INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

X.260

(10/96)

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

SERIES X: DATA NETWORKS AND OPEN SYSTEM COMMUNICATION

Open System Interconnection – Protocol Identification

Information technology – Framework for protocol identification and encapsulation

ITU-T Recommendation X.260

(Previously "CCITT Recommendation")

ITU-T X-SERIES RECOMMENDATIONS

DATA NETWORKS AND OPEN SYSTEM COMMUNICATION

PUBLIC DATA NETWORKS	X.1-X.199
Services and facilities	X.1-X.19
Interfaces	X.20-X.49
Transmission, signalling and switching	X.50-X.89
Network aspects	X.90-X.149
Maintenance	X.150-X.179
Administrative arrangements	X.180-X.199
OPEN SYSTEM INTERCONNECTION	**X.200-X.299**
Model and notation	X.200-X.209
Service definitions	X.210-X.219
Connection-mode protocol specifications	X.220-X.229
Connectionless-mode protocol specification	X.230-X.239
PICS proformas	X.240-X.259
Protocol Identification	**X.260-X.269**
Security Protocols	X.270-X.279
Layer Managed Objects	X.280-X.289
Conformance testing	X.290-X.299
INTERWORKING BETWEEN NETWORKS	X.300-X.399
General	X.300-X.349
Satellite data transmission systems	X.350-X.399
MESSAGE HANDLING SYSTEMS	X.400-X.499
DIRECTORY	X.500-X.599
OSI NETWORKING AND SYSTEM ASPECTS	X.600-X.699
Networking	X.600-X.629
Efficiency	X.630-X.649
Naming, Addressing and Registration	X.650-X.679
Abstract Syntax Notation One (ASN.1)	X.680-X.699
OSI MANAGEMENT	X.700-X.799
Systems Management framework and architecture	X.700-X.709
Management Communication Service and Protocol	X.710-X.719
Structure of Management Information	X.720-X.729
Management functions	X.730-X.799
SECURITY	X.800-X.849
OSI APPLICATIONS	X.850-X.899
Commitment, Concurrency and Recovery	X.850-X.859
Transaction processing	X.860-X.879
Remote operations	X.880-X.899
OPEN DISTRIBUTED PROCESSING	X.900-X.999

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

FOREWORD

ITU (International Telecommunication Union) is the United Nations Specialized Agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of the ITU. Some 179 member countries, 84 telecom operating entities, 145 scientific and industrial organizations and 38 international organizations participate in ITU-T which is the body which sets world telecommunications standards (Recommendations).

The approval of Recommendations by the Members of ITU-T is covered by the procedure laid down in WTSC Resolution No. 1 (Helsinki, 1993). In addition, the World Telecommunication Standardization Conference (WTSC), which meets every four years, approves Recommendations submitted to it and establishes the study programme for the following period.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC. The text of ITU-T Recommendation X.260 was approved on 5th of October 1996. The identical text is also published as ISO/IEC International Standard 14765.

NOTE

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

CONTENTS

Summary

This Recommendation | International Standard provides a framework for explicit identification and encapsulation of Network Layer protocols.

INTERNATIONAL STANDARD

ITU-T RECOMMENDATION

INFORMATION TECHNOLOGY – FRAMEWORK FOR PROTOCOL IDENTIFICATION AND ENCAPSULATION

1 Scope

In a layered approach to protocol architecture, protocols have a relationship to one another such that a protocol at layer (n) uses the services of the layer below it – the (n – 1) services – which, in turn, are provided by a layer (n – 1) protocol. One of the services used by a layer (n) protocol is the *encapsulation* of its (n) Protocol Data Units (PDUs) in a way which is transparent to it. Such encapsulation is realized by the carriage of the (n) PDUs as user data in an (n – 1) Service Data Unit (SDU).

In a limited case, the operation of a particular protocol at layer (n – 1) implies the operation, above layer (n – 1), of a single layer (n) protocol or single set of related (n) / (n + 1)… protocols. However, in a more general case, there may be more than one protocol (or set of related protocols starting) at layer (n) that can operate above layer (n – 1) in a given environment. In such cases, there is a need for explicit *identification* of the protocol (or set of protocols starting) at layer (n).

There also may be a need to manipulate the (n – 1) protocol (i.e. the *encapsulating* protocol) in certain ways specific to the layer (n) protocol (i.e. the *encapsulated* protocol). Such manipulations form the basis of a set of procedures that must be specified for the layer (n) protocol.

The above observations regarding protocol identification and encapsulation are also applicable in cases where an (n) layer is further divided into sublayers.

Cases in which an (n) protocol operates for the purpose of establishing a parallel universe of protocols (regardless of the layered structure of that universe) also give rise to a need for the (n) protocol to be able to identify the protocol(s) in the parallel universe. In these cases, however, there is no encapsulating/encapsulated relationship between the (n) protocol and the parallel universe set of protocols.

The above principles lead to a need to establish a framework for protocol identification and encapsulation. These principles apply to the relationship between two protocols (recognizing that one of them may be a set of related protocols) and can be applied recursively. This Recommendation | International Standard provides a framework for explicit protocol identification and for protocol encapsulation. Implicit protocol identification (see 4.2) is beyond the scope of this Recommendation | International Standard.

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 International 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 edition of the Recommendations and International 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 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 – Basic Reference Model: The Basic Model*.
- ITU-T Recommendation X.263 (1995) | ISO/IEC TR 9577:1996, *Information technology – Protocol identification in the network layer*.

2.2 Additional references

– ITU-T Recommendation X.37 (1995), *Encapsulation in X.25 packets of various protocols including frame relay*.

– ISO/IEC 13515[1], *Information technology – Telecommunications and information exchange between systems – Generic Multiprotocol Encapsulation (GME): Application to frame relay and ATM*.

3 Abbreviations

EdP	Encapsulated Protocol
EF	Encapsulation Function
EgP	Encapsulating Protocol
EPIF	Encapsulated Protocol Information Field
IdP	Identified Protocol(s)
IgP	Identifying Protocol
IPI	Initial Protocol Identifier
PCI	Protocol Control Information
PDU	Protocol Data Unit
PEM	Protocol Encapsulation Method
PId	Protocol Identification
PIE	Protocol Identification and Encapsulation
PIM	Protocol Identification Method
SDU	Service Data Unit
SPI	Subsequent Protocol Identifier

4 Definitions and concepts

4.1 Basic Reference Model concepts

The following concepts from ITU-T Rec. X.200 | ISO/IEC 7498-1 are used here:

a) concatenation;

b) layer;

c) protocol;

d) Protocol Control Information (PCI);

e) Protocol Data Unit (PDU);

f) protocol identification;

g) protocol identifier;

h) segmentation/reassembly;

i) Service Data Unit (SDU);

j) sublayer.

4.2 Additional definitions and concepts

The definitions and concepts below apply to this Recommendation | International Standard.

4.2.1 explicit protocol identification method: An explicit PIM is one in which Protocol Control Information (PCI) is used to identify a protocol, a set of related protocols, or a family of protocols.

[1] Presently at the stage of draft.

4.2.2 implicit protocol identification method: An implicit PIM is one in which there is no PCI used to identify a protocol. Identification occurs through mechanisms such as coupling in a Recommendation or International Standard of an IgP with an IdP [e.g. stating that a specific (n) protocol is used above an (n – 1) protocol]; association of a physical port of a system with one or a set of related protocols; or association at provisioning of a "permanent" connection.

4.2.3 set of alternative protocols: Given protocols $prot_1$, $prot_2$, etc., then $prot_1$, $prot_2$, etc. all operate at the same layer or sublayer.

4.2.4 set of related protocols: Given protocols $prot_1$, $prot_2$, etc., then $prot_1$ operates at layer (n), $prot_2$ operates at layer (n + 1), etc. (where the layers may also be hierarchical sublayers).

4.2.5 family of protocols: Given a set of alternative protocols $prot_1$, $prot_2$, etc., a single identifier is used to identify the set of alternative protocols as a whole, thereby requiring additional methods to identify one member of the family.

5 Overview

5.1 General

As discussed above, there may exist a relationship between protocols that gives rise to a need for one protocol – the Identifying Protocol (IgP) – to identify one of a set of alternative protocols, a set of related protocols, or a family of protocols – the Identified Protocol(s) (IdP). As a result of the identification process, a second relationship may be created between an Encapsulating Protocol (EgP) and an Encapsulated Protocol(s) (EdP). In some cases, the IgP and the EgP may be the same protocol. It is usually the case that an IdP and an EdP are the same.

To provide a basis for developing the necessary relationships among specific protocols, a framework is developed here to set out the principles of Protocol Identification and Encapsulation (PIE). These principles recognize the following aspects of PIE:

a) development of Protocol Identification Methods (PIMs) for identifying an IdP (e.g. the location in the IgP – such as in a particular field: header, trailer, etc. – used to identify the IdP);

b) for each PIM, registration of values of IdPs;

c) requirement on an IgP to specify the PIM it uses to identify IdPs and any further IgP-specific procedures involving the PIM;

d) development of Protocol Encapsulation Methods (PEMs) for use by EgPs; and

e) specification of operations (e.g. limitations, specific manipulations, etc.) of an EgP for a specific EdP.

