INTERNATIONAL STANDARD

ISO/IEC 8473-1

Second edition
1994-mm-dd

Information technology — Protocol for providing the connectionless–mode network service

Technique de l’information — Protocole fournissant le service deréseau en mode sans connexion

Reference number
ISO/IEC 8473-1 : 1994 (E)

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 8473-1 was prepared by the Joint Technical Committee ISO/IEC JTC1, *Information technology*. The identical text is published as ITU-T Recommendation X.233.

The second edition of ISO/IEC 8473-1 incorporates amendments 1, 2, 3, 4, and 6, and technical corrigendum 1, to the first edition of ISO/IEC 8473 (1988).

ISO/IEC 8473 consists of the following parts, under the general title *Information technology — Protocol for providing the connectionless–mode network service:*

— Part 1: Protocol specification

— Parts 2–*n*: Provision of the underlying service by specific types of subnetwork

Introduction

This is one of a set of Recommendations and International Standards produced to facilitate the interconnection of open systems. The set covers the services and protocols required to achieve such interconnection.

This Recommendation | International Standard is positioned with respect to other related Recommendations and International Standards by the layers defined in ITU-T Rec. X.200 | ISO/IEC 7498-1. In particular, it is a protocol of the Network layer. The protocol specified by this Recommendation | International Standard may be used between Network entities in end systems, between Network entities in intermediate systems, or between a Network entity in an end system and a Network entity in an intermediate system. In an end system, it provides the connectionless-mode Network service defined in CCITT Rec. X.213 | ISO/IEC 8348.

The interrelationship of the protocol specification and the related service definitions is illustrated in Figure 1.

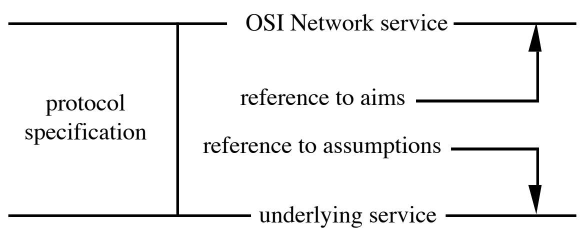

Figure 1 — Interrelationship of protocol and services

In order to evaluate the conformance of a particular implementation of this protocol, it is necessary to have a statement of which of the protocol's capabilities and options have been implemented. Such a statement is called a Protocol Implementation Conformance Statement (PICS), as defined in CCITT Rec. X.290 | ISO/IEC 9646-1. A PICS proforma, from which a PICS may be prepared for a specific implementation, is included in this Recommendation | International Standard as normative Annex A.

INTERNATIONAL STANDARD

ITU-T RECOMMENDATION

INFORMATION TECHNOLOGY — PROTOCOL FOR PROVIDING THE CONNECTIONLESS–MODE NETWORK SERVICE: PROTOCOL SPECIFICATION

1 Scope

This Recommendation | International Standard specifies a protocol that is used to provide the connectionless-mode Network service described in CCITT Rec. X.213 | ISO/IEC 8348 and to perform certain Network layer management functions. The protocol relies upon the provision of an underlying connectionless-mode service by real subnetworks and/or data links. The underlying connectionless-mode service assumed by the protocol may be obtained either directly, from a connectionless-mode real subnetwork, or indirectly, through the operation of an appropriate Subnetwork Dependent Convergence Function (SNDCF) or Protocol (SNDCP) over a connection-mode real subnetwork, as described in ISO/IEC 8648. This Recommendation | part of this International Standard specifies the operation of the protocol with respect to a uniform, abstract "underlying subnetwork service". Other Recommendations | parts of this International Standard specify the way in which this "underlying subnetwork service" is obtained from real subnetworks, such as those which conform to ISO/IEC 8802 or ISO/IEC 8208. The "underlying subnetwork service" may be obtained from real subnetworks other than those that are specifically covered by the other Recommendations | parts of this International Standard.

This Recommendation | International Standard specifies

a) procedures for the connectionless transmission of data and control information from one Network entity to a peer Network entity;

b) the encoding of the protocol data units (PDUs) used for the transmission of data and control information, comprising a variable-length protocol header format;

c) procedures for the correct interpretation of protocol control information; and

d) the functional requirements for implementations claiming conformance to this Recommendation | International Standard.

The procedures are defined in terms of

a) the interactions among peer Network entities through the exchange of protocol data units;

b) the interactions between a Network entity and a Network service user through the exchange of Network service primitives; and

c) the interactions between a Network entity and an abstract underlying service provider through the exchange of service primitives.

This Recommendation | International Standard also provides the PICS proforma for this protocol, in compliance with the relevant requirements, and in accordance with the relevant guidance, given in CCITT Rec. X.290 | ISO/IEC 9646-1.

2 Normative references

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent editions of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of the currently valid ITU-T Recommendations.

2.1 Identical Recommendations | International Standards

— ITU-T Recommendation X.200 (1994) | ISO/IEC 7498-1 : 1994, *Information technology — Open Systems Interconnection — Reference Model: Basic Reference Model*.

— ITU-T Recommendation X.210 (1993) | ISO/IEC 10731 : 1993, *Information technology — Open Systems Interconnection — Conventions for the definition of OSI services*.

— CCITT Recommendation X.213 (1992) | ISO/IEC 8348 : 1992, *Information technology — Network service definition for Open Systems Interconnection*.

2.2 Paired Recommendations | International Standards identical in technical content

— ITU-T Recommendation X.224 (1993), *Protocol for providing the OSI connection-mode Transport service*.

ISO/IEC 8073 : 1992, *Information technologgy — Open Systems Interconnection — Protocol for providing the connection-mode Transport service*.

— CCITT Recommendation X.290 (1992), *OSI conformance testing methodology and framework for protocol Recommendations for CCITT applications — General concepts*.

ISO/IEC 9646-1 : 1991, *Information technology — Open Systems Interconnection — Conformance testing methodology and framework — Part 1: General concepts*.

2.3 Additional references

— ITU-T Recommendation X.25 (1993), *Interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE) for terminals operating in the packet mode and connected to public data networks by dedicated circuit*.

— ISO/IEC 8208 : 1990, *Information technology — Data communications — X.25 Packet Layer Protocol for Data Terminal Equipment*.

— ISO/IEC 8648 : 1988, *Information processing systems — Open Systems Interconnection — Internal organization of the network layer*.

— ISO/IEC 8802 : 1990, *Information processing systems — Data communications — Local area networks*.

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply.

3.1 Reference model definitions

This Recommendation | International Standard makes use of the following terms defined in ITU-T Rec. X.200 | ISO/IEC 7498-1:

a) end system

b) Network entity
c) Network layer
d) Network protocol
e) Network protocol data unit
f) Network relay
g) Network service
h) Network service access point
i) Network service access point address
j) routeing
k) service
l) service data unit
m) service primitive

3.2 Service conventions definitions

This Recommendation | International Standard makes use of the following terms defined in ITU-T Rec. X.210 | ISO/IEC 10731:

a) service provider
b) service user

3.3 Network layer architecture definitions

This Recommendation | International Standard makes use of the following terms defined in ISO/IEC 8648:

a) intermediate system
b) relay system
c) subnetwork
d) subnetwork dependent convergence protocol
e) subnetwork dependent convergence function
f) subnetwork independent convergence protocol
g) subnetwork independent convergence function
h) subnetwork access protocol

3.4 Network layer addressing definitions

This Recommendation | International Standard makes use of the following terms defined in CCITT Rec. X.213 | ISO/IEC 8348:

a) Network addressing domain
b) Network protocol address information
c) subnetwork point of attachment

3.5 Local area network definitions

This Recommendation | International Standard makes use of the following term defined in ISO/IEC 8802:

a) local area network

3.6 PICS definitions

This Recommendation | International Standard makes use of the following terms defined in CCITT Rec. X.290 | ISO/IEC 9646-1:

a) PICS proforma

b) protocol implementation conformance statement

3.7 Additional definitions

3.6.1 derived PDU: A protocol data unit the fields of which are identical to those of an initial PDU, except that it carries only a segment of the user data from an N-UNITDATA request.

3.6.2 initial PDU: A protocol data unit carrying the whole of the user data from an N-UNITDATA request.

3.6.3 local matter: A decision made by a system concerning its behaviour in the Network layer that is not prescribed or constrained by this Recommendation | International Standard.

3.6.4 Network entity title: An identifier for a Network entity which has the same abstract syntax as an NSAP address, and which can be used to unambiguously identify a Network entity in an end or intermediate system.

3.6.5 reassembly: The act of regenerating an initial PDU from two or more derived PDUs.

3.6.6 segment: A distinct unit of data consisting of part of the user data provided in the N-UNITDATA request and delivered in the N-UNITDATA indication.

3.6.7 segmentation: The act of generating two or more derived PDUs from an initial or derived PDU. The derived PDUs together carry the entire user data of the initial or derived PDU from which they were generated.

4 Abbreviations

4.1 Data units

NSDU	Network service data unit
PDU	protocol data unit
SDU	service data unit
SNSDU	subnetwork service data unit

4.2 Protocol data units

DT PDU	data protocol data unit
ER PDU	error report protocol data unit
ERP PDU	echo reply protocol data unit
ERQ PDU	echo request protocol data unit

4.3 Protocol data unit fields

DA	destination address
DAL	destination address length
DUID	data unit identifier
E/R	error report flag
LI	length indicator
LT	lifetime
MS	more segments flag
NLPID	Network layer protocol identifier
SA	source address
SAL	source address length
SL	segment length
SO	segment offset

SP segmentation permitted flag

4.4 Parameters

DA destination address
QoS quality of service
SA source address

4.5 Miscellaneous

CLNP connectionless-mode network protocol (i.e. the protocol defined in this Recommendation | International Standard)
NPAI Network protocol address information
NS Network service
NSAP Network service access point
PICS protocol implementation conformance statement
SN subnetwork
SNAcP subnetwork access protocol
SNDCF subnetwork dependent convergence function
SNDCP subnetwork dependent convergence protocol
SNICP subnetwork independent convergence protocol
SNPA subnetwork point of attachment

5 Overview of the protocol

5.1 Internal organization of the Network layer

The architectural organization of the Network layer is described in ISO/IEC 8648. ISO/IEC 8648 identifies and categorizes the way in which functions can be performed within the Network layer by Network layer protocols, thus providing a uniform framework for describing how protocols operating either individually or cooperatively in the Network layer can be used to provide the OSI Network service. This protocol is designed to be used in the context of the internetworking protocol approach to the provision of the connectionless-mode Network service defined in ISO/IEC 8648.

This protocol is intended for use in the Subnetwork Independent Convergence Protocol (SNICP) role. A protocol which fulfills the SNICP role operates to construct the OSI Network service over a defined set of underlying services, performing functions which are necessary to support the uniform appearance of the OSI connectionless-mode Network service over a homogeneous or heterogeneous set of interconnected subnetworks. This protocol is defined to accommodate variability where subnetwork dependent convergence protocols and/or subnetwork access protocols do not provide all of the functions necessary to support the connectionless-mode Network service over all or part of the path from one Network Service Access Point (NSAP) to another.

As described in ISO/IEC 8648, a protocol at the Network layer may fulfill different roles in different configurations. Although this protocol is designed particularly to be suitable for a SNICP role in the context of the internetworking protocol approach to the provision of the connectionless-mode Network service, it may also be used to fulfill other roles, and may therefore be used in the context of other approaches to subnetwork interconnection.

The operation of this protocol is specified with respect to an “underlying subnetwork service” which is made available through the operation of other Network layer protocols or through provision of the Data Link service. The “underlying subnetwork service” assumed by this protocol is described in 5.5.

5.2 Subsets of the protocol

Two subsets of the full protocol are defined, which exploit the known subnetwork characteristics of particular configurations and are therefore not subnetwork independent.

The Inactive Network Layer Protocol Subset is a null-function subset which can be used when it is known that the source and destination end systems are connected by a single subnetwork, and when none of the functions performed by the full protocol is required to provide the connectionless-mode Network service between any pair of end systems.

The Non-segmenting Protocol Subset permits simplification of the header when it is known that the source and destination end systems are connected by subnetworks whose individual service data unit sizes are greater than or equal to a known bound which is large enough so that segmentation is not required. This subset is selected by setting the segmentation permitted flag to zero (see 6.7).

5.3 Addresses and titles

The following clauses describe the addresses and titles used by this protocol.

5.3.1 Addresses

The source address and destination address parameters referred to in 7.3 are NSAP addresses. The syntax and semantics of an NSAP address are described in CCITT Rec. X.213 | ISO/IEC 8348.

The encoding used by this protocol to convey NSAP addresses is the "preferred encoding" specified in CCITT Rec. X.213 | ISO/IEC 8348. The NSAP address, encoded as a string of binary octets according to CCITT Rec. X.213 | ISO/IEC 8348, is conveyed in its entirety in the address fields described in 7.3.

5.3.2 Network entity titles

A Network Entity Title (NET) is an identifier for a Network entity in an end system or intermediate system. Network entity titles are allocated from the same name space as NSAP addresses, and the determination of whether a name is an NSAP address or a Network entity title depends on the context in which the name is interpreted. The values of the source route and record route parameters defined in 7.5.4 and 7.5.5 respectively are Network entity titles. The values of the source address and destination address parameters in the Error Report PDU defined in 7.9, in the Echo Request PDU defined in 7.10, and in the Echo Response PDU defined in 7.11 are also Network entity titles.

The encoding used by this protocol to convey Network entity titles is the "preferred encoding" specified in CCITT Rec. X.213 | ISO/IEC 8348. The Network entity title, encoded as a string of binary octets according to CCITT Rec. X.213 | ISO/IEC 8348, is conveyed in its entirety in the appropriate fields.

5.4 Service provided by the protocol

This protocol provides the connectionless-mode Network service described in CCITT Rec. X.213 | ISO/IEC 8348. The relevant Network service primitive and its parameters are shown in Table 1.

NOTE — CCITT Rec. X.213 | ISO/IEC 8348 states that the maximum size of a connectionless-mode Network service data unit (NSDU) is 64512 octets.

Primitive		Parameters
N-UNITDATA	Request Indication	NS-Source-Address, NS-Destination-Address, NS-Quality-of-Service, NS-Userdata

Table 1 — Connectionless-mode Network service primitive

5.5 Underlying service assumed by the protocol

It is intended that this protocol be capable of operating over connectionless-mode services derived from a wide variety of real subnetworks and data links. Therefore, in order to simplify the specification of the protocol, its operation is defined (in clause 6) with respect to an abstract "underlying subnetwork service" rather than any particular real subnetwork service. This underlying service consists of a single SN-UNITDATA primitive which conveys the source and destination subnetwork point of attachment addresses, a subnetwork quality of service parameter, and a certain number of octets of user data.

The SN-UNITDATA primitive is used to describe the abstract interface that exists between the CLNP protocol machine and an underlying real subnetwork or a subnetwork dependent convergence function that operates over a real subnetwork or real data link to provide the required underlying service.

The primitive provided and its parameters are shown in Table 2.

Provision of the “underlying subnetwork service” by real subnetworks and data links is described in clause 8 and in other Recommendations | parts of this International Standard.

Primitive		Parameters
SN-UNITDATA	Request Indication	SN-Source-Address, SN-Destination-Address, SN-Quality-of-Service, SN-Userdata

Table 2 — Underlying service primitive

6 Protocol functions

This clause describes the functions performed as part of the protocol.

Not all of the functions must be performed by every implementation. 6.21 specifies which functions may be omitted, and the correct behaviour when requested functions are not implemented.

6.1 PDU composition function

This function is responsible for the construction of a protocol data unit according to the rules governing the encoding of PDUs given in clause 7. The Protocol Control Information (PCI) required is determined from current state and local information and from the parameters associated with the N-UNITDATA request.

Network Protocol Address Information (NPAI) for the source address and destination address fields of the PDU header is derived from the NS-Source-Address and NS-Destination-Address parameters. The NS-Destination-Address and NS-Quality-of-Service parameters, together with current state and local information, are used to determine which optional functions are to be selected. User data passed from the Network service user (NS-Userdata) form the data part of the protocol data unit.

During the composition of the protocol data unit, a Data Unit Identifier (DUID) is assigned to distinguish this request to transmit NS-Userdata to a particular destination Network service user from other such requests. The originator of the PDU shall choose the DUID so that it remains unique (for this source and destination address pair) for the maximum lifetime of the Initial PDU in the network; this rule applies for any PDUs derived from the Initial PDU as a result of the application of the segmentation function (see 6.7). Derived PDUs are considered to correspond to the same Initial PDU, and hence to the same N-UNITDATA request, if they have the same source address, destination address, and data unit identifier.

The DUID is also available for ancillary functions such as error reporting (see 6.10).

The total length of the PDU in octets is determined by the originator and placed in the total length field of the PDU header. This field is not changed for the lifetime of the protocol data unit, and has the same value in the Initial PDU and in each of any Derived PDUs that may be created from the Initial PDU.

