INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.227 *bis*

(09/98)

SERIES X: DATA NETWORKS AND OPEN SYSTEM COMMUNICATIONS

Open Systems Interconnection – Connection-mode protocol specifications

---

Information technology – Open Systems Interconnection – Connection-mode protocol for the Application Service Object Association Control Service Element

ITU-T Recommendation X.227 *bis*

(Previously CCITT Recommendation)

ITU-T X-SERIES RECOMMENDATIONS

DATA NETWORKS AND OPEN SYSTEM COMMUNICATIONS

PUBLIC DATA NETWORKS	
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 SYSTEMS INTERCONNECTION	
Model and notation	X.200–X.209
Service definitions	X.210–X.219
Connection-mode protocol specifications	**X.220–X.229**
Connectionless-mode protocol specifications	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	
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	
Networking	X.600–X.629
Efficiency	X.630–X.639
Quality of service	X.640–X.649
Naming, Addressing and Registration	X.650–X.679
Abstract Syntax Notation One (ASN.1)	X.680–X.699
OSI MANAGEMENT	
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 and ODMA functions	X.730–X.799
SECURITY	X.800–X.849
OSI APPLICATIONS	
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.

INTERNATIONAL STANDARD 15954

ITU-T RECOMMENDATION X.227 *bis*

INFORMATION TECHNOLOGY – OPEN SYSTEMS INTERCONNECTION – CONNECTION-MODE PROTOCOL FOR THE APPLICATION SERVICE OBJECT ASSOCIATION CONTROL SERVICE ELEMENT

Summary

This Recommendation | International Standard specifies the connection-mode protocol for the application service element for ASO-associated control: the Association Control Service Element (ACSE).

Source

The ITU-T Recommendation X.227 *bis* was approved on the 25th of September 1998. The identical text is also published as ISO/IEC International Standard 15954.

FOREWORD

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

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

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

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

NOTE

In this Recommendation the term *recognized operating agency (ROA)* includes any individual, company, corporation or governmental organization that operates a public correspondence service. The terms *Administration*, *ROA* and *public correspondence* are defined in the *Constitution of the ITU (Geneva, 1992).*

INTELLECTUAL PROPERTY RIGHTS

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

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

CONTENTS

Introduction

This Recommendation | International Standard is one of a set of ITU-T Recommendations | International Standards produced to facilitate the interconnection of information processing systems. It is related to other ITU-T Recommendations and International Standards in the set as defined by the Reference Model for Open Systems Interconnection (see ITU-T Rec. X.200 | ISO/IEC 7498-1). The Reference model subdivides the area of standardization for interconnection into a series of layers of specification, each of manageable size.

The goal of Opens Systems Interconnection is to allow, with a minimum of technical agreement outside the interconnection standards, the interconnection of information processing systems:

- from different manufacturers;
- under different managements;
- of different levels of complexity; and
- of different technologies.

This Recommendation | International Standard specifies the connection-mode protocol for the application service element for ASO-association control: the Association Control Service Element (ACSE). The protocol for ACSE connectionless mode service (A-UNIT-DATA) is specified in ITU-T Rec. X.237 *bis* | ISO/IEC 15955. The ACSE provides services for establishing and releasing associations. The ACSE protocol includes three optional functional units. One functional unit supports the exchange of information in support of authentication during association establishment. The second functional unit supports the negotiation of ASO-context during association establishment. The optional Higher Level Association functional unit provides for the facility to identify ASO-associations and transparently pass data to child ASOs and allows the ASO-context or the presentation context on an ASO-association to be modified during the lifetime of the association.

The fast-associate mechanism allows a session connection, including its embedded presentation connection and application association, to be established using a compressed form of the information that would otherwise be sent on the S-CONNECT exchange. The compressed form, called the upper layer context identifier, is a reference to an upper-layer context specification, which is a definition of the fields of the application, ACSE, presentation, and session protocols that would be sent on the full-form connect messages. The upper-layer context identifier may be parameterized to include values for variable fields allowed by the full form protocols for the upper layers.

Within the ACSE protocol, the addition is the definition of the construction of the User summary parameter of the P-CONNECT primitives from the semantics of the AARQ fields and the User summary parameter of the corresponding A-ASSOCIATE primitive.

This Recommendation | International Standard maintains compatibility with earlier editions of ACSE. This Recommendation | International Standard does not support X.410 mode nor Session Version 1.

This Recommendation | International Standard includes an annex that describes the protocol machine of ACSE in terms of a state table. This protocol machine is referred to as the Association Control Protocol Machine (ACPM).

The protocol defined in this Recommendation | International Standard is also governed by the use of the Presentation service (see ITU-T Rec. X.216 | ISO/IEC 8822).

INTERNATIONAL STANDARD

ITU-T RECOMMENDATION

INFORMATION TECHNOLOGY – OPEN SYSTEMS INTERCONNECTION – CONNECTION-MODE PROTOCOL FOR THE APPLICATION SERVICE OBJECT ASSOCIATION CONTROL SERVICE ELEMENT

1 Scope

The ACSE supports two modes of communication: connection-mode and connectionless-mode. The ACSE service definition (see ITU-T Rec. X.217 *bis* | ISO/IEC 15953) includes both modes of communication. This Recommendation | International Standard only includes the connection mode of communication. This Recommendation | International Standard for the connectionless mode of communication is contained in ITU-T Rec. X.237 *bis* | ISO/IEC 15955.

This Recommendation | International Standard defines procedures that are applicable to instances of communication between systems which wish to interconnect in an open systems interconnection environment in a connection mode. This Recommendation | International Standard includes the Kernel functional unit that is used to establish and release ASO-associations. The Authentication functional unit provides additional facilities for exchanging information in support of authentication during association establishment without adding new services. The ACSE authentication facilities can be used to support a limited class of authentication methods. The ASO-context negotiation functional unit provides the additional facility for the recipient to select the ASO-context from a list offered by the initiator during association establishment. The optional Higher Level Association functional unit provides for the facility to identify ASO-associations and transparently pass data to child ASOs and allows the ASO-context or the presentation context on an ASO-association to be modified during the lifetime of the association.

This Recommendation | International Standard specifies:

a) procedures for the transfer of information for ASO-association control and the authentication of ASOs and application-entities; and

b) the abstract syntax for the representation of the ACSE APDUs.

The ACSE procedures are defined in terms of:

a) the interactions among peer ACSE protocol machines through the use of Presentation services or supporting ACSE services; and

b) the interaction between an ACSE protocol machine and its service-user.

This Recommendation | International Standard also specifies conformance requirements for systems implementing these procedures. It does not contain tests which can be used to demonstrate conformance.

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 this 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 edition 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 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.207 (1993) | ISO/IEC 9545:1994, *Information technology – Open Systems Interconnection – Application layer structure.*

- ITU-T Recommendation X.210 (1993) | ISO/IEC 10731:1994, *Information technology – Open Systems Interconnection – Basic Reference Model: Conventions for the definition of OSI services.*
- ITU-T Recommendation X.215 (1995) | ISO/IEC 8326:1996, *Information technology – Open Systems Interconnection – Session service definition.*
- ITU-T Recommendation X.215 (1995)/Amd.1 (1997) | ISO/IEC 8326:1996/Amd.1:1998, *Information technology – Open Systems Interconnection – Session service definition – Amendment 1: Efficiency enhancements.*
- ITU-T Recommendation X.215 (1995)/Amd.2 (1997) | ISO/IEC 8326:1996/Amd.2:1998, *Information technology – Open Systems Interconnection – Session service definition – Amendment 2: Nested connections functional unit.*
- ITU-T Recommendation X.216 (1994) | ISO/IEC 8822:1994, *Information technology – Open Systems Interconnection – Presentation service definition.*
- ITU-T Recommendation X.216 (1994)/Amd.1 (1997) | ISO/IEC 8822:1994/Amd.1:1998, *Information technology – Open Systems Interconnection – Presentation service definition – Amendment 1: Efficiency enhancements.*
- ITU-T Recommendation X.216 (1994)/Amd.2 (1997) | ISO/IEC 8822:1994/Amd.2:1998, *Information technology – Open Systems Interconnection – Presentation service definition – Amendment 2: Nested connections functional unit.*
- ITU-T Recommendation X.217 *bis* (1998) | ISO/IEC 15953:1999, *Information technology – Open Systems Interconnection – Service definition for the application service object – Association control service element.*
- ITU-T Recommendation X.225 (1995) | ISO/IEC 8327-1:1996, *Information technology – Open Systems Interconnection – Connection-oriented session protocol: Protocol specification.*
- ITU-T Recommendation X.225 (1995)/Amd.1 (1997) | ISO/IEC 8327-1:1996/Amd.1:1998, *Information technology – Open Systems Interconnection – Connection-oriented session protocol: Protocol specification – Amendment 1: Efficiency enhancements.*
- ITU-T Recommendation X.225 (1995)/Amd.2 (1998) | ISO/IEC 8327-1:1996/Amd.2:1998, *Information technology – Open Systems Interconnection – Connection-oriented session protocol: Protocol specification – Amendment 2: Nested connections functional unit.*
- ITU-T Recommendation X.226 (1994) | ISO/IEC 8823-1:1994, *Information technology – Open Systems Interconnection – Connection-oriented presentation protocol: Protocol specification.*
- ITU-T Recommendation X.226 (1994)/Amd.1 (1997) | ISO/IEC 8823-1:1994/Amd.1:1998, *Information technology – Open Systems Interconnection – Connection-oriented presentation protocol: Protocol specification – Amendment 1: Efficiency enhancements.*
- ITU-T Recommendation X.226 (1994)/Amd.2 (1997) | ISO/IEC 8823-1:1994/Amd.2:1998, *Information technology – Open Systems Interconnection – Connection-oriented presentation protocol: Protocol specification – Amendment 2: Nested connections functional unit.*
- ITU-T Recommendation X.237 *bis* (1998) | ISO/IEC 15955:1999, *Information technology – Open Systems Interconnection – Connectionless protocol for the application service Object-Association control service element.*
- ITU-T Recommendation X.501 (1993) | ISO/IEC 9594-2:1995, *Information technology – Open Systems Interconnection – The Directory: Models.*
- ITU-T Recommendation X.650 (1996) | ISO/IEC 7498-3:1997, *Information technology – Open Systems Interconnection – Basic Reference Model: Naming and addressing.*
- CCITT Recommendation X.660 (1992) | ISO/IEC 9834-1:1993, *Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: General procedures.*
- CCITT Recommendation X.665 (1992) | ISO/IEC 9834-6 (1993), *Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: Application processes and application entities.*
- ITU-T Recommendation X.680 (1994) | ISO/IEC 8824-1:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.*

- ITU-T Recommendation X.680 (1994)/Amd.1 (1995) | ISO/IEC 8824-1:1995/Amd.1:1996 *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation – Amendment 1: Rules of extensibility*.
- ITU-T Recommendation X.681 (1994) | ISO/IEC 8824-2:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification.*
- ITU-T Recommendation X.681 (1994)/Amd.1 (1995) | ISO/IEC 8824-2:1995/Amd.1:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification – Amendment 1: Rules of extensibility*.
- ITU-T Recommendation X.682 (1994) | ISO/IEC 8824-3:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification.*
- ITU-T Recommendation X.683 (1994) | ISO/IEC 8824-4:1995, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications.*
- ITU-T Recommendation X.691 (1995) | ISO/IEC 8825-2:1996, *Information technology – ASN.1 Encoding Rules: Specification of Packed Encoding Rules (PER).*

2.2 Paired Recommendations | International Standards equivalent in technical content

- CCITT Recommendation X.209 (1988), *Specification of basic encoding rules for Abstract Syntax Notation One (ASN.1).*

  ISO/IEC 8825:1990, *Information technology – Open Systems Interconnection – Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1).*
- CCITT Recommendation X.800 (1991), *Security architecture for Open Systems Interconnection for CCITT applications.*

  ISO 7498-2:1989, *Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture*.

2.3 Additional references

- ISO 6523:1984, *Data interchange – Structures for the identification of organizations.*

3 Definitions

3.1 Reference Model definitions

3.1.1 Basic Reference Model definitions

This Recommendation | International Standard is based on the concepts developed in ITU-T Rec. X.200 | ISO/IEC 7498-1 and makes use of the following terms defined in it:

a) Application Layer;

b) application-process;

c) application-entity;

d) application-service-element;

e) application-protocol-data-unit;

f) application-protocol-control-information;

g) presentation-service;

h) presentation-connection;

i) concrete syntax;

j) session-service;

k) session-protocol; and

l) session-connection.

3.1.2 Security architecture definitions

This Recommendation | International Standard makes use of the following term defined in CCITT Rec. X.800 | ISO 7498-2:

– password.

3.1.3 Naming and addressing definitions

This Recommendation | International Standard makes use of the following terms defined in ITU-T Rec. X.650 | ISO/IEC 7498-3:

a) application-process title;

b) application-entity qualifier;

c) application-entity title;[1)]

d) application-process invocation-identifier;

e) application-entity invocation-identifier; and

f) presentation address.

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;

c) confirmed service;

d) non-confirmed service;

e) provider-initiated service;

f) primitive;

g) request (primitive);

h) indication (primitive);

i) response (primitive);

j) confirm (primitive);

k) submit (primitive); and

l) deliver (primitive).

3.3 Presentation service definitions

This Recommendation | International Standard makes use of the following terms defined in ITU-T Rec. X.216 | ISO/IEC 8822:

a) abstract syntax;

b) abstract syntax name;

c) default context;

d) defined context set;

e) functional unit [presentation];

f) presentation context;

g) presentation data value.

---

1) As defined in ITU-T Rec. X.650 | ISO/IEC 7498-3, an application-entity title is composed of an application-process title and an application-entity qualifier. The ACSE protocol provides for the transfer of an application-entity title value by the transfer of its component values.

3.4 Application Layer Structure definitions

This Recommendation | International Standard makes use of the following terms defined in ITU-T Rec. X.207 | ISO/IEC 9545:

a) ASO-context;
b) application-entity invocation;
c) control function;
d) application-service-object (ASO);
e) ASO-association;
f) ASO-association-identifier;
g) ASO-invocation;
h) ASOI-identifier;
i) ASOI-tag;
j) ASO-name;
k) ASO-qualifier;
l) ASO-title;
m) child ASO; and
n) parent ASO.

3.5 ACSE service definitions

This Recommendation | International Standard makes use of the following terms defined in ITU-T Rec. X.217 *bis* | ISO/IEC 15953:

a) Association Control Service Element;
b) ACSE service-user;
c) ACSE service-provider;
d) requestor;
e) acceptor;
f) association-initiator;
g) association-responder;
h) authentication;
i) authentication-function;
j) authentication-value;
k) authentication-mechanism;
l) disrupt;
m) establishment phase; and
n) data transfer phase.

3.6 Association Control protocol specification definitions

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

3.6.1 Association Control Protocol Machine: The protocol machine for the Association Control Service Element specified in this Recommendation | International Standard.

3.6.2 requesting Association Control Protocol Machine: The Association Control Protocol Machine whose service-user is the requestor of a particular Association Control Service Element service.

3.6.3 accepting Association Control Protocol Machine: The Association Control Protocol Machine whose service-user is the acceptor for a particular Association Control Service Element service.

3.6.4 external event [to an ASE]: A service primitive that is not directly referenced by an ASE but that may disrupt a service procedure of the ASE.

4 Abbreviations

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

4.1 Data units

APDU application-protocol-data-unit

4.2 Types of application-protocol-data-units

The following abbreviations have been given to the application-protocol-data-units defined in this Recommendation | International Standard:

AARE A-ASSOCIATE-RESPONSE APDU

AARQ A-ASSOCIATE-REQUEST APDU

ABRT A-ABORT APDU

ACRP A-ALTER-CONTEXT-RESPONSE APDU

ACRQ A-ALTER-CONTEXT-REQUEST APDU

A-DT A-DATA APDU

RLRE A-RELEASE-RESPONSE APDU

RLRQ A-RELEASE-REQUEST APDU

4.3 Other abbreviations

The following abbreviations are also used in this Recommendation | International Standard:

ACPM Association Control Protocol Machine

ACSE Association Control Service Element

AE application-entity

AEI application-entity invocation

Amd. Amendment of an ITU-T Recommendation and of an International Standard

AP application-process

APCI application-protocol-control-information

ASE application-service-element

ASN.1 Abstract Syntax Notation One

ASO application-service-object

ASOI ASO-invocation

CF control function

cnf confirm primitive

ind indication primitive

IEC International Electrotechnical Commission

ISO International Organization for Standardization

ITU-T International Telecommunications Union – Telecommunications Standardization Sector

OSI Open Systems Interconnection

QoS Quality of Service

Rec. Recommendation [ITU-T]

req request primitive

ROA Recognized Operating Agency

5 Conventions

This Recommendation | International Standard employs a tabular presentation of its APDU fields. In clause 7, tables are presented for each ACSE APDU. Each field is summarized using the following notation:

ACPM	Source or sink is the ACPM
cnf	Sink is related confirm primitive
ind	Sink is related indication primitive
M	Presence is mandatory
O	Presence is ACPM option
req	Source is related request primitive
rsp	Source is related response primitive
U	Presence is ACSE service-user option

The structure of each ACSE APDU is specified in clause 9 using the abstract syntax notation of ASN.1 (see ITU-T Rec. X.680 | ISO/IEC 8824-1).

6 Overview of the protocol

6.1 Service provision

The protocol in this Recommendation | International Standard provides the connection-mode services defined in ITU-T Rec. X.217 *bis* | ISO/IEC 15953. Both the connection-mode and connectionless-mode services are listed in Table 1. The protocol for the connectionless A-UNIT-DATA service is specified in ITU-T Rec. X.237 *bis* | ISO/IEC 15955.

Table 1 – ACSE services

Communication mode	Service	Type
Connection	A-ASSOCIATE	Confirmed
	A-RELEASE	Confirmed
	A-ABORT	Non-confirmed
	A-P-ABORT	Provider-initiated
	A-DATA	Non-confirmed
	A-ALTER-CONTEXT	Confirmed
Connectionless	A-UNIT-DATA	Non-confirmed

6.2 Functional units

Functional units are used by this Recommendation | International Standard to negotiate ACSE user requirements during association establishment. Four functional units are defined:

a) Kernel functional unit;

b) Authentication functional unit;

c) ASO-context negotiation functional unit; and

d) Higher Level Association functional unit.

The ACSE requirements fields on the AARQ and AARE APDUs are used to select the functional units for the association. The Kernel functional unit is always available. It is the default functional unit. To be included, the Authentication functional unit, ASO-context negotiation functional unit, and Higher Level Association functional unit shall be explicitly requested on the AARQ APDU and accepted on the AARE APDU.

The selection of the Authentication functional unit supports additional fields on the AARQ, AARE, and RLRQ APDUs. The selection of the ASO-Context negotiation functional unit supports an additional field on the AARQ. Neither functional unit affects the elements of procedure. The optional Higher Level Association functional unit provides for the facility to identify ASO-associations and transparently pass data to child ASOs and allows the ASO-context or the

presentation context on an ASO-association to be modified during the lifetime of the association. Note that some higher level associations may have a scope that precludes the use of Session Functional Units other than Kernel and Full Duplex.

For details on the use of Higher Level Association functional units, see Annexes C, D, and E. Table 2 shows the services, APDUs and APDU fields associated with the ACSE functional units.

Table 2 – Functional Unit APDUs and their fields

Functional Unit	Service	APDU	Field Name
Kernel	A-ASSOCIATE	AARQ	Protocol Version ASO-context-name Calling AP-title Calling AE-qualifier Calling AP-invocation-identifier Calling AE-invocation-identifier Called AP-title Called AE-qualifier Called AP-invocation-identifier Called AE-invocation-identifier Implementation Information User-information Called ASOI-Tag Calling ASOI-Tag Presentation contex definition list
		AARE	Protocol Version ASO-context-name Responding AP-title Responding AE-qualifier Responding AP-invocation-identifier Responding AE-invocation-identifier Result Result source – Diagnostic Implementation Information User-information Responding ASOI-Tag Presentation context result list
	A-RELEASE	RLRQ	Reason ASO-qualifier ASOI-identifier User-information
		RLRE	Reason ASO-qualifier ASOI-identifier User-information

Table 2 *(concluded)*

Functional Unit	Service	APDU	Field Name
		ABRT	Reason ASO-qualifier ASOI-identifier User-information
Authentication	A-ASSOCIATE	AARQ	ACSE requirements Authentication-mechanism Name Authentication-value
		AARE	ACSE requirements Authentication-mechanism Name Authentication-value
	A-ABORT	ABRT	Diagnostic
ASO-context-negotiation	A-ASSOCIATE	AARQ	ASO-context-name-list ACSE requirements
	A-ASSOCIATE	AARE	ASO-context-name-list ACSE requirements
Higher Level Association	A-DATA	A-DT	ASO-qualifier ASOI-identifier User data
	A-ALTER-CONTEXT	ACRQ	ASO-qualifier ASOI-identifier ASO-context P-context User data
		ACRP	ASO-qualifier ASOI-identifier ASO-context P-context User data

6.3 Use of supporting services

There are two distinct cases that must be considered in mapping ACSE to a supporting service:

1) ACSE may use a Presentation Connection as a supporting service; or

2) ACSE may use an ASO-association that is not an application-association, i.e. an ASO-association is being established over another ASO-association (which may be an application association) or any equivalent supporting service.

ACSE defines a service definition for a generic supporting service (called the IA-service, see clause 8) to define its mapping to a supporting service[2]. Annexes to this Recommendation | International Standard specify mappings of this IA-service to specific supporting services, e.g. the Presentation service and the ACSE service. Other standards may define other mappings as required.

