOMATIC-INSTANCE-NAMING;
DELETE
    ONLY-IF-NO-CONTAINED-OBJECTS;
REGISTERED AS {m3100NameBinding 5};

```

connectionTerminationPointSink-trailTerminationPointBidirectional NAME BINDING

```

SUBORDINATE OBJECT CLASS connectionTerminationPointSink;
NAMED BY
SUPERIOR OBJECT CLASS trailTerminationPointBidirectional
    AND SUBCLASSES;
WITH ATTRIBUTE cTPId;
BEHAVIOUR
    cTPSink-TTPBehaviour;
CREATE
    WITH-REFERENCE-OBJECT,
    WITH-AUTOMATIC-INSTANCE-NAMING;
DELETE
    ONLY-IF-NO-CONTAINED-OBJECTS;

```

REGISTERED AS {m3100NameBinding 6};

cTPSink-TTPBehaviour BEHAVIOUR  
DEFINED AS

“The name binding represents a relationship in which a TTP sends information (traffic) to a sink CTP.

When automatic instance naming is used, the choice of name bindings left as a local matter.”;

## 6.7 Cross-Connection

**crossConnection-fabric NAME BINDING**

**SUBORDINATE OBJECT CLASS** crossConnection

**AND SUBCLASSES;**

**NAMED BY**

**SUPERIOR OBJECT CLASS** fabric

**AND SUBCLASSES;**

**WITH ATTRIBUTE** crossConnectionId;

**BEHAVIOUR**

**crossConnection-fabricBehaviour BEHAVIOUR**

**DEFINED AS**

“The value of the fromTermination attribute in the Cross-Connection object shall not be NULL. When an instance of cross-connection is deleted, the following attributes will be affected. The crossConnectionObjectPointer attribute in the termination point objects or in the GTP objects that were pointing to the deleted cross-connection instance shall be set to point to the Fabric responsible for the connection of the termination points.

The counters in the appropriate TP Pool objects (if applicable) shall be updated. The connectivityPointer attributes in the disconnected termination points shall be set to NULL. Deleting a cross-connection object instance has no effect on the composition of any GTP.”

**::**

**DELETE**

**ONLY-IF-NO-CONTAINED-OBJECTS;**

**REGISTERED AS {m3100NameBinding 7};**

**crossConnection-mpCrossConnection NAME BINDING**

**SUBORDINATE OBJECT CLASS** crossConnection

**AND SUBCLASSES;**

**NAMED BY**

**SUPERIOR OBJECT CLASS** mpCrossConnection

**AND SUBCLASSES;**

**WITH ATTRIBUTE** crossConnectionId;

**BEHAVIOUR**

**crossConnection-mpCrossConnectionBehaviour BEHAVIOUR**

**DEFINED AS**

“The value of the fromTermination attribute in the Cross-Connection object must be NULL. When an instance of cross-connection is deleted, the following attributes will be affected. The crossConnectionObjectPointer attribute in the termination point object or in the GTP object that was pointing to the deleted cross-connection instance shall be set to point to the Fabric responsible for the connection of the termination points. The counters in the appropriate TP Pool objects (if applicable) shall be updated. The connectivity pointers in the disconnected termination point shall be set to NULL.

Deleting the last cross-connection contained in a multipoint cross-connection object instance has the effect of also deleting the multipoint cross-connect object instance (and updating the appropriate pointers). Deleting a cross-connection object instance has no effect on the composition of any GTP.”;

**DELETE**

**ONLY-IF-NO-CONTAINED-OBJECTS;**

**REGISTERED AS {m3100NameBinding 8};**

## 6.8 Equipment

**equipment-managedElement NAME BINDING**

**SUBORDINATE OBJECT CLASS** equipment **AND SUBCLASSES;**

**NAMED BY**

**SUPERIOR OBJECT CLASS** managedElement **AND SUBCLASSES;**

**WITH ATTRIBUTE** equipmentId;

**BEHAVIOUR**

**equipmentNameBindingBehaviour;**

```

CREATE
    WITH-REFERENCE-OBJECT,
    WITH-AUTOMATIC-INSTANCE-NAMING;
DELETE
    ONLY-IF-NO-CONTAINED-OBJECTS;

REGISTERED AS {m3100NameBinding 9};

equipment-equipment NAME BINDING
    SUBORDINATE OBJECT CLASS      equipment AND SUBCLASSES;
    NAMED BY
    SUPERIOR OBJECT CLASS      equipment AND SUBCLASSES;
    WITH ATTRIBUTE      equipmentId;
    BEHAVIOUR
        equipmentNameBindingBehaviour;
    CREATE
        WITH-REFERENCE-OBJECT,
        WITH-AUTOMATIC-INSTANCE-NAMING;
    DELETE
        ONLY-IF-NO-CONTAINED-OBJECTS;

REGISTERED AS {m3100NameBinding 10};

equipmentNameBindingBehaviour
    BEHAVIOUR
    DEFINED AS

        "When automatic instance naming is used, the choice of name binding is left as a local matter.";

```

## 6.9 Equipment Holder

```

equipmentHolder-equipmentHolder NAME BINDING
    SUBORDINATE OBJECT CLASS equipmentHolder AND SUBCLASSES;
    NAMED BY SUPERIOR OBJECT CLASS
        equipmentHolder AND SUBCLASSES;
    WITH ATTRIBUTE equipmentId;
    BEHAVIOUR equipmentHolder-equipmentHolderBeh;
    CREATE
        WITH-REFERENCE-OBJECT,
        WITH-AUTOMATIC-INSTANCE-NAMING
        createErrorParameter;
    DELETE
        ONLY-IF-NO-CONTAINED-OBJECTS;

REGISTERED AS {m3100NameBinding 31};

equipmentHolder-equipmentHolderBeh BEHAVIOUR
    DEFINED AS

        "This name binding is used to name an instance of an equipmentHolder relative to another equipmentHolder
        instance. When an equipmentHolder contains another equipmentHolder, the superior equipmentHolder shall not
        contain any circuitPack.";

```

## 6.10 Event Forwarding Discriminator

```

eventForwardingDiscriminator-managedElement NAME BINDING
    SUBORDINATE OBJECT CLASS
        "Recommendation X.721:1992":eventForwardingDiscriminator;
    NAMED BY
    SUPERIOR OBJECT CLASS      managedElement AND SUBCLASSES;
    WITH ATTRIBUTE      "Recommendation X.721:1992":discriminatorId;
    CREATE
        WITH-REFERENCE-OBJECT,
        WITH-AUTOMATIC-INSTANCE-NAMING;
    DELETE
        ONLY-IF-NO-CONTAINED-OBJECTS;

REGISTERED AS {m3100NameBinding 11};

```



## 6.11 Fabric

**fabric-managedElement** NAME BINDING  
SUBORDINATE OBJECT CLASS **fabric**  
AND SUBCLASSES;  
NAMED BY  
SUPERIOR OBJECT CLASS **managedElement** AND SUBCLASSES;  
WITH ATTRIBUTE **fabricId**;  
CREATE  
WITH-REFERENCE-OBJECT,  
WITH-AUTOMATIC-INSTANCE-NAMING;  
DELETE  
ONLY-IF-NO-CONTAINED-OBJECTS;  
REGISTERED AS {m3100NameBinding 12};

## 6.12 GTP

**gtp-fabric** NAME BINDING  
SUBORDINATE OBJECT CLASS **gtp**;  
NAMED BY  
SUPERIOR OBJECT CLASS **fabric**  
AND SUBCLASSES;  
WITH ATTRIBUTE **gtpId**;  
REGISTERED AS {m3100NameBinding 13};

## 6.13 Log

**log-managedElement** NAME BINDING  
SUBORDINATE OBJECT CLASS **"Recommendation X.721:1992":log**;  
NAMED BY  
SUPERIOR OBJECT CLASS **managedElement** AND SUBCLASSES;  
WITH ATTRIBUTE **"Recommendation X.721:1992":logId**;  
CREATE  
WITH-REFERENCE-OBJECT,  
WITH-AUTOMATIC-INSTANCE-NAMING;  
DELETE  
ONLY-IF-NO-CONTAINED-OBJECTS;  
REGISTERED AS {m3100NameBinding 14};

## 6.14 Managed Element

**managedElement-network** NAME BINDING  
SUBORDINATE OBJECT CLASS **managedElement** AND SUBCLASSES;  
NAMED BY  
SUPERIOR OBJECT CLASS **network**;  
WITH ATTRIBUTE **managedElementId**;  
BEHAVIOUR  
**managedElementCreateBehaviour** BEHAVIOUR  
DEFINED AS

“Managed Element object is not created or deleted by system management protocol. The object is created when initializing the managed element.”;;

REGISTERED AS {m3100NameBinding 15};

**managedElement-organization** NAME BINDING  
SUBORDINATE OBJECT CLASS **managedElement** AND SUBCLASSES;  
NAMED BY SUPERIOR OBJECT CLASS  
**"CCITT X.521": organization** AND SUBCLASSES;  
WITH ATTRIBUTE **managedElementId**;  
BEHAVIOUR **managedElement-organizationBeh**;  
REGISTERED AS {m3100NameBinding 27};

**managedElement-organizationBeh BEHAVIOUR**  
**DEFINED AS**

“This name binding is used to name the managedElement object to an organization directory object. The managedElement object is not created or deleted by system management protocol. The managed element object is created when initializing the network element.”;

**managedElement-organizationalUnit NAME BINDING**  
**SUBORDINATE OBJECT CLASS** managedElement **AND SUBCLASSES;**  
**NAMED BY SUPERIOR OBJECT CLASS**  
"CCITT X.521": organizationalUnit **AND SUBCLASSES;**  
**WITH ATTRIBUTE** managedElementId;  
**BEHAVIOUR** managedElement-organizationalUnitBeh;  
**REGISTERED AS** {m3100NameBinding 28};

**managedElement-organizationalUnitBeh BEHAVIOUR**  
**DEFINED AS**

“This name binding is used to name the managedElement object to an organizationalUnit directory object. The managedElement object is not created or deleted by system management protocol. The managed element object is created when initializing the network element.”;

**managedElement-managedElementComplex NAME BINDING**  
**SUBORDINATE OBJECT CLASS** managedElement **AND SUBCLASSES;**  
**NAMED BY**  
**SUPERIOR OBJECT CLASS** managedElementComplex **AND SUBCLASSES;**  
**WITH ATTRIBUTE** managedElementId;  
**BEHAVIOUR**  
managedElementCreateBehaviour **BEHAVIOUR**  
**DEFINED AS**

“Managed Element object is not created or deleted by system management protocol. The object is created when initializing the managed element.”;

**REGISTERED AS** {m3100NameBinding 34};

## **6.15 Managed Element Complex**

**managedElementComplex-organization NAME BINDING**  
**SUBORDINATE OBJECT CLASS** managedElementComplex **AND SUBCLASSES;**  
**NAMED BY SUPERIOR OBJECT CLASS**  
"CCITT X.521": organization **AND SUBCLASSES;**  
**WITH ATTRIBUTE** managedElementComplexId;  
**BEHAVIOUR** managedElementComplex-organizationBeh;  
**REGISTERED AS** {m3100NameBinding 35};

**managedElementComplex-organizationBeh BEHAVIOUR**  
**DEFINED AS**

“This name binding is used to name the managedElementComplex object to an organization directory object. The managedElementComplex object is not created or deleted by system management protocol.”;

**managedElementComplex-organizationalUnit NAME BINDING**  
**SUBORDINATE OBJECT CLASS** managedElementComplex **AND SUBCLASSES;**  
**NAMED BY SUPERIOR OBJECT CLASS**  
"CCITT X.521": organizationalUnit **AND SUBCLASSES;**  
**WITH ATTRIBUTE** managedElementComplexId;  
**BEHAVIOUR** managedElementComplex-organizationalUnitBeh;  
**REGISTERED AS** {m3100NameBinding 36};

**managedElementComplex-organizationalUnitBeh BEHAVIOUR**  
**DEFINED AS**

“This name binding is used to name the managedElementComplex object to an organizationalUnit directory object. The managedElementComplex object is not created or deleted by system management protocol.”;

## 6.16 Multi-Point Cross-Connection

**mpCrossConnection-fabric NAME BINDING**  
SUBORDINATE OBJECT CLASS **mpCrossConnection**  
AND SUBCLASSES;  
NAMED BY  
SUPERIOR OBJECT CLASS **fabric**  
AND SUBCLASSES;  
WITH ATTRIBUTE **mpCrossConnectionId**;  
REGISTERED AS {m3100NameBinding 16};

## 6.17 Network

**network-network NAME BINDING**  
SUBORDINATE OBJECT CLASS **network** AND SUBCLASSES;  
NAMED BY  
SUPERIOR OBJECT CLASS **network** AND SUBCLASSES;  
WITH ATTRIBUTE **networkId**;  
BEHAVIOUR  
**networkCreateBehaviour** BEHAVIOUR  
DEFINED AS

“Network object is not created or deleted by system management protocol. The object is created when initializing the network.”;

REGISTERED AS {m3100NameBinding 17};

**network-organization NAME BINDING**  
SUBORDINATE OBJECT CLASS **network** AND SUBCLASSES;  
NAMED BY SUPERIOR OBJECT CLASS  
“CCITT X.521”: **organization** AND SUBCLASSES;  
WITH ATTRIBUTE **networkId**;  
BEHAVIOUR **network-organizationBeh**;  
REGISTERED AS {m3100NameBinding 29};

**network-organizationBeh BEHAVIOUR**  
DEFINED AS

“This name binding is used to name the network object to an organization directory object. The network object is not created or deleted by system management protocol. The object is created when initializing the network.”;

**network-organizationalUnit NAME BINDING**  
SUBORDINATE OBJECT CLASS **network** AND SUBCLASSES;  
NAMED BY SUPERIOR OBJECT CLASS  
“CCITT X.521”: **organizationalUnit** AND SUBCLASSES;  
WITH ATTRIBUTE **networkId**;  
BEHAVIOUR **network-organizationalUnitBeh**;  
REGISTERED AS {m3100NameBinding 30};

**network-organizationalUnitBeh BEHAVIOUR**  
DEFINED AS

“This name binding is used to name the network object to an organizationalUnit directory object. The network object is not created or deleted by system management protocol. The object is created when initializing the network.”;

## 6.18 Software

**software-equipment NAME BINDING**  
SUBORDINATE OBJECT CLASS **software** AND SUBCLASSES;  
NAMED BY  
SUPERIOR OBJECT CLASS **equipment** AND SUBCLASSES;  
WITH ATTRIBUTE **softwareId**;  
BEHAVIOUR  
**softwareNameBindingBehaviour**;  
CREATE  
WITH-REFERENCE-OBJECT,  
WITH-AUTOMATIC-INSTANCE-NAMING;

**DELETE**  
**ONLY-IF-NO-CONTAINED-OBJECTS;**

**REGISTERED AS {m3100NameBinding 18};**

**software-software NAME BINDING**  
**SUBORDINATE OBJECT CLASS** software **AND SUBCLASSES;**  
**NAMED BY**  
**SUPERIOR OBJECT CLASS** software **AND SUBCLASSES;**  
**WITH ATTRIBUTE** softwareId;  
**BEHAVIOUR**  
softwareNameBindingBehaviour;  
**CREATE**  
**WITH-REFERENCE-OBJECT,**  
**WITH-AUTOMATIC-INSTANCE-NAMING;**  
**DELETE**  
**ONLY-IF-NO-CONTAINED-OBJECTS;**

**REGISTERED AS {m3100NameBinding 19};**

**software-managedElement NAME BINDING**  
**SUBORDINATE OBJECT CLASS** software **AND SUBCLASSES;**  
**NAMED BY**  
**SUPERIOR OBJECT CLASS** managedElement **AND SUBCLASSES;**  
**WITH ATTRIBUTE** softwareId;  
**BEHAVIOUR**  
softwareNameBindingBehaviour;  
**CREATE**  
**WITH-REFERENCE-OBJECT,**  
**WITH-AUTOMATIC-INSTANCE-NAMING;**  
**DELETE**  
**ONLY-IF-NO-CONTAINED-OBJECTS;**

**REGISTERED AS {m3100NameBinding 20};**

**softwareNameBindingBehaviour**  
**BEHAVIOUR**  
**DEFINED AS**

“When automatic instance naming is used, the choice of name binding is left as a local matter.”;

## **6.19 TP Pool**

**tpPool-fabric NAME BINDING**  
**SUBORDINATE OBJECT CLASS** tpPool;  
**NAMED BY**  
**SUPERIOR OBJECT CLASS** fabric  
**AND SUBCLASSES;**  
**WITH ATTRIBUTE** tpPoolId;

**REGISTERED AS {m3100NameBinding 21};**

## **6.20 TrailR1**

**trailR1-network NAME BINDING**  
**SUBORDINATE OBJECT CLASS** trailR1;  
**NAMED BY**  
**SUPERIOR OBJECT CLASS** network;  
**WITH ATTRIBUTE** trailId;  
**CREATE**  
**WITH-REFERENCE-OBJECT,**  
**WITH-AUTOMATIC-INSTANCE-NAMING;**  
**DELETE**  
**ONLY-IF-NO-CONTAINED-OBJECTS;**

**REGISTERED AS {m3100NameBinding 26};**

## 6.21 Trail Termination Point Source