When the non-segmenting protocol subset is employed, neither the total length field nor the data unit identifier field is present. The rules governing the PDU composition function are modified in this case as follows. During the composition of the protocol data unit, the total length of the PDU in octets is determined by the originator and placed in the segment length field of the PDU header. This field is not changed for the lifetime of the PDU. No data unit identification is provided.

6.2 PDU decomposition function

This function is responsible for removing the protocol control information from the protocol data unit. During this process, information pertinent to the generation of the N-UNITDATA indication is determined as follows. The NS-Source-Address and NS-Destination-Address parameters of the N-UNITDATA indication are recovered from the NPAI in the source address and destination address fields of the PDU header. The data part of the received PDU is retained until all segments of the original service data unit have been received; collectively, these form the NS-Userdata parameter of the N-UNITDATA indication. Information relating to the Quality of Service (QoS) provided during the transmission of the PDU is determined from the quality of service and other information contained in the options part of the PDU header. This information constitutes the NS-Quality-of-Service parameter of the N-UNITDATA indication.

6.3 Header format analysis function

This function determines whether the full protocol or the inactive Network layer protocol is in use, and whether or not a received PDU has reached its final destination. If the Network layer protocol identifier (NLPID) field in a received PDU contains a value that identifies the protocol defined by this Recommendation | International Standard, then either the full protocol or the non-segmenting subset is in use; the header format analysis function determines whether or not the received PDU has reached its destination, using the destination address in the PDU header. If the destination address provided in the PDU identifies either a Network entity title of this Network entity or an NSAP served by this Network entity, then the PDU has reached its destination; if not, it shall be forwarded.

If the NLPID field contains a value that identifies the inactive Network layer protocol, then no further analysis of the PDU header is required. The Network entity in this case determines that either the Subnetwork Point of Attachment (SNPA) address encoded as NPAI in the supporting subnetwork protocol (see 8.1) corresponds directly to an NSAP address serviced by this Network entity, or that an error has occurred.

6.4 PDU lifetime control function

This function is used to enforce the maximum PDU lifetime. It determines whether a received PDU may be forwarded or whether its assigned lifetime has expired, in which case it shall be discarded.

The operation of the PDU lifetime control function depends upon the lifetime field in the PDU header. This field contains, at any time, the remaining lifetime of the PDU (represented in units of 500ms). The lifetime of the Initial PDU is determined by the originating Network entity and placed in the lifetime field of the PDU. If and when the segmentation function is applied to a PDU, the value of the lifetime field of the Initial PDU is copied into all of the corresponding Derived PDUs.

The value of the lifetime field of a PDU is decremented by every Network entity that processes the PDU. When a Network entity processes a PDU, it decrements the PDU lifetime by at least one. The value of the PDU lifetime field shall be decremented by more than one if the sum of

a) the transit delay in the underlying service from which the PDU was received, and

b) the delay within the system processing the PDU

exceeds or is estimated to exceed 500ms. In this case, the lifetime field shall be decremented by one for each additional 500ms of actual or estimated delay. The determination of delay need not be precise, but where a precise value cannot be ascertained, the value used shall be an overestimate, not an underestimate.

If the lifetime field reaches a value of zero before the PDU is delivered to its destination, the PDU shall be discarded. The error reporting function shall be invoked as described in 6.10. This may result in the generation of an Error Report PDU.

It is a local matter whether or not the destination Network entity performs the lifetime control function.

6.5 Route PDU function

This function determines the Network entity to which a PDU should be forwarded and the underlying service that must be used to reach that Network entity, using the destination address field and either the segment length field (if present) or the total length field (if the segment length field is not present). Where segmentation is required, the route PDU function further determines over which underlying service Derived PDUs shall be sent in order to reach that Network entity. The results of the route PDU function are passed to the forward PDU function (along with the PDU itself) for further processing.

Selection of the underlying service that shall be used to reach the “next” system in the route to the destination is initially influenced by the NS-Quality-of-Service parameter of the N-UNITDATA request, which specifies the QoS requested by the sending NS user. Whether this QoS is to be provided directly by the protocol, through the selection of the quality of service maintenance parameter and other optional parameters, or through the QoS facilities offered by each of the underlying services, or both, is determined prior to invocation of the forward PDU function. Route selection by intermediate systems may subsequently be influenced by the values of the quality of service maintenance parameter (if present), and other optional parameters (if present).

6.6 Forward PDU function

This function issues an SN-UNITDATA request primitive (see 5.5), supplying the subnetwork or SNDCF identified by the route PDU function with the protocol data unit as user data to be transmitted, the address information required by that subnetwork or SNDCF to identify the “next” system within the subnetwork-specific addressing domain (this may be an intermediate system or the destination end system), and quality of service constraints (if any) to be considered in the processing of the user data.

When the PDU to be forwarded is longer than the maximum service data unit size provided by the underlying service, the segmentation function is applied (see 6.7).

6.7 Segmentation function

Segmentation is performed when the length of a protocol data unit is greater than the maximum service data unit size supported by the underlying service to be used to transmit the PDU.

Segmentation consists of composing two or more new PDUs (Derived PDUs) from the too-long Initial or Derived PDU that is to be segmented. All of the header information from the PDU to be segmented, with the exception of the segment length and checksum fields of the fixed part, and the segment offset field of the segmentation part, is duplicated in each Derived PDU, including all of the address part, the data unit identifier and total length of the segmentation part, and the options part (if present).

NOTE — The rules for forwarding and segmentation guarantee that the header length is the same for all segments (Derived PDUs) of an Initial PDU, and is the same as the header length of the Initial PDU. The size of a PDU header therefore will not change due to the operation of any protocol function.

The user data field of the PDU to be segmented is divided and apportioned among the user data fields of the Derived PDUs in such a way that the Derived PDUs satisfy the maximum-length requirements of the SN-Userdata parameter of the SN-UNITDATA request primitive used to access the selected underlying service. The user data field of each derived PDU, except for the last, shall contain a number of octets that is a non-zero multiple of 8. Thus, the value of the segment offset field in any PDU is either zero or a non-zero multiple of 8. Segmentation shall not result in the generation of a Derived PDU containing fewer than eight octets of user data.

Derived PDUs are identified as being from the same Initial PDU by means of

a) the source address field,

b) the destination address field, and

c) the data unit identifier field.

The following fields of the PDU header are used in conjunction with the segmentation function:

a) Segment offset — identifies the octet at which the segment begins with respect to the start of the data part of the Initial PDU;

b) Segment length — specifies the number of octets in the Derived PDU, including both header and data;

c) More segments flag — set to one if this Derived PDU does not contain the final octet of the user data from the Initial PDU as its final octet of user data; and

d) Total length — specifies the number of octets in the Initial PDU, including both header and data.

Derived PDUs may be further segmented without constraining the routeing of the individual Derived PDUs.

The segmentation permitted flag is set to one to indicate that segmentation is permitted. If the Initial PDU is not to be segmented at any point during its lifetime, the flag is set to zero by the source Network entity. The setting of the segmentation permitted flag may not be changed by any other Network entity for the lifetime of the Initial PDU and any Derived PDUs.

6.8 Reassembly function

The reassembly function reconstructs the Initial PDU from the Derived PDUs generated by the operation of the segmentation function on the Initial PDU (and, recursively, on subsequent Derived PDUs).

A bound on the time during which segments (Derived PDUs) of an Initial PDU may be held at a reassembly point before being discarded is provided, so that reassembly resources may be released when it is no longer expected that missing segments of the Initial PDU will arrive at the reassembly point. Upon reception of a Derived PDU, a reassembly timer shall be initiated with a value that indicates the amount of time that shall elapse before any unreceived (missing) segments of the Initial PDU are assumed to be lost. When this timer expires, all segments (Derived PDUs) of the Initial PDU held at the reassembly point shall be discarded, the resources allocated for those segments may be freed, and, if selected, an error report shall be generated (see 6.10).

While the exact relationship between reassembly lifetime and PDU lifetime is a local matter, the reassembly function shall preserve the intent of the PDU lifetime. Consequently, the reassembly function shall discard PDUs whose lifetime would otherwise have expired had they not been under the control of the reassembly function; that is, the reassembly lifetime for a given PDU shall be less than the PDU lifetime in all derived PDUs being held at the reassembly point.

NOTES

1 Methods of bounding reassembly lifetime are discussed in Annex B.

2 The segmentation and reassembly functions are intended to be used in such a way that the fewest possible segments are generated at each segmentation point and reassembly takes place at the final destination of a PDU. However, other schemes which

a) interact with the routeing algorithm to favor paths on which fewer segments are generated, or

b) generate more segments than absolutely required in order to avoid additional segmentation at some subsequent point

are not precluded. The information necessary to enable the use of one of these alternative strategies may be made available through the operation of a Network layer management function or by other means.

3 The originator of the Initial PDU determines the value of the segmentation permitted flag in the Initial PDU and all Derived PDUs (if any). An intermediate system may not change this value in the Initial PDU or any PDU derived from it, and may not therefore add or remove the segmentation part of the header.

6.9 Discard PDU function

This function performs all of the actions necessary to free the resources reserved by the Network entity when any of the following situations is encountered.

NOTE 1 — This list is not exhaustive.

a) A violation of protocol procedure has occurred.

b) A PDU is received whose checksum is inconsistent with its contents.

c) A PDU is received, but due to local congestion, it cannot be processed.

d) A PDU is received whose header cannot be analyzed.

e) A PDU is received which cannot be segmented and cannot be forwarded because its length exceeds the maximum service data unit size supported by any underlying service available for transmission of the PDU to the next Network entity on the chosen route.

f) A PDU is received whose destination address is unreachable or unknown.

g) Incorrect or invalid source routeing was specified. This may include a syntax error in the source routeing field, an unknown or unreachable Network entity title in the source routeing field, or a path which is not acceptable for other reasons.

h) A PDU is received whose PDU lifetime has expired or whose lifetime expires during reassembly.

i) A PDU is received which contains an unsupported option corresponding to a Type 2 function (see 6.21).

NOTE 2 — In general, it is not always possible to determine whether a destination NSAP address is invalid (does not follow CCITT Rec. X.213 | ISO/IEC 8348), unprocessable (in that there is no routeing table entry for the address), or incorrectly coded (as NPAI). Therefore, with respect to generating an Error Report PDU, the situation described in (f) may or may not be distinguished from the situation described in (d), and the "reason for discard" (see 6.10 and Table 8) may be "header syntax error" or "destination address unknown".

6.10 Error reporting function

6.10.1 Overview

This function attempts to return an Error Report PDU to the source Network entity when a protocol data unit originated by that Network entity is discarded in accordance with 6.9.

The Error Report PDU identifies the discarded PDU, specifies the type of error detected, and identifies the location in the header of the discarded PDU at which the error was detected. At least the entire header of the discarded PDU and, at the discretion of the originator of the Error Report PDU, none, all, or part of the data part of the discarded PDU are placed in the data part of the Error Report PDU.

The originator of a PDU controls the subsequent generation of Error Report PDUs that refer to it. The error report (E/R) flag in the original PDU is set by the source Network entity to indicate that an Error Report PDU is to be generated if the Initial PDU or any PDUs derived from it are discarded; if the flag is not set, error reports are not generated.

NOTES

1) The suppression of Error Report PDUs is controlled by the originating Network entity and not by the NS user. Care should be exercised by the originator with regard to suppressing ER PDUs so that error reporting is not suppressed for every PDU generated.

2) Non-receipt of an Error Report PDU does not imply correct delivery of a PDU issued by a source Network entity.

6.10.2 Requirements

An Error Report PDU shall not be generated to report the discard of an Error Report PDU.

An Error Report PDU shall not be generated to report the discard of a PDU unless that PDU has the error report flag set to allow error reports.

If a PDU is discarded, and the error report flag in the discarded PDU is set to allow error reports, an Error Report PDU shall be generated if the reason for discard is one of the reasons for discard enumerated in 6.9, subject to the conditions described in 6.10.4. If a PDU with the E/R flag set to allow error reports is discarded for any other reason, an ER PDU may be generated (as an implementation option).

Error reports may be suppressed in circumstances in which the validity of the information in the PDU that caused the error condition is uncertain. These circumstances include, but are not limited to, those described in items b, c, and d of 6.9.

6.10.3 Processing of error reports

An Error Report PDU is composed from information contained in the header of the discarded PDU to which the error report refers. The contents of the source address field of the discarded PDU are used as the destination address of the Error Report PDU. This value, which in the context of the discarded PDU was used as an NSAP address, is used in the context of the Error Report PDU as the Network entity title of the Network entity that originated the discarded PDU. The Network entity title of the originator of the Error Report PDU is conveyed in the source address field of the header of the Error Report PDU. The value of the lifetime field is determined in accordance with 6.4. Optional parameters are selected in accordance with 6.10.4.

The segmentation of Error Report PDUs is not permitted; hence, no segmentation part is present. The total length of the ER PDU in octets is placed in the segment length field of the ER PDU header. This field is not changed during the lifetime of the ER PDU. If the originator of the ER PDU determines that the size of the ER PDU exceeds the maximum service data unit size of the underlying service, the ER PDU shall be truncated to the maximum service data unit size (see 8.3) and forwarded with no other change. Error Report PDUs are routed and forwarded by intermediate system Network entities in the same way as Data PDUs.

NOTE — The requirement stated in 8.3 that the underlying service assumed by the protocol shall be capable of supporting a service data unit size of 512 octets guarantees that at least the entire header of the discarded PDU can be conveyed in the data part of an ER PDU.

When an ER PDU is decomposed upon reaching its destination, information that may be used to interpret and act upon the error report is obtained as follows. The Network entity title recovered from the NPAI in the source address field of the ER PDU header is used to identify the Network entity that generated the error report. The reason for generating the error report is extracted from the options part of the PDU header. The entire header of the discarded PDU, and part or all of the original user data (if present), are extracted from the data part of the ER PDU to assist in ascertaining the nature of the error.

6.10.4 Relationship of discarded PDU options to error reports

The generation of an error report is affected by options that are present in the corresponding discarded PDU. The presence of options in the discarded PDU that are not supported by the system that has discarded that PDU, or the presence of an unrecognized PDU type code parameter, may cause the suppression of an error report even if the discarded PDU indicated that an error report should be generated in the event of a discard.

The processing of an error report is also affected by options that are present in the corresponding discarded PDU. In particular, options selected in the discardedPDU affect which options are included in the corresponding Error Report PDU. The selection of options for an Error Report PDU is governed by the following requirements:

a) If the priority, QoS maintenance, or security option is selected in the discarded PDU, and the system generating the Error Report PDU supports the option, then the Error Report PDU shall specify the same option, using the value that was specified in the discarded PDU.

b) If the system generating the Error Report PDU does not support the security option, an error report shall not be generated for a discarded PDU that selected the security option.

c) If the complete source route option is selected in the discarded PDU, and the system generating the Error Report PDU supports the option, then the Error Report PDU shall specify the complete source route option. The source route parameter value is obtained by extracting from the discarded PDU that portion of the complete source route list that has already been processed, and reversing the order of Network entity titles which comprise that portion of the list.

d) If the system generating the Error Report PDU does not support the complete source route option, an Error Report PDU shall not be generated for a discarded PDU that selects the complete source route option.

e) The padding, partial source route, and record route options, if supported, may be specified in the Error Report PDU.

NOTE — The values of the optional parameters in (e) above may be derived as a local matter, or they may be based upon the corresponding values in the discarded PDU.

6.11 PDU header error detection function

The PDU header error detection function protects against failure of intermediate or end system Network entities due to the processing of erroneous information in the PDU header. The function is realized by a checksum computed on the entire PDU header. The checksum is verified at each point at which the PDU header is processed. If the checksum calculation fails, the PDU shall be discarded. If PDU header fields are modified (for example, due to the operation of the lifetime function), then the checksum shall be modified so that the checksum remains valid.

The use of the header error detection function is optional and is selected by the originating Network entity. If the function is not used, the checksum field of the PDU header shall be set to zero.

If the function is selected by the originating Network entity, the value of the checksum field is calculated so as to cause the following formulae to be satisfied:

$$\sum_{i=1}^{L} a_{\mathrm{i}} \pmod{255} = 0$$

$$\sum_{i=1}^{L} (L - i + 1)\, a_{\mathrm{i}} \pmod{255} = 0$$

in which L is the number of octets in the PDU header, and a_i is the value of the octet at position i. The first octet in the PDU header is considered to occupy position $i = 1$.

When the function is in use, neither octet of the checksum field may be set to zero.

To ensure that inadvertent modification of a header while a PDU is being processed by an intermediate system (for example, due to a memory fault) may still be detected by the PDU header error detection function, an intermediate system Network entity shall not recompute the checksum for the entire header, even if fields are modified.

NOTE — Annex C contains descriptions of algorithms which may be used to calculate the correct value of the checksum field when the PDU is created, and to update the value of the checksum field when the header is modified.

6.12 Padding function

The padding function is provided to allow space to be reserved in the PDU header which is not used to support any other function. Octet alignment shall be maintained.

NOTE — An example of the use of this function is to cause the data part of a PDU to begin on a convenient boundary, such as a computer word boundary.