In the connectionless mode, the A-UNIT-DATA APDU may be mapped to either a connection-mode or connectionless-mode supporting service. If the higher-level functional unit is selected, all connection-mode ACSE APDUs can be mapped to the A-UNIT-DATA service primitives.

6.4 Model

The Association Control Protocol Machine (ACPM) is modeled as a finite state machine whose specification is given in this Recommendation | International Standard. The ACPM communicates with its service-user by means of the ACSE service primitives defined in ITU-T Rec. X.217 *bis* | ISO/IEC 15953. The ACPM communicates with its supporting service-provider by means of either the ACSE service or Presentation services defined in ITU-T Rec. X.216 | ISO/IEC 8822.

NOTE 1 – An application ASE or ASO specification that references ACSE need not specify the use of ACSE service primitive parameters that are irrelevant to its operation. The control function (CF) that references ACSE can be modeled to pass such parameters between the ACPM and that part of the ASOI to which the parameters are relevant.

The ACPM is driven by the receipt of input events from its ACSE service-user and from its service-provider that supports the association. The input events from the ACSE service-user are ACSE request and response primitives.

The ACPM responds to input events by issuing output events to its service-provider and to its ACSE service-user. The output events to its ACSE service-user are ACSE indication and confirm primitives.

The receipt of an input event, the generation of dependent actions, and the resultant output event are considered to be an indivisible action.

During the establishment of an association among ASOs, the existence of invocations of the requesting and responding ASOs is presumed. How they are created is outside of the scope of this Recommendation | International Standard.

A new invocation of an ACPM is employed upon the receipt of an A-ASSOCIATE request/indication primitive or a IA-BIND-REQUEST.deliver primitive. Each such invocation controls exactly one association.

NOTE 2 – Each association may be identified in an end system by a local mechanism so that the ACSE service-user and the ACPM can refer to the association.

When the ACPM is a component of the outermost ASO, an ACPM communicates with its peer ACPM in support of an association by transferring ACSE Application Protocol Data Units (APDUs) defined in clause 9. An ACSE APDU is transferred as the User data parameter of an A-DATA primitive or a value in the User data parameter of a service primitive of the supporting service.

6.5 User summary mechanism

If the fast associate mechanism is used during association establishment, the initiating ACPM as well as forming an AARQ APDU to be passed to the Presentation service-provider in the User data parameter of a P-CONNECT request, also passes the semantic content of the AARQ in the User summary parameter of the P-CONNECT request. The User summary parameter references an Upper-Layer Context specification and is a purely abstract parameter. If the A-ASSOCIATE request User-information parameter was present, the semantic content of this will have been supplied to the ACPM in the User summary parameter of the A-ASSOCIATE request, and is conceptually included in the User summary parameter of the P-CONNECT request.

If the Presentation provider (via the Session service and protocol) makes use of the fast associate mechanism, the responding ACPM will receive only the User summary parameter on the P-CONNECT indication, and not the User data. The responding implementation will reconstruct the semantic content of the AARQ that would have been present in the P-CONNECT User data, and issue an A-ASSOCIATE indication with a User summary parameter in place of its User data.

---

[2] It is recommended that ASO designers provide this kind of supporting service definition for their ASO and define the mapping to a specific supporting service definition separately (rather than specifying a mapping to an existing service definition). This approach emphasizes that the ASO can be used with any supporting service that provides an equivalent service. This facilitates the re-use of the ASO and makes it easier for subsequent designers to utilize this ASO with alternate supporting services. Of course, if the designer's intent is that the ASO can only be supported by a specific supporting service, then it should reference that service definition.

Similarly, the responding ACPM will form a User summary parameter on the P-CONNECT response from the AARE APDU, including the semantic content of the User summary parameter of the A-ASSOCIATE response (if present) by reference to the same Upper-Layer Context specification. The initiating ACPM reconstructs the AARE.

NOTE – The passing of the User summary parameters and reconstruction of the ACSE APDUs from the Presentation User summary parameters is abstract. There is no requirement for a real implementation to perform these actions.

7 Elements of procedure

The ACSE protocol consists of the following procedures:

a) association establishment;

b) normal release of an association;

c) abnormal release of an association;

d) A-DATA; and

e) ALTER CONTEXT.

In this clause, a summary of each of these elements of procedure is presented. This consists of a summary of the relevant APDUs, and a high-level overview of the relationship between the ACSE services, the APDUs involved, and the mapping to the IA-service. Clause 8 describes the IA-service definition, which defines the abstract service that ACSE requires of any supporting service. In clause 9, a detailed specification of the ACSE APDUs is given using the notation of ASN.1 (see ITU-T Rec. X.680 | ISO/IEC 8824-1). Annex A presents the state table for the ACPM. Annex B specifies a simple password authentication-mechanism. Annex C provides the specific mapping of the IA-service to the Presentation service. Annex D provides the specific mapping of the IA-service to the ACSE Service. Annex E provides guidance on the use of the Higher Level Association functional units.

7.1 Association establishment

7.1.1 Purpose

The association establishment procedure is used to establish an association among ASOs. It supports the A-ASSOCIATE service.

7.1.2 APDUs used

The association establishment procedure uses the A-ASSOCIATE-REQUEST (AARQ) and the A-ASSOCIATE-RESPONSE (AARE) APDUs. The fields of the AARQ APDU are listed in Table 3. The fields of the AARE APDU are listed in Table 4.

7.1.3 Association establishment procedure

This procedure is driven by the following events:

a) invocation of an A-ASSOCIATE request primitive from the requestor;

b) receipt of an AARQ APDU;

c) receipt of a User summary parameter on a P-CONNECT indication primitive;

d) invocation of an A-ASSOCIATE response primitive from the acceptor; and

e) receipt of an AARE APDU or a User summary parameter.

7.1.3.1 A-ASSOCIATE request primitive

The requesting ACPM forms an AARQ APDU from parameter values of the A-ASSOCIATE request primitive and the protocol version and optionally implementation information. The AARQ PDU is mapped to the User-information parameter of the IA-BIND-REQUEST.submit primitive. The ACPM transitions to the Awaiting AARE (STA1) state.

If the fast-associate mechanism is supported, the requesting ACPM identifies the semantic content of the AARQ, including the User data, in the User summary parameter of the P-CONNECT request.

7.1.3.2 AARQ APDU

The accepting ACPM receives an AARQ APDU as the User-information parameter of an IA-BIND-REQUEST.deliver primitive or reconstructs an AARQ APDU from the User summary parameter of the P-CONNECT indication primitive.

The ACPM determines if the AARQ APDU is acceptable based on the rules for extensibility (see 7.6). If the AARQ APDU is not acceptable, a protocol error results (see 7.3.3.4). The association establishment procedure is disrupted. An A-ASSOCIATE indication primitive is not issued. An AARE APDU is formed indicating the error. This APDU is mapped to the User-information parameter of an IA-BIND-RESPONSE.submit primitive, which is invoked. The ACPM transitions to the Idle-Unassociated (STA0) state.

Otherwise, the ACPM next inspects the value of the Protocol Version field of the AARQ APDU. If the ACPM does not support a common protocol version, it forms an AARE APDU with the following assigned fields:

a) Protocol Version field (optional) with the value that indicates the protocol version(s) that it could support (see 7.1.5.1);

b) ASO-context-name field with the same value as on the AARQ APDU;

c) Result field with the value "rejected (permanent);" and source – Diagnostic field with the values "ACSE service-provider" and "no common ACSE version."

In this case, the ACPM sends the AARE APDU. The ACPM does not issue an A-ASSOCIATE indication primitive. The AARE is mapped to the User-information parameter of the IA-BIND-RESPONSE.submit primitive and it is invoked. The association is not established and the ACPM is in the Idle-Unassociated (STA0) state.

If the AARQ APDU is acceptable, the ACPM performs the following actions.

If the ACSE requirements field is present, it forms an ACSE requirements field for the AARE APDU that is the intersection of the functional units requested in the AARQ and those that it supports.

If the ASO-context field or Presentation context definition list are present, it forms the proper result parameters for the AARE.

The ACPM then issues an A-ASSOCIATE indication primitive to the acceptor, and transitions to the Awaiting A-ASCrsp (STA2) state.

7.1.3.3 A-ASSOCIATE response primitive

When the accepting ACPM receives the A-ASSOCIATE response primitive, the Result parameter specifies whether the service-user has accepted or rejected the association. The ACPM forms an AARE APDU using the A-ASSOCIATE response primitive parameters and parameters received in the AARQ. The ACPM sets the Result source – Diagnostic field to "ACSE service-user" and the value derived from the Diagnostic parameter of the response primitive. The AARE APDU is mapped to the User-information parameter of an IA-BIND-RESPONSE.submit primitive.

Table 3 – AARQ APDU parameters

Field name	Presence	Source	Sink
Protocol Version	M	ACPM	ACPM
ASO-context-name	U	req	ind
ASO-context-name-list	U	req	ind
Calling AP-Title	U	req	ind
Calling AE qualifier	U	req	ind
Calling AP-invocation-identifier	U	req	ind
Calling AE-invocation-identifier	U	req	ind
Called AP-title	U	req	ind
Called AE-qualifier	U	req	ind
Called AP-invocation-identifier	U	req	ind
Called AE-invocation-identifier	U	req	ind
ACSE requirements	U	req	ind
Authentication-mechanism Name	U	req	ind

Table 3 *(concluded)*

Field name	Presence	Source	Sink
Authentication-value	U	req	ind
Implementation Information	O	ACPM	ACPM
User-information	U	req	ind
Called ASOI-tag	U	req	ind
Calling ASOI-tag	U	req	ind
Presentation context definition list	C	req	ind
NOTE 1 – The Authentication-mechanism Name and Authentication-value fields are only present if the ACSE requirements field includes the Authentication functional unit. NOTE 2 – The ASO-context-name-List field is only present if the ACSE requirements field includes the ASO-context negotiation functional unit. The value of the ASO-context-name parameter may be different from any of the names in the ASO-context-name-list parameter or it may be equal to one of the names in the list.			

If the fast-associate mechanism is supported, the accepting ACPM identifies the semantic content of the AARE, including the User data, in the User summary parameter of the P-CONNECT response, by reference to the Upper-Layer Context specification identified by the User summary parameter of the received P-CONNECT indication.

If the acceptor accepted the association request, the Result field of the outgoing AARE APDU specifies "accepted". The association is established and the ACPM transitions to the Associated (STA5) state.

If the acceptor rejected the association request, the Result field of the AARE APDU contains the appropriate rejection value. The association is not established and the ACPM issues an IA-UNBIND.request primitive and transitions to the Idle-Unassociated (STA0) state.

7.1.3.4 AARE APDU

The requesting ACPM receives an AARE APDU as the User-information parameter of an IA-BIND-RESPONSE.deliver primitive. The following situations are possible:

a) the association has been accepted;

b) the accepting ACPM or the acceptor has rejected the association; or

c) the supporting service-provider has rejected the request for the underlying service.

Table 4 – AARE APDU Fields

Field name	Presence	Source	Sink
Protocol Version	O	ACPM	ACPM
ASO-context-name	U	rsp	cnf
ASO-context-name-list	U	req	ind
Responding AP-title	U	rsp	cnf
Responding AE-qualifier	U	rsp	cnf
Responding AP-invocation-identifier	U	rsp	cnf
Responding AE-invocation-identifier	U	rsp/ACPM	cnf
Result	O	rsp/ACPM	cnf
Result source – Diagnostic	O	rsp	cnf
ACSE Requirements	U	req	ind
Authentication-mechanism Name	U	req	ind
Authentication-value	U	req	ind

Table 4 *(concluded)*

Field name	Presence	Source	Sink
Implementation Information	O	ACPM	ACPM
User-information	U	rsp	cnf
Responding ASOI-tag	U	req	ind
Presentation context result identifier	C	rsp	cnf
NOTE 1 – The authentication-mechanism and authentication-value fields are only present if the ACSE requirements field includes the Authentication functional unit. NOTE 2 – The ASO-context-name-list field is only present if the ACSE requirements field includes the ASO-context negotiation functional unit and the result = rejected. The value of the ASO-context-name parameter may be different from any of the names in the ASO-context-name-list parameter or it may be equal to one of the names in the list. NOTE 3 – This ASO-context field is optional. If backward compatibility with older implementations of ACSE is desired, it must be present.			

If the association was accepted, the Result field of the AARE APDU specifies "accepted". If it exists, the User summary parameter is a value from which the requesting ACPM can reconstruct the AARE APDU. The requesting ACPM issues an A-ASSOCIATE confirm primitive to the requestor derived from parameters from the supporting service and the AARE APDU. The A-ASSOCIATE confirm primitive Result parameter specifies "accepted". The association is established and the ACPM transitions to the Associated (STA5) state.

If the association was rejected by either the accepting ACPM or by the acceptor, the AARE APDU result parameter specifies "user-rejection". The requesting ACPM issues an A-ASSOCIATE confirm primitive to the requestor derived from parameters from the supporting service and the AARE APDU. The A-ASSOCIATE confirm primitive Result parameter indicates "rejected (transient)" or "rejected (permanent)". The Result source parameter indicates "ACSE service-user" or "ACSE service-provider". The association is not established, and the ACPM issues an IA-UNBIND-REQUEST.submit primitive and transitions to the Idle-Unassociated (STA0) state.

If the request was rejected by the supporting service-provider, the underlying confirm primitive Result parameter specifies "provider-rejection". The requesting ACPM issues an A-ASSOCIATE confirm primitive with the Result parameter indicating "rejected (permanent)". The Result source parameter indicates "supporting service-provider". The association is not established, and the ACPM issues an IA-UNBIND request primitive and transitions to the Idle-Unassociated (STA0) state.

7.1.4 Use of the AARQ APDU fields

The AARQ APDU fields are used by the requesting and accepting ACPMs as specified below.

7.1.4.1 Protocol Version

For the requesting ACPM: the value assigned to this field is determined within the implementation of the ACPM. It is a bit string where each bit that is set to one indicates the version of ACSE protocol that this ACPM supports. Bit 0 represents version 1; bit 1 represents version 2, etc. Multiple bits may be set indicating support of multiple versions. No trailing bits higher than the highest version of this Recommendation | International Standard that the requesting ACPM supports are included. That is, the last bit of the string is set to one.

For the accepting ACPM: the ACPM ignores trailing bits of this field that are higher than the one indicating the latest version of this Recommendation | International Standard that it supports.

7.1.4.2 ASO-context-name

For the requesting ACPM: this value is determined by the value of the ASO-context-name parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: this value is used to determine the value of the ASO-context-name parameter of the A-ASSOCIATE indication primitive, if issued.

NOTE – This field is optional. If backward compatibility with older implementations of ACSE is desired, it must be present.

7.1.4.3 ASO-context-name-list

For the requesting ACPM: the values assigned to this field are determined by the values of the ASO-context-name-list parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: the values are used to determine the values of the ASO-context-name-list parameter on the A-ASSOCIATE indication primitive, if issued.

7.1.4.4 Calling AP-title

For the requesting ACPM: this value is determined by the value of the Calling AP-title parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: this value is used to determine the value of the Calling AP-title parameter of the A-ASSOCIATE indication primitive, if issued.

7.1.4.5 Calling AE-qualifier

7.1.4.5.1 For the requesting ACPM

If the A-ASSOCIATE request is not for a nested association, but the Calling ASOI-tag parameter of the A-ASSOCIATE request contains precisely one (ASO-qualifier, ASOI-identifier) element, the value of the Calling AE-qualifier field is determined by the value of the ASOI-identifier in that element. Otherwise the field shall be absent.

If the A-ASSOCIATE request is for a nested association (i.e. the A-ASSOCIATE request was issued in the context of an established, nesting association), the value of the Calling AE-qualifier shall be determined by the value of the ASOI-identifier in the last element of the Calling ASOI-tag parameter of the A-ASSOCIATE request.

NOTE – The other elements of the ASOI-tag will be implicit in the nesting association.

7.1.4.5.2 For the accepting ACPM

If the AARQ is not for a nested association, the value of the Calling AE-qualifier field shall determine the value of the ASOI-identifier in the first and only element of the Calling ASOI-tag parameter of the A-ASSOCIATE indication.

If the received AARQ is for a nested association (i.e. it is received in the context of an established association), the accepting ACPM shall create a value for the Calling ASOI-tag parameter of the A-ASSOCIATE indication from the value of the ASOI-tag for the peer side as sent or received during the establishment of the nesting association, with one additional (ASO-qualifier, ASOI-identifier) element. The value of the Calling AE-qualifier field of the received AARQ shall determine the value of the ASOI-identifier in the new, last element of the Calling ASOI-tag parameter of the A-ASSOCIATE indication.

If the nesting and nested associations are established in same directions, the ASOI-tag for the peer side is the Calling ASOI-tag on the A-ASSOCIATE request that established the nesting association. If the nesting and nested associations are established from the same direction, the ASOI-tag for the peer side will be the Responding ASOI-tag (if present) on the A-ASSOCIATE confirm that established the nesting association, or the Called ASOI-tag on the A-ASSOCIATE indication that established the nesting association, if the Responding ASOI-tag was not present on the A-ASSOCIATE confirm.

7.1.4.6 Calling AP-invocation-identifier

For the requesting ACPM: this value is determined by the value of the Calling AP-invocation-identifier parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: this value is used to derive the value of the Calling AP-invocation-identifier parameter of the A-ASSOCIATE indication primitive, if issued.

7.1.4.7 Calling AE-invocation-identifier

For the requesting ACPM: this value is determined by the value of the Calling AE-invocation-identifier parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: this value is used to derive the value of the Calling AE-invocation-identifier parameter of the A-ASSOCIATE indication primitive, if issued.

7.1.4.8 Called AP-title

For the requesting ACPM: this value is determined by the value of the Called AP-title parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: this value is used to determine the value of the Called AP-title parameter of the A-ASSOCIATE indication primitive, if issued.

7.1.4.9 Called AE-qualifier

7.1.4.9.1 For the requesting ACPM

If the A-ASSOCIATE request is not for a nested association, but the Called ASOI-tag parameter of the A-ASSOCIATE request contains precisely one (ASO-qualifier, ASOI-identifier) element, the value of the Called AE-qualifier field is determined by the value of the ASO-qualifier in that element. Otherwise the field shall be absent.

If the A-ASSOCIATE request is for a nested association (i.e. the A-ASSOCIATE request was issued in the context of an established, nesting association), the value of the Called AE-qualifier shall be determined by the value of the ASO-qualifier in the last element of the Called ASOI-tag parameter of the A-ASSOCIATE request.

NOTE – The other elements of the ASOI-tag will be implicit in the nesting association.

7.1.4.9.2 For the accepting ACPM

If the AARQ is not for a nested association, the value of the Called AE-qualifier field shall determine the value of the ASO-qualifier in the first and only element of the Called ASOI-tag parameter of the A-ASSOCIATE indication.

If the received AARQ is for a nested association (i.e. it is received in the context of an established association), the accepting ACPM shall create a value for the Called ASOI-tag parameter of the A-ASSOCIATE indication from the value of the ASOI-tag for this side as sent or received during the establishment of the nesting association, with one additional (ASO-qualifier, ASOI-identifier) element. The value of the Called AE-qualifier field of the received AARQ shall determine the value of the ASO-qualifier in the new, last element of the Called ASOI-tag parameter of the A-ASSOCIATE indication.

If the nesting and nested associations are established from the same direction, the ASOI-tag for this side will be the Responding ASOI-tag (if present) on the A-ASSOCIATE confirm that established the nesting association, or the Called ASOI-tag on the A-ASSOCIATE indication that established the nesting association, if the Responding ASOI-tag was not present on the A-ASSOCIATE confirm. If the nesting and nested associations are established in different directions, the ASOI-tag for this side is the Calling ASOI-tag on the A-ASSOCIATE request that established the nesting association.

7.1.4.10 Called AP-invocation-identifier

For the requesting ACPM: this value is determined by the value of the Called AP-invocation-identifier parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: this value is used to derive the value of the Called AP-invocation-identifier parameter of the A-ASSOCIATE indication primitive, if issued.

7.1.4.11 Called AE-invocation-identifier

7.1.4.11.1 For the requesting ACPM

If the A-ASSOCIATE request is not for a nested association, but the Called ASOI-tag parameter of the A-ASSOCIATE request contains precisely one (ASO-qualifier, ASOI-identifier) element, the value of the Called AE-qualifier field is determined by the value of the ASOI-identifier in that element. Otherwise the field shall be absent.

If the A-ASSOCIATE request is for a nested association (i.e. the A-ASSOCIATE request was issued in the context of an established, nesting association), the value of the Called AE-qualifier shall be determined by the value of the ASOI-identifier in the last element of the Called ASOI-tag parameter of the A-ASSOCIATE request.

NOTE – The other elements of the ASOI-tag will be implicit in the nesting association.

7.1.4.11.2 For the accepting ACPM

If the AARQ is not for a nested association, the value of the Called AE-qualifier field shall determine the value of the ASOI-identifier in the first and only element of the Called ASOI-tag parameter of the A-ASSOCIATE indication.

If the received AARQ is for a nested association (i.e. it is received in the context of an established association), the accepting ACPM shall create a value for the Called ASOI-tag parameter of the A-ASSOCIATE indication from the value of the ASOI-tag for this side as sent or received during the establishment of the nesting association, with one additional (ASO-qualifier, ASOI-identifier) element. The value of the Called AE-qualifier field of the received AARQ shall

determine the value of the ASOI-identifier in the new, last element of the Called ASOI-tag parameter of the A-ASSOCIATE indication.