```
trailTerminationPointSource-managedElement NAME BINDING
  SUBORDINATE OBJECT CLASS      trailTerminationPointSource AND SUBCLASSES;
  NAMED BY
  SUPERIOR OBJECT CLASS      managedElement AND SUBCLASSES;
  WITH ATTRIBUTE      tPId;
  BEHAVIOUR
  trailTerminationPointNameBindingBehaviour;
  CREATE
    WITH-REFERENCE-OBJECT,
    WITH-AUTOMATIC-INSTANCE-NAMING;
  DELETE
    ONLY-IF-NO-CONTAINED-OBJECTS;

REGISTERED AS {m3100NameBinding 23};

trailTerminationPointNameBindingBehaviour
  BEHAVIOUR
  DEFINED AS
```

“When automatic instance naming is used, the choice of name binding is left as a local matter.”;

## 6.22 Trail Termination Point Sink

```
trailTerminationPointSink-managedElement NAME BINDING
  SUBORDINATE OBJECT CLASS      trailTerminationPointSink AND SUBCLASSES;
  NAMED BY
  SUPERIOR OBJECT CLASS      managedElement AND SUBCLASSES;
  WITH ATTRIBUTE      tPId;
  BEHAVIOUR
  trailTerminationPointNameBindingBehaviour;
  CREATE
    WITH-REFERENCE-OBJECT,
    WITH-AUTOMATIC-INSTANCE-NAMING;
  DELETE
    ONLY-IF-NO-CONTAINED-OBJECTS;

REGISTERED AS {m3100NameBinding 24};
```

# 7 Actions

## 7.1 Add TPs To GTP

```
addTpsToGTP ACTION
  BEHAVIOUR
  addTpsToGtpBehaviour BEHAVIOUR
  DEFINED AS
```

“This action is used to arrange termination points into GTPs. If the group termination point instance does not exist then a new one is automatically created and its identity returned in the result. Otherwise the termination points are added to those already in the GTP. Termination points may be members of zero or one GTP. This action will fail if the GTP is involved in a cross-connection, if the GTP is a member of a TP pool or if the termination point is already a member of a GTP. A bidirectional termination point that can provide independent unidirectional connectivity may be a member of zero or one GTP for each direction of connectivity.”;

```
MODE CONFIRMED;
WITH INFORMATION SYNTAX ASN1DefinedTypesModule.AddTpsToGtpInformation;
WITH REPLY SYNTAX      ASN1DefinedTypesModule.AddTpsToGtpResult;
```