6.13 Security function

The provision of protection services (e.g., data origin authentication, data confidentiality, and data integrity of a single connectionless-mode NSDU) is performed by the security function.

The security function is related to the protection from unauthorized access quality of service parameter described in CCITT Rec. X.213 | ISO/IEC 8348. The function is realized through the selection of the security parameter in the options part of the PDU header.

This Recommendation | International Standard does not specify the way in which protection services are to be provided; it provides only for the encoding of security information in the PDU header. To facilitate interoperation among end systems and intermediate systems by avoiding different interpretations of the same encoding, a means to distinguish user-defined security encodings from standardized security encodings is described in 7.5.3.

NOTE — As an implementation consideration, data origin authentication may be provided through the use of a cryptographically generated or enciphered checksum (distinct from the PDU header error detection mechanism); data confidentiality and data integrity may be provided via route control mechanisms.

6.14 Source routeing function

The source routeing function allows a Network entity to specify the path that a generated PDU shall take. Source routeing may be selected only by the originator of a PDU. Source routeing is accomplished using a list of Network entity titles held in a parameter within the options part of the PDU header. The length of this parameter is determined by the originating Network entity, and does not change during the lifetime of a PDU. Only the titles of intermediate system Network entities shall be included in the list; the Network entity titles of the source and destination of the PDU shall not be included in the list.

Associated with the list of Network entity titles is an indicator that identifies the next entry in the list to be used; this indicator is advanced by the receiver of a PDU when the next title in the list matches its own. The indicator is updated as the PDU is forwarded so as to identify the appropriate next entry at each point along the route.

Two forms of the source routeing function are provided. The first form, referred to as complete source routeing, requires that the specified path shall be taken; that is, only those systems identified in the list may be visited by the PDU while *en route* to the destination, and each system shall be visited in the order specified. If the specified path cannot be taken, the PDU shall be discarded. 6.10 describes the circumstances in which an attempt shall be made to inform the PDU's originator of the discard using the error reporting function.

The second form is referred to as partial source routeing. As with complete source routeing, each system identified in the list shall be visited in the order specified while *en route* to the destination. However, with this form of source routeing the PDU may take any path necessary to arrive at the next intermediate system in the list, which may include visiting intermediate systems that are not identified in the list. The PDU shall not be discarded (for source routeing related reasons) unless one of the systems specified cannot be reached by any available route.

6.15 Record route function

The record route function records the path taken by a PDU as it traverses a series of intermediate systems. A recorded route consists of a list of Network entity titles held in a parameter within the options part of the PDU header. The length of this parameter is determined by the originating Network entity, and does not change during the lifetime of the PDU.

The list is constructed as the PDU is forwarded along a path towards its destination. Only the titles of intermediate system Network entities shall be included in the recorded route. The Network entity title of the originator of the PDU shall not be recorded in the list.

When an intermediate system Network entity processes a PDU containing the record route parameter, the Network entity adds its own Network entity title at the end of the list of recorded Network entity titles. An indicator is maintained to identify the next available octet to be used for recording of route. This indicator is updated as entries are added to the list as follows. The length of the entry to be added to the list is added to the value of the next available octet indicator, and this sum is compared with the length of the record route parameter. If the addition of the entry to the list would exceed the size of the parameter, the next available octet indicator is set to indicate that route recording has been terminated. The Network entity title is not added to the list. The PDU may still be forwarded to its final destination, without further addition of Network entity titles.

If the addition of the entry would not exceed the size of the record route parameter, the next available octet indicator is updated with the new value, and the Network entity title is added to the end of the list.

Two forms of the record route function are provided. The first form is referred to as complete route recording. It requires that the list of Network entity titles be a complete and accurate record of all intermediate systems visited by a PDU (including Derived PDUs), except when a shortage of space in the record route option field causes termination of recording of route, as described above. When complete route recording is selected, PDU reassembly at intermediate systems may be performed only when the Derived PDUs that are reassembled all took the same route.

The second form is referred to as partial route recording. It also requires a record of intermediate systems visited by a PDU. When partial route recording is selected, PDU reassembly at intermediate systems may be performed whether or not the Derived PDUs that are reassembled all took the same route; the route recorded in any of the Derived PDUs may be placed in the PDU resulting from the reassembly.

NOTE — The record route function is intended to be used in the diagnosis of subnetwork problems and/or to provide a return path that could be used as a source route in a subsequent PDU.

6.16 Quality of service maintenance function

The quality of service maintenance function provides information to Network entities in intermediate systems which may be used to make routeing decisions where such decisions affect the overall QoS provided to NS users. This information is conveyed to intermediate system Network entities in a parameter in the options part of the PDU header.

In those instances in which the QoS requested cannot be maintained, intermediate system Network entities shall attempt to deliver the PDU at a QoS different from the QoS requested. Intermediate system Network entities may, but need not, provide a notification of failure to meet the requested quality of service.

6.17 Priority function

The priority function allows a PDU to be processed preferentially with respect to other PDUs. The function is realized through the selection of a parameter in the options part of the PDU header.

The lowest priority value is zero; numerically greater values signify successively higher priority. The priority function provides a means whereby the resources of end and intermediate system Network entities, such as outgoing transmission queues and buffers, can be used preferentially to process higher-priority PDUs ahead of lower-priority PDUs. The specific action taken by an individual Network entity to support the priority function is a local matter.

6.18 Congestion notification function

To allow NS users to take appropriate action when congestion is experienced within the NS provider, intermediate systems may inform the destination Network entity of congestion through the use of a flag in the QoS maintenance parameter in the options part of the PDU header. The value of this flag is initially set to zero (0) by the originator of the PDU and may be set to one (1) by any intermediate system which processes the PDU to indicate that it is experiencing congestion. The criteria for determining when this action is to be taken are a local matter.

NOTE — Congestion typically corresponds to the unavailability of buffer space to maintain output queues. An appropriate policy for indicating congestion may be based upon the depth of the output queue selected for a PDU (according to its destination address or other routeing information). When the depth of a particular output queue exceeds a certain proportion of the maximum depth of that queue, an intermediate system may start to discard PDUs. The intermediate system may then set the congestion experienced flag in the next PDU to be forwarded and may continue to do so until the congestion is alleviated.

6.19 Echo request function

This function is invoked by Network layer management to obtain information about the dynamic state of the Network layer with respect to (a) the reachability of specific Network entities, and (b) the characteristics of the path or paths that can be created between Network entitites through the operation of Network layer routeing functions.

When invoked, the echo request function causes an Echo Request (ERQ) PDU to be created. The ERQ PDU shall be constructed and processed by Network entities in end systems and in intermediate systems that support the echo request function in exactly the same way as the DT PDU, with the following exceptions:

a) Since the echo request function is invoked by Network layer management, rather than by a N-UNITDATA request, the information available to the PDU composition function (see 6.1) consists of current state, local information, and information supplied by Network layer management; the references in 6.1 to information obtained from parameters of the N-UNITDATA request do not apply to the composition of an ERQ PDU.

b) The source address field of the ERQ PDU shall contain either the Network entity title of the originating Network entity or the NSAP address of an NSAP within the originating end or intermediate system. The destination address field of the ERQ PDU shall contain either the Network entity title of the destination Network entity or the NSAP address of an NSAP within the destination end or intermediate system.

NOTE 1 — A Network entity title is syntactically indistinguishable from an NSAP address. The additional information in an NSAP address, if any, beyond that which is present in a Network entity title, is relevant only to the operation of the PDU decomposition function in a destination end system, and therefore is not needed for the processing of an ERQ PDU (from which no N-UNITDATA indication is ever produced). The fact that the source and destination address fields of the ERQ PDU may contain either NETs or NSAP addresses therefore does not affect the processing of an ERQ PDU by any Network entity, and the information that is obtained through the operation of the echo request function is the same in either case.

c) When an ERQ PDU has reached its destination, as determined by the header format analysis function, the echo response function (see 6.20), rather than the PDU decomposition function, shall be invoked. It is a local matter whether or not this involves an interaction with Network layer management.

NOTE 2 — Since the echo response function is a Type 2 function (see 6.21), the destination Network entity may or may not perform the echo response function upon receiving an ERQ PDU. Network layer management must therefore consider, when the echo request function is invoked, that non-receipt of a corresponding echo response PDU may be due to non-support of the echo response function by the destination Network entity.

d) The maximum length of the ERQ PDU is equal to the maximum length of the Echo Response PDU minus the maximum length of the Echo Response PDU header. This ensures that the entire ERQ PDU can be contained within the data field of the Echo Response PDU (see 6.20).

e) The data part of the ERQ PDU may, as a local matter, contain zero or more octets with any values (subject to the overall maximum length of the ERQ PDU specified in (d) above). If the first octet of the data part contains the binary value 1000 0001 (the NLPID for this protocol), then the first *n* octets of the data part (where *n* is the value of the second octet of the data part) shall contain an entire Echo Response PDU header, in which every field in the fixed part and address part, except the segment length and checksum fields, shall contain a valid value. The more segments flag shall have the value zero. If and only if the segmentation permitted flag is set to 1, the segmentation part shall be present. The options part, if present, may contain any of the options described in 7.5.

NOTE 3 — This Echo Response PDU header, if present in the data part of an ERQ PDU, may be, but is not required to be, used in whole or in part by the destination Network entity to compose an Echo Response PDU (see 6.20 (d)). If this information is *not* present in the data part of the ERQ PDU, it may not be possible for the echo response function of the destination Network entity to select an appropriate value for the lifetime field of the Echo Response PDU.

6.20 Echo response function

This function is performed by a Network entity when it has received an ERQ PDU that has reached its destination, as determined by the header format analysis function — that is, an ERQ PDU that contains, in its destination address field, a Network entity title or NSAP address that identifies the Network entity or any of its NSAPs.

When invoked, the echo response function causes an Echo Response (ERP) PDU to be created. The ERP PDU shall be constructed and processed by Network entities in end systems and in intermediate systems that support the echo response function in exactly the same way as the DT PDU, with the following exceptions:

a) Since the echo response function is not invoked by a N-UNITDATA request, the information available to the PDU composition function consists of current state, local information, and information contained in the corresponding ERQ PDU; the references in 6.1 to information obtained from parameters of the N-UNITDATA request do not apply to the composition of an ERP PDU.

b) The source address field of the ERP PDU shall contain the value of the destination address field of the corresponding ERQ PDU. The destination address field of the ERP PDU shall contain the value of the source address field of the corresponding ERQ PDU.

   NOTE: The observation contained in NOTE 1 of 6.19 applies also to the ERP PDU.

c) The ERQ PDU, in its entirety, shall be placed into the data part of the ERP PDU. The data part of the ERP PDU shall contain *only* the corresponding ERQ PDU.

d) If the data part of the ERQ PDU contains an ERP PDU header (see 6.19 (e)), the PDU composition function may, but is not required to, use some or all of the information contained therein to select values for the fields of the ERP PDU header. In this case, however, the value of the lifetime field contained in the ERP PDU header in the ERQ PDU data part shall be used as the value of the lifetime field in the ERP PDU. The values of the segment length and checksum fields shall be computed by the Network entity regardless of the contents of those fields in the ERP PDU header in the data part of the ERQ PDU.

e) The options part of the ERP PDU may contain any (or none) of the options described in 7.5. The values for these options, if present, are determined by the Network entity as a local matter. They may be, but are not required to be, either identical to or derived from the corresponding options in the ERQ PDU and/or the ERP PDU header contained in the data part of the ERQ PDU (if present). The source routeing option in the ERP PDU shall not be identical to (copied from) the source routeing option in the ERQ PDU header. If the recording of route option in the ERP PDU is identical to (copied from) the recording of route option in the ERQ PDU header, the second octet of the parameter value field shall be set to the value 3.

f) It is a local matter whether or not the destination Network entity performs the lifetime control function on an ERQ PDU before performing the echo response function. The destination Network entity shall make the same decision in this regard that it would make, as a local matter, for a DT PDU in accordance with 6.4.

6.21 Classification of functions

Implementations of this Recommendation | International Standard are not required to support all of the functions described in 6.1 through 6.20. Functions are divided into three categories:

Type 1: These functions shall be supported.

Type 2: These functions may or may not be supported. If an implementation does not support a Type 2 function and the function is selected in a PDU, then that PDU shall be discarded, and an Error Report PDU shall be generated and forwarded to the originating Network entity, providing that the error report flag is set and the conditions of 6.10.4 are satisfied.

Type 3: These functions may or may not be supported. If an implementation does not support a Type 3 function and the function is selected in a PDU, then the function is not performed, and the PDU is processed exactly as though the function had not been selected. The PDU shall not be discarded for this reason.

Table 3 shows how the functions are divided into these three categories.

Function	Full Protocol	Non-Segmenting Subset	Inactive Subset
PDU Composition	1	1	1
PDU Decomposition	1	1	1
Header Format Analysis	1	1	1
PDU Lifetime Control	1	1	N/A
Route PDU	1	1	N/A
Forward PDU	1	1	N/A
Segment PDU	1	N/A	N/A
Reassemble PDU	1	N/A	N/A
Discard PDU	1	1	N/A
Error Reporting	1	1	N/A
Header Error Detection	1	1	N/A
Security	2	2	N/A
Complete Source Routeing	2	2	N/A
Complete Route Recording	2	2	N/A
Echo request	2	2	N/A
Echo response	2	2	N/A
Partial Source Routeing	3	3	N/A
Partial Route Recording	3	3	N/A
Priority	3	3	N/A
QoS Maintenance	3	3	N/A
Congestion Notification	3	3	N/A
Padding	3	3	N/A

Table 3 — Categorization of protocol functions

NOTES

1 While the error reporting and header error detection functions shall be provided, they are invoked only when selected by the originating Network entity.

2 The rationale for the definition of Type 3 functions is that in the case of some functions it is more important to forward the PDUs between intermediate systems or deliver them to an end system than it is to support the functions. Type 3 functions should be used in those cases in which they are of an advisory nature; they cannot cause a PDU to be discarded when they are not supported.

7 Structure and encoding of PDUs

7.1 Structure

All protocol data units shall contain an integral number of octets. The octets in a PDU are numbered starting from one (1) and increasing in the order in which they are submitted to the underlying service. The bits in an octet are numbered from one (1) to eight (8), where bit one (1) is the low-order (least significant) bit.

When consecutive octets are used to represent a binary number, the lower-numbered octet has the most significant value.

Any implementation supporting this protocol is required to state in its specification the way in which octets are transferred, using the terms "most significant bit" and "least significant bit". The PDUs of this protocol are defined using the terms "most significant bit" and "least significant bit".

NOTE — When the encoding of a PDU is represented using a diagram in this clause, the following representation is used:

a) octets are shown with the lowest-numbered octet to the left, higher-numbered octets being further to the right; and

b) within an octet, bits are shown with bit eight (8) to the left and bit one (1) to the right.

With the exception of the inactive Network layer subset, PDUs shall contain, in the following order:

a) the fixed part;

b) the address part;

c) the segmentation part, if present;

d) the options part, if present;

e) the reason for discard parameter (ER PDU only); and

f) the data part, if present.

Items (a) through (e) comprise the PDU header.

In the case of the inactive Network layer subset, only the elements identified in 7.8 are present. 7.2 through 7.5 do not apply to the inactive Network layer subset.

The structure is illustrated in Figure 2. For the purposes of Figure 2 and 7.5, the reason for discard parameter contained in the ER PDU is considered to be the final element of the options part.

Part	**Described in**
Fixed Part	Clause 7.2
Address Part	Clause 7.3
Segmentation Part	Clause 7.4
Options Part	Clause 7.5
Data	Clause 7.6

Figure 2 — PDU structure

7.2 Fixed part

7.2.1 General

The fixed part has the format illustrated in Figure 3.

				Octet
Network Layer Protocol Identifier				1
Length Indicator				2
Version/Protocol Id Extension				3
Lifetime				4
SP	MS	E/R	Type	5
Segment Length				6,7
Checksum				8,9

Figure 3 — PDU header — Fixed part

7.2.2 Network layer protocol identifier

The value of this field is set to binary 1000 0001 to identify this Network layer protocol. The value of this field is set to binary 0000 0000 to identify the inactive Network layer protocol subset.

7.2.3 Length indicator

The length is indicated by a binary number, with a maximum value of 254 (1111 1110). The length indicated is the length in octets of the header, as described in 7.1. The value 255 (1111 1111) is reserved for possible future extensions.

NOTE — The rules for forwarding and segmentation guarantee that the header length is the same for all segments (Derived PDUs) of an Initial PDU, and is the same as the header length of the Initial PDU. The size of a PDU header therefore will not change due to the operation of any protocol function.

7.2.4 Version/protocol identifier extension

The value of this field is binary 0000 0001, which identifies the standard version 1 of this protocol.

7.2.5 PDU lifetime

The PDU lifetime field is encoded as a binary number representing the remaining lifetime of the PDU, in units of 500ms.