If the nesting and nested associations are established from the same direction, the ASOI-tag for this side will be the Responding ASOI-tag (if present) on the A-ASSOCIATE confirm that established the nesting association, or the Called ASOI-tag on the A-ASSOCIATE indication that established the nesting association, if the Responding ASOI-tag was not present on the A-ASSOCIATE confirm. If the nesting and nested associations are established in different directions, the ASOI-tag for this side is the Calling ASOI-tag on the A-ASSOCIATE request that established the nesting association.

7.1.4.12 ACSE requirements

For the requesting ACPM: the value assigned to this field is determined by the value of the ACSE requirements parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: this value is used to determine the value of the ACSE requirements parameter of the A-ASSOCIATE indication primitive, if issued. The ACPM inspects the ACSE requirements field and removes any functional units not supported by the ACPM before issuing it to the service-user.

7.1.4.13 Authentication-mechanism Name

For the requesting ACPM: the value assigned to this field is determined by the value of the Authentication-mechanism Name parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: this value is used to determine the value of the Authentication-mechanism Name parameter of the A-ASSOCIATE indication primitive, if issued.

7.1.4.14 Authentication-value

For the requesting ACPM: the value assigned to this field is determined by the value of the Authentication-value parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: this value is used to determine the value of the Authentication-value parameter of the A-ASSOCIATE indication primitive, if issued.

7.1.4.15 Implementation Information

For the requesting ACPM: the value assigned to this field is determined within the implementation of the ACPM. It contains information specific to the individual implementation of that ACPM. It is not used in negotiation.

For the accepting ACPM: this field does not affect the operation of the ACPM. Any use depends on a common understanding between the requesting and accepting ACPMs.

7.1.4.16 User-information

For the requesting ACPM: this value is determined by the value of the User-information parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: this value is used to determine the value of the User-information parameter of the A-ASSOCIATE indication primitive, if issued.

7.1.4.17 Called ASOI-tag

For the accepting ACPM: if the higher-level functional unit is proposed in the ACSE requirements parameter, the value assigned to this field shall be determined by the value of the Called ASOI-tag parameter of the A-ASSOCIATE request.

For the receiving ACPM: the value of this field shall determine the value of the Called ASOI-tag parameter of the A-ASSOCIATE indication.

7.1.4.18 Calling ASOI-tag

For the accepting ACPM: if the higher-level functional unit is proposed in the ACSE requirements parameter, the value assigned to this field shall be determined by the value of the Calling ASOI-tag parameter of the A-ASSOCIATE request.

For the receiving ACPM: the value of this field shall determine the value of the Calling ASOI-tag parameter of the A-ASSOCIATE indication.

7.1.4.19 Presentation context definition list

This parameter is defined in ITU-T Rec. X.226 | ISO/IEC 8823-1.

7.1.5 Use of the AARE APDU fields

The AARE APDU fields are used by the accepting and requesting ACPMs as specified below.

7.1.5.1 Protocol Version

For the accepting ACPM: the value of this field assigned by the ACPM depends on whether the association request is accepted or rejected by the ACPM and the acceptor, as specified below:

a) If the association is accepted, the value assigned by the ACPM is a variable length bit string that indicates the protocol version selected by the ACPM from those proposed in the AARQ APDU. Only the bit indicating the version selected is set to one.

b) If the association is rejected, the value assigned by the ACPM is a variable length bit string that indicates the protocol version(s) of this Recommendation | International Standard that could be supported by the ACPM.

For the requesting ACPM: the use of the value in this field depends on whether the association request is accepted or rejected.

a) If the association is accepted, this value defines the protocol version of this Recommendation | International Standard to be used for this association.

b) If the association is rejected, the use of this value is a local option.

7.1.5.2 ASO-context-name

For the accepting ACPM: this value is determined by the value of the ASO-context-name parameter of the A-ASSOCIATE response primitive.

For the requesting ACPM: this value is used to determine the value of the ASO-context-name parameter of the A-ASSOCIATE confirm primitive.

NOTE – This field is optional. If backward compatibility with older implementations of ACSE is desired, it must be present.

7.1.5.3 ASO-context-name-list

For the accepting ACPM: the values assigned to this field are determined by the values of the ASO-context-name-list parameter on the A-ASSOCIATE response primitive.

For the requesting ACPM: the values are used to determine the values of the ASO-context-name-list parameter on the A-ASSOCIATE confirm primitive.

7.1.5.4 Responding AP-title

For the accepting ACPM: this value is determined by the value of the Responding AP-title parameter of the A-ASSOCIATE response primitive.

For the requesting ACPM: this value is used to determine the value of the Responding AP-title parameter of the A-ASSOCIATE confirm primitive, if issued.

7.1.5.5 Responding AE-qualifier

For the accepting ACPM: if the higher-level functional unit is proposed in the ACSE requirements parameter, the value assigned to this field shall be determined by the value of the Responding AE-qualifier parameter of the A-ASSOCIATE request.

For the receiving ACPM: the value of this field shall determine the value of the Responding AE-qualifier parameter of the A-ASSOCIATE indication.

7.1.5.6 Responding AP-invocation-identifier

For the accepting ACPM: if the higher-level functional unit is proposed in the ACSE requirements parameter, the value assigned to this field shall be determined by the value of the Responding AP-invocation-identifier parameter of the A-ASSOCIATE request.

For the receiving ACPM: the value of this field shall determine the value of the Responding AP-invocation-identifier parameter of the A-ASSOCIATE indication.

7.1.5.7 Responding AE-invocation-identifier

For the accepting ACPM: if the higher-level functional unit is proposed in the ACSE requirements parameter, the value assigned to this field shall be determined by the value of the Responding AE-invocation-identifier parameter of the A-ASSOCIATE request.

For the receiving ACPM: the value of this field shall determine the value of the Responding AE-invocation-identifier parameter of the A-ASSOCIATE indication.

7.1.5.8 Result

For the accepting ACPM: the value is determined by the ACPM or by the acceptor as specified below:

a) if the AARQ APDU is rejected by the ACPM (i.e. an A-ASSOCIATE indication primitive is not issued to the acceptor), the value of "rejected (permanent)" or "rejected (transient)" is assigned by the ACPM;

b) otherwise, the value is determined by the Result parameter of the A-ASSOCIATE response primitive.

For the requesting ACPM: this value is used to determine the value of the Result parameter of the A-ASSOCIATE confirm primitive.

NOTE – This field is optional. If backward compatibility with older implementations of ACSE is desired, it must be present.

7.1.5.9 Result source – Diagnostic

This field contains both the Result source value and the Diagnostic value.

NOTE – This field is optional. If backward compatibility with older implementations of ACSE is desired, it must be present.

7.1.5.9.1 Result source value

For the accepting ACPM: this value is assigned by the ACPM as specified below:

a) if the AARQ APDU is rejected by the ACPM (i.e. an A-ASSOCIATE indication primitive is not issued to the acceptor), it assigns the value "ACSE service-provider";

b) otherwise, the ACPM assigns the value "ACSE service-user".

For the requesting ACPM: this value is used to determine the value of the Result source parameter of the A-ASSOCIATE confirm primitive.

7.1.5.9.2 Diagnostic value

For the accepting ACPM: this value is determined by the ACPM or by the acceptor as specified below:

a) If the AARQ APDU is rejected by the ACPM (i.e. an A-ASSOCIATE indication primitive is not issued to the acceptor), the appropriate value is assigned by the ACPM.

b) Otherwise, the value is determined by the value of the Diagnostic parameter of the A-ASSOCIATE response primitive. If the Diagnostic parameter is not included on the response primitive, the ACPM assigns the value of "null".

For the requesting ACPM: this value is used to determine the value of the Diagnostic parameter of the A-ASSOCIATE confirm primitive, unless it has the value of "null". In this case, a Diagnostic value is not included.

7.1.5.10 ACSE requirements

For the accepting ACPM: the value assigned to this field is determined by the value of the ACSE requirements parameter of the A-ASSOCIATE response primitive. This value shall only include functional units that were on the indication primitive.

For the requesting ACPM: this value is used to determine the value of the ACSE requirements parameter of the A-ASSOCIATE confirm primitive.

7.1.5.11 Authentication-mechanism Name

For the accepting ACPM: the value assigned to this field is determined by the value of the Authentication-mechanism Name parameter of the A-ASSOCIATE response primitive.

For the requesting ACPM: this value is used to determine the value of the Authentication-mechanism Name parameter of the A-ASSOCIATE confirm primitive.

7.1.5.12 Authentication-value

For the accepting ACPM: the value assigned to this field is determined by the value of the Authentication-value parameter of the A-ASSOCIATE response primitive.

For the requesting ACPM: this value is used to determine the value of the Authentication-value parameter of the A-ASSOCIATE confirm primitive.

7.1.5.13 Implementation Information

For the accepting ACPM: the value assigned to this field is determined within the implementation of the ACPM. It contains information specific to the individual implementation of that ACPM. It is not used in negotiation.

For the requesting ACPM: this field does not affect the operation of the ACPM. Any use depends on a common understanding between the accepting and requesting ACPMs.

7.1.5.14 User-information

For the accepting ACPM: this value is determined by the value of the User-information parameter of the A-ASSOCIATE response primitive.

For the requesting ACPM: this value is used to determine the value of the User-information parameter of the A-ASSOCIATE confirm primitive.

7.1.5.15 Responding ASOI-tag

For the accepting ACPM: if the higher-level functional unit is proposed in the ACSE requirements parameter, the value assigned to this field shall be determined by the value of the Responding ASOI-tag parameter of the A-ASSOCIATE request.

For the receiving ACPM: the value of this field shall determine the value of the Responding ASOI-tag parameter of the A-ASSOCIATE indication.

7.1.5.16 Presentation context result identifier

This parameter is defined in ITU-T Rec. X.226 | ISO/IEC 8823-1.

7.1.6 Collisions and interactions

7.1.6.1 A-ASSOCIATE service

For an outermost ACPM, an A-ASSOCIATE collision cannot occur (see 6.4). In this case, the two distinct ACPMs would be involved that represent the processing for two distinct associations:

a) an ACPM that processes the initial A-ASSOCIATE request primitive that results in the sending of an AARQ; and

b) an ACPM that processes the subsequently received AARQ APDU.

For ACPMs that are used to establish higher level associations, events can occur when the ACPM is in the Awaiting AARE (STA1) state. There are two distinct cases that must be considered:

1) The ASOI-tags are present, but the Called ASOI-tag of the received AARQ is not the same as the calling ASOI-tag of the AARQ sent (and vice versa). In this case, two associations would be formed and ASOIs would be created if necessary. (If one AARQ has ASO-name and no ASOI-tag and the other has an ASO-name and ASOI-tag, then two distinct associations are created.)

2) The ASOI-tags are present, and the Called ASOI-tag of the received AARQ is identical to the calling ASOI-tag of the AARQ sent (and vice versa). In this case, the action is determined by the control function. To effect this, the A-ASSOCIATE indication primitives are invoked and the CF (potentially after interaction with other ASEs/ASOs and the User) determines the appropriate action.

NOTE – Since the Directory is not intended to maintain ASOI-tag information, such use must be supported by the application.

7.1.6.2 A-ABORT service

If an ACPM receives an A-ABORT request primitive or an IA-ABORT.deliver primitive, it discontinues the normal association establishment procedure, and instead follows the abnormal release procedure.

7.2 Normal release of an association

7.2.1 Purpose

This procedure is used for the normal release of an association by an ASO. It supports the A-RELEASE service.

7.2.2 APDUs used

The normal release procedure uses the A-RELEASE-REQUEST (RLRQ) APDU and the A-RELEASE-RESPONSE (RLRE) APDU. The fields of the RLRQ APDU are listed in Table 5. The fields of the RLRE APDU are listed in Table 6.

Table 5 – RLRQ APDU fields

Field name	Presence	Source	Sink
Reason	U	req	ind
ASO-qualifier	C	req	ind
ASOI-identifier	C	req	ind
User-information	U	req	ind

Table 6 – RLRE APDU fields

Field name	Presence	Source	Sink
Reason	U	rsp	cnf
ASO-qualifier	C	req	ind
ASOI-identifier	C	req	ind
User-information	U	rsp	cnf

7.2.3 Normal release procedure

This procedure is driven by the following events:

a) an A-RELEASE request primitive from the requestor;

b) an RLRQ APDU;

c) an A-RELEASE response primitive from the acceptor; or

d) an RLRE APDU.

7.2.3.1 A-RELEASE request primitive

When an A-RELEASE request primitive is received, the ACPM generates an RLRQ APDU using the parameters from the A-RELEASE request primitive. The RLRQ is mapped to the User-information field of an IA-RELEASE-REQUEST.submit primitive. The IA-RELEASE-REQUEST.submit is invoked and the ACPM transitions to the Awaiting RLRE (STA3) state.

NOTE – The requestor is required to meet the supporting service requirements in order to issue an A-RELEASE request primitive (see Annex C).

The requesting ACPM now waits for a primitive from the underlying service-provider. It does not accept any primitives from the service user other than an A-ABORT request primitive.

7.2.3.2 RLRQ APDU

When the accepting ACPM receives the RLRQ APDU as the contents of the User-information parameter of an IA-DATA.deliver or an IA-RELEASE-REQUEST.deliver, it issues an A-RELEASE indication primitive to the acceptor. It does not accept any ACSE primitives from its service-user other than an A-RELEASE response primitive or an A-ABORT request primitive. The ACPM transitions to the Awaiting A-RLSrsp (STA4) state.

7.2.3.3 A-RELEASE response primitive

The Result parameter on the A-RELEASE response primitive specifies whether the acceptor accepts or rejects the release of the association. The accepting ACPM forms an RLRE APDU from the response primitive parameters.

If the acceptor accepted the release, the Result parameter of the A-RELEASE response primitive has a Result parameter value of "affirmative". The RLRE is mapped to the User-information field of an IA-RELEASE-RESPONSE.submit primitive. An IA-UNBIND.submit primitive is invoked to notify the supporting service that the supported association no longer exists, and the ACPM transitions to the Idle-Unassociated (STA0) state.

If the acceptor rejected the release, the Result parameter of the A-RELEASE response primitive has a Result parameter value of "negative". The RLRE is mapped to the User-information field of an IA-RELEASE-REFUSE.submit primitive and the primitive is invoked. The association continues; the ACPM transitions to the Associated (STA5) state.

NOTE – To give a negative response, the acceptor is required to meet the related underlying service requirements (see Annex C).

7.2.3.4 RLRE APDU

The requesting ACPM receives an RLRE APDU as the contents of the User data parameter of an IA-DATA.deliver or an IA-RELEASE-RESPONSE.deliver from its peer. The Reason field specifies either that the acceptor agrees or disagrees that the association may be released. The requesting ACPM forms an A-RELEASE confirm primitive from the RLRE APDU fields:

a) If the Result parameter on the RLRE APDU specifies "affirmative", the association is released. An IA-UNBIND.submit primitive is invoked to notify the supporting service that the supported association no longer exists, and the ACPM transitions to the Idle-Unassociated (STA0) state.

b) If the Result parameter on the RLRE APDU specifies "negative", the association continues. The requesting ACPM again accepts primitives from its service-user, and the ACPM transitions to the Associated (STA5) state. It invokes an IA-RELEASE-REFUSE primitive.

7.2.3.5 A-RELEASE service collision

An A-RELEASE service collision occurs when an ACPM has sent out an RLRQ APDU (as a result of receiving an A-RELEASE request primitive from its service-user). Instead of receiving the expected RLRE APDU from its peer, it receives an RLRQ APDU.

The ACPM issues an A-RELEASE indication primitive to its service-user. The procedure then followed by an ACPM depends on whether its service-user was the association-initiator or the association-responder:

a) *For the association-initiator:*

1) The ACPM waits for an A-RELEASE response primitive from its service-user. The ACPM transitions to the Collision-Association initiator (STA6) state.

2) When it receives the response primitive, it forms an RLRE APDU from the response primitive's parameters. The RLRE is mapped to an IA-RELEASE-RESPONSE.submit which is invoked and the association continues. The ACPM transitions to the Awaiting RLRE (STA3) state.

3) This ACPM now waits for an RLRE from its peer. It does not accept any primitive from its service-user other than an A-ABORT request primitive.

4) When the ACPM receives an RLRE, it forms an A-RELEASE confirm primitive from the RLRE fields and issues it to its service-user. The association is released. An IA-UNBIND.submit primitive is invoked to notify the supporting service that the supported association no longer exists. The ACPM transitions to the Idle-Unassociated (STA0) state.

In summary, the sequence of events that drive the ACPM of the association-initiator is:

– A-RELEASE request primitive;

– RLRQ APDU (causing the collision);

– A-RELEASE response primitive; and finally,

– RLRE APDU.

b) *For the association-responder:*

1) The ACPM waits for an RLRE from its peer. It does not accept a primitive from its service-user other than an A-RELEASE response or an A-ABORT request primitive. The ACPM transitions to the Collision Association responder (STA7) state.

2) When this ACPM receives an RLRE, it forms an A-RELEASE confirm primitive from the RLRE fields. The association continues and the ACPM transitions to the Awaiting A-RLSrsp (STA4) state.

3) If the ACPM has already received the A-RELEASE response primitive, it forms a RLRE APDU from the response primitive's parameters and maps the RLRE to an IA-RELEASE-RESPONSE.submit to send the APDU. An IA-UNBIND.submit primitive is invoked to notify the supporting service that the supported association no longer exists, and the ACPM transitions to the Idle-Unassociated (STA0) state. Otherwise, the ACPM now waits for an A-RELEASE response primitive from its service-user. When it receives the response primitive, it forms a RLRE APDU from the response primitive's parameters, which is mapped to an IA-RELEASE-RESPONSE.submit primitive which is invoked. An IA-UNBIND.submit primitive is invoked to notify the supporting service that the supported association no longer exists. The ACPM transitions to the Idle-Unassociated (STA0) state.

   In summary, the sequence of events that drive the ACPM of the association-responder for the release of a presentation-connection is:

   – A-RELEASE request primitive;
   – RLRQ APDU (causing the collision);
   – RLRE APDU; and
   – A-RELEASE response primitive.

7.2.4 Use of the RLRQ APDU fields

The RLRQ APDU fields are used by the requesting and accepting ACPMs as specified below.

7.2.4.1 Reason

For the requesting ACPM: this value is determined by the value of the Reason parameter of the A-RELEASE request primitive.

For the accepting ACPM: this value is used to determine the value of the Reason parameter of the A-RELEASE indication primitive.

7.2.4.2 User-information

For the requesting ACPM: this value is determined by the value of the User-information parameter of the A-RELEASE request primitive.

For the accepting ACPM: this value is used to determine the value of the User-information parameter of the A-RELEASE indication primitive.

7.2.4.3 ASO-qualifier

This parameter is used to distinguish APDUs for different child ASOs within the scope of a parent ASO. This parameter can be considered analogous to a protocol-id. The parameter is only required if there are multiple child ASOs within a parent ASO. Its use is therefore conditional.

7.2.4.4 ASOI-identifier

This parameter is used to distinguish APDUs for multiple instances of the same ASO existing concurrently. This parameter is only required if there are multiple instances of a given ASO active at the same time. Its use is therefore conditional.

7.2.5 Use of the RLRE APDU fields

The RLRE APDU fields are used by the accepting and requesting ACPMs as specified below.

7.2.5.1 Reason

For the accepting ACPM: this value is determined by the value of the Reason parameter of the A-RELEASE response primitive.

For the requesting ACPM: this value is used to determine the value of the Reason parameter of the A-RELEASE confirm primitive.

7.2.5.2 User-information

For the accepting ACPM: this value is determined by the value of the User-information parameter of the A-RELEASE response primitive.

For the requesting ACPM: this value is used to determine the value of the User-information parameter of the A-RELEASE confirm primitive.

7.2.5.3 ASO-qualifier

This parameter is used to distinguish APDUs for different child ASOs within the scope of a parent ASO. This parameter can be considered analogous to a protocol-id. The parameter is only required if there are multiple child ASOs within a parent ASO. Its use is therefore conditional.

7.2.6 Collisions and disruptions

7.2.6.1 A-RELEASE service collision

For a given ACPM, an A-RELEASE service collision can occur. The processing for such a collision is described in 7.2.3.5.

NOTE – An A-RELEASE service collision can only occur if no session tokens were selected for the association.

7.2.6.2 A-RELEASE disruption

The normal release procedure is disrupted when the ACPM receives an A-ABORT request primitive, IA-ABORT.deliver primitive. The processing for the ACPM follows the abnormal release procedure (see Annex D).

7.3 Abnormal release of an association

7.3.1 Purpose

The abnormal release procedure can be used at any time to force the abrupt release of the association by a requestor in either ASO, by either ACPM or by the supporting service-provider. When the abnormal release procedure is applied during an attempt to establish an association, the association is not established. The abnormal release procedure supports the A-ABORT and A-P-ABORT services.

7.3.2 APDUs used

The abnormal release procedure uses the A-ABORT (ABRT) APDU. The fields of the ABRT APDU are listed in Table 7.

NOTE – No APDUs are defined for the IA-ABORT.deliver service since it is directly mapped from the underlying Provider ABORT service. This requires any supporting service to be able to make this mapping.