```
REGISTERED AS {m3100Action 1};
```

## 7.2 Add TPs To TP Pool

```
addTpsToTpPool ACTION
  BEHAVIOUR
  addTpsToTpPoolBehaviour BEHAVIOUR
```

## DEFINED AS

“This action is used to arrange termination points or GTPs into pools of termination points or GTPs that are all equivalent for some management purpose such as routing. If the tpPool instance does not exist then a new one is automatically created and its identity returned in the result. Otherwise the termination points or GTPs are added to those already in the tpPool. If an Indirect Adaptor is specified, a GTP representing the CTPs contained from the Indirect Adaptor will be created and it will be added to the tpPool.”;

**MODE**     **CONFIRMED;**

**WITH INFORMATION SYNTAX**   **ASN1DefinedTypesModule.AddTpsToTpPoolInformation ;**

**WITH REPLY SYNTAX**         **ASN1DefinedTypesModule.AddTpsToTpPoolResult;**

**REGISTERED AS** {m3100Action 2};

## 7.3        **Allow Audible Visual Local Alarm**

**allowAudibleVisualLocalAlarm** **ACTION**

**BEHAVIOUR**   **allowAudibleVisualLocalAlarmBehaviour;**

**REGISTERED AS** {m3100Action 3};

**allowAudibleVisualLocalAlarmBehaviour** **BEHAVIOUR**

**DEFINED AS**

      “The allow Audible Visual Local Alarm action allows a managed system to present audible and/or visual indications.”;

## 7.4        **Connect**

**connect** **ACTION**

**BEHAVIOUR**

**connectBehaviour** **BEHAVIOUR**

**DEFINED AS**

      “This action is used to establish a connection between termination points or GTPs. The termination points to be connected can be specified in one of two ways:

- 1)by explicitly identifying the two termination points or GTPs;
- 2)by specifying one termination point or GTP, and specifying a tpPool from which any idle termination point/GTP may be used. The result, if successful, always returns an explicit list of termination points or GTP.

There are two basic forms of cross-connection arrangement: point-to-point and point to multi-point (broadcast). A single cross-connection is created if either the explicitPtoP or ptoTpPool option is selected in this action. This cross-connection object points to the termination points or GTPs involved in the cross-connection. Connections are indicated in termination points by the connectivityPointer attribute. If the administrativeState in the crossConnection object is unlocked, this attribute is set, as a result of this action, to the local name of the termination point to which it is connected. Also, the crossConnectionObjectPointer in the termination points or GTPs points to the cross-connection object.

For point to multi-point cross-connection (indicated by choosing the explicitPtoMp or ptoMPools option), one multi-point cross-connection object will be created containing one crossConnection object for each termination point specified in the toTps parameter. In the source TP the crossConnectionObjectPointer will point at the newly created Multi-point cross-connect object. In each Tp named in the toTPs list (possibly selected from a specified tpPool), the CrossConnectionObject pointer will point at the corresponding cross-connection object. The connectivity pointers in the connected termination points will be updated to reflect the new connectivity.

The idleTPcount and the connectedTPcount attributes in the tpPool object (if any) are updated as a result of the action. If a GTP is implicitly defined by specifying several termination points to be connected together, the GTP object will be automatically created and its id will be returned in the action reply.

If an Indirect Adaptor is specified, a GTP representing the CTPs contained from the Indirect Adaptor will be created and it will be connected.

The administrative state of the created cross-connection or multi-point cross-connection objects is specified as an optional parameter of this action. If this parameter is omitted, the administrative state will be set to 'unlocked' (unless the addLegs parameter is specified). This action will fail if any of the termination points specified is already involved in a cross-connection or if part of an existing GTP is specified.

If the addLeg parameter is specified, one or more Legs will be added to an existing multi-point cross-connection arrangement. Selected termination points or GTPs must support a similar signal type to that of the termination points already connected to the arrangement. The result, if successful, always returns the termination points or GTPs involved in the multi-point cross-connection. A cross-connection object is created as a result of this action. This object will be named from the specified mpCrossConnection object instance. The administrative state of the created cross-connection object will be the same as that of the containing multi-point cross-connection object unless otherwise specified in the action parameters.”;

**MODE CONFIRMED;**

**WITH INFORMATION SYNTAX ASN1DefinedTypesModule.ConnectInformation;**

**WITH REPLY SYNTAX ASN1DefinedTypesModule.ConnectResult;**

**REGISTERED AS {m3100Action 4};**

## 7.5 Disconnect

**disconnect ACTION**

**BEHAVIOUR**

**disconnectBehaviour BEHAVIOUR**

**DEFINED AS**

“This action is used to take down a cross-connection. The connection to be taken down is specified by identifying termination point(s) [or GTP(s)] of the connection. If the connection was point-to-point, then the other termination point or GTP is implicitly disconnected as well and the cross-connection object is deleted. If the connection was point to multi-point and the action referred to the master, all the termination points or GTPs that are legs are implicitly disconnected as well and the multi-point cross-connection and cross-connection objects are deleted.

If the connection was point-to-multi-point and the action referred to a leg, then just that leg is disconnected, unless it is the last leg, in which case the master termination point is also implicitly disconnected and the multipoint cross-connection and cross-connection objects are deleted. The idleTPcount and the connectedTPcount attributes in the tpPool objects (if any) are updated as a result of the action. The connectivity pointers in the disconnected termination points will be set to NULL as a result of this action.

This action has no effect on the composition of GTPs and GTPs are not deleted as a result of this action. This action will fail if part of a GTP is specified.”;

**MODE CONFIRMED;**

**WITH INFORMATION SYNTAX ASN1DefinedTypesModule.DisconnectInformation;**

**WITH REPLY SYNTAX ASN1DefinedTypesModule.DisconnectResult;**

**REGISTERED AS {m3100Action 5};**

## 7.6 Inhibit Audible Visual Local Alarm

**inhibitAudibleVisualLocalAlarm ACTION**

**BEHAVIOUR inhibitAudibleVisualLocalAlarmBehaviour;**

**REGISTERED AS {m3100Action 6};**

**inhibitAudibleVisualLocalAlarmBehaviour BEHAVIOUR**

**DEFINED AS**

“The Inhibit Audible/Visual Local Alarm action inhibits a managed system from presenting audible and/or visual indications.”;

## 7.7 Remove TPs From GTP

**removeTpsFromGTP ACTION**

**BEHAVIOUR**

**removeTpsFromGtpBehaviour BEHAVIOUR**

## **DEFINED AS**

“This action is used to remove termination points from GTPs. This action will fail if the GTP is involved in a cross-connection or if it is a member of a TP pool. Removing the last termination point from a GTP has the effect of deleting the GTP object. If the GTP is deleted, the name of the GTP will be sent back in the ACTION reply.”;

**MODE** **CONFIRMED;**

**WITH INFORMATION SYNTAX** **ASN1DefinedTypesModule.RemoveTpsFromGtpInformation;**

**WITH REPLY SYNTAX** **ASN1DefinedTypesModule.RemoveTpsFromGtpResult;**

**REGISTERED AS** {m3100Action 7};

## **7.8 Remove TPs From TP Pool**

**removeTpsFromTpPool ACTION**

**BEHAVIOUR**

**removeTpsFromTpPoolBehaviour BEHAVIOUR**

**DEFINED AS**

“This action is used to remove termination points from termination point pools. Removing the last termination point from a pool has the effect of deleting the TP Pool object. If the TP pool is deleted, the name of the TP Pool will be sent back in the ACTION reply.”;

**MODE** **CONFIRMED;**

**WITH INFORMATION SYNTAX** **ASN1DefinedTypesModule.RemoveTpsFromTpPoolInformation;**

**WITH REPLY SYNTAX** **ASN1DefinedTypesModule.RemoveTpsFromTpPoolResult;**

**REGISTERED AS** {m3100Action 8};

## **7.9 Reset Audible Alarm**

This action is defined in Recommendation Q.821[7].

## **7.10 Switch Over**

**switchOver ACTION**

**BEHAVIOUR**

**switchOverBehaviour BEHAVIOUR**

**DEFINED AS**

“This action offers the capability, in an atomic way, to: Switch an existing connection to another one of the same type maintaining one of the original cross-connected termination points. If the operation succeeds, this will result in the deletion of the old connection indicated in the action information and the connection of the new termination point. The new termination point indicated in the action information must be available (disconnected for the respective direction) in order to establish the new cross-connection. The switch over of an individual connection is considered an atomic operation.

Switch a bundle of existing connections. Each of these connections are switched over as described above. In this case, the action will behave according to a best effort policy and only those connections which can be successfully switched over will be performed independently of each other.

In any case, a connection designates a unidirectional or bidirectional point-to-point connection (i.e. crossConnection), a leg of a multi-point connection (i.e. crossConnection contained in a mpCrossConnection), or a leg of a multipoint connection protection defined in Recommendation G.774.04.”;

**MODE** **CONFIRMED;**

**WITH INFORMATION SYNTAX** **ASN1DefinedTypesModule.SwitchOverInformation;**

**WITH REPLY SYNTAX** **ASN1DefinedTypesModule.SwitchOverResult;**

**REGISTERED AS** { m3100Action 9};

# **8 Notifications**

## **8.1 Attribute Value Change**

This notification type is used to report when there is a change in some of the attribute values of a managed object. It is defined in Recommendation X.721[5].



## 8.2 Communications Alarm

This notification type is used to report when the managed object detects a communication error. It is defined in Recommendation X.721[5].

## 8.3 Environmental Alarm

This notification type is used to report when the managed object detects a problem in the environment. It is defined in Recommendation X.721[5].

## 8.4 Equipment Alarm

This notification type is used to report a failure in the equipment. It is defined in Recommendation X.721[5].

## 8.5 Object Creation

This notification type is used to report the creation of a managed object if defined in the managed object class specification. It is defined in Recommendation X.721[5].

## 8.6 Object Deletion

This notification type is used to report the deletion of a managed object if defined in the managed object class specification. It is defined in Recommendation X.721[5].

## 8.7 Processing Error Alarm

This notification type is used to report a processing failure in a managed object. It is defined in Recommendation X.721[5].

## 8.8 State Change

This notification type is used to report when there is a change in some of the state values of a managed object. It is defined in Recommendation X.721[5].

# 9 Parameters

## 9.1 Alarm Effect On Service Parameter

**alarmEffectOnServiceParameter** PARAMETER

CONTEXT EVENT-INFO;

WITH SYNTAX ASN1DefinedTypesModule.AlarmEffectOnServiceParameter;

BEHAVIOUR

**alarmEffectOnServiceParameterBehaviour** BEHAVIOUR  
DEFINED AS

“The **alarmEffectOnServiceParameter** is a parameter to be included as an element of a set in the **AdditionalInformation** parameter of the **AlarmInfo** defined in Recommendation X.721. The **alarmEffectOnServiceParameter** indicates whether the service is affected by the alarm.”;

REGISTERED AS {m3100Parameter 1};

## 9.2 Create Error

**createErrorParameter** PARAMETER

CONTEXT SPECIFIC-ERROR;

WITH SYNTAX ASN1DefinedTypesModule.CreateError;

BEHAVIOUR

**createErrorParameterBehaviour** BEHAVIOUR

## DEFINED AS

“If the maximum number of instances of the object class exist within the superior (containing) managed object, attempts to create additional instances will result in the CMIP processing failure error. The syntax of the specific error is an integer defined by CreateError. The value of the integer indicates the number of instances that are currently contained in the superior object.”;

REGISTERED AS {m3100Parameter 2};

## 10 ASN.1 defined types module

### 10.1 Rules of Extensibility

The following types will be indicated as being extensible:

- ENUMERATED;
- named INTEGER;
- named BIT STRING;
- tagged SET;
- tagged SEQUENCE;
- tagged CHOICE.

Under the rules of extensibility new enumerations (for ENUMERATED types), new bit name assignments (for named BIT STRING types), new named numbers (for named INTEGER types), and new tagged elements (for tagged SET, SEQUENCE, and CHOICE types) may be added in future versions of this Recommendation.

When processing information in a System Management Application Protocol (SMAP) PDU, the accepting SMAP-machine shall ignore:

- enumerations not recognized;
- unrecognized named numbers;
- unrecognized named bits;
- unrecognized tagged elements of sets, sequences, and choices.

### 10.2 ASN.1 Module

ASN1DefinedTypesModule {ccitt recommendation m gnm(3100) informationModel(0) asn1Modules(2)  
asn1DefinedTypesModule(0)}

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

-- EXPORTS everything

IMPORTS

RDNSSequence

FROM InformationFramework {joint-iso-ccitt ds(5) modules(1) informationFramework(1)}

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

ProbableCause, AdministrativeState, AvailabilityStatus FROM Attribute-ASN1Module {joint-iso-ccitt ms(9) smi(3) part2(2)  
asn1Module (2) 1} ;

m3100InformationModel OBJECT IDENTIFIER ::= {ccitt recommendation m gnm(3100) informationModel(0) }

m3100standardSpecificExtension OBJECT IDENTIFIER ::= {m3100InformationModel standardSpecificExtension(0)}

m3100ObjectClass OBJECT IDENTIFIER ::= {m3100InformationModel managedObjectClass(3)}

m3100Package OBJECT IDENTIFIER ::= {m3100InformationModel package(4)}

m3100Parameter OBJECT IDENTIFIER ::= {m3100InformationModel package(5)}

m3100Attribute OBJECT IDENTIFIER ::= {m3100InformationModel attribute(7)}

m3100NameBinding OBJECT IDENTIFIER ::= {m3100InformationModel nameBinding(6)}

m3100Action OBJECT IDENTIFIER ::= {m3100InformationModel action(9)}

m3100Notification OBJECT IDENTIFIER ::= {m3100InformationModel notification(10)}

-- Reserved arcs below m3100InformationModel are (8) for attribute groups

```

characteristicInfo OBJECT IDENTIFIER ::= {m3100standardSpecificExtension 0}

opticalSTM1SPICI CharacteristicInformation ::= {characteristicInfo 1}
-- opticalSPITTP* object instances with stmLevel attribute = 1

opticalSTM4SPICI CharacteristicInformation ::= {characteristicInfo 2}
-- opticalSPITTP* object instances with stmLevel attribute = 4

opticalSTM16SPICI CharacteristicInformation ::= {characteristicInfo 3}
-- opticalSPITTP* object instances with stmLevel attribute = 16

electricalSTM1SPICI CharacteristicInformation ::= {characteristicInfo 4}
-- electricalSPITTP* object instances with stmLevel attribute = 1

rsSTM1SPICI CharacteristicInformation ::= {characteristicInfo 5}
-- rsCTP* object instances with stmLevel attribute = 1

rsSTM4SPICI CharacteristicInformation ::= {characteristicInfo 6}
-- rsCTP* object instances with stmLevel attribute = 4

rsSTM16SPICI CharacteristicInformation ::= {characteristicInfo 7}
-- rsCTP* object instances with stmLevel attribute = 16

msSTM1SPICI CharacteristicInformation ::= {characteristicInfo 8}
-- msCTP* object instances with stmLevel attribute = 1

msSTM4SPICI CharacteristicInformation ::= {characteristicInfo 9}
-- msCTP* object instances with stmLevel attribute = 4

msSTM16SPICI CharacteristicInformation ::= {characteristicInfo 10}
-- msCTP* object instances with stmLevel attribute = 16

au3TU3VC3CI CharacteristicInformation ::= {characteristicInfo 11}

au4VC4CI CharacteristicInformation ::= {characteristicInfo 12}

tu11VC11CI CharacteristicInformation ::= {characteristicInfo 13}

tu12VC12CI CharacteristicInformation ::= {characteristicInfo 14}

tu2VC2CI CharacteristicInformation ::= {characteristicInfo 15}

tu12VC11CI CharacteristicInformation ::= {characteristicInfo 16}

-- The following value assignments are for the Probable Cause when Integer Choice is used within the
-- TMN application context. These values shall always be assigned by this Recommendation in the context
-- of TMN.
indeterminate ProbableCause ::= localValue : 0
-- The following are used with communications alarm.
aIS ProbableCause ::= localValue : 1
callSetUpFailure ProbableCause ::= localValue : 2
degradedSignal ProbableCause ::= localValue : 3
farEndReceiverFailure ProbableCause ::= localValue : 4
framingError ProbableCause ::= localValue : 5
lossOffFrame ProbableCause ::= localValue : 6
lossOfPointer ProbableCause ::= localValue : 7
lossOfSignal ProbableCause ::= localValue : 8
payloadTypeMismatch ProbableCause ::= localValue : 9
transmissionError ProbableCause ::= localValue : 10
remoteAlarmInterface ProbableCause ::= localValue : 11
excessiveBER ProbableCause ::= localValue : 12
pathTraceMismatch ProbableCause ::= localValue : 13
unavailable ProbableCause ::= localValue : 14
signalLabelMismatch ProbableCause ::= localValue : 15
lossOfMultiFrame ProbableCause ::= localValue : 16
-- Values 17-50 are reserved for communications alarm related probable causes