The above aspects are depicted in Figure 1.

Annex A presents the current status of Recommendations and International Standards in alignment with the framework depicted in Figure 1.

5.2 Interworking and encapsulation

Interworking and *encapsulation* of protocols are two closely related concepts. For the purposes of this Recommendation | International Standard, the following distinctions are made.

Interworking occurs between two or more protocols at the same layer (or sublayer). It is concerned only with the semantic aspects of the (n) layer protocols. In particular, interworking is concerned with the transformation between the semantics of an (n) layer protocol used on one interface and the semantics of other (n) layer protocols used on the other interfaces. The protocols used on the different interfaces may or may not be the same. The transformation between protocols may result in the preservation of the semantic content of all protocols on an end-to-end basis. The transformation only applies to the set of abstract capabilities (or *service)* which the protocols have in common. On the other hand, the transformation may result in a loss of semantic content when crossing interfaces.

Encapsulation (or *tunneling* as it is sometimes called) occurs when a given protocol's PDU (or set of PDUs if the protocol provides segmentation/reassembly capabilities) is used to carry the PDUs of another protocol [that is, the user data parameter of an (n – 1) SDU is used to carry the (n) PDU(s)]. In the general case, no other relationships, such as a strict layering relationship, need exist between the two protocols (e.g. allowing for sublayering or for a given protocol to be encapsulated by protocols with different layer classifications). Encapsulation completely preserves the semantics of the EdP.

From the perspective of this Recommendation | International Standard, *port access*, as a method for interworking as defined in Recommendation X.300, is viewed as a method of encapsulation.

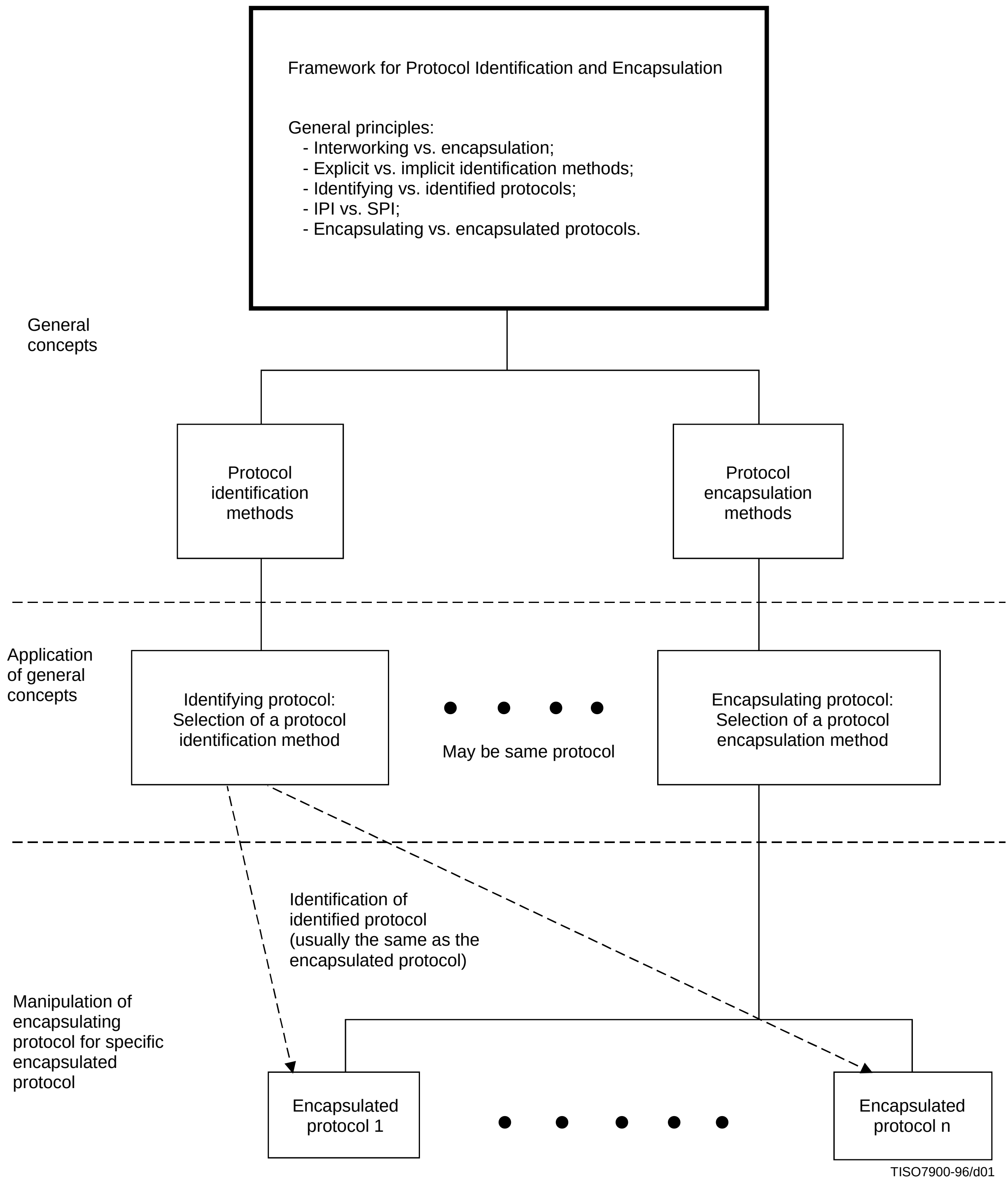


Figure 1 – Framework for protocol identification and encapsulation principles

6 Principles of protocol identification

6.1 Need for protocol identification

The need for PId arises, for the general case, when there is more than one IdP (or set of related IdPs) that can be used in a specific environment (e.g. layer or parallel universe). In such cases, identification of the IdP (or set of related IdPs) is necessary to allow for meaningful communication. The process of PId needs to be performed for a specific instance of communication. Such instances can be:

a) for the lifetime of a connection of the IgP, so that identification or negotiation/selection of one of a number of alternative IdPs (or alternative sets of related IdPs) is required to be done during the IgP's connection establishment phase;

b) for the transmission of a single SDU (in the case of a connection-mode IgP, the selection of allowing multiple concurrent IdPs would have been identified during the IgP's connection establishment phase).

For cases where a multiplicity of alternative IdPs is selected for use in an instance of communication of the IgP, the IdPs may operate concurrently or sequentially with respect to the IgP. The use of multiple IdPs may require agreement of the identities of the specific set of alternative protocols to be used for the instance of communication.

It is also possible that a set of alternative protocols can be identified as a single family, in which case further identification methods are needed to identify a specific member of a family in an instance of communication.

As a result of the need developed above for protocol identification, the following are necessary:

a) registries of values to identify protocols;

b) Protocol Identification Methods (PIMs) to provide a basis for negotiating/selecting one or more IdPs;

c) explicit PCI in the IgP to identify the specific IdP (or family or set of related protocols).

These elements are discussed below.

6.2 Protocol identifier registries and values

A register of values (which itself can be a Recommendation or International Standard or part thereof) is used to record how a protocol, when used as an IdP, is identified. Such a register should be easily modifiable and authority for such modifications shall be identified.

It is permissible for an IdP to appear in more than one register, with the same or a different value.

6.3 Protocol identification methods

A PIM is used to identify a specific IdP (or family or set of related protocols) for use in a specific instance of communication. The PIM can be either implicit or explicit (see 4.2). Implicit PIMs are beyond the scope of this Recommendation | International Standard.

Associated with an explicit PIM is a register of allowed protocol identifier values (see 6.2). It is possible for the same register to be associated with many PIMs rather than developing a new register for use with different PIMs.

An explicit PIM requires the use of PCI to identify protocols. There can be many PIMs, although a particular IgP may support only a few (usually one). An IgP shall specify the PIM it uses. Such specification shall also include the location and number of octets of the PCI used for the PIM.

When an IgP supports several PIMs, it may be desirable to identify an IdP using the PIM that results in the least number of octets. In any case, the particular PIM used to identify an IdP should be specified to ensure interworking.

A PIM may allow for negotiation/selection of IdPs for a specific instance of communication as follows:

a) only one IdP to be selected (for use with a connection of the IgP, where the IdP is identified by the PIM during the connection request phase of the IgP or just identification of the specific IdP during the data transfer phase of the IgP);

b) only one IdP to be selected for use with a connection of the IgP but where negotiation of the specific IdP (from a set of alternative IdPs) takes place during the connection establishment phase of the IgP;

c) a multiplicity of alternative IdPs to be selected (perhaps requiring negotiation of the specific set of alternative IdPs by the PIM during the connection establishment phase of the IgP or just identification of the specific IdP during the data transfer phase of the IgP).