7.2.6 Flags

7.2.6.1 Segmentation permitted

The segmentation permitted flag indicates whether segmentation is permitted. Its value is determined by the originator of the PDU and cannot be changed by any other Network entity for the lifetime of the Initial PDU and any Derived PDUs.

A value of one (1) indicates that segmentation is permitted. A value of zero (0) indicates that segmentation is not permitted. When the value of zero is selected, the segmentation part of the PDU header is not present, and the value of the segment length field gives the total length of the PDU (see 7.2.8 and 7.4.3).

7.2.6.2 More segments

The more segments flag indicates whether or not the data part of this PDU contains (as its last octet) the last octet of the user data in the NSDU. When the more segments flag is set to one (1), segmentation has occurred and the last octet of the NSDU is not contained in this PDU. The more segments flag shall not be set to one (1) if the segmentation permitted flag is not set to one (1).

When the more segments flag is set to zero (0), the last octet of the data part of the PDU is the last octet of the NSDU.

7.2.6.3 Error report

When the error report flag is set to one (1), the rules in 6.10 are used to determine whether or not to generate an Error Report PDU if it is necessary to discard this PDU.

When the error report flag is set to zero (0), discard of the PDU will not cause the generation of an Error Report PDU.

7.2.7 Type code

The type code field identifies the type of the protocol data unit. Allowed values are given in Table 4.

PDU Type	Type Code					
	Bits	5	4	3	2	1
DT PDU		1	1	1	0	0
ER PDU		0	0	0	0	1
ERQ PDU		1	1	1	1	0
ERP PDU		1	1	1	1	1

Table 4 — PDU type codes

7.2.8 PDU segment length

The segment length field specifies the entire length of the PDU in octets, including both header and data (if present). When the full protocol is employed and a PDU is not segmented, the value of this field is identical to the value of the total length field located in the segmentation part of the header.

When the non-segmenting protocol subset is employed, no segmentation part is present in the header. In this case, the segment length field specifies the entire length of the Initial PDU, including both header and data (if present).

The value of the segment length field shall not be changed for the lifetime of the PDU.

7.2.9 PDU checksum

The checksum is computed on the entire PDU header. For the Data, Echo Request, and Echo Reply PDUs, this includes the segmentation and options parts (if present). For the Error Report PDU, this includes the reason for discard field as well.

A checksum value of zero (0) is reserved to indicate that the checksum is to be ignored. The operation of the PDU header error detection function (see 6.11) ensures that the value zero does not represent a valid checksum. A non-zero value indicates that the checksum shall be processed; if the checksum calculation fails, the PDU shall be discarded.

7.3 Address part

7.3.1 General

The address part immediately follows the fixed part of the PDU header. The address part is illustrated in Figure 4.

	Octet
Destination Address Length Indicator	10
Destination Address	11 $m - 1$
Source Address Length Indicator	m
Source Address	$m + 1$ $n - 1$

Figure 4 — PDU header — Address part

7.3.2 Destination and source addresses

The destination and source addresses used by this protocol are Network service access point addresses or Network entity titles as defined in CCITT Rec. X.213 | ISO/IEC 8348.

The destination and source addresses are of variable length. The destination and source addresses are encoded as Network protocol address information in the destination address and source address fields using the "preferred encoding" defined in CCITT Rec. X.213 | ISO/IEC 8348.

The destination address length indicator field specifies the length of the destination address in octets. The destination address field follows the destination address length indicator field.

The source address length indicator field specifies the length of the source address in octets. The source address length indicator field follows the destination address field. The source address field follows the source address length indicator field.

Each address parameter is encoded as illustrated in Figure 5.

Octet n	Address parameter Length Indicator (e.g., ‘m’)
Octets $n + 1$ to $n + m$	Address Parameter Value

Figure 5 — Address parameters

7.4 Segmentation part

7.4.1 General

If the segmentation permitted flag in the fixed part of the PDU header (see 7.2.6.1) is set to one (1), the segmentation part of the header, illustrated in Figure 6, shall be present.

If the segmentation permitted flag is set to zero (0), the segmentation part shall not be present (the non-segmenting protocol subset is in use).

	Octet
Data Unit Identifier	n, $n + 1$
Segment Offset	$n + 2$, $n + 3$
Total Length	$n + 4$, $n + 5$

Figure 6 — PDU header — Segmentation part

7.4.2 Data unit identifier

The data unit identifier identifies an Initial PDU (and hence, its Derived PDUs) so that a segmented data unit may be correctly reassembled. The data unit identifier size is two (2) octets.

7.4.3 Segment offset

For each Derived PDU, the segment offset field specifies the relative position of the segment contained in the data part of the Derived PDU with respect to the start of the data part of the Initial PDU. The offset is measured in units of octets. The offset of the first segment (and hence, the Initial PDU) is zero (0); an unsegmented (Initial) PDU has a segment offset value of zero (0). The value of this field shall be a multiple of eight (8).

7.4.3 PDU total length

The total length field specifies the entire length of the Initial PDU in octets, including both the header and data. The value of this field shall not be changed for the lifetime of the Initial PDU (and hence, its Derived PDUs).

7.5 Options part

7.5.1 General

The options part of the PDU header is illustrated in Figure 7.

	Octet
Options	$n + 6$ p

Figure 7 — PDU header — Options part

If the options part is present, it may contain one or more parameters. The number of parameters that may be contained in the options part is constrained by the length of the options part, which is determined by the following formula:

Options part length = PDU header length – (length of fixed part + length of address part + length of segmentation part)

and by the length of the individual optional parameters.

Parameters defined in the options part may appear in any order. Duplication of options is not permitted. Receipt of a PDU with a duplicated option shall be treated as a protocol error. The rules governing the treatment of protocol errors are described in 6.10.

The encoding of parameters contained within the options part of the PDU header is illustrated in Figure 8.

Octets	
n	Parameter Code
$n + 1$	Parameter Length (e.g., 'm')
$n + 2$ to $n + m + 1$	Parameter Value

Figure 8 — Encoding of option parameters

The parameter code field is encoded in binary and provides for a maximum of 255 different parameters. No parameter code uses bits 8 and 7 with the value 00, so the actual maximum number of parameters is lower. A parameter code of 255 (binary 1111 1111) is reserved for possible future extensions.

The parameter length field indicates the length, in octets, of the parameter value field. The length is indicated by a positive binary number, m, with a minimum value of 1 and a theoretical maximum value of 254. The practical maximum value of m is lower. For example, in the case of a single parameter contained within the options part, two octets are required for the parameter code and the parameter length indicators. Thus, the value of m is limited to:

$$m = 252 - (\text{length of fixed part} + \text{length of address part} + \text{length of segmentation part})$$

Accordingly, for each successive parameter the maximum value of m decreases.

The parameter value field contains the value of the parameter identified in the parameter code field.

The following parameters are permitted in the options part.

7.5.2 Padding

The padding parameter is used to lengthen the PDU header to a convenient size (see 6.12).

Parameter Code: 1100 1100
Parameter Length: variable
Parameter Value: any value is allowed.

Notwithstanding the requirement stated in 7.5.1 that the value of the parameter length field be no less than 1, the receiver of a PDU containing a value of 0 for the parameter length field of the padding option (and containing, therefore, no parameter value field for the padding option) may, but is not required to, treat this as a protocol error.

7.5.3 Security

This parameter allows a unique and unambiguous security level to be assigned to a protocol data unit (see 6.13).

Parameter Code: 1100 0101
Parameter Length: variable
Parameter Value: The high order two bits of the first octet specify the security format code, as shown in Table 5.

Security Format Code	Type of Security Field
00	Reserved
01	Source Address Specific
10	Destination Address Specific
11	Globally Unique

Table 5 — Security format codes

The rest of the first octet is reserved and shall be set to zero (0) when transmitted and ignored when received. The remainder of the parameter value field specifies the security level as described in the following clauses.

7.5.3.1 Source address specific

The security format code value of binary 01 indicates that the remaining octets of the parameter value field specify a security level which is unique and unambiguous in the context of the security classification system employed by the authority responsible for assigning the source NSAP address.

7.5.3.2 Destination address specific

The security format code value of binary 10 indicates that the remaining octets of the parameter value field specify a security level which is unique and unambiguous in the context of the security classification system employed by the authority responsible for assigning the destination NSAP address.

7.5.3.3 Globally unique security

The security format code value of binary 11 indicates that the remaining octets of the parameter value field specify a globally unique and unambiguous security level. This security classification system is not specified by this Recommendation | International Standard.

7.5.4 Source routeing

The source routeing parameter specifies, either completely or partially, the route to be taken from sthe originating Network entity to the destination Network entity (see 6.14).

Parameter Code: 1100 1000
Parameter Length: variable
Parameter Value: 2 octets of control information followed by a concatenation of Network entity title entries ordered from source to destination.

The first octet of the parameter value is the type code, which has the following significance:

0000 0000 partial source routeing
0000 0001 complete source routeing
<all other values reserved>

The second octet indicates the octet offset of the next Network entity title entry to be processed in the list. It is relative to the start of the parameter, such that a value of three (3) indicates that the next Network entity title entry begins immediately after this control octet. Successive octets are indicated by correspondingly larger values of this indicator.

The third octet begins the Network entity title list. The list consists of variable length Network entity title entries. The first octet of each entry gives the length of the Network entity title that comprises the remainder of the entry.

7.5.5 Recording of route

The recording of route parameter identifies the intermediate systems traversed by the PDU (see 6.15).

Parameter Code: 1100 1011
Parameter Length: variable
Parameter Value: 2 octets of control information followed by a concatenation of Network entity title entries ordered from source to destination.

The first octet of the parameter value is the type code, which has the following significance:

0000 0000 partial recording of route in progress
0000 0001 complete recording of route in progress
<all other values reserved>

The second octet identifies the first octet not currently used for a recorded Network entity title, and therefore also the current end of the list. It is encoded relative to the start of the parameter value, such that a value of three (3) indicates that no Network entity titles have yet been recorded. The value of 255 is used to indicate that route recording has been terminated.

The third octet begins the Network entity title list. The list consists of variable length Network entity title entries. The first octet of each entry gives the length of the Network entity title comprising the remainder of the entry. Network entity title entries are always added to the end of the list.

NOTE — The length of the record route parameter is determined by the originator of the PDU and is not changed during the lifetime of the PDU; hence, the operation of the record route function does not affect the length of the header.

7.5.6 Quality of service maintenance

The quality of service maintenance parameter conveys information about the quality of service requested by the originating NS user.

Network entities in intermediate systems may, but are not required to, make use of this information as an aid in selecting a route when more than one route satisfying other routeing criteria is available and the available routes are known to differ with respect to quality of service (see 6.16).

Parameter Code: 1100 0011
Parameter Length: variable
Parameter Value: The high order two bits of the first octet specify the QoS format code, as shown in Table 6.

QoS Format Code	Type of QoS Field
00	Reserved
01	Source Address Specific
10	Destination Address Specific
11	Globally Unique

Table 6 — QoS format codes

The rest of the first octet is reserved for use by the globally unique QoS format, as described in 7.5.6.3. If any other QoS format code is selected, bits 6–1 of the first octet shall be zero (0). The remainder of the parameter value field specifies the QoS as described in the following clauses.

7.5.6.1 Source address specific

The QoS format code value of binary 01 indicates that the remaining octets of the parameter value field specify a QoS which is unique and unambiguous in the context of the QoS maintenance system employed by the authority responsible for assigning the source NSAP address.

7.5.6.2 Destination address specific

The QoS format code value of binary 10 indicates that the remaining octets of the parameter value field specify a QoS which is unique and unambiguous in the context of the QoS maintenance system employed by the authority responsible for assigning the destination NSAP address.

7.5.6.3 Globally unique QoS

The QoS format code value of binary 11 indicates that the remainder of the parameter value field specifies a globally unique QoS maintenance field. When the globally unique QoS maintenance function is employed, the parameter value field shall have a total length of one octet, which is assigned the values shown in Table 7.

Bits	Usage
8 and 7	QoS format code of binary 11
6	reserved
5	sequencing vs. transit delay
4	congestion experienced
3	transit delay vs. cost
2	residual error probability vs. transit delay
1	residual error probability vs. cost

Table 7 — Globally unique QoS parameter values

Bit 6 is reserved, and shall be set to zero (0) when transmitted and ignored when received.

Bit 5 is set to one to indicate that, where possible, routeing decisions should favor sending all PDUs to the specified destination NSAP address over a single path (in order to maintain sequence) over minimizing transit delay. A value of zero (0) indicates that, where possible, routeing decisions should favor low transit delay over sequence preservation.

Bit 4 is set to zero by the Network entity which originates the protocol data unit. It is set to one by an intermediate system to indicate that this PDU has visited a congested intermediate system, and appropriate action should be taken by the destination Network entity. Once the congestion experienced bit is set by an intermediate system, it may not be reset by any intermediate system traversed by the PDU further along the path towards the destination.

Bit 3 is set to one to indicate that, where possible, routeing decisions should favor low transit delay over low cost. A value of 0 indicates that routeing decisions should favor low cost over low transit delay.

Bit 2 is set to one to indicate that, where possible, routeing decisions should favor low residual error probability over low transit delay. A value of zero indicates that routeing decisions should favor low transit delay over low residual error probability.

Bit 1 is set to one to indicate that, where possible, routeing decisions should favor low residual error probability over low cost. A value of 0 indicates that routeing decisions should favor low cost over low residual error probability.

7.5.7 Priority

The value of the priority parameter indicates the relative priority of the protocol data unit. Intermediate systems that support this option shall make use of this information in routeing and in ordering PDUs for transmission (see 6.17).

Parameter Code: 1100 1101
Parameter Length: 1 octet
Parameter Value: 0000 0000 — Normal (Default)
through
0000 1110 — Highest
<all other values reserved>

The values 0000 0001 through 0000 1110 are to be used for higher priority protocol data units. If an intermediate system does not support this option, all PDUs shall be treated as if the field had the value 0000 0000.

7.6 Data part

The data part of the PDU header is illustrated in Figure 9.

	Octet
Data	$p + 1$ … z

Figure 9 — PDU header — Data part

7.7 Data PDU

7.7.1 Structure

The Data PDU has the format illustrated in Figure 10.

	Octet
Network Layer Protocol Identifier	1
Length Indicator	2
Version/Protocol Id Extension	3
Lifetime	4
SP | MS | E/R | Type	5
Segment Length	6,7
Checksum	8,9
Destination Address Length Indicator	10
Destination Address	11 … $m - 1$
Source Address Length Indicator	m
Source Address	$m + 1$ … $n - 1$
Data Unit Identifier	$n, n + 1$
Segment Offset	$n + 2, n + 3$
Total Length	$n + 4, n + 5$
Options	$n + 6$ … p
Data	$p + 1$ … z

Figure 10 — Data PDU

7.7.2 Fixed part

1) Network Layer Protocol Identifier See 7.2.2
2) Length Indicator See 7.2.3
3) Version/Protocol Id Extension See 7.2.4
4) Lifetime See 7.2.5

5)	SP, MS, E/R	See 7.2.6
6)	Type Code	See 7.2.7
7)	Segment Length	See 7.2.8
8)	Checksum	See 7.2.9

7.7.3 Addresses

See 7.3.

7.7.4 Segmentation

See 7.4.

7.7.5 Options

See 7.5.

7.7.6 Data

See 7.6.

7.8 Inactive Network layer protocol

7.8.1 Structure

The Inactive Network Layer Protocol PDU has the format illustrated in Figure 11.

	Octet
Network Layer Protocol Identifier	1
Data	2 2 - *n*

Figure 11 — Inactive Network Layer Protocol PDU

7.8.2 Network layer protocol identifier

The value of the Network layer protocol identifier field is binary zero (0000 0000).

7.8.3 Data part

The data part may contain any number of octets up to one less than the maximum number that can be placed in the SN-Userdata parameter of the underlying SN-UNITDATA primitive. Therefore, the inactive Network layer protocol can be used only when the length of the NS-Userdata parameter in the N-UNITDATA primitive is constrained to be less than or equal to the value of the length of the SN-Userdata parameter minus one (see 7.6).

7.9 Error Report PDU

7.9.1 Structure

The format of the Error Report PDU is illustrated in Figure 12.

Field	Octet
Network Layer Protocol Identifier	1
Length Indicator	2
Version/Protocol Id Extension	3
Lifetime	4
SP | MS | E/R | Type	5
Segment Length	6,7
Checksum	8,9
Destination Address Length Indicator	10
Destination Address	11 … $m - 1$
Source Address Length Indicator	m
Source Address	$m + 1$ … $n - 1$
Options	n … p - 1
Reason for Discard	p … q - 1
Data Part	q … z

Figure 12 — Error Report PDU

7.9.2 Fixed part

The fixed part of the Error Report PDU is composed in the same way as a new (Initial) Data PDU.

1) Network Layer Protocol Identifier See 7.2.2
2) Length Indicator See 7.2.3
3) Version/Protocol Id Extension See 7.2.4
4) Lifetime See 7.2.5
5) SP, MS, E/R *(Always set to zero)* See 6.10
6) Type Code See 7.2.7
7) Segment Length See 7.2.8
8) Checksum See 7.2.9

7.9.3 Addresses

The destination address specifies the Network entity title of the originator of the discarded PDU. The source address specifies the title of the intermediate system or end system Network entity initiating the Error Report PDU (see 7.3).