Table 7 – ABRT APDU fields

Field name	Presence	Source	Sink
Abort source	M	ACPM	ind
Diagnostic	U	req	ind
ASO-qualifier	C	req	ind
ASOI-identifier	C	req	ind
User-information	C	req	ind
User data	U	req	ind

7.3.3 Abnormal release procedure

This procedure is driven by any of the following events:

a) an A-ABORT request primitive from the requestor;

b) ABRT APDU;

c) an IA-ABORT.deliver primitive or an IA-UNBIND.deliver; or

d) a protocol error detected by an ACPM.

7.3.3.1 A-ABORT request primitive

When an ACPM receives an A-ABORT request service primitive, it will formulate an ABRT APDU and map it to the User-information field of the IA-ABORT.submit and deliver the primitive to the supporting service to forward it to its peer ACPM. An IA-UNBIND.submit primitive is invoked to notify the supporting service that the supported association no longer exists, and the ACPM transitions to the Idle-Unassociated (STA0) state.

7.3.3.2 ABRT APDU

If IA-DATA.deliver or IA-ABORT.deliver primitive contains an ABRT APDU, the ACPM issues an A-ABORT indication primitive using the Abort source field of the ABRT APDU. If a User Information field is contained in the ABRT APDU, it is included on the A-ABORT indication primitive. An IA-UNBIND.submit primitive is invoked to notify the supporting service that the supported association no longer exists, and the ACPM transitions to the Idle-Unassociated (STA0) state.

7.3.3.3 IA-UNBIND.deliver

The receipt of an IA-UNBIND.deliver informs the ACPM that the support for the association by the supporting service has terminated. The supported association is terminated, since there is no longer an instance of use of the supporting service to provide a context for further service primitives or APDUs.

7.3.3.4 Protocol errors

Two types of ACSE protocol errors are possible:

a) for a particular ACPM state, an unexpected APDU is received; or

b) an invalid field is encountered during the processing of an incoming APDU (see 7.6).

If an unexpected APDU is received, the abnormal release procedure is invoked. If an invalid field is detected by an ACSE procedure, that procedure is disrupted and the abnormal release procedure is invoked.

As part of the abnormal release procedure, the ACPM issues an A-ABORT indication primitive to its service-user, unless the error occurred during the association establishment procedure as the result of receiving an invalid AARQ (see 7.6).

NOTE – Since an A-ASSOCIATE indication primitive will not be issued, an A-ABORT indication primitive would have no meaning, and, therefore, it is not issued.

If an indication primitive is issued, the value of the Abort Source is "ACSE service-provider". The User-Information parameter is not used as specified below. The association is released. An IA-UNBIND.submit primitive is invoked to notify the supporting service that the supported association no longer exists, and the ACPM transitions to the Idle-Unassociated state (STA0).

7.3.4 Use of the ABRT APDU fields

The ABRT APDU fields are used by the requesting and accepting ACPMs as specified below.

7.3.4.1 Abort source

For the requesting ACPM: this value is assigned by the ACPM as specified below:

a) if the ACPM initiated the abort procedure, the ACPM assigns the value of "ACSE service-provider";

b) otherwise, the ACPM assigns the value of "ACSE service-user".

For the accepting ACPM: this value is used to determine the value of the Abort source parameter of the A-ABORT indication primitive.

7.3.4.2 Diagnostic

For the requesting ACPM: if initiated by the service user, this value is determined by the value of the Diagnostic parameter of the A-ABORT request primitive. If generated by the ACPM, the ACPM determines the value.

For the accepting ACPM: this value is used to determine the value of the Diagnostic parameter of the A-ABORT indication primitive.

7.3.4.3 ASO-qualifier

This parameter is used to distinguish APDUs for different child ASOs within the scope of a parent ASO. This parameter can be considered analogous to a protocol-id. The parameter is only required if there are multiple child ASOs within a parent ASO. Its use is therefore conditional.

7.3.4.4 ASOI-identifier

This parameter is used to distinguish APDUs for multiple instances of the same ASO existing concurrently. This parameter is only required if there are multiple instances of a given ASO active at the same time. Its use is therefore conditional.

7.3.4.5 User-information

For the requesting ACPM: this value is determined by the value of the User-information parameter of the A-ABORT request primitive.

For the accepting ACPM: this value is used to determine the value of the User-information parameter of the A-ABORT indication primitive.

7.3.5 Collisions and interactions

The abnormal release procedure may be used whenever an association is established, is in the process of being established, or is being normally released. This procedure disrupts any other currently active procedure. An IA-ABORT.deliver primitive can disrupt the A-ABORT procedure with loss of the A-ABORT information. Collisions of ABRT APDUs are governed by the IA-ABORT services.

7.4 A-DATA

7.4.1 Purpose

The purpose of the A-DATA procedure is to provide the mechanism by which data is transferred on an ASO-association transparently.

7.4.2 APDUs used

The A-DATA procedure uses the A-DT APDU. The fields are listed in Table 8.

Table 8 – A-DT parameters

Parameter name	Presence	Source	Sink
ASO-qualifier	C	req	ind
ASOI-identifier	C	req	ind
A-User-data	M	submit	deliver

7.4.3 A-DATA procedure

This procedure is driven by the following events:

a) an A-DATA.submit primitive; and

b) an A-DT APDU.

7.4.3.1 A-DATA.submit primitive

When the ASOI wishes to send data on an existing ASO-association, it formulates and sends an A-DT APDU by invoking an IA-DATA.submit primitive. The A-DT APDU is sent to the peer ASOI on the existing ASO-association.

7.4.3.2 A-DT APDU

The accepting ACPM receives the A-DT APDU from its peer. The ACPM uses the ASO-qualifier to determine to which child ASO the APDU is to be delivered and the ASO-association-identifier to determine which ASO-association within that child ASO the APDU belongs. If there is only one ASO, the ASO-identifier may not be present (or necessary). If there is only one ASO-association active in this child ASO, the ASO-association-identifier may not be present (or necessary). The accepting ACPM delivers the A-User-data by invoking an A-DATA.deliver service primitive.

7.4.4 Use of the A-DT APDU fields

7.4.4.1 ASO-qualifier

This parameter is used to distinguish APDUs for different child ASOs within the scope of a parent ASO. This parameter can be considered analogous to a protocol-id. The parameter is only required if there are multiple child ASOs within a parent ASO. Its use is therefore conditional.

7.4.4.2 ASOI-identifier

This parameter is used to distinguish APDUs for multiple instances of the same ASO existing concurrently. This parameter is only required if there are multiple instances of a given ASO active at the same time. Its use is therefore conditional.

7.4.4.3 A-User-data

The A-User-data parameter is mandatory. This parameter has meaning only to the ASO on which the ASO-association terminates.

7.5 A-ALTER-CONTEXT

7.5.1 Purpose

The A-ALTER-CONTEXT procedure is used to modify either the ASO-context, the presentation context or both on an established ASO-association. The use of this facility is optional. The use of the facility is specified in the ASO-context definition. The A-ALTER-CONTEXT is used during the lifetime of the association, therefore, it is not used to change protocols, i.e. whole CFs, but to modify the use of the protocol on the association.

7.5.2 APDUs used

The A-ALTER-CONTEXT procedure uses the ACRQ and ACRP APDUs. The fields of these PDUs are listed below.

7.5.3 A-ALTER-CONTEXT procedure

7.5.4 A-ALTER-CONTEXT-REQUEST procedure

When the A-ALTER-CONTEXT request primitive is invoked, the ACPM will formulate an ACRQ APDU and send it on the appropriate ASO-association by invoking an IA-DATA.submit primitive. The User-information field is assumed to be interpreted by the receiving ASO as being in the new contexts specified by this APDU. The A-ALTER-CONTEXT only affects the contexts of the ASO-association. It never has an effect on the Defined Context List maintained by the Presentation Layer, or any other supporting service.

NOTE – The existence or non-existence of A-User-information can be used to create a synchronized or unsynchronized context change (analogous to using major or minor synchronize). ASOs making use of this property must explicitly specify when the User-information field can and cannot be used.

7.5.4.1 ACRQ APDU

When the accepting ACPM receives an ACRQ APDU, it attempts to modify the ASO and presentation contexts according to the fields in the request. Any User-information is interpreted in terms of this new context. An ACPM can only have one outstanding ACRQ APDU at a time. Based on the acceptance or rejection of the proposed context changes, the ACPM formulates an ACRP APDU and sends it to the peer ACPM by invoking an IA-DATA.submit primitive. If the new context changes are not acceptable, the accepting ACPM discards the contents of the User-information field.

7.5.4.2 ACRP procedure

When the ACPM receives an ACRP APDU on an IA-DATA.deliver, it determines whether or not the context changes were accepted and invokes a A-ALTER-CONTEXT confirm primitive to the ACSE user. If the context changes were refused, the ACPM maintains the contexts that were in place before the ACRQ APDU was sent. If the context changes were accepted, the new context is assumed to be in place.

7.5.5 Use of the ACRQ fields

See Table 9.

Table 9 – ACRQ parameters

Parameter name	Presence	Source	Sink
ASO-qualifier	C	req	ind
ASOI-identifier	C	req	ind
ASO-context-name	U	req	ind
ASO-context-name-list	U	req	ind
Presentation context definition list	U	req	ind
User-information	U	req	ind

7.5.5.1 ASO-qualifier

This parameter is used to distinguish APDUs for different child ASOs within the scope of a parent ASO. This parameter can be considered analogous to a protocol-id. The parameter is only required if there are multiple child ASOs within a parent ASO. Its use is therefore conditional.

7.5.5.2 ASOI-identifier

This parameter is used to distinguish APDUs for multiple instances of the same ASO existing concurrently. This parameter is only required if there are multiple instances of a given ASO active at the same time. Its use is therefore conditional.

7.5.5.3 ASO-context-name

This parameter identifies the ASO-context proposed by the requestor. The acceptor returns either the same or refuses the context. The returned name specifies the ASO-context to be used for this association.

7.5.5.4 ASO-context-name-list

For the requesting ACPM: the values assigned to this field are determined by the values of the ASO-context-name-list parameter of the A-ASSOCIATE request primitive.

For the accepting ACPM: the values are used to determine the values of the ASO-context-name-list parameter on the A-ASSOCIATE indication primitive, if issued.

7.5.5.5 Presentation context definition list

{see ITU-T Rec. X.226 | ISO/IEC 8823-1}

7.5.5.6 User-information

The User-information parameter is optional. This parameter has meaning only to the ASO on which the ASO-association terminates. The User-information in this primitive will be considered to be interpreted in terms of whatever context changes were specified by this service. It should be noted that not using this parameter can be used to create a "synchronized" change of context. An ASO wishing to utilize this feature must specify precisely when this parameter should not be used.

7.5.6 Use of the ACRP fields

7.5.6.1 ASO-qualifier

This parameter is used to distinguish APDUs for different child ASOs within the scope of a parent ASO. This parameter can be considered analogous to a protocol-id. The parameter is only required if there are multiple child ASOs within a parent ASO. Its use is therefore conditional.

Table 10 – ACRP parameters

Parameter name	Presence	Source	Sink
ASO-qualifier	C	req	ind
ASOI-identifier	C	req	ind
ASO-context-name	C	rsp	cnf
Presentation context result list	U	rsp	cnf
User-information	U	rsp	cnf

7.5.6.2 ASOI-identifier

This parameter is used to distinguish APDUs for multiple instances of the same ASO existing concurrently. This parameter is only required if there are multiple instances of a given ASO active at the same time. Its use is therefore conditional.

7.5.6.3 ASO-context-name

This parameter identifies the ASO-context proposed by the acceptor. The acceptor returns either the same or refuses the context. The returned name specifies the ASO-context to be used for this association.

7.5.6.4 Presentation context result list

{see ITU-T Rec. X.226 | ISO/IEC 8823-1}

7.5.6.5 User-information

The User-information parameter is optional. This parameter has meaning only to the ASO on which the ASO-association terminates. The User-information in this primitive will be considered to be interpreted in terms of whatever context changes were specified by this service. It should be noted that this parameter can be used to create a "synchronized" change of context. An ASO wishing to utilize this feature must specify precisely when this parameter should not be used.

7.6 Rules for extensibility

When processing an incoming AARQ, the accepting ACPM shall:

a) ignore all tagged values that are not defined in the abstract syntax of this Recommendation | International Standard; and

b) ignore all unknown bit name assignments within a bit string.

After the association has been established or during the establishment of an association, only those ACSE APDUs and related APDU fields defined in the ASN.1 description of the negotiated version of this Recommendation | International Standard shall be issued.

A received APDU or field within an APDU which is not defined in the ASN.1 description of the negotiated version of this Recommendation | International Standard shall be treated as a protocol error.

8 Supporting Service Definition assumed by ACSE

This clause defines the service primitives invoked by the ACSE protocol state machine to pass the ACSE APDUs to and to request services from the supporting service. In general, the supporting service for ACSE will be either the Presentation service or, via the CF of an outer ASO, another instance of ACSE. However, any ASE Service Definition that is compatible with this definition and can be used without perturbing the behaviour of the communicating ACSE protocol machines, can be used to provide the supporting service. Annexes C and D define specific mappings to this service definition. Any ASO using a different supporting service should define a similar mapping.

NOTE – The primary purpose of this clause is to provide a service definition to be used by authors of mappings to specific supporting services. Implementors will only need a cursory knowledge of this material, but should pay careful attention to Annexes D and E.

To support the ability for this Recommendation | International Standard to be used for more than one lower layer mapping, the following terminology is used consistently throughout.

ACPM is the term used to designate an ASOI that implements this Recommendation | International Standard.

Supported association is the term used to designate the ASO-association created by the ACPM generating the IA-primitives.

"Instance of the supporting service" is the term used to designate the binding to an endpoint instance that will be the local context of all APDUs sent or received on the supported association. This can be mapped to a Presentation Connection provided by the Presentation protocol (see Annex C) or the mapping can be managed by the CF of an outer ASO, making use of one or more ASO-associations provided by the ACSE protocol (see annex D), or an equivalent service provided by some other protocol.

For the purposes of this specification, it is assumed that the instance of the supporting service can be modeled as having the following states:

- NULL – No instance of the supporting service exists. For the ACSE protocol, Idle-Unassociated (STA0) is an equivalent state.
- ASSOCIATION PENDING – The ASO has requested an instance of the supporting service and is awaiting a response. Once the request for a supporting service is made the service is in the Establishment Phase. For the ACSE protocol initiator and recipient respectively, Awaiting AARE (STA1) or Awaiting A-ASCrsp (STA2) are equivalent states.
- ESTABLISHED – The ASO has received notification that the supporting service instance has been established. The ASO has entered the Data Transfer Phase. For the ACSE protocol, Associated (STA5) is an equivalent state.
- RELEASE PENDING – The ASO or its peer has requested the instance of the supporting service be terminated. The ASO is awaiting a response from its peer. For the ACSE protocol, Awaiting RLRE (STA3) or the Awaiting A-RLSrsp (STA4) is an equivalent state.

A supporting service will either have this state behaviour as a subset of its overall behaviour or it may use the CF to align its behaviour to this service definition.

8.1 IA-BIND

8.1.1 IA-BIND request

8.1.1.1 IA-BIND-REQUEST.submit

This primitive is invoked by an initiating ACPM to request an instance of the supporting service for the ASO-association that it is attempting to create.

8.1.1.1.1 When Invoked

The initiating ACPM will be in the Awaiting AARE (STA1) state. The supporting service may be in any state except RELEASE PENDING.

8.1.1.1.2 Action upon Receipt

When an IA-BIND request is invoked, the supporting service will attempt to create an instance of the supporting service. If the supporting service is being provided by the outer ASO's CF, this may lead to a new lower-level association being created or the request may be mapped to an existing, lower-level association. The supporting service cannot map this request to any instance of a supporting association that is in the RELEASE PENDING state.

8.1.2 IA-BIND-REQUEST.deliver

This primitive is invoked to notify a recipient ACPM of a new distinct instance of the supporting service that will deliver APDUs to this ACPM.

8.1.2.1 When Invoked

The recipient ACPM will be in the Idle-Unassociated (STA0) state. The supporting service will be in the ASSOCIATION PENDING or ESTABLISHED state.

8.1.2.2 Action upon Receipt

When this primitive is invoked, the recipient ACPM will determine if it is willing to accept this new supporting service. (This determination is specific to the design of the ASO.)

The ACPM invokes a local directory function with the called naming fields (AP-title, AE-qualifier, AP-invocation-identifier, AE-invocation-identifier, and ASOI-tag) to determine the ASO to which the enclosed AARQ is destined, [for a discussion of (N)-directory-functions, (see ITU-T Rec. X.650 | ISO/IEC 7498-3]. If the called naming fields indicate that it is this ASO (which will always be the case unless the higher-level functional unit is selected), then processing of the AARQ continues. If the naming fields refer to a child ASO of this parent, then the AARQ is passed to this ASO; otherwise, the association request is refused and an appropriate AARE is returned to the sender.

8.1.3 IA-BIND request Parameters

The parameters of these primitives contain the parameters necessary to specify the characteristics of the supporting association or connection and are listed in Table 11.

Table 11 – Parameters of the IA-BIND primitives

Parameter	Notes	Req.sub	Req.del	Rsp.sub	Rsp.del
Calling AP-title	1	U	C(=)		
Calling AP-invocation-identifier	1	U	C(=)		
Calling ASOI-tag	1	U	C(=)		
Called AP-title	1	U	C(=)		
Called AP-invocation-identifier	1	U	C(=)		
Called ASOI-tag	1	U	C(=)		
Responding AP-title	1			U	C(=)
Responding AP-invocation-identifier	1			U	C(=)
Responding ASOI-tag	1			U	C(=)
Calling-presentation-address	2	P	P		
Called-presentation-address	2	P	P	P	P
Responding-presentation-address	2			P	P
Presentation context definition list		U	C		
Presentation context result list				U	C
Default presentation context name		U	C		
Default presentation context result				C	C
Quality of service	3	P	P	P	P
Presentation requirements	3	P	P	P	P
Session requirements	3	P	P	P	P
Initial synchronization point serial number	3	P	P	P	P
Initial assignment of tokens	3	P	P	P	P
Session connection identifier	3	P	P	P	P
User-summary		U	C(=)	U	C(=)
User-information		U	C(=)	U	C(=)

NOTE 1 – This parameter is absent unless if the Higher-Level Association functional unit is selected on the supported association.

NOTE 2 – This parameter is absent unless the supported association is the outermost association (i.e. the instance of the supporting service will be a Presentation connection that is not nested).

NOTE 3 – This parameter is not present if the Higher-Level Association functional unit is selected on the association.

8.1.4 IA-BIND response

8.1.4.1 IA-BIND-RESPONSE.submit

This primitive is invoked by a responding ACPM to confirm or to deny the use of the supporting service.

8.1.4.1.1 When Invoked

The ACPM will be in the Awaiting A-ASCrsp (STA2) state and the supporting service will be in the ASSOCIATION PENDING or ESTABLISHED state.

8.1.4.1.2 Action upon Receipt

The supporting service will respond to the initiating peer indicating the success or failure in establishing the supporting service. If the request for supporting service was successful, the instance of the supporting service transitions to the ESTABLISHED state and the responding ACPM transitions to the Associated (STA5) state. If the request is not successful, the instance of the supporting service transitions to the NULL state and the ACPM transitions to the Idle-Unassociated (STA0) state.

8.1.5 IA-BIND-RESPONSE.deliver

8.1.5.1 When Invoked

This primitive is invoked when the ACPM is in the Awaiting AARE (STA3) state and the supporting service must notify the ACPM of the acceptance or refusal of an instance of the supporting service.

8.1.5.2 Action upon Receipt

If the supporting service has been accepted, the AARE is delivered to the ACPM and the supporting service transitions to the ESTABLISHED state. The ACPM is able to use the association. If not accepted, the supporting service transitions to the NULL state.

8.1.6 IA-BIND response Parameters

The parameters of these primitives contain the parameters necessary to specify the characteristics of the supporting association or connection and are listed in Table 11.

8.2 IA-DATA

8.2.1 IA-DATA.submit

8.2.1.1 When Invoked

This primitive may be invoked when ACPM is in the Awaiting AARE (STA1) or Associated (STA5) state and the supporting service is in the ASSOCIATION PENDING or ESTABLISHED states.

8.2.1.2 Action upon Receipt

When this primitive is invoked, the supporting service conveys the contents of the service primitive to the recipient.

8.2.2 IA-DATA.deliver

8.2.2.1 When Invoked

This primitive is invoked when the supporting service is in the ESTABLISHED state.

8.2.2.2 Action upon Receipt

The contents of the User data parameter are delivered to the using ASO.

8.2.3 IA-DATA Parameters

The parameters of these primitives consist of the parameters of the A-DATA primitives.

8.3 IA-ALTER-CONTEXT (optional)

8.3.1 IA-ALTER-CONTEXT-REQUEST.submit

8.3.1.1 When Invoked

This primitive may be invoked when the ACPM is in the Awaiting AARE (STA1) or Associated (STA5) state and the supporting service is in the ASSOCIATION PENDING state or the ESTABLISHED state.

8.3.1.2 Action upon Receipt

When this primitive is invoked, the supporting service is requested to change the ASO-context or the P-context of the supporting service.

8.3.2 IA-ALTER-CONTEXT-REQUEST.deliver

8.3.2.1 When Invoked

This primitive is invoked when the supporting service is in the ESTABLISHED state to notify the supported ACPM that the ASO-context or P-context of the supporting service has been changed.

8.3.2.2 Action upon Receipt

When this primitive is invoked, the supporting service notifies the supported service that there has been a request to change the ASO-context or P-context of the supported service.