-- The following are used with equipment alarm.
backplaneFailure ProbableCause ::= localValue : 51
dataSetProblem ProbableCause ::= localValue : 52
equipmentIdentifierDuplication ProbableCause ::= localValue : 53
externalIFDeviceProblem ProbableCause ::= localValue : 54

```

**lineCardProblem ProbableCause ::= localValue : 55**  
**multiplexerProblem ProbableCause ::= localValue : 56**  
**nEIdentifierDuplication ProbableCause ::= localValue : 57**  
**powerProblem ProbableCause ::= localValue : 58**  
**processorProblem ProbableCause ::= localValue : 59**  
**protectionPathFailure ProbableCause ::= localValue : 60**  
**receiverFailure ProbableCause ::= localValue : 61**  
**replaceableUnitMissing ProbableCause ::= localValue : 62**  
**replaceableUnitTypeMismatch ProbableCause ::= localValue : 63**  
**synchronizationSourceMismatch ProbableCause ::= localValue : 64**  
**terminalProblem ProbableCause ::= localValue : 65**  
**timingProblem ProbableCause ::= localValue : 66**  
**transmitterFailure ProbableCause ::= localValue : 67**  
**trunkCardProblem ProbableCause ::= localValue : 68**  
**replaceableUnitProblem ProbableCause ::= localValue : 69**  
*-- Values 70-100 are reserved for equipment alarm related probable causes*

*-- The following are used with environmental alarm.*

**airCompressorFailure ProbableCause ::= localValue : 101**  
**airConditioningFailure ProbableCause ::= localValue : 102**  
**airDryerFailure ProbableCause ::= localValue : 103**  
**batteryDischarging ProbableCause ::= localValue : 104**  
**batteryFailure ProbableCause ::= localValue : 105**  
**commercialPowerFailure ProbableCause ::= localValue : 106**  
**coolingFanFailure ProbableCause ::= localValue : 107**  
**engineFailure ProbableCause ::= localValue : 108**  
**fireDetectorFailure ProbableCause ::= localValue : 109**  
**fuseFailure ProbableCause ::= localValue : 110**  
**generatorFailure ProbableCause ::= localValue : 111**  
**lowBatteryThreshold ProbableCause ::= localValue : 112**  
**pumpFailure ProbableCause ::= localValue : 113**  
**rectifierFailure ProbableCause ::= localValue : 114**  
**rectifierHighVoltage ProbableCause ::= localValue : 115**  
**rectifierLowFVvoltage ProbableCause ::= localValue : 116**  
**ventilationsSystemFailure ProbableCause ::= localValue : 117**  
**enclosureDoorOpen ProbableCause ::= localValue : 118**  
**explosiveGas ProbableCause ::= localValue : 119**  
**fire ProbableCause ::= localValue : 120**  
**flood ProbableCause ::= localValue : 121**  
**highHumidity ProbableCause ::= localValue : 122**  
**highTemperature ProbableCause ::= localValue : 123**  
**highWind ProbableCause ::= localValue : 124**  
**iceBuildUp ProbableCause ::= localValue : 125**  
**intrusionDetection ProbableCause ::= localValue : 126**  
**lowFuel ProbableCause ::= localValue : 127**  
**lowHumidity ProbableCause ::= localValue : 128**  
**lowCablePressure ProbableCause ::= localValue : 129**  
**lowTemperatue ProbableCause ::= localValue : 130**  
**lowWater ProbableCause ::= localValue : 131**  
**smoke ProbableCause ::= localValue : 132**  
**toxicGas ProbableCause ::= localValue : 133**  
*-- Values 134-150 are reserved for environmental alarm related probable causes*

*-- The following are used with Processing error alarm.*

**storageCapacityProblem ProbableCause ::= localValue : 151**  
**memoryMismatch ProbableCause ::= localValue : 152**  
**corruptData ProbableCause ::= localValue : 153**  
**outOfCPUCycles ProbableCause ::= localValue : 154**  
**sfwrEnvironmentProblem ProbableCause ::= localValue : 155**  
**sfwrDownloadFailure ProbableCause ::= localValue : 156**

*-- Service ProblemType is for further study*

**AcceptableCircuitPackTypeList ::= SET OF PrintableString**

```

AddedTps ::= SEQUENCE {
    gtp      ObjectInstance,
    tpsAdded SEQUENCE OF ObjectInstance
}

```

```

AddLeg ::= SEQUENCE {
    mpCrossConnection ObjectInstance,
    legs              SET OF ToTermSpecifier
}

```

```

AddTpsToGtpInformation ::= SEQUENCE OF SEQUENCE {
    tpsAdded SEQUENCE OF TerminationPointInformation,
    gtp      ObjectInstance OPTIONAL
}

```

```

AddTpsToGtpResult ::= SEQUENCE OF CHOICE {
    failed [0] Failed,
    addedTps [1] AddedTps
}

```

-- the n-th element in the "SEQUENCE OF" is related to the n-th element in the "SEQUENCE OF" of the  
-- "AddTpsToGtpInformation" type.

```

AddTpsToTpPoolInformation ::= SEQUENCE OF SEQUENCE {
    tps      SET OF TerminationPointInformation,
    toTpPool ObjectInstance OPTIONAL
}

```

```

AddTpsToTpPoolResult ::= SEQUENCE OF CHOICE {
    failed [0] Failed,
    tpsAddedToTpPool [1] TpsAddedToTpPool
}

```

-- the n-th element in the "SEQUENCE OF" is related to the n-th element in the "SEQUENCE OF" of the  
-- "AddTpsToTpPoolInformation" type.

AlarmEffectOnServiceParameter ::= BOOLEAN -- TRUE implies service is affected

```

AlarmSeverityAssignment ::= SEQUENCE {
    problem                      ProbableCause,
    severityAssignedServiceAffecting [0]AlarmSeverityCode OPTIONAL,
    severityAssignedNonServiceAffecting [1]AlarmSeverityCode OPTIONAL,
    severityAssignedServiceIndependent [2]AlarmSeverityCode OPTIONAL
}

```

AlarmSeverityAssignmentList ::= SET OF AlarmSeverityAssignment

```

AlarmSeverityCode ::= ENUMERATED {
    non-alarmed (0),
    minor (1),
    major (2),
    critical (3),
    warning (4) }

```

```

AlarmStatus ::= ENUMERATED {
    cleared (0),
    activeReportable-Indeterminate (1),
    activeReportable-Warning (2),
    activeReportable-Minor (3),
    activeReportable-Major (4),
    activeReportable-Critical (5),
    activePending (6)
}

```

Boolean ::= BOOLEAN

```

Bundle ::= SEQUENCE {
    characteristicInfoType CharacteristicInformation,
    bundlingFactor          INTEGER
}

```

ChannelNumber ::= INTEGER

CharacteristicInformation ::= OBJECT IDENTIFIER

CircuitDirectionality ::= ENUMERATED {onewayOut(0), onewayIn(1), twoway(2)}

```

CircuitPackAvailabilityStatus ::= AvailabilityStatus (WITH COMPONENT(notInstalled))

CircuitPackType ::= PrintableString

Connected ::= CHOICE {
    pointToPoint          [0] PointToPoint,
    pointToMultipoint     [1] PointToMultipoint
}

ConnectInformation ::= SEQUENCE OF SEQUENCE {
    CHOICE {
        unidirectional     [0] ConnectionType,
        bidirectional      [1] ConnectionTypeBi,
        addleg              [2] AddLeg
    },
    administrativeState AdministrativeState OPTIONAL,
    namedCrossConnection [3] NamedCrossConnection OPTIONAL
}

ConnectivityPointer ::= CHOICE { none      NULL,
                                   single   ObjectInstance,
                                   concatenated SEQUENCE OF ObjectInstance}

ConnectResult ::= SEQUENCE OF CHOICE {
    failed    Failed,
    connected Connected
}
-- the n-th element in the "SEQUENCE OF" is related to the n-th element in the "SEQUENCE OF" of the
-- "ConnectInformation" type.

ConnectionType ::= CHOICE {
    explicitPToP      [0] ExplicitPtoP,
    ptoTpPool        [1] PtoTPPool,
    explicitPtoMP     [2] ExplicitPtoMP,
    ptoMPools         [3] PtoMPools
}

ConnectionTypeBi ::= CHOICE {
    explicitPToP [0] ExplicitPtoP,
    ptoTpPool   [1] PtoTPPool
}

Count ::= INTEGER

CreateError ::= INTEGER

CrossConnectionName ::= GraphicString

CrossConnectionObjectPointer ::= CHOICE {
    notConnected [0] ObjectInstance, -- Fabric object --
    connected    [1] ObjectInstance, -- Cross-connection object --
    multipleConnections MultipleConnections
}

CTPUpstreamPointer ::= ConnectivityPointer(WITH COMPONENTS { ...,
-- the other two choices are present
concatenated ABSENT})

CTPDownstreamPointer ::= DownstreamConnectivityPointer (WITH COMPONENTS
{...,
concatenated ABSENT,
broadcastConcatenated ABSENT
-- other choices are present
})

CurrentProblem ::= SEQUENCE {
    problem      [0] ProbableCause,
    alarmStatus  [1] AlarmStatus
}

CurrentProblemList ::= SET OF CurrentProblem

Directionality ::= ENUMERATED { unidirectional(0),
bidirectional(1) }

```

```

DisconnectInformation ::= SEQUENCE OF ObjectInstance -- tps

DisconnectResult ::= SEQUENCE OF CHOICE {
    failed      Failed,
    disconnected  ObjectInstance      -- tp
}
-- the n-th element in the "SEQUENCE OF" is related to the n-th element in the "SEQUENCE OF" of the
-- "DisconnectInformation" type.

DownstreamConnectivityPointer ::= CHOICE {
    none      NULL,
    single    ObjectInstance,
    concatenated SEQUENCE OF ObjectInstance,
    broadcast  SET OF ObjectInstance,
    broadcastConcatenated [1] SET OF SEQUENCE OF ObjectInstance}

ExplicitPtoMP ::= SEQUENCE {
    fromTp    ExplicitTP,
    toTPs     SET OF ExplicitTP
}

ExplicitPtoP ::= SEQUENCE {
    fromTp    ExplicitTP,
    toTp      ExplicitTP
}

ExplicitTP ::= CHOICE {
    oneTPorGTP ObjectInstance,
    listofTPs  SEQUENCE OF ObjectInstance
}

ExternalTime ::= GeneralizedTime

EquipmentHolderAddress ::= SEQUENCE OF PrintableString

EquipmentHolderType ::= GraphicString

Failed ::= CHOICE {
    logicalProblem    LogicalProblem,
    resourceProblem   ResourceProblem}

HolderStatus ::= CHOICE {
    holderEmpty      [0] NULL,
    inTheAcceptableList [1] CircuitPackType,
    notInTheAcceptableList [2] CircuitPackType,
    unknownType      [3] NULL
}

IndividualSwitchOver ::= SEQUENCE {
    connection      ObjectInstance,
    -- crossConnection of connectionProtection (defined in Recommendation G.774.04)
    unchangedTP     ObjectInstance,
    newTP           ObjectInstance
    -- designates the newTP to which the unchanged TP has to be connected with
}

IndividualResult ::= CHOICE {
    failed [0] Failed,
    pass   [1] Connected}

InformationTransferCapabilities ::= ENUMERATED {speech(0),
audio3pt1(1),audio7(2),audioComb(3),digitalRestricted56(4), digitalUnrestricted64(5)
-- "..." these ellipses defined in ASN.1 amendment are used here to indicate that this is --
-- an extensible type and additional enumerations may be added in future -- }

ListOfCharacteristicInformation ::= SET OF CharacteristicInformation

ListOfTPs ::= SET OF ObjectInstance

LocationName ::= GraphicString

LogicalProblem ::= SEQUENCE {
    problemCause      ProblemCause,
    incorrectInstances SET OF ObjectInstance OPTIONAL}

```

```

MultipleConnections ::= SET OF CHOICE {
    downstreamNotConnected    [0] ObjectInstance,
    downstreamConnected        [1] ObjectInstance,
    upstreamNotConnected        [2] ObjectInstance,
    upstreamConnected           [3] ObjectInstance }

NamedCrossConnection ::= SEQUENCE {   redline  BOOLEAN,
                                         name     CrossConnectionName}

NameType ::=    CHOICE {
                                     numericName    INTEGER,
                                     pString    GraphicString
                                     }

NumberOfCircuits ::= INTEGER

ObjectList ::=    SET OF ObjectInstance

Pointer ::= ObjectInstance

PointerOrNull ::= CHOICE{
    pointer    ObjectInstance,
    null       NULL}

PointToPoint ::= SEQUENCE {
    fromTp    ObjectInstance,
    toTp      ObjectInstance,
    xCon      ObjectInstance
}

PointToMultipoint ::= SEQUENCE {
    fromTp    ObjectInstance,
    toTps     SET OF SEQUENCE {
        tp      ObjectInstance,
        xConnection    ObjectInstance
    },
    mpXCon    ObjectInstance
}

ProblemCause ::= CHOICE {
    unknown    NULL,
    integerValue    INTEGER}

-- The values of integer value for ProblemCause and integerValue for ResourceProblem shall always be
-- assigned by this Recommendation. No values of integerValue for ResourceProblem have been assigned.
-- The following values are used for integerValue of ProblemCause.

noSuchTpInstance           ProblemCause ::= integerValue : 0
noSuchGtpInstance          ProblemCause ::= integerValue : 1
noSuchTpPoolInstance       ProblemCause ::= integerValue : 2
mismatchingTpInstance      ProblemCause ::= integerValue : 3
mismatchingGtpInstance     ProblemCause ::= integerValue : 4
partOfGtp                  ProblemCause ::= integerValue : 5
involvedInCrossConnection  ProblemCause ::= integerValue : 6
memberOfTpPool             ProblemCause ::= integerValue : 7
alreadyMemberOfGtp         ProblemCause ::= integerValue : 8
noTpInTpPool               ProblemCause ::= integerValue : 9
noMoreThanOneTpIsAllowed   ProblemCause ::= integerValue : 10
noMoreThanTwoTpsAreAllowed ProblemCause ::= integerValue : 11
alreadyConnected            ProblemCause ::= integerValue : 12

-- alreadyConnected is used to indicate the two termination points requested to be cross-connected are already
-- cross-connected versus involvedInCrossConnection is used to indicate one or more termination points are
-- cross-connected but not to each other.