In cases where usage of a multiplicity of alternative IdPs has been agreed for an instance of communication, a PIM may also provide for the specification of whether, during the transfer of data, only one, or, alternatively, more than one IdP is to be used in conjunction with a single SDU. That is, the PIM may also specify aspects of encapsulation (see clause 7).

In cases a) and b) above, there is no need to further identify the IdP during the data transfer phase of the IgP. In case c), further identification of the IdP(s) is required in the IgP's PDUs or IdP's SDUs.

6.4 Protocol identifiers

Protocol identifiers, when explicit, occur in PCI and are based on values maintained in a register (see 6.2).

The circumstance in which an IdP is used determines whether there is a need for it to identify itself. Such need arises when:

a) the IdP is one of a specific family of protocols which has been identified;

b) the IdP has not been identified by an IgP and alternative IdPs exist that may be used.

The second case may be regarded as a general or null family. Nevertheless, the IdP must identify itself in both cases. For both of these cases, the location of the protocol identifier must be specified. Such an identifier is known as an *Initial Protocol Identifier* (IPI). Typically, the IPI will be at the beginning of the (n – 1) layer's SDU; in this case, the IdPs may be regarded as *header-based protocols*. However, the identifier may also be at the end of the SDU in the case of *trailer-based protocols*. When both header- and trailer-based protocols use the same (n – 1) protocol, identifying mechanisms are needed in the (n – 1) protocol to properly differentiate between the header- and trailer-based protocols in the (n) layer.

An IdP need not (but, nevertheless, still may) identify itself when its usage has been unambiguously identified by an IgP.

An IdP may also be an IgP. In such cases, the identifier used for identifying subsequent protocols is known as a *Subsequent Protocol Identifier* (SPI). As stated in 6.3, the PIM used by the SPI (including the location of the SPI in the IgP's PCI) shall be specified. It may be the case that what an IgP views as an SPI may be an IPI from the perspective of the IdP.

A relationship between IPI and SPI is depicted in Figure 2.

It is possible for a subsequent protocol, in turn, to identify further protocols within a layer (i.e. to have a nesting of protocols). It is also possible, in some limited cases, for there to be multiple "initial" protocols. For example, when a data compression protocol is used as the initial protocol, the compressed protocol itself is identified by an IPI.

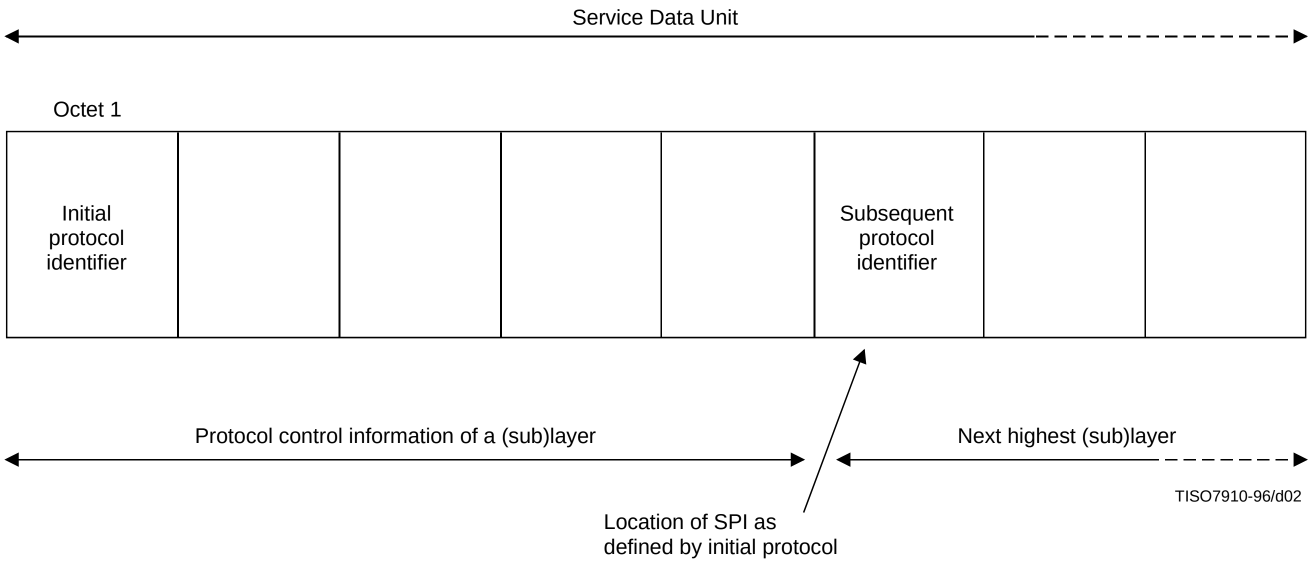


Figure 2 – Relationship of IPI and SPI

7 Principles of protocol encapsulation

Protocol encapsulation implies a relationship between two protocols – an EgP and an EdP. This relationship involves the following dimensions:

a) manipulations of an EdP;

b) manipulations/limitations of a specific EgP for a specific EdP;

c) identification of the EdP, as needed, as it may be encapsulated in an EgP;

d) encapsulation of one or more of the EdP's PDUs, including their delimitation, in the EgP.

These dimensions are embodied in an Encapsulation Function (EF).

The operation of an EF involves two elements:

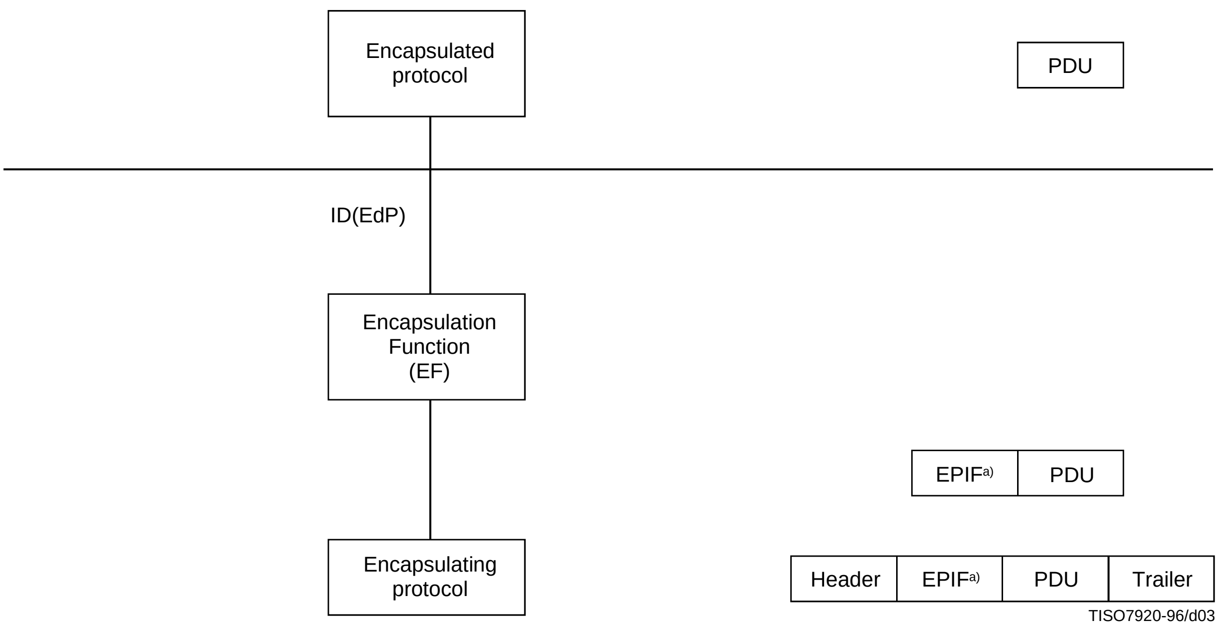
a) the static definition of the above dimensions of the EF;

b) the dynamic operation of the EF in conjunction with zero or more other EFs, in the context of their respective static definitions, to encapsulate the PDU(s) of EdP(s), as provided in the user data parameter in primitives of the service supported by the EgP, during a particular instance of communication.

It is beyond the scope of this framework to specify any limitations during instances of communication on how many EdPs may be encapsulated in an EgP or how EFs with similar characteristics in one or more of the above dimensions may be combined.

7.1 Encapsulation function

An EF performs encapsulation as discussed above in clause 7. The EF resides in the same system as the EgP and EdP. The generic operation of the EF is depicted in Figure 3.



[a] EPIF: Encapsulated Protocol Information Field, when present, may contain:
– EdP identification; and/or
– EdP-PDU delimiting information (e.g. length information).

Figure 3 – Generic operation of an encapsulation function

A specific EF is defined by the realization of:

a) the dimensions enumerated in clause 7 as applied to a coupled set of an EdP and an EgP; and

b) the definition of the EF's PCI (encapsulated protocol information field as shown in Figure 3), if any.

