INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Q.823
(07/96)

SERIES Q: SWITCHING AND SIGNALLING
Specifications of Signalling System No. 7 – Q3 interface

Stage 2 and Stage 3 functional specifications for traffic management

ITU-T Recommendation Q.823

(Previously CCITT Recommendation)

ITU-T Q-SERIES RECOMMENDATIONS

SWITCHING AND SIGNALLING

SIGNALLING IN THE INTERNATIONAL MANUAL SERVICE	Q.1–Q.3
INTERNATIONAL AUTOMATIC AND SEMI-AUTOMATIC WORKING	Q.4–Q.59
FUNCTIONS AND INFORMATION FLOWS FOR SERVICES IN THE ISDN	Q.60–Q.99
CLAUSES APPLICABLE TO ITU-T STANDARD SYSTEMS	Q.100–Q.119
SPECIFICATION OF SIGNALLING SYSTEMS No. 4 AND No. 5	Q.120–Q.249
SPECIFICATIONS OF SIGNALLING SYSTEM No. 6	Q.250–Q.309
SPECIFICATIONS OF SIGNALLING SYSTEM R1	Q.310–Q.399
SPECIFICATIONS OF SIGNALLING SYSTEM R2	Q.400–Q.499
DIGITAL EXCHANGES	Q.500–Q.599
INTERWORKING OF SIGNALLING SYSTEMS	Q.600–Q.699
SPECIFICATIONS OF SIGNALLING SYSTEM No. 7	Q.700–Q.849
General	Q.700
Message transfer part	Q.701–Q.709
Simplified message transfer part	Q.710
Signalling connection control part	Q.711–Q.719
Telephone user part	Q.720–Q.729
ISDN supplementary services	Q.730–Q.739
Data user part	Q.740–Q.749
Signalling System No. 7 management	Q.750–Q.759
ISDN user part	Q.760–Q.769
Transaction capabilities application part	Q.770–Q.779
Test specification	Q.780–Q.799
Q3 interface	**Q.800–Q.849**
DIGITAL SUBSCRIBER SIGNALLING SYSTEM No. 1	Q.850–Q.999
General	Q.850–Q.919
Data link layer	Q.920–Q.929
Network layer	Q.930–Q.939
User-network management	Q.940–Q.949
Stage 3 description for supplementary services using DSS 1	Q.950–Q.999
PUBLIC LAND MOBILE NETWORK	Q.1000–Q.1099
INTERWORKING WITH SATELLITE MOBILE SYSTEMS	Q.1100–Q.1199
INTELLIGENT NETWORK	Q.1200–Q.1999
BROADBAND ISDN	Q.2000–Q.2999

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

ITU-T RECOMMENDATION Q.823

STAGE 2 AND STAGE 3 FUNCTIONAL SPECIFICATIONS FOR TRAFFIC MANAGEMENT

Summary

The scope of this Recommendation is to provide an information model which covers the management aspects of the traffic management service functions in an exchange. The scope is limited only to circuit switched networks using hierarchical routing.

This Recommendation focuses on the Stage 2 and Stage 3 description of the Q3 interface between Network Elements (NEs) and Operations Systems (OSs). The Stage 1 description is provided in E.410-Series Recommendations and in Recommendation E.502.

Source

ITU-T Recommendation Q.823 was prepared by ITU-T Study Group 11 (1993-1996) and was approved under the WTSC Resolution No. 1 procedure on the 9th of July 1996.

CONTENTS

Page

Recommendation Q.823

STAGE 2 AND STAGE 3 FUNCTIONAL SPECIFICATIONS FOR TRAFFIC MANAGEMENT

(Geneva, 1996)

1 Scope

The scope of this Recommendation is to provide an information model which covers the management aspects of the traffic management service functions in an exchange. The scope is limited only to circuit switched networks using hierarchical routing.

This Recommendation focuses on the Stage 2 and Stage 3 description of the Q3 interface between Network Elements (NEs) and Operations Systems (OSs). The Stage 1 description is provided in E.410-Series Recommendations and in Recommendation E.502.

The following restrictions on the scope are to be regarded:

– For traffic surveillance and management, this Recommendation makes use of the traffic measurements defined in the E-Series of Recommendations.

– The handling and processing of traffic management related information on the OS level and the forwarding of these data on the OS level are out of the scope of this Recommendation.

– Network performance data is needed as input for the traffic management controls. Sometimes the same data can be used for traffic measurements, but this coincidence is ignored in this context. The network performance data is identified and modelled as far as it is relevant and when the models are not available from other Recommendations.

– Functions which are supportable by information models defined in other Recommendations are not redefined in this Recommendation. Those information models are either referenced directly by the incorporation of the objects or indirectly by subclassing from other information.

– As far as possible and sensible, the functions defined in the OSI Systems Management Functions are considered.

2 Introduction

The objective of traffic management is to enable as many calls as possible to be completed successfully. This objective is met by maximizing the use of available resources in any situation. It is also considered a part of this function to supervise the performance of a network and to be able, if necessary, to take action to control the flow of traffic for optimizing the utilization of the network capacity.

The information model presented in this Recommendation provides a common view for the performance data retrieval from the NE, and for the administration of controls and instructions from the OS to the NE. The performance data provides information for the activation of traffic management controls and for the validation of traffic management actions. It also serves as input data for further traffic management actions.

2.1 Functional requirements

The traffic management control functions considered in the model are based on controls specified in Recommendation E.412.

The performance monitoring functions in this model are based on measurement items selected out of Recommendation E.502.

Appendix I identifies the individual traffic management service functions incorporated in Recommendation M.3200 that are covered by this Recommendation and those that are out of scope of this Recommendation.

Appendix II provides a mapping between E.412 controls and the control objects covered in this Recommendation.

2.2 Modelling methodology

The modelling techniques described in Recommendation M.3020 (TMN Interface Specification Methodology) are used in this Recommendation.

The definitions of the supporting managed object classes given in the OSI Recommendations (X.700-Series Recommendations) have been adopted in the modelling work.

3 Normative references

The following ITU-T 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.

- CCITT Recommendation E.410 (1992), *International network management – General information*.
- CCITT Recommendation E.411 (1992*), International network management – Operational guidance*.
- CCITT Recommendation E.412 (1992), *Network management controls.*
- CCITT Recommendation E.502 (1992*), Traffic measurement requirements for digital telecommunication exchanges.*
- ITU-T Recommendation M.3010 (1996*), Principles for a Telecommunications Management Network.*
- ITU-T Recommendation M.3100 (1995), *Generic network information model.*
- ITU-T Recommendation Q.542 (1993), *Digit exchange design objectives – Operations and maintenance*.
- ITU-T Recommendation Q.763 (1993), *Formats and codes of the ISDN User Part of Signalling System No. 7*.
- ITU-T Recommendation Q.822 (1994), *Stage 1, Stage 2 and Stage 3 description for the Q3 interface – Performance management.*
- CCITT Recommendation X.720 (1992), *Information technology – Open Systems Interconnection – Structure of management information: Management information model.*

- CCITT Recommendation X.721 (1992), *Information technology – Open Systems Interconnection – Structure of management information: Definitions of management information*.
- CCITT Recommendation X.722 (1992), *Information technology – Open Systems Interconnection – Structure of management information: Guidelines for the definition of managed objects*.
- CCITT Recommendation X.730 (1992), *Information technology – Open Systems Interconnection – Systems Management Object management function.*
- CCITT Recommendation X.731 (1992), *Information technology – Open Systems Interconnection – Systems Management: State management function.*
- CCITT Recommendation X.734 (1992), *Information technology – Open Systems Interconnection – Systems Management: Event report management function.*
- ITU-T Recommendation X.738 (1993), *Information technology – Open Systems Interconnection – Systems Management: Summarization function.*

4 Definitions

This Recommendation defines the following terms:

4.1 Alternate routed

4.1.1 alternated routed traffic: Alternate routed traffic is a collection of offered calls for which a circuit subgroup is not the first choice; i.e. the overflow case.

4.1.2 answer signal: Information sent in the backward direction indicating that the call is answered. This term is defined in Annex A/E.410.

4.1.3 bid: An attempt to obtain a circuit in a circuit subgroup or to a destination. A bid may be successful or unsuccessful in seizing a circuit in that circuit group or to that destination. This term is defined in Annex A/E.410.

4.1.4 circuit: A circuit is a transmission means which allows communication between two exchanges. This term is defined in Annex A/E.410.

4.1.5 circuit group: The set of all switched circuits which directly interconnect one exchange with another. This term is defined in Annex A/E.410. For traffic management, a circuit group is not visible; i.e. not modelled on the NE management level.

4.1.6 circuit subgroup: A set of circuits within a circuit group which are uniquely identifiable for operational or technical reasons. A circuit group may consist of one or more circuit subgroups. This term is defined in Annex A/E.410.

4.1.7 destination: A country, an area, an exchange or other location, or a special service, in which the called subscriber is located and that may be specified within the country. A destination is identified by one or more destination codes. This term is defined in Annex A/E.410.

4.1.8 destination code: A destination code is identified by the digits used in analysing and processing the call. This may be defined to whatever accuracy is necessary. This term is defined in Annex A/E.410.

4.2 Direct routed

4.2.1 traffic: Direct routed traffic is a collection of offered calls for which a circuit subgroup is the first choice.

4.2.2 seizure: A seizure is a bid for a circuit in a circuit group which succeeds in obtaining a circuit in that circuit group. This term is defined in Annex A/E.410.

4.2.3 switching node: A switching node is an exchange represented by an instance of a managed element or a subclass of the managed element.

5 Symbols and abbreviations

The following is a list of symbols and abbreviations used in this Recommendation, it does not include the managed object class and attribute names as they occur in the formal definitions of this Recommendation.

ABR	Answer Bid Ratio
ACC	Automatic Congestion Control
ADC	Automatic Destination Control
ASR	Answer Seizure Ratio
HTR	Hard-to-Reach
Id	Identifier
ISDN	Integrated Services Digital Network
M/C/O	Mandatory/Conditional/Optional
M/S	Multivalued/Single-valued
MIB	Management Information Base
NA	Network Aspects
NE	Network Element
OA&M	Operation, Administration and Maintenance
OS	Operations System
OSI	Open Systems Interconnection
PSTN	Public Switched Telephone Network
RDN	Relative Distinguished Name

6 Information model

6.1 Entity relationship diagram

See Figures 6-1 and 6-2.

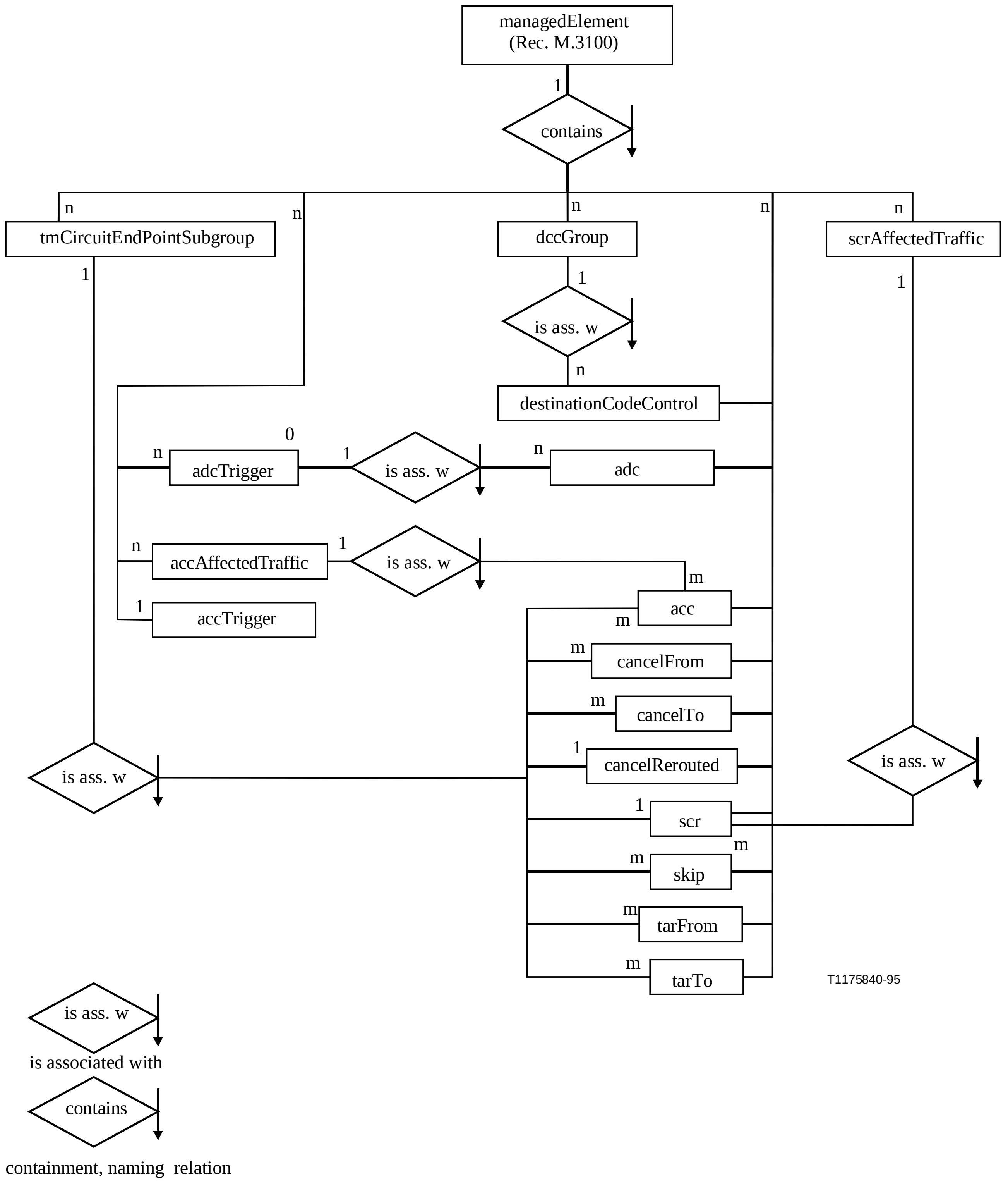

Figure 6-1/Q.823 – Entity relation diagram for control fragment

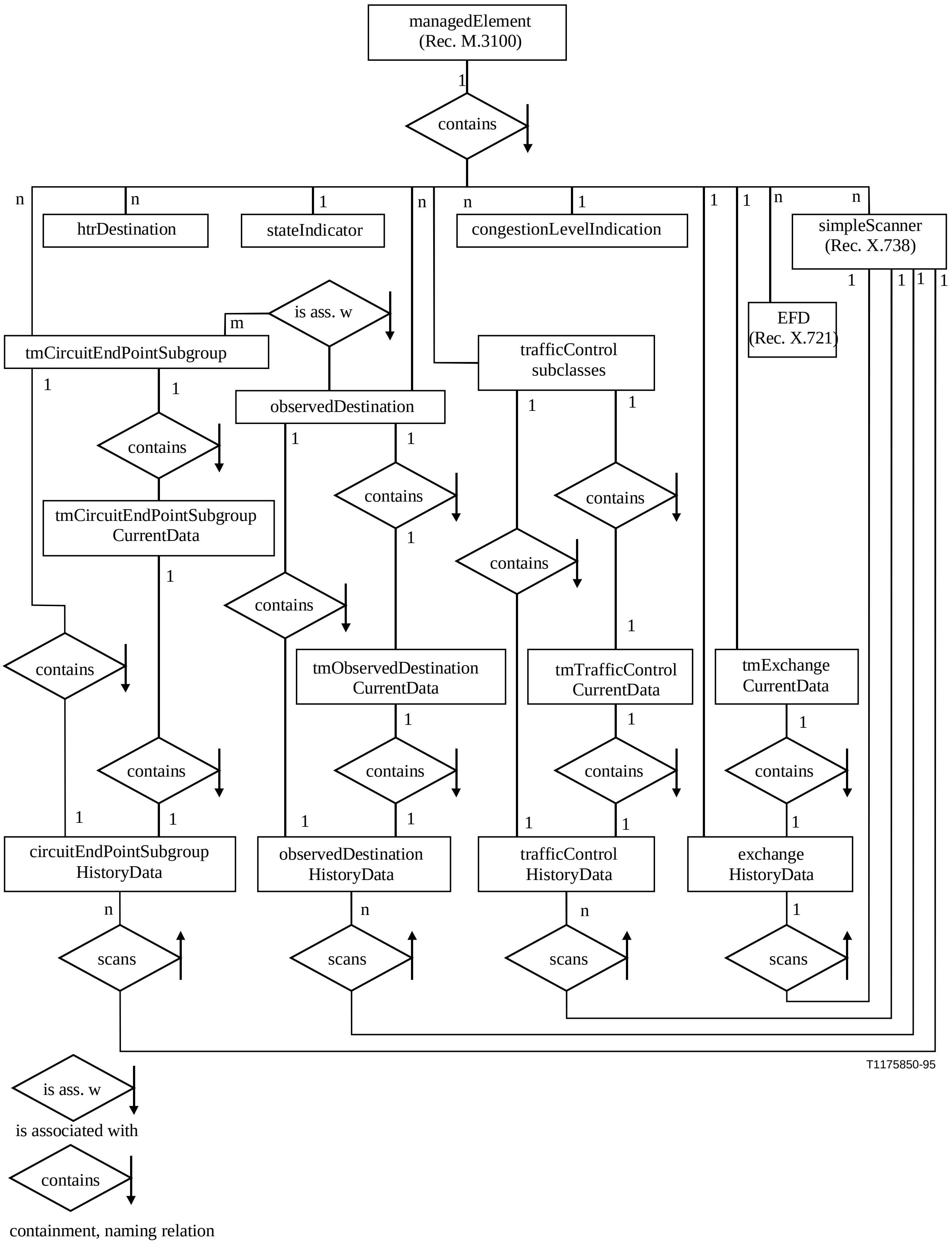
managedElement (Rec. M.3100)
contains
htrDestination
stateIndicator
congestionLevelIndication
simpleScanner (Rec. X.738)
is ass. w
tmCircuitEndPointSubgroup
EFD (Rec. X.721)
trafficControl subclasses
observedDestination
contains
tmCircuitEndPointSubgroup CurrentData
tmObservedDestination CurrentData
tmTrafficControl CurrentData
tmExchange CurrentData
circuitEndPointSubgroup HistoryData
observedDestination HistoryData
trafficControl HistoryData
exchange HistoryData
scans
T1175850-95
is associated with
containment, naming relation

Figure 6-2/Q.823 – Entity relation diagram for managed element, performance, state and administration fragments

6.2 Inheritance hierarchy

See Figures 6-3 and 6-4.

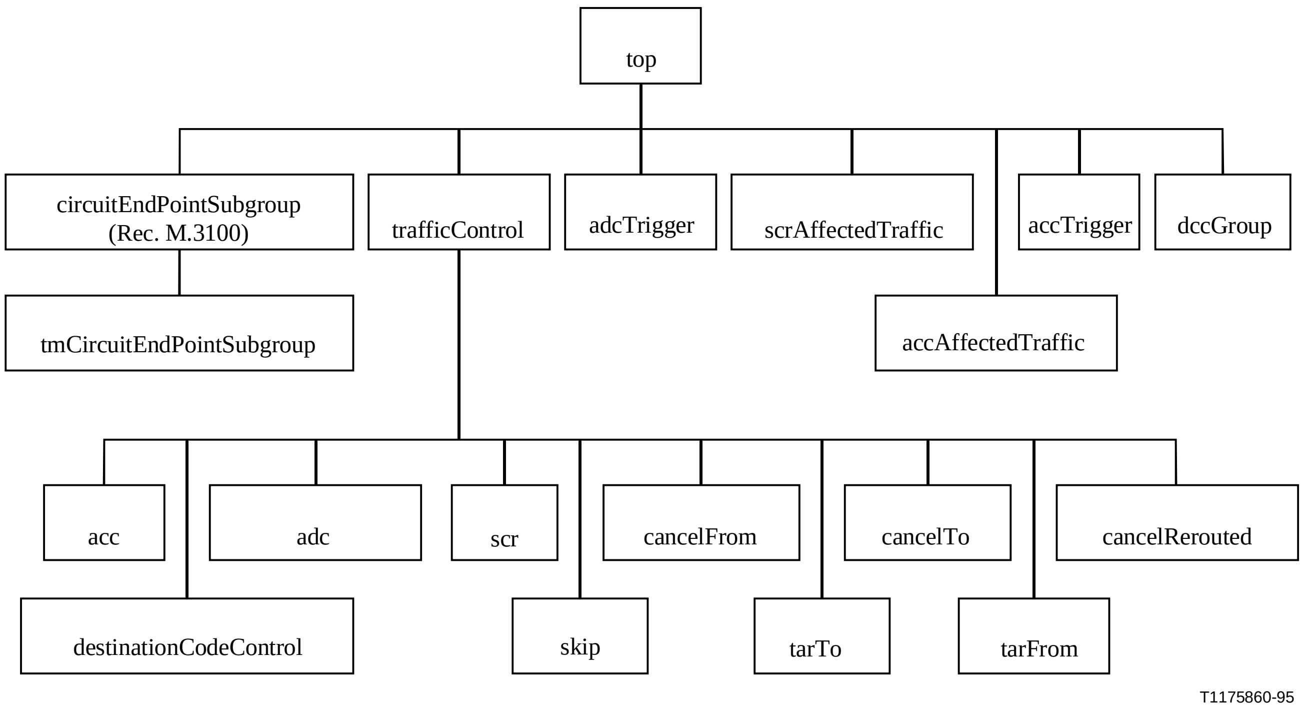

Figure 6-3/Q.823 – Inheritance hierarchy for control fragment

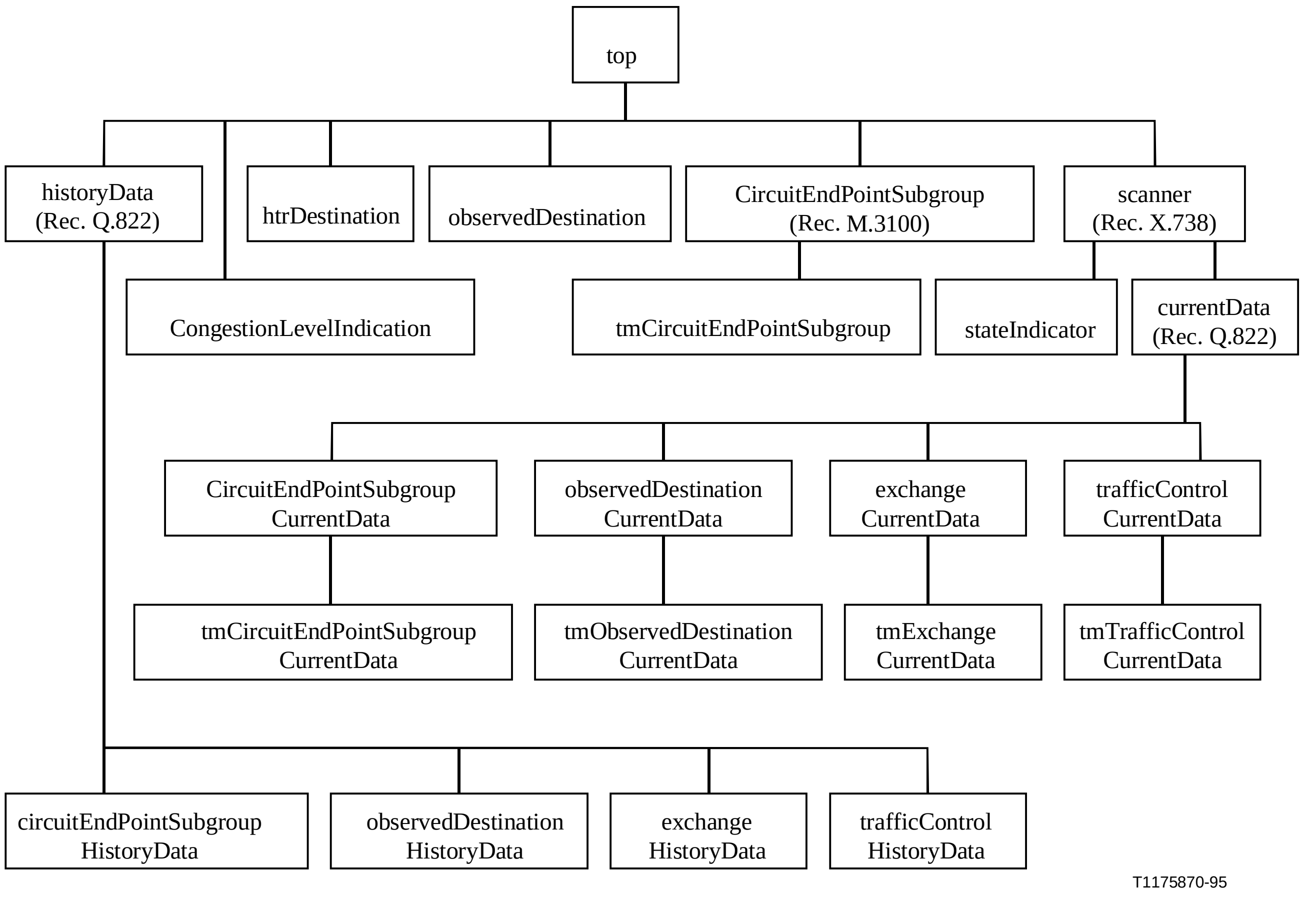
top
historyData (Rec. Q.822)
htrDestination
observedDestination
CircuitEndPointSubgroup (Rec. M.3100)
scanner (Rec. X.738)
CongestionLevelIndication
tmCircuitEndPointSubgroup
stateIndicator
currentData (Rec. Q.822)
CircuitEndPointSubgroup CurrentData
observedDestination CurrentData
exchange CurrentData
trafficControl CurrentData
tmCircuitEndPointSubgroup CurrentData
tmObservedDestination CurrentData
tmExchange CurrentData
tmTrafficControl CurrentData
circuitEndPointSubgroup HistoryData
observedDestination HistoryData
exchange HistoryData
trafficControl HistoryData
T1175870-95

Figure 6-4/Q.823 – Inheritance hierarchy for managed element, performance, state and administrative fragments

6.3 Naming hierarchy

See Figures 6-5 and 6-6.

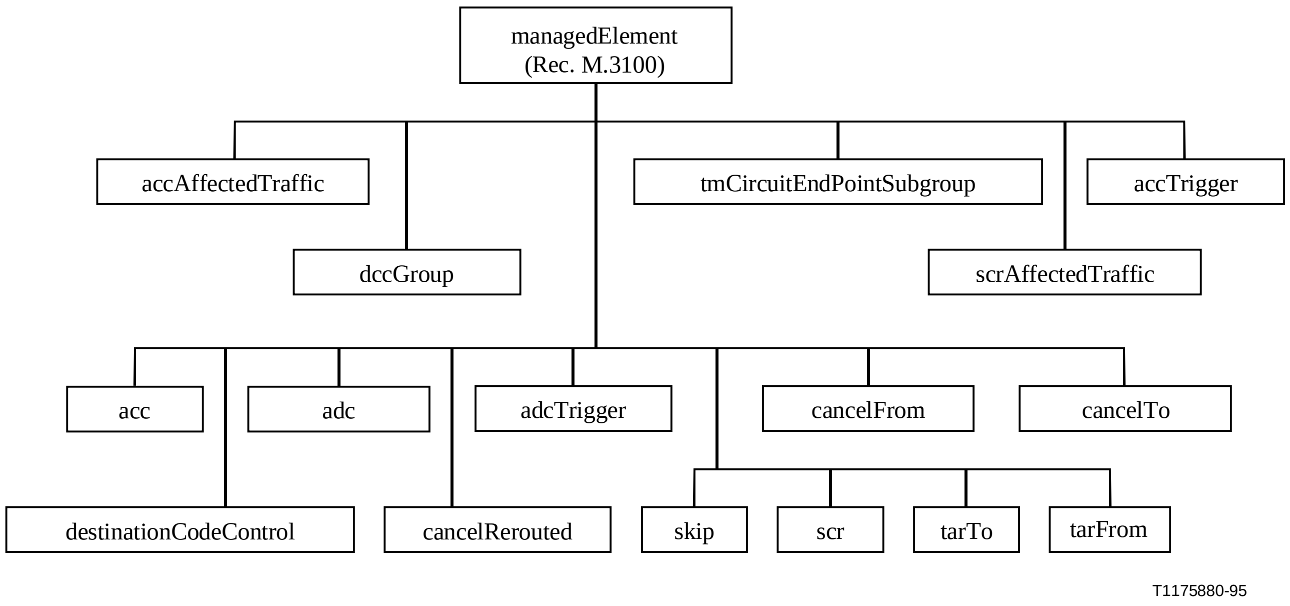

Figure 6-5/Q.823 – Name binding for control fragment

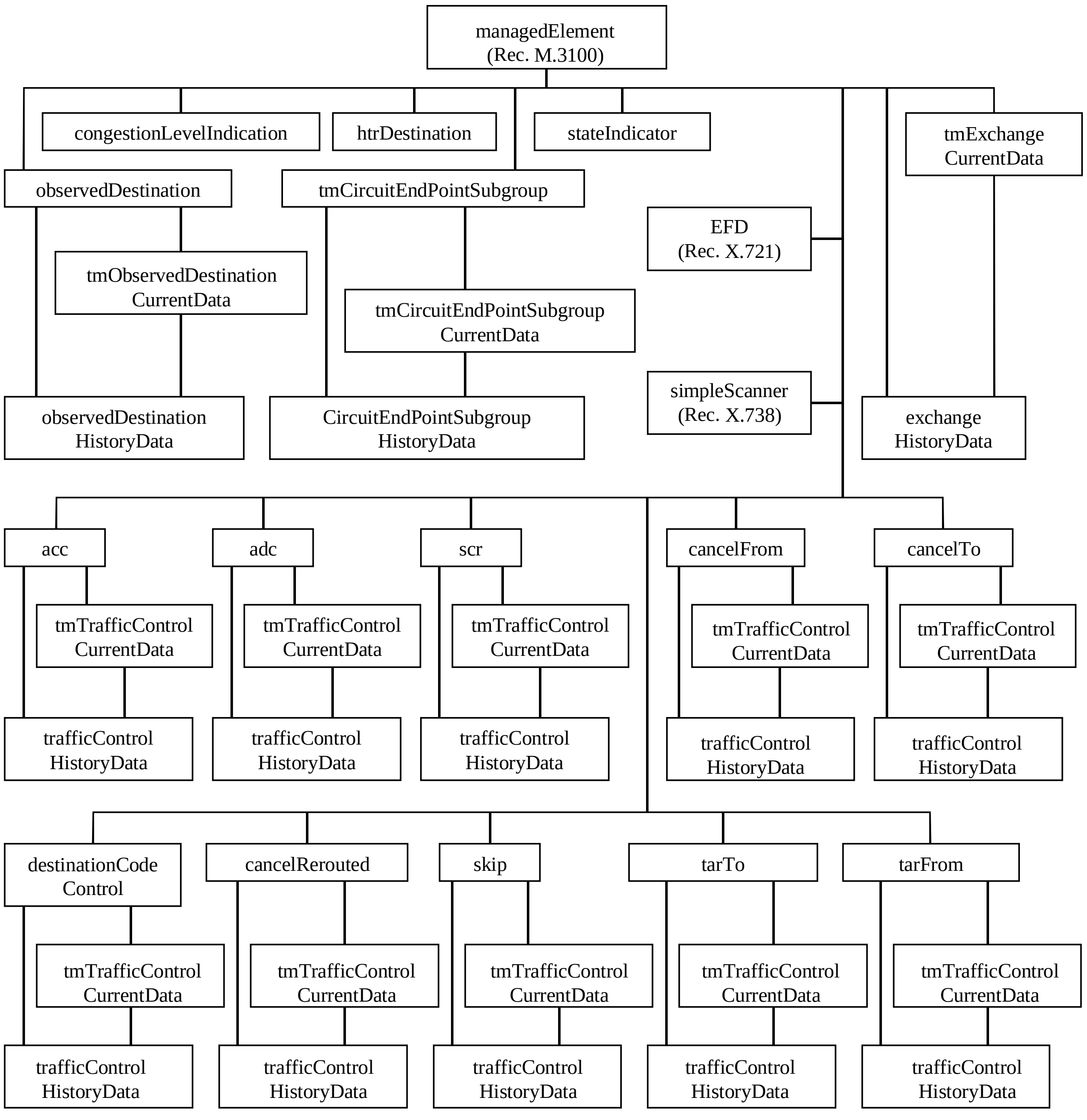
managedElement (Rec. M.3100)
congestionLevelIndication
htrDestination
stateIndicator
tmExchange CurrentData
observedDestination
tmCircuitEndPointSubgroup
EFD (Rec. X.721)
tmObservedDestination CurrentData
tmCircuitEndPointSubgroup CurrentData
simpleScanner (Rec. X.738)
observedDestination HistoryData
CircuitEndPointSubgroup HistoryData
exchange HistoryData
acc
adc
scr
cancelFrom
cancelTo
tmTrafficControl CurrentData
trafficControl HistoryData
destinationCode Control
cancelRerouted
skip
tarTo
tarFrom
T1175890-95

Figure 6-6/Q.823 – Name binding for managed element, performance, state and administrative fragments

7 Information model description

This clause provides a high-level informal description of the traffic management information model.