8.3.3 IA-ALTER-CONTEXT-RESPONSE.submit

8.3.3.1 When Invoked

This primitive may be invoked when the ACPM is in the Associated (STA5) state and the supporting service is in the ESTABLISHED state.

8.3.3.2 Action upon Receipt

When this primitive is invoked, the ACPM is either accepting or rejecting the requested change to the ASO-context or P-context of the supporting service. If the change is accepted the new context is in effect; otherwise there is no change in the context.

8.3.4 IA-ALTER-CONTEXT-RESPONSE.deliver

8.3.4.1 When Invoked

This primitive is invoked to notify the A-ALTER-CONTEXT request the result of the request.

8.3.4.2 Action upon Receipt

If the request was accepted, the ASO can assume that the context change has been made. If the request was rejected, the ASO assumes that the context is unchanged.

8.3.5 IA-ALTER-CONTEXT Parameters

The parameters of these primitives consist of the parameters of the A-ALTER-CONTEXT primitives necessary to specify context changes to the supporting service.

8.4 IA-ABORT

8.4.1 IA-ABORT.submit

8.4.1.1 When Invoked

This primitive is invoked by the ASO to notify the supporting service that the supported service has been aborted.

NOTE – This allows the supporting service to determine whether its use of supporting associations or connections should also be aborted.

8.4.1.2 Action upon Receipt

This primitive is invoked to notify the supporting service that there has been a request to abort the supported ASO-association. An IA-ABORT.submit may be invoked when either ACPM is in any state. The ACPM transitions to the NULL state.

8.4.2 IA-ABORT.deliver

8.4.2.1 When Invoked

This primitive is invoked to notify the ACPM that the supporting service has aborted the instance of the supporting service.

8.4.2.2 Action upon Receipt

The instance of the supporting service transitions to the NULL state. The ACPM transitions to the Idle-Unassociated (STA0) state.

8.4.3 IA-ABORT Parameters

The parameters of these primitives consist of the parameters of the A-ABORT primitives necessary to notify the supporting service.

8.5 IA-RELEASE

8.5.1 IA-RELEASE-REQUEST.submit

8.5.1.1 When Invoked

This primitive is invoked to notify the supporting service that the ACPM is requesting the release of the supported association. The ACPM transitions to the Awaiting RLRE (STA3) state.

8.5.1.2 Action upon Receipt

When this primitive is generated, then the supporting service conveys the RLRQ APDU (passed as User-information) to the peer ACPM.

8.5.2 IA-RELEASE-REQUEST.deliver

8.5.2.1 When Invoked

This primitive is invoked when an RLRQ APDU (received as User-information/User data) is delivered to the ACPM. The ACPM will be in the Associated (STA5) state or the Awaiting RLRE (STA3) state.

8.5.2.2 Action upon Receipt

If the ACPM is in the Associated (STA5) state, it transitions to the Awaiting RLRE (STA4) state. See Annex A for state transitions in the release-collision case.

8.5.3 IA-RELEASE-ACCEPT.submit

8.5.3.1 When Invoked

This primitive is invoked to notify the supporting service that peer ACPM has accepted the release of the supported association.

8.5.3.2 Action Upon Receipt

When this primitive is received, the contents of the User-information field (an RLRE) are sent to the peer. The ACPM will issue an IA-UNBIND.submit and transitions to the Idle-Unassociated (STA0) state (unless a release collision occurred – see Annex A)

8.5.4 IA-RELEASE-REFUSE.submit

8.5.4.1 When Invoked

This primitive is invoked to notify the supporting service that the request for the release of the supported association has been refused.

8.5.4.2 Action upon Receipt

The ACPM transitions to the Associated (STA5) state.

8.5.5 IA-RELEASE-ACCEPT.deliver

8.5.5.1 When Invoked

This primitive is invoked when an RLRE APDU (received as User-information/User Data) with result "affirmative" is delivered to the ACPM. The ACPM will be in the Awaiting RLSrsp (STA4) state, or one of the collision-resolution states.

8.5.5.2 Action Upon Receipt

The ACPM will issue an IA-UNBIND.submit and transitions to the Idle-Unassociated (STA0) state (unless a release collision occurred – see Annex A)

8.5.6 IA-RELEASE-REFUSE.submit

8.5.6.1 When Invoked

This primitive is invoked when an RLRE APDU (received as User-information/User data) with result "negative" is delivered to the ACPM. The ACPM will be in the Awaiting RLSrsp (STA4) state, or one of the collision-resolution states.

8.5.6.2 Action upon Receipt

The ACPM transitions to the Associated (STA5) state.

8.6 IA-UNBIND

8.6.1 IA-UNBIND.submit

8.6.1.1 When Invoked

This primitive is invoked to notify the supporting service of the release of the instance of the supported service. The ACPM may be in any state.

8.6.1.2 Action upon Receipt

The ACPM transitions to the Idle-Unassociated (STA0) state. Any action by the supporting service depends on the specification of that ASO.

8.6.2 IA-UNBIND.deliver

8.6.2.1 When Invoked

This primitive is Invoked to notify the ACPM that the instance of the supported service has been terminated (and thus there is no longer a local context for the issue or receipt of APDUs on the supported association).

8.6.2.2 Action upon Receipt

When the ACPM receives an IA-UNBIND.deliver, it is notified that the instance of the supporting association has terminated. The ACPM transitions to the Idle-Unassociated (STA0) state.

8.6.3 IA-UNBIND Parameters

This primitive has a provider-reason parameter.

9 Syntax of ACSE

9.1 Structure of ACSE APDUs

The abstract syntax of each of the ACSE APDUs is specified in this clause using ASN.1 (see ITU-T Rec. X.680 | ISO/IEC 8824-1).

```
ACSE-1 { joint-iso-itu-t association-control(2) modules(0) acse1(1) version1(1) }
        -- ACSE-1 refers to ACSE version 1

        DEFINITIONS ::=

        BEGIN

        IMPORTS  Name, RelativeDistinguishedName
                        FROM InformationFramework
                                { joint-iso-ccitt ds(5) module(1) informationFramework(1)  2 };
```

```
-- The data types Name and RelativeDistinguishedName are imported from
-- ITU-T Rec. X.501 | ISO/IEC 9594-2.
-- object identifier assignments

acse-as-id OBJECT IDENTIFIER ::=
 { joint-iso-itu-t association-control(2) abstract-syntax(1) apdus(0)  version1(1) }
-- may be used to reference the abstract syntax of the ACSE APDUs.

aCSE-id OBJECT IDENTIFIER ::=
 { joint-iso-itu-t association-control(2) ase-id(3) acse-ase(1) version(1) }
-- may be used to identify the Association Control ASE.
-- top level CHOICE

ACSE-apdu  ::=  CHOICE
 {
        aarq  AARQ-apdu,
        aare  AARE-apdu,
        rlrq  RLRQ-apdu,
        rlre  RLRE-apdu,
        abrt  ABRT-apdu,
        ... ,
-- Extensions for higher level association FU

        adt  A-DT-apdu,
        acrq  ACRQ-apdu,
        acrp  ACRP-apdu
 }

AARQ-apdu  ::=  [ APPLICATION 0 ]    IMPLICIT SEQUENCE
 { protocol-version          [0]         IMPLICIT BIT STRING
                                         { version1 (0) }
                                                 DEFAULT { version1 },
  aSO-context-name           [1]         ASO-context-name,
  called-AP-title      [2]   AP-title OPTIONAL,
  called-AE-qualifier        [3]         AE-qualifier OPTIONAL,
  called-AP-invocation-identifier    [4]         AP-invocation-identifier  OPTIONAL,
  called-AE-invocation-identifier    [5]         AE-invocation-identifier  OPTIONAL,
  calling-AP-title           [6]         AP-title    OPTIONAL,
  calling-AE-qualifier       [7]         AE-qualifier  OPTIONAL,
  calling-AP-invocation-identifier   [8]         AP-invocation-identifier OPTIONAL,
  calling-AE-invocation-identifier   [9]         AE-invocation-identifier OPTIONAL,
  -- The following field shall not be present if only the Kernel is used.

  sender-acse-requirements   [10]        IMPLICIT ACSE-requirements OPTIONAL,
  -- The following field shall only be present if the Authentication functional unit is selected.

  mechanism-name             [11]        IMPLICIT Mechanism-name OPTIONAL,
  -- The following field shall only be present if the Authentication functional unit is selected.

  calling-authentication-value       [12]        EXPLICIT Authentication-value OPTIONAL,
  aSO-context-name-list      [13]        IMPLICIT ASO-context-name-list OPTIONAL,
-- The above field shall only be present if the Application Context Negotiation functional unit is selected.

  implementation-information         [29]        IMPLICIT Implementation-data OPTIONAL,
  ...,
  -- Extensions for higher level association FU

  p-context-definition-list          [14]  Syntactic-context-list OPTIONAL,
  called-asoi-tag            [15]  IMPLICIT ASOI-tag OPTIONAL,
  calling-asoi-tag           [16]  IMPLICIT ASOI-tag OPTIONAL,
  -- End of extensions for higher level association FU

  ...,
  user-information           [30]        IMPLICIT Association-data OPTIONAL
  }

AARE-apdu  ::=  [ APPLICATION 1 ]  IMPLICIT SEQUENCE
  { protocol-version         [0]         IMPLICIT BIT STRING{ version1 (0) }
                                                 DEFAULT { version1 },
  aSO-context-name           [1]         ASO-context-name
  result                     [2]         Associate-result,
  result-source-diagnostic   [3]         Associate-source-diagnostic,
```

```
    responding-AP-title      [4]     AP-title    OPTIONAL,
    responding-AE-qualifier  [5]     AE-qualifier OPTIONAL,
    responding-AP-invocation-identifier  [6]  AP-invocation-identifier OPTIONAL,
    responding-AE-invocation-identifier  [7]  AE-invocation-identifier OPTIONAL,
  -- The following field shall not be present if only the Kernel is used.

    responder-acse-requirements [8]  IMPLICIT ACSE-requirements OPTIONAL,
  -- The following field shall only be present if the Authentication functional unit is selected.

    mechanism-name       [9]     IMPLICIT Mechanism-name OPTIONAL,
  -- This following field shall only be present if the Authentication functional unit is selected.

    responding-authentication-value  [10]  EXPLICIT Authentication-value OPTIONAL,
    aSO-context-name-list  [11]  IMPLICIT ASO-context-name-list OPTIONAL,
-- The above field shall only be present if the Application Context Negotiation functional unit is selected.

    implementation-information  [29]  IMPLICIT Implementation-data OPTIONAL,
    ...,
  -- Extensions for higher level association FU

    p-context-result-list  [12]  IMPLICIT P-context-result-list OPTIONAL,
    called-asoi-tag        [13]  IMPLICIT ASOI-tag OPTIONAL,
    calling-asoi-tag       [14]  IMPLICIT ASOI-tag OPTIONAL,
  -- End of extensions for higher level association FU

    ...,
    user-information       [30]  IMPLICIT Association-data OPTIONAL
  }

RLRQ-apdu  ::=  [ APPLICATION 2 ]   IMPLICIT SEQUENCE
  { reason                 [0]     IMPLICIT Release-request-reason  OPTIONAL,
    ...,
    -- Extensions for higher level association FU

    aso-qualifier      [13]  ASO-qualifier OPTIONAL,
    asoi-identifier    [14]  IMPLICIT ASOI-identifier OPTIONAL,
    -- End of extensions for higher level association FU

    ...,
    user-information       [30]  IMPLICIT Association-data       OPTIONAL
  }

RLRE-apdu  ::=  [ APPLICATION 3 ] IMPLICIT SEQUENCE
  { reason                 [0]     IMPLICIT Release-response-reason  OPTIONAL,
    ...,
    -- Extensions for higher level association FU

    aso-qualifier      [13]  ASO-qualifier OPTIONAL,
    asoi-identifier    [14]  IMPLICIT ASOI-identifier OPTIONAL,
    -- End of extensions for higher level association FU

    ...,
    user-information       [30]  IMPLICIT Association-data       OPTIONAL
  }

ABRT-apdu ::= [ APPLICATION 4 ] IMPLICIT SEQUENCE
  { abort-source           [0]     IMPLICIT ABRT-source,
    abort-diagnostic       [1]     IMPLICIT ABRT-diagnostic OPTIONAL,
  -- This field shall not be present if only the Kernel is used.

    ...,
    -- Extensions for higher level association FU

    aso-qualifier  [13]    ASO-qualifier OPTIONAL,
    asoi-identifier        [14]    IMPLICIT ASOI-identifier OPTIONAL,
    -- End of extensions for higher level association FU

    ...,
    user-information       [30]    IMPLICIT Association-data  OPTIONAL
  }
```

```
A-DT-apdu ::= [APPLICATION 5] IMPLICIT SEQUENCE
 { aso-qualifier          [0]    ASO-qualifier OPTIONAL,
  asoi-identifier                [1]    IMPLICIT ASOI-identifier OPTIONAL,
  ..., ...,
  a-user-data             [30]    IMPLICIT User-Data     }

ACRQ-apdu ::= [APPLICATION 6] IMPLICIT SEQUENCE
 { aso-qualifier          [0]    IMPLICIT ASO-qualifier OPTIONAL,
  asoi-identifier                [1]    IMPLICIT ASOI-identifier OPTIONAL,
  aSO-context-name               [3]    IMPLICIT ASO-context-name OPTIONAL,
  aSO-context-name-list                 [4]    IMPLICIT ASO-context-name-list OPTIONAL,
  p-context-definition-list             [5]    Syntactic-context-list OPTIONAL,
  ..., ...,
  user-information               [30]    IMPLICIT User-information OPTIONAL      }

ACRP-apdu ::= [APPLICATION 7] IMPLICIT SEQUENCE
 { aso-qualifier          [0]    ASO-qualifier OPTIONAL,
  asoi-identifier                [1]    IMPLICIT ASOI-identifier OPTIONAL,
  aSO-context-name               [3]    IMPLICIT ASO-context-name-list OPTIONAL,
  p-context-result-list          [4]    IMPLICIT P-context-result-list OPTIONAL,
  ..., ...,
  user-information               [30]    IMPLICIT User-information OPTIONAL   }

ABRT-diagnostic ::= ENUMERATED
 { no-reason-given (1),
  protocol-error (2),
  authentication-mechanism-name-not-recognized (3),
  authentication-mechanism-name-required (4),
  authentication-failure (5),
  authentication-required (6),
...
 }

ABRT-source ::= INTEGER { acse-service-user (0), acse-service-provider (1)} (0..1, ...)

ACSE-requirements ::= BIT STRING
 { authentication (0),
  aSO-context-negotiation (1),
  higher-level-association (2),
  nested-association (3)
 }

Application-context-name ::= ASO-context-name

ASO-context-name ::= OBJECT IDENTIFIER
-- Application-entity title productions follow (not in alphabetical order).

AP-title ::= CHOICE {
                ap-title-form1    AP-title-form1,
                ap-title-form2    AP-title-form2,
                ... ,
                ap-title-form3    AP-title-form3
                 }

AE-qualifier ::= ASO-qualifier

ASO-qualifier ::= CHOICE {
                aso-qualifier-form1        ASO-qualifier-form1,
                aso-qualifier-form2        ASO-qualifier-form2,
                ...,
                aso-qualifier-form3        ASO-qualifier-form3
          }
-- When both AP-title and AE-qualifier data values are present in an AARQ or AARE APDU, both must
-- have the same form to allow the construction of an AE-title as discussed in CCITT Rec. X.665 |
-- ISO/IEC 9834-6

AP-title-form1 ::= Name
-- The value assigned to AP-title-form1 is The Directory Name of an application-process title.

ASO-qualifier-form1 ::= RelativeDistinguishedName
-- The value assigned to AE-qualifier-form1 is the relative distinguished name of a particular
-- application-entity of the application-process identified by AP-title-form1.
```

```
AP-title-form2 ::= OBJECT IDENTIFIER

ASO-qualifier-form2 ::= INTEGER

AP-title-form3 ::= PrintableString

ASO-qualifier-form3 ::= PrintableString

AE-title ::= CHOICE {
                    ae-title-form1      AE-title-form1,
                    ae-title-form2      AE-title-form2,
                    ...         }
-- As defined in ITU-T Rec. X.650 | ISO/IEC 7498-3, an application-entity title is composed of an application process
-- title and an application-entity qualifier. The ACSE protocol provides for the transfer of an application-entity title
-- value by the transfer of its component values. However, the following data type is provided for International
-- Standards that reference a single syntactic structure for AE titles.
        AE-title-form1 ::= Name
-- For access to The Directory (see ITU-T Rec. X.500 series | ISO/IEC 9594), an AE title has AE-title-form1. This value
-- can be constructed from AP-title-form1 and AE-qualifier-form1 values contained in an AARQ or AARE APDU.
-- A discussion of forming an AE-title-form1 from AP-title-form1 and AE-qualifier form1 may be found in CCITT
-- Rec X.665 | ISO/IEC 9834-6.
        AE-title-form2 ::= OBJECT IDENTIFIER
        -- A discussion of forming an AE-title-form2 from AP-title-form2 and AE-qualifier-form2 may be found in
        -- CCITT Rec. X.665 | ISO/IEC 9834-6.

        AE-invocation-identifier ::= INTEGER

        AP-invocation-identifier ::= INTEGER

        ASOI-identifier ::= INTEGER ( 1 .. 128, ... )

        ASOI-tag  ::=  SEQUENCE SIZE ( 0 .. 7, ... )  OF SEQUENCE
                                { qualifier         [0]     ASO-qualifier OPTIONAL,
                                  identifier        [1]     ASOI-identifier OPTIONAL }
        -- End of Application-entity title productions

        ASO-context-name-list  ::=  SEQUENCE OF ASO-context-name

        Syntactic-context-list  ::= CHOICE
        {  context-list             [0]   Context-list,
           default-contact-list     [1]   Default-Context-List
        }

        Context-list  ::=  SEQUENCE OF SEQUENCE
        {  pci              Presentation-context-identifier,
           abstract-syntax Abstract-syntax-name,
           transfer-syntaxes SEQUENCE OF Transfer-syntax-name}

        Default-Context-List  ::=  SEQUENCE OF SEQUENCE
                    { abstract-syntax-name          [0]         IMPLICIT Abstract-syntax-name OPTIONAL,
                      transfer-syntax-name          [1]         IMPLICIT Transfer-syntax-name
        }

        Abstract-syntax-name  ::=  OBJECT IDENTIFIER

        P-context-result-list  ::=   SEQUENCE OF SEQUENCE
                  {  result             [0]         IMPLICIT Result,
                     concrete-syntax-name       [1]         IMPLICIT Concrete-syntax-name OPTIONAL,
                     provider-reason            [2]         IMPLICIT INTEGER
                           {    reason-not-specified                (0),
                                abstract-syntax-not-supported  (1),
                                proposed-transfer-syntaxes-not-supported (2),
                                local-limit-on-DCS-exceeded     (3)  } OPTIONAL
        }

        Result  ::=  INTEGER  {  acceptance (0),
                  user-rejection (1),
                  provider-rejection (2)
                   }

        Concrete-syntax-name ::=  Transfer-syntax-name

        Transfer-syntax-name  ::=  OBJECT IDENTIFIER
```

```
Associate-result  ::= INTEGER
        { accepted (0),
        rejected-permanent (1),
        rejected-transient (2)
         } (0..2, ...)

Associate-source-diagnostic  ::=  CHOICE
 { acse-service-user          [1] INTEGER
                                  { null (0),
                                   no-reason-given (1),
                                   application-context-name-not-supported (2),
                                   calling-AP-title-not-recognized (3),
                                   calling-AP-invocation-identifier-not-recognized (4),
                                   calling-AE-qualifier-not-recognized (5),
                                   calling-AE-invocation-identifier-not-recognized (6),
                                   called-AP-title-not-recognized (7),
                                   called-AP-invocation-identifier-not-recognized (8),
                                   called-AE-qualifier-not-recognized (9),
                                   called-AE-invocation-identifier-not-recognized (10),
                                   authentication-mechanism-name-not-recognized (11),
                                   authentication-mechanism-name-required (12),
                                   authentication-failure (13),
                                   authentication-required (14)
                                  } (0..14 ,  ... ),
      acse-service-provider [2] INTEGER
                                  { null (0),
                                  no-reason-given (1),
                                  no-common-acse-version (2)
                                  }(0..2 , ... )
 }

User-information ::= Association-data

Association-data ::=  SEQUENCE SIZE (1, ..., 0 | 2..MAX) OF EXTERNAL

Simply-encoded-data ::= OCTET STRING

User-Data ::=  CHOICE {
        user-information User-information,
        simply-encoded-data        Simply-encoded-data,
        fully-encoded-data         [0]        PDV-list
 }
-- see ITU-T Rec. X.226 | ISO/IEC 8823-1.

PDV-list  ::=  SEQUENCE   {
          transfer-syntax-name              Transfer-syntax-name OPTIONAL,
          presentation-context-identifier   Presentation-context-identifier,
          presentation-data-values      CHOICE   {
               simple-ASN1-type             [0]       ABSTRACT-SYNTAX.&Type
               (CONSTRAINED BY  {
-- Type corresponding to presentation context identifier

                } ),
               octet-aligned                [1]       IMPLICIT OCTET STRING,
               arbitrary                    [2]       IMPLICIT BIT STRING    }
          -- see ITU-T Rec. X.226 | ISO/IEC 8823-1.

}

Presentation-context-identifier  ::=  INTEGER

Authentication-value  ::=   CHOICE
 { charstring      [0] IMPLICIT GraphicString,
   bitstring       [1] IMPLICIT BIT STRING,
   external        [2] IMPLICIT EXTERNAL,
   other           [3] IMPLICIT SEQUENCE {
                   other-mechanism-name  MECHANISM-NAME.&id ({ObjectSet}),
                   other-mechanism-value  MECHANISM-NAME.&Type ({ObjectSet}{@.other-mechanism-name})
                  }
 }
```

```
-- The abstract syntax of (calling/responding) authentication-value is determined by the authentication mechanism used
-- during association establishment. The authentication mechanism is either explicitly denoted by the &id field (of type
-- OBJECT IDENTIFIER) for a mechanism belonging to the class MECHANISM-NAME, or it is known implicitly by
-- prior agreement between the communicating partners. If the "other" component is chosen, then the
-- "mechanism-name" component must be present in accordance with ITU-T Rec. X.680 | ISO/IEC 8824-1. If the value
-- "mechanism-name" occurs in the AARQ-apdu or the AARE-apdu, then that value must be the same as the value for
-- "other-mechanism-name".

Implementation-data ::= GraphicString

Mechanism-name ::= OBJECT IDENTIFIER

MECHANISM-NAME ::= TYPE-IDENTIFIER

ObjectSet MECHANISM-NAME ::= {...}

Release-request-reason ::= INTEGER
    { normal (0) ,
    urgent (1) ,
    user-defined (30) }
            (0 | 1 | 30, ...)

Release-response-reason ::= INTEGER
    { normal (0) ,
    not-finished (1) ,
    user-defined (30) }
            (0 | 1 | 30, ...)

END
```

10 Conformance

A system claiming to implement the procedures specified in this Recommendation | International Standard shall comply with the requirements in 10.1 through 10.3.