For a specific coupling of an EdP-Y with an EgP-X and the EF's PCI, the EF may be designated as EF(Y,X).

For every EF(Y,X) in a system, there is a peer EF(Y,X) in another system that performs the same functions.

7.1.1 Manipulations/limitations of EgPs and EdPs

There may be an explicit need to specify certain operations applicable to an EgP when used with a specific EdP. Such operations include:

a) manipulation of the protocol elements of the EgP (e.g. when a connection for the EgP should be established); and/or

b) limitations on the usage of certain elements of the EgP.

Similarly, there may also be an explicit need to specify certain operations pertaining to the EdP. An example of this is that an EF need not encapsulate the entire PDU of the EdP. For example, error-protection fields included in the EdP may be stripped off before encapsulation in the EgP since the latter may provide its own error-protection capability.

The specifications of the above operations are associated with a particular EdP/EgP pair. They should be outside of the Recommendation or International Standard in which the EdP or the EgP is documented, since:

a) they are specific to a particular EdP and an EgP may support more than one EdP;

b) an EdP may have a different set of manipulations and limitations for each EgP in which it can be encapsulated.

Figure 4 illustrates possible static relationships among a set of EdPs and EgPs. Dynamic relationships during instances of communication would involve a proper subset of those shown in Figure 4.

NOTE – The concept of a set of manipulations/limitations pertaining to an EgP is defined for the Network Layer in terms of a Subnetwork Dependent Convergence Protocol (see ISO/IEC 8648). The concept here, as it applies to an EF, is meant to be more general and apply to any layer. Further, it does not apply to the provision of a layer service.

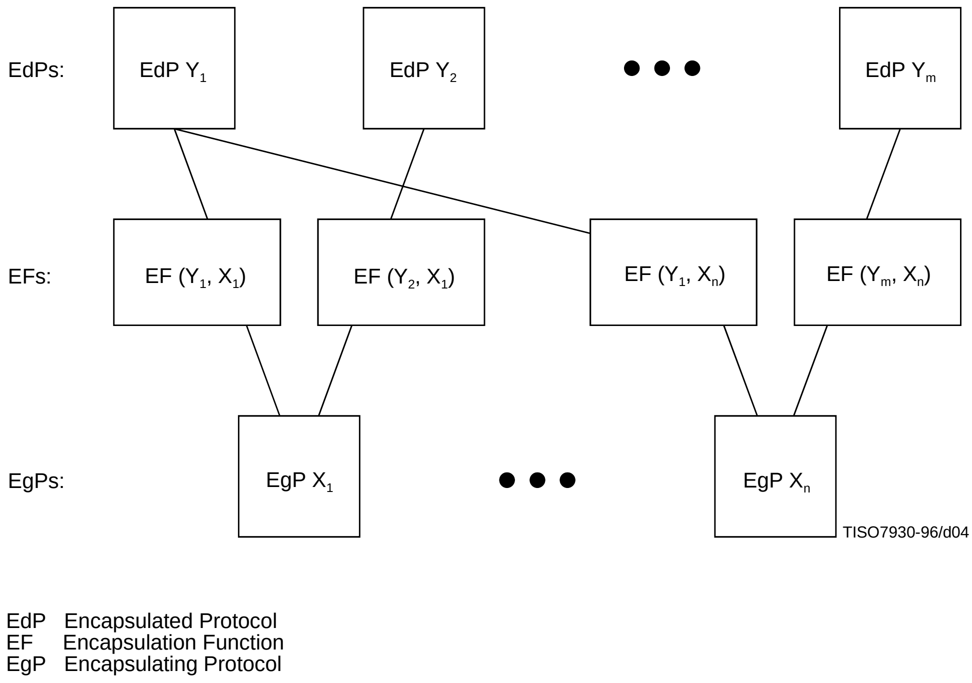


Figure 4 – Possible relationships of EdPs and EgPs

7.1.2 Identification of EdP

Identification of the EdP encapsulated in an EgP may be:

a) implicit when only one EdP is to be encapsulated in an EgP; or

b) explicit when more than one EdP may be encapsulated in an EgP.

NOTE – Identification may be explicit even when only one EdP is to be encapsulated in an EgP.

Identification of the EdP may use a PIM (see 6.3) or the register of values associated with a PIM.

7.1.3 Delimiting of EdP's PDUs

The EF shall clearly identify the boundaries of each PDU of the EdP. Depending upon the characteristics of the encapsulation, this can require the EF to provide PCI in the EPIF (see 7.1). Considerations to be taken into account are:

a) the possibility of padding octets in the EPIF to permit alignment of the EdP's PDU;

b) the use of length information, particularly when more than one PDU of the EdP can be encapsulated by the EF in the EgP during an instance of communication (length information may be provided even when only one PDU of the EdP is encapsulated in the EgP).

NOTE 1 – The case where an EdP performs concatenation of multiple of its own PDUs into one (n – 1), SDU does not require the EF to provide PCI since this is a function of the EdP rather than the EF. To the EF, this appears to be one PDU of the EdP since the concatenated PDUs are conveyed in one SDU.

It is strongly recommended that an EF that permits optional padding octets in the EPIF should specify the value '0000-0000' for such octets.

NOTE 2 – The issue underlying the above recommendation is that of explicit identification of the EdP (see 7.1.2). A receiving EF needs to distinguish padding octets from the first actual octet of the EdP's PDU. Having a single value for padding octets maximizes the potential applicability of explicitly PIMs in a wide variety of multi-protocol environments.

7.2 Protocol encapsulation methods

In real systems, it is possible that many EdPs may be encapsulated in one EgP during instances of its communication – thereby involving the respective EFs for each EdP with the common EgP. In such cases, PCI is needed to identify the EdPs – therefore, the EFs need to share common PCI for this purpose. Where multiple PDUs of the EdPs can be encapsulated by the EFs, common PCI is also needed for this purpose. Likewise, any manipulations of the EgP must also be common. A Protocol Encapsulation Method (PEM) can be viewed as the collection of EFs operating over the same EgP using the same PCI. The PEM can be depicted as shown in Figure 5.

Note that Figure 5 closely resembles Figure 3. An EF may be viewed as the special case of a PEM where only one EdP is encapsulated in an EgP.

The PEM provides for the EgP to encapsulate, in one (or, if the EgP provides segmentation/ reassembly, a series) of its own PDUs:

a) one PDU of the only EdP supported in an instance of communication;

b) multiple PDUs of the only EdP supported in an instance of communication;

c) if support for multiple EdPs has been negotiated for the instance of communication (see 6.3), multiple PDUs from the same EdP;

d) if support for multiple EdPs has been negotiated for the instance of communication (see 6.3), PDUs from the different EdPs.

Figures 6 through 11 show examples of the above relationships.

7.3 Relationships among EFs, EdPs, and EgPs

The relationship among EFs, EdPs, and EgPs is illustrated in Figure 12. Note the following concepts shown in the figure:

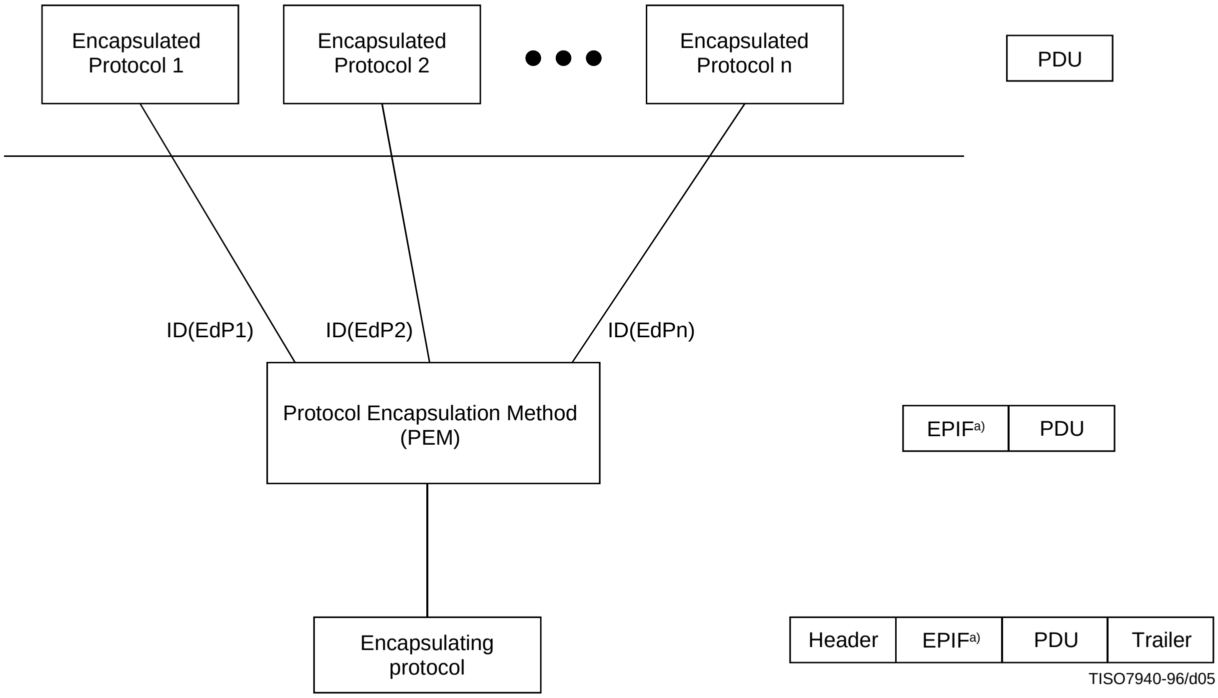
a) Protocols X_1 and X_2 being encapsulated in the same Protocol A in System 1 using respective EFs;

b) Protocol X_1 also serving as an encapsulating protocol of Protocol Y in System 1;

c) Protocol X_1 being encapsulated in Protocol A in System 1 while being encapsulated in Protocol Z in System 2;

d) the peer-to-peer relationships of the various EFs.