Subclause 7.1 contains a brief description for each object class used in the information model and provides the following information:

– the purpose of the object class;

– the attributes defined for the object class; and

– the relationship of the object class to other object classes.

Attributes which are common to several object classes are described in 7.2.

Subclause 7.3 describes actions which influence several object classes in the information model.

Subclause 7.4 describes the common aspects of notifications used in the information model.

In the tables listing the attributes of the object classes, the attributes inherited from "Recommendation X.721" top are not mentioned explicitly, although they are present in these objects by inheritance.

7.1 Object class descriptions

This subclause is divided into subsections in which the object classes of the information model are described, to the extent that they are not defined in other Recommendations. For object classes defined in other Recommendations the appropriate reference is specified.

7.1.1 Managed element fragment

This fragment provides definition for the resources associated with the exchange.

7.1.1.1 Congestion level indication (congestionLevelIndication)

Congestion level indication provides an indication of the current congestion level of the managed element object instance in which it is contained. This object instance is not allowed to be instantiated by an OS. See Table 1.

Table 1/Q.823 – Entity – congestionLevelIndication

Attributes	M/S	O/M/C
congestionLevelIndicationId	S	M
congestionLevel	S	M

The following attributes define the entity congestionLevelIndication:

- congestionLevelIndicationID – this attribute is used as the RDN.
- congestionLevel – indicates the present congestion situation in an exchange. It is expressed by the following Machine Congestion Levels (MCL):
  - MCL0: No exchange congestion.
  - MCL1: Moderate exchange congestion, the exchange keeps working. Some calls may be rejected.
  - MCL2: Serious congestion level, the exchange is no longer able to handle all offered traffic. A larger number of calls are rejected than MCL1.
  - MCL3: Unable to process calls. While it is desirable that they do so, some NE may not be able to provide a level 3 indicator during catastrophic failures.

7.1.1.2 Hard-to-reach destination (htrDestination)

An instance of the htrDestination object represents a destination identified as hard-to-reach. The decision whether a destination is hard or easy to reach is made either on the basis of external information (e.g. earthquake) or on the answer bid ratio or answer seizure ratio either by the OS or by the resource management of the exchange. The adminstrativeState attribute provides the opportunity to lock the hard-to-reach status so that it can be temporarily disregarded. The HTR status of a destination can also be correlated with circuit subgroups. If this attribute is an empty set, the destination is considered HTR via all possible circuit subgroups.

An instance of htrDestinaton may be either explicitly created by a manager (in the case of manual htr destination) or automatically created by the NE (in case of automatic determination by the agent).

In this model, the mechanism for the recognition of the hard-to-reach status of a destination by the resource management of the exchange is a local matter and therefore is not modelled.

A destination for which no htrDestination is instantiated or which is inhibited (administrativeState = locked) is to be considered as non-Hard-to-Reach.

All instances of the htrDestination object class form the HTR list. See Table 2.

Table 2/Q.823 – Entity – htrDestination

Attributes	M/S	O/M/C
htrDestinationId	S	M
creatorIdentity	S	M
destinationType	S	C
destinationCode	S	M
administrativeState	S	M
tmCircuitEndPointSubgroupList	M	C

The following attributes describe the entity htrDestination:

- htrDestinationId – this attribute is used as the RDN.
- creatorIdentity – identifies who created the object instance.
- destinationType – identifies the type of destination. It is either the nature of address in a seven-bit string according to Recommendation Q.763, or the type of destination as an enumerated list.
- destinationCode – this is a country code, or/and an area code, or/and an exchange, or/and other location number.
- administrativeState – it indicates whether the destinationCode has been inhibited from HTR status. If the value is "locked" it is inhibited, if the value is "unlocked" the code is not inhibited. The default value is "unlocked".
- tmCircuitEndPointSubgroupList – identifies a list of circuit subgroups on which the destinationCode is HTR.

7.1.1.3 Managed element

The managed element managed object class is defined in Recommendation M.3100.

7.1.1.4 Traffic management circuit subgroup (tmCircuitEndPointSubgroup)

The tmCircuitEndPointSubgroup is a subclass of M.3100 circuitEndPointSubgroup. It is used for performance monitoring and controls for traffic management purposes. See Table 3.

Table 3/Q.823 – Entity – tmCircuitEndPointSubgroup

Attributes	M/S	O/M/C
tmSurveillance	S	M

The following attributes describe the entity tmCircuitEndPointSubgroup:

- tmSurveillance – indicates if the tmCircuitEndPointSubgroup is being surveilled for traffic management purposes. When the value is TRUE, the NE will automatically instantiate the corresponding tmCircuitEndPointSubgroupCurrentData. When the value is FALSE, the NE will automatically delete the corresponding tmCircuitEndPointSubgroupCurrentData.

7.1.2 Performance monitoring fragment

In order to perform the traffic management functions, performance data shall be periodically collected about the following managed resources: circuit subgroup, destination, exchange and the traffic control instances. The performance data shall be collected using the Q.822 information model (such as currentData and historyData) and the X.738 simpleScanner. Appendix V provides a tutorial on how performance data can be gathered either through the polled or autonomous mechanism.

7.1.2.1 Current data

The current data managed object class is defined in Recommendation Q.822. This Recommendation provides subclasses that exist for the different monitored entities where the attributes to be monitored are defined. In this Recommendation, a minimum set of attributes is defined. For technology specific or individual implementation which requires more than the minimum set, further subclassing of the currentData is recommended.

7.1.2.2 Circuit endpoint subgroup current data (CircuitEndPointSubgroupCurrentData)

The circuit subgroup current data is a subclass of the currentData object class. It is used for monitoring circuit subgroup related performance data as defined in Recommendation E.502.

The performance monitoring attributes for the circuit subgroup are based on the circuit subgroup directionality characteristic, which can be one-way-out, one-way-in or two-way.

For traffic management purposes, this object class shall not be instantiated, but its subclass defined in this Recommendation or any subclass derived from that may be instantiated. See Table 4.

Table 4/Q.823 – Entity – circuitEndPointSubgroupCurrentData

Attributes	M/S	O/M/C
incomingSeizures	S	M
outgoingBids	S	M
outgoingSeizures	S	M
answeredOutgoingSeizures	S	M
answeredIncomingSeizures	S	O
overflow	S	M
incomingTrafficUsage	S	M
outgoingTrafficUsage	S	M
numberOfAvailCircuits	S	M

The following attributes define the entity circuitEndPointSubgroupCurrentData:

- incomingSeizures – identifies the number of incoming seizures on the circuit subgroup [4.2/E.502, Type 21 b)]. This count is valid for one-way-in or two-way circuit subgroups.
- outgoingBids – identifies the number of successful or unsuccessful attempts to seize a circuit in the circuit subgroup [4.2/E.502, Type 21 a)]. This count is valid for one-way-out or two-way circuit subgroups. It includes bids prior to applying any NM controls.

- outgoingSeizures – identifies the number of outgoing bids which succeeded in obtaining a circuit within the circuit subgroup [4.2.6/E.502, Type 21 b)]. This count is valid for one-way-out or two-way circuit subgroups.
- answeredOutgoingSeizures – identifies the number of outgoing seizures where an answer signal was received [4.2/E.502, Type 21 c)]. This count is valid for one-way-out or two-way circuit subgroups.
- answeredIncomingSeizures – identifies the number of incoming seizures where an answer signal was transmitted back to the preceding exchange. This count is valid for one-way-in or two-way circuit subgroups.
- overflow – identifies the number of outgoing bids overflowing from this circuit subgroup [4.2.6/E.502, Type 21 d)]. This count is valid for one-way-out or two-way circuit subgroups. It will not include calls affected by cancel rerouted overflow, tar from, or cancel from.
- incomingTrafficUsage – identifies the incoming traffic carried in erlang secs. [4.2.6/E.502, Type 21 e)]. This count is valid for one-way-in or two-way circuit subgroups. Typically, this is provided via the sampling method.
- outgoingTrafficUsage – identifies the outgoing traffic carried in erlang secs. [4.2.6/E.502, Type 21 e)]. This count is valid for one-way-out or two-way circuit subgroups. Typically, this is provided via sampling method.
- numberOfAvailCircuits – identifies the number of circuits that can carry traffic including the ones that are currently carrying traffic [4.2.6/E.502, Type 21 f)]. This count is valid for one-way-out, one-way-in and two-way circuit subgroups. Whether this value is provided as a snapshot or as a mean value is left to the implementation due to the normally high frequency of changes to the circuits. Both methods are equivalent.

7.1.2.3 Exchange current data (exchangeCurrentData)

The exchange current data is a subclass of the currentData object class. It is used for monitoring exchange related performance data as defined in Recommendation E.502.

For traffic management purposes, this object class shall not be instantiated, but its subclass defined in this Recommendation or any subclass derived from that shall be instantiated.

The performance monitoring attributes for the exchange is based on the main traffic flow characteristics defined in Figure 4/E.502. See Table 5.

Table 5/Q.823 – Entity – exchangeCurrentData

Attributes	M/S	O/M/C
incomingTraffic	S	M
outgoingTraffic	S	M
transitTraffic	S	M
terminatingTraffic	S	M
originatingTraffic	S	M
internalTraffic	S	M
callsBlockedByLoadShedding	S	M

The following attributes define the entity exchangeCurrentData:

- incomingTraffic – identifies the number of incoming calls in the exchange for which reception of at least one digit has been acknowledged [Figure 4/E.502, traffic flow B, 4.2/E.502, Type 20 a), Type 4 a) and Appendix I].
- outgoingTraffic – identifies the number of outgoing calls (seizures) from the exchange which have successfully seized a circuit [Figure 4/E.502, traffic flow R, 4.2/E.502, Type 10 a), Appendix I].
- transitTraffic – identifies the number of transit calls (seizures) in the exchange [Figure 4/E.502, traffic flow L, 4.2/E.502, Type 6 a)].
- terminatingTraffic – identifies the number of terminating calls (seizures to lines) in the exchange [Figure 4/E.502, traffic flow Q, 4.2/E.502, Type 12 d)].
- originatingTraffic – identifies the number of originating calls (seizures) in the exchange [Figure 4/E.502, traffic flow A, 4.2/E.502, Type 1 a)].
- internalTraffic – identifies the number of internal calls (seizures) in the exchange [Figure 4/E.502, traffic flow F, 4.2/E.502, Type 2 a)].
- callsBlockedByLoadShedding – identifies the number of calls which cannot be handled due to the application of an exchange internal overload protection mechanism [4.2/E.502, Type 20 g)].

7.1.2.4 Observed destination current data (observedDestinationCurrentData)

The observed destination current data is a subclass of the currentData object class. It is used for monitoring destination related performance data as defined in Recommendation E.502.

For traffic management purposes, this object class shall not be instantiated, but its subclass defined in this Recommendation or a subclass derived from that. See Table 6.

Table 6/Q.823 – Entity – observedDestinationCurrentData

Attributes	M/S	O/M/C
bids	S	M
outgoingSeizures	S	M
answeredOutgoingSeizures	S	M
noCircuitsAvailable	S	M
callsAffectedByDcc	S	O

The following attributes define the entity observedDestinationCurrentData:

- bids – identifies the number of bids to the observed destination [4.2/E.502, Type 22 a)].
- outgoingSeizures – identifies the number of successful bids to the observed destination [4.2/E.502, Type 22 b)].
- answeredOutgoingSeizures – identifies the number of outgoing seizures to the observed destination where an answer signal was received [4.2/E.502, Type 22 c)].
- noCircuitsAvailable – identifies the number of bids resulting in unsuccessful calls due to the fact that no free circuits leading to the observed destination were available; i.e. overflow resulting on the final circuit subgroup to that destination. In general, noCircuitsAvailable <= (bids – outgoingSeizures) [4.2.6/E.502, Type 22 d)].
- callsAffectedByDcc – identifies the number of calls to the observed destination which have been affected by destinations code control function [4.2/E.502, Type 22 e)].

7.1.2.5 Traffic control current data (trafficControlCurrentData)

The traffic control current data object class is a subclass of the currentData object class defined in Recommendation Q.822.

It is used for monitoring the effectiveness of a traffic control as defined in Recommendation E.502.

The trafficControlCurrentData is to be provided for each control instance. If there is a need to provide trafficControlCurrentData by control object class, or how a control impacts a managed entity, it can be aggregated at the OS from the corresponding control instance of traffic ControlCurrentData.

For traffic management purposes, this object class shall not be instantiated, but its subclass defined in this Recommendation or any subclass derived from that shall be instantiated. See Table 7.

Table 7/Q.823 – Entity – trafficControlCurrentData

Attributes	M/S	O/M/C
callsOfferedToTrafficControl	S	O
callsAffectedByTrafficControl	S	M

The following attributes define the entity trafficControlCurrentData:

- callsOfferedToTrafficControl – identifies the number of calls which were offered to the traffic control instance.
- callsAffectedByTrafficControl – identifies the number of calls which are actually affected by the traffic control instance.

7.1.2.6 Traffic management circuit subgroup current data (tmCircuitEndPointSubgroupCurrentData)

The traffic management circuit subgroup current data object class is a subclass of the circuitEndPointSubgroupCurrentData object class.

It is used for monitoring circuit subgroup related performance data as defined in Recommendation E.502 in the traffic management context. This object or its subclasses are instantiated for traffic management purposes.

A tmCircuitEndPointSubgroupCurrentData object instance shall generate only one instance of tmCircuitEndPointSubgroupHistoryData.

All tmCircuitEndPointSubgroupCurrentData object instances within the same managed element shall have the same granularity period.

In order to synchronize the granularityPeriod to the next integral time period after the currentData subclasses are instantiated, it is recommended that the value of periodSynchronizationTime (attribute of periodSynchronizationPackage) shall be set to twelve midnight (according to exchange time). See Table 8.

Table 8/Q.823 – Entity – tmCircuitEndPointSubgroupCurrentData

Attributes	M/S	O/M/C
historyRetention	S	M

The following attributes define the entity tmCircuitEndPointSubgroupCurrentData:

- historyRetention – defined in Recommendation Q.822. For the purpose of traffic management, its permitted value is "1".

7.1.2.7 Traffic management exchange current data (tmExchangeCurrentData)

The traffic management exchange performance current data object class is a subclass of the exchangeCurrentData object class.

It is used for monitoring exchange related performance data as defined in Recommendation E.502 in the traffic management context. This object class or its subclasses are instantiated for traffic management purposes.

A tmExchangeCurrentData object instance shall generate only one instance of tmExchangePerformanceHistoryData.

In order to synchronize the granularityPeriod to the next integral time period after the currentData subclasses are instantiated, it is recommended that the value of periodSynchronizationTime (attribute of periodSynchronizationPackage) shall be set to twelve midnight (according to exchange time). See Table 9.

Table 9/Q.823 – Entity – tmExchangeCurrentData

Attributes	M/S	O/M/C
historyRetention	S	M

The following attributes define the entity tmExchangeCurrentData:

- historyRetention – defined in Recommendation Q.822. For the purpose of traffic management, its permitted value is "1".

7.1.2.8 Traffic management observed destination current data (tmObservedDestinationCurrentData)

The traffic management destination current data object class is a subclass of the observedDestinationCurrentData object class.

It is used for monitoring observed destination related performance data as defined in CCITT Recommendation E.502 in the traffic management context. This object or its subclasses are instantiated for traffic management purposes.

A tmObservedDestinationCurrentData object instance shall generate only one instance of tmObservedDestinationHistoryData.

All tmDestinationCurrentData object instances within the same managed element shall have the same granularity period.

In order to synchronize the granularityPeriod to the next integral time period after the currentData subclasses are instantiated, it is recommended that the value of periodSynchronizationTime (attribute of periodSynchronizationPackage) shall be set to twelve midnight (according to exchange time). See Table 10.

Table 10/Q.823 – Entity – tmObservedDestinationCurrentData

Attributes	M/S	O/M/C
historyRetention	S	M

The following attributes define the entity tmObservedDestinationCurrentData:

- historyRetention – defined in Recommendation Q.822. For the purpose of traffic management, its permitted value is "1".

7.1.2.9 Traffic management traffic control current data (tmTrafficControlCurrentData)

The traffic management traffic control current data is a subclass of the trafficControlCurrentData object.

It is used for monitoring the effectiveness of a traffic control as defined in Recommendation E.502 in the traffic management context. This object class or its subclasses are instantiated for traffic management purposes.

A tmTrafficControlCurrentData object instance shall generate only one instance of tmTrafficControlHistoryData. All tmTrafficControlCurrentData object instances within the same managed element shall have the same granularity period.

In order to synchronize the granularityPeriod to the next integral time period after the currentData subclasses are instantiated, it is recommended that the value of periodSynchronizationTime (attribute of periodSynchronizationPackage) shall be set to twelve midnight (according to exchange time). See Table 11.

Table 11/Q.823 – Entity – tmTrafficControlCurrentData

Attributes	M/S	O/M/C
historyRetention	S	M

The following attributes define the entity tmTrafficControlCurrentData:

- historyRetention – defined in Recommendation Q.822. For the purpose of traffic management, its permitted value is "1".

7.1.2.10 History data

The history data managed object class is defined in Recommendation Q.822.

For each of the monitored entities, the historyData instances contain a copy of the performance attributes that are present in the corresponding currentData subclasses object instances at the end of the granularity period.

In line with the Recommendation Q.822, two types of name binding for history data are supported in this Recommendation: historyData-to-currentData and historyData-to-monitored entity. Either of these two name bindings can be used for traffic management purposes, but the name binding should be used consistently for all instances of historyData within a managed system; i.e. all shall be named from either the currentData and its subclasses or all shall be named from the monitored entity and its subclasses.

7.1.2.11 CircuitEndPointSubgroup history data (circuitEndPointSubgroupHistoryData)

The traffic management circuitEndPointSubgroup history data is a subclass of the historyData object class. It is used for monitoring circuitSubgroup related performance data as defined in Recommendation E.502.

The performance monitoring attributes are the same as those in the corresponding currentData. See Table 12.

Table 12/Q.823 – Entity – circuitEndPointSubgroupHistoryData

Attributes	M/S	O/M/C
incomingSeizures	S	M
outgoingBids	S	M
outgoingSeizures	S	M
answeredOutgoingSeizures	S	M
answeredIncomingSeizures	S	C
overflow	S	M
incomingTrafficUsage	S	M
outgoingTrafficUsage	S	M
numberOfAvailCircuits	S	M

The following attributes define the entity circuitEndPointSubgroupHistoryData:

- incomingSeizures – identifies the number of incoming seizures on the circuit subgroup [4.2/E.502 Type 21 b)]. This count is valid for one-way-in or two-way circuit subgroups.
- outgoingBids – identifies the number of successful or unsuccessful attempts to seize a circuit in the circuit subgroup [4.2/E.502, Type 21 a)]. This count is valid for one-way-out or two-way circuit subgroups. It includes bids prior to applying any NM controls.
- outgoingSeizures – identifies the number of outgoing bids which succeeded in obtaining a circuit within the circuit subgroup [4.2.6/E.502, Type 21 b)]. This count is valid for one-way-out or two-way circuit subgroups.
- answeredOutgoingSeizures – identifies the number of outgoing seizures where an answer signal was received [4.2/E.502, Type 21 c)]. This count is valid for one-way-out or two-way circuit subgroups.
- answeredIncomingSeizures – identifies the number of incoming seizures where an answer signal was transmitted back to the preceding exchange. This count is valid for one-way-in or two-way circuit subgroups.
- overflow – identifies the number of outgoing bids overflowing from this circuit subgroup [4.2.6/E.502, Type 21 d)]. This count is valid for one-way-out or two-way circuit subgroup. It will not include calls affected by cancel rerouted overflow, tar from, or cancel from.
- incomingTrafficUsage – identifies the incoming traffic carried in erlang secs. [4.2.6/E.502, Type 21 e)]. This count is valid for one-way-in or two-way circuit subgroup. Typically, this is provided via the sampling method.
- outgoingTrafficUsage – identifies the outgoing traffic carried in erlang secs. [4.2.6/E.502, Type 21 e)]. This count is valid for one-way-out or two-way circuit subgroups. Typically, this is provided via the sampling method.
- numberOfAvailCircuits – identifies the number of circuits that can carry traffic including the ones currently carrying traffic [4.2.6/E.502, Type 21 f)]. This count is valid for one-way-out, one-way-in and two-way circuit subgroups. Whether this value is provided as a snapshot or as a mean value is left to the implementation due to the normally high frequency of changes to the circuits. Both methods are equivalent.

7.1.2.12 Exchange history data (exchangeHistoryData)

The exchange history data is a subclass of the historyData object class. It is used for monitoring exchange related performance data as defined in Recommendation E.502.

The performance monitoring attributes are the same as those in the corresponding currentData.

The performance monitoring attributes for the exchange is based on the main traffic flow characteristics defined in Figure 4/E.502. See Table 13.

Table 13/Q.823 – Entity – exchangeHistoryData

Attributes	M/S	O/M/C
incomingTraffic	S	M
outgoingTraffic	S	M
transitTraffic	S	M
terminatingTraffic	S	M
originatingTraffic	S	M
internalTraffic	S	M
callsBlockedByLoadShedding	S	M

The following attributes define the entity exchangeHistoryData:

- incomingTraffic – identifies the number of incoming calls in the exchange for which reception of at least one digit has been acknowledged [Figure 4/E.502, traffic flow B, 4.2/E.502, Type 20 a), Type 4 a) and Appendix I].
- outgoingTraffic – identifies the number of outgoing calls (seizures) from the exchange which have successfully seized a circuit [Figure 4/E.502, traffic flow R, 4.2/E.502, Type 10 a), Appendix I].
- transitTraffic – identifies the number of transit calls (seizures) in the exchange [Figure 4/E.502, traffic flow L, 4.2/E.502, Type 6 a)].
- terminatingTraffic – identifies the number of terminating calls (seizures to lines) in the exchange [Figure 4/E.502, traffic flow Q, 4.2/E.502, Type 12 d)].
- originatingTraffic – identifies the number of originating calls (seizures) in the exchange [Figure 4/E.502, traffic flow A, 4.2/E.502, Type 1 a)].
- internalTraffic – identifies the number of internal calls (seizures) in the exchange [Figure 4/E.502, traffic flow F, 4.2/E.502, Type 2 a)].
- callsBlockedByLoadShedding – identifies the number of calls which cannot be handled due to the application of an exchange internal overload protection mechanism [4.2/E.502, Type 20 g)].

7.1.2.13 Observed destination (observedDestination)

An instance of observedDestination is instantiated when it is to be monitored for performance management, such as determining hard-to-reach. A destination is a country, an area, an exchange or other location in which the called subscriber is located and that may be specified within the country. It is defined in Annex A/E.410.

A destination can be also selectively observed on a set of circuit subgroups. If the tmCircuitEndPointSubgroupList is not present, the destination shall be observed for all circuit subgroups. See Table 14.

Table 14/Q.823 – Entity – observedDestination

Attributes	M/S	O/M/C
observedDestinationId	S	M
creatorIdentity	S	O
destinationType	S	C
destinationCode	S	M
tmCircuitEndPointSubgroupList	M	C
tmSurveillance	S	M

The following attributes describe the entity observedDestination:

- observedDestinationId – this attribute is used as RDN.
- creatorIdentity – indicates who created the instance of observedDestination.
- destinationType – identifies the type of destination. It is either the nature of address in a seven-bit string according to Recommendation Q.763, or the type of destination as an enumerated list.
- destinationCode – this is a country code, or/and an area code, or/and an exchange, or/and other location number.
- tmCircuitEndPointSubgroupList – identifies a list of circuit subgroups on which the destination shall be observed.
- tmSurveillance – indicates if the observedDestination instance is being surveilled. If the attribute value is TRUE, the tmObservedDestinationCurrentData is automatically instantiated by the NE. If the attribute value is FALSE, tmObservedDestinationCurrentData is automatically deleted by the NE.

7.1.2.14 Observed destination history data (observedDestinationHistoryData)

The observed destination history data is a subclass of the historyData object class. It is used for monitoring destination related performance data as defined in Recommendation E.502.

The performance monitoring attributes are the same as those in the corresponding currentData. See Table 15.

Table 15/Q.823 – Entity – observedDestinationHistoryData

Attributes	M/S	O/M/C
bids	S	M
outgoingSeizures	S	M
answeredOutgoingSeizures	S	M
noCircuitsAvailable	S	M
callsAffectedByDcc	S	C

The following attributes define the entity observedDestinationHistoryData:

- bids – identifies the number of bids to the observed destination [4.2.6/E.502, Type 22 a)].
- outgoingSeizures – identifies the number of successful bids to the observed destination [4.2.6/E.502, Type 22 b)].
- answeredOutgoingSeizures – identifies the number of outgoing seizures to the observed destination where an answer signal was received [4.2.6/E.502, Type 22 c)].

- noCircuitsAvailable – identifies the number of bids resulting in unsuccessful calls due to the fact that no free circuits leading to the observed destination were available; i.e. overflow resulting on the final circuit subgroup to that destination. In general, noCircuitsAvailable <= (bids – outgoingSeizures) [4.2.6/E.502, Type 22 d)].
- callsAffectedByDcc – identifies the number of calls affected by destinations related traffic controls [4.2/E.502, Type 22 e)].

7.1.2.15 Traffic control history data (trafficControlHistoryData)

The traffic control history data object class is a subclass of the historyData object class defined in Recommendation Q.822.

The performance monitoring attributes are the same as those in the corresponding currentData. See Table 16.

Table 16/Q.823 – Entity – trafficControlHistoryData

Attributes	M/S	O/M/C
callsOfferedToTrafficControl	S	C
callsAffectedByTrafficControl	S	M

The following attributes define the entity trafficControlHistoryData:

- callsOfferedToTrafficControl – identifies the number of calls which were offered to the traffic control instance.
- callsAffectedByTrafficControl – identifies the number of calls which are actually affected by the traffic control instance.

7.1.3 Control fragment

Traffic management controls are used to ensure efficient utilization of network capacity and maintain satisfactory performance in the face of fluctuating traffic demands and emergency conditions. Controls may be exercised by specific input from an OS or automatically in response to an internal or external stimulus.

Both for automatic and manual controls, network traffic performance data is crucial to allow determination of when controls are to be applied or removed. Similarly, in order to evaluate the effect a control has on the traffic carrying capacity and performance of the network, data must be obtained that provides this information. Thus, for each entity that may be controlled, e.g. destination, circuit subgroup, etc., traffic performance data must be defined and for each active control a metering capability must exist that will measure the effect of the control. These capabilities are modelled by performance monitoring managed objects.

Manual and automatic traffic controls are modelled in this Recommendation.

Manual controls

Manual controls are activated and removed by specific management actions from an operations system. In the model, activation and removal of manual controls are modelled by creation and deletion of the control managed object. Control managed objects cannot be set to a deactivated state; however, it is possible to set the parameters of the control object to values that will not impact the normal traffic volume or routing characteristics, e.g. the percentage of calls to be blocked could be set to 0.

Automatic controls

Automatic controls are modelled as consisting of a trigger mechanism and a control mechanism. The trigger mechanism defines the condition under which a control is to be applied and automatically emits indications when the control is to be activated. The control mechanism, which is preassigned, specifies the nature and intensity of the control to be applied in response to the trigger indication. Both of these mechanisms are modelled as managed objects. The control mechanism is preset in the NE by the network manager. The trigger object may or may not be allowed to be created by the network manager. The trigger mechanism is always resident in the NE. The trigger mechanism and control mechanism may be located within the same network element (e.g. decentralized ADC) or in two different network elements (e.g. centralized ADC).

Automatic controls and triggers may be activated or deactivated. Automatic controls are activated and deactivated by received trigger indications. Automatic controls and their trigger mechanisms may also be inhibited or de-inhibited in response to management actions by the OS by setting the administrativeState to "locked" or "unlocked" respectively.

When a trigger is locked, it will not emit an indication when the triggering condition becomes true. This will result in any new automatic controls (in this or any other network element) being activated.

When an automatic control is "locked", the control will not become activated even if a trigger indication is received.

To allow the managing operations system to have full control of automatic controls, two status attributes have been introduced into all control objects that can be activated automatically. The autoActivated attribute indicates whether or not a trigger to activate the control is currently in effect. The administrativeState attribute indicates whether or not the control has been locked by the manager. An automatic control will only be active if it is both autoActivated and unlocked. If a control is locked when receiving a trigger indication, it will enter the autoActivated status but will not become active unless and until the manager changes the administrativeState to unlocked.

Appendix III depicts the state transition diagrams under which manual and automatic controls are active and deactive.

7.1.3.1 Automatic Congestion Control (ACC)

The automatic congestion control is a preassigned control which is automatically activated when a congestion trigger is received from the adjacent exchange. It is defined in 4.1/E.412. At the receiving exchanges, all circuit subgroups that terminate to the congested exchange should be subjected to traffic controls which will limit the amount of traffic sent to the congested exchange. Disposition of calls that may be used to throttle traffic are typically cancel or skip.

This control relies on an automatic timer set internally by the NE. When the congestion indicator is received, a timer is started in the receiving exchange. Subsequently, received congestion indicators restart the timer. If no subsequent congestion indicator is received within the timer, the control is automatically deactivated. The selection of the timer value is vendor specific and is not manageable by an OS. The timer aspects for ACC are described in 5.5.2 b)/Q.542.

If a different dispositionOfCalls for each congestion level is desired, a separate acc control object should be instantiated. Furthermore, instances of acc having the same congestion level and controlTmCircuitEndPointSubgroup must use the same dispositionOfCall.

An instance of accAffectedTraffic pointed to by assocAccAffectedTraffic must exist before an acc object class is instantiated.

This is a protective control. It is a subclass of trafficControl object class. See Table 17.

Table 17/Q.823 – Entity – acc

Attributes	M/S	O/M/C
controlTmCircuitEndPointSubgroup	S	M
dispositionOfCalls	S	M
administrativeState	S	M
autoActivated	S	M
assocAccAffectedTraffic	S	M

The following attributes describe the entity acc:

- controlTmCircuitEndPointSubgroup – identifies the circuit subgroup to which this control applies.
- dispositionOfCalls – identifies whether the calls will be skipped to the next available circuit subgroup (if the value is NULL) or will be cancelled (if the value is treatment).
- administrativeState – identifies if the control has been locked by the network manager. When the value is "locked", the control will not be activated. If the value is "unlocked", the control will be activated. When the control is created, it is created with the value of unlocked. The only allowed values for administrative state are "locked" and "unlocked".
- autoActivated – it indicates whether a trigger for this control is outstanding. When this attribute is "TRUE" and administrativeState is unlocked, the automatic control is active. The automatic control is deactivated for all other combinations of the values of the autoActivated and administrativeState attributes. A value change for this attribute will not generate an automatic attribute value change notification.
- assocAccAffectedTraffic – points to the accAffectedTraffic object instance for congestion level 1 and 2. It should be non-null.

7.1.3.2 Automatic Congestion Control (ACC) affected traffic (accAffectedTraffic)

The accAffectedTraffic object identifies the strength of the control for various congestion levels for a combination of routing aspect, destination aspect, origination aspect and/or additional criteria specified.

The origination aspect, destination aspect, routing aspect and additional criteria are all keys in that order.

If an instance of accAffectedTraffic is deleted, all acc object instances referring to this instance must be deleted. See Table 18.

Table 18/Q.823 – Entity – accAffectedTraffic

Attributes	M/S	O/M/C
accAffectedTrafficId	S	M
cl1ResponseCategories	S	M
cl2ResponseCategories	S	M

The following attributes describe the entity accAffectedTraffic:

- accAffectedTrafficId – this attribute is used as the RDN.
- cl1ResponseCategories – identifies in a sequence the percentage strength for a combination of routing aspect, origination aspect, destination aspect and/or other criteria for congestion

level 1. The strength of the control is applied to the combination of one or more of these criteria. It is expected that only one of the percentage choices will be supported.

- cl2ResponseCategories – identifies in a sequence the percentage strength for a combination of routing aspect, origination aspect, destination aspect and/or other criteria for congestion level 2. The strength of the control is applied to the combination of one or more of these criteria. It is expected that only one of the percentage choices will be supported.

7.1.3.3 ACC trigger (accTrigger)

ACC trigger is resident in the congested exchange. When it is unlocked, it will emit congestionLevel indications to the adjacent exchanges for them to automatically activate the preassigned ACC so that traffic towards the congested exchange is throttled. When it is locked, the congested exchange will not emit the congestionLevel indication to the adjacent exchanges.

Note that the congestionLevel is modelled in congestionLevelIndication object class. See Table 19.