7.9.4 Options

See 7.5.

7.9.5 Reason for discard

This parameter is valid only for the Error Report PDU.

Parameter Code: 1100 0001
Parameter Length: two octets
Parameter Value: type of error encoded in binary.

The parameter values are listed in Table 8.

The first octet of the parameter value contains an error type code. If the error in the discarded PDU can be localized to a particular field, the number of the first octet of that field is stored in the second octet of the reason for discard parameter field. If the error cannot be localized to a particular field, or if the error is a checksum error, then the value zero (0) is stored in the second octet of the reason for discard parameter field.

7.9.6 Data part

This field contains the entire header of the discarded PDU, and may contain none, some, or all of the data part of the discarded PDU.

Parameter Value		Class of Error	Meaning
0000	0000	General	Reason not specified
	0001		Protocol procedure error
	0010		Incorrect checksum
	0011		PDU discarded due to congestion
	0100		Header syntax error (cannot be parsed)
	0101		Segmentation needed but not permitted
	0110		Incomplete PDU received
	0111		Duplicate option
1000	0000	Address	Destination address unreachable
	0001		Destination address unknown
1001	0000	Source Routeing	Unspecified source routeing error
	0001		Syntax error in source routeing field
	0010		Unknown address in source routeing field
	0011		Path not acceptable
1010	0000	Lifetime	Lifetime expired while data unit in transit
	0001		Lifetime expired during reassembly
1011	0000	PDU Discarded	Unsupported option not specified
	0001		Unsupported protocol version
	0010		Unsupported security option
	0011		Unsupported source routeing option
	0100		Unsupported recording of route option
1100	0000	Reassembly	Reassembly interference

Table 8 — Reason for discard parameter values

7.10 Echo Request PDU

The ERQ PDU has the same format as the DT PDU (see 7.7).

7.11 Echo Response PDU

The ERP PDU has the same format as the DT PDU (see 7.7).

8 Provision of the underlying service

Subnetwork dependent convergence functions may be performed to provide an underlying connectionless-mode service when a real subnetwork does not inherently provide the underlying connectionless-mode service assumed by the protocol. If a subnetwork inherently provides a connection-mode service, a subnetwork dependent convergence function provides a mapping into the required underlying connectionless-mode service. Subnetwork dependent convergence functions may also be required in those cases in which functions assumed from the underlying service are not performed. In some cases, this may require the operation of an explicit protocol (i.e., a protocol involving explicit exchanges of protocol control information between peer Network entities) in the subnetwork dependent convergence protocol (SNDCP) role. However, there may also be cases in which the functionality required to fulfill the SNDCP role consists simply of a set of rules for manipulating the underlying service (without the exchange of PCI between peer Network entities).

8.1 Subnetwork points of attachment

The source address and destination address parameters in the SN-UNITDATA primitive specify the points of attachment to a public or private subnetwork(s). Subnetwork point of attachment addresses (SNPAs) are defined by each individual subnetwork authority. The syntax and semantics of SNPAs are not defined by this Recommendation | International Standard.

8.2 Subnetwork quality of service

Associated with each connectionless-mode transmission, certain measures of quality of service are requested when the SN-UNITDATA primitive action is initiated. These requested measures (or parameter values and options) are based on *a priori* knowledge of the service available from the subnetwork. Knowledge of the nature and type of service available is typically obtained prior to an invocation of the underlying connectionless-mode service.

The quality of service parameters identified for the underlying connectionless-mode service may in some circumstances be directly derivable from or mappable onto those identified in the connectionless-mode Network service. The following parameters as defined in CCITT Rec. X.213 | ISO/IEC 8348 may be employed:

a) transit delay;

b) protection against unauthorized access;

c) cost determinants;

d) priority; and

e) residual error probability.

NOTE — For those real subnetworks which do not inherently provide quality of service as a parametereter, it is a local matter as to how the semantics of the service requested might be preserved. In particular, there may be instances in which the quality of service requested cannot be maintained. In such circumstances, an attempt shall be made to deliver the protocol data unit at whatever quality of service is available.

In general, either the SNDCF or the subnetwork itself may perform functions associated with specific QoS requests. These functions may be optionally selected by the CLNP. The relevant subnetwork QoS parameters are classified as follows:

a) those QoS parameters for which the SNDCF or the subnetwork itself performs functions expressly designed to provide information for the route PDU function of the CLNP;

b) those QoS parameters for which the SNDCF or the subnetwork itself performs functions expressly designed to provide the desired QoS; and

c) those QoS parameters for which the SNDCF or the subnetwork itself may be called upon to perform either of the functions (a) or (b) above.

The determination of values for these QoS parameters is provided in the following clauses.

8.2.1 Transit delay

Transit delay is the elapsed time between an SN-UNITDATA request and the corresponding SN-UNITDATA indication. Elapsed time values are calculated on SNSDUs that are successfully transmitted. Successful transmission of an SNSDU is defined to occur when an SNSDU transmitted by the sending SNDCF is delivered to the intended destination SNDCF. Transit delay is based on an SNSDU size of 512 octets, and is specified in units of 500ms.

Transit delay is determined by the SNDCF prior to the processing of any user data by the subnetwork. The mechanism whereby transit delay information is passed to the route PDU function of the CLNP is a local matter. Transit delay may be either measured or estimated. The SNDCFs described herein do not provide any means for measuring or estimating transit delay beyond any such means provided by the underlying subnetwork.

NOTES

1 If transit delay is to be measured, an SNDCP designed to bound the transit time of SNSDUs that cross the subnetwork should be used prior to the processing of any data requests to determine the actual delay.

2 Transit delay within a given subnetwork may vary. Where transit delay is measured, it may be necessary to periodically repeat the measurement process in order to maintain accurate measures in any routeing information maintained by the Network entity.

3 If no better measures are available, transit delay may be estimated by sending an SNSDU (via some uniquely identified protocol data unit which prompts a response) and by measuring the elapsed time between the SN-UNITDATA requests and the corresponding SN-UNITDATA indications. This results in an overestimate of delay such that the CLNP may be expected to operate correctly. If transit delay is estimated, it is preferred that estimates be high rather than low in order that uncertainties in transit delay do not prevent the CLNP from discarding protocol data units whose intended lifetime has expired.

8.2.2 Protection from unauthorized access

No recommendation is made concerning how to provide protection against passive monitoring, modification, replay, addition, or deletion of SN-Userdata.

8.2.3 Residual error probability

Residual error probability is estimated as the ratio of lost, duplicated, or incorrectly delivered SNSDUs to total SNSDUs transmitted by the SNDCF during a measurement period. The mechanism whereby residual error probability is passed to the route PDU function of the CLNP is a local matter.

Residual error probability is known by the SNDCF prior to the processing of any user data by the subnetwork, either as a result of the SNDCF having maintained a history of measures of residual error probability, or as a result of information obtained from the provider of the underlying service.

NOTE — For subnetworks which provide a connection-mode service, residual error probability is determined on an individual connection basis.

8.2.4 Cost determinants

This subclause is applicable only to ISO/IEC 8473.

The attempt to satisfy the constraints imposed by the NS user via the cost determinants quality of service parameter is performed by the route PDU function invoked by the CLNP. Where pertinent, information relating to tariff(s) assessed on a per packet or per connection basis is passed to the route PDU function of the CLNP. The mechanism by which this is accomplished is a local matter.

NOTE — The route PDU function invoked by the CLNP may be required to perform the following cost assessments. If:

a) there is to be no incrememtal cost incurred in the processing of the SNSDU submitted, and there is a tariff assessed on a per packet basis;

b) there is to be no additional cost incurred, and no connection is currently available to the specified destination, and a tariff is assessed on a per connection basis by the subnetwork (e.g. for virtual circuit setup, holding time of the virtual circuit, etc.); or

c) a maximum acceptable cost has been specified for the processing of the NSDU, and that cost is likely to be exceeded,

then the route PDU function should return a result indicating that the CLNP should attempt to deliver the NSDU via some alternate route. If an alternate route cannot be found, a local function may be invoked to notify the NS user of the inability of the NS provider to deliver this NSDU (and possibly subsequent NSDUs) under the stated constraint.

8.3 Subnetwork user data

The SN-Userdata is an ordered multiple of octets, and is transferred transparently between the specified subnetwork points of attachment.

The underlying service assumed by the CLNP is required to support a service data unit size of at least 512 octets.

If the minimum service data unit sizes supported by all of the subnetworks involved in the transmission of a particular PDU are known to be large enough that segmentation is not required, then either the full protocol or the non-segmenting protocol subset may be used.

Data received from a subnetwork with protocol identification specifying this protocol (see 7.2.2) shall be processed according to this Recommendation | International Standard.

NOTE — Data with other protocol identification should be ignored, since it may have been sent by an implementation supporting additional protocols intended for use with this protocol.

8.4 Subnetwork dependent convergence functions

The general model for providing the underlying service assumed by the protocol in conjunction with a real subnetwork that uses a connectionless subnetwork access protocol is as follows. The generation of an SN-UNITDATA request by the CLNP results in the generation of a corresponding subnetwork-specific UNITDATA request by the subnetwork dependent convergence function. The receipt of a subnetwork-specific UNITDATA indication associated with delivery of a connectionless data unit to its destination causes the SNDCF to generate an SN-UNITDATA indication to the CLNP.

The general model for providing the underlying service assumed by the CLNP in conjunction with a real subnetwork that uses a connection-mode subnetwork access protocol is as follows. The generation of an SN-UNITDATA request by the CLNP causes a connection (logical channel, logical link, or the equivalent) to be made available for the transmission of SN-Userdata. If a connection cannot be made available, the SN-UNITDATA request is discarded. The receipt of subnetwork-specific PDUs containing SN-Userdata causes the SNDCF to generate an SN-UNITDATA indication to the CLNP.

Where a real subnetwork is designed to use either a connectionless-mode or a connection-mode subnetwork access protocol, the provision of the underlying service assumed by the CLNP is achieved by using the connectionless-mode alternative.

The way in which the underlying service is provided by specific subnetwork types is defined in other Recommendations | parts of this International Standard.

9 Conformance

9.1 Static conformance

9.1.1 End systems

An implementation claiming conformance to this Recommendation | International Standard as an end system shall:

a) support the transmission and reception of NPDUs using the full protocol;

b) support the reception of NPDUs conveyed using the non-segmenting protocol subset;

c) support the protocol functions identified in Table 9 as mandatory for end systems; and

d) be capable of operating over one or more subnetworks, using the appropriate subnetwork dependent convergence function(s) specified in other Recommendations | parts of this International Standard.

Such an end system may (as implementation options), but is not required to:

e) support the transmission of NPDUs using the non-segmenting protocol subset;

f) support the transmission and reception of NPDUs using the inactive Network layer protocol subset; and

g) support any of the protocol functions identified in Table 9 as optional for end systems.

NOTE — Although item (a) above requires end systems to support both the transmission and the reception of NPDUs, the requirements for transmission and reception are specified separately in Table 9. In general, the procedures to be followed in order to support a given function are different for the sending and receiving senses. The separate specification (1) distinguishes between the requirements for two functions (PDU lifetime control and padding) for which support is mandatory for one sense of PDU transfer and optional for the other; and (2) clarifies the fact that support of several of the functions is applicable only for one sense of PDU transfer.

9.1.2 Intermediate systems

An implementation claiming conformance to this Recommendation | International Standard as an intermediate system shall:

a) support the protocol functions identified in Table 9 as mandatory for intermediate systems; and

b) be capable of operating over one or more subnetworks, using the appropriate subnetwork dependent convergence function(s) specified in other Recommendations | parts of this International Standard.

Such an intermediate system may (as an implementation option), but is not required to:

c) support any of the protocol functions identified in Table 9 as optional for intermediate systems.

9.2 Dynamic conformance

An implementation claiming conformance to this Recommendation | International Standard shall exhibit externally observable behaviour consistent with its having implemented:

a) each protocol function that it supports in accordance with the function's specification, as contained in the subclause referenced from Table 9; and

b) the relevant subnetwork dependent convergence function(s) in accordance with the specification contained in other Recommendations | parts of this International Standard.

All PDUs transmitted shall be structured as specified in clause 7.

An implementation that does not support a function identified in Table 9 as optional shall, upon receiving a PDU in which that function is selected, either discard the PDU and invoke the error reporting function, or process the PDU as though the function had not been selected, in accordance with the specification contained in 6.21.

9.3 PICS proforma

The supplier of a protocol implementation that claims to conform to this Recommendation | International Standard shall complete a copy of the PICS proforma provided in Annex A, including the information necessary to identify both the supplier and the implementation.

Protocol function	Reference	End system (note 1)		Intermediate system
		Sending	Receiving	
PDU Composition (note 2)	6.1	M	N/A	N/A
PDU Decomposition (note 2)	6.2	N/A	M	N/A
Header Format Analysis	6.3	N/A	M	M
PDU Lifetime Control	6.4	M	O	M
Route PDU	6.5	M	N/A	M
Forward PDU	6.6	M	N/A	M
Segmentation (note 2)	6.7	M	N/A	Note 3
Reassembly (note 2)	6.8	N/A	M	O (Note 4)
Discard PDU	6.9	N/A	M	M
Error Reporting	6.10	M	M	M
Header Error Detection	6.11	M	M	M
Security	6.13	O	O (Note 4)	O (Note 4)
Complete Source Routeing	6.14	O	N/A	O (Note 4)
Complete Route Recording	6.15	O	O (Note 4)	O (Note 4)
Echo request	6.19	O	O (Note 4)	O (Note 4)
Echo response	6.20	N/A	O (Note 4)	O (Note 4)
Partial Source Routeing	6.14	O	N/A	O (Note 4)
Partial Route Recording	6.15	O	O (Note 4)	O (Note 4)
Priority	6.17	O	O (Note 4)	O (Note 4)
QoS Maintenance	6.16	O	O (Note 4)	O (Note 4)
Congestion Notification	6.18	N/A	O (Note 4)	O (Note 4)
Padding	6.12	O	M	M

Key:
M: Mandatory function; this function shall be implemented
O: Implementation option, as described in the text
N/A: Not applicable

Table 9 — Static conformance requirements

NOTES

1 The status in the "sending" column applies to the support of the given function for DT, ER, ERQ, and ERP PDUs sent by the end system; similarly, the status in the "receiving" column applies to the support of the given function for DT, ER, ERQ, and ERP PDUs received by the end system.

2 The PDU composition, PDU decomposition, segmentation, and reassembly functions are not relevant for ER PDUs.

3 The segment PDU function is in general mandatory for an intermediate system. However, a system which is to be connected only to subnetworks that all offer the same maximum SDU size (such as identical local area networks) will not need to perform this function, and therefore does not need to implement it.

4 See 9.2 for related dynamic conformance requirements that apply when this option is not supported.

Annex A[1]

PICS proforma

(This annex forms an integral part of this Recommendation | International Standard.)

A.1 Introduction

The supplier of a protocol implementation which is claimed to conform to this Recommendation | International Standard shall complete the following Protocol Implementation Conformance Statement (PICS) proforma.

A completed PICS proforma is the PICS for the implementation in question. The PICS is a statement of which capabilities and options of the protocol have been implemented. The PICS can have a number of uses, including use

— by the protocol implementor, as a check-list to reduce the risk of failure to conform to the standard through oversight;

— by the supplier and acquirer — or potential acquirer — of the implementation, as a detailed indication of the capabilities of the implementation, stated relative to the common basis for understanding provided by the standard PICS proforma;

— by the user — or potential user — of the implementation, as a basis for initially checking the possibility of interworking with another implementation (note that, while interworking can never be guaranteed, failure to interwork can often be predicted from incompatible PICSs);

— by a protocol tester, as the basis for selecting appropriate tests against which to assess the claim for conformance of the implementation.

A.2 Abbreviations and special symbols

A.2.1 Status symbols

M	mandatory
O	optional
O.<n>	optional, but support of at least one of the group of options labelled by the same numeral <n> is required
X	prohibited
<pred>:	conditional-item symbol, including predicate identification (see A.3.4)
^	logical negation, applied to a conditional item's predicate

A.2.2 Other symbols

<r>	receive aspects of an item
<s>	send aspects of an item

---

[1] Copyright release for PICS proformas

Users of this Recommendation | International Standard may freely reproduce the PICS proforma in this Annex so that it can be used for its intended purpose and may further publish the completed PICS.

A.3 Instructions for completing the PICS proforma

A.3.1 General structure of the PICS proforma

The first part of the PICS proforma — Implementation Identification and Protocol Summary — is to be completed as indicated with the information necessary to identify fully both the supplier and the implementation.

The main part of the PICS proforma is a fixed-format questionnaire divided into a number of major subclauses; these can be divided into further subclauses each containing a group of individual items. Answers to the questionnaire items are to be provided in the rightmost column, either by simply marking an answer to indicate a restricted choice (usually Yes or No), or by entering a value or a set or range of values.