Note that an EdP may itself be an EgP with its own EF', thereby providing a recursive encapsulation capability. Any such encapsulated protocols are not known to other EFs except the peer EF'.



[a] EPIF: Encapsulated Protocol Information Field; when present, may contain:
– EdP identification; and/or
– EdP-PDU delimiting information (e.g. length information).

Figure 5 – Combination of multiple EFs over a single EgP

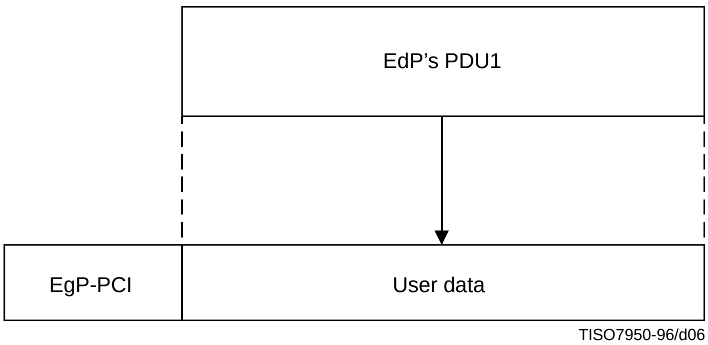


Figure 6 – One PDU of only supported EdP encapsulated in EgP

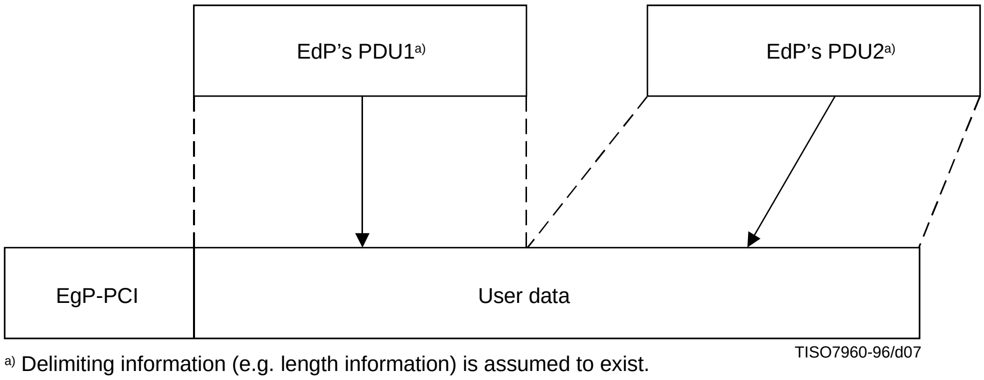


Figure 7 – Multiple PDUs of only supported EdP encapsulated in EgP

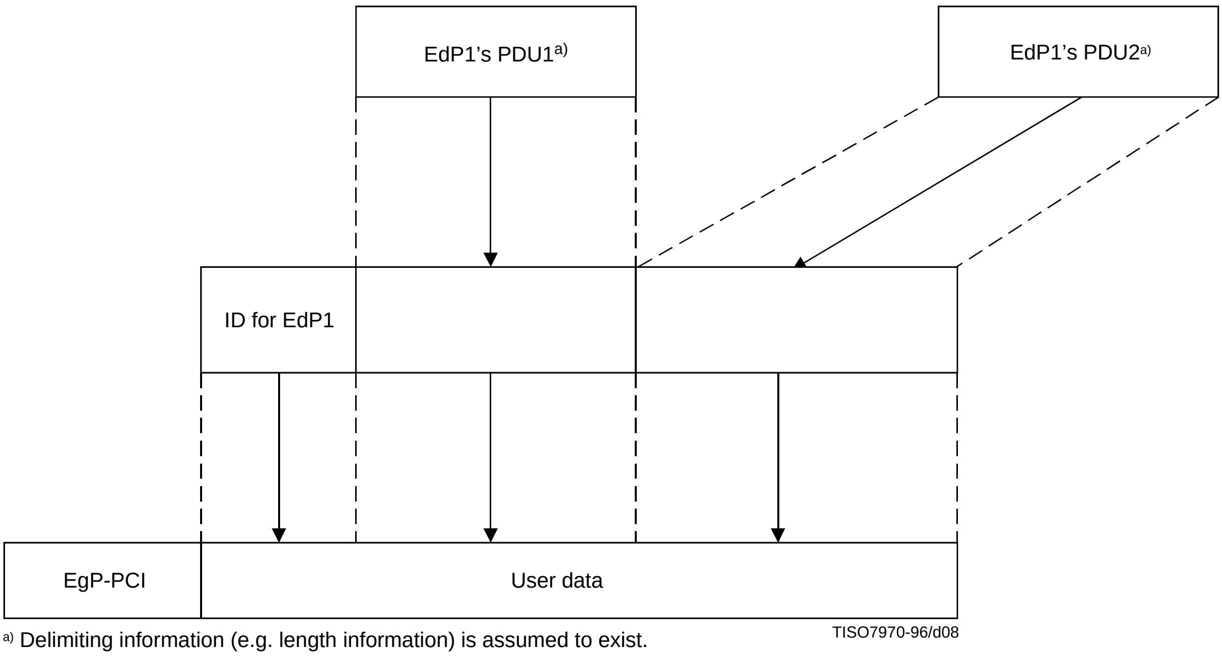


Figure 8 – Multiple PDUs of one of the EdPs encapsuled in EgP

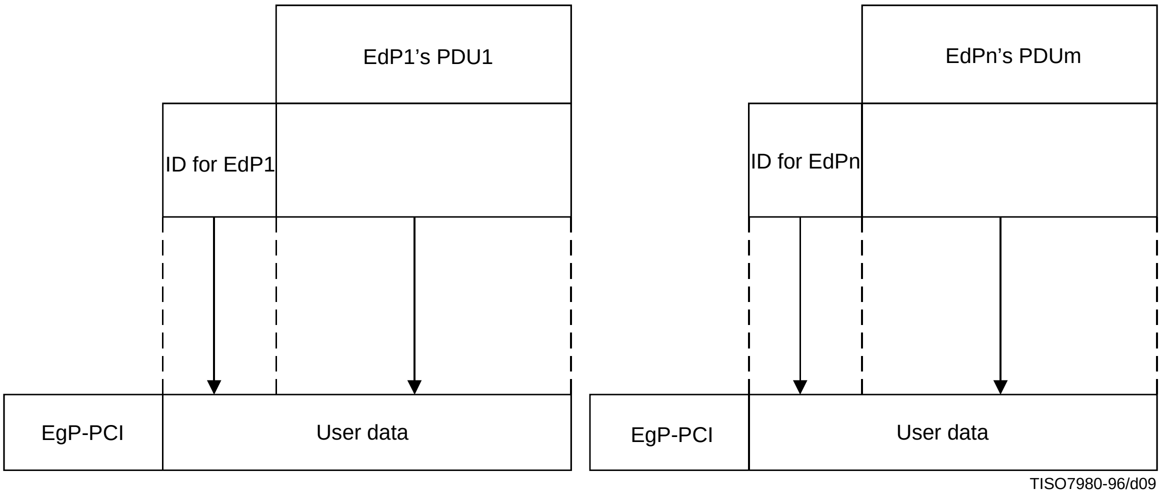


Figure 9 – PDUs from multiple EdPs encapsulated in EgP

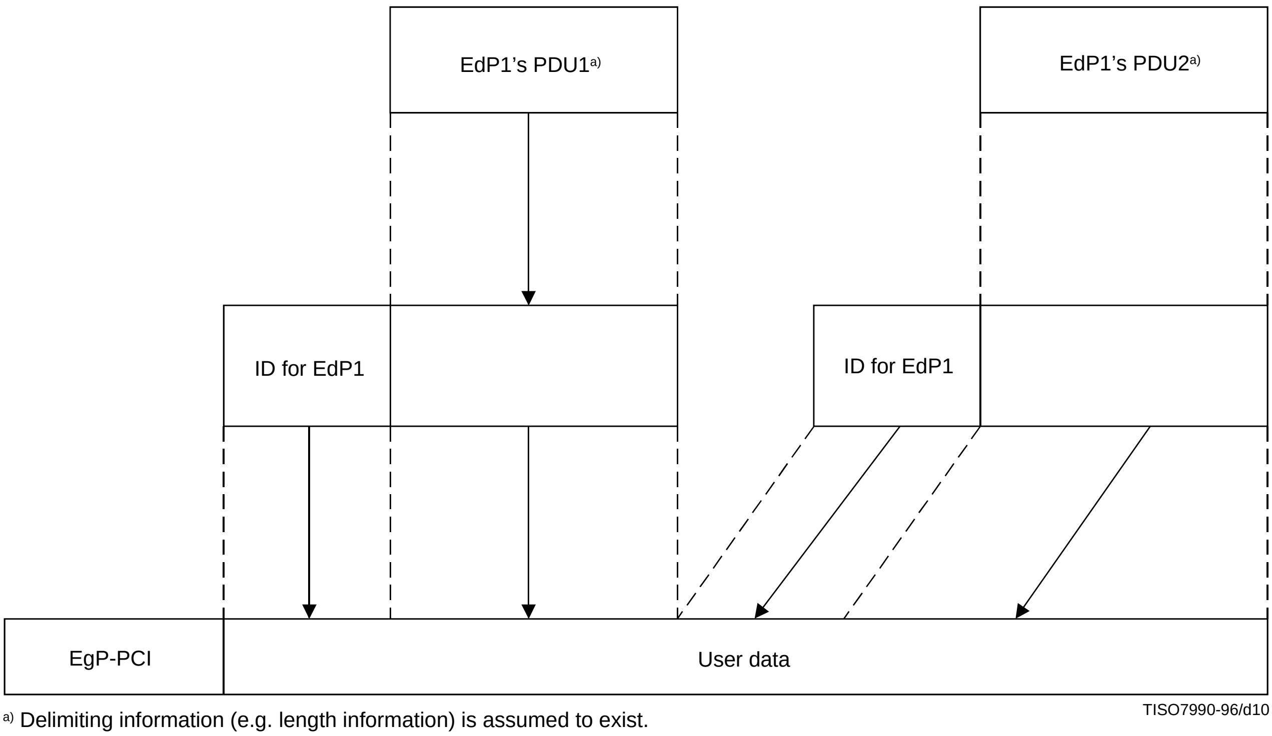


Figure 10 – PDUs from same EdP encapsulated in EgP

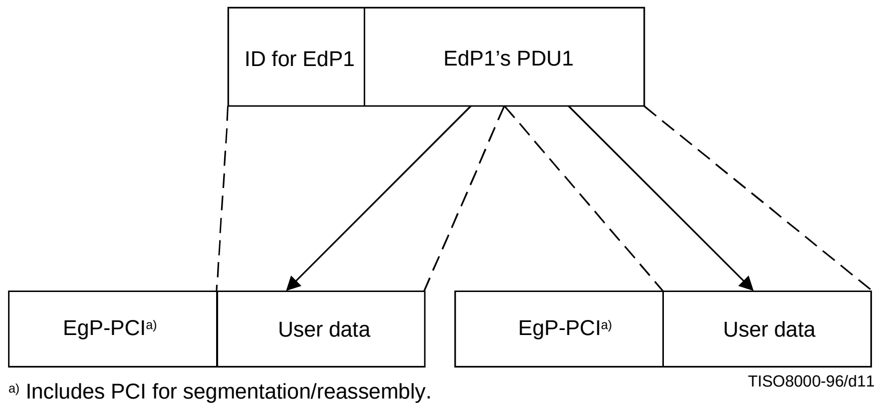