Table 19/Q.823 – Entity – accTrigger

Attributes	M/S	O/M/C
accTriggerId	S	M
administrativeState	S	M

The following attributes describe the entity accTrigger:

- accTriggerId – this attribute is used as the RDN.
- administrativeState – it identifies if the congestion indication will be emitted from the congested exchange to adjacent exchanges. If the value is "unlocked" the indication will be emitted; if the value is "locked" the congestion indication will not be emitted. Only "locked" and "unlocked" values are valid for this attribute.

7.1.3.4 Automatic Destination Control (ADC)

ADC is of two types: centralized and decentralized.

In ADC centralized, detection of congestion to a destination code is done at the destination node when the call arrival rate exceeds the threshold set for that destination code. It is described in 4.3/E.412. The adc trigger identifies when the threshold is exceeded, which results in an indication to be sent to each originating node via signalling message indicating that the offered traffic to that destination is to be reduced. The signalling message specifies the details of the control along with the expiration time of the control. Upon receiving the message, the receiving node will instantiate the ADC object.

In the ADC decentralized, destination congestion is detected at the source NE by observing the bids and answers for a selected destination code. It is described in 4.3/E.412. The ADC trigger (adcTrigger) object identifies the destination to be monitored, the threshold value and the strength of the control. When the threshold value is crossed at the source NE, it will automatically instantiate the ADC object. If the ADC control is locked, it will not become active even if the trigger is active. The ADC control will be active only when the administrativeState is set to unlocked state.

The type of ADC is identified by the adcType attribute which can take the value of "centralized" or "decentralized".

This control object is instantiated only by the NE and not by the OS when either it receives a message from the destination node (in the case of centralized) or when the threshold value is crossed at the source NE (in the case of decentralized).

Destination type and destination code are keys to the control in that order.

Exactly one of the strength attributes (adcPercentage or adcContinuousTimer or adcAsynchronousTimer or adcLeakyBucket), must be present when the control object is instantiated. The adc object instance is deleted at the end of time expiration. If a subsequent trigger is received prior to time expiration, the adc instance is replaced with the new values identified in the latest trigger.

This object is a subclass of the trafficControl object class. See Table 20.

Table 20/Q.823 – Entity – adc

Attributes	M/S	O/M/C
adcType	S	M
destinationType	S	C
destinationCode	S	M
adcPercentage	S	C
adcContinuousTimer	S	C
adcAsynchronousTimer	S	C
adcLeakyBucket	S	C
treatment	S	M
timeExpiration	S	M
administrativeState	S	M
autoActivated	S	M

The following attributes describe the entity adc:

- adcType – identifies the type of ADC. It can take the value of "centralized" or "decentralized".
- destinationType – type of destination. It is either the nature of address in a seven-bit string according to Recommendation Q.763, or the type of destination as an enumerated list.
- destinationCode – identifies the destination code of the congested destination.
- percentage – identifies the percentage of calls that will be cancelled as a result of control activation. When this control is active the specified percentage of calls is blocked (cancelled).
- continuousTimer – identifies the continuous timer which includes the number of calls and a time. Once the number of call attempts has been handled within a timer cycle no further attempts are allowed until the timer expires (e.g. 5 calls in 60 seconds).
- asynchronousTimer – identifies the asynchronous timer specifying a time. The timer is set when the call attempt is allowed and no further call attempts are allowed until the timer expires.
- leakyBucket – identifies the leaky bucket which includes the bucket size (maximum allowed counter value) and the decrement per time unit. If the counter exceeds the defined maximum size, the call attempt is cancelled. If the counter is less than the maximum size, the call attempt is allowed and the counter is incremented. The counter is decremented at defined intervals making it possible for new calls to be accepted.
- treatment – indicates the treatment to the call when the call is cancelled as a result of the control (e.g. announcement). The treatment is determined by the NE when the control object is instantiated.

- timeExpiration – identifies the time interval after which the control will automatically be deleted. It acts as a hysteresis to avoid frequent oscillation between instantiating and de-instantiating the control.
- administrativeState – identifies if the control has been inhibited by the network manager. When the value is "locked" the control will not be activated. If the value is "unlocked", the control will be activated. The value lock and unlock are only valid in this object class definition.
- autoActivated – identifies if the trigger for this control is outstanding. Since the control is instantiated only when the trigger is outstanding, the value of autoActivated is always TRUE. The value of FALSE is not valid for this attribute.

7.1.3.5 Automatic Destination Control (ADC) trigger (adcTrigger)

The ADC trigger object is set up by the OS with the necessary attributes so that the ADC object may be automatically instantiated (in the same NE where the adcTrigger resides or adjacent NE) when the trigger threshold value is crossed. OS can specify the destination which is to be monitored by the NE for congestion, the trigger threshold value, the strength for the control and the time expiration.

The adcTrigger can be of type "centralized" or "decentralized" or "both" as identified in adcType attribute.

If the adcType is "centralized", the adcTrigger and the adc control are in two different NEs. A signalling message is sent from the source NE (where the adcTrigger object resides) to all the adjacent NEs to create the adc control.

If the adcType is "decentralized", the adcTrigger and the adc control are in the same NE. The source NE contains both the adcTrigger and the adc control.

If the adcType is "both", the source NE (where the adcTrigger resides) sends a signalling message to all the adjacent NEs (i.e. centralized adc) to create an adc control. It will also instantiate the adc control within itself (decentralized adc).

An OS can instantiate the ADC trigger object, but not the ADC control object. The ADC trigger can also be set to the locked state. When it is in the locked state, it will not automatically send a message to the adjacent exchange or will not instantiate the ADC control even if the trigger threshold is crossed.

Exactly one strength attribute (percentage or continuousTimer or asynchronousTimer or leakyBucket) must be present when this object is instantiated.

Both destination type and destination code are keys to the control in that order. See Table 21.

Table 21/Q.823 – Entity – adcTrigger

Attributes	M/S	O/M/C
adcTriggerId	S	M
adcTriggerType	S	M
destinationType	S	C
destinationCode	S	C
triggerThreshold	S	C
percentage	S	C
continuousTimer	S	C
asynchronousTimer	S	C
leakyBucket	S	C
administrativeState	S	M
timeExpiration	S	M

The following attributes define the entity adcTrigger:

- adcTriggerId – this attribute is used as the RDN.
- adcTriggerType – the type of adcTrigger.
- destinationType – type of destination. It is either the nature of address in a seven-bit string according to Recommendation Q.763, or the type of destination as an enumerated list.
- destinationCode – identifies the destination to be monitored for congestion.
- triggerThreshold- identifies the trigger threshold value. If crossed, the adc control will be instantiated.
- percentage – identifies the percentage of calls that will be cancelled as a result of control activation. When this control is active the specified percentage of calls is blocked (cancelled).
- continuousTimer – identifies the continuous timer which includes the number of calls and a time. Once the number of call attempts has been handled within a timer cycle, no further attempts are allowed until the timer expires (e.g. 5 calls in 60 seconds).
- asynchronousTimer – identifies the asynchronous timer specifying a time. The timer is set when the call attempt is allowed and no further call attempts are allowed until the timer expires.
- leakyBucket – identifies the leaky bucket which includes the bucket size (maximum allowed counter value) and the decrement per time unit. If the counter exceeds the defined maximum size, the call attempt is cancelled. If the counter is less than the maximum size, the call attempt is allowed and the counter is incremented. The counter is decremented at defined intervals making it possible for new calls to be accepted.
- administrativeState – identifies if the trigger has been locked by the network manager. When the value is "locked", the adc control will not be instantiated. If the value is "unlocked" the adc control will be instantiated.
- timeExpiration – identifies the time interval after which the control shall automatically be deleted.

7.1.3.6 Cancel from (cancelFrom)

This control is manually activated on an outgoing circuit subgroup and prohibits traffic from overflowing to the next in-chain circuit subgroups. It is defined in 3.2.1/E.412.

This is a protective circuit subgroup control. It is a subclass of trafficControl object class.

Cancel from control can be used for prohibiting overflow of both direct routed traffic and alternate traffic. Appendix IV provides an explanation of how to prohibit overflow of direct and alternate routed traffic using cancelFrom control. The distinction between these two types of traffic is made via the assignment of appropriate traffic types to the routingAspect attribute.

The controlTmCircuitEndPointSubgroup attribute is the key with the highest priority, followed by each component in the originationAspect, destinationAspect and routingAspect in that order. The more specific value has the priority. See Table 22.

Table 22/Q.823 – Entity – cancelFrom

Attributes	M/S	O/M/C
controlTmCircuitEndPointSubgroup	S	M
routingAspect	S	M
destinationAspect	S	M
originationAspect	S	M
percentage	S	M
treatment	S	M

The following attributes describe the entity cancelFrom:

- controlTmCircuitEndPointSubgroup – identifies the circuit subgroup for which this control object instance applies.
- routingAspect – identifies the routing aspect (direct routed traffic or alternate routed traffic) for which this control is valid. If the attribute value is NULL, the traffic control is applied to all routing aspects.
- destinationAspect – identifies the destination aspect (e.g. HTR) for which this control is valid. If this attribute is NULL, the traffic control is valid for all destination aspects.
- originationAspect – identifies the origin and calling party's category (according to Recommendation Q.763) for which this control is valid. If this attribute is empty sequence, the traffic control is valid for all origination aspects.
- percentage – identifies the percentage of calls that will be cancelled as a result of control activation. When this control is active, the specified percentage of calls is blocked (cancelled).
- treatment – indicates the treatment to the call when the call is cancelled as a result of the control (e.g. announcement).

7.1.3.7 Cancel rerouted overflow (cancelRerouted)

This control prevents additional rerouting or alternate routing of calls which have been already rerouted. Rerouted calls are not allowed to overflow the circuit subgroup to which the cancel rerouted overflow control is applied, while normal overflow traffic is not affected. It is defined in 3.2.4/E.412.

Note that in order to provide this control, each call has to be marked via signalling message whenever a TAR is applied to that call.

This is a protective manual control. It is a subclass of trafficControl object class. See Table 23.

Table 23/Q.823 – Entity – cancelRerouted

Attributes	M/S	O/M/C
controlTmCircuitEndPointSubgroup	S	M
treatment	S	M

The following attributes describe the entity cancelRerouted:

- controlTmCircuitEndPointSubgroup – identifies the circuit subgroup to which this control object instance applies.
- treatment – identifies how the traffic flow impacted by the cancellation will be treated (e.g. announcement).

7.1.3.8 Cancel to (cancelTo)

This control covers both cancellation of direct routing and cancellation of alternative routing to a given controlTmCircuitEndPointSubgroup.

The cancellation of direct routing blocks the amount of direct routed traffic accessing a circuit subgroup. It is defined in 3.1.2/E.412.

The cancellation of alternative routing to control is activated on an outgoing circuit subgroup and prohibits overflow traffic from accessing the controlled circuit subgroup. It is defined in 3.2.1/E.412.

The distinction between these two types is made via the assignment of appropriate routingAspect attribute.

Appendix IV provides an explanation of how to prohibit direct and alternate routed traffic from accessing a circuit subgroup using cancel to control.

The controlTmCircuitEndPointSubgroup attribute is the key with the highest priority, followed by each component in the originationAspect, destinationAspect and routingAspect in that order. The more specific value has the priority.

This is a protective manual control. It is a subclass of the trafficControl object class. See Table 24.

Table 24/Q.823 – Entity – cancelTo

Attributes	M/S	O/M/C
controlTmCircuitEndPointSubgroup	S	M
routingAspect	S	M
destinationAspect	S	M
originationAspect	S	M
percentage	S	M
treatment	S	M

The following attributes describe the entity cancelTo:

- controlTmCircuitEndPointSubgroup – identifies the circuit subgroup for which this control object instance applies.

- routingAspect – identifies the routing aspect (direct routed traffic or alternate routed traffic) for which this control is valid. If the attribute value is NULL, the traffic control is applied to all routing aspects.
- destinationAspect – identifies the destination aspect (e.g. HTR) for which this control is valid. If this attribute has the NULL value, the traffic control is valid for all destination aspects.
- originationAspect – identifies the origin and calling party's category (according to Recommendation Q.763) for which this control is valid. If this attribute is empty sequence, the traffic control is valid for all origination aspects.
- percentage – identifies the percentage of calls that will be cancelled as a result of control activation. When this control is active, the specified percentage of calls is blocked (cancelled).
- treatment – indicates the treatment to the call when the call is cancelled as a result of the control (e.g. announcement).

7.1.3.9 Destination code control (destinationCodeControl)

This manual control bars routing for a specific destination either on a percentage basis (referred to in E.412 as code blocking) or on a rate basis (referred to in Recommendation E.412 as call gapping).

Destination code control can be done on a country code, or/and an area code, or/and an exchange code or/and another location number. It is defined in 3.1.1.1/ E.412.

When this control is active in the code blocking mode, the specified percentage of calls is blocked (cancelled).

When this control is active in the call gapping mode, the rate at which calls are released to the destination will be controlled. It can be specified in one of the three methods: continuous timer or asynchronous timer or leaky bucket.

The strength (code blocking or call gapping) of the destination code control can be within the same object class or can be defined in a separate object class. The former case is applicable when the strength of the control is applied to only one destination. In this case, four possibilities (percentage or continuousTimer or asynchronousTimer or leakyBucket) exist for specifying the strength in the same object class, exactly one of which must be specified. Under this scenario, the single object instance acts as a code control function.

The latter case of defining the strength in a separate object class is applicable only when it is desired to apply the same strength to a group of destinations (leaky bucket is an example of where such behaviour may be useful). In this case, assocOwnerDccGroup attribute is required, which points to the object class containing the strength of the control. Under this scenario, two object classes together represent a code control function.

The following rules are valid independently whether one or two object classes are used to represent a destination code control function:

– It is not possible to create two or more destinationCodeControl object instances with an identical value combination of the key attributes destinationType, destinationCode and originationAspect.

– The destinationCode attribute is the key with the highest priority, followed by the destinationType (if present) and components of the originationAspect attribute in that order.

– When more than one control exists with overlapping digits, the more specific control of the same type applies instead of the less specific control, both of which could affect the same call.

If assocOwnerDccGroup attribute is specified, the associated dccGroup must exist. Instances of this object class belong to the same group if and only if they have the same value of the assocOwnerDccGroup.

This control is a subclass of the trafficControl object class. See Table 25.

Table 25/Q.823 – Entity – destinationCodeControl

Attributes	M/S	O/M/C
destinationType	S	C
destinationCode	S	M
originationAspect	S	M
percentage	S	C
continuousTimer	S	C
asynchronousTimer	S	C
leakyBucket	S	C
assocOwnerDccGroup	S	C
treatment	S	M

The following attributes describe the entity destinationCodeControl:

- destinationType – identifies the type of destination. It is either the nature of address in a seven-bit string according to Recommendation Q.763, or the type of destination as an enumerated list.
- destinationCode – this is a country code, or/and an area code, or/and an exchange, or/and other location number.
- originationAspect – identifies the origin and calling party's category (according to Recommendation Q.763) for which this control is valid. If this attribute is empty sequence, the control is valid for all origination aspects.
- percentage – identifies the percentage of calls that will be cancelled as a result of control activation. When this control is active the specified percentage of calls are blocked (cancelled). This attribute is used only if the strength of the control is within the same object class as the control itself.
- continuousTimer – identifies the continuous timer which includes the number of calls and a time. Once the number of call attempts has been handled within a timer cycle no further attempts are allowed until the timer expires (e.g. 5 calls in 60 seconds). This attribute is used only if the strength of the control is within the same object class as the control itself.
- asynchronousTimer – identifies the asynchronous timer specifying a time. The timer is set when the call attempt is allowed and no further call attempts are allowed until the timer expires. This attribute is used only if the strength of the control is within the same object class as the control itself.
- leakyBucket – identifies the leaky bucket which includes the bucket size (maximum allowed counter value) and the decrement per time unit. If the counter exceeds the defined maximum size, the call attempt is cancelled. If the counter is less than the maximum size, the call attempt is allowed and the counter is incremented. The counter is decremented at defined intervals making it possible for new calls to be accepted. This attribute is used only if the strength of the control is within the same object class as the control itself.

- assocOwnerDccGroup – this attribute is present if the same strength is to be applied to a group of destinations; i.e. this destinationCodeControl instance is an element of a group of destination controls to which the same strength is to be applied. It points to the dccGroup object instance where the strength for the control is defined.
- treatment – indicates the treatment to the call when the call is cancelled as a result of the control (e.g. announcement).

7.1.3.10 Destination code control group (dccGroup)

This object class defines the strength of the destination code control when it is desired to apply the same strength to a group of destinations. Instances of this object class are associated with instances of the destinationCodeControl object class by an (1:n) relationship. This object class by itself does not provide the destination code control function; when associated with the destinationCodeControl instance(s), it represents the destination code control function.

Exactly one of the strength attributes (percentage or continuousTimer or asynchronousTimer or leakyBucket) must be present when the control object is instantiated. See Table 26.

Table 26/Q.823 – Entity – dccGroup

Attributes	M/S	O/M/C
dccGroupId	S	M
percentage	S	C
continuousTimer	S	C
asynchronousTimer	S	C
leakyBucket	S	C

The following attributes describe the entity dccGroup:

- dccGroupId – this attribute is the object identifier. It is used as the RDN.
- percentage – identifies the percentage of calls that will be cancelled as a result of control activation. When this control is active the specified percentage of calls is blocked (cancelled).
- continuousTimer – identifies the continuous timer which includes the number of calls and a time. Once the number of call attempts has been handled within a timer cycle, no further attempts are allowed until the timer expires (e.g. 5 calls in 60 seconds).
- asynchronousTimer – identifies the asynchronous timer specifying a time. The timer is set when the call attempt is allowed and no further call attempts are allowed until the timer expires.
- leakyBucket – identifies the leaky bucket which includes the bucket size (maximum allowed counter value) and the decrement per time unit. If the counter exceeds the defined maximum size, the call attempt is cancelled. If the counter is less than the maximum size, the call attempt is allowed and the counter is incremented. The counter is decremented at defined intervals making it possible for new calls to be accepted.

7.1.3.11 Selective circuit reservation control (scr)

The selective circuit reservation control enables an exchange to automatically give preference to specific traffic attributes over others (e.g. direct routed calls over alternate routed calls) at the moment when circuit congestion is present or imminent. It can be provided with one or two thresholds, with the latter providing greater selectivity. The control is preassigned with thresholds of how many circuits should be kept idle for the various traffic types in a circuit subgroup. When the number of idle circuits or the idle capacity in the circuit subgroup is less than or equal to the reservation threshold, the call is either cancelled or skipped to the next circuit subgroup in the route chain. This control is defined in 4.2/E.412.

SCR control has the following operating variables:

- reservation thresholds;
- control response;
- control action option.

The reservation thresholds and the related control response are determined by the activationThresholds attribute and by the associated scrAffectedTraffic object instance. The control action option for processing of calls denied access to the circuit subgroup is given in the dispositionOfCalls attribute.

When the number of circuits or the idle capacity in the circuit subgroup is less than or equal to the reservation threshold, the exchange shall check the indicated scrAffectedTraffic object instance to determine if the call shall be controlled. The dispositions that may be used to throttle traffic are cancel or skip.

The defined levels in the activationThresholds attribute and the reference given in the associated scr affected traffic attributes shall correspond to a single or multithreshold control.

The scrAffectedTraffic object instance must be present before scr object class can be instantiated.

This is a protective automatic control. It is a subclass of trafficControl object class. See Table 27.

Table 27/Q.823 – Entity – scr

Attributes	M/S	O/M/C
controlTmCircuitEndPointSubgroup	S	M
dispositionOfCalls	S	M
administrativeState	S	M
autoActivated	S	M
activationThresholds	S	M
assocScrAffectedTraffic	S	M

The following attributes define the entity scr:

- controlTmCircuitEndPointSubgroup – identifies the circuit subgroup to which this control applies.
- dispositionOfCalls – identifies whether the calls will be skipped to the next available circuit subgroup (if the value is NULL) or will be cancelled (if the value is treatment).
- administrativeState – it identifies if the control has been locked by the network manager. When the value is "locked", the control will not be activated. If the value is "unlocked", the control will be activated.
- autoActivated – it indicates whether a trigger for this control is outstanding. When this attribute is "TRUE" and administrativeState is "unlocked", the automatic control is active. The automatic control is deactivated for all other combinations of the values of the autoActivated and administrativeState attributes. The value change for this attribute will not generate an automatic attributeValue change notification.

- activationThresholds – identifies the threshold levels for triggering the control.
- assocScrAffectedTraffic – points to the object instance of scrAffectedTraffic for reservation level 1 and 2. It should be non-null.

7.1.3.12 Selective circuit reservation affected traffic (scrAffectedTraffic)

The selective circuit reservation affected traffic object class represents the control response category for the SCR control.

It determines per individual routing aspect, destination aspect, origination aspect and/or additional criteria, the strength of the traffic to be controlled.

If level2ResponseCategories is specified, level1 and level2 activationThresholds must be specified using the same unit (number or percentage).

The origination aspect, destination aspect, routing aspect and additional criteria are keys in that order. See Table 28.

Table 28/Q.823 – Entity – scrAffectedTraffic

Attributes	M/S	O/M/C
scrAffectedTrafficId	S	M
level1ResponseCategories	S	M
level2ResponseCategories	S	C

The following attributes describe the entity scrAffectedTraffic:

- scrAffectedTrafficId – this attribute is used as the RDN.
- level1ResponseCategories – identifies in a sequence the strength for a combination of routing aspect and/or destinationAspect, OriginationAspect and any other additional criteria for reservation level 1.
- level2ResponseCategories – identifies in a sequence the strength for a combination of routing aspect and/or destinationAspect, OriginationAspect and any other additional criteria for reservation level 2.

7.1.3.13 Skip (skip)

This control is activated on an outgoing circuit group and is used to force traffic to the next in-chain circuit subgroup in the routing table. The skip control can affect both direct and alternate routed traffic. It is defined in 3.2.2/E.412.

The controlTmCircuitEndPointSubgroup attribute is the key with the highest priority, followed by each component in the originationAspect, destinationAspect and routingAspect in that order. The more specific value has the priority.

This is a protective manual control. It is a subclass of trafficControl object class. See Table 29.

Table 29/Q.823 – Entity – skip

Attributes	M/S	O/M/C
controlTmCircuitEndPointSubgroup	S	M
routingAspect	S	M
destinationAspect	S	M
originationAspect	S	M
percentage	S	M

The following attributes describe the entity skip:

- controlTmCircuitEndPointSubgroup – identifies the circuit subgroup for which this control object instance applies.
- routingAspect – identifies the routing aspect (direct routed traffic or alternate routed traffic) for which this control is valid. If the attribute value is NULL, the traffic control is applied to all routing aspects.
- destinationAspect – identifies the destination aspect (e.g. HTR) for which this control is valid. If this attribute has a NULL value, the traffic control is valid for all destination aspects.
- originationAspect – identifies the origin and calling party's category (according to Recommendation Q.763) for which this control is valid. If this attribute has a NULL value, the traffic control is valid for all origination aspects.
- percentage – identifies the percentage of calls that will be skipped as a result of control activation.

7.1.3.14 Temporary alternative routing from (tarFrom)

Temporary Alternative Routing (TAR) from is an expansive control which temporarily increases the number of routing possibilities to complete calls by:

– adding new circuit subgroups at the end of routing table; or

– by inserting new circuit subgroups into the routing table between existing circuit subgroups to provide additional overflow paths which are not normally available in the normal routing plan.

These two types of TAR (add at the end or insert between circuit subgroups) are described in 3.2.3/E.412. TarFrom is only applied at circuit subgroup level and impacts all routing tables where the controlTmCircuitEndPointSubgroup appears. The temporary alternative routing circuit groups must terminate on an exchange that has the capability of reaching the final destination.

The effect of this control can be limited to destinations which are on HTR list by setting the destinationAspect attribute, or by explicitly specifying the destination in the destinationCode attribute. If the destination code is an empty string, the control applies to destinations served by that circuit subgroup.

The effect of this control can also be limited to directed or alternated routed traffic.

controlTmCircuitEndPointSubgroup, destinationCode, destinationType (if present), components of originationAspect, components of destinationAspect and components of routingAspect are all keys to the control in that order.

This is expansive manual control. It is a subclass of trafficControl object class. See Table 30.

Table 30/Q.823 – Entity – tarFrom

Attributes	M/S	O/M/C
controlTmCircuitEndPointSubgroup	S	M
newTmCircuitEndPointSubgroups	S	M
routingAspect	S	M
destinationAspect	S	M
originationAspect	S	M
percentage	S	M
destinationType	S	C
destinationCode	S	M
returnAction	S	M

The following attributes describe the entity tarFrom:

- controlTmCircuitEndPointSubgroup – identifies the circuit subgroup to which the tarFrom control applies.
- newTmCircuitEndPointSubgroups – sequence of new circuit groups. At least one element in the sequence must be present.
- routingAspect – identifies the routing aspect (direct routed traffic or alternate routed traffic) for which this control is valid. If the attribute value is NULL, the traffic control is applied to all routing aspects.
- destinationAspect – identifies the destination aspect (e.g. HTR) for which this control is valid. If this attribute has NULL value, the traffic control is valid for all destination aspects.
- originationAspect – identifies the origin and calling party's category (according to Recommendation Q.763) for which this control is valid. If this attribute is empty sequence, the traffic control is valid for all origination aspects.
- percentage – identifies the percentage of calls that will be rerouted as a result of control activation.
- destinationType – identifies the type of destination. It is either the nature of address in a seven-bit string according to Recommendation Q.763, or the type of destination as an enumerated list.
- destinationCode – this is a country code, or/and an area code, or/and an exchange, or/and other location number for which this TAR applies. If the attribute value is an empty string, the control is valid for all destinations on that controlTmCircuitEndPointSubgroup.
- returnAction – identifies the disposition of overflowing traffic from the newTmCircuitEndPointSubgroups. Two possibilities exist:
  - setting the returnAction to NULL inserts newTmCircuitEndPointSubgroups after controlTmCircuitEndPointSubgroup;
  - setting the returnAction to Treatment will cancel the traffic which has overflown from the new TmCircuitEndPointSubgroup.

7.1.3.15 Temporary alternative routing to (tarTo)

Temporary Alternative Routing (TAR) to is an expansive control which temporarily increases the number of routing possibilities to complete calls by adding new circuit subgroups at the beginning of routing tables so that traffic will be first offered to the new circuit subgroup, or by replacing the existing circuit subgroup with a set of new circuit subgroups. One or several circuit groups, which are not normally available in the normal routing plan, but have idle capacity are made available.

These two types of TAR (add at beginning or replace existing circuit subgroup) are described in 3.2.3/E.412. TarTo is only applied at the circuit subgroup level and impacts all routing tables where the tmCircuitEndPointSubgroup is present. The temporary alternative routing circuit groups must terminate on an exchange that has the capability of reaching the final destination.

The effect of this control can be limited to destinations which are on the HTR list by setting the routing aspect attribute, or by explicitly specifying the destination in the destinationCode attribute. If the destinationCode attribute value is an empty string, the control is valid for all destinations on that tmCircuitEndPointSubgroup.

The effect of this control can also be limited to directed or alternated routed traffic.

The controlTmCircuitEndPointSubgroup, destinationCode, destinationType (if present), components of originationAspect, components of destinationAspect and components of routingAspect are all keys to the control in that order.

This is expansive manual control. It is a subclass of trafficControl object class. See Table 31.

Table 31/Q.823 – Entity – tarTo

Attributes	M/S	O/M/C
controlTmCircuitEndPointSubgroup	S	M
newTmCircuitEndPointSubgroups	S	M
routingAspect	S	M
destinationAspect	S	M
originationAspect	S	M
percentage	S	M
destinationType	S	C
destinationCode	S	M
returnAction	S	M

The following attributes describe the entity tarTo:

- controlTmCircuitEndPointSubgroup – identifies the circuit subgroup to which the tarTo control applies.
- newTmCircuitEndPointSubgroups – sequence of new circuit groups. At least one element in the sequence must be present.
- routingAspect – identifies the routing aspect (direct routed traffic or alternate routed traffic) for which this control is valid. If the attribute value is NULL, the traffic control is applied to all routing aspects.
- destinationAspect – identifies the destination aspect (e.g. HTR) for which this control is valid. If this attribute has a NULL value, the traffic control is valid for all destination aspects.
- originationAspect – identifies the origin and calling parties' category (according to Recommendation Q.763) for which this control is valid. If this attribute is empty sequence, the traffic control is valid for all origination aspects.

- percentage – identifies the percentage of calls that will be rerouted as a result of control activation.
- destinationType – identifies the type of destination. It is either the nature of address in a seven-bit string according to Recommendation Q.763, or the type of destination as an enumerated list.
- destinationCode – this is a country code, or/and an area code, or/and an exchange, or/and other location number for which this TAR applies. If the attribute value is an empty string, the control is valid for all destinations on that controlTmCircuitEndPointSubgroup.
- returnAction – identifies the disposition of overflowing traffic from the newTmCircuitEndPointSubgroups. Three possibilities exist:
  - setting returnAction to NULL inserts the newTmCircuitEndPointSubgroups before the controlTmCircuitEndpointSubgroup;
  - setting returnAction to skip replaces the controlTmCircuitEndPointSubgroup with the new TmCircuitEndPointSubgroups;
  - setting returnAction to Treatment will replace the controlTmCircuitEndPointSubgroup and the remaining circuit subgroups in the routing plan with the new TmCircuitEndPointSubgroups.

7.1.3.16 Traffic control (trafficControl)

The traffic control object class is a superclass for all object classes representing traffic controls as defined in Recommendation E.412. This superclass is not instantiated.

In case of several controls of the same type are instantiated on the same managed resource, a mechanism (referred to as key) is described in each traffic control subclass to select an instance of the control. The control with the more specific criteria has precedence.

In case where multiple controls are active, the hierarchy of controls as specified in Recommendation E.412 shall apply. Until such hierarchy of control is added to Recommendation E.412, Appendix X shall be used as guidelines. See Table 32.

Table 32/Q.823 – Entity – trafficControl

Attributes	M/S	O/M/C
trafficControlId	S	M
creatorIdentity	S	O
tmSurveillance	S	M

The following attributes describe the entity trafficControl:

- trafficControlId – this attribute is the object identifier. It is used as the RDN.
- creatorIdentity – identifies who created the object instance.
- tmSurveillance – indicates if the traffic control instance is being surveilled. If the attribute value is TRUE, the tmTrafficControlCurrentData is automatically instantiated by the NE. If the attribute value is FALSE, tmTrafficControlCurrentData is automatically deleted by the NE.

7.1.4 State fragment

The purpose of the state fragment is to provide a mechanism to receive a finite set of status information of various NE events/activities/anomalies which are of interest to a network manager in a condensed form at periodic, but rather short intervals.

7.1.4.1 State indicator (stateIndicator)

This object class defines action and notification containing a bit for each condition. If at any time during the period if the condition has been present, the value of the bit is 1; otherwise the value of the bit is 0. The following bits are defined:

a) exchangeCongestionLevel1: this bit is equal to 1 if during the granularityPeriod the NE experiences congestion level 1.

b) exchangeCongestionLevel2: this bit is equal to 1 if during the granularityPeriod the NE experiences congestion level 2.

c) congestionLevel1Received: this bit is equal to 1 if during the granularityPeriod the NE receives CL1 indicator from any adjacent exchanges.

d) congestionLevel2Received: this bit is set equal to 1 if during the granularityPeriod when the NE receivedCL2 indicator from any adjacent exchanges.

e) scrTriggered: this bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked and autoActivated is TRUE for at least one instance of SCR control.

f) accTriggered: this bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked and autoActivated is TRUE for at least one instance of acc control.

g) protectiveControlActive: this bit is equal to 1 if during the granularityPeriod one or more instances of manual protective controls (i.e. cancelTo, cancelFrom, skip, cancelRerouted) are present in the NE.

h) expansiveControlActive: this bit is equal to 1 if during the granularityPeriod one or more instances of manual expansive controls (i.e. tarTo, tarFrom) are present in the NE.

i) destinationControlActive: this bit is equal to 1 if during the granularityPeriod one or more instances of manual destination controls (i.e. destinationCodeControl functions) are present in the NE.

j) htrDestinationActive: this bit is equal to 1 if during the granularityPeriod one or more instances of htrDestination are present in the NE.

k) circuitEndPointSubgroupAddedOrDeleted: this bit is equal to 1 during the granularityPeriod when a circuit subgroup is added or deleted in the NE.

l) accTransmissionInhibited: this bit is equal to 1 if during the granularityPeriod the administrativeState of accTrigger is set to locked.

m) adcTriggered: this bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked and autoActivated is TRUE for at least one instance of adc control.

The granularityPeriod of this object class shall be less than or equal to the one used for the currentData defined in this Recommendation, and it will be specified in seconds or minutes.

7.1.5 Administrative fragment

7.1.5.1 Event forwarding discriminator

The event forwarding discriminator managed object class is defined in Recommendation X.721.

7.1.5.2 Simple scanner

The simple scanner managed object class is defined in Recommendation X.738.