PtoMPools ::= SEQUENCE {
    fromTp    ExplicitTP,
    toTPPools ToTPPools
}

PtoTPPool ::= SEQUENCE {
    fromTp    ExplicitTP,
    toTpPool  ObjectInstance
}

```



```

RelatedObjectInstance ::= CHOICE {
    notAvailable NULL,
    relatedObject ObjectInstance
}

RemoveTpsFromGtpInformation ::= SEQUENCE OF SEQUENCE {
    fromGtp ObjectInstance,
    tps SET OF ObjectInstance
}

RemoveTpsFromGtpResult ::= SEQUENCE OF CHOICE {
    failed [0] Failed,
    removed [1] RemoveTpsResultInformation
}
-- the n-th element in the "SEQUENCE OF" is related to the n-th element in the "SEQUENCE OF" of the
-- "RemoveTPsFromGtpInformation" type.

RemoveTpsFromTpPoolInformation ::= SEQUENCE OF SEQUENCE {
    fromTpPool ObjectInstance,
    tps SET OF ObjectInstance
}

RemoveTpsFromTpPoolResult ::= SEQUENCE OF CHOICE {
    failed [0] Failed,
    removed [1] RemoveTpsResultInformation
}
-- the n-th element in the "SEQUENCE OF" is related to the n-th element in the "SEQUENCE OF" of the
-- "RemoveTPsFromTpPoolInformation" type.

RemoveTpsResultInformation ::= SEQUENCE {
    deletedTpPoolOrGTP ObjectInstance OPTIONAL,
    tps SET OF ObjectInstance
}
-- If the TP Pool or GTP is deleted, the deleted TP Pool or GTP should be provided in the
-- RemoveTpsResultInformation

Replaceable ::= ENUMERATED {
    yes (0),
    no (1),
    notapplicable (2)
}

ResourceProblem ::= CHOICE {
    unknown NULL,
    integerValue INTEGER
}

SequenceOfObjectInstance ::= SEQUENCE OF ObjectInstance
SerialNumber ::= GraphicString
SignallingCapabilities ::= ENUMERATED {isup(0), isup92(1),ccittNo5(2),r2(3),ccittNo6(4), tup(5)}
-- "... these ellipses defined in ASN.1 amendment are used here to indicate that this is --
-- an extensible type and additional enumerations may be added in future -- }

SignalType ::= CHOICE {
    simple CharacteristicInformation,
    bundle Bundle,
    complex [0] SEQUENCE OF Bundle
}

SubordinateCircuitPackSoftwareLoad ::= CHOICE {
    notApplicable NULL,
    softwareInstances [0] SEQUENCE OF ObjectInstance,
    softwareIdentifiers [1] SEQUENCE OF PrintableString
}

SupportableClientList ::= SET OF ObjectClass
SystemTiming ::= SEQUENCE {
    sourceType ENUMERATED { internalTimingSource(0),
        remoteTimingSource(1), slavedTimingTerminationSignal(2)},
    sourceID ObjectInstance OPTIONAL -- not needed for internal source
}