Figure 11 – One PDU of EDP encapsulated within multiple PDUs of EgP

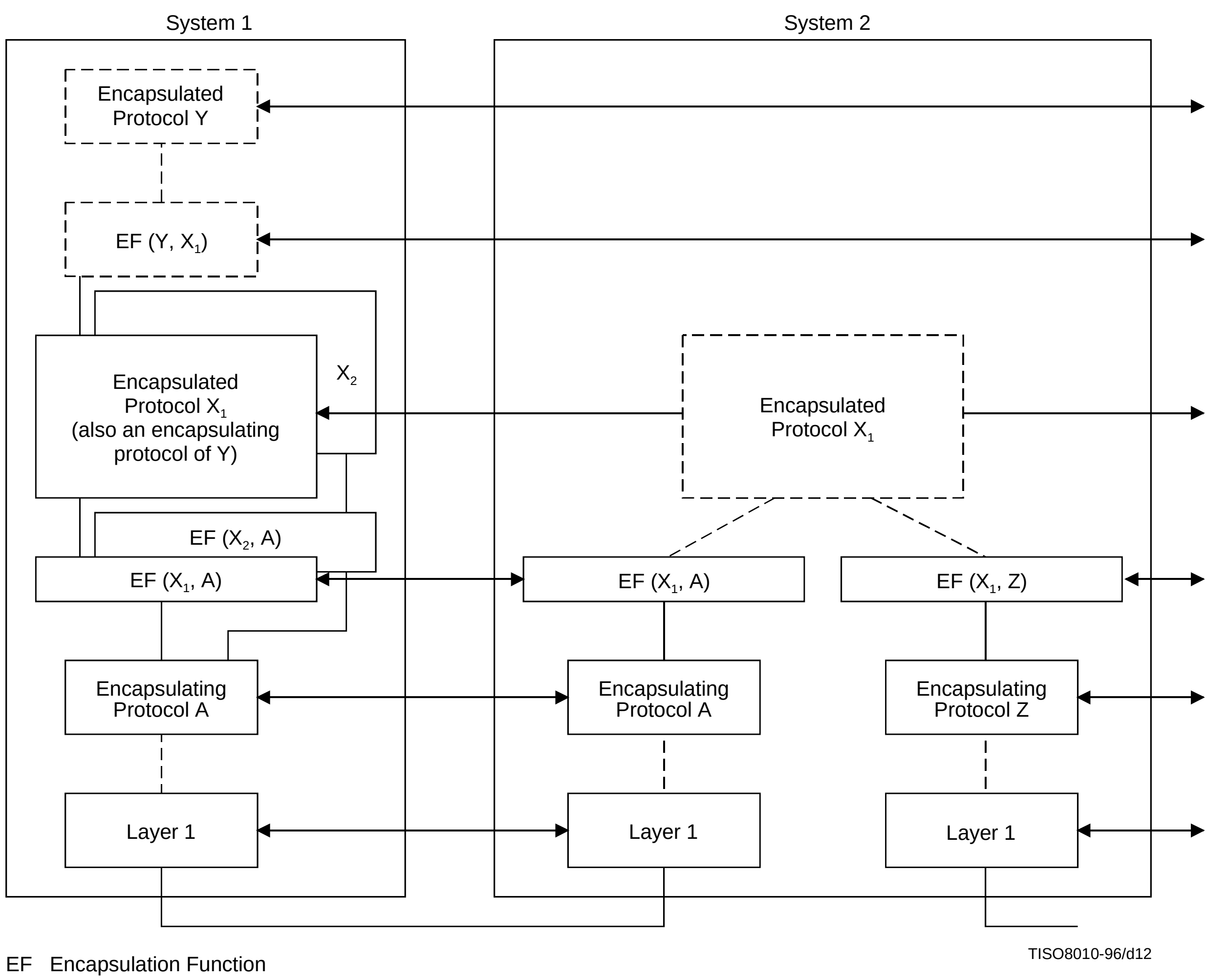


Figure 12 – Relationships among EFs, EdPs, and EgPs

Annex A

Current Recommendations | International Standards supporting PIE principles

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

Figure A.1 depicts current Recommendations and International Standards supporting the principles discussed in the main body of this Recommendation | International Standard. In Figure A.1, this Recommendation | International Standard is shown at the top in box (1). Recommendations and International Standards in boxes (2) specify protocol registries and PIMs for use in various environments while those in boxes (3) deal with encapsulation for general or specific cases. A protocol can be identified in numerous ways. For example, the X.25 Packet Layer Protocol can be identified by an LLC address in a LAN or by an LLC value in an ISDN.