7.2 Definition of common attributes

This subclause provides the description of all generic attributes used within this information model. The following generic attributes have been identified and their definition can be found within the appropriate standards referenced in this Recommendation:

– relative distinguished name;

– state attributes.

The attributes specific to this information model are defined as part of the object class definitions.

7.2.1 Relative distinguished name

The semantics of the relative distinguished name attribute type are specified in Recommendation X.720. This attribute type is used to identify an instance of a managed object uniquely within the context of its superior managed object. The constraints on the attribute syntax and semantics are specified in Recommendation X.720.

7.2.2 State attributes

State related attributes of managed objects in this information model are comprised of the generic state model attributes as defined in Recommendation X.731 and every specific state attribute that is specified for any of the specific object classes defined in this Recommendation.

7.2.2.1 Administrative state

The semantics of the administrativeState attribute are specified in Recommendation X.731. The syntax of the administrativeState attribute is specified in Recommendation X.721.

7.3 Description of actions

All actions described below are performed on the object classes identified in the table.

Action	Affected object class	Remarks
stateIndicatorAction	stateIndicator	None

7.4 Description of notifications

The following generic notifications are utilized by the information model defined in this Recommendation:

Notification	Defining Recommendations
Object creation notification	X.721 and X.730
Object deletion notification	X.721 and X.730
Attribute value change notification	X.721 and X.730
State change notification	X.721 and X.731
Scan report notification	X.738

The following specific notification is utilized by the information model defined in this Recommendation:

Notification	Affected object class	Remarks
stateIndicatorNotification	stateIndicator	None

8 Functional units

The following functional units are defined in this Recommendation:

A) Traffic Control Functional Unit: this functional unit allows a manager to enable/disable the traffic controls and modify the attributes and triggers of these controls (create/delete/set on subclasses of trafficControl, parameter tables and triggers).

B) Traffic Control Measurements Functional Unit: this functional unit allows a manager to collect the measurements for traffic controls.

C) Exchange Measurements Functional Unit: this functional unit allows a manager to collect the collection of exchange traffic measurements.

D) Circuit EndPoint Subgroup Measurements Functional Unit: this functional unit allows a manager to collect traffic measurements for circuit endpoint subgroups.

E) Destination Measurements Functional Unit: this functional unit allows a manager to collect measurements for destinations.

F) History Data Access Functional Unit: this functional unit allows a manager to perform gets on subclasses of history data.

G) Hard-To-Reach Functional Unit: this functional unit allows a manager to identify destinations as hard-to-reach (create/delete/set on instances of htrDestination).

H) Reference Data Access Functional Unit: this functional units allows a manager to access instances of object classes other than historyData or currentData using GET operation.

I) Autonomous State Indicator Notification Functional Unit: this functional unit requires the support of stateIndicatorNotification defined in this Recommendation.

J) Polled State Indicator Notification Functional Unit: this functional unit requires the support of stateIndicatorAction defined in this Recommendation.

K) Enabling/Disabling Measurements: this functional unit allows a manager to specify the individual entities of a type for which measurements are to be collected (set on tmSurveillance attribute) and/or creation of currentData subclass(es).

In order to ensure interoperability, the manager and the agent have to share at least one common functional unit and a common set of object class.

Appendix VI provides a mapping of traffic management functions (listed in Appendix I) to functional units.

8.1 Functional units from other Recommendations

Besides the above functional units, this Recommendation also supports the following functional units out of other CCITT Recommendations:

i) Scan Stimulation Functional Unit (Recommendation X.738).

ii) Summarization Event Reporting Functional Unit (Recommendation X.738).

iii) Event Report Management Functional Unit (Recommendation X.734).

8.2 Negotiation of functional units

This Recommendation assigns the following object identifier value:

{joint-iso-ccitt ms(9) function Recommendation(0) q(17) q823(823) functionalUnitPackage(1)}

as a value of the ASN.1 type FunctionalUnitPackageId defined in Recommendation X.701 for negotiating the availability of one of the following functional units:

0	Traffic Control Functional Unit
1	Traffic Control Measurements Functional Unit
2	Exchange Measurements Functional Unit
3	Circuit EndPoint Subgroup Measurements Functional Unit
4	Destination Measurements Functional Unit
5	History Data Access Functional Unit
6	Hard-to-Reach Functional Unit
7	Reference Data Access Functional Unit
8	Autonomous State Indicator Notification Functional Unit
9	Polled State Indicator Notification Functional Unit
10	Enabling/Disabling measurements

where the number identifies the bit position in the BIT STRING assigned to the functional units, and the names referencing the functional units.

Within the systems management application context, the mechanism for negotiating the functional units is described by Recommendation X.701.

9 Relationship with other Recommendations

- This Recommendation uses definitions from Recommendation X.721, Definitions of Management Information.
- This Recommendation uses services defined in Recommendation X.730, Object Management Function, for the creation and deletion of managed objects, retrieval and update of attributes, the notifications on object creation, object deletion and attribute value changes.
- This Recommendation uses services defined in Recommendation X.731, State Management Function, for the notification of state changes.
- This Recommendation uses definitions and services from Recommendation X.734, Event Report Management Function.
- This Recommendation uses definitions and services from Recommendation X.738, Summarization Function.
- This Recommendation uses definitions and services from Recommendation Q.822, Performance Management.

10 Conformance

Implementations claiming to conform to this Recommendation shall comply with the conformance requirements as defined in the following subclauses.

10.1 Static conformance

The implementation shall conform to the requirements of this Recommendation in the manager role, the agent role, or both roles. An implementation conformance statement shall claim conformance to at least one role when the ICS is available.

To claim conformance in either a manager or an agent role an implementation shall support at least one of the following capabilities:

- performance monitoring; or
- control.

The performance monitoring requires at least one of functional units i), ii) or F), and at least one B), C), D) or E). Administrative and auditing capability may be optionally included in an implementation.

NOTE – If the auditing capability is negotiated, one may consider to negotiate the Multiple Object Selection Functional Unit and Filter Functional Unit of Recommendation X.710.

Capabilities	**Functional units**	**Object classes**
Performance monitoring	i), ii), F), B), C), D), E)	objects subclassed from historyData for surveillance (e.g. tmObservedDestinationHistoryData, tmExchangeHistoryData, tmTrafficControlHistoryData)
Control application	A)	objects subclassed from trafficControl (e.g. acc, adc, cancelFrom), triggers (e.g. accTrigger), and parameter tables (e.g. accAffectedTraffic)
Auditing	i), H)	objects modelling resources and controls (e.g. circuitEndPointSubgroups, simpleScanner, cancelFrom)
	I), J)	stateIndicator
Administration		objects managed for NTM purposes:
	iii)	efd
	(Note)	simpleScanner
	G)	htrDestination
NOTE – The current Recommendation X.738 does not provide the management of subclasses of scanner at the granularity desired for traffic management. Therefore, the management of subclasses of scanner needs to be negotiated at the implementation time.		

The managed objects supported by the open system for traffic management shall comply with the syntax and semantics of the information model specified in 9.

The implementation shall support the transfer syntax derived from the encoding rules specified in Recommendation X.209 named {joint-iso-ccitt asn1(1) basicEncoding(1)} for the abstract data types referenced by the definitions for which support is claimed.

10.2 Dynamic conformance

This system shall, in the role(s) for which conformance is claimed, support the elements of procedure defined in:

- Recommendation X.730 for the PT-GET, PT-CREATE, PT-DELETE, PT-SET services.
- Recommendation X.730 for the object creation report, object deletion reporting and attribute value change reporting services if the appropriate object class is supported.
- Recommendation X.731 for the state change report service if the appropriate object class is supported.

10.3 Management implementation conformance statement requirements

Any MCS proforma, MICS proforma, MOCS proforma and PICS proforma which conform to this Recommendation shall be technically identical to the proformas specified in future annexes of this Recommendation. They should preserve table numbering and the index numbers of items, and differ only in pagination and page headers.

The supplier of an implementation which is claimed to conform to this Recommendation shall complete a copy of the management conformance summary (MCS) (to be provided) as part of the conformance requirements together with any other ICS proformas referenced as applicable from that MCS. An ICS which conforms to this Recommendation shall:

- describe an implementation which conforms to this Recommendation;
- have been completed in accordance with the instructions for completion given in Recommendation X.724; and
- include the information necessary to uniquely identify both the supplier and the implementation.

Claims of conformance to the management information defined in this Recommendation in managed object classes defined elsewhere shall include the requirements of the MIDS proforma, as will be specified in the future MOCS for the managed object class.

11 Formal object class definitions

11.1 Definition of object classes

11.1.1 ACC

```
acc MANAGED OBJECT CLASS
DERIVED FROM trafficControl;
CHARACTERIZED BY
        "ITU-T Rec. M.3100":stateChangeNotificationPackage,
        accPackage PACKAGE
BEHAVIOUR
        accBehaviour BEHAVIOUR
DEFINED AS
        "The automatic congestion control is a preassigned control which is automatically activated when a
        congestion trigger is received from the adjacent exchange. It is defined in 4.1/E.412. At the receiving
        exchanges, all circuit subgroups that terminate to the congested exchange should be subjected to traffic
        controls which will limit the amount of traffic sent to the congested exchange. Disposition of calls that may
        be used to throttle traffic are typically cancel or skip. This control relies on an automatic timer set
        internally by the NE. When the congestion indicator is received, a timer is started in the receivingexchange.
        Subsequently received congestion indicators restart the timer. If no subsequent congestion indicator is
        received within the timer, the control is automatically deactivated. The selection of the timer value is
        vendor specific and is not manageable by an OS. The timer aspects for ACC are described in
        5.5.2 b)/Q.542. If a different dispositionOfCalls for each congestion level is desired, a separate acc control
        object should be instantiated. Furthermore, instances of acc having the same congestion level and
        tmCircuitEndPointSubgroup must use the same dispositionOfCall.
        The origination aspect, destination aspect, routing aspect and additional criteria, are all keys in that order.
        An instance of accAffectedTraffic pointed to by assocAccAffectedTraffic must exist before an acc object
        class is instantiated.
        This is a protective control. It is a subclass of trafficControl object class.";;
ATTRIBUTES
        controlTmCircuitEndPointSubgroup           GET,
        dispositionOfCalls                         GET-REPLACE,
        "CCITT Rec. X.721:1992":administrativeState
```

```
                                        DEFAULT         VALUE                Q823-TM-
ASN1Module.defaultAdministrativeState
                                   PERMITTED VALUES Q823-TM-ASN1Module.PermittedState
                                             GET-REPLACE,
     autoActivated                 GET,
     assocAccAffectedTraffic                 GET-REPLACE;
     ;;
REGISTERED AS {q823ObjectClass 1};
```

11.1.2 accAffectedTraffic

```
accAffectedTraffic MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721:1992":top;
CHARACTERIZED BY
     "ITU-T Rec. M.3100":objectManagementNotificationsPackage,
     accAffectedTrafficPackage PACKAGE
BEHAVIOUR
     accAffectedTrafficBehaviour BEHAVIOUR
DEFINED AS
     "The accAffectedTraffic object identifies the strength of the control for various congestion levels for a
     combination of routing aspect, destination aspect, origination aspect and/or additional criteria specified.
     If an instance of accAffectedTraffic is deleted, all acc object instances referring to this instance must be
     deleted.";;
ATTRIBUTES
     accAffectedTrafficId          GET,
     cl1ResponseCategories         GET-REPLACE,
     cl2ResponseCategories         GET-REPLACE;
          ;;
REGISTERED AS {q823ObjectClass 2};
```

11.1.3 accTrigger

```
accTrigger MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721:1992":top;
CHARACTERIZED BY
     "ITU-T Rec. M.3100":stateChangeNotificationPackage,
     accTriggerPackage PACKAGE
BEHAVIOUR
     accTriggerBehaviour BEHAVIOUR
DEFINED AS
     "ACC trigger is resident in the congested exchange. When it is unlocked, it will emit congestionLevel
     indications to the adjacent exchanges for them to automatically activate the preassigned ACC so that
     traffic towards the congested exchange is throttled. When it is locked, the congested exchange will not emit
     the congestionLevel indication to the adjacent exchanges. Note that the congestionLevel is modelled in
     congestionLevelIndication object class.";;
ATTRIBUTES
     accTriggerId               GET,
     "CCITT Rec. X.721:1992":administrativeState
          DEFAULT VALUE Q823-TM-ASN1Module.defaultAdministrativeState
          PERMITTED VALUES Q823-TM-ASN1Module.PermittedState
          GET-REPLACE;
     ;;
REGISTERED AS {q823ObjectClass 3};
```

11.1.4 adc

```
adc MANAGED OBJECT CLASS
DERIVED FROM trafficControl;
CHARACTERIZED BY
     "ITU-T Rec. M.3100":stateChangeNotificationPackage,
     adcPackage PACKAGE
```

```
BEHAVIOUR
        adcBehaviour BEHAVIOUR
DEFINED AS
        "ADC is of two types: centralized and decentralized.
        In ADC centralized, detection of congestion to a destination code is done at the destination node when the
        call arrival rate exceeds the threshold set for that destination code. It is described in 4.3/E.412. The adc
        trigger identifies when the threshold is exceeded, which results in an indication to be sent to each
        originating node via signalling message indicating that the offered traffic to that destination is to be
        reduced.
        The signalling message specifies the details of the control along with the expiration time of the control.
        Upon receiving the message, the receiving node will instantiate the ADC object.
        In the ADC decentralized, destination congestion is detected at the source NE by observing the bids and
        answers for a selected destination code. It is described in 4.3/E.412. The ADC Trigger (adcTrigger) object
        identifies the destination to be monitored, the threshold value and the strength of the control. When the
        threshold value is crossed at the source NE, it will automatically instantiate the ADC object. If the ADC
        control is locked, it will not become active even if the Trigger is active. The ADC control will be active only
        when the administrativeState is set to unlocked state.
        The type of ADC is identified by the adcType attribute which can take the value of "centralized" or
        "decentralized".
        This control object is instantiated only by the NE and not by the OS when either it receives a message from
        the destination node (in the case of centralized) or when the  threshold value is crossed at the source NE (in
        the case of decentralized).
        Destination type and destination code are keys to the control in that order.
        Exactly one of the strength attributes (adcPercentage or adcContinuousTimer or adcAsynchronousTimer
        or adcLeakyBucket), must be present when the control object is instantiated. The adc object instance is
        deleted at the end of time expiration. If a subsequent trigger is received prior to time expiration, the adc
        instance is replaced with the new values identified in the latest trigger. The value of autoActivated is always
        TRUE. This object is a subclass of the trafficControl object class.";;
ATTRIBUTES
        adcType                 GET,
        destinationCodeGET,
        treatment               GET,
        "CCITT Rec. X.721:1992":administrativeState
                    DEFAULT VALUE Q823-TM-ASN1Module.defaultAdministrativeState
                    PERMITTED VALUES Q823-TM-ASN1Module.PermittedState
                    GET-REPLACE,
        autoActivated           GET,
        timeExpiration          GET;
;;
CONDITIONAL PACKAGES
        destinationTypePackage
        PRESENT IF "destination type is required to unambiguously identify the destinationCode",
        adcPercentagePackage
        PRESENT IF "adcContinuousTimerPackage and adcAsynchronousTimerPackage and
        adcLeakyBucketPackage are not present, and NE supports it.",
        adcContinuousTimerPackage
        PRESENT IF "adcPercentagePackage and adcAsynchronousTimerPackage and
        adcLeakyBucketPackage are not present, and NE supports it.",
        adcAsynchronousTimerPackage
        PRESENT IF "adcPercentagePackage and adcContinuousTimerPackage and
        adcLeakyBucketPackage are not present, and NE supports it.",
        adcLeakyBucketPackage
        PRESENT IF "adcPercentagePackage and adcContinuousTimerPackage and
        adcAsynchronousTimerPackage are not present, and NE supports it.";
REGISTERED AS {q823ObjectClass 4};
```

11.1.5 adcTrigger

```
adcTrigger MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721:1992":top;
CHARACTERIZED BY
```

```
        "ITU-T Rec. M.3100":objectManagementNotificationsPackage,
        "ITU-T Rec. M.3100":stateChangeNotificationPackage,
        adcTriggerPackage PACKAGE
BEHAVIOUR
        adcTriggerBehaviour BEHAVIOUR
DEFINED AS
        "The ADC trigger object is set up by the OS with the necessary attributes so that the ADC object may be
        automatically instantiated (in the same NE where the adcTrigger resides or adjacent NE) when the trigger
        threshold value is crossed. OS can specify the destination which is to be monitored by the NE for
        congestion, the trigger threshold value, the strength for the control and the time expiration.
        The adcTrigger can be of type "centralized" or "decentralized" or "both" as identified in adcType
        attribute.
        If the adcType is "centralized", the adcTrigger and the adc control are in two different NEs. A signalling
        message is sent from the source NE (where the adcTrigger object resides) to all the adjacent NEs to create
        the adc control.
        If the adcType is ”decentralized", the adcTrigger and the adc control are in the same NE. The source NE
        contains both the adcTrigger and the adc control.
        If the adcType is "both", the source NE (where the adcTrigger resides) sends a signalling message to all the
        adjacent NEs (i.e. centralized adc) to create an adc control. It will instantiate the adc control within itself
        (decentralized adc).
        An OS can instantiate the ADC trigger object, but not the ADC control object. The ADC trigger can also
        be set to the locked state. When it is in the locked state, it will not automatically send a message to the
        adjacent exchange or will not instantiate the ADC control even if the trigger threshold is crossed.
        Exactly one strength attribute (percentage or continuousTimer or asynchronousTimer or leakyBucket)
        must be present when this object is instantiated.
        Both destination type and destination code are keys to the control in that order.";;
ATTRIBUTES
        adcTriggerId            GET,
        adcTriggerType          GET-REPLACE,
        "CCITT Rec. X.721:1992":administrativeState
                    DEFAULT VALUE Q823-TM-ASN1Module.defaultAdministrativeState
                    PERMITTED VALUES Q823-TM-ASN1Module.PermittedState
                    GET-REPLACE,
        timeExpiration          GET-REPLACE;
        ;;
CONDITIONAL PACKAGES
        destinationCodePackage
        PRESENT IF "if the triggering of the adc control is restricted to a predetermined destination.",
        destinationTypePackage
        PRESENT IF "destinationCodePackage is present and destination type is required to
        unambiguously identify the destinationCode.",
        triggerThresholdPackage
        PRESENT IF "a threshold is to be defined by the manager for adcTrigger to activate",
        percentagePackage
        PRESENT IF "continuousTimerPackage and asynchronousTimerPackage and
        leakyBucketPackage are not present, and NE supports it.",
        continuousTimerPackage
        PRESENT IF "percentagePackage and asynchronousTimerPackage and leakyBucketPackage are not
        present, and NE supports it.",
        asynchronousTimerPackage
        PRESENT IF "percentagePackage and continuousTimerPackage and leakyBucketPackage are not present,
        and NE supports it.",
        leakyBucketPackage
        PRESENT IF "percentagePackage and continuousTimerPackage and asynchronousTimerPackage Are not
        present, and NE supports it.";
REGISTERED AS {q823ObjectClass 5};
```

11.1.6 cancelFrom

```
cancelFrom MANAGED OBJECT CLASS
DERIVED FROM trafficControl;
```

```
CHARACTERIZED BY
        cancelFromPackage PACKAGE
BEHAVIOUR
        cancelFromBehaviour BEHAVIOUR
DEFINED AS
        "This control is manually activated on an outgoing circuit subgroup and prohibits traffic from overflowing
        to the next in-chain circuit subgroups. It is defined in 3.2.1/E.412.
        This is a protective circuit subgroup control. It is a subclass of trafficControl object class.
        Cancel From control can be used for prohibiting overflow of both direct routed traffic and alternate
        traffic. The distinction between these two types of traffic is made via the assignment of appropriate traffic
        types to the routingAspect attribute.
        The controlTmCircuitEndPointSubgroup attribute is the key with the highest priority, followed by each
        component in the originationAspect, destinationAspect and routingAspect in that order.
        The more specific value has the priority.";;
ATTRIBUTES
        controlTmCircuitEndPointSubgroup            GET,
        routingAspect                               REPLACE-WITH-DEFAULT
                    DEFAULT VALUE Q823-TM-ASN1Module.defaultRoutingAspects
                                                    GET-REPLACE,
        destinationAspect                           REPLACE-WITH-DEFAULT
                    DEFAULT VALUE Q823-TM-ASN1Module.defaultDestinationAspect
                                                    GET-REPLACE,
        originationAspect                           REPLACE-WITH-DEFAULT
                    DEFAULT VALUE Q823-TM-ASN1Module.defaultOriginAspect
                                                    GET-REPLACE,
        percentage                                  GET-REPLACE,
        treatment                                   GET-REPLACE;
        ;;
REGISTERED AS {q823ObjectClass 6};
```

11.1.7 cancelRerouted

```
cancelRerouted MANAGED OBJECT CLASS
DERIVED FROM trafficControl;
CHARACTERIZED BY
        cancelReroutedPackage PACKAGE
BEHAVIOUR
        cancelReroutedBehaviour BEHAVIOUR
DEFINED AS
        "This control prevents additional rerouting or alternate routing of calls which have been already rerouted.
        Rerouted calls are not allowed to overflow the circuit subgroup to which the Cancel Rerouted Overflow
        control is applied, while normal overflow traffic is not affected. It is defined in 3.2.4/E.412.
        Note that in order to provide this control, each call has to be marked via signalling message whenever a
        TAR is applied to that call.
        This is a protective manual control. It is a subclass of trafficControl object class.";;
ATTRIBUTES
        controlTmCircuitEndPointSubgroup            GET,
        treatment                                   GET-REPLACE
        ;;
;
REGISTERED AS {q823ObjectClass 7};
```

11.1.8 cancelTo

```
cancelTo MANAGED OBJECT CLASS
DERIVED FROM trafficControl;
CHARACTERIZED BY
        cancelToPackage PACKAGE
BEHAVIOUR
        cancelToBehaviour BEHAVIOUR
DEFINED AS
```

```
        "This control covers both cancellation of direct routing and cancellation of alternative routing to a given
        controlTmCircuitEndPointSubgroup.
        The cancellation of direct routing blocks the amount of direct routed traffic accessing a circuit subgroup. It
        is defined in 3.1.2/E.412.
        The cancellation of alternative routing to control is activated on an outgoing circuit subgroup and prohibits
        overflow traffic from accessing the controlled circuit subgroup. It is defined in 3.2.1/E.412.
        The distinction between these two types is made via the assignment of appropriate routingAspect attribute.
        Appendix IV provides an explanation of how to prohibit direct and alternate routed traffic from accessing
        a circuit subgroup using Cancel To control.
        The controlTmCircuitEndPointSubgroup attribute is the key with the highest priority, followed by each
        component in the originationAspect, destinationAspect and routingAspect in that order.
        The more specific value has the priority.
        This is a protective manual control. It is a subclass of the trafficControl object class.";;
ATTRIBUTES
        controlTmCircuitEndPointSubgroup          GET,
        routingAspect                             REPLACE-WITH-DEFAULT
                                  DEFAULT VALUE Q823-TM-ASN1Module.defaultRoutingAspects
                                                  GET-REPLACE,
        destinationAspect                         REPLACE-WITH-DEFAULT
                                  DEFAULT VALUE Q823-TM-ASN1Module.defaultDestinationAspect
                                                  GET-REPLACE,
        originationAspect                         REPLACE-WITH-DEFAULT
                                  DEFAULT VALUE Q823-TM-ASN1Module.defaultOriginAspect
                                                  GET-REPLACE,
        percentage                                GET-REPLACE,
        treatment                                 GET-REPLACE;
        ;;
REGISTERED AS {q823ObjectClass 8};
```

11.1.9 CircuitEndPointSubgroupCurrentData

```
circuitEndPointSubgroupCurrentData MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. Q.822:1994":currentData;
CHARACTERIZED BY
        circuitEndPointSubgroupCurrentDataPackage PACKAGE
BEHAVIOUR
        circuitEndPointSubgroupCurrentDataBehaviour BEHAVIOUR
DEFINED AS
        "The circuit subgroup current data is a subclass of the currentData object class. It is used for monitoring
        circuit subgroup related performance data as defined in Recommendation E.502.
        The performance monitoring attributes for the circuit subgroup are based on the circuit subgroup
        directionality characteristic, which can be one-way-out, one-way-in or two-way.
        For traffic management purposes, this object class shall not be instantiated, but its subclass defined in this
        Recommendation or any subclass derived from that may be instantiated.";;
ATTRIBUTES
        incomingSeizures
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        outgoingBids
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        outgoingSeizures
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        answeredOutgoingSeizures
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        overflow
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        incomingTrafficUsage
        INITIAL VALUE Q823-TM-ASN1Module.initialInteger,
        outgoingTrafficUsage
        INITIAL VALUE Q823-TM-ASN1Module.initialInteger,
        numberOfAvailCircuits
        INITIAL VALUE Q823-TM-ASN1Module.initialGauge;
        ;;
CONDITIONAL PACKAGES
        answeredIncomingSeizuresPackage
        PRESENT IF "this performance measurement is supported by the NE";
REGISTERED AS {q823ObjectClass 9};
```

11.1.10 circuitEndPointSubgroupHistoryData

```
circuitEndPointSubgroupHistoryData MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. Q.822:1994":historyData;
CHARACTERIZED BY
        circuitEndPointSubgroupHistoryDataPackage PACKAGE
BEHAVIOUR
    circuitEndPointSubgroupHistoryDataBehaviour BEHAVIOUR
DEFINED AS
        "The traffic management circuitEndPointSubgroup history data is a subclass of the historyData object
        class. It is used for monitoring circuitSubgroup related performance data as defined in
        Recommendation E.502.
    The performance monitoring attributes are the same as those in the corresponding currentData.";;
ATTRIBUTES
        incomingSeizures              GET,
        outgoingBids                  GET,
        outgoingSeizures              GET,
        answeredOutgoingSeizures      GET,
        overflow                      GET,
        incomingTrafficUsage          GET,
        outgoingTrafficUsage          GET,
        numberOfAvailCircuits         GET;
        ;;
CONDITIONAL PACKAGES
        answeredIncomingSeizureHistoryPackage
        PRESENT IF "this package is present in the corresponding currentData";
REGISTERED AS {q823ObjectClass 10};
```

11.1.11 congestionLevelIndication

```
congestionLevelIndication MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721:1992":top;
CHARACTERIZED BY
        "ITU-T Rec. M.3100":attributeValueChangeNotificationPackage,
        congestionLevelIndicationPackage PACKAGE
BEHAVIOUR
        congestionLevelIndicationBehaviour BEHAVIOUR
DEFINED AS
        "Congestion Level Indication provides an indication of the current congestion level of the managed element
        object instance in which it is contained. This object instance is not allowed to be instantiated by an OS.";;
ATTRIBUTES
        congestionLevelIndicationId   GET,
        congestionLevel               GET;
        ;;
REGISTERED AS {q823ObjectClass 11};
```

11.1.12 destinationCodeControl

```
destinationCodeControl MANAGED OBJECT CLASS
DERIVED FROM trafficControl;
CHARACTERIZED BY
        destinationCodeControlPackage PACKAGE
BEHAVIOUR
        destinationCodeControlBehaviour BEHAVIOUR
DEFINED AS
```

```
        "This manual control bars routing for a specific destination either on a percentage basis (referred to in
        Recommendation E.412 as code blocking) or on a rate basis (referred to in Recommendation E.412 as call
        gapping).
        Destination code control can be done on a country code, or/and an area code, or/and an exchange code
        or/and another location number. It is defined in 3.1.1.1/E.412.
        When this control is active in the code blocking mode, the specified percentage of calls is blocked
        (cancelled).
        When this control is active in the call gapping mode, the rate at which calls are released to the destination
        will be controlled. It can be specified in one of the three methods: continuous timer or asynchronous timer
        or leaky bucket.
        The strength (code blocking or call gapping) of the destination code control can be within the same object
        class or can be defined in a separate object class. The former case is applicable when the strength of the
        control is applied to only one destination. In this case, four possibilities (percentage or continuousTimer or
        asynchronousTimer or leakyBucket) exist for specifying the strength in the same object class, exactly one of
        which must be specified. Under this scenario, the single object instance acts as a code control function.
        The latter case of defining the strength in a separate object class is applicable only when it is desired to
        apply the same strength to a group of destinations (leaky bucket is an example of where such may be
        useful). In this case, assocOwnerDccGroup attribute is required, which points to the object class containing
        the strength of the control. Under this scenario, two object classes together represent a code control
        function.
        The following rules are valid independently whether one or two object classes are used to represent a
        destination code control function:
        – It is not possible to create two or more destinationCodeControl object instances with an identical value
        combination of the key attributes destinationType, destinationCode, and originationAspect.
        – The destinationCode attribute is the key with the highest priority, followed by the destinationType (if
        present) and components of the originationAspect attribute in that order.
        When more than one control exists with overlapping digits, the more specific control of the same type
        applies instead of the less specific control, both of which could affect the same call.
        If assocOwnerDccGroup attribute is specified, the associated dccGroup must exist. Instances of this object
        class belong to the same group if and only if they have the same value of the assocOwnerDccGroup.
        This control is a subclass of the trafficControl object class.";;
ATTRIBUTES
        destinationCode         GET,
        originationAspect                   REPLACE-WITH-DEFAULT
                                DEFAULT VALUE Q823-TM-ASN1Module.defaultOriginAspect
                                            GET-REPLACE,
        treatment                           GET-REPLACE;
        ;;
CONDITIONAL PACKAGES
        destinationTypePackage
        PRESENT IF "destination type is required to unambiguously identify the destinationCode.",
        percentagePackage
        PRESENT IF "continuousTimerPackage and asynchronousTimerPackage and leakyBucketPackage and
        assocOwnerDccGroupPackage are not present, and the NE supports it.",
        continuousTimerPackage
        PRESENT IF "percentagePackage and asynchronousTimerPackage and leakyBucketPackage and
        assocOwnerDccGroupPackage are not present, and the NE supports it.",
        asynchronousTimerPackage
        PRESENT IF "percentagePackage and continuousTimerPackage and leakyBucketPackage and
        assocOwnerDccGroupPackage are not present, and the NE supports it.",
        leakyBucketPackage
        PRESENT IF "percentagePackage and continuousTimerPackage and asynchronousTimerPackage and
        assocOwnerDccGroupPackage are not present, and the NE supports it.",
        assocOwnerDccGroupPackage
        PRESENT IF "percentagePackage and continuousTimerPackage and asynchronousTimerPackage and
        leakyBucketPackage are not present, and the NE supports it.";
REGISTERED AS {q823ObjectClass 12};
```

11.1.13 dccGroup

```
dccGroup MANAGED OBJECT CLASS
```

```
DERIVED FROM "CCITT Rec. X.721:1992":top;
CHARACTERIZED BY
        "ITU-T Rec. M.3100":objectManagementNotificationsPackage,
        dccGroupPackage PACKAGE
BEHAVIOUR
      dccGroupBehaviour BEHAVIOUR
DEFINED AS
        "This object class defines the strength of the destination code control when it is desired to apply the same
        strength to a group of destinations. Instances of this object class are associated with instances of the
        destinationCodeControl object class by an (1:n) relationship. This object class by itself does not provide the
        destination code control function; when associated with the destinationCodeControl instance(s), it
        represents the destination code control function.
        Exactly one of the strength attribute (percentage or continuousTimer or asynchronousTimer or
        leakyBucket) must be present when the control object is instantiated.";;
ATTRIBUTES
    dccGroupId                          GET;
    ;;
CONDITIONAL PACKAGES
        percentagePackage
        PRESENT IF "continuousTimerPackage and asynchronousTimerPackage and leakyBucketPackage are
        not present, and the NE supports it.",
        continuousTimerPackage
        PRESENT IF "percentagePackage and asynchronousTimerPackage and leakyBucketPackage are not
        present, and the NE supports it.",
        asynchronousTimerPackage
        PRESENT IF "percentagePackage and continuousTimerPackage and leakyBucketPackage are not present,
        and the NE supports it.",
        leakyBucketPackage
        PRESENT IF "percentagePackage and continuousTimerPackage and asynchronousTimerPackage are not
        present, and the NE supports it.";
REGISTERED AS {q823ObjectClass 13};
```

11.1.14 exchangeCurrentData

```
exchangeCurrentData MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. Q.822:1994":currentData;
CHARACTERIZED BY
        exchangeCurrentDataPackage PACKAGE
BEHAVIOUR
      exchangeCurrentDataBehaviour BEHAVIOUR
DEFINED AS
        "The exchange current data is a subclass of the currentData object class. It is used for monitoring exchange
        related performance data as defined in Recommendation E.502.
        For traffic management purposes, this object class shall not be instantiated, but its subclass defined in this
        Recommendation or any subclass derived from that shall be instantiated.
        The performance monitoring attributes for the exchange is based on the main traffic flow characteristics
        defined in Figure 4/E.502.";;
ATTRIBUTES
        incomingTraffic
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        outgoingTraffic
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        transitTraffic
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        terminatingTraffic
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        originatingTraffic
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        internalTraffic
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        callsBlockedByLoadShedding
        INITIAL VALUE Q823-TM-ASN1Module.initialCount;
        ;;
REGISTERED AS {q823ObjectClass 14};
```