SystemTimingSource ::= SEQUENCE {
    primaryTimingSource SystemTiming,
    secondaryTimingSource SystemTiming OPTIONAL
}

```

**SwitchOverInformation ::= SEQUENCE OF IndividualSwitchOver**

**SwitchOverResult ::= SEQUENCE OF IndividualResult**

**TerminationPointInformation ::= CHOICE {**

<b>tPOrGTP</b>	<b>[0]</b>	<b>ObjectInstance,</b>
<b>sourceTP</b>	<b>[1]</b>	<b>ObjectInstance,</b>
<b>sinkTP</b>	<b>[2]</b>	<b>ObjectInstance}</b>

**ToTermSpecifier ::= CHOICE {**

<b>toTpOrGTP</b>	<b>[0]</b>	<b>ExplicitTP,</b>
<b>toPool</b>	<b>[1]</b>	<b>ObjectInstance</b>

**}**

**ToTPPools ::= SET OF SEQUENCE {**

<b>tpPoolId</b>	<b>ObjectInstance,</b>
<b>numberOfTPs</b>	<b>INTEGER</b>

**}**

**TpsAddedToTpPool ::= SEQUENCE {**

<b>tpPool</b>	<b>ObjectInstance,</b>
<b>tps</b>	<b>SET OF ObjectInstance</b>

**}**

**TpsInGtpList ::= SEQUENCE OF ObjectInstance**

**TransmissionCharacteristics ::= BIT STRING {satellite(0), dCME(1), echoControl(2)}**

**UserLabel ::= GraphicString**

**VendorName ::= GraphicString**

**Version ::= GraphicString**

**END -- end of ASN1 DefinedTypesModule**

## **11 TMN application context**

The object identifier value

{ccitt recommendation m(13) gnm(3100) protocolSupport(1) applicationContext(0) tmnApplicationContextOne(1)}

is assigned to the application context that has the same capabilities as the systems management application context defined in Recommendation X.701, but also supports the integer values for ProbableCause. These integer value assignments are specified in this Recommendation.

## **12 Entity-Relationship diagrams**

Figures 1-8 depict the various relationships between the managed object classes specified in this Recommendation. These figures are representative of the relationships and do not contain all the relationships. All the relationships can be determined from the templates in clause 3.

## Annex A

### Index

(This annex forms an integral part of this Recommendation)

#### A.1 Managed objects

Alarm Record .....	3.6.1
Alarm Severity Assignment Profile.....	3.6.2
Attribute Value Change Record .....	3.6.3
Circuit Pack .....	3.2.1
Circuit End Point Subgroup.....	3.4.1
Connection.....	Appendix II.1.3
ConnectionR1 .....	3.4.2
Connectivity .....	Appendix II.1.1
Connection Termination Point Bidirectional.....	3.3.1
Connection Termination Point Sink .....	3.3.2
Connection Termination Point Source .....	3.3.3
Cross-Connection .....	3.5.1
Current Alarm Summary Control.....	3.6.4
Discriminator.....	3.6.5
Equipment .....	3.2.2
Equipment Holder .....	3.2.3
Equipment R1 .....	3.2.4
Event Forwarding Discriminator .....	3.6.6
Event Log Record.....	3.6.7
Fabric.....	3.5.2
FabricR1 .....	3.5.3
Group Termination Point.....	3.5.4
Log .....	3.6.8
Log Record .....	3.6.9
Managed Element.....	3.2.5
Managed Element R1 .....	3.2.6
Managed Element Complex .....	3.2.7
Management Operations Schedule .....	3.6.10
Multi-Point Cross-Connection.....	3.5.5
Named Cross-Connection.....	3.5.6
Named Multi-Point Cross-Connection .....	3.5.7
Network .....	3.1.1
NetworkR1 .....	3.1.2
Object Creation Record .....	3.6.11
Object Deletion Record .....	3.6.12
Pipe .....	3.4.3
Software.....	3.2.8
Software R1 .....	3.2.9
State Change Record .....	3.6.13
Termination Point .....	3.3.4
TP Pool.....	3.5.8
Trail .....	Appendix II.1.2
Trail R1.....	3.4.4
Trail Termination Point Bidirectional .....	3.3.5
Trail Termination Point Sink.....	3.3.6
Trail Termination Point Source .....	3.3.7

## A.2 Packages

Administrative Operational States .....	4.1
Affected Object List .....	4.2
Alarm Severity Assignment Pointer .....	4.3
Attribute Value Change Notification .....	4.4
Audible Visual Local Alarm.....	4.5
Channel Number.....	4.6
Characteristic Information .....	4.7
Client Connection List.....	4.8
Client Trail.....	4.9
Create Delete Notifications .....	4.10
Cross-Connection Pointer.....	4.11
CTP Instance .....	4.12
Current Problem List .....	4.13
Environmental Alarm .....	4.14
Environmental AlarmR1.....	4.15
Equipment Alarm Effect on Service.....	4.16
Equipments Equipment Alarm .....	4.17
Equipments Equipment AlarmR1 .....	4.18
External Time .....	4.19
Location Name .....	4.20
Named Cross-Connection.....	4.21
Network Level.....	4.22
Operational State .....	4.23
Object Management Notifications.....	4.24
Processing Error Alarm .....	4.25
Processing Error AlarmR1 .....	4.26
Protected.....	4.27
Reset Audible Alarm .....	4.28
Server Connection List.....	4.29
Server Trail List.....	4.30
Software Processing Error Alarm.....	4.31
Software Processing Error AlarmR1 .....	4.32
Subordinate Circuit Pack.....	4.33
Supportable Client List.....	4.34
State Change Notification.....	4.35
System Timing Source.....	4.36
TMN Communications Alarm Information.....	4.37
TTP Instance.....	4.38
User Label .....	4.39
Vendor Name .....	4.40
Version .....	4.41

## A.3 Attributes

A-Termination Point Instance .....	5.1
Acceptable Circuit Pack Type List.....	5.2
Administrative State .....	5.3
Affected Object List .....	5.4
Alarm Severity Assignment List .....	5.5
Alarm Severity Assignment Profile Id .....	5.6
Alarm Severity Assignment Profile Pointer .....	5.7
Alarm Status .....	5.8
Channel Number.....	5.9
Characteristic Information .....	5.10
Circuit Directionality.....	5.11
Circuit End Point Subgroup Id .....	5.12
Circuit Pack Type.....	5.13
Client Connection List.....	5.14
Client Trail.....	5.15
Connected Termination Point Count .....	5.16

Connection Id .....	5.17
Connection Termination Point Id .....	5.18
Cross-Connection Id .....	5.19
Cross-Connection Name .....	5.20
Cross-Connection Object Pointer .....	5.21
Current Problem List .....	5.22
Directionality .....	5.23
Downstream Connectivity Pointer .....	5.24
Equipment Holder Address .....	5.25
Equipment Holder Type .....	5.26
Equipment Id .....	5.27
External Time .....	5.28
Fabric Id .....	5.29
From Termination .....	5.30
Group Termination Point Id .....	5.31
Holder Status .....	5.32
Idle TP Count .....	5.33
Information Transfer Capabilities .....	5.34
Label of Far End Exchange .....	5.35
List of Characteristic Info .....	5.36
Location Name .....	5.37
Managed Element Id .....	5.38
Managed Element Complex Id .....	5.39
Multi-Point Cross-Connection Id .....	5.40
Network Id .....	5.41
Network Level Pointer .....	5.42
Number of Circuits .....	5.43
Operational State .....	5.44
Protected .....	5.45
Redline .....	5.46
Replaceable .....	5.47
Serial Number .....	5.48
Server Connection List .....	5.49
Server Trail List .....	5.50
Signalling Capabilities .....	5.51
Signal Type .....	5.52
Software Id .....	5.53
Subordinate Circuit Pack Software Load .....	5.54
Supportable Client List .....	5.55
Supported By Object List .....	5.56
System Timing Source .....	5.57
System Title .....	5.58
Total TP Count .....	5.59
To Termination .....	5.60
TP Pool Id .....	5.61
TPs In GTP List .....	5.62
TPs In TP Pool List .....	5.63
Trail Id .....	5.64
Trail Termination Point Id .....	5.65
Transmission Characteristics .....	5.66
Upstream Connectivity Pointer .....	5.67
Usage State .....	5.68
User Label .....	5.69
Vendor Name .....	5.70
Version .....	5.71
Z-Termination Point Instance .....	5.72

## **A.4 Name bindings**

Alarm Record .....	6.1
Alarm Severity Assignment Profile .....	6.2
Circuit Pack .....	6.3
Connection R1 .....	6.4
Connection Termination Point Source .....	6.5
Connection Termination Point Sink .....	6.6
Cross-Connection .....	6.7
Equipment .....	6.8
Equipment Holder .....	6.9
Event Forwarding Discriminator .....	6.10
Fabric.....	6.11
GTP .....	6.12
Log .....	6.13
Managed Element.....	6.14
Managed Element Complex .....	6.15
Multi-Point Cross-Connection.....	6.16
Network .....	6.17
Software.....	6.18
TP Pool.....	6.19
Trail .....	Appendix II.4.1
TrailR1.....	6.20
Trail Termination Point Source .....	6.21
Trail Termination Point Sink.....	6.22

## **A.5 Actions**

Add TPs To GTP .....	7.1
Add TPs To TP Pool .....	7.2
Allow Audible Visual Local Alarm.....	7.3
Connect.....	7.4
Disconnect .....	7.5
Inhibit Audible Visual Local Alarm.....	7.6
Remove TPs From GTP .....	7.7
Remove TPs From TP Pool.....	7.8
Reset Audible Alarm .....	7.9
Switch Over.....	7.10

## **A.6 Notifications**

Attribute Value Change .....	8.1
Communications Alarm.....	8.2
Environmental Alarm .....	8.3
Equipment Alarm .....	8.4
Object Creation.....	8.5
Object Deletion.....	8.6
Processing Error Alarm .....	8.7
State Change.....	8.8

## **A.7 Parameters**

Alarm Effect On Service .....	9.1
Create Error .....	9.2

# Appendix I

## User guidelines

(This appendix does not form an integral part of this Recommendation)

### I.1 Introduction

This appendix contains user guidelines to clarify the use of object classes, attributes defined in this Recommendation. These guidelines are also applicable to Recommendation M.3100 (1992). The clarification provided here to aid the users of this Recommendation in gaining further understanding of the model. In some cases examples are provided on how to use the model for specific technology.

### I.2 Use of supported by object list

This attribute is used to represent a dependency of the object this is contained in to other objects. For example if contained in termination point, this attribute may point to a power source. Changes of state (e.g. disabled) of these other objects may affect the state of the of the object containing this attribute. In the above example, the termination point may become disabled because of the power source.

### I.3 Use of upstream and downstream connectivity pointers

#### I.3.1 Downstream connectivity pointer

This attribute indicates the instance(s) of termination point from which information (traffic) is received by the termination point object containing this attribute. This attribute points to (one or many) termination point(s) in the same managed element. However a value of NULL can be used when the related object is in a different managed element or when the termination point is not connected. This attribute is read-only and cannot be modified directly. It will be updated as a side-effect of operations modifying the connectivity in the element (such as connect, disconnect operations on the fabric).