Abbreviations used in this annex:

ATM	Asynchronous Transfer Mode
CLNP	Connectionless-mode Network Protocol
HLC	High Layer Compatibility (Information Element)
IP	Internetwork Protocol
ISDN	Integrated Services Digital Network
LAN	Local Area Network
LLC	Logical Link Control (as used in LANs)
LLC	Low Layer Compatibility (Information Element as used in ISDN)
NLPID	Network Layer Protocol Identification
SNAP	Subnetwork Access Protocol

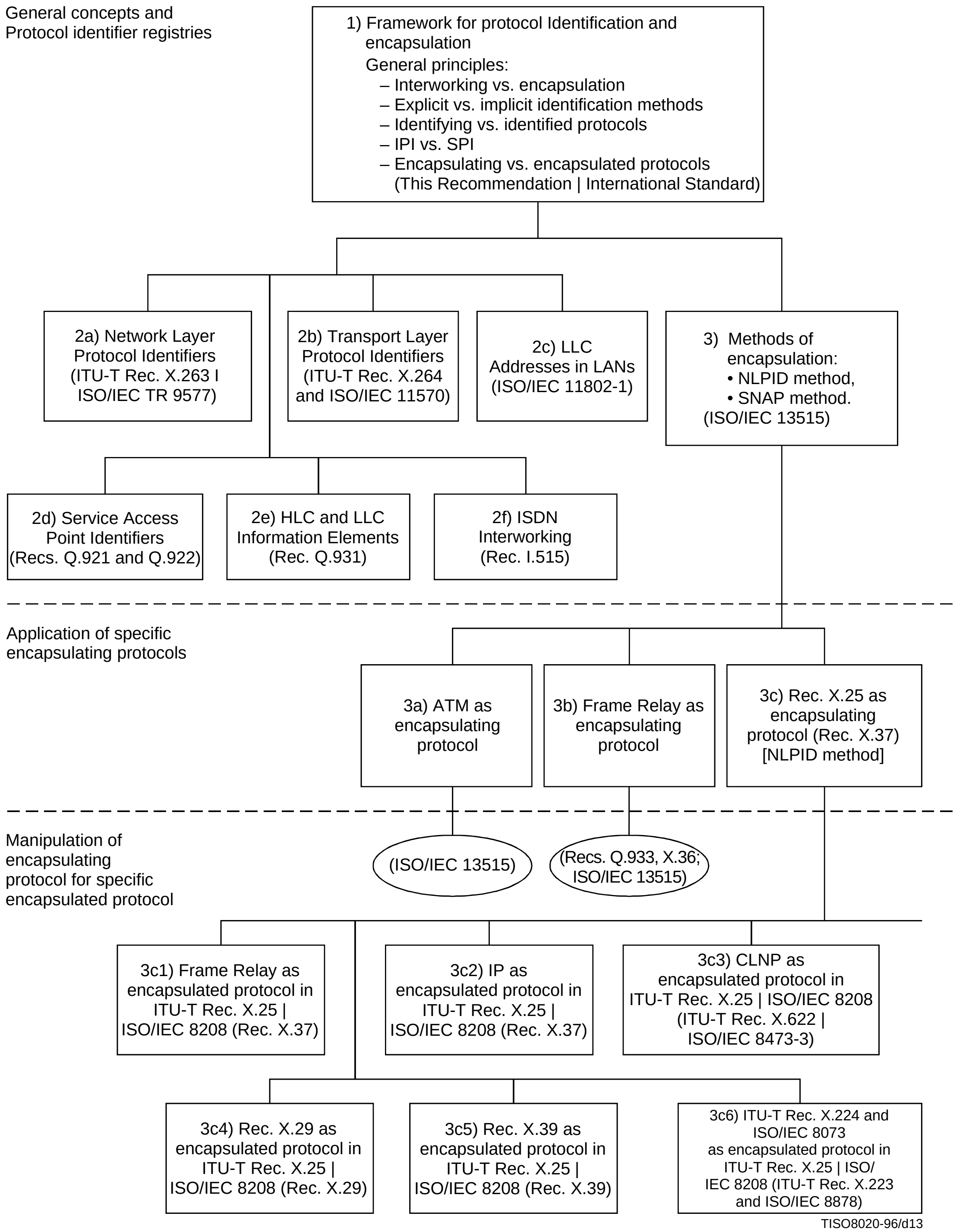


Figure A.1 – Relationship of Recommendations | International Standards supporting principles of protocol identification and encapsulation

Annex B

Examples of protocol identification and encapsulation methods

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

Examples of how protocol identification and encapsulation are provided in different environments are depicted in Figures B.1 to B.6. These examples are not exhaustive; other known examples include:

a) X.25 with LAPB and X.25 PLP (implicit PIM);

b) LANs: SNAP which is also used as a PIM in other environments;

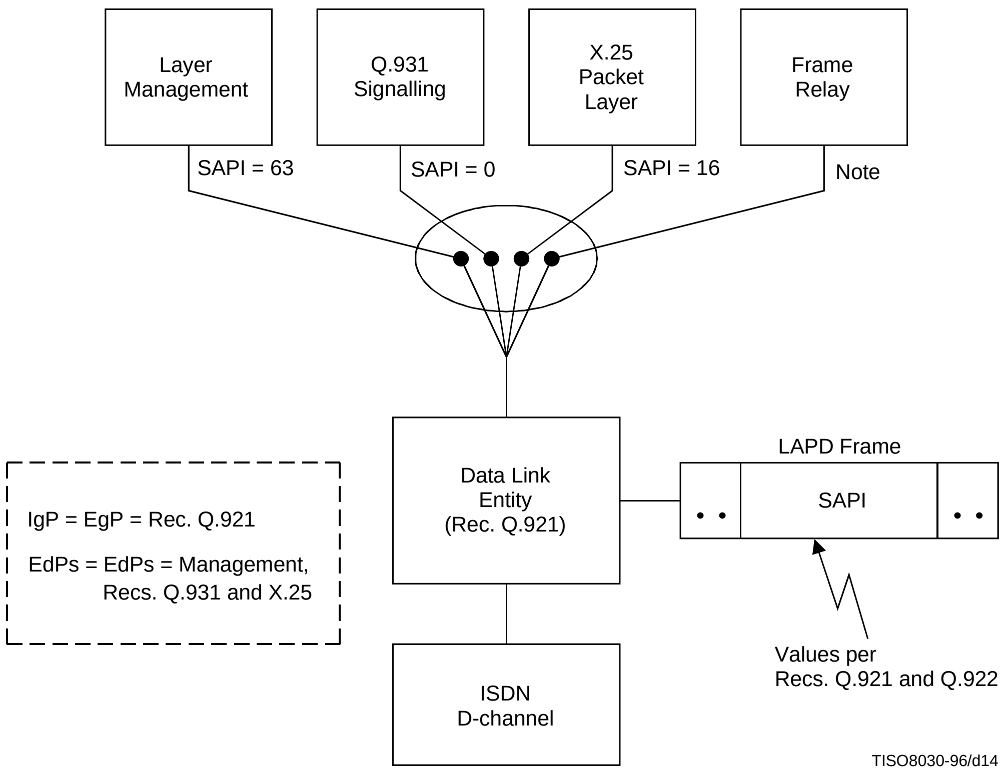
c) Q.931 HLC Information Element;

d) X.37 PIM, with values per ITU-T Rec. X.263 | ISO/IEC TR 9577 but allowing for negotiation;

e) ITU-T Rec. X.273 | ISO/IEC 11577;

f) PPP;

g) identifiers in X.25 DATA packets when Q-bit = 1.



NOTE – Values of Data Link connections on the D-channel that correspond to SAPI values from 32 to 61 are used for Frame Relay service.

Figure B.1 – ISDN D-channel with “SAPI” identification

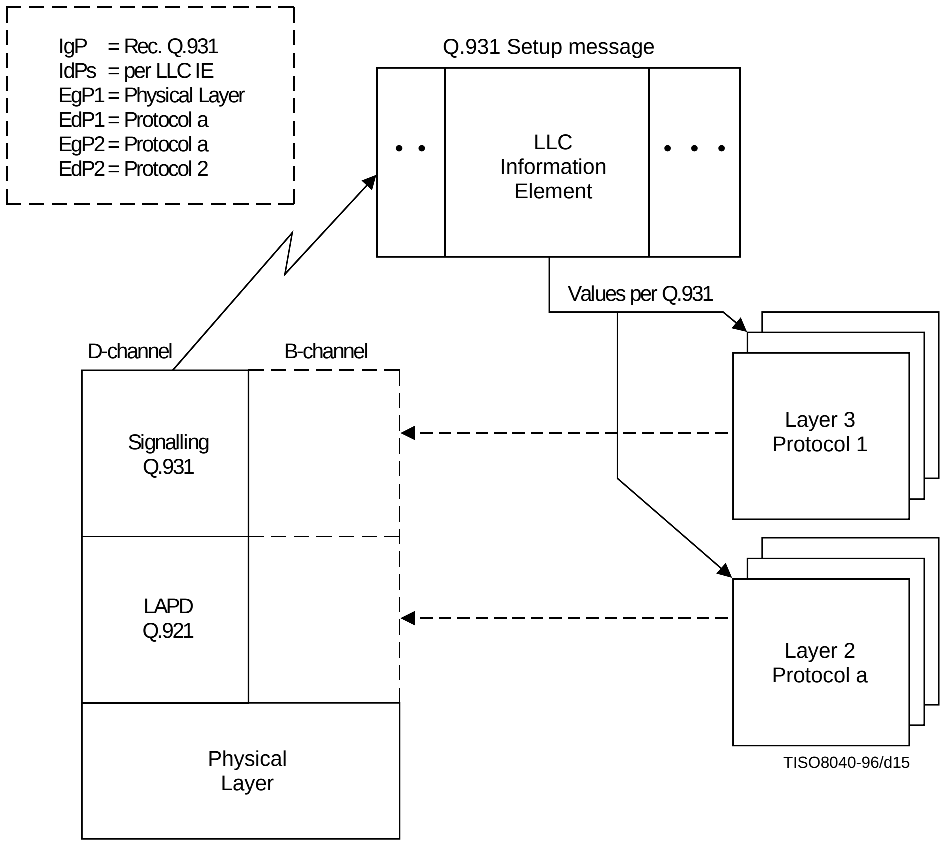


Figure B.2 – Q.931 Identification by Low Layer Compatibility (LLC) Information Element in ISDN