11.1.15 exchangeHistoryData

```
exchangeHistoryData MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. Q.822:1994":historyData;
CHARACTERIZED BY
        exchangeHistoryDataPackage PACKAGE
BEHAVIOUR
      exchangeHistoryDataBehaviour BEHAVIOUR
DEFINED AS
        "The exchange history data is a subclass of the historyData object class. It is used for monitoring exchange
        related performance data as defined in Recommendation E.502.
        The performance monitoring attributes are the same as those in the corresponding currentData.
        The performance monitoring attributes for the exchange is based on the main traffic flow characteristics
        defined in Figure 4/E.502.";;
ATTRIBUTES
        incomingTraffic                 GET,
        outgoingTraffic                 GET,
        transitTraffic                  GET,
        terminatingTraffic              GET,
        originatingTraffic              GET,
        internalTraffic                 GET,
        callsBlockedByLoadShedding      GET
        ;;
;
REGISTERED AS {q823ObjectClass 15};
```

11.1.16 htrDestination

```
htrDestination MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721:1992":top;
CHARACTERIZED BY
        "ITU-T Rec. M.3100":objectManagementNotificationsPackage,
        "ITU-T Rec. M.3100":stateChangeNotificationPackage,
        creatorPackage,
        htrDestinationPackage PACKAGE
BEHAVIOUR
        htrDestinationBehaviour BEHAVIOUR
DEFINED AS
        "An instance of the htrDestination object represents a destination identified as hard-to-reach. The decision
        whether a destination is hard or easy to reach is made either on the basis of external information (e.g.
        earthquake) or on the answer bid ratio or answer seizure ratio either by the OS or by the resource
        management of the exchange. The administrativeState attribute provides the opportunity to lock the
        hard-to-reach status so that it can be temporarily disregarded. The HTR status of a destination can also be
        correlated with circuit subgroups. If this attribute is an empty set, the destination is considered HTR via all
        possible circuit subgroups.
        An instance of htrDestinaton may be either explicitly created by a manager (in the case of manual htr
        destination) or automatically created by the NE (in case of automatic determination by the agent). In this
        model, the mechanism for the recognition of the hard-to-reach status of a destination by the resource
        management of the exchange is a local matter and therefore is not modelled.
        A destination for which no htrDestination is instantiated or which is inhibited (administrativeState =
        locked) is to be considered as non Hard-to-Reach.
        All instances of the htrDestination object class form the HTR List.";;
ATTRIBUTES
        htrDestinationId                          GET,
        destinationCode             GET,
        "CCITT Rec. X.721:1992":administrativeState
                                    DEFAULT VALUE Q823-TM-ASN1Module.defaultAdministrativeState
```

```
                                        PERMITTED VALUES Q823-TM-ASN1Module.PermittedState
                                                   GET-REPLACE;
        ;;
CONDITIONAL PACKAGES
        destinationTypePackage
        PRESENT IF "destination type is required to unambiguously identify the destination code",
        tmCircuitEndPointSubgroupListPackage
        PRESENT IF "the htrDestination is correlated with tmCircuitEndPointSubgroups";
REGISTERED AS {q823ObjectClass 16};
```

11.1.17 observedDestination

```
observedDestination MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721:1992":top;
CHARACTERIZED BY
        "ITU-T Rec. M.3100":objectManagementNotificationsPackage,
        observedDestinationPackage PACKAGE
BEHAVIOUR
        observedDestinationBehaviour BEHAVIOUR
DEFINED AS
        "An instance of observedDestination is instantiated when it is to be monitored for performance
management, such as determining Hard-to-Reach. A destination is a country, an area, an exchange or other
location in which the called subscriber is located and that may be specified within the country. It is defined in
Annex A/E.410. A destination can be also selectively observed on a set of circuit subgroups. If the
tmCircuitEndPointSubgroupList is not present, the destination shall be observed for all circuit subgroups.";;
ATTRIBUTES
        observedDestinationId       GET,
        destinationCode             GET,
        tmSurveillance              GET-REPLACE;
        ;;
CONDITIONAL PACKAGES
        destinationTypePackage
        PRESENT IF "destination type is required to unambiguously identify the destination code",
        creatorPackage
        PRESENT IF "if an instance supports it",
        tmCircuitEndPointSubgroupListPackage
        PRESENT IF "a destination's performance should be monitored in correlation with certain
        tmCircuitEndPointSubgroups";
REGISTERED AS {q823ObjectClass 17};
```

11.1.18 observedDestinationCurrentData

```
observedDestinationCurrentData MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. Q.822:1994":currentData;
CHARACTERIZED BY
        observedDestinationCurrentDataPackage PACKAGE
BEHAVIOUR
      observedDestinationCurrentDataBehaviour BEHAVIOUR
DEFINED AS
        "The Observed Destination current data is a subclass of the currentData object class. It is used for
        monitoring destination related performance data as defined in Recommendation E.502.
        For traffic management purposes, this object class shall not be instantiated, but its subclass defined in this
        Recommendation or a subclass derived from that.";;
ATTRIBUTES
        bids
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        outgoingSeizures
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        answeredOutgoingSeizures
        INITIAL VALUE Q823-TM-ASN1Module.initialCount,
        noCircuitsAvailable
        INITIAL VALUE Q823-TM-ASN1Module.initialCount;
        ;;
CONDITIONAL PACKAGES
        callsAffectedByDccPackage PRESENT IF "an instance supports it";
REGISTERED AS {q823ObjectClass 18};
```

11.1.19 observedDestinationHistoryData

```
observedDestinationHistoryData MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. Q.822:1994":historyData;
CHARACTERIZED BY
        observedDestinationHistoryDataPackage PACKAGE
BEHAVIOUR
      observedDestinationHistoryDataBehaviour BEHAVIOUR
DEFINED AS
        "The observed destination history data is a subclass of the historyData object class. It is used for
        monitoring destination related performance data as defined in Recommendation E.502.
        The performance monitoring attributes are the same as those in the corresponding currentData.";;
ATTRIBUTES
        bids                          GET,
        outgoingSeizures              GET,
        answeredOutgoingSeizures      GET,
        noCircuitsAvailable           GET;
        ;;
CONDITIONAL PACKAGES
        callsAffectedByDccHistoryPackage
        PRESENT IF "this package is present if it is present in the corresponding instance of currentData";
REGISTERED AS {q823ObjectClass 19};
```

11.1.20 scr

```
scr MANAGED OBJECT CLASS
DERIVED FROM trafficControl;
CHARACTERIZED BY
    "ITU-T Rec. M.3100":stateChangeNotificationPackage,
        scrPackage PACKAGE
BEHAVIOUR
    scrBehaviour BEHAVIOUR
DEFINED AS
        "The Selective Circuit Reservation Control enables an exchange to automatically give preference to specific
        traffic attributes over others (e.g. direct routed calls over alternate routed calls) at the moment later
        providing greater selectivity. The control is preassigned with thresholds of how many circuits should be
        kept idle for the various traffic types in a circuit subgroup. When the number of idle circuits or the idle
        capacity in the circuit subgroup is less than or equal to the reservation threshold, the call is either cancelled
        or skipped to the next circuit subgroup in the route chain. This control is defined in 4.2/E.412.
        SCR control has the following operating variables:
        reservation thresholds;
        control response;
        control action option.
        The reservation thresholds and the related control response are determined by the activationThresholds
        attribute and by the associated scrAffectedTraffic object instance. The control action option for processing
        of calls denied access to the circuit subgroup is given in the dispositionOfCalls attribute.
        When the number of circuits or the idle capacity in the circuit subgroup is less than or equal to the
        reservation threshold, the exchange shall check the indicated scrAffectedTraffic object instance to
        determine If call shall be controlled. The dispositions that may be used to throttle traffic are cancel or skip.
        The defined levels in the activationThresholds attribute and the reference given in the associated scr
        affected traffic attributes shall correspond to a single or multi threshold control.
        The scrAffectedTraffic object instance must be present before scr object class can be instantiated.
        This is a protective automatic control. It is a subclass of trafficControl object class.";;
ATTRIBUTES
        controlTmCircuitEndPointSubgroup    GET,
        dispositionOfCalls              GET-REPLACE,
         "CCITT Rec. X.721:1992":administrativeState
                                        DEFAULT         VALUE               Q823-TM-ASN1Module
defaultAdministrativeState
                                        PERMITTED VALUES Q823-TM-ASN1Module.PermittedState
                                             GET-REPLACE,
        autoActivated                   GET,
        activationThresholds                 GET-REPLACE,
        assocScrAffectedTraffic              GET-REPLACE;
        ;;
REGISTERED AS {q823ObjectClass 20};
```

11.1.21 scrAffectedTraffic

```
scrAffectedTraffic MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721:1992":top;
CHARACTERIZED BY
        "ITU-T Rec. M.3100":objectManagementNotificationsPackage,
        scrAffectedTrafficPackage PACKAGE
BEHAVIOUR
        scrAffectedTrafficBehaviour BEHAVIOUR
DEFINED AS
        "The selective circuit reservation affected traffic object class represents the control response category for
        the SCR control.
        It determines per individual routing aspect, destination aspect, origination aspect and/or additional
        criteria, the strength of the traffic to be controlled.
        If level2ResponseCategories is specified, level1 and level2 activationThresholds must be specified using the
        same unit (number or percentage).
        The origination aspect, destination aspect, routing aspect and additional criteria are keys in that order.";;
ATTRIBUTES
        scrAffectedTrafficId            GET,
        level1ResponseCategories  GET-REPLACE;
        ;;
CONDITIONAL PACKAGES
        level2ResponseCategoriesPackage
        PRESENT IF "instance supports multi threshold selective circuit reservation";
REGISTERED AS {q823ObjectClass 21};
```

11.1.22 skip

```
skip MANAGED OBJECT CLASS
DERIVED FROM trafficControl;
CHARACTERIZED BY
        skipPackage PACKAGE
BEHAVIOUR
        skipBehaviour BEHAVIOUR
DEFINED AS
        "This control is activated on an outgoing circuit group and is used to force traffic to the next in-chain
        circuit subgroup in the routing table. The skip control can affect both direct and alternate routed traffic. It
        is defined in  3.2.2/E.412.
        The controlTmCircuitEndPointSubgroup attribute is the key with the highest priority, followed by each
        component in the originationAspect, destinationAspect and routingAspect in that order. The more specific
        value has the priority.
        This is a protective manual control. It is a subclass of trafficControl object class.";;
ATTRIBUTES
        controlTmCircuitEndPointSubgroup     GET,
        routingAspect                        REPLACE-WITH-DEFAULT
                              DEFAULT VALUE Q823-TM-ASN1Module.defaultRoutingAspects
                                             GET-REPLACE,
        destinationAspect                    REPLACE-WITH-DEFAULT
                              DEFAULT VALUE Q823-TM-ASN1Module.defaultDestinationAspect
```

```
                                                GET-REPLACE,
        originationAspect                       REPLACE-WITH-DEFAULT
                        DEFAULT VALUE Q823-TM-ASN1Module.defaultOriginAspect
                                                GET-REPLACE,
      percentage                                GET-REPLACE
        ;;
;
REGISTERED AS {q823ObjectClass 22};
```

11.1.23 StateIndicator

```
stateIndicator MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. X.739:1993":scanner;
CHARACTERIZED BY
stateIndicatorPackage PACKAGE
BEHAVIOUR
      stateIndicatorBehaviour BEHAVIOUR
DEFINED AS
        "This object class defines action and notification containing a bit for each condition. If at any time during
        the period if the condition has been present, the value of the bit is 1; otherwise the value of the bit is 0. The
        following bits are defined:
        a) exchangeCongestionLevel1: This bit is equal to 1 if during the granularityPeriod the NE experiences
        congestion level 1.
        b) exchangeCongestionLevel2: This bit is equal to 1 if during the granularityPeriod the NE experiences
        congestion level 2.
        c) congestionLevel1Received: This bit is equal to 1 if during the granularityPeriod the NE receives CL1
        indicator from any adjacent exchanges.
        d) congestionLevel2Received: This bit is set equal to 1 if during the granularityPeriod when the NE
        receivedCL2 indicator from any adjacent exchanges.
        e) scrTriggered: This bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked
        and autoActivated is TRUE for at least one instance of SCR control.
        f) accTriggered: This bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked
        and autoActivated is TRUE for at least one instance of acc control.
        g) protectiveControlActive: This bit is equal to 1 if during the granularityPeriod one or more instances of
        manual protective controls (i.e. cancelTo, cancelFrom, skip, cancelRerouted) are present in the NE.
        h) expansiveControlActive: This bit is equal to 1 if during the granularityPeriod one or more instances of
        manual expansive controls (i.e. tarTo, tarFrom) are present in the NE.
        i) destinationControlActive: This bit is equal to 1 if during the granularityPeriod one or more instances of
        manual destination controls (i.e. destinationCodeControl functions) are present in the NE.
        j) htrDestinationActive: This bit is equal to 1 if during the granularityPeriod one or more instances of
        htrDestination are present in the NE.
        k) circuitEndPointSubgroupAddedOrDeleted: This bit is equal to 1 during the granularityPeriod when a
        circuit subgroup is added or deleted in the NE.
        l) accTransmissionInhibited: This bit is equal to 1 if during the granularityPeriod the administrativeState
        of accTrigger is set to locked.
        m) adcTriggered: This bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked
        and autoActivated is TRUE for at least one instance of adc control.
        The granularityPeriod of this object class shall be less than or equal to the one used for the currentData
        defined in this Recommendation, and it will be specified in seconds or minutes.";;;;
ATTRIBUTES
        stateIndicatorId                        GET;
CONDITIONAL PACKAGES
        stateIndicatorActionPackage
        PRESENT IF "stateIndicatorNotificationPackage is not present",
        stateIndicatorNotificationPackage
        PRESENT IF "stateIndicatorActionPackage is not present";
REGISTERED AS {q823ObjectClass 23};
```

11.1.24 tarFrom

```
tarFrom MANAGED OBJECT CLASS
```

```
DERIVED FROM trafficControl;
CHARACTERIZED BY
        tarFromPackage PACKAGE
BEHAVIOUR
        tarFromBehaviour BEHAVIOUR
DEFINED AS
        "Temporary Alternative Routing (TAR) From is an expansive control which temporarily increases the
        number of routing possibilities to complete calls by:
        –       adding new circuit subgroups at the end of routing table, or
        –       by inserting new circuit subgroups into the routing table between existing circuit subgroups to
        provide additional overflow paths which are not normally available in the normal routing plan.
        These two types of TAR (add at the end or insert between circuit subgroups) are described in 3.2.3/E.412.
        returnAction of skip is not valid for tarFrom.
        TarFrom is only applied at circuit subgroup level and impacts all routing tables where the
        controlTmCircuitEndPointSubgroup appears. The temporary alternative routing circuit groups must
        terminate on an exchange that has the capability of reaching the final destination.
        The effect of this control can be limited to destinations which are on HTR List by setting the
        destinationAspect attribute, or by explicitly specifying the destination in the destinationCode. Attribute. If
        the destination code is an empty string, the control applies to destinations served by that circuit subgroup.
        The effect of this control can also be limited to directed or alternated routed traffic.
        controlTmCircuitEndPointSubgroup, destinationType (if present), destinationCode, components of
        originationAspect, components of destinationAspect and components of routingAspect are all keys to the
        control in that order.
        This is expansive manual control. It is a subclass of trafficControl object class.";;
ATTRIBUTES
        controlTmCircuitEndPointSubgroup    GET,
        newTmCircuitEndPointSubgroups       GET-REPLACE,
        routingAspect                       REPLACE-WITH-DEFAULT
                DEFAULT VALUE Q823-TM-ASN1Module.defaultRoutingAspects
                                            GET-REPLACE,
        destinationAspect                   REPLACE-WITH-DEFAULT
                DEFAULT VALUE Q823-TM-ASN1Module.defaultDestinationAspect
                                            GET-REPLACE,
        originationAspect                   REPLACE-WITH-DEFAULT
                DEFAULT VALUE Q823-TM-ASN1Module.defaultOriginAspect
                                            GET-REPLACE,
        percentage                          GET-REPLACE,
        destinationCode     DEFAULT VALUE Q823-TM-ASN1Module.defaultAllDestinationCodes
                            GET,
        returnAction    PERMITTED    VALUES    Q823-TM-ASN1Module.PermittedTarFromReturnAction
                            GET
        ;;
;
CONDITIONAL PACKAGES
        destinationTypePackage
        PRESENT IF "destination type is required to unambiguously identify the destinationCode";
REGISTERED AS {q823ObjectClass 24};
```

11.1.25 tarTo

```
tarTo MANAGED OBJECT CLASS
DERIVED FROM trafficControl;
CHARACTERIZED BY
tarToPackage PACKAGE
BEHAVIOUR
        tarToBehaviour BEHAVIOUR
DEFINED AS
        "Temporary Alternative Routing (TAR) To is an expansive control which temporarily increases the
        number of routing possibilities to complete calls by adding new circuit subgroups at the beginning of
        routing table so that traffic will be first offered to the new circuit subgroup, or by replacing the existing
        circuit subgroup with a set of new circuit subgroups. One or several circuit groups, which are not normally
        available in the normal routing plan, but have idle capacity are made available.
        These two types of TAR (add at beginning or replace existing circuit subgroup) are described in
        3.2.3/E.412. Setting returnAction to NULL inserts the new TmCircuitEndPointSubgroups before the
        controlTmCircuitEndPointSubgroup. Setting returnAction to skip replaces the
        controlTmCircuitEndPointSubgroup with the new TmCircuitEndPointSubgroups. Setting returnAction to
        Treatment will replace the controlTmCircuitEndPointSubgroup and the remaining circuit subgroups in the
        routing plan with the newTmCircuitEndPointSubgroups.
        TarTo is only applied at the circuit subgroup level and impacts all routing tables where the
        controlTmCircuitEndPointSubgroup is present. The temporary alternative routing circuit groups must
        terminate on an exchange that has the capability of reaching the final destination.
        The effect of this control can be limited to destinations which are on the HTR List by setting the routing
        aspect attribute, or by explicitly specifying the destination in the destinationCode Attribute. If the
        destinationCode attribute value is an empty string, the control is valid for all destinations on that
        controlTmCircuitEndPointSubgroup.
        The effect of this control can also be limited to directed or alternated routed traffic.
        controlTmCircuitEndPointSubgroup, destinationType (if present), destinationCode, components of
        originationAspect, components of destinationAspect and components of routingAspect are all keys to the
        control in that order.
        This is expansive manual control. It is a subclass of trafficControl object class.";;
ATTRIBUTES
        controlTmCircuitEndPointSubgroup    GET,
        newTmCircuitEndPointSubgroups       GET-REPLACE,
        routingAspect                       REPLACE-WITH-DEFAULT
                                   DEFAULT VALUE Q823-TM-ASN1Module.defaultRoutingAspects
                                   GET-REPLACE,
        destinationAspect                   REPLACE-WITH-DEFAULT
                                   DEFAULT VALUE Q823-TM-ASN1Module.defaultDestinationAspect
                                            GET-REPLACE,
        originationAspect                   REPLACE-WITH-DEFAULT
                                   DEFAULT VALUE Q823-TM-ASN1Module.defaultOriginAspect
                                            GET-REPLACE,
        percentage                          GET-REPLACE,
        destinationCode
                              DEFAULT VALUE Q823-TM-ASN1Module.defaultAllDestinationCodes
                                   GET
        returnAction                   GET;
        ;;
CONDITIONAL PACKAGES
        destinationTypePackage
        PRESENT IF "destination type is required to unambiguously identify the destinationCode";
REGISTERED AS {q823ObjectClass 25};
```

11.1.26 tmCircuitEndPointSubgroup

```
tmCircuitEndPointSubgroup MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. M.3100":circuitEndPointSubgroup;
CHARACTERIZED BY
        tmCircuitEndPointSubgroupPackage PACKAGE
BEHAVIOUR
      tmCircuitEndPointBehaviour BEHAVIOUR
DEFINED AS
        "The tmCircuitEndPointSubgroup is a subclass of M.3100 circuitEndPointSubgroup. It is used for
        performance monitoring and controls for traffic management purposes.";;
ATTRIBUTES
        tmSurveillance             GET-REPLACE
        ;;
;
REGISTERED AS {q823ObjectClass 26};
```

11.1.27 tmCircuitEndPointSubgroupCurrentData

```
tmCircuitEndPointSubgroupCurrentData MANAGED OBJECT CLASS
DERIVED FROM circuitEndPointSubgroupCurrentData;
CHARACTERIZED BY
"ITU-T Rec. X.739:1993":periodSynchronizationPackage,
tmCircuitEndPointSubgroupCurrentDataPackage PACKAGE
BEHAVIOUR
        tmCircuitEndPointSubgroupCurrentDataBehaviour BEHAVIOUR
DEFINED AS
        "The traffic management circuit subgroup current data object class is a subclass of the
        circuitEndPointSubgroupCurrentData object class.
        It is used for monitoring circuit subgroup related performance data as defined in Recommendation E.502
        in the traffic management context. This object or its subclasses are instantiated for traffic management
        purposes.
        A tmCircuitEndPointSubgroupCurrentData object instance shall generate only one instance of
        tmCircuitEndPointSubgroupHistoryData.
        All tmCircuitEndPointSubgroupCurrentData object instances within the same managed element shall have
        the same granularity period.
        In order to synchronize the granularityPeriod to the next integral time period after the currentData
        subclasses are instantiated, it is recommended that the value of periodSynchronizationTime (attribute of
        periodSynchronizationPackage) shall be set to twelve midnight (according to exchange time).";;
ATTRIBUTES
        "ITU-T Rec. Q.822:1994":historyRetention
        PERMITTED VALUES Q823-TM-ASN1Module.PermittedHistoryRetention
        GET
        ;;
;
REGISTERED AS {q823ObjectClass 27};
```

11.1.28 tmExchangeCurrentData

```
tmExchangeCurrentData MANAGED OBJECT CLASS
DERIVED FROM exchangeCurrentData;
CHARACTERIZED BY
"ITU-T Rec. X.739:1993":periodSynchronizationPackage,
tmExchangeCurrentDataPackage PACKAGE
BEHAVIOUR
      tmExchangeCurrentDataBehaviour BEHAVIOUR
DEFINED AS
        "The traffic management exchange performance current data object class is a subclass of the
        exchangeCurrentData object class.
        It is used for monitoring exchange related performance data as defined in Recommendation E.502 in the
        traffic management context. This object class or its subclasses are instantiated for traffic management
        purposes.
        A tmExchangeCurrentData object instance shall generate only one instance of
        tmExchangePerformanceHistoryData.
        In order to synchronize the granularityPeriod to the next integral time period after the currentData
        subclasses are instantiated, it is recommended that the value of periodSynchronizationTime (attribute of
        periodSynchronizationPackage) shall be set to twelve midnight (according to exchange time).";;
ATTRIBUTES
        "ITU-T Rec. Q.822:1994":historyRetention
        PERMITTED VALUES Q823-TM-ASN1Module.PermittedHistoryRetention
        GET
        ;;
;
REGISTERED AS {q823ObjectClass 28};
```

11.1.29 tmObservedDestinationCurrentData

```
tmObservedDestinationCurrentData MANAGED OBJECT CLASS
DERIVED FROM observedDestinationCurrentData;
CHARACTERIZED BY
        "ITU-T Rec. X.739:1993":periodSynchronizationPackage,
        tmObservedDestinationCurrentDataPackage PACKAGE
BEHAVIOUR
      tmObservedDestinationCurrentDataBehaviour BEHAVIOUR
DEFINED AS
        "The traffic management destination current data object class is a subclass of the
        observedDestinationCurrentData object class.
        It is used for monitoring observed destination related performance data as defined in Recommendation
        E.502 in the traffic management context. This object or its subclasses are instantiated for traffic
        management purposes.
        A tmObservedDestinationCurrentData object instance shall generate only one instance of
        tmObservedDestinationHistoryData.
        All tmDestinationCurrentData object instances within the same managed element shall have the same
        granularity period.
        In order to synchronize the granularityPeriod to the next integral time period after the currentData
        subclasses are instantiated, it is recommended that the value of periodSynchronizationTime (attribute of
        periodSynchronizationPackage) shall be set to twelve midnight (according to exchange time).";;
ATTRIBUTES
        "ITU-T Rec. Q.822:1994":historyRetention
        PERMITTED VALUES Q823-TM-ASN1Module.PermittedHistoryRetention
        GET
        ;;
;
REGISTERED AS {q823ObjectClass 29};
```

11.1.30 tmTrafficControlCurrentData

```
tmTrafficControlCurrentData MANAGED OBJECT CLASS
DERIVED FROM trafficControlCurrentData;
CHARACTERIZED BY
        "ITU-T Rec. X.739:1993":periodSynchronizationPackage,
        tmTrafficControlCurrentDataPackage PACKAGE
BEHAVIOUR
     tmTrafficControlCurrentDataBehaviour BEHAVIOUR
DEFINED AS
        "The traffic management traffic control current data is a subclass of the trafficControlCurrentData object.
        It is used for monitoring the effectiveness of a traffic control as defined in Recommendation E.502 in the
        traffic management context. This object class or its subclasses are instantiated for traffic management
        purposes.
        A tmTrafficControlCurrentData object instance shall generate only one instance of
        tmTrafficControlHistoryData.
        All tmTrafficControlCurrentData object instances within the same managed element shall have the same
        granularity period.
        In order to synchronize the granularityPeriod to the next integral time period after the currentData
        subclasses are instantiated, it is recommended that the value of periodSynchronizationTime (attribute of
        periodSynchronizationPackage) shall be set to twelve midnight (according to exchange time).";;
ATTRIBUTES
        "ITU-T Rec. Q.822:1994":historyRetention
        PERMITTED VALUES Q823-TM-ASN1Module.PermittedHistoryRetention
        GET
        ;;
;
REGISTERED AS {q823ObjectClass 30};
```

11.1.31 trafficControl

```
trafficControl MANAGED OBJECT CLASS
DERIVED FROM "CCITT Rec. X.721:1992":top;
CHARACTERIZED BY
        "ITU-T Rec. M.3100":objectManagementNotificationsPackage,
        trafficControlPackage PACKAGE
BEHAVIOUR
        trafficControlBehaviour BEHAVIOUR
DEFINED AS
        "The Traffic Control object class is a superclass for all object classes representing traffic controls as
        defined in Recommendation E.412. This superclass is not instantiated.
        In case of several controls of the same type are instantiated on the same managed resource, a mechanism
        (referred to as key) is described in each traffic control subclass to select an instance of the control. The
        control with the more specific criteria has precedence.";;
ATTRIBUTES
        trafficControlId            GET,
        tmSurveillance              GET-REPLACE;
        ;;
CONDITIONAL PACKAGES
        creatorPackage
        PRESENT IF "an instance supports it";
REGISTERED AS {q823ObjectClass 31};
```

11.1.32 trafficControlCurrentData

```
trafficControlCurrentData MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. Q.822:1994":currentData;
CHARACTERIZED BY
        trafficControlCurrentDataPackage PACKAGE
BEHAVIOUR
    trafficControlCurrentDataBehaviour BEHAVIOUR
DEFINED AS
        "The traffic control current data object class is a subclass of the currentData object class defined in
        Recommendation Q.822.
        It is used for monitoring the effectiveness of a traffic control as defined in Recommendation E.502.
        The trafficControlCurrentData is to be provided for each control instance. If there is a need to provide
        trafficControlCurrentData by control object class, or how a control impacts a managed entity, it can be
        aggregated at the OS from the corresponding control instance of traffic ControlCurrentData.
        For traffic management purposes, this object class shall not be instantiated, but its subclass defined in this
        Recommendation or any subclass derived from that shall be instantiated.";;
ATTRIBUTES
        callsAffectedByTrafficControl
        INITIAL VALUE Q823-TM-ASN1Module.initialCount;
        ;;
CONDITIONAL PACKAGES
        callsOfferedPackage
        PRESENT IF "an instance supports it";
REGISTERED AS {q823ObjectClass 32};
```

11.1.33 trafficControlHistoryData

```
trafficControlHistoryData MANAGED OBJECT CLASS
DERIVED FROM "ITU-T Rec. Q.822:1994":historyData;
CHARACTERIZED BY
        trafficControlHistoryDataPackage PACKAGE
BEHAVIOUR
    trafficControlHistoryDataBehaviour BEHAVIOUR
DEFINED AS
```

```
        "The traffic control history data object class is a subclass of the historyData object class defined in ITU-T
        Recommendation Q.822.
        The performance monitoring attributes are same as those in the corresponding currentData.";;
ATTRIBUTES
callsAffectedByTrafficControl GET;
        ;;
CONDITIONAL PACKAGES
        callsOfferedHistoryPackage
        PRESENT IF "this package is present if it is present in the corresponding instance of  currentData";
REGISTERED AS {q823ObjectClass 33};
```