10.1 Statement requirements

The following shall be stated by the implementor:

a) whether the system is capable of acting in the role of association-initiator, or association-responder, or both;

b) that the system supports version 1 of this protocol; and

c) which of the following optional functional units are supported:

- Authentication;
- ASO-context negotiation;
- Higher Level Association;
- Nested Association.

10.2 Static requirements

The use of the Association Control Service Element is required for an ASO to meet the minimum requirements for establishing and releasing communication with a peer entity.

The system shall:

a) act in the role of an association-initiator (by sending an AARQ APDU), or in the role of an association-acceptor (by responding properly to an AARQ APDU with an appropriate AARE APDU), or act in both roles; and

b) support (as a minimum) encoding which results from applying the basic ASN.1 encoding rules to the ASN.1 specified in clause 9 for the purpose of exchanging ACSE APCI.

10.3 Dynamic requirements

The system shall:

a) follow all the procedures specified in clause 7 (including the rules for extensibility) and Annex A; and

b) support the mapping onto the supporting service and to ACSE defined in 8.1 to clause 9.

The requesting AE may choose to send either form1 or form2 for AP-title and AE-qualifier. The accepting AE may respond with either form. Thus, both the requesting and responding AE must be capable of receiving both forms.

c) state which of the following optional functional units are supported:

- Authentication;
- ASO-context negotiation;
- Higher Level Association;
- Nested Association.

11 Precedence

The aspects of the protocol for ACSE are specified in several clauses in this Recommendation | International Standard. This clause states the rules of precedence for possible situations where the same aspect may be specified in more than one place in an apparently inconsistent manner. The relevant aspects of protocol specification covered are:

a) sequencing rules;

b) mapping to the supporting service; and

c) structure and encoding of ACSE APDUs.

Annex A and clause 7 specify the elements of procedure which govern the behaviour of the ACPM. Annex A takes precedence over any other clause in this Recommendation | International Standard which may state or imply apparently inconsistent sequencing rules.

Clause 8 specifies how the supporting service primitives are used by the ACPM. Clause 8 takes precedence over any other part of this Recommendation | International Standard which may state or imply mapping to the supporting-service.

Clause 9 specifies the structure of ACSE APDUs. Clause 9 takes precedence over any other part of this Recommendation | International Standard which may state or imply structure of ACSE APDUs.

NOTE – Any person encountering an inaccuracy or ambiguity in this Recommendation | International Standard is requested to notify their National Body of ISO without delay in order that the matter may be investigated and appropriate action taken.

12 Registration requirements

This Recommendation | International Standard identifies the requirement to register four types of information objects: application titles; ASO-contexts; and authentication-mechanisms; and upper-layer context specifications. Each is discussed below.

No International Registration Authority is currently planned for the registration of any of the above objects. The assignment of a name to any of these objects shall be in accordance with the general provisions of CCITT Rec. X.660 | ISO/IEC 9834-1, except as specified below.

In accordance with CCITT Rec. X.660 | ISO/IEC 9834-1, an organization that wishes to assign names to objects shall find an appropriate superior of the naming tree. The superior assigns an arc of the naming tree to the organization. The organization can then assign names below that arc.

NOTE – Appropriate superiors in the registration tree include ISO/IEC national bodies, organizations with International Code Designators assigned in accordance with ISO 6523, as well as ITU-T network Administrations, and recognized operating agencies (ROAs).

12.1 Application titles

The application titles requiring registration are application-process title, application-entity qualifier, and application-entity title. The registration requirements for these information objects are contained in ITU-T Rec. X.207 | ISO/IEC 9545, clause 9. ITU-T Rec. X.665 | ISO/IEC 9834-6 specifies both the relationship between these information objects and the procedures to register them.

12.2 ASO-context

The registration requirements for an ASO-context is contained in ITU-T Rec. X.207 | ISO/IEC 9545, clause 9. CCITT Rec. X.660 | ISO/IEC 9834-1 specifies the procedures to register it.

12.3 Authentication-mechanism

An authentication-mechanism may be specified as part of a Recommendation | International Standard. For example, Annex B includes an authentication-mechanism based on AE-title and password. Such an authentication-mechanism is, in effect, specified and registered within this Recommendation | International Standard.

An authentication-mechanism may also be specified outside of Recommendations | International Standards. In this situation, CCITT Rec. X.660 | ISO/IEC 9834-1 specifies the procedures to register such an authentication-mechanism.

12.4 Upper-layer context specifications

An upper-layer context specification is a definition of all the field values that are required to format the full-form ACSE, presentation, and session establishment PDUs for a given application context and a given peer presentation address.

NOTE – In practice, it is expected that an upper-layer context specification will be parameterized to allow for values which may be expected to be different for each establishment exchange between two peers (e.g. ACSE user information), or for the same application between different peers (e.g. addressing information).

An upper-layer context specification may be specified as part of an Recommendation | International Standard.

An upper-layer context specification may also be specified outside of Recommendations | International Standards. In this situation, CCITT Rec. X.660 | ISO/IEC 9834-1 specifies the procedures to register such an upper-layer context specification.

Annex A

ACPM state table

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

A.1 General

A.1.1 This annex defines a single Association Control Protocol Machine (ACPM) in terms of a state table. The state table shows the interrelationship between the state of an ACPM, the incoming events that occur in the protocol, the actions taken, and, finally, the resultant state of the ACPM.

A.1.2 The ACPM state table does not constitute a formal definition of the ACPM. It is included to provide a more precise specification of the elements of procedure defined in clause 7.

A.1.3 This annex contains the following tables:

a) Table A.1 specifies the abbreviated name, source, and name/description of each incoming event. The sources are:

   1) ACSE service-user (AC-user);

   2) Peer ACPM (AC-peer); and

   3) IA service-provider (IA-provider).

b) Table A.2 specifies the abbreviated name of each state.

c) Table A.3 specifies the abbreviated name, target, and name/description of each outgoing event. The targets are:

   1) ACSE service-user (AC-user); and

   2) peer ACPM (AC-peer).

d) Table A.4 specifies the predicates.

e) Table A.5 specifies the ACPM state table using the abbreviations of the above tables.

A.2 Conventions

A.2.1 The intersection of an incoming event (row) and a state (column) forms a cell.

A.2.2 In the state table, a blank cell represents the combination of an incoming event and a state that is not defined for the ACPM (see A.3.1).

A.2.3 A non-blank cell represents an incoming event and state that is defined for the ACPM. Such a cell contains one or more action lists. An action list may be either mandatory or conditional. If a cell contains a mandatory action list, it is the only action list in the cell.

A.2.4 A mandatory action list contains:

a) an outgoing event; and

b) a resultant state.

A.2.5 A conditional action list contains:

a) a predicate expression comprising predicates and Boolean operators (^ represents the Boolean NOT); and

b) a mandatory action list. (This mandatory action list is used only if the predicate expression is true.)

A.3 Actions to be taken by the ACPM

The ACPM state table defines the action to be taken by the ACPM in terms of an outgoing event and the resultant state of the ACPM.

A.3.1 Invalid intersections

Blank cells indicate an invalid intersection of an incoming event and state. If such an intersection occurs, one of the following actions is taken:

a) if the incoming event comes from the ACSE service-user, any action taken by the ACPM is a local matter;

b) if the incoming event is related to a received APDU or IA-provider event, the ACPM issues both an A-ABRind outgoing event (to its AC-user) and an ABRT outgoing event (to its peer ACPM).

A.3.2 Valid intersections

If the intersection of the state and incoming event is valid, one of the following actions is taken:

a) if a cell contains a mandatory action list, the ACPM takes the actions specified;

b) if a cell contains one or more conditional action lists, for each predicate expression that is true, the ACPM takes the actions specified. If none of the predicate expressions are true, the ACPM takes one of the actions defined in A.3.1.

A.4 Relationship to Presentation and other ASEs

The ACPM state table (see Table A.5) only defines the interactions of the ACPM, its ACSE service-user and the presentation-services used by the ACPM.

NOTE – The occurrence of other events from the presentation-service or other application-service-elements is not included in the ACPM state Table A.5 because they do not affect the ACPM.

Table A.1 – Incoming event list

Abbreviated name	Source	Name and description
A-ASCreq	AC-user	A-ASSOCIATE request primitive
A-ASCrsp+	AC-user	A-ASSOCIATE response primitive (accepted)
A-ASCrsp–	AC-user	A-ASSOCIATE response primitive (rejected)
AARQ	AC-peer	A-ASSOCIATE request APDU
AARE+	AC-peer	A-ASSOCIATE-RESPONSE APDU (accepted)
AARE–	AC-peer	A-ASSOCIATE-RESPONSE APDU (rejected)
SupportCnf	IA-BIND-RESPONSE.deliver	P-CONNECT, A-ASSOCIATE APDU, or A-DATA APDU
A-DTreq	AC-user	A-DATA request primitive
DT	AC-peer	A-DATA APDU
A-ACRQreq	AC-user	A-ALTER-CONTEXT request primitive
A-ACRQrsp	AC-user	A-ALTER-CONTEXT response primitive
ACRQ	AC-peer	A-ALTER-CONTEXT APDU
ACRP	AC-peer	A-ALTER-CONTEXT-RESPONSE APDU
A-RLSreq	AC-user	A-RELEASE request primitive
A-RLSrsp+	AC-user	A-RELEASE response primitive (accepted)
A-RLSrsp–	AC-user	A-RELEASE response primitive (rejected)
RLRQ	AC-peer	A-RELEASE APDU
RLRE+	AC-peer	A-RELEASE APDU (accepted)
RLRE–	AC-peer	A-RELEASE APDU (rejected)
A-ABRreq	AC-user	A-ABORT request primitive
ABRT[a)]	AC-peer	A-ABORT APDU
supportingAB	IA-ABT.deliver	P-P-ABORT or A-ABORT
EXTERN	Supporting Upper Layers	External Event

a) When supported by version 1 of the session protocol (ISO/IEC 8327), the A-ABORT APDU has no APCI. The receipt of the P-U-ABORT indication implies its existence.

Table A.2 – ACPM states

Abbreviated name	Description
STA0	Idle: Unassociated
STA1	Awaiting AARE APDU
STA2	Awaiting A-ASSOCIATE response
STA3	Awaiting RLRE APDU
STA4	Awaiting A-RELEASE response
STA5	Associated
STA6	Awaiting A-RELEASE response (Association-initiator)
STA7	Awaiting RLRE APDU (Association-responder 1)
STA8	Awaiting RLRE APDU (Association-responder 2)

Table A.3 – Outgoing event list

Abbreviated name	Target	Name and description
A-ASCind	AC-user	A-ASSOCIATE indicate primitive
A-ASCcnf+	AC-user	A-ASSOCIATE confirm primitive (accepted)
A-ASCcnf–	AC-user	A-ASSOCIATE confirm primitive (rejected)
AARQ	AC-peer	A-ASSOCIATE request primitive
AARE+	AC-peer	A-ASSOCIATE-RESPONSE APDU (accepted)
AARE–	AC-peer	A-ASSOCIATE-RESPONSE APDU (rejected)
A-DTind	AC-user	A-DATA indicate primitive
DT	AC-peer	A-DATA APDU
A-ACRQind	AC-user	A-ALTER-CONTEXT indicate primitive
A-ACRQcnf	AC-user	A-ALTER-CONTEXT confirm primitive
ACRQ	AC-peer	A-ALTER-CONTEXT APDU
ACRP	AC-peer	A-ALTER-CONTEXT-RESPONSE APDU
A-RLSind	AC-user	A-RELEASE indicate primitive
A-RLScnf+	AC-user	A-RELEASE confirm primitive (accepted)
A-RLScnf–	AC-user	A-RELEASE confirm primitive (rejected)
RLRQ	AC-peer	A-RELEASE APDU
RLRE+	AC-peer	A-RELEASE APDU (accepted)
RLRE–	AC-peer	A-RELEASE APDU (rejected)
A-ABRind	AC-user	A-ABORT indicate primitive
ABRT[a)]	AC-peer	A-ABORT APDU
P-P-ABind	AC-user	A-P-ABORT indicate primitive

a) When supported by version 1 of the session protocol (ISO/IEC 8327), the A-ABORT APDU has no APCI. The receipt of the P-U-ABORT indication implies its existence.

Table A.4 – Predicates

Code	Meaning
p1	ACPM can support requested association
p2	ACPM originated this association

Table A.5 – ACPM state table

	STA0 Idle-Unassoc	STA1 Awaiting AARE	STA2 Awaiting A-ASCrsp	STA3 Awaiting RLRE	STA4 Awaiting A-RLSrsp	STA5 Associated	STA6 Collision assoc-init	STA7 Coll rsp1	STA8 Coll rsp2
A-ASCreq	p1 AARQ STA1								
A-ASCrsp+			AARE+ STA5 (Note)						
A-ASCrsp–			AARE– STA0						
AARQ	p1 A-ASCind STA2 ^p1 AARE– STA0								
AARE+		A-ASCcnf+ STA5							
AARE–		A-ASCcnf– STA0							
SupportCnf		A-ASCcnf– STA0							
A-DTreq		A-DT STA1	A-DT STA2			A-DT STA5			
DT			(Note) STA2			A-DTind STA5			
A-ACRQreq		ACRQ STA1				ACRQ STA5			
A-ACRQrsp						ACRP STA5			
ACRQ			(Note) STA2			A-ACRQind STA5			
ACRP						A-ACRPind STA5			
A-RLSreq						RLRQ STA3			
A-RLSrsp+					RLRE+ STA0		RLRE+ STA3	STA8	
A-RLSrsp–					RLRE– STA5				
RLRQ				p2 A-RLSind STA6 ^p2 A-RLSind STA7		A-RLSind STA4			
RLRE+				A-RLScnf+ STA0				A-RLScnf+ STA4	A-RLScnf+, RLRE+ STA0
RLRE–				A-RLScnf– STA5					
A-ABRreq		ABRT STA0	ABRT STA0	ABRT STA0	ABRT STA0	ABRT STA0	ABRT STA0	ABRT STA0	ABRT STA0
ABRT[a)]		A-ABRind STA0	A-ABRind STA0	A-ABRind STA0	A-ABRind STA0	A-ABRind STA0	A-ABRind STA0	A-ABRind STA0	A-ABRind STA0
supporting ABRT		SupABind STA0	SupABind STA0	SupABind STA0	SupABind STA0	SupABind STA0	SupABind STA0	SupABind STA0	Sup ABind STA0
EXTRN-1				STA5		STA5			
EXTRN-2					STA5	STA5			

a) When supported by version 1 of the session protocol (ISO/IEC 8327), the A-ABORT APDU has no APCI. The receipt of the P-U-ABORT indication implies its existence.

NOTE – The DATA and ALTER CONTEXT indications are queued by the receiving ACPM and only delivered when a A-ASCrsp+ is received.

Annex B[3)]

Authentication-mechanism using password

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

B.0 Introduction

This annex specifies a simple authentication-mechanism that uses a password with an AE-Title. This authentication-mechanism is intended for general use. It is also an example of an authentication-mechanism specification.

B.1 Assigned name

The following name (of ASN.1 datatype OBJECT IDENTIFIER) is assigned to this authentication-mechanism:

```
{ joint-iso-ccitt
    association-control (2)
    authentication-mechanism (3)
    password-1 (1)
  }
```

B.2 Authentication-value ASN.1 datatype

For this authentication-mechanism, the password is the authentication-value. The data type of authentication-value shall be "GraphicString" in accordance with the production for "authentication-value" in clause 9.

B.3 Processing specification

In this annex, the term "sending" denotes the AEI (or its authentication-function) requesting authentication by its peer. The term "receiving" denotes the AEI (or its authentication-function) performing authentication of its peer.

B.3.1 Requesting authentication

The sending authentication-function retrieves a password value from stored data for its AEI to be corroborated by the receiving AEI. The password value is mapped to the datatype of the authentication-value defined in B.2.

When the A-ASSOCIATE request or response primitive is issued by the sending AEI, the Authentication-value parameter shall contain this value. The primitive shall also contain the appropriate AP-title and AE-qualifier parameters that indicates its AE-title.

Depending on prior agreements between the sending and the receiving AE, the authentication-mechanism name (defined in B.1) may or may not be included on the A-ASSOCIATE primitive.

B.3.2 Performing authentication

The receiving authentication-function receives the Authentication-value parameter value on the incoming A-ASSOCIATE indication or confirm primitive.

Depending on prior agreements between the sending and the receiving AE, the authentication-mechanism Name (defined in B.1) may or may not be included on the A-ASSOCIATE primitive.

If an authentication-mechanism Name is required but not received, the authentication-function indicates that an A-ABORT request primitive shall be issued. The Diagnostic parameter value shall indicate "authentication-mechanism Name required."

If an authentication-mechanism Name is included, it shall be semantically equivalent to that specified in B.1. If this authentication-mechanism Name is not correct, the authentication-function indicates that an A-ABORT request primitive shall be issued. The Diagnostic parameter value shall indicate "authentication-mechanism Name not recognized."

---

3) The implementation of the authentication-mechanism specified in this annex is not required for conformance to this Protocol Specification. However, if employed, the entire specification is binding for the authentication-functions in both AEIs.

The authentication-function then determines if this authentication-mechanism (i.e. the authentication-mechanism defined in this annex) is allowed for the sending AEI based on the AE-Title of the sending AEI. If this authentication-mechanism is not allowed, the authentication-function indicates that an A-ABORT request primitive shall be issued. The Diagnostic parameter value shall indicate "authentication failure".

If this authentication-mechanism is allowed for the sending AEI, the authentication-function compares the value of the Authentication-value parameter against its stored data for this mechanism based on the sender’s AE-title. If the two are semantically equivalent, the authentication-function shall indicate successful authentication.

If the two values are not semantically equivalent, the authentication-function indicates that an A-ABORT request primitive shall be issued. The diagnostic parameter value shall indicate "authentication failure".

Annex C

Definition of the IA-service mapping to the Presentation service

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

C.1 Procedures for Lower Boundary Mapping the Presentation service

This annex defines the specific lower boundary mapping to be used when using the Presentation service as a supporting service, as well as specific constraints on the use of Session and Presentation. This annex constitutes a specific application of clause 8 of the ACSE Protocol Specification and defines the lower mapping of ACSE to the Presentation service.

C.2 Use of the Presentation service

C.2.1 General

When the Presentation service is used as the supporting service for ACSE, there is a one-to-one correspondence between the ASO-association and a presentation-connection. The presentation-connection is established at the same time as the ASO-association, and the normal or abnormal termination of the presentation-connection results in the termination of the ASO-association.

The IA-services representing the use of the supporting service by ACSE uses the P-CONNECT, P-RELEASE, P-U-ABORT and P-P-ABORT services. The IA-service shall be the sole user of these services. Other presentation services are not used, nor is their use constrained. However, the A-RELEASE procedure is disrupted by a request or indication primitive of the P-RESYNCHRONIZE, P-U-EXCEPTION-REPORT or P-P-EXCEPTION-REPORT services.

NOTE – This specification covers only "normal-mode" ACSE. For the specification of ACSE X.410 (1984) and the mapping to X.410-mode Presentation, see ITU-T Rec. X.227 | ISO/IEC 8650-1.

C.2.2 Nested associations

A nested ASO-association has a one-to-one correspondence with a nested presentation-connection. The nesting presentation-connection of this nested presentation-connection shall be the presentation-connection that supports the ASO-association which is the context of the A-ASSOCIATE request for the new nested ASO-association.

The ACPMs of both nested and nesting associations use the normal mode of the presentation-service (see ITU-T Rec. X.216 | ISO/IEC 8822) with the presentation-service Nested-connection functional unit (see ITU-T Rec. X.216/Amd.2 | ISO/IEC 8822/Amd.2). The use of additional presentation-service functional units is an ACSE service-user choice. This choice does not affect the operation of the ACPM.

C.3 Use of the Session service

C.3.1 General

Session version 2 (or higher) shall be used.