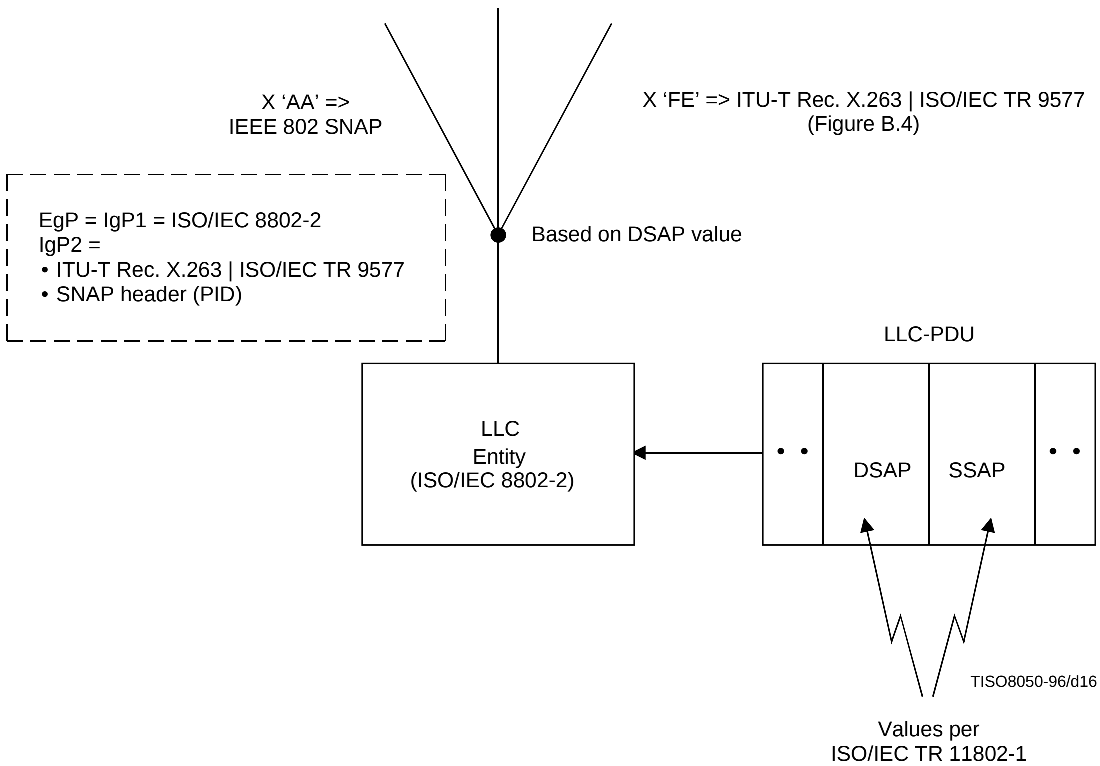


Figure B.3 – Identification by Logical Link Control Address in LANs

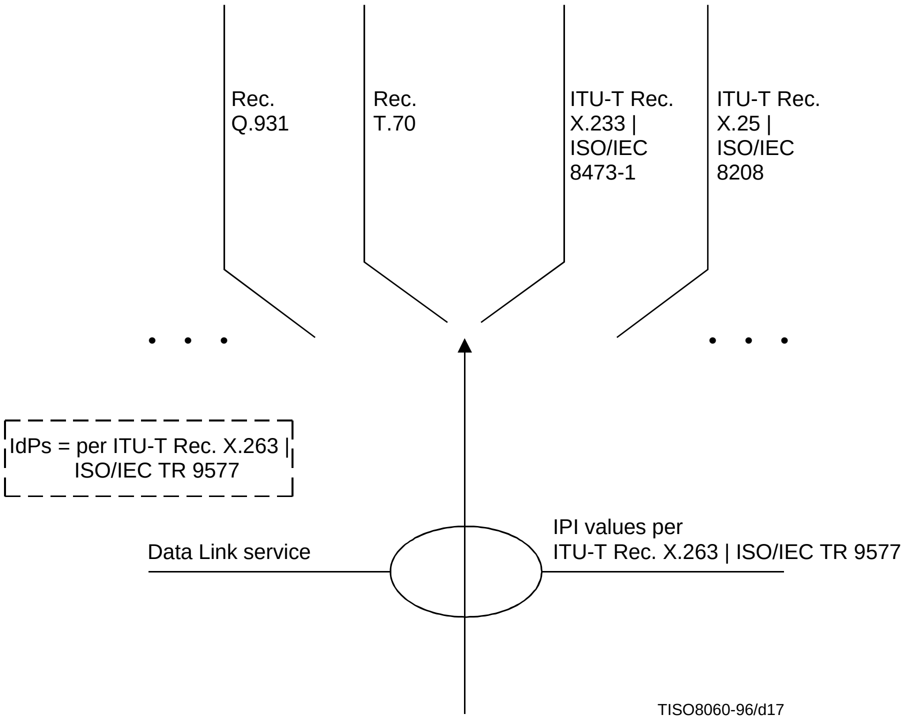


Figure B.4 – ITU-T Rec. X.263 | ISO/IEC TR 9577 IPI

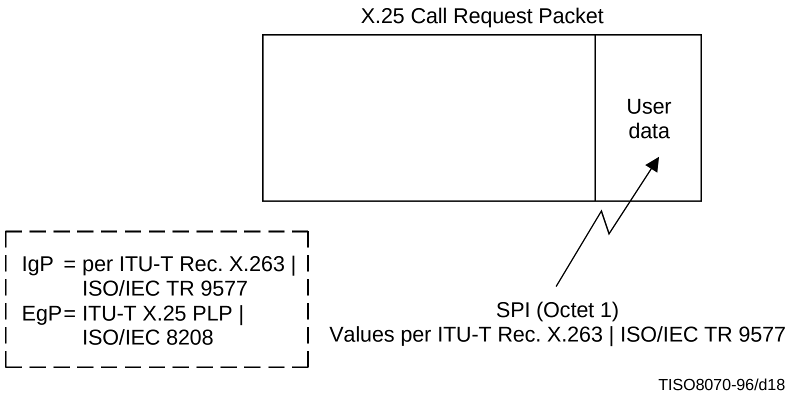


Figure B.5 – ITU-T Rec. X.263 ISO/IEC TR 9577 SPI in X.25 call request

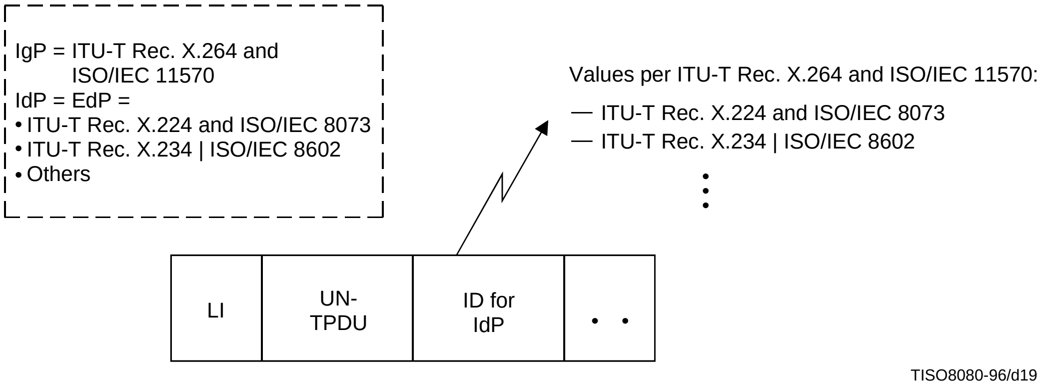


Figure B.6 – ITU-T Rec. X.264 and ISO/IEC 11570 identification of Transport Protocols

ITU-T RECOMMENDATIONS SERIES

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