11.2 Definition of name binding

The historyData-currentData name binding is defined in Recommendation Q.822. It is used for the currentData and historyData subclasses defined in this Recommendation.

```
accAffectedTraffic-managedElement NAME BINDING
        SUBORDINATE OBJECT CLASS accAffectedTraffic;
        NAMED BY
        SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
        WITH ATTRIBUTE accAffectedTrafficId;
        BEHAVIOUR
        accAffectedTrafficContainmentBehaviour                BEHAVIOUR
        DEFINED AS "the delete operation will fail if one or more instance of acc points to this instance.";;
        CREATE WITH-REFERENCE-OBJECT,
                WITH-AUTOMATIC-INSTANCE-NAMING;
        DELETE;
        REGISTERED AS {q823NameBinding 1};

accTrigger-managedElement NAME BINDING
        SUBORDINATE OBJECT CLASS accTrigger AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
        WITH ATTRIBUTE accTriggerId;
        REGISTERED AS {q823NameBinding 2};

adcTrigger-managedElement NAME BINDING
        SUBORDINATE OBJECT CLASS adcTrigger AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
        WITH ATTRIBUTE adcTriggerId;
        CREATE WITH-REFERENCE-OBJECT,
                WITH-AUTOMATIC-INSTANCE-NAMING;
        DELETE;
        REGISTERED AS {q823NameBinding 3};

autoHTRDestination-managedElement NAME BINDING
        SUBORDINATE OBJECT CLASS htrDestination AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100": managedElement AND SUBCLASSES;
        WITH ATTRIBUTE htrDestinationId;
        BEHAVIOUR
        autoHTRDestinationBehaviour            BEHAVIOUR
        DEFINED AS "This name binding is used when an instance of htrDestination or its subclasses is
                created by the agent after determining by a local means that a destination is hard-to-reach.";;
        REGISTERED AS {q823NameBinding 10};

circuitEndPointSubgroupHistoryData-tmCircuitEndPointSubgroup NAME BINDING
        SUBORDINATE OBJECT CLASS circuitEndPointSubgroupHistoryData AND
SUBCLASSES;
```

```
	NAMED BY
	SUPERIOR OBJECT CLASS tmCircuitEndPointSubgroup AND
SUBCLASSES;
	WITH ATTRIBUTE "ITU-T Rec. Q.822":historyDataId;
	DELETE DELETES-CONTAINED-OBJECTS;
	REGISTERED AS {q823NameBinding 4};

congestionLevelIndication-managedElement NAME BINDING
	SUBORDINATE OBJECT CLASS congestionLevelIndication AND SUBCLASSES;
	NAMED BY
	SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
	WITH ATTRIBUTE congestionLevelIndicationId;
	BEHAVIOUR
	congestionLevelIndicationToMEBehaviour			BEHAVIOUR
	DEFINED AS "this object class is inherently created by NE";;
	REGISTERED AS {q823NameBinding 5};

dccGroup-managedElement NAME BINDING
	SUBORDINATE OBJECT CLASS dccGroup AND SUBCLASSES;
	NAMED BY
	SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
	WITH ATTRIBUTE dccGroupId;
	BEHAVIOUR
	dccGroupContainmentBehaviour		BEHAVIOUR
	DEFINED AS
	"delete operation will fail if one or more instances of destinationCodeControl points to this object
	instance.";;
	CREATE WITH-REFERENCE-OBJECT,
	WITH-AUTOMATIC-INSTANCE-NAMING;
	DELETE;
	REGISTERED AS {q823NameBinding 6};

exchangeHistoryData-managedElement NAME BINDING
	SUBORDINATE OBJECT CLASS exchangeHistoryData AND SUBCLASSES;
	NAMED BY
	SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
	WITH ATTRIBUTE "ITU-T Rec. Q.822:1994":historyDataId;
	DELETE DELETES-CONTAINED-OBJECTS;
	REGISTERED AS {q823NameBinding 8};

htrDestination-managedElement NAME BINDING
	SUBORDINATE OBJECT CLASS htrDestination AND SUBCLASSES;
	NAMED BY
	SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
	WITH ATTRIBUTE htrDestinationId;
	BEHAVIOUR
	htrDestinationToMEBehaviour		BEHAVIOUR
	DEFINED AS "this name binding is used when the object is explicitly created by the manager";;
	CREATE WITH-REFERENCE-OBJECT,
		WITH-AUTOMATIC-INSTANCE-NAMING;
	DELETE;
	REGISTERED AS {q823NameBinding 9};

observedDestination-managedElement NAME BINDING
	SUBORDINATE OBJECT CLASS observedDestination AND SUBCLASSES ;
	NAMED BY
	SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
	WITH ATTRIBUTE observedDestinationId;
	CREATE WITH-REFERENCE-OBJECT,
		WITH-AUTOMATIC-INSTANCE-NAMING;
	DELETE DELETES-CONTAINED-OBJECTS;
```

```
        REGISTERED AS {q823NameBinding 11};

observedDestinationHistoryData-observedDestination NAME BINDING
        SUBORDINATE OBJECT CLASS observedDestinationHistoryData AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS observedDestination AND SUBCLASSES;
        WITH ATTRIBUTE "ITU-T Rec.Q.822:1994":historyDataId;
        DELETE DELETES-CONTAINED-OBJECTS;
        REGISTERED AS {q823NameBinding 12};

scrAffectedTraffic-managedElement NAME BINDING
        SUBORDINATE OBJECT CLASS scrAffectedTraffic AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
        WITH ATTRIBUTE scrAffectedTrafficId;
        BEHAVIOUR
        scrAffectedContainmentBehaviour         BEHAVIOUR
        DEFINED AS "delete operation will fail if one or more instances of scr points to this object
                instance.";;
        CREATE WITH-REFERENCE-OBJECT,
                WITH-AUTOMATIC-INSTANCE-NAMING;
        DELETE DELETES-CONTAINED-OBJECTS;
        REGISTERED AS {q823NameBinding 13};

simpleScanner-managedElement NAME BINDING
        SUBORDINATE OBJECT CLASS "ITU-T Rec. X.738:1993":simpleScanner AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":
        managedElement AND SUBCLASSES;
        WITH ATTRIBUTE "ITU-T Rec. X.739:1993":scannerId;
        CREATE WITH-REFERENCE-OBJECT,
                WITH-AUTOMATIC-INSTANCE-NAMING;
        DELETE ONLY-IF-NO-CONTAINED-OBJECTS;
        REGISTERED AS {q823NameBinding 14};

stateIndicator-managedElement NAME BINDING
        SUBORDINATE OBJECT CLASS stateIndicator AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
        WITH ATTRIBUTE stateIndicatorId;
        BEHAVIOUR
        stateIndicatorContainmentBehaviour      BEHAVIOUR
        DEFINED AS "this object class is inherently created by the NE.";;
        REGISTERED AS {q823NameBinding 15};

tmCircuitEndPointSubgroupCurrentData-tmCircuitEndPointSubgroup NAME BINDING
        SUBORDINATE OBJECT CLASS tmCircuitEndPointSubgroupCurrentData AND
        SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS tmCircuitEndPointSubgroup AND SUBCLASSES;
    WITH ATTRIBUTE "ITU-T Rec. X.739:1993":scannerId;
        BEHAVIOUR
        tmircuitEndPointSubgroupCurrrentDataContainmentBehaviour     BEHAVIOUR
        DEFINED AS "instance of tmCircuitEndPointSubgroupCurrentData will be automatically instantiated
        when the tmSurveillance of the superior object class, tmCircuitEndPointSubgroup is set to TRUE. Instance
        of tmCircuitEndPointSubgroupCurrentData will be automatically deleted when the tmSurveillance
        attribute of the superior object class, tmCircuitEndPointSubgroup is set to FALSE. Instances of this class
        may also be inherently created by the NE.";;
        REGISTERED AS {q823NameBinding 16};

        -- NOTE – the value of currentData attributes are determined by the NE using local means.
```

```
tmExchangeCurrentData-managedElement NAME BINDING
        SUBORDINATE OBJECT CLASS tmExchangeCurrentData AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
        WITH ATTRIBUTE "ITU-T Rec. X.739:1993":scannerId;
        BEHAVIOUR
        tmExchangeCurrentDataContainmentBehaviour          BEHAVIOUR
        DEFINED AS "only one instance of tmExchangeCurrentData is allowed within managedElement Instance
        of this class is inherently created by the NE";;
        CREATE
                WITH-AUTOMATIC-INSTANCE-NAMING;
        DELETE DELETES-CONTAINED-OBJECTS;
        REGISTERED AS {q823NameBinding 17};

tmObservedDestinationCurrentData-observedDestination NAME BINDING
        SUBORDINATE OBJECT CLASS tmObservedDestinationCurrentData AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS observedDestination AND SUBCLASSES;
        WITH ATTRIBUTE "ITU-T Rec. X.739:1993":scannerId;
        BEHAVIOUR
        tmObservedDestinationCurrrentDataContainmentBehaviour          BEHAVIOUR
        DEFINED AS "instance of tmObservedDestinationCurrentData will be automatically instantiated when
        the tmSurveillance of the superior object class, observedDestination is set to TRUE. Instance of
        tmObservedDestinationCurrentData will be automatically deleted when the tmSurveillance attribute of the
        superior object class, observedDestination is set to FALSE. Instances of this class may also be inherently
        created by the NE";;
        REGISTERED AS {q823NameBinding 18};

        -- NOTE – the value of currentData attributes are determined by the NE using local means.

tmTafficControlCurrentData-trafficControl NAME BINDING
        SUBORDINATE OBJECT CLASS tmTrafficControlCurrentData AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS trafficControl AND SUBCLASSES;
        WITH ATTRIBUTE "ITU-T Rec. X.739:1993":scannerId;
        BEHAVIOUR
        tmTrafficControlCurrentDataContainmentBehaviour          BEHAVIOUR
        DEFINED AS "instance of tmTrafficControlCurrentData will be automatically instantiated when the
        tmSurveillance of the superior object class, trafficControl is set to TRUE. Instance of tmTraffic
        ControlCurrentData will be automatically deleted when the tmSurveillance attribute of the superior object
        class, trafficControl is set to FALSE. Instances of this class may also be inherently created by the NE ";;
        REGISTERED AS {q823NameBinding 19};

        -- NOTE – the value of currentData attributes are determined by the NE using local means.

trafficControl-managedElement NAME BINDING
        SUBORDINATE OBJECT CLASS trafficControl AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS "ITU-T Rec. M.3100":managedElement AND SUBCLASSES;
        WITH ATTRIBUTE trafficControlId;
        CREATE WITH-REFERENCE-OBJECT,
                WITH-AUTOMATIC-INSTANCE-NAMING;
        DELETE DELETES-CONTAINED-OBJECTS;
        REGISTERED AS {q823NameBinding 20};

trafficControlHistoryData-trafficControl NAME BINDING
        SUBORDINATE OBJECT CLASS trafficControlHistoryData AND SUBCLASSES;
        NAMED BY
        SUPERIOR OBJECT CLASS trafficControl AND SUBCLASSES;
        WITH ATTRIBUTE "ITU-T Rec. Q.822:1994":historyDataId;
        DELETE DELETES-CONTAINED-OBJECTS;
        REGISTERED AS {q823NameBinding 21};
```

11.3 Definition of packages

```
adcAsynchronousTimerPackage PACKAGE
BEHAVIOUR
        adcAsynchronousTimerBehaviour           BEHAVIOUR
DEFINED AS
        "identifies the asynchronous timer specifying a time value. The timer is set when the call attempt is allowed
        and no further call attempts are allowed until the timer expires.";;
ATTRIBUTES
        asynchronousTimer       GET;
REGISTERED AS {q823Package 1};

adcContinuousTimerPackage PACKAGE
BEHAVIOUR
        adcContinuousTimerBehaviour   BEHAVIOUR
DEFINED AS
        "identifies the continuous timer which includes the number of calls and a time. Once the number of call
        attempts has been handled with a timer cycle no further attempts are allowed until the time expires
        (e.g. 5 calls in 60 seconds.)";;
ATTRIBUTES
        continuousTimer         GET;
REGISTERED AS {q823Package 2};

adcLeakyBucketPackage PACKAGE
BEHAVIOUR
        adcLeakyBucketTimerBehaviour BEHAVIOUR
DEFINED AS
        "identifies the leaky bucket which includes the bucket size (maximum allowed counter value) and the
        decrement per time unit. If the counter exceeds the defined maximum size, the call attempt is cancelled. If
        the count value is less than the maximum size, the call attempt is allowed and the counter is incremented.
        The counter is decremented at defined intervals making it possible for new calls to be accepted.";;
ATTRIBUTES
        leakyBucket         GET;
REGISTERED AS {q823Package 3};

adcPercentagePackage PACKAGE
BEHAVIOUR
        adcPercentageBehaviour          BEHAVIOUR
DEFINED AS
        "identifies the strength of the control in percentages. It is expected that only one of the choices of
        percentage for the strength will be supported.";;
ATTRIBUTES
        percentage          GET;
REGISTERED AS {q823Package 4};

answeredIncomingSeizuresPackage PACKAGE
BEHAVIOUR
        answeredIncomingSeizuresPackageBehaviour BEHAVIOUR
DEFINED AS
        "Identifies the number of incoming seizures where an answer signal was transmitted back to the preceding
        exchange.";;
ATTRIBUTES
        answeredIncomingSeizures INITIAL VALUE Q823-TM-ASN1Module.initialCount;
REGISTERED AS {q823Package 5};

answeredIncomingSeizureHistoryPackage PACKAGE
BEHAVIOUR
        answeredIncomingSeizureHistoryPackageBehaviour BEHAVIOUR
```

```
DEFINED AS
        "Identifies the number of incoming seizures where an answer signal was transmitted back to the preceding
        exchange.";;
ATTRIBUTES
        answeredIncomingSeizures        GET;
REGISTERED AS {q823Package 6};

assocOwnerDccGroupPackage PACKAGE
BEHAVIOUR
        assocOwnerDccGroupBehaviour BEHAVIOUR
DEFINED AS
        "points to dccGroup instance.";;
ATTRIBUTES
        assocOwnerDccGroup      GET-REPLACE;
REGISTERED AS {q823Package 7};

asynchronousTimerPackage PACKAGE
BEHAVIOUR
        asynchronousTimerPackageBehaviour           BEHAVIOUR
DEFINED AS
        "identifies the asynchronous timer specifying a time value. The timer is set when the call attempt is allowed
        and no further call attempts are allowed until the timer expires.";;
ATTRIBUTES
        asynchronousTimer       GET-REPLACE;
REGISTERED AS {q823Package 8};

callsAffectedByDccPackage PACKAGE
BEHAVIOUR
        callsAffectedByDccPackageBehaviour          BEHAVIOUR
DEFINED AS
        "Identifies the number of calls to the observed destination which have been affected by the destination
        control function, by type of control.";;
ATTRIBUTES
        callsAffectedByDcc      INITIAL VALUE Q823-TM-ASN1Module.initialCount;
REGISTERED AS {q823Package 9};

callsAffectedByDccHistoryPackage PACKAGE
BEHAVIOUR
        callsAffectedByDccHistoryBehaviour          BEHAVIOUR
DEFINED AS
        "Identifies the number of calls to the observed destination affected by the destination controls.";;
ATTRIBUTES
        callsAffectedByDcc GET;
REGISTERED AS {q823Package 10};

callsOfferedPackage PACKAGE
BEHAVIOUR
        callsOfferedBehaviour           BEHAVIOUR
DEFINED AS
        "Identifies the number of calls which were offered to the traffic control instance.";;
ATTRIBUTES
        callsOfferedToTrafficControl    INITIAL VALUE Q823-TM-ASN1Module.initialCount;
REGISTERED AS {q823Package 11};

callsOfferedHistoryPackage PACKAGE
BEHAVIOUR
        callsOfferedHistoryBehaviour        BEHAVIOUR
DEFINED AS
        "Identifies the number of calls which were offered to the traffic control instance";;
ATTRIBUTES
        callsOfferedToTrafficControl    GET;
```

```
REGISTERED AS {q823Package 12};

continuousTimerPackage PACKAGE
BEHAVIOUR
        continuousTimerPackageBehaviour     BEHAVIOUR
DEFINED AS
        "identifies the continuous timer which includes the number of calls and a time. Once the number of call
        attempts has been handled with a timer cycle no further attempts are allowed until the time expires (e.g. 5
        calls in 60 seconds.)";;
ATTRIBUTES
        continuousTimer GET-REPLACE;
REGISTERED AS {q823Package 13};

creatorPackage PACKAGE
BEHAVIOUR
        creatorBehaviour      BEHAVIOUR
DEFINED AS
        "Identifies the creator of the object instance. The value of this attribute will be set be the agent at creation
        time. The means for determining the creatorIdentity is a local matter.";;
ATTRIBUTES
        creatorIdentity     GET;
REGISTERED AS {q823Package 14};

destinationCodePackage PACKAGE
BEHAVIOUR
        destinationCodePackageBehaviour     BEHAVIOUR
DEFINED AS
        "identifies the destination.";;
ATTRIBUTES
        destinationCode     GET;
REGISTERED AS {q823Package 15};

destinationTypePackage PACKAGE
BEHAVIOUR
        destinationTypePackageBehaviour     BEHAVIOUR
DEFINED AS
        "identifies the type of destination. It is either the nature of address in a seven bit string according to
        CCITT Recommendation Q.763, or the type of destination as an enumerated list.";;
ATTRIBUTES
        destinationType     GET;
REGISTERED AS {q823Package 16};

leakyBucketPackage PACKAGE
BEHAVIOUR
        leakyBucketTimerBehaviour     BEHAVIOUR
DEFINED AS
        "identifies the leaky bucket which includes the bucket size (maximum allowed counter value) and the
        decrement per time unit. If the counter exceeds the defined maximum size, the call attempt is cancelled. If
        the count value is less than or equal to the maximum size, the call attempt is allowed and the counter is
        incremented. The counter is decremented at defined intervals making it possible for new calls to be
        accepted.";;
ATTRIBUTES
        leakyBucket GET-REPLACE;
REGISTERED AS {q823Package 17};

level2ResponseCategoriesPackage PACKAGE
BEHAVIOUR
        level2ResponseCategoriesPackageBehaviour   BEHAVIOUR
DEFINED AS
        "Identifies in a sequence the strength for a combination of routing aspect and/or origination aspect and/or
        destination aspect and/or additional criteria for threshold level2.";;
```

```
ATTRIBUTES
        level2ResponseCategories GET-REPLACE;
REGISTERED AS {q823Package 18};

percentagePackage PACKAGE
BEHAVIOUR
        percentagePackageBehaviour      BEHAVIOUR
DEFINED AS
        "identifies the strength of the control in percentages. It is expected that only one of the choices of
        percentage for the strength will be supported.";;
ATTRIBUTES
        percentage        GET-REPLACE;
REGISTERED AS {q823Package 19};

stateIndicatorActionPackage PACKAGE
BEHAVIOUR
        stateIndicatorActionPackageBehaviour            BEHAVIOUR
DEFINED AS "to trigger the emission of the ACTION REPLY containing bit map showing the condition of the
        NE during the granularityPeriod.";;
ACTIONS
        stateIndicatorAction;
REGISTERED AS {q823Package 20};

stateIndicatorNotificationPackage PACKAGE
BEHAVIOUR
stateIndicatorNotificationPackageBehaviour              BEHAVIOUR
DEFINED AS" This notification is automatically emitted at the end of granularityPeriod, and provides a bit map
        showing the condition of the NE during the granularityPeriod. ";;
NOTIFICATIONS
        stateIndicatorNotification;
REGISTERED AS {q823Package 21};

tmCircuitEndPointSubgroupListPackage PACKAGE
BEHAVIOUR
        tmCircuitEndPointSubgroupListPackageBehaviour       BEHAVIOUR
DEFINED AS
        "Identifies list of circuit subgroups";;
ATTRIBUTES
        tmCircuitEndPointSubgroupList
        GET-REPLACE
        ADD-REMOVE;

REGISTERED AS {q823Package 22};

triggerThresholdPackage PACKAGE
BEHAVIOUR
        triggerThresholdPackageBehaviour            BEHAVIOUR
DEFINED AS
        "identifies the threshold the trigger threshold value, if crossed will instantiate the adc control.";;
ATTRIBUTES
        triggerThreshold                GET;
REGISTERED AS {q823Package 23};
```

11.4 Definition of common behaviour

```
instancePointerAndNameBehaviour                 BEHAVIOUR
DEFINED AS "if the string choice for the syntax is used, matching of the substrings is permitted. If the number
choice for the syntax is used, then matching on ordering is permitted.";
```

11.5 Definition of attributes

```
accAffectedTrafficId                    ATTRIBUTE
                    DERIVED FROM trafficManagementObjectRdn;
REGISTERED AS {q823Attribute 1};

accResponseCategories                   ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.ResponseCategories;
                    BEHAVIOUR
                    accResponseValueBehaviour           BEHAVIOUR
                    DEFINED AS
                    "Identifies in a sequence the strength for combination of routing aspect, and/or origination
                    aspect and/or destination aspect and/or additional criteria.";;
REGISTERED AS {q823Attribute 2};

activationThresholds                    ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.ActivationThresholds;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                            activationThresholdBehaviour BEHAVIOUR
                    DEFINED AS "identifies the thresholds for activation of the control";;
REGISTERED AS {q823Attribute 3};

accTriggerId ATTRIBUTE
                    DERIVED FROM trafficManagementObjectRdn;
REGISTERED AS {q823Attribute 4};

adcTriggerId                    ATTRIBUTE
                    DERIVED FROM trafficManagementObjectRdn;
REGISTERED AS {q823Attribute 5};

adcTriggerType      ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.AdcTriggerType;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                    adcTriggerTypeBehaviour         BEHAVIOUR
                    DEFINED AS
                    "Identifies whether the adc control object instance to be created is 'centralized' or
                      'decentralized' or 'both'.";;
REGISTERED AS {q823Attribute 6};

adcType             ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.AdcType;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                    adcTypeBehaviour                BEHAVIOUR
                    DEFINED AS
                    "Identifies whether the adc control object instance to be created is 'centralized' or
                      'decentralized'.";;
REGISTERED AS {q823Attribute 7};

answer  ATTRIBUTE
        DERIVED FROM "CCITT Rec. X.721:1992":counter;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        answerBehaviour         BEHAVIOUR
        DEFINED AS
        "information sent in the backward direction indication that the call is answered. This term is defined in
        Annex A/E.410" ;;
REGISTERED AS {q823Attribute 8};
```

```
answeredIncomingSeizures        ATTRIBUTE
         DERIVED FROM answer;
     BEHAVIOUR
     answeredIncomingSeizuresBehaviour BEHAVIOUR
         DEFINED AS
         "identifies the number of incoming seizures where an answer signal was transmitted back to the
         preceding exchange.";;
REGISTERED AS {q823Attribute 9};

answeredOutgoingSeizures        ATTRIBUTE
         DERIVED FROM answer;
     BEHAVIOUR
     answeredOutgoingSeizuresBehaviour BEHAVIOUR
         DEFINED AS
         "identifies the number of outgoing seizures where an answer signal was received.";;
REGISTERED AS {q823Attribute 10};

assocAccAffectedTraffic         ATTRIBUTE
                 WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.AssocAccAffectedTraffic;
                 MATCHES FOR EQUALITY;
                 BEHAVIOUR
                 assocAccAffectedTrafficBehaviour   BEHAVIOUR
                 DEFINED AS
                 "points to the accAffectedTraffic instance for congestion level 1 and 2";;
REGISTERED AS {q823Attribute 66};

assocOwnerDccGroup              ATTRIBUTE
                 WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.AssocOwnerDccGroup;
                 MATCHES FOR EQUALITY;
                 BEHAVIOUR
                 assocOwnerDccGroupPointerBehaviour            BEHAVIOUR
                 DEFINED AS
                 "points to dccGroup Object instance.";;
REGISTERED AS {q823Attribute 11};

assocScrAffectedTraffic         ATTRIBUTE
                 WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.AssocScrAffectedTraffic;
                 MATCHES FOR EQUALITY;
                 BEHAVIOUR
                 assocScrAffectedTrafficBehaviour           BEHAVIOUR
                 DEFINED AS
                 "points to the scrAffectedTraffic instance for reservation level 1 and 2";;
REGISTERED AS {q823Attribute 53};

asynchronousTimer         ATTRIBUTE
                 WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.Timer;
                 MATCHES FOR EQUALITY;
                 BEHAVIOUR
                 asynchronousTimerBehaviour                 BEHAVIOUR
                 DEFINED AS
                 "identifies the asynchronous timer specifying a time value. The timer is set when the call
                 attempt is allowed and no further call attempts are allowed until the timer expires. It is expected
                 that only one of the timer choices will be supported.";;
REGISTERED AS {q823Attribute 12};

autoActivated    ATTRIBUTE
                 WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.TrueFalse;
                 MATCHES FOR EQUALITY;
                 BEHAVIOUR
                 autoActivatedBehaviour        BEHAVIOUR
                 DEFINED AS
```

```
                    "Identifies whether a trigger for an automatic control is outstanding. When the value is ’TRUE’
                    and the administrativeState is ’unlocked’, the automatic control is active. The automatic control
                    is deactivated for all other combinations of the values of the autoActivated and
                    administrativeState. The value of this attribute is initialized by and maintained by the NE.";;
REGISTERED AS {q823Attribute 13};

bids                ATTRIBUTE
        DERIVED FROM "CCITT Rec. X.721:1992":counter;
        MATCHES FOR EQUALITY;
        BEHAVIOUR
        bidsBehaviour       BEHAVIOUR
        DEFINED AS
        "An attempt to obtain a circuit in a circuit subgroup or to a destination. A bid may be successful or
        unsuccessful in seizing a circuit in that circuit subgroup or to that destination. This term is defined in
        Annex A/E.410.";;
REGISTERED AS {q823Attribute 14};

callsAffectedByDcc ATTRIBUTE
          DERIVED FROM "CCITT Rec. X.721:1992":counter;
          MATCHES FOR EQUALITY;
          BEHAVIOUR
          callsAffectedByDccBehaviour       BEHAVIOUR
          DEFINED AS
          "identifies the number of calls to the observed destination affected by destination code control.";;
REGISTERED AS {q823Attribute 15};

callsAffectedByTrafficControl     ATTRIBUTE
          DERIVED FROM "CCITT Rec. X.721:1992":counter;
          BEHAVIOUR
          callsAffectedByTrafficControlBehaviour      BEHAVIOUR
          DEFINED AS
          "identifies the number of calls which are actually affected by traffic control instance.";;
REGISTERED AS {q823Attribute 16};

callsBlockedByLoadShedding      ATTRIBUTE
          DERIVED FROM "CCITT Rec. X.721:1992":counter;
          BEHAVIOUR
          callsBlockedByLoadSheddingBehaviour     BEHAVIOUR
          DEFINED AS
          "identifies the number of calls which cannot be handled due to application of an exchange internal
          overload protection mechanism.";;
REGISTERED AS {q823Attribute 17};

callsOfferedToTrafficControl          ATTRIBUTE
DERIVED FROM "CCITT Rec. X.721:1992":counter;
          BEHAVIOUR
          callsOfferedToTrafficControlBehaviour       BEHAVIOUR
          DEFINED AS
          "identifies the number of calls which were offered to traffic control instance.";;
REGISTERED AS {q823Attribute 18};

c11ResponseCategories                 ATTRIBUTE
                         WITH        ATTRIBUTE                  SYNTAX            Q823-TM-
ASN1Module.ResponseCategories;
                         MATCHES FOR EQUALITY;
                         BEHAVIOUR
                         cl1ResponseCategoriesBehaviour                  BEHAVIOUR
                         DEFINED AS
                         "response categories for congestion level 1.";;
REGISTERED AS {q823Attribute 19};

cl2ResponseCategories           ATTRIBUTE
```

```
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module. ResponseCategories;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                    cl2ResponseCategoriesBehaviour          BEHAVIOUR
                    DEFINED AS
                    "response categories for congestion level 2.";;
REGISTERED AS {q823Attribute 20};

congestionLevel           ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.CongestionLevel;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                    congestionLevelBehaviour      BEHAVIOUR
                    DEFINED AS
                    "Identifies the present congestion situation in an exchange. It is expressed by the following
                    Machine Congestion Levels:
                    – MCL0: no exchange congestion
                    – MCL1: Moderate exchange congestion; the exchange keeps working. Some calls may be
                            rejected.
                    – MCL2: Serious congestion level; the exchange is no longer able to handle all offered traffic.
                            A large number of calls are rejected than MCL1.
                    – MCL3: Unable to process call. While it is desirable, some NE may not be able to provide a
                            level 3 indicator during catastrophic failures.
                    The value of this Attribute will be automatically updated by the NE to reflect the present
                    congestion situation. Modification of the value of this Attribute shall generate an
                    AttributeValueChange Notification.";;
REGISTERED AS {q823Attribute 21};

congestionLevelIndicationId           ATTRIBUTE
                  DERIVED FROM trafficManagementObjectRdn;
REGISTERED AS {q823Attribute 22};

continuousTimer          ATTRIBUTE
                  WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.ContinuousTimer;
                  MATCHES FOR EQUALITY;
                  BEHAVIOUR
                  continuousTimerBehaviour        BEHAVIOUR
                  DEFINED AS
                  "identifies the continuous timer which includes the number of calls and a time. Once the number
                  of call attempts has been handled within a timer cycle, no further attempts are allowed until the
                  timer expires. e.g. 5 calls in 60 seconds. It is expected that only one of the timer choices will be
                  supported.";;
REGISTERED AS {q823Attribute 23};

controlTmCircuitEndPointSubgroup          ATTRIBUTE
             WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.InstancePointerOrName;
             MATCHES FOR EQUALITY, SUBSTRINGS, ORDERING;
             BEHAVIOUR
             instancePointerAndNameBehaviour,
             controlTmCircuitEndPointSubgroupBehaviour BEHAVIOUR
             DEFINED AS
             "identifies the controlled circuit subgroup. The SUBSTRINGS and ORDERING matching rules only
             apply if this Attribute is specified as a name."
             ;;
REGISTERED AS {q823Attribute 58};

creatorIdentity           ATTRIBUTE
                  WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.CreatorIdentity;
                  MATCHES FOR EQUALITY;
                  BEHAVIOUR
                  creatorIdentitiyBehaviour       BEHAVIOUR
```

```
                    DEFINED AS
                    "Identifies the entity which created the object instance";;
REGISTERED AS {q823Attribute 24};

dccGroupId                ATTRIBUTE
                    DERIVED FROM trafficManagementObjectRdn;
REGISTERED AS {q823Attribute 25};

destinationAspect               ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.DestinationAspect;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                    destinationAspectBehaviour     BEHAVIOUR
                    DEFINED AS
                    "Identifies the destination aspects (e.g. HTR) for which this control is valid. If this Attribute
                    has a NULL value, the traffic control is valid for all destination aspects.";;
REGISTERED AS {q823Attribute 26};

destinationCode           ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.DestinationCode;
                    MATCHES FOR EQUALITY, SUBSTRINGS, ORDERING;
                    BEHAVIOUR
                    destinationCodeBehaviour    BEHAVIOUR
                    DEFINED AS
                    "Identifies the country code, or/and area code, or/and exchange code or/and other location
                    number to which object instance applies";;
REGISTERED AS {q823Attribute 27};

destinationType           ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.DestinationType;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                    destinationTypeBehaviour    BEHAVIOUR
                    DEFINED AS
                    "Identifies the type of destination. It is either the nature of address in a seven bit string according
                    to Recommendation Q.763, or the type of destination as an enumerated list.";;
REGISTERED AS {q823Attribute 28};

dispositionOfCalls              ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.DispositionOfCall;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                    dispositionOfCallBehaviour          BEHAVIOUR
                    DEFINED AS
                    "Identifies whether the calls will be skipped to the next available circuit subgroup
                    (if the value is NULL) or will be cancelled (if the value is treatment)";;
REGISTERED AS {q823Attribute 29};

htrDestinationId          ATTRIBUTE
                    DERIVED FROM trafficManagementObjectRdn;
REGISTERED AS {q823Attribute 30};

incomingSeizures          ATTRIBUTE
            DERIVED FROM seizure;
        BEHAVIOUR
        incomingSeizuresBehaviour BEHAVIOUR
            DEFINED AS
            "identifies the number of incoming seizures on the circuit subgroup.";;
REGISTERED AS {q823Attribute 31};
```

```
incomingTraffic    ATTRIBUTE
        DERIVED FROM "CCITT Rec. X.721:1992":counter;
        BEHAVIOUR
        incomingTrafficBehaviour        BEHAVIOUR
        DEFINED AS
        "identifies the number of incoming calls in the exchange for which reception of at least one digit has been
        acknowledged.";;
REGISTERED AS {q823Attribute 32};

incomingTrafficUsage       ATTRIBUTE
     WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.Integer;
        BEHAVIOUR
        incomingTrafficUsageBehaviour    BEHAVIOUR
        DEFINED AS
        "identifies the incoming traffic carried in erlang sec. Typically, this is provided via the sampling
        method.";;
REGISTERED AS {q823Attribute 33};

internalTraffic     ATTRIBUTE
        DERIVED FROM "CCITT Rec. X.721:1992":counter;
        BEHAVIOUR
        internalTrafficBehaviour          BEHAVIOUR
        DEFINED AS
        "identifies the number of internal calls (seizures) in the exchange.";;
REGISTERED AS {q823Attribute 34};

leakyBucket                    ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.LeakyBucket;
                MATCHES FOR EQUALITY;
                BEHAVIOUR
                leakyBucketBehaviour           BEHAVIOUR
                DEFINED AS
                "identifies the leaky bucket which includes the bucket size (maximum allowed counter value) and
                decrement per time unit. If the counter exceeds the defined maximum size, the call attempt is
                cancelled. If the counter is less than or equal the maximum size, the call attempt is allowed and
                the counter is incremented. The counter is decremented at defined intervals making it possible for
                new calls to be accepted. The NE will provide the bucketSize if it is not provided. It is expected
                that only one of the timer choices will be supported.";;
REGISTERED AS {q823Attribute 35};

level1ResponseCategories  ATTRIBUTE
                WITH            ATTRIBUTE        SYNTAX                Q823-TM-
ASN1Module.ResponseCategories;
                MATCHES FOR EQUALITY;
                BEHAVIOUR
                level1ResponseCategoriesBehaviour BEHAVIOUR
                DEFINED AS
                "response categories for reservation level 1";;
REGISTERED AS {q823Attribute 36};

level2ResponseCategories  ATTRIBUTE
                WITH            ATTRIBUTE        SYNTAX                Q823-TM-
ASN1Module.ResponseCategories;
                MATCHES FOR EQUALITY;
                BEHAVIOUR
                level2ResponseCategoriesBehaviour BEHAVIOUR
                DEFINED AS
                "response categories for reservation level 2";;
REGISTERED AS {q823Attribute 37};

newTmCircuitEndPointSubgroups                      ATTRIBUTE
```

```
                WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.NewTmCircuitEndPointSubgroups;
                MATCHES FOR EQUALITY;
                BEHAVIOUR
                newTmCircuitEndPointSubgroupsBehaviour          BEHAVIOUR
                DEFINED AS
                "Identifies in sequence the circuit subgroups which have idle capacity to complete calls.";;
REGISTERED AS {q823Attribute 38};

noCircuitsAvailable          ATTRIBUTE
        DERIVED FROM "CCITT Rec. X.721:1992":counter;
        BEHAVIOUR
        noCircuitsAvailableBehaviour  BEHAVIOUR
        DEFINED AS
        "identifies the number of bids resulting in unsuccessful call due to the fact that no free circuits leading to
        the observed destination was available; i.e. overflow resulting on the final circuit subgroup of that
        destination.";;
REGISTERED AS {q823Attribute 39};

numberOfAvailCircuits    ATTRIBUTE
        DERIVED FROM "CCITT Rec. X.721:1992":gauge;
        BEHAVIOUR
        numberOfAvailCircuitsBehaviour      BEHAVIOUR
        DEFINED AS
        "identifies the number of circuits that can carry traffic including the ones currently carrying traffic.
        Whether this value is provided as snapshot or as a mean value is left to the implementation as due to the
        normally high frequency of changes to the circuits, both methods are equivalent.";;
REGISTERED AS {q823Attribute 40};

observedDestinationId ATTRIBUTE
                DERIVED FROM trafficManagementObjectRdn;
REGISTERED AS {q823Attribute 41};

originationAspect            ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.OriginationAspect;
                MATCHES FOR EQUALITY;
                BEHAVIOUR
                instancePointerAndNameBehaviour,
                originationAspectBehaviour             BEHAVIOUR
                DEFINED AS
                "Identifies the origin and calling parties' category (according to Recommendation Q.763) for
                which the control is valid. If the value of this attribute is empty sequence, the traffic control is
                valid for all origination aspects.";;
REGISTERED AS {q823Attribute 42};

originatingTraffic           ATTRIBUTE
        DERIVED FROM "CCITT Rec. X.721:1992":counter;
        BEHAVIOUR
        originatingTrafficBehaviour     BEHAVIOUR
        DEFINED AS
        "identifies the number of originating calls (seizures) in the exchange for which at least one digit has been
        acknowledged.";;
REGISTERED AS {q823Attribute 43};

outgoingBids ATTRIBUTE
        DERIVED FROM bids;
     BEHAVIOUR
     outgoingBidsBehaviour BEHAVIOUR
        DEFINED AS
        "identifies the number of successful or unsuccessful attempts to seize a circuit in the circuit subgroup";;
REGISTERED AS {q823Attribute 44};
```

```
outgoingSeizures              ATTRIBUTE
            DERIVED FROM seizure;
        BEHAVIOUR
        outgoingSeizuresBehaviour BEHAVIOUR
            DEFINED AS
            "identifies the number of outgoing bids which succeeded in obtaining a circuit within a circuit
            subgroup. ";;
REGISTERED AS {q823Attribute 45};

outgoingTraffic       ATTRIBUTE
            DERIVED FROM "CCITT Rec. X.721:1992":counter;
            BEHAVIOUR
            outgoingTrafficBehaviour        BEHAVIOUR
            DEFINED AS
            "identifies the number of outgoing calls (seizures) from the exchange which has successfully seized a
            circuit.";;
REGISTERED AS {q823Attribute 46};

outgoingTrafficUsage          ATTRIBUTE
        WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.Integer;
            BEHAVIOUR
            outgoingTrafficUsageBehaviour           BEHAVIOUR
            DEFINED AS
            "identifies the outgoing traffic carried in erlang sec. Typically, this is provided via the sampling
            method."; ;
REGISTERED AS {q823Attribute 47};

overflow            ATTRIBUTE
            DERIVED FROM "CCITT Rec. X.721:1992":counter;
            BEHAVIOUR
            overflowBehaviour         BEHAVIOUR
            DEFINED AS
            "identifies the number of outgoing bids overflowing from this circuit subgroup. It will not include calls
            affected by cancel rerouted overflow, tar from and cancel from controls.";;
REGISTERED AS {q823Attribute 48};

percentage          ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.Percentage;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                    percentageBehaviour           BEHAVIOUR
                    DEFINED AS
                    "identifies the percentage of calls that shall be affected as a result of control activation. It is
                    expected that only one of the choices of percentage for the strength will be supported.";;
REGISTERED AS {q823Attribute 49};

returnAction        ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.ReturnAction;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                    returnActionBehaviour                 BEHAVIOUR
                    DEFINED AS
                    "Identifies the disposition of overflowing traffic from the new CircuitEndPointSubgroups.";;
REGISTERED AS {q823Attribute 50};

routingAspect       ATTRIBUTE
                    WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.RoutingAspect;
                    MATCHES FOR EQUALITY;
                    BEHAVIOUR
                    routingAspectBehaviour      BEHAVIOUR
                    DEFINED AS
                    "Identifies the routing aspect (direct routed traffic or alternate routed traffic) for which this
                    control is valid. If the Attribute value is NULL, the traffic control is applied all routing
                    aspects.";;
REGISTERED AS {q823Attribute 51};

scrAffectedTrafficId        ATTRIBUTE
                    DERIVED FROM trafficManagementObjectRdn;
REGISTERED AS {q823Attribute 52};

seizure             ATTRIBUTE
          DERIVED FROM "CCITT Rec. X.721:1992":counter;
          BEHAVIOUR
          seizureBehaviour        BEHAVIOUR
          DEFINED AS
          "a bid for a circuit in a circuit subgroup which succeeds in obtaining a circuit in that circuit subgroup.
          This term is defined in Annex A/E.410.";;
REGISTERED AS {q823Attribute 54};

stateIndicatorId    ATTRIBUTE
                    DERIVED FROM trafficManagementObjectRdn;
REGISTERED AS {q823Attribute 55};

terminatingTraffic       ATTRIBUTE
          DERIVED FROM "CCITT Rec. X.721:1992":counter;
          BEHAVIOUR
          terminatingTrafficBehaviour    BEHAVIOUR
          DEFINED AS
          "identifies the number of terminating calls (seizures to lines) in the exchange.";;
REGISTERED AS {q823Attribute 56};

timeExpiration              ATTRIBUTE
               WITH ATTRIBUTE SYNTAX
               Q823-TM-ASN1Module.Timer;
               MATCHES FOR EQUALITY;
               BEHAVIOUR
               timeExpiratingBehaviour           BEHAVIOUR
               DEFINED AS
               "Identifies the time after which the control will automatically be deleted";;
REGISTERED AS {q823Attribute 57};

tmCircuitEndPointSubgroupList          ATTRIBUTE
               WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.TmCircuitEndPointSubgroupList;
               BEHAVIOUR
               tmCircuitEndPointSubgroupListBehaviour     BEHAVIOUR
               DEFINED AS
               "Identifies the circuit subgroups for which this object instance applies. If the value is empty set, the
               object instance applies to all circuit subgroups.";;
REGISTERED AS {q823Attribute 59};

tmSurveillance      ATTRIBUTE
          WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.TrueFalse;
          MATCHES FOR EQUALITY;
          BEHAVIOUR
          tmSurveillanceBehaviour BEHAVIOUR
          DEFINED AS
          "identifies whether the object instance is being surveilled for traffic management purposes. Setting the
          value of this Attribute to TRUE will result in the creation of an instance of a subclass of currentData,
          named from this object instance and having a value of integer '1' for its scannerId attribute (naming
          attribute). If the creation of currentData object is not successful, the set operation will fail and the value
          of this attribute will remain unchanged. If the value of this attribute was already TRUE before the set
          operation, the operation will succeed but will have no effect on related object.
```

```
        Setting the value of this attribute to FALSE will result in the deletion of an instance of a subclass of
        currentData, named from this object instance and having a value of integer '1' for its scannerId attribute
        (naming attribute). If the deletion of currentData object is not successful, the set operation will fail and
        the value of this attribute will remain unchanged. If the value of this attribute was already FALSE before
        the set operation, the operation will succeed but will have no effect on related object.";;
REGISTERED AS {q823Attribute 60};

trafficControlId            ATTRIBUTE
                DERIVED FROM trafficManagementObjectRdn;
REGISTERED AS {q823Attribute 61};

trafficManagementObjectRdn              ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.NameType;
                MATCHES FOR EQUALITY;
                BEHAVIOUR
                trafficManagmentObjectRdnBehaviour   BEHAVIOUR
                DEFINED AS
                "It is the object identifier";;
REGISTERED AS {q823Attribute 62};

transitTraffic      ATTRIBUTE
        DERIVED FROM "CCITT Rec. X.721:1992":counter;
        BEHAVIOUR
        transitTrafficBehaviour  BEHAVIOUR
        DEFINED AS
        "identifies the number of transit calls (seizures) in the exchange.";;
REGISTERED AS {q823Attribute 63};

treatment                   ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.Treatment;
                MATCHES FOR EQUALITY;
                BEHAVIOUR
                instancePointerAndNameBehaviour,
                treatmentBehaviour          BEHAVIOUR
                DEFINED AS
                "Identifies how the traffic flow impacted by the cancellation will be treated
                (e.g. announcement).";;
REGISTERED AS {q823Attribute 64};

triggerThreshold            ATTRIBUTE
                WITH ATTRIBUTE SYNTAX Q823-TM-ASN1Module.IntegerValue;
                MATCHES FOR EQUALITY;
                BEHAVIOUR
                triggerThresholdBehaviour          BEHAVIOUR
                DEFINED AS
                "Identifies the trigger threshold value that will instantiate an automatic control.";;
REGISTERED AS {q823Attribute 65};
```