NOTE — There are some items for which two or more choices from a set of possible answers can apply. All relevant choices are to be marked in these cases.

Each item is identified by an item reference in the first column; the second column contains the question to be answered; and the third column contains the reference or references to the material that specifies the item in the main body of this Recommendation | International Standard. The remaining columns record the status of the item — whether support is mandatory, optional, prohibited, or conditional — and provide space for the answers (see also A.3.4).

A supplier may also provide further information, categorized as either Additional Information or Exception Information. When present, each kind of further information is to be provided in a further subclause of items labelled A<i> or X<i>, respectively, for cross-referencing purposes, where <i> is any unambiguous identification for the item (e.g., a number); there are no other restrictions on its format or presentation.

A completed PICS proforma, including any Additional Information and Exception Information, is the Protocol Implementation Conformance Statement for the implementation in question.

NOTE — Where an implementation is capable of being configured in more than one way, a single PICS may be able to describe all such configurations. However, the supplier has the choice of providing more than one PICS, each covering some subset of the implementation's configuration capabilities, in cases where this makes for easier and clearer presentation of the information.

A.3.2 Additional information

Items of Additional Information allow a supplier to provide further information intended to assist in the interpretation of the PICS. It is not intended or expected that a large quantity will be supplied, and a PICS can be considered complete without any such information. Examples might be an outline of the ways in which a (single) implementation can be set up to operate in a variety of environments and configurations, or a brief rationale — based perhaps upon specific application needs — for the exclusion of features which, although optional, are nonetheless commonly present in implementations of this protocol.

References to items of Additional Information may be entered next to any answer in the questionnaire, and may be included in items of Exception Information.

A.3.3 Exception information

It may occasionally happen that a supplier will wish to answer an item with mandatory or prohibited status (after any conditions have been applied) in a way that conflicts with the indicated requirement. No pre-printed answer will be found in the support column for this; instead, the supplier shall write the missing answer into the Support column, together with an X<i> reference to an item of Exception Information, and shall provide the appropriate rationale in the Exception Information item itself.

An implementation for which an Exception Information item is required in this way does not conform to this Recommendation | International Standard.

NOTE — A possible reason for the situation described above is that a defect in the standard has been reported, a correction for which is expected to change the requirement not met by the implementation.

A.3.4 Conditional status

A.3.4.1 Conditional items

The PICS proforma contains a number of conditional items. These are items for which the status — mandatory, optional, or prohibited — that applies is dependent upon whether or not certain other items are supported, or upon the values supported for other items.

In many cases, whether or not the item applies at all is conditional in this way, as well as the status when the item does apply.

Where a group of items is subject to the same condition for applicability, a separate preliminary question about the condition appears at the head of the group, with an instruction to skip to a later point in the questionnaire if the "Not Applicable" answer is selected. Otherwise, individual conditional items are indicated by one or more conditional symbols (on separate lines) in the status column.

A conditional symbol is of the form "<pred>:<x>" where "<pred>" is a predicate as described in A.3.4.2, and "<x>" is one of the status symbols M, O, O.<n>, or X.

If the value of the predicate in any line of a conditional item is true (see A.3.4.2), then the conditional item is applicable, and its status is that indicated by the status symbol following the predicate; the answer column is to be marked in the usual way. If the value of a predicate is false, the Not Applicable (N/A) answer is to be marked in the relevant line. Each line in a multi-line conditional item should be marked: at most one line will require an answer other than N/A.

A.3.4.2 Predicates

A predicate is one of the following:

a) an item-reference for an item in the PICS proforma: the value of the predicate is true if the item is marked as supported, and is false otherwise;

b) a predicate name, for a predicate defined elsewhere in the PICS proforma (usually in the Major Capabilities section or at the end of the section containing the conditional item): see below; or

c) the logical negation symbol "^" prefixed to an item-reference or predicate name: the value of the predicate is true if the value of the predicate formed by omitting the "^" is false, and vice versa.

The definition for a predicate name is one of the following

a) an item-reference, evaluated as at (a) above;

b) a relation containing a comparison operator (=, < , etc.) with at least one of its operands being an item-reference for an item taking numerical values as its answer; the predicate is true if the relation holds when each item-reference is replaced by the value entered in the Support column as an answer to the item referred to; or

c) a boolean expression constructed by combining simple predicates, as in (a) and (b), using the boolean operators AND, OR, and NOT, and parentheses, in the usual way; the value of such a predicate is true if the boolean expression evaluates to true when the simple predicates are interpreted as described above.

Each item whose reference is used in a predicate or predicate definition is indicated by an asterisk in the Item column.

A.4 Identification

A.4.1 Implementation identification

Supplier	
Contact point for queries about the PICS	
Implementation name(s) and version(s)	
Other information necessary for full identification (e.g., name(s) and version(s) of machines and/or operating systems, system name(s))	

NOTES

1 Only the first three items are required for all implementations; other information may be completed as appropriate in meeting the requirement for full identification.

2 The terms Name and Version should be interpreted appropriately to correspond with a supplier’s terminology (e.g., Type, Series, Model).

A.4.2 Protocol summary

Identification of protocol specification	ITU-T Recommendation X.233 (1994) | ISO/IEC 8473-1 : 1994
Identification of corrigenda and amendments to the PICS proforma	
Protocol version(s) supported	
Have any Exception Information items been required (see A.3.3)? YES ☐ NO ☐ (The answer YES means that the implementation does not conform to this Recommendation | International Standard)	

Date of statement	

A.5 Major capabilities

Item	Capability	Reference	Status	Support
* ES	End system		O.1	YES ☐ NO ☐
* IS	Intermediate system		O.1	YES ☐ NO ☐
FL-r	<r> Full protocol	6	M	YES ☐
FL-s	<s> Full protocol	6	M	YES ☐
NSS-r	<r> Non-segmenting subset	5.2	M	YES ☐
* NSS-s	<s> Non-segmenting subset	5.2	IS:M	N/A ☐ YES ☐
			^IS:O	N/A ☐ YES ☐ NO ☐
* IAS-r	<r> Inactive subset	5.2	ES:O	N/A ☐ YES ☐ NO ☐
* IAS-s	<s> Inactive subset	5.2	IAS-r:M	N/A ☐ YES ☐
			^IAS-r:X	N/A ☐ NO ☐

A.6 End systems

A.6.1 Applicability

The PICS proforma items in clause A.6 are applicable only to end system implementations; i.e. those in which item ES in clause A.5 is supported. The items in clause A.6.4.4 are applicable only to end system implementations that support the echo request function; i.e., those in which item eEreq in clause A.6.2 is supported. The items in clause A.6.4.5 are applicable only to end system implementations that support the echo response function; i.e., those in which item eErsp in clause A.6.2 is supported.

A.6.2 Supported functions

Item	Function	Reference	Status	Support
ePDUC	PDU composition	6.1	M	YES ☐
ePDUD	PDU decomposition	6.2	M	YES ☐
eHFA	Header format analysis	6.3	M	YES ☐
ePDUL-s	<s> PDU lifetime control	6.4	M	YES ☐
ePDUL-r	<r> PDU lifetime control	6.4	O	YES ☐ NO ☐
eRout	Route PDU	6.5	M	YES ☐
eForw	Forward PDU	6.6	M	YES ☐
eSegm	Segment PDU	6.7	M	YES ☐
eReas	Reassemble PDU	6.8	M	YES ☐
eDisc	Discard PDU	6.9	M	YES ☐
eErep	Error reporting	6.10	M	YES ☐
eEdec-s	<s> Header error detection	6.11	M	YES ☐
eEdec-r	<r> Header error detection	6.11	M	YES ☐
* eSecu-s	<s> Security	6.13	O	YES ☐ NO ☐
* eSecu-r	<r> Security	6.13	O	YES ☐ NO ☐
* eCRR-s	<s> Complete route recording	6.15	O	YES ☐ NO ☐
* eCRR-r	<r> Complete route recording	6.15	O	YES ☐ NO ☐
* ePRR-s	<s> Partial route recording	6.15	O	YES ☐ NO ☐
* ePRR-r	<r> Partial route recording	6.15	O	YES ☐ NO ☐
* eCSR	Complete source routeing	6.14	O	YES ☐ NO ☐
* ePSR	Partial source routeing	6.14	O	YES ☐ NO ☐
* ePri-s	<s> Priority	6.17	O	YES ☐ NO ☐
* ePri-r	<r> Priority	6.17	O	YES ☐ NO ☐
* eQOSM-s	<s> QOS maintenance	6.16	O	YES ☐ NO ☐
* eQOSM-r	<r> QOS maintenance	6.16	O	YES ☐ NO ☐
* eCong-s	<s> Congestion notification	6.18	eQOSM-s:M	N/A ☐ YES ☐
* eCong-r	<r> Congestion notification	6.18	O	YES ☐ NO ☐
* ePadd-s	<s> Padding	6.12	O	YES ☐ NO ☐
ePadd-r	<r> Padding	6.12	M	YES ☐
eEreq	Echo request	6.19	O	YES ☐ NO ☐
eErsp	Echo response	6.20	O	YES ☐ NO ☐
eSegS	Create segments smaller than necessary	6.8	O	YES ☐ NO ☐

A.6.3 Supported PDUs

Item	NPDU	Reference	Status	Support
eDT-t	DT (full protocol) transmit	7.7	M	YES ☐
eDT-r	DT (full protocol) receive	7.7	M	YES ☐
eDTNS-t	DT (non-segmenting) transmit	7.7	NSS-s:M	N/A ☐ YES ☐
eDTNS-r	DT (non-segmenting) receive	7.7	M	YES ☐
eER-t	ER transmit	7.9	M	YES ☐
eER-r	ER receive	7.9	M	YES ☐
eIN-t	Inactive PDU transmit	7.8	IAS-s:M	N/A ☐ YES ☐
eIN-r	Inactive PDU receive	7.8	IAS-r:M	N/A ☐ YES ☐
eERQ-t	ERQ transmit	7.10	eEreq:M	N/A ☐ YES ☐
eERQ-r	ERQ receive	7.10	M	YES ☐
eERP-t	ERP transmit	7.11	eErsp:M	N/A ☐ YES ☐
eERP-r	ERP receive	7.11	M	YES ☐

A.6.4 Supported parameters

A.6.4.1 DT parameters

Item	Parameter	Reference	Status	Support
edFxPt-s	<s> Fixed part	7.2	M	YES ☐
edFxPt-r	<r> Fixed part	7.2	M	YES ☐
edAddr-s	<s> Addresses	7.3	M	YES ☐
edAddr-r	<r> Addresses	7.3	M	YES ☐
edSeg-s	<s> Segmentation part	7.4	M	YES ☐
edSeg-r	<r> Segmentation part	7.4	M	YES ☐
edPadd-s	<s> Padding	7.5.2	ePadd-s:M	N/A ☐ YES ☐
edPadd-r	<r> Padding	7.5.2	M	YES ☐
edSecu-s	<s> Security	7.5.3	eSecu-s:M	N/A ☐ YES ☐
edSecu-r	<r> Security	7.5.3	eSecu-r:M	N/A ☐ YES ☐
edCRR-s	<s> Complete route recording	7.5.5	eCRR-s:M	N/A ☐ YES ☐
edCRR-r	<r> Complete route recording	7.5.5	eCRR-r:M	N/A ☐ YES ☐
edPRR-s	<s> Partial route recording	7.5.5	ePRR-s:M	N/A ☐ YES ☐
edPRR-r	<r> Partial route recording	7.5.5	ePRR-r:M	N/A ☐ YES ☐
edCSR-s	<s> Complete source routeing	7.5.4	eCSR:M	N/A ☐ YES ☐
edPSR-s	<s> Partial source routeing	7.5.4	ePSR:M	N/A ☐ YES ☐
edQOSM-s	<s> QOS maintenance	7.5.6	c1:M	N/A ☐ YES ☐
edQOSM-r	<r> QOS maintenance	7.5.6	c2:M	N/A ☐ YES ☐
edPri-s	<s> Priority	7.5.7	ePri-s:M	N/A ☐ YES ☐
edPri-r	<r> Priority	7.5.7	ePri-r:M	N/A ☐ YES ☐
edData-s	<s> Data	7.6	M	YES ☐
edData-r	<r> Data	7.6	M	YES ☐
edUnSup2	Are received PDUs containing parameters selecting unsupported Type 2 functions discarded and where appropriate an Error Report PDU generated?	6.21	M	YES ☐
edUnSup3	Are parameters selecting unsupported Type 3 functions ignored?	6.21	M	YES ☐

Definition of conditional status entries:

c1: eQOSM-s OR eCong-s
c2: eQOSM-r OR eCong-r

A.6.4.2 ER parameters

Item	Parameter	Reference	Status	Support
eeFxPt-s	<s> Fixed part	7.2	M	YES ☐
eeFxPt-r	<r> Fixed part	7.2	M	YES ☐
eeAddr-s	<s> Addresses	7.3	M	YES ☐
eeAddr-r	<r> Addresses	7.3	M	YES ☐
eePadd-s	<s> Padding	7.5.2	ePadd-s:M	N/A ☐ YES ☐
eePadd-r	<r> Padding	7.5.2	M	YES ☐
eeSecu-s	<s> Security	7.5.3	eSecu-s:M	N/A ☐ YES ☐
eeSecu-r	<r> Security	7.5.3	eSecu-r:M	N/A ☐ YES ☐
eeCRR-s	<s> Complete route recording	7.5.5	eCRR-s:M	N/A ☐ YES ☐
eeCRR-r	<r> Complete route recording	7.5.5	eCRR-r:M	N/A ☐ YES ☐
eePRR-s	<s> Partial route recording	7.5.5	ePRR-s:M	N/A ☐ YES ☐
eePRR-r	<r> Partial route recording	7.5.5	ePRR-r:M	N/A ☐ YES ☐
eeCSR-s	<s> Complete source routeing	7.5.4	eCSR:M	N/A ☐ YES ☐
eePSR-s	<s> Partial source routeing	7.5.4	ePSR:M	N/A ☐ YES ☐
eeQOSM-s	<s> QOS maintenance	7.5.6	c1:M	N/A ☐ YES ☐
eeQOSM-r	<r> QOS maintenance	7.5.6	c2:M	N/A ☐ YES ☐
eePri-s	<s> Priority	7.5.7	ePri-s:M	N/A ☐ YES ☐
eePri-r	<r> Priority	7.5.7	ePri-r:M	N/A ☐ YES ☐
eeDisc-s	<s> Reason for discard	7.9.5	M	YES ☐
eeDisc-r	<r> Reason for discard	7.9.5	M	YES ☐
eeData-s	<s> Data	7.9.6	M	YES ☐
eeData-r	<r> Data	7.9.6	M	YES ☐
eeUnSup2	Are received PDUs containing parameters selecting unsupported Type 2 functions discarded?	6.21	M	YES ☐
eeUnSup3	Are parameters selecting unsupported Type 3 functions ignored?	6.21	M	YES ☐

Definition of conditional status entries:

c1: eQOSM-s OR eCong-s
c2: eQOSM-r OR eCong-r

A.6.4.3 Inactive network layer protocol PDU parameters

Item	Parameter	Reference	Status	Support
eiNLPI-s	<s> Inactive network layer protocol identifier	7.8.2	IAS-s:M	N/A ☐ YES ☐
eiNLPI-r	<r> Inactive network layer protocol identifier	7.8.2	IAS-r:M	N/A ☐ YES ☐
eiData-s	<s> Data	7.8.3	IAS-s:M	N/A ☐ YES ☐
eiData-r	<r> Data	7.8.3	IAS-r:M	N/A ☐ YES ☐

A.6.4.4 ERQ parameters

Item	Parameter	Reference	Status	Support
eqFxPt-s	<s> Fixed part	7.2	M	YES ☐
eqFxPt-r	<r> Fixed part	7.2	M	YES ☐
eqAddr-s	<s> Addresses	7.3	M	YES ☐
eqAddr-r	<r> Addresses	7.3	M	YES ☐
eqSeg-s	<s> Segmentation part	7.4	M	YES ☐
eqSeg-r	<r> Segmentation part	7.4	M	YES ☐
eqPadd-s	<s> Padding	7.5.2	ePadd-s:M	N/A ☐ YES ☐
eqPadd-r	<r> Padding	7.5.2	M	YES ☐
eqSecu-s	<s> Security	7.5.3	eSecu-s:M	N/A ☐ YES ☐
eqSecu-r	<r> Security	7.5.3	eSecu-r:M	N/A ☐ YES ☐
eqCRR-s	<s> Complete route recording	7.5.5	eCRR-s:M	N/A ☐ YES ☐
eqCRR-r	<r> Complete route recording	7.5.5	eCRR-r:M	N/A ☐ YES ☐
eqPRR-s	<s> Partial route recording	7.5.5	ePRR-s:M	N/A ☐ YES ☐
eqPRR-r	<r> Partial route recording	7.5.5	ePRR-r:M	N/A ☐ YES ☐
eqCSR-s	<s> Complete source routeing	7.5.4	eCSR:M	N/A ☐ YES ☐
eqPSR-s	<s> Partial source routeing	7.5.4	ePSR:M	N/A ☐ YES ☐
eqQOSM-s	<s> QOS maintenance	7.5.6	c1:M	N/A ☐ YES ☐
eqQOSM-r	<r> QOS maintenance	7.5.6	c2:M	N/A ☐ YES ☐
eqPri-s	<s> Priority	7.5.7	ePri-s:M	N/A ☐ YES ☐
eqPri-r	<r> Priority	7.5.7	ePri-r:M	N/A ☐ YES ☐
eqData-s	<s> Data	7.6	M	YES ☐
eqData-r	<r> Data	7.6	M	YES ☐
eqUnSup2	Are received PDUs containing parameters selecting unsupported Type 2 functions discarded and where appropriate an Error Report PDU generated?	6.21	M	YES ☐
eqUnSup3	Are parameters selecting unsupported Type 3 functions ignored?	6.21	M	YES ☐