NOTE – For the use of ACSE with Session version 1, see ITU-T Rec. X.227 | ISO/IEC 8650-1.

The session functional units required for the session-connection that supports the presentation-connection (that in turn supports the association) are determined by the A-ASSOCIATE service requestor and acceptor. They accomplish this using the Session requirements parameter on the A-ASSOCIATE primitives. The session functional units are described in ITU-T Rec. X.215 | ISO/IEC 8326.

An A-ASSOCIATE request to establish an application-association that is intended to be the nesting-association for one or more nested ASO-associations shall include the session Nested-connection functional unit in the Session requirements parameter. An established application association shall only be available for use as a nesting-association if the Nested-connection functional unit has been selected on the underlying session connection.

The rules of the session-service affect the operation of the ACPM and its service-user. The ACSE service-user must be aware of these constraints. This Specification assumes that a local mechanism enforces them. Some examples of session-service constraints that affect the ACSE service-user are:

a) the availability of negotiated release; and

b) the possibility of release collisions.

C.3.2 Disruption of A-RELEASE by external event

The normal release procedure is disrupted by the occurrence of an external event (external to the ACPM) resulting from a service primitive of one of the following services:

a) P-RESYNCHRONIZE; or

b) P-U-EXCEPTION-REPORT; or

c) P-P-EXCEPTION-REPORT.

When the normal release procedure is disrupted, the association is again available for use as though the A-RELEASE procedure had not been invoked.

C.4 Mapping to Presentation service

This subclause specifies the mapping of the IA-service supporting normal-mode ACSE to the Presentation service.

C.4.1 IA-BIND-REQUEST.submit

C.4.1.1 Nested association

If the IA-BIND-REQUEST.submit primitive is for a nested association a P-CONNECT request is invoked to establish a nested presentation-connection on the presentation-connection supporting the nesting ASO-association.

The addressing parameters of the IA-BIND-REQUEST.submit primitive are not used in this case.

NOTE – The addressing parameters of the IA-BIND-REQUEST.submit will be equivalent to the addressing parameters of the P-CONNECT primitive that established the outermost nesting presentation-connection.

The other presentation parameters of the IA-BIND are mapped to the corresponding parameters of the P-CONNECT request primitive. The User-information parameter (containing the AARQ) is mapped to the User data parameter of the P-CONNECT request. The requesting ACPM waits for a P-CONNECT confirm primitive from the supporting service and does not accept any other primitive from the requestor other than an A-ABORT request primitive. The ACPM is in the Awaiting AARE (STA1) state.

C.4.1.2 Application-association

If the IA-BIND-REQUEST.submit primitive is issued from an ACPM in the AE or the primitive is issued from an ACPM in an inner ASO and the CF of the enclosing ASO determines that the new association shall be an application-association (i.e. un-nested), then a P-CONNECT request is invoked to establish a new, un-nested presentation-connection using the addressing parameters provided in the IA-BIND request.

The parameters of the IA-BIND are mapped to the corresponding parameters of the P-CONNECT request primitive. The User-information parameter (containing the AARQ) is mapped to the User data parameter of the P-CONNECT request. The Mode parameter shall have the value "normal". The requesting ACPM waits for a P-CONNECT confirm primitive from the supporting service and does not accept any other primitive from the requestor other than an A-ABORT request primitive. The ACPM is in the Awaiting AARE (STA1) state.

C.4.2 IA-BIND-REQUEST.deliver (P-CONNECT indication)

When the supporting service generates a P-CONNECT indication primitive in which the Mode parameter has the value "normal", the primitive and its parameters are mapped to an IA-BIND-REQUEST.deliver.

C.4.3 IA-BIND-RESPONSE.submit

The primitive shall be mapped to a P-CONNECT response primitive to confirm the P-connection using the parameters provided in the IA-BIND-RESPONSE.submit primitive. The User-information parameter (containing the AARE) is mapped to the User data parameter of the P-CONNECT response. The Mode parameter shall have the value "normal".

C.4.4 IA-BIND-RESPONSE.deliver

When the supporting service generates a P-CONNECT confirm primitive in which the Mode parameter has the value "normal", the primitive and its parameters are mapped to an IA-BIND-RESPONSE.deliver

C.4.5 IA-DATA.submit

NOTE – This service is only used if the Higher Level Association functional unit is in use on the supported association.

The CF maps this primitive to a P-DATA request primitive. Because the P-service is blocking, i.e. no data will be sent until the association establishment completes and the ACSE service is not blocking, the CF must queue the IA-DATA until the supporting service, in this case the Presentation service, transitions to the ESTABLISHED state.

C.4.6 IA-DATA.deliver (P-DATA indication, P-RESYNCHRONIZE, P-U-EXCEPTION REPORT, P-P-EXCEPTION REPORT)

NOTE – This service is only used if the Higher Level Association functional unit is in use on the supported association.

This primitive is invoked when a P-DATA indication is received and the Higher Level Association functional unit is selected on the supported association. It is also issued when any Presentation indication or confirm not identified in other subclauses is received and the higher-level functional unit is selected.

The contents of the User data parameter are delivered to the supported ACPM.

C.4.7 A-ALTER-CONTEXT-REQUEST.submit

NOTE – This service is only used if the Higher Level Association functional unit is selected on the supported association.

A P-ALTER-CONTEXT request primitive is invoked using the parameters of the IA-ALTER-CONTEXT-REQUEST.submit primitive.

C.4.8 IA-ALTER-CONTEXT-REQUEST.deliver (P-ALTER-CONTEXT indication)

NOTE – This service is only used if the Higher Level Association functional unit is selected on the supported association.

This primitive is invoked when an P-ALTER-CONTEXT indication is received and the supporting service is in the ESTABLISHED state to notify the using ACSE that the P-context of the supporting service has been changed.

When this primitive is invoked, an IA-ALTER-CONTEXT-RESPONSE.submit primitive to indicate that the recipient ASO accepts the change to the P-context of the supporting service.

C.4.9 IA-ALTER-CONTEXT-RESPONSE.submit

NOTE – This service is only used if the Higher Level Association functional unit is selected on the supported association.

The P-ALTER-CONTEXT-RESPONSE.submit primitive is invoked to confirm (positively or negatively) the change the P-context of the supporting service.

C.4.10 IA-ALTER-CONTEXT-RESPONSE.deliver (P-ALTER-CONTEXT confirm)

NOTE – This service is only used if the Higher Level Association functional unit is selected on the supported association.

This primitive is invoked when an P-ALTER-CONTEXT confirm primitive is received to notify the A-ALTER CONTEXT request the result of the request.

C.4.10.1 IA-ALTER-CONTEXT Parameters

The parameters of these primitives consist of the naming parameters of the A-ALTER-CONTEXT primitives.

C.4.11 IA-ABORT.submit

This primitive is mapped to a P-U-ABORT request.

C.4.12 IA-ABORT.deliver (P-U-ABORT indication)

When the supporting service generates P-U-ABORT indication, the primitive and its parameters are mapped to an IA-ABORT.deliver. The user data will contain an ABRT APDU.

C.4.13 IA-RELEASE-REQUEST.submit (P-RELEASE request)

The primitive shall be mapped to a P-RELEASE request primitive.

NOTE – The change of state in the Presentation service (and protocol) is not reflected in the simplified state machine of the IA-service.

C.4.14 IA-RELEASE-REQUEST.deliver (P-RELEASE indication)

When the supporting service generates a P-RELEASE indication primitive, the primitive and its parameters are mapped to an IA-RELEASE-REQUEST.deliver.

C.4.15 IA-RELEASE-ACCEPT.submit [P-RELEASE (result = affirmative) response]

The primitive shall be mapped to a P-RELEASE response primitive with a Result parameter set to "affirmative".

C.4.16 IA-RELEASE-ACCEPT.deliver [P-RELEASE confirm (accepted)]

When the supporting service generates a P-RELEASE indication primitive in which the Result parameter has the value "affirmative", the primitive and its parameters are mapped to an IA-RELEASE-REQUEST.deliver.

C.4.17 IA-RELEASE-REFUSE.submit [P-RELEASE (result = negative) response]

The primitive shall be mapped to a P-RELEASE response primitive with a Result parameter set to "negative".

C.4.18 IA-RELEASE-REFUSE.deliver [P-RELEASE confirm (accepted)]

When the supporting service generates a P-RELEASE indication primitive in which the Result parameter has the value "negative", the primitive and its parameters are mapped to an IA-RELEASE-REQUEST.deliver.

C.4.19 A-UNBIND.submit

No action is taken.

NOTE – The Presentation connection will have been terminated by the event which triggered the IA-UNBIND.submit.

C.4.20 IA-UNBIND.deliver (P-P-ABORT indication)

When the supporting service generates a P-P-ABORT indication, the primitive and its parameters are mapped to an IA-UNBIND.deliver.

Annex D

Definition of the IA-service mapping to ACSE

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

D.1 Procedures for lower boundary mapping to ACSE or the Presentation service

This subclause defines the specific lower boundary mapping when using the ACSE as a supporting service used with the Higher Level Association functional unit. This annex constitutes a specific application of clause 8 of the ACSE Protocol Specification and defines the lower mapping of ACSE to another ACSE instance of service.

D.2 IA-BIND-REQUEST.submit

D.2.1 When Invoked

This primitive is invoked by an initiating ACSE to request an instance of the supporting service for the ASO-association that it is attempting to create. The initiating ACPM will be in the Awaiting AARE (STA1) state. The supporting service may be in any state except RELEASE PENDING.

D.2.2 Action upon Receipt

When this primitive is received, the CF inspects the naming parameters of the IA-BIND request:

If the ASO-name does not name this ASO, the CF creates a binding between the requesting ASO and this ASO's parent and invokes an IA-BIND request with the same parameters as those received in the IA-BIND request that initiated this procedure.

If the ASO-name does name this ASO or there are no naming parameters nor a Presentation-address, then this ASO is being requested to provide supporting service for a new ASO-association being created by its child ASO.

In this case, the ASO has the choice of creating a new instance of service or mapping this request to an existing ASO-association of its own:

1) If it chooses to create a new instance of service, an A-ASSOCIATE request is invoked to create a supporting ASO-association with the ASO described in the naming and addressing parameters passed in the IA-BIND request. The Presentation context definition list parameter of this primitive is set to the P-context required to support the AARQ contained in the IA-BIND request User-information. It is a decision in the design of the ASO whether the IA-BIND request User-information (containing the AARQ) is mapped to the User-information field of the A-ASSOCIATE request or the CF waits for the supporting association to be established and then maps the IA-BIND request User-information to the User data field of an A-DATA request. In most cases, the former will occur.

2) If it chooses to map this to an existing instance of service, then the supporting service may be in the Awaiting AARE (STA1) state or the Associated (STA5) state[4)].

   If the P-context for the IA-BIND request is NOT part of the Defined Context Set for the supporting association, then:

   An A-ALTER-CONTEXT is invoked requesting the P-context for this association. It is a decision in the design of the ASO whether the User-information parameter of the IA-BIND request (containing the AARQ) is placed in the User-information parameter of the A-ALTER-CONTEXT request or mapped to an A-DATA request.

   The resulting primitive is invoked.

   If the P-context for the protocol of the IA-BIND request is contained in the Defined Context Set for the supporting association, then:

   The User-information parameter of the IA-BIND request (containing the AARQ) is placed in the User-information parameter of an A-DATA request and the primitive is invoked.

---

4) If the supporting service is either the Awaiting RLRE (STA3) state or the Collision-association-initialization (STA6) state, the CF cannot use this instance of the service to support the new association.

NOTE – If the ACSE is in STA1 state, then the Defined Context Set will not have been confirmed. The CF should act on the assumption that the P-context proposed in the outstanding AARQ will be accepted. When the supporting ACSE is in the STA1 state, the CF can always wait until the association request has either succeeded (and ACSE is now in STA5 and may send the AARQ), or failed (and ACSE is now in STA0) in which case the point is moot.

If there are ASO-naming parameters, but there is a Presentation-address, see D.2.

D.3 IA-BIND-REQUEST.deliver (A-ASSOCIATE indication)

D.3.1 When Invoked

This primitive is invoked to notify a recipient ACPM that a new distinct instance of the supporting service capable of delivering APDUs to this ACPM is being requested. The recipient ACPM will be in the Idle-Unassociated (STA0) state. The supporting service will be in the ASSOCIATION PENDING state.

D.3.2 Action upon Receipt

When this primitive is invoked, the recipient ACPM will determine if it is willing to accept this new supporting service. The User-information parameter may contain an APDU that will be delivered to the appropriate ASE/ASO[5)]. If this is an AARQ, then the naming fields are inspected and the local directory function invoked to determine the ACPM to receive this AARQ. If the naming fields did not specify an ASOI, then an ASOI is created and the AARQ delivered to it. If it did specify an ASOI, then the AARQ is delivered to the ASOI. (The handling of collisions is specific to the design of the ASO.) The ACPM will invoke an IA-BIND-RESPONSE.submit primitive to indicate the acceptance or rejection of the instance of the supporting service.

D.4 IA-BIND-RESPONSE.submit

D.4.1 When Invoked

This primitive is invoked by a responding ACPM to confirm or deny a request for a supporting service. The responding ACPM will be in the Associated (STA5) state, and the supporting service may be in the ASSOCIATION PENDING state or the ESTABLISHED state.

D.4.2 Action upon Receipt

When this primitive is received, the CF will perform one of the following actions:

1) If the supporting service is in the Awaiting A-ASCrsp (STA2) or equivalent state, then an A-ASSOCIATE response is invoked to respond to the creation an ASO-association using the naming parameters in the A-ASSOCIATE indication. The Presentation context definition list parameter of the A-ASSOCIATE response is set to the P-context required to support the higher level association being confirmed by the IA-BIND response User-information. It is a decision in the design of the ASO whether the IA-BIND response User-information (containing the AARE) is mapped to the User-information parameter of the A-ASSOCIATE response or the CF waits for the supporting association to be established, i.e. is in the STA5 state, and then maps the IA-BIND response User-information to the User data parameter of an A-DATA request. In most cases, the former will occur.

   NOTE – The supporting service must be able to discard any data that arrives after an association request has been refused.

2) If the supporting service is in the Associated (STA5) state, then the User-information parameter (containing the AARE) of the IA-BIND-RESPONSE.submit is mapped to the User data field of an A-DATA-REQUEST.submit, or equivalent service of the parent ASO.

3) The supporting service cannot be either in the Awaiting A-RLSrsp (STA4) state, or in the Collision-association-initialization (STA6) state or the Collision-associate-response (STA7) state. In this case, the CF should either create a new instance of the supporting service [see 1) and 2) above] or discard the IA-BIND response and generates an IA-ABORT.deliver

5) This would frequently be an AARQ, but might be another APDU if the supported association is already established.

D.5 IA-BIND-RESPONSE.deliver (A-ASSOCIATE confirm)

D.5.1 When Invoked

When the supporting service receives an A-ASSOCIATE confirm primitive, the primitive and its parameters are mapped to an IA-BIND-RESPONSE.deliver. This primitive is invoked when the ACPM is either in the Awaiting RLRE state (STA3), or the Associated (STA5) state.

D.5.2 Action upon Receipt

If the supporting service has been accepted, the supporting service transitions to the Associated (STA5) state and the ACPM is able to use this instance of the supporting service. The User-information parameter may contain an APDU that will be delivered to the appropriate ASO[6)].

D.6 IA-DATA.submit

D.6.1 When Invoked

This primitive may be invoked when ACPM is in the Awaiting AARE (STA1) or Associated (STA5) state and the supporting service is in the ASSOCIATION PENDING or ESTABLISHED states.

D.6.2 Action upon Receipt

This primitive is mapped to the User data parameter of an A-DATA.submit primitive of the Supporting ACSE or equivalent primitive.

D.7 IA-DATA.deliver (A-DATA.deliver)

D.7.1 When Invoked

This primitive is invoked when the supporting service is in the ESTABLISHED state and has received an A-DATA.submit.

D.7.2 Action upon Receipt

The contents of the User data parameter are delivered to the supported ASO.

D.8 IA-ALTER-CONTEXT-REQUEST.submit

D.8.1 When Invoked

This primitive may be invoked when the ACPM is in the Awaiting AARE (STA1) or Associated (STA5) state and the supporting service is in the ASSOCIATION PENDING state or the ESTABLISHED state and it wishes to modify the ASO-context or the Presentation context of the supporting service.

D.8.2 Action upon Receipt

An A-ALTER-CONTEXT-REQUEST.submit primitive is invoked using the parameters of the IA-ALTER-CONTEXT-REQUEST.submit primitive.

D.9 IA-ALTER-CONTEXT-REQUEST.deliver (A-ALTER-CONTEXT-REQUEST.deliver)

D.9.1 When Invoked

This primitive is invoked when an A-ALTER-CONTEXT-REQUEST.deliver is received and the supporting service is in the ESTABLISHED state to notify the using ACSE that there has been a request to change the ASO-context or P-context of the supporting service.

---

6) This would frequently be an AARE, but might be another APDU if the supported association is already established.

D.9.2 Action upon Receipt

When this primitive is invoked, the ACPM invokes an IA-ALTER-CONTEXT-RESPONSE.submit primitive to indicate that the recipient ASO accepts of the change in ASO-context or P-context of the supporting service.

D.10 IA-ALTER-CONTEXT-RESPONSE.submit

D.10.1 When Invoked

This primitive may be invoked when the ACPM is in the Associated (STA5) state and the supporting service is in the ESTABLISHED state to notify the supported service of a modification in the contexts supported by the supporting service.

D.10.2 Action upon Receipt

The A-ALTER-CONTEXT-RESPONSE.submit primitive is invoked to confirm (positively or negatively) the change in the ASO-context or in the P-context of the supporting service.

D.11 IA-ALTER-CONTEXT-RESPONSE.deliver (A-ALTER-CONTEXT-RESPONSE.deliver)

D.11.1 When Invoked

This primitive is invoked when an A-ALTER-CONTEXT-RESPONSE.deliver primitive is received to notify the A-ALTER-CONTEXT request the result of the request. The ACPM is in the Associated (STA5) state and the supporting service is in the ESTABLISHED state.

D.11.2 Action upon Receipt

If the request was accepted, the ASO can assume that the context change has been made. If the request was rejected, the ASO assumes that the context is unchanged.

D.11.3 IA-ALTER-CONTEXT Parameters

The parameters of these primitives consist of the naming parameters of the A-ALTER-CONTEXT primitives.

D.12 IA-ABORT.submit

D.12.1 When Invoked

An IA-ABORT request may be invoked when the using ACSE or the supporting service is in any state. This primitive is invoked to notify the supporting service that the supported association has terminated.

D.12.2 Action upon Receipt

Any action by the supporting service depends on the design of the supporting ASO. The supported ACPM transitions to the Unassociated (STA0) state.

NOTE – For the ACPM to abort an instance of the supporting service without affecting the supported association, the ACPM generates an A-ABORT request.

D.13 IA-ABORT.deliver (A-ABORT indication)

D.13.1 When Invoked

This primitive is invoked when the supporting service generates an A-ABORT indication to notify the ACPM that the supporting service has aborted the instance of the supporting association.

D.13.2 Action upon Receipt

The supporting service transitions to the NULL state. The User-information parameter is delivered to the appropriate component of the ASO. It is a design decision in creating the CF, if the supported association is maintained. If so, then no action is taken. However before any APDUs can be sent on this association, an instance of the supporting service must be re-established. If the association is not maintained, the ACPM transitions to the Idle-Unassociated (STA0) state.

D.14 IA-RELEASE-REQUEST.submit

D.14.1 When Invoked

This primitive is invoked to notify the supporting service of the request for the release of the supported association. The ACPM may be in the Associated (STA5) state.

D.14.2 Action upon Receipt

The CF maps the User-information parameter (containing the RLRQ) to the User data parameter and invokes an A-DATA.submit primitive. No action is taken by the supporting service.

NOTE 1 – The IA-RELEASE-REQUEST is used here as a combination of an IA-DATA and an IA-RELEASE. Some ASOs may be designed such that these are distinct operations, or such that only an IA-DATA is generated, i.e. it is not necessary to notify the supporting service of the release of the supported service.

NOTE 2 – RLRQ APDUs for associations supported by ACSE are delivered by A-DATA.deliver primitives.

D.15 IA-RELEASE-ACCEPT.submit

D.15.1 When Invoked

This primitive is invoked to notify the supporting service that the request for the release of the supported association has been accepted. The ACPM may be in the Awaiting A-RLSrsp (STA4) state.

D.15.2 Action upon Receipt

When this primitive is generated and the ACPM is in the Awaiting A-RLSrsp (STA4) state, the User-information parameter if present (containing an RLRE) is mapped to an A-DATA.submit primitive which is invoked and the ACPM transitions to the Idle-Unassociated (STA0) state. The ACPM invokes an IA-UNBIND.submit.

D.16 IA-RELEASE-REFUSE.submit

D.16.1 When Invoked

This primitive is invoked to notify the supporting service that the request for the release of the supported association has been refused. The ACPM may be in the Awaiting A-RLSrsp (STA4) state.

D.16.2 Action upon Receipt

When this primitive is generated and the ACPM is in the Awaiting A-RLSrsp (STA4) state, the User-information parameter if present (containing an RLRE) is mapped to an A-DATA.submit primitive which is invoked and the ACPM transitions to the Associated (STA5) state.

NOTE – RLRE APDUs for associations supported by ACSE are delivered by A-DATA.deliver primitives.

D.17 IA-UNBIND.submit

D.17.1 When Invoked

This primitive is invoked to notify the supporting service of the release of a supported service. The ACPM may be in the Awaiting RLRE (STA3) or Awaiting A-RLSrsp (STA4) state.