11.6 Definition of actions

```
stateIndicatorAction                ACTION
        BEHAVIOUR
        stateIndicatorActionBehaviour           BEHAVIOUR
        DEFINED AS "to trigger the emission of the ACTION REPLY containing bit map showing the condition
        of the NE during the granularityPeriod. The bit position and their definitions are as follows:
        bit-0: exchangeCongestionLevel1: This bit is equal to 1 if during the granularityPeriod the NE
        experiences congestion level 1.
        bit-1: exchangeCongestionLevel2: This bit is equal to 1 if during the granularityPeriod the NE
        experiences congestion level 2.
        bit-2: congestionLevel1Received: This bit is equal to 1 if during the granularityPeriod the NE receives
        CL1 Indicator from any adjacent exchanges.
```

bit-3: congestionLevel2Recevied: This bit is set equal to 1 if during the granularityPeriod when the NE receivedCL2 indicator from any adjacent exchanges.
bit-4: scrTriggered: This bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked and autoActivated is TRUE for at least one instance of SCR control.
bit-5: accTriggered: This bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked and autoActivated is TRUE for at least one instance of acc control.
bit-6: protectiveControlActive: This bit is equal to 1 if during the granularityPeriod one or more instances of protective controls (i.e. cancelTo, cancelFrom, skip, acc, scr, cancelRerouted) is present in the NE.
bit-7: expansiveControlActive: This bit is equal to 1 if during the granularityPeriod when one or more instances of expansive controls (i.e. tarTo, tarFrom) is present in the NE.
bit-8: destinationControlActive: This bit is equal to 1 during the granularityPeriod when one or more instances of destination controls (i.e. destinationCodeControl functions, adc) are present in the NE.
bit-9: htrDestinationActive: This bit is equal to 1 during the granularityPeriod when one or more instances of htrDestination are present in the NE.
bit-10: circuitEndPointSubgroupAddedOrDeleted: This bit is equal to 1 during the granularityPeriod when a circuit subgroup is added or deleted in the NE.
bit-11: accTransmissionInhibited: This bit is equal to 1 if during the granularityPeriod the administrativeState of accTrigger is set to locked.
bit-12: adcTriggered: This bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked and autoActivated is TRUE for at least one instance of adc control";;
MODE CONFIRMED;
WITH REPLY SYNTAX Q823-TM-ASN1Module.StateIndicatorBitMap;
REGISTERED AS {q823Action 1};

11.7 Definition of notifications

stateIndicatorNotification NOTIFICATION
BEHAVIOUR
stateIndicatorNotificationBehaviour BEHAVIOUR
DEFINED AS "This notification is automatically emitted at the end of granularityPeriod, and provides a bitmap showing the condition of the NE during the granularityPeriod. The bit position and their definitions are as follows:
bit-0: exchangeCongestionLevel1: This bit is equal to 1 if during the granularityPeriod the NE experiences congestion level 1.
bit-1: exchangeCongestionLevel2: This bit is equal to 1 if during the granularityPeriod the NE experiences congestion level 2.
bit-2: congestionLevel1Received: This bit is equal to 1 if during the granularityPeriod the NE receives CL1 Indicator from any adjacent exchanges.
bit-3: congestionLevel2Recevied: This bit is set equal to 1 during the granularityPeriod when the NE receivedCL2 indicator from any adjacent exchanges.
bit-4: scrTriggered: This bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked and autoActivated is TRUE for at least one instance of SCR control.
bit-5: accTriggered: This bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked and autoActivated is TRUE for at least one instance of acc control.
bit-6: protectiveControlActive: This bit is equal to 1 if during the granularityPeriod one or more instances of protective controls (i.e. cancelTo, cancelFrom, skip, acc, scr, cancelRerouted) are present in the NE.
bit-7: expansiveControlActive: This bit is equal to 1 during the granularityPeriod when one or more instances of expansive controls (i.e. tarTo, tarFrom) are present in the NE.
bit-8: destinationControlActive: This bit is equal to 1 during the granularityPeriod when one or more instances of destination controls (i.e. destinationCodeControl functions, adc) are present in the NE.
bit-9: htrDestinationActive: This bit is equal to 1 during the granularityPeriod when one or more instances of htrDestination are present in the NE.
bit-10: circuitEndPointSubgroupAddedOrDeleted: This bit is equal to 1 during the granularityPeriod when a circuit subgroup is added or deleted in the NE.
bit-11: accTransmissionInhibited: This bit is equal to 1 if during the granularityPeriod the administrativeState of accTrigger is set to locked.
bit-12: adcTriggered: This bit is equal to 1 if during the granularityPeriod the administrativeState is unlocked and autoActivated is TRUE for at least one instance of adc control";;

```
        WITH INFORMATION SYNTAX Q823-TM-ASN1Module.StateIndicatorBitMap;

REGISTERED AS {q823Notification 1};
```

11.8 ASN.1 defined types module

```
-- Extensibility rules
-- As per ISO 8824-DAM on extensibility rules, the productions that are of extensible types are to
-- be indicated by including three (3) ellipses (...) in their type descriptions.

Q823 – TM-ASN1Module {ccitt(0) recommendation(0) q(17) q823(823) asn1Module(2) q823ASN1Module(0)}

DEFINITIONS IMPLICIT TAGS ::=
BEGIN

-- EXPORTS everything

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

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

    HistoryRetention
    FROM
    Q822-PM-ASN1Module {ccitt(0) recommendation(0) q(17) q822(822) asn1Module(2) q822ASN1Module(0)}

    NameType,
    FROM
    ASN1DefinedTypesModule {ccitt recommendation m(13) gnm(3100) informationModel(0) asn1Modules(2)
asn1DefinedTypesModule(0)};

q823-InformationModel OBJECT IDENTIFIER ::= {ccitt(0) recommendation(0) q(17) q823(823)
informationModel(0)}

q823ObjectClass OBJECT IDENTIFIER ::= {q823-InformationModel managedObjectClass(3)}

q823Package OBJECT IDENTIFIER ::= {q823-InformationModel package(4)}

q823NameBinding OBJECT IDENTIFIER ::= {q823-InformationModel nameBinding(5)}

q823Attribute OBJECT IDENTIFIER ::= {q823-InformationModel attribute(6)}

q823Action OBJECT IDENTIFIER ::= {q823-InformationModel action(7)}

q823Notification OBJECT IDENTIFIER ::= {q823-InformationModel notification(8)}

--value assignments for origin aspect extension

        extendOrigin OBJECT IDENTIFIER ::= {q823-InformationModel standardSpecificExtension(0)
extendOrigin(0)}

originated Origin ::= originExtension : {extendOrigin 1}
```

```
transit Origin ::= originExtension : {extendOrigin 2}

inboundTerminating Origin ::= originExtension : {extendOrigin 3}

--value assignment for destination aspect extension

extendDestination OBJECT IDENTIFIER ::= {q823-InformationModel standardSpecificExtension(0)
extendDestination(1)}

hardToReach DestinationAspect ::= destinationExtension : {extendDestination 1}

nonHardToReach DestinationAspect ::= destinationExtension : {extendDestination 2}

        -- default value definition

defaultAdministrativeState AdministrativeState ::= unlocked

defaultAllTmCircuitEndPointSubgroups TmCircuitEndPointSubgroupList ::= {}

defaultAllDestinationCodes DestinationCode ::= ""

defaultDestinationAspect DestinationAspect ::= definedDestinationAspect : null

defaultOriginAspect OriginationAspect ::= {}

defaultRoutingAspects RoutingAspect ::= null

--initial value definitions

initialCount Count ::= 0

initialGauge ObservedValue ::= integer: 0

initialInteger INTEGER ::= 0

-- permitted value definitions

PermittedState ::= AdministrativeState(locked | unlocked)

PermittedHistoryRetention ::= HistoryRetention(1)

PermittedTarFromReturnAction ::= ReturnAction (WITH COMPONENTS {return, cancelTreatment})

--supporting productions

ActivationThresholds ::=  SEQUENCE {
        level1  [0]     ThresholdLevel,
        level2  [1]     ThresholdLevel OPTIONAL}

AdcTriggerType ::= ENUMERATED {
        centralized             (0),
        decentralized           (1),
        both                    (2) -- both centralized and decentralized --
        }

AdcType ::= ENUMERATED {
        centralized             (0),
        decentralized           (1)}
AssocAccAffectedTraffic ::= ObjectInstance -- points to a accAffectedTraffic
```

```
AssocOwnerDccGroup ::= ObjectInstance -- points to dccGroup object instance

AssocScrAffectedTraffic ::= ObjectInstance -- points to scrAffectedTraffic

Category ::= BIT STRING (SIZE(8)) -- value according to Recommendation Q.763

CongestionLevel ::= ENUMERATED {
        mcl0  (0),
        mcl1  (1),
        mcl2  (2),
        mcl3  (3)}

ContinuousTimer ::= SEQUENCE {
        calls           [0]     INTEGER,
        perTimeUnit     [1]     Timer
        }

CreatorIdentity ::= CHOICE {
        sourceCls           SourceClass,
        sourceName          GraphicString
        }

DefinedOrigin ::= ENUMERATED {
        originated              (0),
        transit                 (1),
        inboundTerminating      (2)
        }

DestinationAspect ::= CHOICE {
        definedDestinationAspect        DefinedDestinationAspect,
        destinationExtension                    OBJECT IDENTIFIER
        }

DefinedDestinationAspect ::= ENUMERATED {
        null    (0), --all destination aspects
        htr     (1),
        nhtr    (2)
}

DestinationCode ::= GraphicString(FROM("0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | "A" | "B" |
"C" | "D" | "E" | "F" | "#"| "*"))

DestinationType ::= CHOICE {
        natureOfAddress         [0]     NatureOfAddress,
        desttype                [1]     DestType,
        -- "..."these ellipses defined in ASN.1 amendment are used here to ind icate that this is --
        -- an extensible type and additional choices may be added in the future--
}

DestType ::= ENUMERATED {
        international       (0),
        national            (1),
        local               (2),
        other               (3)
        }

DispositionOfCall ::= CHOICE {
        skip                NULL,
        treatment           Treatment
        }
```

```
InstancePointerOrName ::= CHOICE {
        objectInstance      [0]    ObjectInstance,
        symbolic            [1]    NameType
        }

Integer ::= INTEGER(0 .. MAX)

IntegerValue ::= INTEGER(0 .. 100)

LeakyBucket ::= SEQUENCE {
        bucketSize          [0]    INTEGER OPTIONAL, -- the NE will define the bucketSize if not provided
        calls               [1]    INTEGER,
        perTimeUnit         [2]    Time (WITH COMPONENTS{scale1, scale2})
        }

NatureOfAddress ::= BIT STRING (SIZE(7)) -- value according to Recommendation Q.763

NewTmCircuitEndPointSubgroups ::= SEQUENCE OF InstancePointerOrName

Origin ::= CHOICE {
        definedOrigin       [0]    DefinedOrigin,
        namedOrigin         [1]    InstancePointerOrName, --it points to or names an origin.
        originExtension     [2]    OBJECT IDENTIFIER
        }

OriginationAspect ::= SEQUENCE {
                callingPartyCategory [0] Category OPTIONAL,
                origin [1] Origin OPTIONAL
                }

Percentage ::= CHOICE {
        granularity12p5     [0]    ENUMERATED {
                                   perc0 (0),          --0%-
                                   perc12p5 (2),       --12.5%--
                                   perc25 (4),    -    --25%--
                                   perc37p5 (6),       --37.5%--
                                   perc50 (8),         --50%--
                                   perc62p5 (10),      --67.5%--
                                   perc75 (12),        --75%--
                                   perc87p5 (13),      --87.5%--
                                   perc100 (15)  -     --100%--
                                   },
        granularity10       [1]    ENUMERATED {
                                   perc0 (0) -- 0%--,
                                   perc10 (1) -- 10%--,
                                   perc20 (2) -- 20%--,
                                   perc30 (3) -- 30% --,
                                   perc40 (4) -- 40%--,
                                   perc50 (5) -- 50%--,
                                   perc60 (6) -- 60%--,
                                   perc70 (7) -- 70%--,
                                   perc80 (8) -- 80%--,
                                   perc90 (9) -- 90%--,
                                   perc100 (10) -- 100%--
                                   }
}

ResponseCategories ::= SET OF SEQUENCE {
        originationAspect      [0]      OriginationAspect,
        destinationAspect      [1]      DestinationAspect,
        routingAspect          [2]      RoutingAspect,
        additionalCriteria      [3]     OBJECT IDENTIFIER OPTIONAL,
        percentage              [4]     Percentage
        }

ReturnAction ::= CHOICE {
        return              [0]     NULL,
        skip                [1]     NULL,
        cancelTreatment     [2]     Treatment
        }
RoutingAspect ::= ENUMERATED {
        directRoutedTraffic         (0),
        alternateRoutedTraffic      (1),
        null                        (2) -- direct and alternate routed traffic
}
SourceClass ::= ENUMERATED {
        tnmos       (0),
        otherNE     (1),
        thisNE      (2),
        other       (3)
        }
StateIndicatorBitMap ::= BIT STRING {
        exchangeCongestionLevel1        (0),
        exchangeCongestionLevel2        (1),
        congestionLevel1Received        (2),
        congestionLevel2Received        (3),
        scrTriggered                    (4),
        accTriggered                    (5),
        protectiveControlActive         (6),
        expansiveControlActive          (7),
        destinationControlActive        (8),
        htrDestinationActive            (9),
        circuitEndPointSubgroupAddedOrDeleted(10),
        accTransmissionInhibited        (11),
        adcTriggered                    (12)
        }
ThresholdLevel ::= CHOICE {
        percentageOfCircuits        [0]     INTEGER(0 .. 100),
        noOfCircuits                [1]     INTEGER
        }
TmCircuitEndPointSubgroupList ::= SET OF InstancePointerOrName
Treatment ::= CHOICE {
        other                       [0]     OBJECT IDENTIFIER,
        announcementNumber          [1]     INTEGER,
        congestionTone              [2]     NULL
        }
Timer ::= CHOICE {
    blockAllCalls       [0]     NULL,
    scale1              [1]     ENUMERATED {
                                sec0 (0)            -- 0 sec --,
                                sec0p1 (1)          -- 0.1 sec --,
                                sec0p25 (2)         -- 0.25 sec --,
                                sec0p5 (3)          -- 0.5 sec --,
                                sec1 (4)            -- 1 sec --,
                                sec2 (5)            -- 2 sec --,
                                sec5 (6)            -- 5 sec --,
                                sec10 (7)           -- 10 sec --,
                                sec15 (8)           -- 15 sec --,
                                sec30 (9)           -- 30 sec --,
                                sec60 (10)          -- 60 sec --,
                              sec120 (11)          -- 120 sec --,
                              sec300 (12)           -- 300 sec --,
                              sec600 (13)          -- 600 sec--
                              },
        scale2          [2]   INTEGER(0 .. 600000) --millisec. scale
}
TrueFalse ::= BOOLEAN
END
```

APPENDIX I

Traffic management TMN management service

This Appendix identifies the traffic network management functions to be incorporated in Recommendation M.3200 that are covered by this Recommendation and those that are outside the scope of this Recommendation. See Table I.1

Table I.1/Q.823 – Q3 information flows

	NE	OS
Network management functions		**Covered By (Rec.)**
1 Status monitoring functions		
Report the service availability of network elements	<-------------	M.3100
Report the status of controls on demand	<-------------	Q.823
Report circuit group busy/idle status	<-------------	Indirectly
Report congestion status of exchange	<-------------	Q.823
Report the receipt of ACC signal	<-------------	Q.823
Report HTR status of destinations on demand	<------------	Q.823
Manually add remove HTR status of destination	------------>	Q.823
Report congestion status of common channel signalling network	<------------	Out of scope
Report the receipt of common channel signalling network management signals	<------------	Out of scope
2 Performance monitoring functions		
Report circuit group data/parameters on scheduled basis	<-------------	Q.822, Q.823
Report circuit group data/parameters on demand	<-------------	Q.822, Q.823
Report exchange load measurements on a scheduled basis	<-------------	Q.822, Q.823
Report exchange load measurements on demand	<-------------	Q.822, Q.823
Report exchange congestion measurements on a scheduled basis	<-------------	Q.822, Q.823
Report exchange congestion measurements on demand	<-------------	Q.822, Q.823
Report CCS load measurement on a scheduled basis	<-------------	Out of scope
Report CCS load measurement on demand	<-------------	Out of scope
Report CCS congestion measurement on a scheduled basis	<-------------	Out of scope
Report CCS congestion measurement on demand	<-------------	Out of scope
Report data on performance of controls on a scheduled basis	<-------------	Q.822, Q.823
Report data on performance of controls on demand	<-------------	Q.822, Q.823

Table I.1/Q.823 – Q3 information flows *(concluded)*

	NE	OS
Network management functions		**Covered By (Rec.)**
3 Control functions		
Apply/modify/remove manual control	------------->	Q.823
Establish/modify/remove an automatic control	------------->	Q.823
Activate/deactivate an automatic control	------------->	Q.823
Apply/modify/remove a special recorded announcement	------------->	Out of scope
4 Administrative functions		
Establish/change/remove measurement schedule	------------->	X.746, Q.823
Establish/change/update a network management database	------------->	Out of scope
Establish/change/remove threshold for status reporting, data reporting and HTR Determination	------------->	Q.822, Q.823
Establish/change/remove schedules for status and data reporting	------------->	Q.822, Q.823
Reporting routing table information on demand		
a) which circuit subgroups are part of which destinations	<-------------	Out of scope
b) which destinations are routed over a given circuit subgroup	<------------	Out of scope

APPENDIX II

Correlation between E.412 traffic controls and object classes covered in this Recommendation

Table II.1 gives the correlation between the traffic controls as defined in Recommendation E.412 and the object class defined in this Recommendation.

Table II.1/Q.823

E.412 traffic control	Covered by object class(es) in Rec. Q.823	Remarks
Code blocking	destinationCodeControl, dccGroup	None
Call gapping	destinationCodeControl, dccGroup	None
Cancel direct routing	cancelTo	None
Cancel alternate routing To	cancelTo	None
Cancel alternate routing From	cancelFrom	None
Circuit directionalization	–	Out of scope
Circuit turndown/busying/blocking	–	Out of scope
Skip	skip	None
Temporary alternate routing on circuit subgroup/destination	tarTo, tarFrom	None
Cancel rerouted overflow	cancelRerouted	None
Automatic congestion control	acc, accTrigger, accAffectedTraffic	None
Selective circuit reservation	scr, scrAffectedTraffic	None
Automatic destination control	adcTrigger, adc	None

APPENDIX III

State transition diagrams for automatic and manual controls

See Figures III.1 and III.2.

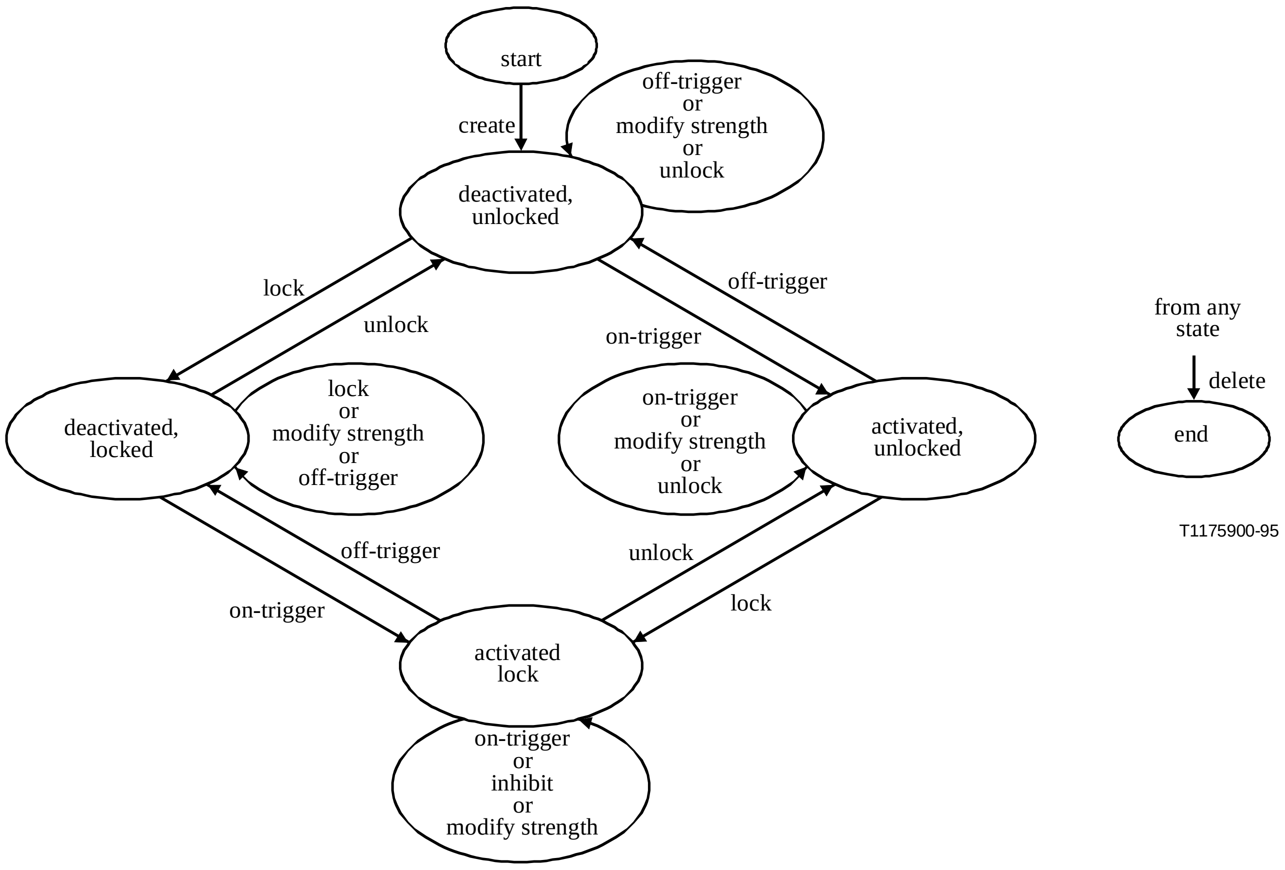

Figure III.1/Q.823 – Automatic control state transition diagram

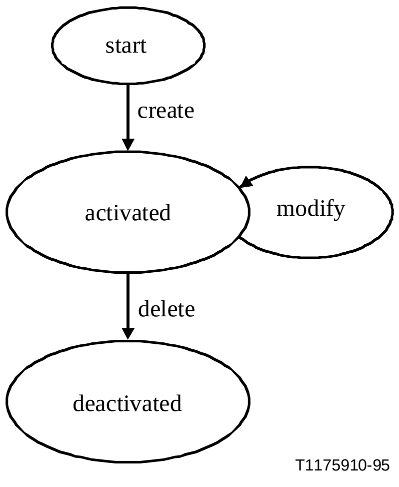

Figure III.2/Q.823 – Manual control state transition diagram

APPENDIX IV

Differentiation of direct and alternate routed traffic for cancel-to and cancel-from controls

In order to effectively use the network resources during stress period, the network manager makes use of Cancel-To and Cancel-From controls on circuit subgroups. When applying these controls, the network manager should have the ability to differentiate and selectively control either direct routed traffic or the alternate routed traffic on both types of controls. Figure IV.1 explains the various types of traffic, and how a network manager can selectively apply the different controls for each traffic type.

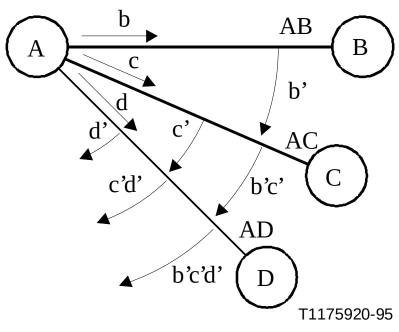

Figure IV.1/Q.823

In the Figure, the following traffic types are depicted:

- b-traffic is direct offered traffic to circuit subgroup AB;
- c-traffic is direct offered traffic to circuit subgroup AC;
- d-traffic is direct offered traffic to circuit subgroup AD;
- b'-traffic is an overflow of b-traffic from circuit subgroup AB which is now alternate routed to circuit subgroup AC;
- c'-traffic is an overflow of c-traffic from circuit subgroup AC which is now alternate routed to circuit subgroup AD;
- d'-traffic is an overflow of d-traffic from circuit subgroup AD which is now alternate routed to the subsequent circuit subgroup;
- b'c'-traffic is an overflow of b'-traffic from circuit subgroup AC which is alternate routed again to circuit subgroup AD;
- c'd'-traffic is an overflow of c'-traffic from circuit subgroup AD which is alternate routed again to the subsequent circuit subgroup;
- b'c'd'-traffic is an overflow of b'c'-traffic from circuit subgroup AD which is alternate routed again to the subsequent circuit subgroup.

Note that b'-traffic and c'-traffic is traffic which has overflown only once (that is direct routed traffic which has overflown); where as b'c'-traffic and c'd'-traffic is traffic which has overflown more than once (that is, alternate routed traffic which has overflown).