Definition of conditional status entries:

c1: eQOSM-s OR eCong-s
c2: eQOSM-r OR eCong-r

A.6.4.5 ERP parameters

Item	Parameter	Reference	Status	Support
epFxPt-s	<s> Fixed part	7.2	M	YES ☐
epFxPt-r	<r> Fixed part	7.2	M	YES ☐
epAddr-s	<s> Addresses	7.3	M	YES ☐
epAddr-r	<r> Addresses	7.3	M	YES ☐
epSeg-s	<s> Segmentation part	7.4	M	YES ☐
epSeg-r	<r> Segmentation part	7.4	M	YES ☐
epPadd-s	<s> Padding	7.5.2	ePadd-s:M	N/A ☐ YES ☐
epPadd-r	<r> Padding	7.5.2	M	YES ☐
epSecu-s	<s> Security	7.5.3	eSecu-s:M	N/A ☐ YES ☐
epSecu-r	<r> Security	7.5.3	eSecu-r:M	N/A ☐ YES ☐
epCRR-s	<s> Complete route recording	7.5.5	eCRR-s:M	N/A ☐ YES ☐
epCRR-r	<r> Complete route recording	7.5.5	eCRR-r:M	N/A ☐ YES ☐
epPRR-s	<s> Partial route recording	7.5.5	ePRR-s:M	N/A ☐ YES ☐
epPRR-r	<r> Partial route recording	7.5.5	ePRR-r:M	N/A ☐ YES ☐
epCSR-s	<s> Complete source routeing	7.5.4	eCSR:M	N/A ☐ YES ☐
epPSR-s	<s> Partial source routeing	7.5.4	ePSR:M	N/A ☐ YES ☐
epQOSM-s	<s> QOS maintenance	7.5.6	c1:M	N/A ☐ YES ☐
epQOSM-r	<r> QOS maintenance	7.5.6	c2:M	N/A ☐ YES ☐
epPri-s	<s> Priority	7.5.7	ePri-s:M	N/A ☐ YES ☐
epPri-r	<r> Priority	7.5.7	ePri-r:M	N/A ☐ YES ☐
epData-s	<s> Data	7.6	M	YES ☐
epData-r	<r> Data	7.6	M	YES ☐
epUnSup2	Are received PDUs containing parameters selecting unsupported Type 2 functions discarded and where appropriate an Error Report PDU generated?	6.21	M	YES ☐
epUnSup3	Are parameters selecting unsupported Type 3 functions ignored?	6.21	M	YES ☐

Definition of conditional status entries:

c1: eQOSM-s OR eCong-s
c2: eQOSM-r OR eCong-r

A.6.5 Timers

Item	Timer	Reference	Status	Values	Support	Values supported
eLifReas	Is reassembly timer <= received derived PDU lifetime?	6.8	M		YES ☐	
eReasLim	What values of the reassembly timer are supported?	6.8		500ms to 127.5s		

A.7 Intermediate systems

A.7.1 Applicability

The PICS proforma items in clause A.7 are applicable only to intermediate system implementations; i.e. those in which item IS in clause A.5 is supported. The items in clause A.7.4.3 are applicable only to intermediate system implementations that support the echo request function; i.e., those in which item eEreq in clause A.7.2 is supported. The items in clause A.7.4.4 are applicable only to intermediate system implementations that support the echo response function; i.e., those in which item eErsp in clause A.7.2 is supported.

A.7.2 Supported functions

Item	Function	Reference	Status	Support
iPDUC	PDU composition	6.1	M	YES ☐
iPDUD	PDU decomposition	6.2	M	YES ☐
iHFA	Header format analysis	6.3	M	YES ☐
iPDUL	<s> PDU lifetime control	6.4	M	YES ☐
iRout	Route PDU	6.5	M	YES ☐
iForw	Forward PDU	6.6	M	YES ☐
iSegm	Segment PDU	6.7	iDSNS:M	N/A ☐ YES ☐
iReas	Reassemble PDU	6.8	iErsp:M	N/A ☐ YES ☐
			^iErsp:O	N/A ☐ YES ☐ NO ☐
iDisc	Discard PDU	6.9	M	YES ☐
iErep	Error reporting	6.10	M	YES ☐
iEdec	<s> Header error detection	6.11	M	YES ☐
* iSecu	<s> Security	6.13	O	YES ☐ NO ☐
* iCRR	<s> Complete route recording	6.15	O	YES ☐ NO ☐
* iPRR	<s> Partial route recording	6.15	O	YES ☐ NO ☐
* iCSR	Complete source routeing	6.14	O	YES ☐ NO ☐
* iPSR	Partial source routeing	6.14	O	YES ☐ NO ☐
* iPri	<s> Priority	6.17	O	YES ☐ NO ☐
* iQOSM	<s> QOS maintenance	6.16	O	YES ☐ NO ☐
* iCong	<s> Congestion notification	6.18	O	YES ☐ NO ☐
iPadd	<s> Padding	6.12	M	YES ☐
iEreq	Echo request	6.19	O	YES ☐ NO ☐
* iErsp	Echo response	6.20	O	YES ☐ NO ☐
iSegS	Create segments smaller than necessary	6.8	O	YES ☐ NO ☐
iDSNS	Simultaneous support of subnetworks with different SN-Userdata sizes	Table 9 note 3	O	YES ☐ NO ☐

A.7.3 Supported PDUs

Item	NPDU	Reference	Status	Support
iDT-t	DT (full protocol) transmit	7.7	M	YES ☐
iDT-r	DT (full protocol) receive	7.7	M	YES ☐
iDTNS-t	DT (non-segmenting) transmit	7.7	M	YES ☐
iDTNS-r	DT (non-segmenting) receive	7.7	M	YES ☐
iER-t	ER transmit	7.9	M	YES ☐
iER-r	ER receive	7.9	M	YES ☐
iERQ-t	ERQ transmit	7.10	iEreq:M	N/A ☐ YES ☐
iERQ-r	ERQ receive	7.10	M	YES ☐
iERP-t	ERP transmit	7.11	iErsp:M	N/A ☐ YES ☐
iERP-r	ERP receive	7.11	M	YES ☐

A.7.4 Supported parameters

A.7.4.1 DT parameters

Item	Parameter	Reference	Status	Support
idFxPt-s	<s> Fixed part	7.2	M	YES ☐
idFxPt-r	<r> Fixed part	7.2	M	YES ☐
idAddr-s	<s> Addresses	7.3	M	YES ☐
idAddr-r	<r> Addresses	7.3	M	YES ☐
idSeg-s	<s> Segmentation part	7.4	M	YES ☐
idSeg-r	<r> Segmentation part	7.4	M	YES ☐
idPadd-s	<s> Padding	7.5.2	M	YES ☐
idPadd-r	<r> Padding	7.5.2	M	YES ☐
idSecu-s	<s> Security	7.5.3	iSecu:M	N/A ☐ YES ☐
idSecu-r	<r> Security	7.5.3	iSecu:M	N/A ☐ YES ☐
idCRR-s	<s> Complete route recording	7.5.5	iCRR:M	N/A ☐ YES ☐
idCRR-r	<r> Complete route recording	7.5.5	iCRR:M	N/A ☐ YES ☐
idPRR-s	<s> Partial route recording	7.5.5	M	YES ☐
idPRR-r	<r> Partial route recording	7.5.5	iPRR:M	N/A ☐ YES ☐
idCSR-s	<s> Complete source routeing	7.5.4	iCSR:M	N/A ☐ YES ☐
idCSR-r	<r> Complete source routeing	7.5.4	iCSR:M	N/A ☐ YES ☐
idPSR-s	<s> Partial source routeing	7.5.4	M	YES ☐
idPSR-r	<r> Partial source routeing	7.5.4	iPSR:M	N/A ☐ YES ☐
idQOSM-s	<s> QOS maintenance	7.5.6	M	YES ☐
idQOSM-r	<r> QOS maintenance	7.5.6	c1:M	N/A ☐ YES ☐
idPri-s	<s> Priority	7.5.7	M	YES ☐
idPri-r	<r> Priority	7.5.7	iPri:M	N/A ☐ YES ☐
idData-s	<s> Data	7.6	M	YES ☐
idData-r	<r> Data	7.6	M	YES ☐
idUnSup2	Are received PDUs containing parameters selecting unsupported Type 2 functions discarded and where appropriate an Error Report PDU generated?	6.21	M	YES ☐
idUnSup3	Are parameters selecting unsupported Type 3 functions ignored?	6.21	M	YES ☐

Definition of conditional status entry:

c1: iQOSM OR iCong

A.7.4.2 ER parameters

Item	Parameter	Reference	Status	Support
ieFxPt-s	<s> Fixed part	7.2	M	YES ☐
ieFxPt-r	<r> Fixed part	7.2	M	YES ☐
ieAddr-s	<s> Addresses	7.3	M	YES ☐
ieAddr-r	<r> Addresses	7.3	M	YES ☐
iePadd-s	<s> Padding	7.5.2	M	YES ☐
iePadd-r	<r> Padding	7.5.2	M	YES ☐
ieSecu-s	<s> Security	7.5.3	iSecu:M	N/A ☐ YES ☐
ieSecu-r	<r> Security	7.5.3	iSecu:M	N/A ☐ YES ☐
ieCRR-s	<s> Complete route recording	7.5.5	iCRR:M	N/A ☐ YES ☐
ieCRR-r	<r> Complete route recording	7.5.5	iCRR:M	N/A ☐ YES ☐
iePRR-s	<s> Partial route recording	7.5.5	M	YES ☐
iePRR-r	<r> Partial route recording	7.5.5	iPRR:M	N/A ☐ YES ☐
ieCSR-s	<s> Complete source routeing	7.5.4	iCSR:M	N/A ☐ YES ☐
ieCSR-r	<r> Complete source routeing	7.5.4	iCSR:M	N/A ☐ YES ☐
iePSR-s	<s> Partial source routeing	7.5.4	M	YES ☐
iePSR-r	<r> Partial source routeing	7.5.4	iPSR:M	N/A ☐ YES ☐
ieQOSM-s	<s> QOS maintenance	7.5.6	M	YES ☐
ieQOSM-r	<r> QOS maintenance	7.5.6	c1:M	N/A ☐ YES ☐
iePri-s	<s> Priority	7.5.7	M	YES ☐
iePri-r	<r> Priority	7.5.7	iPri:M	N/A ☐ YES ☐
ieDisc-s	<s> Reason for discard	7.9.5	M	YES ☐
ieDisc-r	<r> Reason for discard	7.9.5	M	YES ☐
ieData-s	<s> Data	7.9.6	M	YES ☐
ieData-r	<r> Data	7.9.6	M	YES ☐
ieUnSup2	Are received PDUs containing parameters selecting unsupported Type 2 functions discarded?	6.21	M	YES ☐
ieUnSup3	Are parameters selecting unsupported Type 3 functions ignored?	6.21	M	YES ☐

Definition of conditional status entry:

c1: iQOSM OR iCong

A.7.4.3 ERQ parameters

Item	Parameter	Reference	Status	Support
iqFxPt-s	<s> Fixed part	7.2	M	YES ☐
iqFxPt-r	<r> Fixed part	7.2	M	YES ☐
iqAddr-s	<s> Addresses	7.3	M	YES ☐
iqAddr-r	<r> Addresses	7.3	M	YES ☐
iqSeg-s	<s> Segmentation part	7.4	M	YES ☐
iqSeg-r	<r> Segmentation part	7.4	M	YES ☐
iqPadd-s	<s> Padding	7.5.2	M	YES ☐
iqPadd-r	<r> Padding	7.5.2	M	YES ☐
iqSecu-s	<s> Security	7.5.3	iSecu:M	N/A ☐ YES ☐
iqSecu-r	<r> Security	7.5.3	iSecu:M	N/A ☐ YES ☐
iqCRR-s	<s> Complete route recording	7.5.5	iCRR:M	N/A ☐ YES ☐
iqCRR-r	<r> Complete route recording	7.5.5	iCRR:M	N/A ☐ YES ☐
iqPRR-s	<s> Partial route recording	7.5.5	M	YES ☐
iqPRR-r	<r> Partial route recording	7.5.5	iPRR:M	N/A ☐ YES ☐
iqCSR-s	<s> Complete source routeing	7.5.4	iCSR:M	N/A ☐ YES ☐
iqCSR-r	<r> Complete source routeing	7.5.4	iCSR:M	N/A ☐ YES ☐
iqPSR-s	<s> Partial source routeing	7.5.4	M	YES ☐
iqPSR-r	<r> Partial source routeing	7.5.4	iPSR:M	N/A ☐ YES ☐
iqQOSM-s	<s> QOS maintenance	7.5.6	M	YES ☐
iqQOSM-r	<r> QOS maintenance	7.5.6	c1:M	N/A ☐ YES ☐
iqPri-s	<s> Priority	7.5.7	M	YES ☐
iqPri-r	<r> Priority	7.5.7	iPri:M	N/A ☐ YES ☐
iqData-s	<s> Data	7.6	M	YES ☐
iqData-r	<r> Data	7.6	M	YES ☐
iqUnSup2	Are received PDUs containing parameters selecting unsupported Type 2 functions discarded and where appropriate an Error Report PDU generated?	6.21	M	YES ☐
iqUnSup3	Are parameters selecting unsupported Type 3 functions ignored?	6.21	M	YES ☐

Definition of conditional status entry:

c1: iQOSM OR iCong

A.7.4.4 ERP parameters

Item	Parameter	Reference	Status	Support
ipFxPt-s	<s> Fixed part	7.2	M	YES ☐
ipFxPt-r	<r> Fixed part	7.2	M	YES ☐
ipAddr-s	<s> Addresses	7.3	M	YES ☐
ipAddr-r	<r> Addresses	7.3	M	YES ☐
ipSeg-s	<s> Segmentation part	7.4	M	YES ☐
ipSeg-r	<r> Segmentation part	7.4	M	YES ☐
ipPadd-s	<s> Padding	7.5.2	M	YES ☐
ipPadd-r	<r> Padding	7.5.2	M	YES ☐
ipSecu-s	<s> Security	7.5.3	iSecu:M	N/A ☐ YES ☐
ipSecu-r	<r> Security	7.5.3	iSecu:M	N/A ☐ YES ☐
ipCRR-s	<s> Complete route recording	7.5.5	iCRR:M	N/A ☐ YES ☐
ipCRR-r	<r> Complete route recording	7.5.5	iCRR:M	N/A ☐ YES ☐
ipPRR-s	<s> Partial route recording	7.5.5	M	YES ☐
ipPRR-r	<r> Partial route recording	7.5.5	iPRR:M	N/A ☐ YES ☐
ipCSR-s	<s> Complete source routeing	7.5.4	iCSR:M	N/A ☐ YES ☐
ipCSR-r	<r> Complete source routeing	7.5.4	iCSR:M	N/A ☐ YES ☐
ipPSR-s	<s> Partial source routeing	7.5.4	M	YES ☐
ipPSR-r	<r> Partial source routeing	7.5.4	iPSR:M	N/A ☐ YES ☐
ipQOSM-s	<s> QOS maintenance	7.5.6	M	YES ☐
ipQOSM-r	<r> QOS maintenance	7.5.6	c1:M	N/A ☐ YES ☐
ipPri-s	<s> Priority	7.5.7	M	YES ☐
ipPri-r	<r> Priority	7.5.7	iPri:M	N/A ☐ YES ☐
ipData-s	<s> Data	7.6	M	YES ☐
ipData-r	<r> Data	7.6	M	YES ☐
ipUnSup2	Are received PDUs containing parameters selecting unsupported Type 2 functions discarded and where appropriate an Error Report PDU generated?	6.21	M	YES ☐
ipUnSup3	Are parameters selecting unsupported Type 3 functions ignored?	6.21	M	YES ☐

Definition of conditional status entry:

c1: iQOSM OR iCong

A.7.5 Timer and parameter values

Item	Timer	Reference	Status	Values	Support	Values supported
iLifReas	Is reassembly timer <= received derived PDU lifetime?	6.8	iReas:M		N/A ☐ YES ☐	
iReasLim	What values of the reassembly timer are supported?	6.8		500ms to 127.5s		

Annex B

Supporting technical material

(This annex does not form an integral part of this Recommendation | International Standard.)