#### I.3.2 Upstream connectivity pointer

This attribute indicates the instance(s) of termination point to which information (traffic) is sent by the termination point object containing this attribute. This attribute points to (one or many) termination point(s) in the same managed element. However, a value of NULL can be used when the related object is in a different managed element or when the termination point is not connected. This attribute is read-only and cannot be modified directly. It will be updated as a side-effect of operations modifying the connectivity in the element (such as connect, disconnect operations on the fabric).

Figure I.1 gives an example on the use of these pointers in a unidirectional configuration composed of three network elements.

### I.4 Use of cross-connection objects

A connection between two termination points should be modeled using a cross-connection object when the assignment is flexible and can be modified through the management interface. This assignment can then be modified by using the connect/disconnect action on the fabric, and/or deleting the cross-connection objects directly.

In the cases where the connection cannot be modified through the management interface (e.g. the assignment is not flexible or human intervention is required), cross-connection objects should not be used. In those cases, the connection will simply be modeled by the connectivity pointers.

### I.5 Cross-connection use examples

The goal of this subclause is to provide examples of cross-connection modeling using the managed object classes defined in the cross-connection fragment (see 3.1.3). These examples are presented in the form of annotated figures. In these figures, thin lines represent connectivity pointers and thick, shaded lines represent CrossConnectionObject pointers and pointers in associated objects such as Cross-Connection and GTP. Generic examples are first presented, followed by examples applied to a specific technology (SDH).

### **I.5.1 Point-to-point cross-connection between two termination points**

Figure I.2 provides an example of the simplest type of cross-connection: a point-to-point cross-connection between two termination points. The Upstream and Downstream connectivity pointers in each termination point to the other termination point, the cross-connection object pointers in the termination pointers in each termination point to the other termination point, the cross-connection object pointers in the termination points point to the cross-connection object and the From and To termination pointers in the cross-connection object point to the termination points.

### **I.5.2 Cross-connection between two groups of termination points**

There are cases where a group of termination points must be treated as a single entity and cross-connected as such. In those cases, the termination points to be grouped are included in a GTP (Group of Termination Points) object and it is the GTP objects that are cross-connected. Figure I.3 shows an example of such a configuration where each group contains two termination points.

### **I.5.3 Modeling a broadcast**

A Multipoint Cross-Connection object is used to represent a broadcast (or point-to-multipoint) cross-connection. The Multipoint Cross-Connect object only has a pointer to the originator of the broadcast. The destination of the broadcast is reflected in a pointer in the Cross-Connection Objects that are contained in the Multipoint Cross-Connection object. Figure I.4 shows an example of a broadcast configuration.

### **I.5.4 Suspending a cross-connection**

The model allows set-up of a cross-connection in a state which prevents traffic from flowing through it. For example, a cross-connection may be set up and tested but the telephone company wants to prevent traffic from flowing through it before the service is billed to the customer. This can be done either by putting the cross-connection in an intrusive test configuration and allowing a test signal to flow through it, or by locking the cross-connection, in which case the termination points will generate a 'not equipped' signal. The model supports the administrative state in the cross-connection objects to allow this situation. In this case, the crossConnectionObject pointers will remain the same but the connectivity pointers in the termination points will be set to NULL.



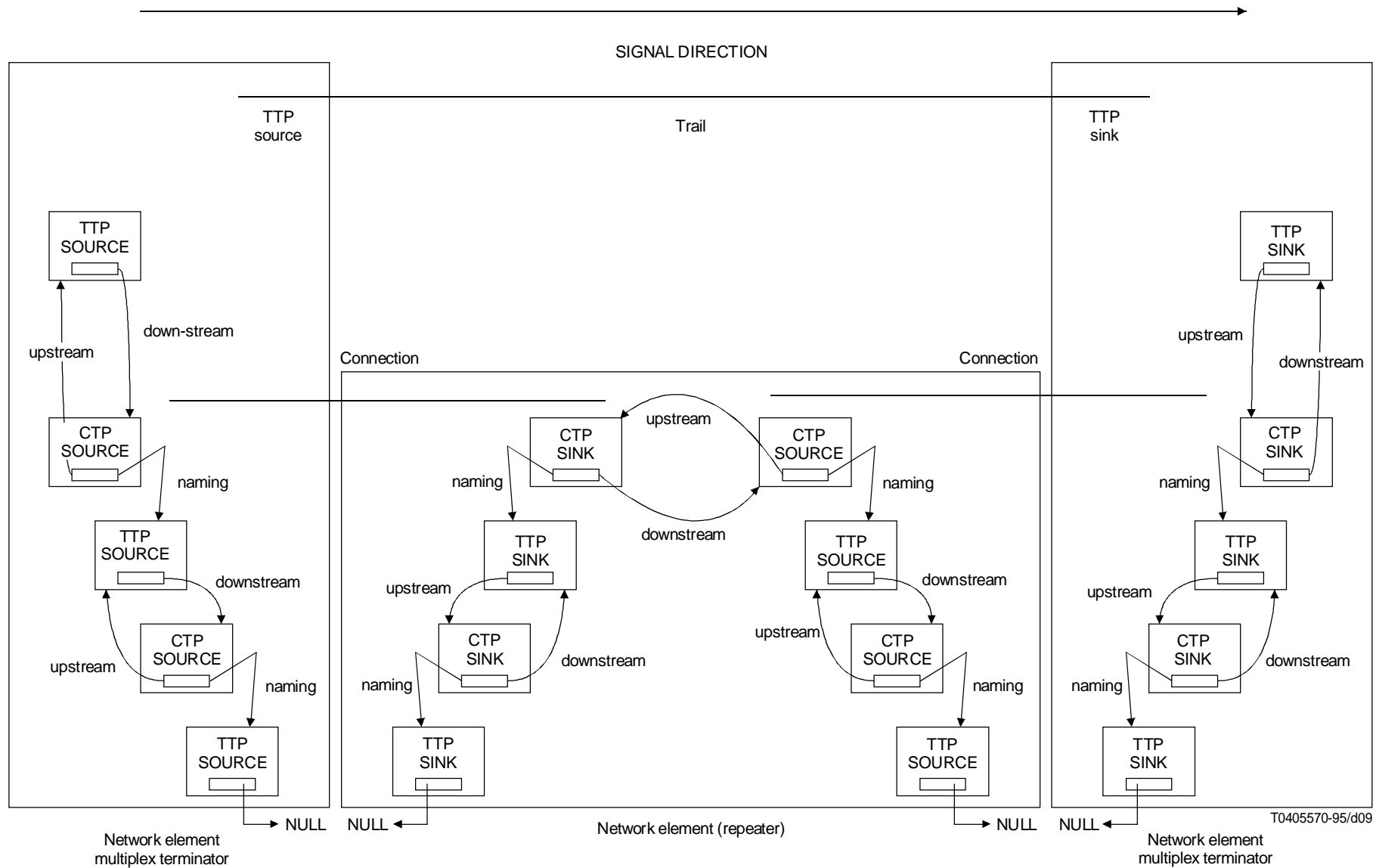


FIGURE I.1/M.3100

**Downstream, upstream pointer use example**

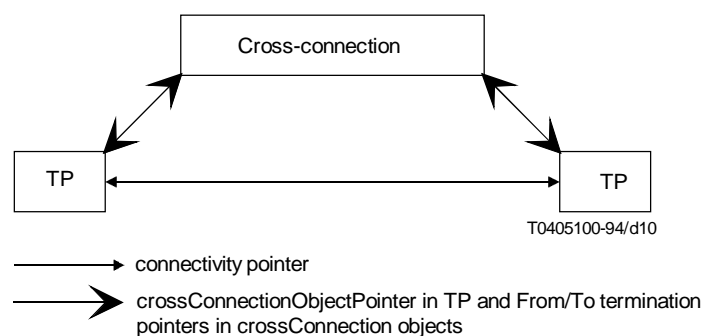


FIGURE I.2/M.3100  
Point-to-point cross-connection between two termination points

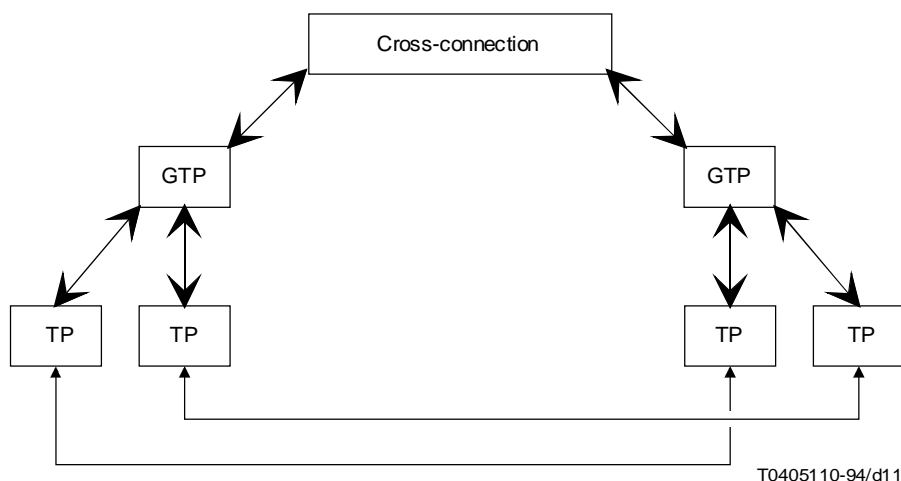


FIGURE I.3/M.3100  
Cross-connection with GTP

#### Applications of the cross-connection model in an SDH context.

NOTE – For a detailed description of the SDH format, please refer to Recommendations G.707, G.708, and G.709.

#### I.5.5 Queries of cross-connections

Given the M.3100 model, it is very easy to retrieve information on cross-connections. Queries based on the state of a cross-connection, its name, one of its endpoints or another attribute of the cross-connection can be performed by simple filtering on the cross-connection objects.

Also, to determine if a termination point is involved in a cross-connection, one simply has to look at the CrossConnectionObject pointer. If the CrossConnectionObject pointer points to the Fabric, the termination point is neither cross-connected nor reserved for cross-connection (assigned to a group). If the CrossConnectionObject pointer does not point to the Fabric, the termination point is assigned to a cross-connection or reserved. In this case, the source of the signal is indicated by the connectivity pointer.

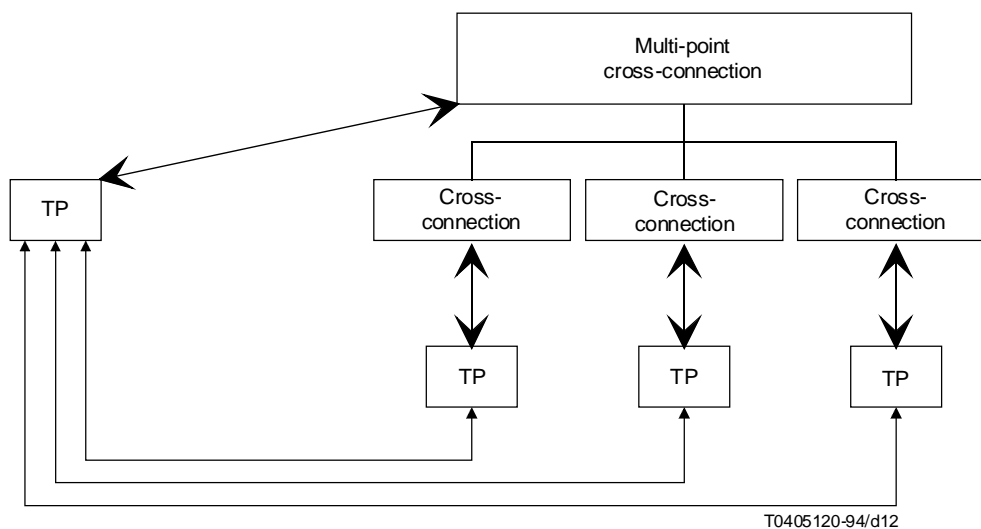


FIGURE I.4/M.3100

### Broadcast

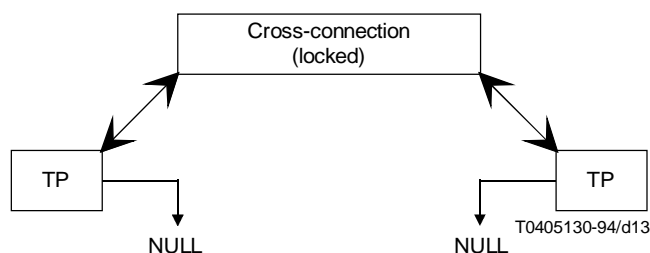


FIGURE I.5/M.3100

### Locked cross-connection

## I.5.6 Unidirectional cross-connection using GTPs

The model presents the cross-connection of groups of unidirectional termination points in an intuitive manner and allows to know the connectivity between the termination points by issuing a single M-GET to retrieve the connectivity pointer in the Termination Point object. Figure I.6 provides an example of how such a cross-connection would be modeled. In this example, GTP objects are used to group two TU-11 (1.728 Mbit/s) termination points that are to be treated as a single entity for management purposes.

## I.5.7 Cross-connection of concatenated payloads

The SDH hierarchy allows for concatenated payloads, that is, several payloads of a lower rate can be combined to form a synchronous payload of a higher rate.

Cross-connection of concatenated payloads is a special case. A termination point whose traffic is carried by  $n$  concatenated payloads can be cross-connected to exactly  $n$  termination points of the lower rate. In all other cases, a point-to-point cross-connection will be established between termination points of the same characteristic information type and each sink or bidirectional termination point will be the sink of exactly one termination point.

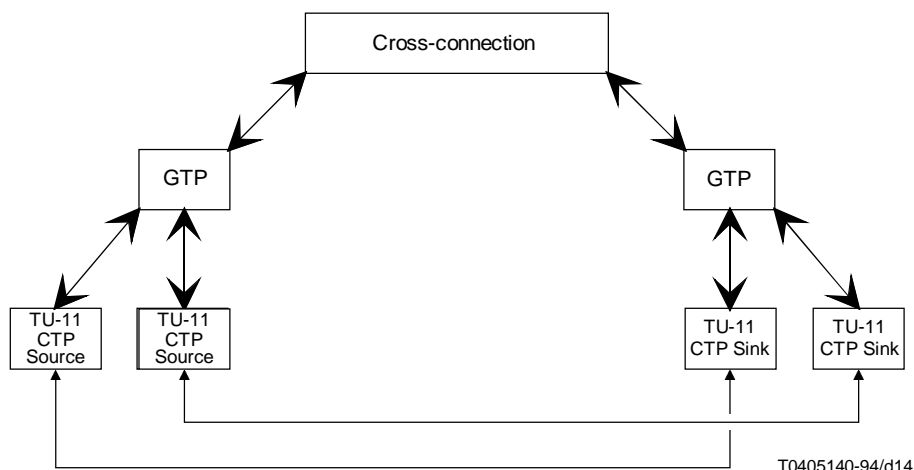


FIGURE I.6/M.3100

### Unidirectional 2xTU-11 point-to-point-cross-connection with proposed model

Since a sink or bidirectional termination point whose traffic is carried by  $n$  concatenated payloads may be the sink of zero or  $n$  source or bidirectional termination points, its connectivity pointer must be special. The connectivity pointer in such a termination point will point to either zero or  $n$  source or bidirectional termination points. All the other sink or bidirectional termination points will have a connectivity pointer that can point only to zero or one termination point.

Figure I.7 is an example of this situation using the VC-4 according to the North American practice. The VC-4 (155 Mbits/s) is carried by three AU-3s (52 Mbits/s each) and, as such, can be cross-connected to three AU-3s. Another configuration is to connect the VC-4 directly to an AU-4 (see Figure I.8).

#### I.5.8 Cross-connection of indirect adaptors

The SDH model introduces the concept of “indirect adaptors” representing an intermediate step in the multiplexing process. One such adaptor is the TUG-2 at 6.9 Mbits/s which represents the multiplexing of either four TU-11 (1.7 Mbits/s), three TU-12 (2.3 Mbits/s), or one TU-2 (6.9 Mbits/s).

The model allows for Cross-Connection of the termination points contained in indirect adaptors. The manager does not have to be aware of the actual content of the group but can still manage the cross-connection of the group as a whole. An example of this situation is the cross-connection of TUG-2s where the content of the TUG-2 may change while the cross-connection remains undisturbed. The connect action requests the connection of two TUG-2s. Two GTPs are automatically created to reflect the content of the TUG-2s. A cross-connection is established between these GTPs. If the content of the TUG-2s changes (for example from three TU-12s to one TU-2) the content of the GTPs will also change to reflect this but the management view of the cross-connection will not change.

#### I.5.9 Cross-connection of arbitrary groups

The model allows for the cross-connection of arbitrary GTPs. The only restriction is that the GTPs must be composed of compatible termination points. (See Figure I.11.)

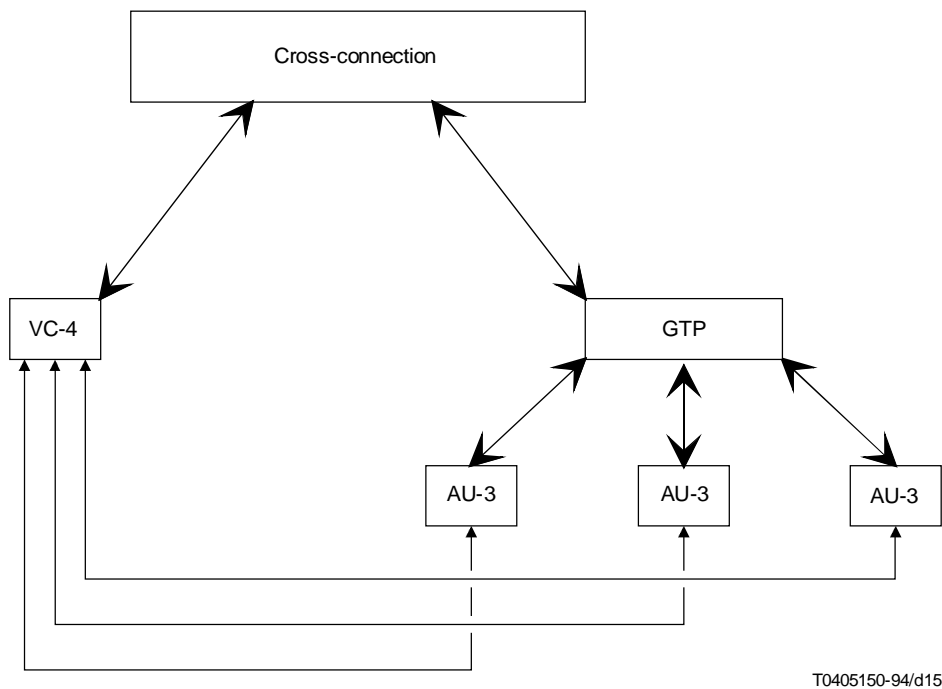


FIGURE I.7/M.3100  
VC-4 connected to three AU-3s

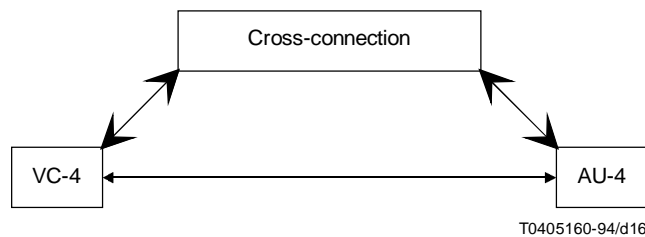


FIGURE I.8/M.3100  
VC-4 connected to one AU-4

#### I.5.10 Use of cross-connection object pointer

The MultipleConnections choice of the crossConnectionObjectPointer attribute syntax is used in SDH when a bidirectional termination point is connected in both directions, using unidirectional cross-connection object in each direction to two other TPs (see Figure I.12) or one another bidirectional TP. This choice is also used in Recommendation G.774.04 for protection of a broadcast where part of the legs can be protected and others are not protected. In this case, two different objects are used, the first one is an mpCrossConnection object that holds the individual unprotected legs and the other is an mpCrossConnectionProtection that holds all protected legs.

#### I.6 Object classes and logical layering

Any object class may be used in any layer of the Logical Layered Architecture (LLA).

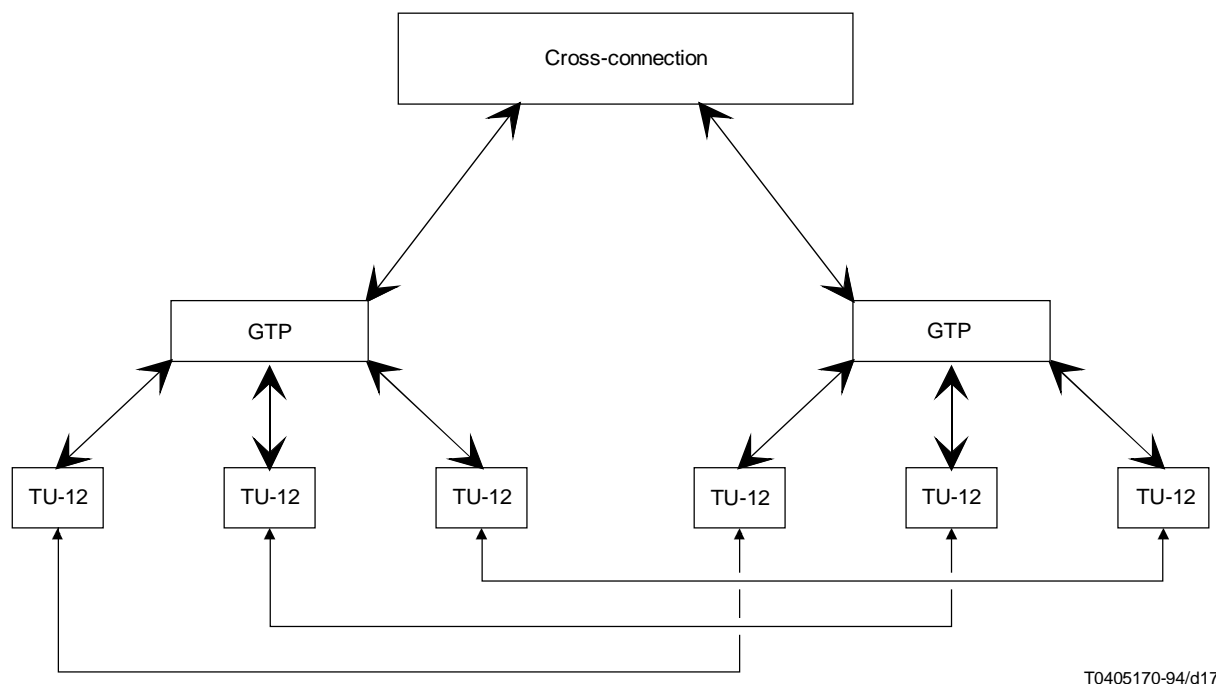


FIGURE I.9/M.3100  
Cross connection of two TUG-2s containing three TU-12s