The network manager should have the flexibility of controlling each type of the traffic listed above using the Cancel-To or Cancel-From control; especially traffic which has overflown only once and traffic which has overflown more than once. Table IV.1 illustrates how this can be accomplished:

Table IV.1/Q.823

Traffic to be controlled	Type of control
b-traffic	Cancel-To direct on circuit subgroup AB
c-traffic	Cancel-To direct on circuit subgroup AC
d-traffic	Cancel-To direct on circuit subgroup AD
b’-traffic	Cancel-From direct on circuit subgroup AB
c’-traffic	Cancel-From direct on circuit subgroup AC
d’-traffic	Cancel-From direct on circuit subgroup AD
b’c’-traffic	Cancel-From alternate on circuit subgroup AC
c’d’-traffic	Cancel-From alternate on circuit subgroup AD
b’c’d’-traffic	Cancel-From alternate on circuit subgroup AD

Note that if circuit subgroups AB, AC and AD are also part of other routing tables, the corresponding overflows in those routing tables will be affected as well.

APPENDIX V

Use of Q.822 history data for traffic measurements

V.1 Introduction

This Appendix provides a tutorial of the application of the Q.822 performance management information model to the collection of traffic measurements. The tutorial provides mechanisms to accomplish both autonomous reporting and polled retrieval of measurements.

V.2 Overview of the performance management information model

The performance management model enables summary reporting of performance parameters using the simpleScanner.

The object model for performance management is shown in Figure 1/Q.822.

The data found in the currentData object class instances (or its subclasses) consists of measurements which are in the process of accumulating and changing during the interval. At the end of the interval, currentData object instances are initialized. Hence, the collected data must either be immediately reported or stored for persistence of completed intervals. The model supports the retention of the last completed interval as historyData object instances (or its subclasses).

At the end of each performance interval a scanReport may be issued by the currentData object directly. However, this would result in a separate notification for each currentData instance being monitored.

It is more efficient to aggregate the measurements from all the currentData instances accumulated during the same period. The recommended mode is to use the simpleScanner to synchronize the historyData instances. The simpleScanner managed object scans the common attributes to be observed in each of the selected objects according to a specified schedule and produces an aggregated report.

The notifications generated by the scanner may be processed by an Event Forwarding Discriminator (EFD) for reporting to the managing system.

The generation of PM summarized reports is depicted in Figure B.5/Q.822. The illustration shows that the simpleScanner can be used to scan attributes of the historyData objects and emit a scanReport to the EFD. The EFD in turn forwards an m-EVENT Report of the summarized data providing the established value assertions are met (e.g. eventType = scanReport).

V.3 Initial configuration

This subclause discusses the process in which the collection mechanism is configured.

The following object classes are instantiated:

1) subclasses of currentData, e.g. tmTmCircuitEndPointSubgroupCurrentData;
2) subclasses of historyData, e.g. xtpsgHistoryData;
3) simpleScanner.

V.3.1 Current data

The collection interval is specified in the attribute granularityPeriod which is inherited from the scanner. currentData is a subclass of scanner. Typical examples of the value are five minutes or sixty seconds.

The historyRetentionPkg conditional package is required so that the appropriate number of intervals of historyData can be established. For traffic management purposes, this value is restricted to 1 in order to minimize the use of memory resources.

V.3.2 History data

The retention period (duration) is specified in the attribute historyRetention. When a new instance of historyData is created, the oldest existing instance may be deleted.

The historyDataSuspectIntervalPkg conditional package is required since it is useful to know whether measurements are suspect.

V.3.3 Simple scanner

It is recommended that the following attributes inherited from the scanner or the homogeneousScanner be set as follows to achieve the desired BEHAVIOUR.

1) scannerId (inherited from Recommendation X.739:scanner)

   The scannerId contains a value that is used to identify an instance of this managed object class (used for naming).

2) administrativeState (inherited from Recommendation X.739:scanner)

   The administrativeState (defined in Recommendation X.731) is used to suspend or resume the scanning function. If the administrative state has the value UNLOCKED, the scanner is administratively permitted to perform scans. Initially the state should be set to UNLOCKED rather than LOCKED so that scanning may be permitted.

3) granularityPeriod (inherited from Recommendation X.739:scanner)

   The granularityPeriod attribute indicates the length of time between successive scans. This should be set to e.g. 5 minutes, to achieve an autonomous scan and report at the end of every period.

4) scope (inherited from Recommendation X.738:homogeneousScanner)

   The scoped selection package is specified. The scanner uses the managed object class identified in the base managed object attribute and checks all the managed objects within the levels indicated by the scope attribute. The baseManagedObject that should be the starting node for selecting managed object is the managedElement (the network element such as the circuit switch). The value of the scope attribute, for example, will be set to individual level 2. This implies all objects contained in the second level of the naming hierarchy relative to the managedElement are candidates for selection. This selection is further restricted by the scanningFilter as explained below.

5) scanningFilter (inherited from Recommendation X.738:homogeneousScanner)

The scanner checks instances of the managed objects within the levels indicated by applying the criteria in the scanningFilter attribute.

The filter may be configured to select between the following types of data:

a) circuit subgroup;

b) destination;

c) exchange.

For example, the filter may be configured to select history data objects for the monitored entities circuit subgroup, destination and exchange (objectClass = tmCircuitEndPointSubgroupHistoryData OR objectClass = destinationHistoryData OR objectClass = exchangePerformanceHistoryData).

6) numericAttributeIdArray (from Recommendation X.738:simpleScanner)

The numericAttributeIdArray must be set. It is the ordered list of identifiers of attributes whose values are to be scanned. The values will be contained in the numericValueArray.

7) supressObjectInstance (from Recommendation X.738:simpleScanner)

The supressObjectInstance attribute is set to TRUE so that object instance identification is not included in the response. This will reduce the number of bytes in the message.

NOTE – supressObjectInstance to TRUE should be avoided if zero suppression package of currentData is included.

V.4 Autonomous reporting

The autonomous reporting mechanism is illustrated in Figure V.1 – Autonomous reporting of measurements.

The scanner may invoke each granularity period autonomously. Autonomous reports are generated periodically by appropriately setting the simplescanner granularityPeriod e.g. 5 minutes. The appropriate history data for each monitored object will be scanned. The appropriate attribute values will be inserted in the numericValueArray of the scanReport NOTIFICATION.

The numericValueArray may have a sequence of integers or reals. The exact order of attribute identifiers is known from the numericAttributeIdArray.

The notification is sent to the EFD.

The scanReport NOTIFICATION is evaluated by the EFD. The discriminator construct should be provisioned for the appropriate notifications to be forwarded. The summarized reports will be forwarded to the manager as m-EVENT reports.

V.5 Polled reporting

The polled reporting mechanism is illustrated in Figure V.2 – Polled reporting of measurements.

A polling mechanism may be achieved by setting the simpleScanner granularityPeriod to zero. This will disable autonomous scanning. The manager (operation system) may initiate a polled scan by invoking the activateScanReport action (ACTION) to the simple scanner. This will cause the scanner to execute a single time (collect the data and respond with results). The action will be executed using parameters for scope and scanningFilter the values of the attributes as in the autonomous reporting case.

The summarized report will be sent to the manager as an m-ACTION response.

V.6 Changes to the report

The manager may suspend the autonomous generation of the simpleScanner reports by setting the administrativeState to locked. This will abruptly discontinue use of scanning. A more graceful approach is accomplished by setting the administrativeState to shutdown. This will shutdown after completing the current interval and issuing the report.

The manager may change the scope, filter and numericAttributeIdArray to effect the contents of the report. Other parameters may also be changed to change the behaviour of the reporting mechanism, e.g. granularityPeriod.

V.7 Conclusions

The standard mechanisms of the Q.822 performance management specification can be used in conjunction with the X.738 simpleScanner to summarize and report traffic measurements. Polled and autonomous reporting can be supported using the two mechanisms described.

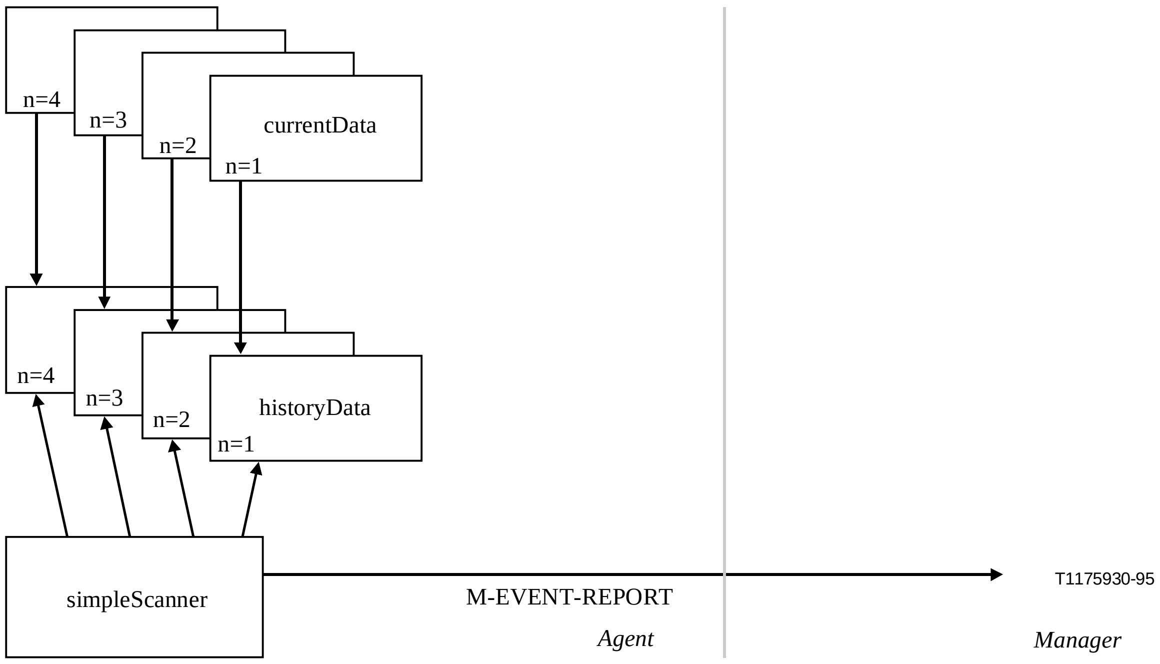

Figure V.1/Q.823 – Autonomous reporting of measurements

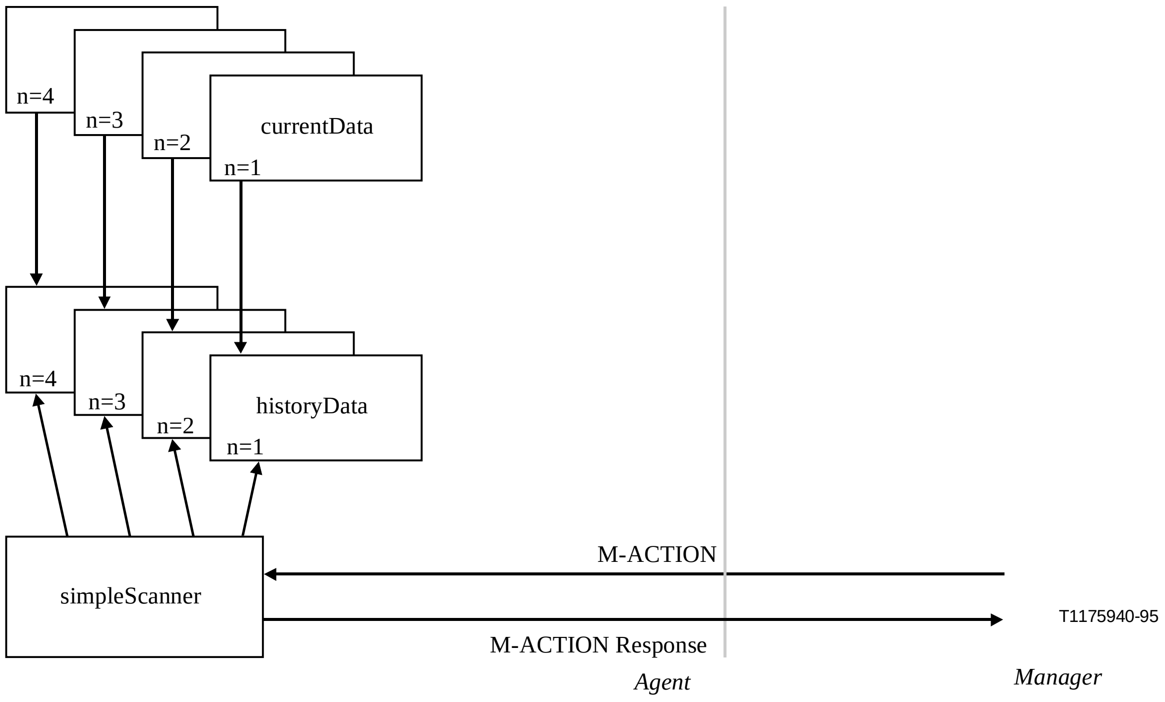

Figure V.2/Q.823 – Polled reporting of measurements

APPENDIX VI

Circuit end point subgroup object definition

Until the following circuitEndPointSubgroup object definition is moved in to Recommendation M.3100, it is included here for the understanding and completeness of Q.823 model.

```
circuitEndPointSubgroup            MANAGED OBJECT CLASS
  DERIVED FROM "Recommendation X.721 :1992":top;
  CHARACTERIZED BY
    circuitEndPointSubgroupPackage PACKAGE
      BEHAVIOUR
        circuitEndPointSubgroupBehaviour BEHAVIOUR
DEFINED AS
        "The set of circuit endpoints within a group that directly interconnects one exchange with another having
        common values for the attributes listed in the package. Note that the term exchange includes PBX where
        applicable. -- Annex A/E.410 defines circuit subgroup.
      ;;
      ATTRIBUTES
      circuitEndPointSubgroupId              GET,
      numberOfCircuits                GET,
      labelOfFarEndExchange                  GET,
      signallingCapabilities          GET,
      informationTransferCapabilities      GET,
      circuitDirectionality           GET,
      transmissionCharacteristics     GET,
      userLabel                       GET;
;;
NOTIFICATIONS
"Recommendation X.721:1992":attributeValueChange,
"Recommendation X.721:1992":objectCreation,
"Recommendation X.721:1992":objectDeletion,
REGISTERED AS { m3100ObjectClass x};

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 Rec. X.721
        ttpIdBehaviour  BEHAVIOUR
        DEFINED AS "The circuit subgroup Id is an attribute type whose distinguished value can be used as a
RDN when naming an instance of the circuit subgroup object class.";;

REGISTERED AS {m3100Attribute x};

numberOfCircuits ATTRIBUTE
        WITH ATTRIBUTE SYNTAX ASN1DefinedTypesModule.NumberOfCircuits;
        MATCHES FOR EQUALITY, ORDERING;
        DEFINED AS "The number of circuits in a circuit subgroup.";;

REGISTERED AS {m3100Attribute x};

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 x};

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 x};

informationTransferCapabilites 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 Kbs unrestricted
data supported by the circuit subgroup.";;

REGISTERED AS {m3100Attribute x};

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 x};

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 x};

ASN.1 productions:

NumberOfCircuits ::= INTEGER

SignallingCapabilities ::= ENUMERATED {isup(0), isup2(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 --}

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

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 --}

TransmissionCharacteristics ::= BITSTRING {satellite(0), dCME(1), echoControl(2)}

--creation and deletion of CircuitEndPointSubgroup using the following name binding is outside the scope of this Recommendation.

```
circuitEndPointSubgroup-managedElement NAME BINDING
	SUBORDINATE OBJECT CLASS circuitEndPointSubgroup;
	NAMED BY
	SUPERIOR OBJECT CLASS "M.3100":managedElement AND SUBCLASSES;
	WITH ATTRIBUTE circuitEndPointSubgroupId;
	CREATE WITH-REFERENCE-OBJECT,
		WITH-AUTOMATIC-INSTANCE-NAMING;
	DELETE;
```

APPENDIX VII

Mapping of NM functions to functional units

Table VII.1 provides a mapping of NM functions (listed in Appendix I) to the functional units.

Table VII.1/Q.823

Network management functions	Functional units (clause 8)
1 Status monitoring functions	
Report the service availability of network elements	H
Report the status of controls on demand	H, I or J
Report circuit group busy/idle status	Indirectly – see circuit group data/parameter on demand
Report congestion status of exchange	H, I or J
Report the receipt of ACC signal	I or J
Report HTR status of destinations on demand	G, H
Manually add remove HTR status of destination	G
Report congestion status of common channel signalling network	Out of scope
Report the receipt of common channel signalling network management signals	Out of scope
2 Performance monitoring functions	
Report circuit group data/parameters on scheduled basis	H, D, ii)
Report circuit group data/parameters on demand	H, D, I
Report exchange load measurements on scheduled basis	C, ii)
Report exchange load measurements on demand	C, I
Report exchange congestion measurements on scheduled basis	C, ii)
Report exchange congestion measurements on demand	C, I
Report CCS load measurement on scheduled basis	Out of scope
Report CCS load measurement on demand	Out of scope
Report CCS congestion measurement on scheduled basis	Out of scope
Report CCS congestion measurement on demand	Out of scope
Report data on performance of controls on a scheduled basis	B, ii)
Report data on performance of controls on a demand	B, i)
3 Control functions	
Apply/modify/remove manual control	A
Establish/modify/remove an automatic control	A
Apply/modify/remove a special recorded announcement	Out of scope

Table VII.1/Q.823 *(concluded)*

Network management functions	Functional units (clause 8)
4 Administrative functions	
Establish/change/remove measurement schedule	K
Establish change/update network management database	Out of scope
Establish change/remove threshold for status reporting, data reporting and HTR determination	ii)
Establish/change/remove schedules for status and data reporting	ii)
Reporting routing table information on demand	Out of scope

APPENDIX VIII

Support of the audit functionality using Q.823

VIII.1 Introduction

This Appendix describes the use of the model in this Recommendation, combined with the model in management knowledge management function (Recommendation X.750) to support the audit capabilities. The functional requirements for audit by the manager are discussed in VIII.2. The mechanisms that may be used to meet these requirements are described in VIII.3.

VIII.2 Functional requirements

Audits are necessary to ensure that the manager and the agent have the same view of the reference data (non-measurement data). The manager requests an audit from the network element of the data in the MIB (different from the surveillance data reported periodically) in order to synchronize the knowledge of the MIB in the manager with the current state of the agent. These audits are initiated by the manager for one or more of the following reasons:

a) In response to anomalies (such as measurement data from a trunk group is missing even though the agent is scheduled to send it) found in the received measurement data.

b) In response to creation, deletion and changes in values of management information.

c) In response to alerting information such as congestion or control activities provided autonomously by the agent (e.g. sateIndicator object class).

d) In order to support periodic synchronization of the shared management knowledge between the manager and the agent.

The mechanisms for retrieval of data from the agent should support customization by the manager to meet one or more of the following criteria: type of data, volume of data frequency of retrieval. This retrieval of data should support audits of all reference data (full audit) as well as selective retrieval of the reference data changed since the last audit (change audit).

These requirements except the change audit can be met using the mechanisms described below. A possible approach to support change audit if required by the manager is described below.

VIII.3 Types of audit

VIII.3.1 Full/partial audit

A full audit retrieves all the information in the NE's MIB for the selected object class. A partial audit retrieves a subset of those objects based on filtering criteria applied to the object attributes. The different mechanisms for requesting audit information is described in section VIII.4.

This audit is requested by the OS:

a) when an OS wants to build an initial view of the NE's MIB;

b) when OS wants to validate its view of the MIB with the NE.

Upon receiving the audit information, it is the OS's responsibility to build the initial MIB or to determine and resolve the incremental differences.

VIII.3.2 Change audit

In addition to full/partial audit, the OS may also desire to get audit information only for the changes made in the NE's MIB since the last audit. This requirement can be met via the use of existing notifications such as create, delete or attribute value change emitted from the reference data objects. This information can be captured by the OS to determine and resolve the incremental differences.

If a true change audit is desired, then the creation, deletion and attribute value change notifications should be logged. The simpleScanner can then be used to retrieve from the log by specifying the timeSelection package in the scanner.

VIII.4 Mechanisms

VIII.4.1 Use of PT-GET service

The manager may initiate a request using the combination of PT-GET service defined in Recommendation X.730 and the scoping and filtering features available in CMISE (Recommendations X.710 and X.711). In order to apply this capability, the support of the profile ISO ISP 11183-Part 2, enhanced communications functions is required. This part of the network management profile makes the optional functional units (specifically the managed object selection, multiple reply, filter and cancel-get) mandatory. In addition, at the association establishment time, these functional units of CMIS should be negotiated for use over the association.

With this mechanism, the manager may issue a single request to retrieve data subject to different criteria. Examples of the conditions (non-exhaustive) that the manager may request the agent to apply in order to determine whether a response should be issued are given below. A separate reply will be sent by the agent for every managed object that meets the conditions provided in the request. At any time during which multiple responses are sent, the manager may cancel the request by issuing a cancel get request.

The CMISFilter parameter is defined in terms of attribute value assertion. It is a Boolean expression that evaluates to true or false. Relative to the model in this Recommendation, assertions may be used to retrieve data on the existing control managed objects, reference data objects. Using this capability, values of attributes of these managed objects are obtained.

A Audit of controls

The following CMISFilter constructs permit selective retrieval based on the type of controls. Note that several of these conditions can be combined if information related to more than one type of controls is to be retrieved by the manager.

a)	Call Gapping	{objectClass = destinationCodeControl}

b)	Cancel direct/alternate routing	{objectClass = cancelTo}
c)	Cancel alternate routing From	{objectClass = cancelFrom}
d)	Skip	{objectClass = skip}
e)	TAR to	{objectClass = tarTo}
f)	TAR from	{objectClass = tarFrom}
g)	Cancel rerouted overflow	{objectClass = cancelRerouted}
h)	ACC	{objectClass = acc}
i)	Selective circuit reservation	{objectClass = scr}
j)	Auto destination	{objectClass = adc}
k)	All manual protective controls	{objectClass = cancelFrom OR objectClass = cancelTo OR objectClass = skip}
l)	Expansive Controls	{objectClass = tarTo OR objectClass = tarFrom}
m)	All controls	{present trafficControlId}
n)	All active automatic controls	{present trafficControlId AND administrativeState = unlocked AND autoActivated = true}

B Reference data

The following types of assertions may be used to retrieve data that are referred to as reference data in the model.

Note that this list is not exhaustive, but is only provided as an example.

a)	Circuit Subgroups	{objectClass = circuitSubGroupEndPoint}
b)	CSGs that are monitored	{objectClass=circuitSubGroupEndPoint AND tmSurveillance = true}
c)	CSGs with a specific far end	{objectClass=circuitSubGroupEndPoint AND labelOfFarEndExchange = <value of the specific exchange >}
d)	HTR destination	{objectClass = htrDestination}
e)	observedDestination	{objectClass = observedDestination}

This mechanism can be used for full audit.

VIII.4.2 Use of simple scanner

The mechanism described above results in a single request followed by a number of responses for each selected objects. Depending on the condition specified, the number of replies linked to the request may be very large. To reduce the number of responses, another mechanism may be used that will provide the response by combining into a single report responses applicable to multiple objects. Appendix V describes how simple scanner may be used to retrieve data from historyData data objects. The audit capability may be supported using the same mechanism. In this case the managed objects subject to audit are derived by configuring the scoped selection package with conditions specified in the previous mechanism, as described in Appendix V. This would be done via polling (as opposed to autonomous report).

It must be noted that while this mechanism is very useful in packaging responses when performing the full audit, some implementations may not support sending and receiving more than 10 000 octets of session protocol data unit (the application encoded data will have to be a few octets less to allow for the header information from session and presentation layers). This scenario is possible because ISP 11183-Part 1 requires an implementation to support as a minimum only 10K octets for sending and receiving data. In that case, reply can be provided multiple link replies.

VIII.4.3 Use of discovery object

The first mechanism described in VIII.3.1 requires that the agent supports the scoping and filtering features in CMISE on the interface. With the mechanism in VIII.3.2, even though these features are not required to be sent in the request by the manager, these capabilities are part of the simple scanner in the agent. Recommendation X.750 defines a managed object class called discoveryObject. This managed object supports a manager to obtain management knowledge when the agent does not support the scoping and filtering feature.

The discoveryObject definition in Recommendation X.750 includes an action mITSearch. Using this action, the manager obtains the instances of the various managed objects. Optionally, the manager can also request the objectClass attribute for those instances. This approach is useful if the manager requires only the names of the instances and the class. However, if values of the other attributes are also required for the audit, further PT-GETs by the manager are necessary.

The manager may issue individual m-Get requests to retrieve data for all or a subset of the instances included in the response from the discoveryObject. Otherwise, the manager may create a simpleScanner object including the list of managed objects from which data must be retrieved along with the attributes whose values should be included in the report.

VIII.5 Conclusions

This Appendix describes how the audit capability may be supported with the existing model. By combining features of CMISE and information models available in ITU-T Recommendations, audits of the reference data are supported.

APPENDIX IX

Examples of addressing errors in Q.823

IX.1 Introduction

In implementing a Q3 interface using the protocol requirements in Recommendations Q.811 and Q.812 and an application specific information model such as Q.823 errors may occur at different levels. These can be categorized as:

– generated by the protocol machine at various layers including ACSE and CMISE;

– generic errors defined in CMISE but detected by the user of CMISE (application);

– application and resource specific errors.

This Appendix describes how the generic errors defined in CMISE may be applied to the traffic management model and services described in this Recommendation. This is described using an example scenario and is not to be considered as all inclusive types of errors.

IX.2 Protocol machine generated errors

These errors arise as a result of violation of the state transitions described for the protocol machine or rules for using the services of the protocol.

As an example, the transport layer provisions are given to detect out of sequence transport PDUs and procedures to recover from errors. At the presentation layer, if the received encoded data does not belong to the agreed set of abstract syntaxes (negotiated via the definition context set), an abort will be generated. In addition, if no extensibility rules are provided, presentation aborts may occur if one system sends for example an enumerated value (that is included in a revised document) to another system that has not been upgraded to recognize this new value.

An example of a violation of a rule that can cause an error to the application is the requirement in CMIP that an association must be established prior to sending CMIP PDUs. If the application issues a CMIP PDU without an association this will be an error.

Subclause IX.3 discusses the use of errors defined in CMISE for use by an application.

IX.3 Use of CMISE errors

In order to illustrate the use of generic errors defined in CMISE, consider the following example. The traffic management model defines a tmCircuitEndPointSubgroup as a subclass of circuitEndPointSubgroup defined in Recommendation M.3100. The subclass includes an attribute called tmSurveillance. The behaviour of this object class has an integrity constraint satisfying the following pre- and post-conditions.

If tmSurveillance is set to true, then a tmCircuitEndPointSubgroupCurrentData must exist. The precondition for creating the current data is tmSurveillance must be settable to true. If the current data is deleted, then the attribute must be set to false. In other words, the post condition for deleting the current data object is tmSurveillance attribute value must be false. If there is a mismatch in this behaviour, namely the value of tmSurveillance is true without a current data object, then the integrity constraint is violated and this will be a non-conforming implementation. The same is true for the reverse.

Let us assume that the manager is interested in collecting surveillance data for circuit endpoint subgroups. A CMIP request to set the value of an attribute may be issued in one of the following methods.

1) a request to modify value(s) of an attribute(s) on a single managed object using an unconfirmed set operation;
2) a request to modify value(s) of attribute(s) on a single managed object using a confirmed set operation;
3) a request to modify value(s) of attribute(s) on multiple objects using scoping (irrespective of with or without filtering) using an unconfirmed operation;
4) a request to modify value(s) of attribute(s) on multiple objects using scoping (irrespective of with or without filtering) using a confirmed operation.

The last two cases may be further split into two modes of synchronization: atomic and best effort.

The following describes the errors that may arise in setting the tmSurveillance attribute.

In the first case, if the system is unable to create the current data object an error response will not be provided. However, if this mode of set is chosen, then the Event Forwarding discriminator must be configured such that create notifications will be issued. Absence of the create notifications will indicate that the current data was not created. It is also possible to issue a read on the attribute to determine this result. How this error is shown in an operators screen is outside the scope of the machine-machine to interface described in this Recommendation. Another option is to have the tmCircuitEndPointSubgroup send an attribute value change notification with the old and new value of the tmSurveillance attribute.

If the attribute tmSurveillance is already true and a current data instance is present, requesting a set to true will have no effect. The behaviour expected when the current data already exist must be specified.

In the second case, an error response is possible to the set request. The error response may use one of the suitable codes (access denied, classInstanceConflict, noSuchObjectClass, noSuchObjectInstance, processingFailure with no additional information assuming there are no parameter templates defined for this object class, setListError with or without the value provided in the request). If the operation is not performed because of Resource limitations, a RO-Reject PDU with the error code of resourceLimitation may be issued. Receipt of this error and/or non-receipt of the object creation notification of the current data should signal that an appropriate error message is provided to the operator. Again the translation of the error code in the PDU to the user friendly interface is outside the scope of TMN.

The third case is similar to the first case, except the tmSurveillance attribute in multiple objects are to be set if corresponding current data objects are created. Because this is an unconfirmed request, the series of object creation notifications will be received for the created current data objects. These notifications will include the names of the current data (they include the name of the containing circuitEndPointSubgroup) object. These notifications may then be used to determine if all the current data objects have been created. This assumes that the manager has knowledge of the names of the circuit endpoint subgroups for which measurements are required. Based on the create notifications, the manager can now provide the operator a user friendly message (outside the scope of this Recommendation) indicating where errors were encountered. The operator may then request a role back. This would imply the translation between the user interface and CMIP interface would have to delete all the instances of current data for which creation notifications have been received. When the delete request is issued, the integrity constraints will require that the tmSurveillance attribute in the contained object be set to false. (This should be stated in the name binding of the specific current data object classes).

The fourth case is similar to case 2. Linked replies with setListError and/or object creation notifications may be analysed to determine the circuit endpoint subgroups for which current data objects have not been created.

The above descriptions of cases 3 and 4 are applicable if the operation either specified the synchronization parameter to be best effort or no value for this parameter was included in the request. If it is desired that the current data objects must be created only if all of them are creatable, then atomic synchronization should be specified. How this is implemented in terms of locking the information until all the operations are completed is outside the scope of the model. In addition, if there is a communication link failure during any of the cases, then recovery procedures will be required to assure consistency of the MIB. This is again outside the scope of the model.

Another example of where errors may occur relates to management of functions not present in a particular type of network element. For example the attribute destinationAspect includes enumerated values corresponding to functions such as hard-to-reach. In any attribute definition, values may be provided to encompass a variety of network element types with varying capabilities. If a set request is issued to set the value corresponding to say hard-to-reach and the NE does not support this functionality, then the CMISE error invalidAttribute value will be issued (assuming this was a confirmed request). The restrictions on values supported in an implementation of a product must be specified in the MOCS proforma. This would then be used to understand why the invalid attribute value is received even though this is a valid value defined in the standard.

IX.4 Use of specific errors

In addition to the generic errors defined by CMISE, there exists as part of CMIP definition the ability to include resource specific errors. These specific errors will have to be specified using parameter templates and identified in the information model (e.g. linked to an attribute to indicate that a specific error with the specific syntax may be issued when modifying an attribute).

The following example was made up to illustrate the point.

Let us suppose that a network element is designed to support the "Cancel To" control and because of the resource limitation the supplier specifies the maximum number of cancelTo objects that can be supported by the NE. The create request will fail if, as a result of the create, the allowed number of cancelTo managed objects is exceeded. A generic reject called resourceLimitation can be provided. On the other hand, in Recommendation M.3100 a specific error called createError parameter is defined. If this parameter is included in the name binding of the cancelTo object class, then a processing failure error accompanied by an integer representing the maximum number of cancelTo objects supportable by the network element can be provided. The user interface can then translate this to a meaningful string and display it to the operator.

IX.5 Conclusion

As shown above, most of the error conditions can be supported by the generic errors defined in CMISE by specifying integrity constraints. In addition, specific errors may be included but they should be included in the templates for managed object class, attribute, action or notification. Errors such as time out and link failure are outside the scope of the information model. These conditions have to be determined by interface implementation specifications. No specific errors have been identified in the model.

APPENDIX X

Control hierarchy

In the absence of control hierarchy in Recommendation E.412, when multiple controls are simultaneously active in the exchange, the following rules apply in order to determine the hierarchy of controls:

- Since destination is determined before hunting for circuit subgroups, the destination controls takes precedence over any circuit subgroup controls.
- Circuit subgroup controls are then applied at two stages: Stage 1 – before searching for idle member in the circuit subgroup; and Stage 2 – after overflowing from all circuits in circuit subgroup but before hunting the next alternate circuit subgroup in the routing chain. Controls at Stage 1 takes precedence over controls at Stage 2.

The following descending hierarchy of traffic control shall be considered during the call handling process:

1)	Destination Code Control		
2)	Automatic Destination Control		
3)	TAR To	)	
4)	Cancel To	)	
5)	Skip	)	Stage 1
6)	Selective Circuit Reservation	)	
7)	ACC	)	

8)	Cancel Rerouted Overflow	)	
9)	Tar From	)	Stage 2
10)	Cancel From	)	

When the precedence between controls is not specified in this Appendix, it is considered as a local matter.

ITU-T RECOMMENDATIONS SERIES

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