B.1 Data unit lifetime

There are two primary purposes for providing a PDU lifetime capability in the protocol defined by this Recommendation | International Standard. One purpose is to ensure against unlimited looping of protocol data units; while the routeing algorithm should ensure that it will be very rare for data to loop, the PDU lifetime field provides additional assurance that loops will be limited in extent.

The other purpose of the lifetime capability is to provide a means by which the originating Network entity can limit the maximum NSDU lifetime. The OSI Transport protocol class 4 (ITU-T Rec. X.224 | ISO/IEC 8073) assumes that there is a particular maximum NSDU lifetime in order to protect against certain error states in the transport connection establishment and termination phases; *viz.*, if a TPDU does not arrive within the maximum NSDU lifetime, then there is no chance that it will ever arrive. It is necessary to make this assumption, even if the Network layer does not guarantee any particular upper bound on NSDU lifetime; however, it is simpler for Transport protocol class 4 to deal with lost TPDUs than to deal with late TPDUs, and for this reason, it is preferable to discard late TPDUs than to deliver them. It should be noted that NSDU lifetime is not directly associated with the retransmission of lost TPDUs; rather, it is most useful for distinguishing old (duplicate) TPDUs from new TPDUs.

Maximum NSDU lifetime must be provided to a Transport protocol entity in units of time in order to be useful in determining Transport timer values (a Transport entity cannot count "hops").

In the absence of any guaranteed upper bound, it is common to estimate a value for maximum NSDU lifetime. This value is often based upon observation of past performance, and may vary with source and destination. There are two possible ways to deal with the requirement for a limit on the maximum NSDU lifetime:

1) provide a mechanism in the Transport layer to recognize and discard old TPDUs; or

2) specify lifetime in units of time.

The second method requires intermediate systems to decrement the lifetime field by a value which is an upper bound on the time elapsed since the PDU visited the previous intermediate system. The Transport layer relies on the Network layer to discard NSDUs (and hence TPDUs) whose lifetime has expired.

A major disadvantage to employing solution 1 is that Transport entities (instances) are created when required and released when their purpose has been fulfilled; hence, they are by nature temporary. In order to determine whether a particular TPDU is old, functions that recognize and discard old TPDUs must be designed (and must always be present) in addition to those performed by each Transport entity instance. Such functions are extremely complex and impose a non-trivial overhead on Transport layer operation.

Conversely, the state machine associated with the provision of a connectionless-mode service does not require knowledge of previous connection state information to operate correctly. As no additional mechanisms beyond those necessary to correctly bound NPDU lifetime are required to ensure that old NSDUs (and hence old TPDUs) are not delivered to the Transport layer, it is preferable to have the Network layer discard NPDUs whose lifetime has expired, and have the Transport layer deal with lost TPDUs (solution 2).

B.1.1 Determining a value for NPDU lifetime

It is not necessary for each intermediate system to subtract a precise measure of the time that elapsed since an NPDU (containing the TPDU or a segment thereof) visited the previous intermediate system. Where a precise measure is not available, it is sufficient to subtract an overestimate of the actual time taken. In most cases, an intermediate system may simply subtract a constant value which depends upon the typical near-maximum delays that are encountered in a specific underlying service. A more accurate measure may be required for those subnetworks that have both a relatively large maximum delay and a relatively large variation in delay.

As an example, assume that a particular local area network has short average delays, with overall delays generally in the 1 to 5ms range and with occasional delays up to 20ms. In this case, although the relative range in delays might be large (a factor of twenty), it would still not be necessary to measure the actual delay for NPDUs. A constant value of 20ms (or more) could be subtracted for all NPDUs. Similarly, if a single hop satellite link had delays ranging from 0,5s to 0,6s then the higher value could always be used.

If a third subnetwork had normal delays ranging from 0,1 to 1s, but occasionally delivered an NPDU after a delay of 15s, then an intermediate system attached to this subnetwork might find it necessary to determine how long it has actually taken the NPDU to be delivered. Even in this last example, it is more useful to have the intermediate systems determine when the delays are extreme and discard very old NPDUs, and allow the Transport protocol to detect the lost TPDU.

In addition to the time delay within each subnetwork, it is important to consider the time delay within intermediate systems. It should be relatively simple for those intermediate systems that expect to hold on to some data units for significant periods of time to decrement the lifetime appropriately.

B.2 Reassembly lifetime control

In order to ensure a bound on the lifetime of NSDUs, and to effectively manage reassembly buffers in the Network layer, the reassembly function described in clause 6 must control the lifetime of segments representing partially assembled PDUs. This clause discusses methods of bounding reassembly lifetime and suggests some implementation guidelines for the reassembly function.

When segments of a PDU arrive at a destination Network entity, they are buffered until an entire PDU is received, assembled, and passed to the PDU decomposition function. The protocol does not guarantee the delivery of PDUs; hence, it is possible for some segments of a PDU to be lost or delayed such that the entire PDU cannot be assembled in a reasonable length of time. In the case of loss of a PDU segment, for example, this could be forever. There are a number of possible schemes to prevent this:

a) per-PDU reassembly timers,

b) extension of the PDU lifetime control function, and

c) coupling of the reassembly lifetime and Transport retransmission timers.

Each of these methods is discussed in the following clauses.

B.2.1 Method (a)

This method assigns a “reassembly lifetime” to each PDU received and identified by its data unit identifier. This is a local, real time which is assigned by the reassembly function and decremented while some, but not all, segments of the PDU are being buffered by the destination Network entity. If the timer expires, all segments of the PDU are discarded, thus freeing the reassembly buffers and preventing a “very old” PDU from being confused with a newer one bearing the same data unit identifier. For this scheme to function properly, the timers must be assigned in such a fashion as to prevent the phenomenon of reassembly interference (discussed below). In particular, the following guidelines should be followed:

a) The reassembly lifetime must be much less than the maximum PDU lifetime of the network (to prevent the confusion of old and new data units).

b) The lifetime should be less than the Transport protocol’s retransmission timers minus the average transit time of the network. If this is not the case, extra buffers are tied up holding data which has already been retransmitted by the Transport protocol. (Note that an assumption has been made that such timers are integral to the Transport protocol, which in some sense dictates that retransmission functions must exist in the Transport protocol employed).

B.2.1 Method (b)

This method is feasible if the PDU lifetime control function operates based on real or virtual time rather than hop count. In this scheme, the lifetime field of each PDU segment continues to be decremented by the reassembly function of the destination Network entity as if the PDU were still in transit (in a sense, it still is). When the lifetime of any segment of a partially reassembled PDU expires, all segments of that PDU are discarded. This scheme is attractive since the delivery behaviour of this protocol would be identical for segmented and unsegmented PDUs.

B.2.3 Method (c)

This method couples the reassembly lifetime directly to the Transport protocol's retransmission timers, and requires that Transport layer management make known to Network layer management (and hence, to the reassembly function) the values of its retransmission timers for each source from which it expects to be receiving traffic. When a PDU segment is received from a source, the retransmission time minus the anticipated transit time becomes the reassembly lifetime of that PDU. If this timer expires before the entire PDU has been reassembled, all segments of the PDU are discarded. This scheme is attractive since it has a low probability of holding PDU segments that have already been retransmitted by the source Transport entity; it has, however, the disadvantage of depending on reliable operation of the Transport protocol to work effectively. If the retransmission timers are not set correctly, it is possible that all PDUs would be discarded too soon, and the Transport protocol would make no progress.

B.3 The power of the header error detection function

B.3.1 General

The form of the checksum used for PDU header error detection is such that it is easily calculated in software or firmware using only two additions per octet of header, yet it has an error detection power approaching (but not quite equaling) that of techniques which involve calculations that are much more time- or space-consuming (such as cyclic polynomial checks). This clause discusses the power of this error detection function.

The checksum consists of two octets, either of which can assume any value except zero. That is, 255 distinct values for each octet are possible. The calculation of the two octets is such that the value of either is independent of the value of the other, so the checksum has a total of 255 x 255 = 65025 values. If one considers all ways in which the PDU header might be corrupted as equally likely, then there is only one chance in 65025 that the checksum will have the correct value for any particular corruption. This corresponds to 0,0015% of all possible errors.

The remainder of this clause considers particular classes of errors that are likely to be encountered. The hope is that the error detection function will be found to be more powerful, or at least no less powerful, against these classes as compared to errors in general.

B.3.2 Bit alteration errors

First considered are classes of errors in which bits are altered, but no bits are inserted or deleted.

A burst error of length b is a corruption of the header in which all of the altered bits (no more than b in number) are within a single span of consecutively transmitted bits that is b bits long. Checksums are usually expected to do well against burst errors of a length not exceeding the number of bits (16) in the header error detection parameter. The PDU header error detection parameter in fact fails to detect only 0,000019% of all such errors, each distinct burst error of length 16 or less being considered to be equally likely. In particular, it cannot detect an 8-bit burst in which an octet of zero is altered to an octet of 255 (all bits = 1) or vice versa. Similarly, it fails to detect the swapping of two adjacent octets only if one is zero and the other is 255.

The PDU header error detection, as should be expected, detects all errors involving only a single altered bit.

Undetected errors involving only two altered bits should occur only if the two bits are widely separated (and even then only rarely). The PDU header error detection detects all double bit errors for which the spacing between the two altered bits is less than 2040 bits = 255 octets. Since this separation exceeds the maximum header length, all double bit errors are detected.

The power to detect double bit errors is an advantage of the checksum algorithm used for the protocol, versus a simple modulo 65536 summation of the header split into 16 bit fields. The simple summation would not catch all such double bit errors. In fact, double bit errors with a spacing as little as 16 bits apart could go undetected.

This annex does not consider the case in which the checksum itself is erroneously set to be all zero; this case is discussed in B.3.4.

B.3.3 Bit insertion/deletion errors

Although errors involving the insertion or deletion of bits are in general neither more nor less likely to go undetected than are all other kinds of general errors, at least one class of such errors is of special concern. If octets, all equal to either zero or 255, are inserted at a point such that the simple sum C_0 in the running calculation (described in B.3) happens to equal zero, then the error will go undetected. This is of concern primarily because there are two points in the calculation for which this value for the sum is not a rare occurrence, but is expected; namely, at the beginning and the end. That is, if the header is preceded or followed by inserted octets all equal to zero or 255, then no error will be detected. Both cases are examined separately.

Insertion of erroneous octets at the beginning of the header completely misaligns the header fields, causing them to be misinterpreted. In particular, the first inserted octet is interpreted as the Network layer protocol identifier, probably eliminating any knowledge that the data unit is related to this protocol, and thereby eliminating any attempt to perform the checksum calculation or invoking a different form of checksum calculation.

Undetected insertion of erroneous octets at the end of the header, in the absence of other errors, is impossible because the length field unequivocally defines where the header ends. Insertion or deletion of octets at the end of the header requires an alteration in the value of the octet defining the header length. Such an alteration implies that the value of the calculated sum at the end of the header would not be expected to have the dangerous value of zero, and consequently that the error is just as likely to be detected as is any error in general.

Insertion of an erroneous octet in the middle of the header is primarily of concern if the inserted octet has either the value zero or 255, and if the variable C_0 happens to have the value zero at this point. In most cases, this error will completely destroy the parsing of the header, which will cause the data unit to be discarded. In addition, in the absence of any other error, the last octet of the header will be thought to be data. This in turn will cause the header to end in the wrong place. In the case in which the header otherwise parses correctly, the last field will be found to be missing. Even in the case in which the last field is the padding option, and therefore not necessary, the length field for the padding function will be inconsistent with the header length field, and therefore the error can be detected.

B.3.4 Checksum non-calculation errors

Use of the header error detection function is optional. The choice of not using it is indicated by a checksum parameter value of zero. This creates the possibility that the two octets of the checksum parameter (neither of which is generated as being zero) could both be altered to zero. This would in effect be an error not detected by the checksum, since the check would not be made. One of three possibilities exists:

a) A burst error of length sixteen (16) which sets the entire checksum to zero. Such an error could not be detected; however, it requires a particular positioning of the burst within the header. (A calculation of its effect on overall detectability of burst errors depends upon the length of the header.)

b) All single bit errors are detected. Since both octets of the checksum field must be non-zero when the checksum is being used, no single bit error can set the checksum to zero.

c) Where each of the two octets of the checksum parameter has a value that is a power of two, such that only one bit in each equals one (1), then a zeroing of the checksum parameter could result in an undetected double bit error. Furthermore, the two altered bits have a separation of less than sixteen (16), and could be consecutive. This is clearly a decline from the complete detectability previously described.

Where there is particular concern about the possibility of accidental zeroing of the checksum among data units within a network addressing domain, then a restriction may be imposed that all data units whose source or destination lie within the network addressing domain must make use of the header error detection function. Any data units which do not could be discarded, or could be prevented from leaving the local network addressing domain. This protects against errors that occur within the network addressing domain, and would protect all data units whose source or destination lies within the network addressing domain, even where the data path between all such pairs crosses other network addressing domains (errors outside the protected network addressing domain notwithstanding).

Annex C

Algorithms for PDU header error detection function

(This annex does not form an integral part of this Recommendation | International Standard.)

C.1 Symbols used in algorithms

C_0, C_1 are variables used in the algorithm;

i is the number (i.e., the position) of an octet within the header (the position of the first octet is $i = 1$);

O_i is the value of octet i of the PDU header;

n is the number (i.e., the position) of the first octet of the checksum parameter ($n = 8$);

L is the length of the PDU header in octets;

X is the value of octet one of the checksum parameter;

Y is the value of octet two of the checksum parameter.

C.2 Arithmetic conventions

Addition is performed in one of the two following modes:

a) modulo 255 arithmetic;

b) eight-bit one's complement arithmetic, in which, if any of the variables has the value minus zero (i.e. 255), it shall be regarded as though it had the value plus zero (i.e. 0).

C.3 Algorithm for generating checksum parameters

Construct the complete PDU header with the value of the checksum parameter field set to zero;

A: $C_0 \leftarrow C_1 \leftarrow 0$

B: Process each octet of the PDU header sequentially from $i = 1$ to L by

$$C_0 \leftarrow C_0 + O_i$$

$$C_1 \leftarrow C_1 + C_0$$

C: Calculate:

$$X \leftarrow (L - 8)C_0 - C_1 \pmod{255}$$

$$Y \leftarrow (L - 7)(-C_0) + C_1 \pmod{255}$$

D: If $X = 0$, then $X \leftarrow 255$;

E: If $Y = 0$, then $Y \leftarrow 255$;

F: Place the values of X and Y in octets 8 and 9 respectively.

C.4 Algorithm for checking checksum parameters

A: If octets 8 and 9 of the PDU header both contain 0, then the checksum calculation has succeeded; else if either but not both of these octets contains the value zero, then the checksum is incorrect; otherwise, initialize

$$C_0 \leftarrow C_1 \leftarrow 0$$

B: Process each octet of the PDU header sequentially from $i = 1$ to L by

$$C_0 \leftarrow C_0 + O_i$$

$$C_1 \leftarrow C_1 + C_0$$

C: If, when all of the octets have been processed, $C_0 = C_1 = 0$, then the checksum calculation has succeeded; otherwise, the checksum calculation has failed.

C.5 Algorithm to adjust the checksum parameter when an octet is altered

This algorithm adjusts the checksum when an octet (such as the lifetime field) is altered. Suppose the value in octet k is changed by Z = newvalue – oldvalue.

If X and Y denote the checksum values held in octets n and $n + 1$, respectively, then adjust X and Y as follows:

A: If $X = 0$ and $Y = 0$ then do nothing; else if $X = 0$ or $Y = 0$ then the checksum is incorrect; else:

$$X \leftarrow (k - n - 1)Z + X \pmod{255}$$

$$Y \leftarrow (n - k)Z + Y \pmod{255}$$

B: If $X = 0$, then $X \leftarrow 255$;

C: If $Y = 0$, then $Y \leftarrow 255$;

For this protocol, $n = 8$. If the octet being altered is the lifetime field, $k = 4$. For the case in which the lifetime is decreased by one unit ($z = -1$), the assignment statements for the new values of X and Y in the immediately preceeding algorithm simplify to:

$$X \leftarrow X + 5 \pmod{255}$$

$$Y \leftarrow Y - 4 \pmod{255}$$

NOTE — To derive this result, assume that when octet k has the value Z added to it, then X and Y have values Z_x and Z_y added to them. For the checksum parameters to satisfy the conditions of 6.11 both before and after the values are added, the following is required:

$$Z + Z_x + Z_y = 0 \pmod{255}$$

and

$$(L - k + 1)Z + (L - n + 1)Z_x + (L - n)Z_y = 0 \pmod{255}$$

Solving these equations simultaneously yields:

$$Z_x = (k - n - 1)Z$$

and

$$Z_y = (n - k)Z$$