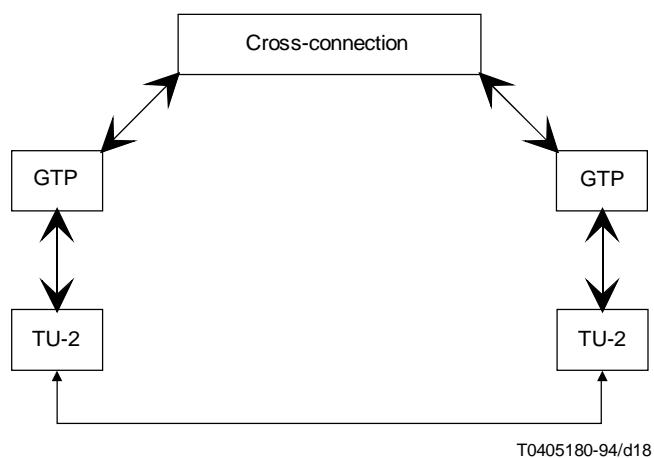
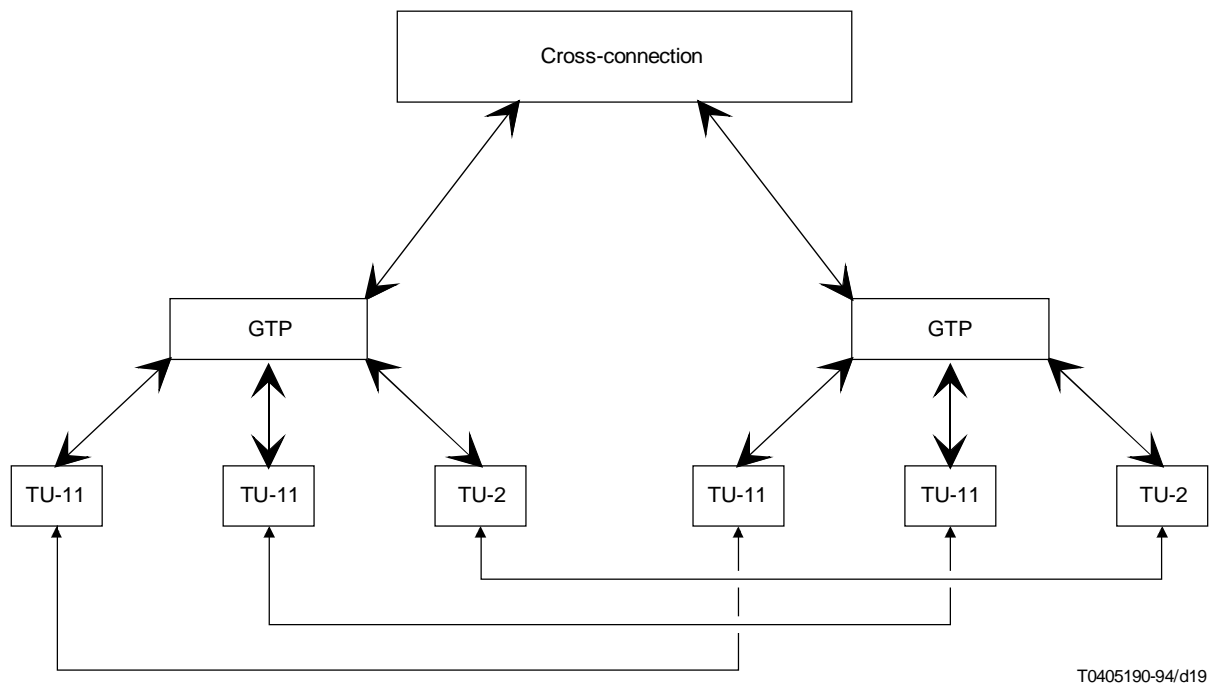


FIGURE I.10/M.3100  
The two TUG-2s now contain one TU-2

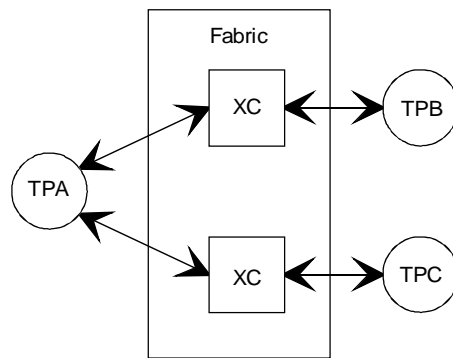
## I.7 Mandatory naming attribute

Rec. X.720 | ISO SMI (10165-1) imposes that naming attributes should be defined as mandatory for instantiable managed object classes. In Recommendation M.3100, object classes such as TTPSource/Sink/Bidirectional have the naming attribute in a conditional package. When these classes are instantiated, this results in the condition evaluating to true (making the naming attribute mandatory).



T0405190-94/d19

FIGURE I.11/M.3100  
Cross-connection of GTPs composed of two TU-11s and one TU-2



T0405200-94/d20

↔ Point relation (cross-connection object pointer in TP and from/to pointer in XC)

FIGURE I.12/M.3100  
Cross-connection object pointer

## **I.8 Interoperability between Recommendation M.3100 (1992) and this Recommendation**

### **I.8.1 Naming attributes**

The naming attributes in Recommendation M.3100 (1992) did not include ordering and substrings even though the syntax permitted it. Implementations of managing and managed systems built to this Recommendation will interoperate in the sense that the managing system may issue a CMISFilter containing the criteria for ordering and string comparisons and the managed system should be capable of executing it. If the managing system implemented Recommendation M.3100 (1992), then it is not expected to include the criteria for ordering and substrings even if the managed system has implemented this Recommendation. In the case where the managing system implements this Recommendation and managed system implemented the M.3100 (1992), it is possible for the criteria requested by the manager to include ordering and substrings. In this case, the invalid filter error defined in CMIS may be returned.

### **I.9 Support for multi-point trails**

Multipoint trails are supported in the model by abstracting each leg as a trail. Note that in this configuration, multiple trails may share one end point.

## **Appendix II**

### **Deprecated/obsolete management information from Recommendation M.3100 (1992)**

(This appendix does not form an integral part of this Recommendation)

### **II.1 Object classes**

#### **II.1.1 Connectivity**

**connectivity MANAGED OBJECT CLASS**

**DERIVED FROM** "Recommendation X.721: 1992":top;

**CHARACTERIZED BY**

**connectivityPackage PACKAGE**

**BEHAVIOUR**

**connectivityBehaviour BEHAVIOUR**

**DEFINED AS**

“The Connectivity object class is a class of managed objects which ensures the transfer of information between two termination points. It is not instantiable because the transfer is effected via the client-server relationship of trail and connection. Connectivity direction is determined by the directionality of the a- and z-termination points.

If an instance of this class is bidirectional, the a- and z-termination points shall also be bidirectional. If an instance of this class is unidirectional, the a point shall be the source TP and the z-termination point shall be the sink TP.

The operational state indicates the capability to carry a signal.”

;;

**ATTRIBUTES**

**directionality GET,**

**"Recommendation X.721: 1992":administrativeState GET-REPLACE,**

**"Recommendation X.721: 1992":operationalState GET,**

**a-TPInstance GET,**

**z-TPInstance GET**

;;;

**CONDITIONAL PACKAGES**

**createDeleteNotificationsPackage PRESENT IF**

**"the objectCreation and objectDeletion notifications defined in Recommendation X.721 are supported by an instance of this managed object class",**

**attributeValueChangeNotificationPackage PRESENT IF**

**"the attributeValueChange notification defined in Recommendation X.721 is supported by an instance of this managed object class",**

**stateChangeNotificationPackage PRESENT IF**



"the stateChange notification defined in Recommendation X.721 is supported by an instance of this managed object class",  
characteristicInformationPackage PRESENT IF  
"an instance supports it.",  
protectedPackage PRESENT IF  
"an instance supports it.",  
tmnCommunicationsAlarmInformationPackage PRESENT IF  
"the communicationsAlarm notification (as defined in Recommendation X.721) is supported by this managed object",  
alarmSeverityAssignmentPointerPackage PRESENT IF  
"the tmnCommunicationsAlarmInformationPackage package is present AND the managed object supports configuration of alarm severities";

REGISTERED AS {m3100ObjectClass 13};

## II.1.2 Trail

trail MANAGED OBJECT CLASS

DERIVED FROM connectivity;

CHARACTERIZED BY

trailPackage PACKAGE

BEHAVIOUR

trailBehaviour BEHAVIOUR

DEFINED AS

“Trail is a class of managed objects in layer networks which is responsible for the integrity of transfer of characteristic information from one or more other layer networks. A trail is composed of two or more Trail Termination Points and one or more connection and associated connection termination points.”

;;

ATTRIBUTES

trailId GET;;;

CONDITIONAL PACKAGES

serverConnectionListPackage PRESENT IF "an instance supports it",

clientConnectionPackage PRESENT IF "an instance supports it";

REGISTERED AS {m3100ObjectClass 14};

## II.1.3 Connection

connection MANAGED OBJECT CLASS

DERIVED FROM connectivity;

CHARACTERIZED BY

connectionPackage PACKAGE

BEHAVIOUR

connectionBehaviour BEHAVIOUR

DEFINED AS

“The Connection object class is a class of managed objects responsible for the transparent transfer of information between connection termination points. A connection is a component of a trail.

Several connections can be bundled into a higher rate trail. A sequence of one or more connections are linked together to form a trail. A connection may be either uni- or bi-directional.”

;;

ATTRIBUTES

connectionId GET;;;

CONDITIONAL PACKAGES

serverTrailListPackage PRESENT IF "an instance supports it",

clientTrailPackage PRESENT IF "an instance supports it";

REGISTERED AS {m3100ObjectClass 12};

## **II.2 Packages**

### **II.2.1 Client connection**

**clientConnectionPackage PACKAGE**

**ATTRIBUTES**

**clientConnection GET;**

**REGISTERED AS {m3100Package 8};**

## **II.3 Attributes**

### **II.3.1 Client connection**

**clientConnection ATTRIBUTE**

**WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.ObjectInstance;**

**MATCHES FOR EQUALITY;**

**BEHAVIOUR**

**clientConnectionBehaviour BEHAVIOUR**

**DEFINED AS**

“The value of this attribute identifies the client object instance that is served by a trail at a higher order networklayer.”;;

**REGISTERED AS {m3100Attribute 9};**

## **II.4 Name binding**

### **II.4.1 Trail**

**trail-network NAME BINDING**

**SUBORDINATE OBJECT CLASS trail;**

**NAMED BY**

**SUPERIOR OBJECT CLASS network;**

**WITH ATTRIBUTE trailId;**

**CREATE**

**WITH-REFERENCE-OBJECT,**

**WITH-AUTOMATIC-INSTANCE-NAMING;**

**DELETE**

**ONLY-IF-NO-CONTAINED-OBJECTS;**

**REGISTERED AS {m3100NameBinding 22};**

### **II.4.2 Connection**

**connection-network NAME BINDING**

**SUBORDINATE OBJECT CLASS connection;**

**NAMED BY**

**SUPERIOR OBJECT CLASS network;**

**WITH ATTRIBUTE connectionId;**

**CREATE;**

**DELETE;**

**REGISTERED AS {m3100NameBinding 2};**

## Appendix III

### Differences from Recommendation M.3100 (1992)

(This appendix does not form an integral part of this Recommendation)

The following revisions to Recommendation M3100 (1992) are incorporated in this Recommendation. Only significant changes (no editorial corrections) are listed below.

**III.1** The following new object classes were introduced: pipe, trailR1, fabricR1, connectionR1, equipmentR1, managedElementR1, managedElementComplex, networkR1, softwareR1, circuitSubgroup, circuitPack, equipmentHolder. It is recommended that future specifications use the revised object classes mentioned above. The addition of new object classes required adding the clientConnectionObjectList package, environmentalAlarmR1Package, equipmentsEquipmentAlarmR1Package, softwareAlarmR1Package, processingErrorAlarmR1Package, new attributes and switchOver action.

**III.2** The following object classes have been removed and moved to Appendix II: connectivity, trail, connection

**III.3** The naming attribute definitions were updated to include ordering and substring in the MATCHES FOR clause along with a behaviour definition.

**III.4** New probable cause values were added in the ASN.1 module.

**III.5** Correction was made to Figure 5.

**III.6** Annex B is deleted and Appendix I has new contents.

**III.7** Three new appendices on “User Guidelines”, “Deprecated/Obsolete Management Information” and “Differences from Recommendation M.3100 (1992)” were added.

**III.8** Name bindings for managed element, managed element complex, and network object classes to X.500 Directory object classes organization and organizational unit are included. Name bindings are also introduced for the object classes circuitPack and equipmentHolder.

**III.9** ASN.1 module includes extensibility rules for some of the types and productions for the new attributes and parameter.

**III.10** The production for connectActionInformation was enhanced to allow creation of redlined cross-connection.

### References

The following Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision: all users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

- [1] CCITT Recommendation M.3010 (1992), *Principles for a telecommunications management network*.
- [2] CCITT Recommendation X.720 (1992), *Information technology – Open Systems Interconnection – Structure of management information: Management information model*.
- [3] CCITT Recommendation X.722 (1992), *Information technology – Open Systems Interconnection – Structure of management information: Guidelines for the definition of managed objects*.

- [4] CCITT Recommendation X.208 (1988), *Specification of Abstract Syntax Notation One, (ASN.1)*.
- [5] CCITT Recommendation X.721 (1992), *Information technology – Open Systems Interconnection – Structure of management information: Definition of management information*.
- [6] ITU-T Recommendation G.803 (1993), *Architectures of transport networks based on the Synchronous Digital Hierarchy (SDH)*.
- [7] ITU-T Recommendation Q.821 (1993), *Stage 2 and Stage 3 Description for the Q3 Interface – Alarm surveillance*.