D.17.2 Action upon Receipt

When this primitive is generated and the ACPM is in the Awaiting RLRE (STA3) state, the ACPM transitions to the Idle-Unassociated (STA0) state and notifies the supporting service that the supported association has been released.

If the ACPM is in the Awaiting A-RLSrsp (STA4) state, then the RLRE is mapped to the User data parameter of the A-DATA.submit primitive and it is invoked. The ACPM transitions to the Idle-Unassociated (STA0) state.

Any action by the supporting service depends on the specification of that ASO, i.e. it may or may not release the supporting association.

D.18 IA-UNBIND.deliver

D.18.1 When Invoked

This primitive is invoked to notify the supported service that the supporting service has been released.

D.18.2 Action upon Receipt

When the ACPM receives an IA-UNBIND.deliver, it is notified that the supporting association has been released. The User-information parameters is delivered to the appropriate component of the ASO. The ACPM cannot send any further APDUs on this association without first initiating a new instance of the supporting service. It is a matter of the design of the ASO as to whether the ACPM terminates, or waits for further action by its user for a new association, or the ASO initiates a new association.

Annex E

Guidance on the use of Higher Level Association functional units

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

E.1 The application layer structure

The specification of the behaviour of an entity in the application layer has traditionally been provided by a number of documents (including ACSE), carefully engineered together to perform the required task over one or more independent presentation connections. Each document provided the specification of an ASE (Application Service Element) and the totality for any one particular application formed an Application Entity.

In the application structure defined in ITU-T Rec. X.207 | ISO/IEC 9545, the concept of an application-service-object (ASO) was defined to promote the re-usability of specifications and to give a more flexible framework within which to develop application layer specifications. This Recommendation | International Standard allowed two or more ASEs to be combined, together with a control function (CF) specification to coordinate their operation to form an application-service-object (ASO). This specification distinguishes between ASOs that contain ACSE (Edition 2 or Edition 3), and those that do not. The former are called "ACSE-supported ASOs". This specification is not concerned with the latter[7)].

An ASO is intended to be an object[8)] in the sense that details of the inner structure or working are not visible to outside observers, only events at its outer boundary are visible. There are two classes of such events: the receipt and issue of service primitives at its upper boundary, reflecting the service that it provides, and the receipt and issue of service primitives at its lower boundary, reflecting its use of services leading to network activity. An Application Entity is an ACSE-supported ASO where the events at the lower boundary issue and receive P-service primitives.

The general structure permits ASOs and ASEs to be combined with a CF to form a new ASO, to any depth.

E.2 Support for association establishment by an embedded ASO

There are a number of possibilities that can arise in relation to requests for associations from an embedded ACSE-supported ASO. The options relate to whether the embedded ASO is supported by ACSE, Edition 2 or ACSE, Edition 3, and to whether, on the new association, the embedded ASO will require a full range of Presentation services (including synchronization services), or whether its needs are limited to the services of establishment, abort, release, and normal data transfer. For the former, we say that it requires a "full association" and for the latter, a "basic association". This is assumed to be known at specification time when the outer ASO definition is built. There is no difference in the functional handling of association establishment requests from an ASO supported by ACSE Edition 2 and those from an ASO supported by ACSE Edition 3. The detailed description in terms of primitives within the enclosing ASO from the embedded ASO does, however, differ in the two cases as ACSE Edition 2 is specified in terms of issuing presentation primitives while ACSE Edition 3 is specified in terms of issuing internal association (IA) primitives when it is embedded. This leads to a clearer description of what is happening, but does not affect the functionality available or the end-result.

There are two mechanisms that can be used to support requests for association establishment by inner ASOs. The first of these is to open up a new association at the outer level, wholly independent of any existing association, and supported by the normal OSI stack. This is clearly the only option if there is no existing association to the required destination. In general however, for reasons of both efficiency and synchronization, it is desirable to provide support for the embedding of associations within an existing presentation connection provided it is going to the correct destination. The other two mechanisms provide this support.

The new mechanism is called a "Higher Level Association" functional unit. In this case, the underlying presentation layer is unaware of the embedded association (seeing only a P-DATA using contexts established for the outermost association). The context negotiation and encoding is logically performed by the ACSE in the ASO enclosing

---

7) Such ASOs either use another mechanism for establishing ASO-associations or are not capable of establishing ASO-associations and must be combined with an ASE such as ACSE to provide that facility. ASEs which claim to support the concepts of ITU-T Rec. X.207 | ISO/IEC 9545 must specify which concepts are supported and how that support is provided.

8) An ASO is a more general concept than the notion of "object" used in the "object-oriented" environment. Although, ASOs for specific domains of applications could be specialized to have specific object-oriented characteristics.

the embedded ASO that requested the association. This mechanism can only be used to support requests for basic associations. Higher-level associations do not provide service primitives such as P-RESYNCHRONISE or P-ALTER-CONTEXT. (See Annex D for details on the use of this mechanism.)

With this mechanism it is, in principle, possible to provide the service over much simpler carriers than the full connection-oriented OSI stack, or to support ASO-association with greater scope than the supporting application associations.

E.2.1 Lower Boundary Service definitions

It is, in general, the case that the services required by a protocol are distinct from the service provided by the layer below. Thus, an integral part of a protocol specification is to define this lower boundary, and the mapping to the service provided by the layer below. The supporting service may be part of the protocol specification or described in separate documents. The protocol specification need not be modified to be used with a different supporting service. The new supporting service merely defines how it maps the lower service of the supported protocol to the service provided by the supporting service.

In the layers of the OSI Reference Model below, the Application Layer, the protocols procured supporting services from one and only one service. Presentation used the Session Layer and only the Session Layer; Session only used Transport, etc. Thus, most of these specifications did not distinguish these two boundaries, but simply defined the mapping of the PDUs of the supported (N + 1)-protocol to the service primitives of the supporting (N)-service.

These simplifications do not always exist in the Application Layer Structure. Not every Application Layer protocol is designed to exclusively use the services of the Presentation Layer. The requirement and the capability to re-use ASOs implies that an ASO may be used in a context with different supporting services (and protocols) which provide equivalent semantics to the ASO. This makes it necessary to separate the specification of the lower boundary of an ASE or ASO from a specific supporting service.

To accomplish this, the ACSE protocol specification defines the IA-service (where IA stands for Intermediate ACSE) as the lower boundary service it requires. This service definition (see clause 8) specifies when these service primitives are invoked and what action ACSE or the supporting service should take when they are received. Necessarily, the IA-service is able to specify the actions of ACSE in more detail than the actions of the supporting service. The IA-service specifies the behaviour of the supporting service in general terms that indicates the kind of action that should be performed. Other documents (in particular Annexes C and D) take this general definition and specify in detail the mappings to specific supporting services: Presentation and ACSE, respectively.

It is strongly recommended that all ASE and ASO specifications adopt this approach of defining a distinct lower boundary service definition that defines the services required by the ASE or ASO. ASO specifications define a detailed mapping of this lower layer boundary to a specific service definition. Most of these specific specifications will be part of the ASO specification that includes the ASE or ASO in its specification[9)].

E.2.2 Requests for the establishment of associations

An ASO must contain only one ACSE to support its needs for association establishment. This ACSE will be either ACSE Edition 2 or ACSE Edition 3. (Future editions of ACSE are outside the scope of this Recommendation | International Standard.)

If the embedded ASO is supported by ACSE Edition 2, such requests will always be expressed by the issue of a P-CONNECT at the lower boundary. If a P-CONNECT request is issued by an embedded ASO (ACSE Edition 2 supporting an embedded ASO), there is nothing to which that service primitive can be passed. The CF specification for any ASO which includes an ACSE-supported ASO which is using Edition 2 ACSE is required to include the following text:

> "When an embedded ASO issues a P-CONNECT from its lower boundary, this shall be interpreted as if the ASO had issued an IA-BIND-REQUEST from its lower boundary, with the "issuing-ASOI-identifier" parameter set to the "ASOI-identifier" of the ASO, and all other parameters mapped from the P-CONNECT in accordance with the parameter names. The resulting IA-BIND-REQUEST shall be delivered to the ACSE supporting this ASO."

Such text can be included without harm in all CF specifications. Note that the ACSE to which the IA-BIND-REQUEST is delivered is required to be a Edition 3 ACSE.

---

9) It has been recognized for some time that while most ASEs were defined to the Presentation service, it was desirable to allow these ASEs to be mapped to other but equivalent services. This lower boundary service definition provides a solution to this problem.

When the IA-BIND-REQUEST is delivered to a Edition 3 ACSE, it can take one of two actions, partly restricted by the nature (full or basic) of the association request and its destination and the existence or not of existing associations to the required destination, and is determined fully by the specification of the CF.

The two possible actions of an ACSE that is not the outer ACSE are as follows:

– To create an embedded ACSE-supported association, resulting in a P-DATA on an existing association (which may itself be an ACSE-supported embedded association, or may be a Nested Connection functional unit).

– To pass the request to the next outermost ACSE, resulting in an IA-BIND-REQUEST being issued to that ACSE, which then processes it as above in accordance with the restrictions identified earlier and the rules of its own CF. When that request has been satisfied, the resulting association can be assigned to satisfy the request from the embedded ASO.

The two possible actions of an outer ACSE are as follows:

– To create an embedded Higher Level Association functional unit, resulting in a P-DATA on an existing association (which may itself be a higher-level association, or may be a nested connection, or may be an outer-level normal association).

– To create a new outer-level normal association. This could then be assigned to satisfy the request from the embedded ASO, or either of the above two options can then be taken.

E.2.3 The Use of the Higher Level Association functional unit

This use of this approach can be considered as follows:

1) the application protocol may or may not use Session functional units other than Full Duplex over the application-association; or

2) The application protocols create ASO-associations that are not application-associations and, in general, do not use Session functional units other than Full Duplex; or

NOTE 1 – While strictly speaking, other Session functional units could be used in approaches 1) and 2) above, the CF of the parent ASO must be able to resolve any conflict that may occur among its child ASOs use and its own use of the Session Facilities. This could lead to a fair degree of complexity in the CF and is not recommended practice.

3) ASO-associations may or may not terminate at the same Application-Entity as the application-association.

NOTE 2 – Session functional units other than Full Duplex can not be used on such associations.

NOTE 3 – There are no restrictions of the creation of ASO-associations that are relayed by supporting ASO-associations through the use of the Higher Level Association functional unit. These application protocols may use ASN.1 or any other syntax scheme.

The Higher Level Association functional unit allows application protocols to be constructed that create ASO-associations with a scope greater than the scope of an application association. The design of the relaying schemes is specific to the design of those applications and is beyond the scope of this Recommendation | International Standard. The naming parameters in ACSE provide sufficient information for the construction of such functionality.

E.3 Concept of operations

The Higher Level Association functional unit is intended for use by ASOs that make little or no use of Session Facilities or provide comparable services by other means and/or make heavy use of higher level ASO-associations, especially for application layer relaying. The concept of operation of the higher level association builds on the same architectural concepts as those of the lower six layers, primarily the distinction of (N)-PCI and (N)-user-data and the addressing concepts described above. There are four primary capabilities that are available to higher level ASO-associations:

1) the ACSE Model;

2) the A-DATA APDU;

3) the negotiation of transfer syntax; and

4) the negotiation of ASO-context.

E.3.1 The ACSE Model

ACSE is an Application Service Element (ASE) that provides the establishment mechanism of an application protocol. ACSE should not be considered a protocol in and of itself, but a component used in combination with other ASE and ASO components to create a protocol. The application protocol designer has considerable flexibility in how ACSE is used and what it can be used to accomplish.

The lower layers are based on a connection model, see Figure E.1, while the application layer is based on the more primitive association model, see Figure E.2. The connection model can be seen as being composed of an (N)-association with local bindings between the (N+1)- and (N)-entities in the requesting and accepting systems. The upper and lower service primitives of ACSE manage this binding. (This is especially apparent in the IA-primitives, whose operations are directed at the supporting association rather than the supported association.) Strictly speaking, all PDUs are passed by data primitives. The control primitives, i.e. the non-data primitives, are used to manage the binding with the (N)-entities above or below. In many cases, a control and a data primitive are combined so that the PDU is passed as the User-information parameter of the Control primitive. The same model applies to the application layer except there are no (N)-SAPs. Application protocols can be designed to provide the connection model by adding the necessary semantics to the control function of the ASO. ASOs that require the ability to re-establish the supporting association without relinquishing the supported association can use ASOI naming fields and the appropriate CF behaviour to provide such a capability. It is beyond the scope of this annex to outline wide range of possible connection and association semantics that might be created by various ASO designs.

While all of the fields in the ACSE set of APDUs are well-defined, there is considerable flexibility in what they can be used to accomplish. ACSE is integrated with the other components to create an application protocol. It is recognized that while ACSE performs certain generic functions, a specific application may need to communicate similar application-specific information. For example, while the AARQ carries the generic information for establishing the association, there may also be application-specific initialization that must be exchanged. Each ACSE control PDU has a User-information parameter as opposed to a User data parameter to indicate that it is intended to contain application protocol-specific control information, i.e. part of the application protocol, rather than uninterpreted User data which may be some higher level application protocol. Clearly, there is no requirement that this be used for such application protocol specific functions and may be User data.

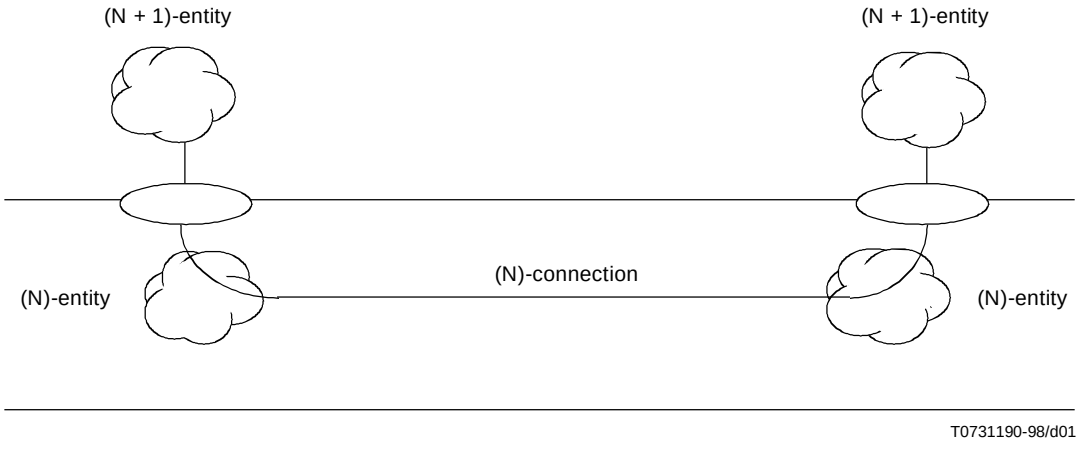

Figure E.1 – An (N)-connection is between (N + 1)-entities

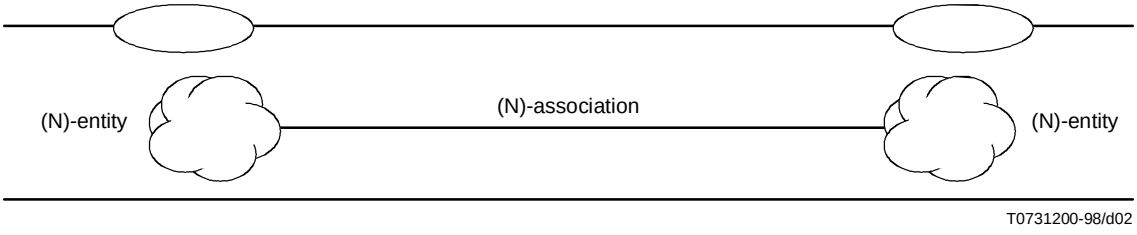

Figure E.2 – An (N)-association is between (N)-entities

E.3.2 The A-DATA APDU

The A-DATA APDU provides the means for an ASO to pass data transparently to a peer ASO.

The A-DATA APDU allows ASO-associations to pass data to other ASO-associations without requiring that they have any knowledge of the data being passed. The A-DATA APDU is used in the same way data PDUs are used in the lower layers to pass data transparently and to provide multiplexing of ASO-associations onto lower ASO-associations. The A-DATA APDU contains the necessary (and only the necessary) identifiers to distinguish an APDU for a given ASO-association from all other ASO-associations mapped to this supporting service. In the worst case, there are two identifiers required in an A-DATA APDU:

a) an ASO-qualifier to distinguish different ASOs, analogous to a protocol-identifier; and

b) an ASOI-identifier to distinguish different ASOIs, analogous to a connection-endpoint-identifier.

All of these identifiers are optional, only those identifiers that are required need to be present.

ACSE is not a layer. A-DATA is not used to encapsulate the data transfer phase components of the application protocol, but to encapsulate a higher-level application protocol. The use of A-DATA is especially important in the construction of ASOs that perform application layer relaying. Once again, the actual form of such relaying is determined by the ASO, not by ACSE.

E.3.3 Syntax Negotiation

To support the capabilities of higher level associations, especially relaying, the Higher Level Association functional unit extends the concept of transfer syntax negotiation. Recognizing the separation of PCI and User data and that a protocol cannot negotiate its own syntax, the Higher Level Association functional unit assumes that the Presentation Layer negotiates the syntax of the PCI of the protocol immediately above it and no more. ACSE then provides the means to negotiate the syntax of any protocols it encapsulates. Those encapsulated protocols negotiate the syntax of their encapsulated protocols and so on. Since encapsulated protocols may initiate a new protocol with a syntax not known before the supporting ASO-association was established, it is necessary to be able to notify the peer ASO of the new syntax. The A-ALTER-CONTEXT APDUs is provided for this purpose. It may be sent at anytime after the association establishment has been initiated to change either the presentation-context or the ASO-context (see below).

One of the elements of an application protocol specification defines the abstract syntax for the protocol. This abstract syntax defines the format of the APCI, not the User data of the protocol. There is only one abstract syntax for a protocol. However, this abstract syntax is composed of multiple abstract syntax modules. Generally, there is one such module for each component ASE in the protocol, although an ASE could have multiple syntax modules. (Since a CF is not allowed to generate APDUs, only an ASE can generate APDUs and thus have a syntax module associated with it.) These modules are distinguished by a presentation-context-identifier, so that when they are combined to form a protocol there is no ambiguity in the format of the APDUs. Since for an ASO containing ACSE, the service, component ASEs and ASOs, and CF are completely known at specification time, all of the syntax modules defining the APCI are known at specification time and the presentation-context-identifiers can be assigned at this time as well.

An ACSE-containing ASO may have different concrete syntaxes on an ASO-association at different times, or different concrete syntaxes on different ASO-associations at the same time. But different modules within the same complete protocol will not have different concrete syntaxes on the same ASO-association at the same time. All modules in the same complete protocol, i.e. the PCI of all APDUs, will have the same concrete syntax on a given association at any point in time[10]. Thus, only the concrete syntax is negotiated at association-establishment time.

It is useful to distinguish User-information and User data. User-information is application-specific APCI associated with the APDUs of one ASE that are specified by information (PCI) from another component ASE or ASO. User-information in ACSE is PCI that is not part of the ACSE module, but is part of the PCI of the application protocol of which ACSE is a component. User-data is not interpreted by ACSE or any other module of this protocol. For example, ACSE may include as User-information PCI from another component ASE in the AARQ APDU as application specific initialization information, or User-information on the ABRT APDU as application-specific Abort reasons. User-information occurs on all ACSE control PDUs, i.e. all APDUs except A-DATA, and is represented as "Sequence

---

10) This constraint applies only to the PCI i.e. the protocol, not the User data. This constraint is not required by ALS or by the Presentation Layer, but more by practical considerations. It is hard to fathom any practical advantage to removing the constraint, but it is clear that considerable complexity would result from removing the constraint.

of External". User-information can also be encoded as User data to allow APDUs of higher level ASO-associations to be carried on control APDUs, if so desired. However, the preferred practice is to use User-information for PCI of this protocol, rather than for User data, i.e. for an encapsulated protocol. If User-information is used to carry User data, this option should be defined by each application (ASO) specification explicitly indicating any constraints on which APDUs can be carried as User data on control APDUs.

Some applications contain information specific to that application which may be in a syntax different from the PCI that is not known at specification time. For example, an FTAM application or a query application could have file types or query responses that used syntaxes different from the PCI. These are not, strictly speaking, part of the protocol, but describe its User data. They are specific to the application and should be specified by an application specific parameter that specifies their syntax[11)].

Given the two kinds of application protocols discussed above, the following cases must be accommodated:

- When an application protocol, i.e. a parent ASO, is used only to encapsulate other application protocols, only the syntax of the PCI should be identified to maintain the separation of levels, i.e. only the concrete syntax need be negotiated.
- In an application protocol which does not (by intent) encapsulate other protocols, such as FTAM, it may be necessary to specify the abstract and concrete syntax of its User-information which may only be known at establishment time and may be changed during the life-time of the association. This can be accommodated, either by:
  - using the full syntax negotiation facilities of ACSE to describe abstract and concrete syntax; or
  - including a parameter in the application-specific ASE/ASO of the protocol, such as FTAM document types to indicate the syntax of the User data.

Thus, a complete ASO has three means for negotiating the syntax on the ASO-association: one is equivalent to the negotiation of P-context in the Presentation Layer in that it negotiates the P-context-id, abstract and concrete syntaxes; one negotiates only the concrete syntax; and one is an application specific field in the application protocol.

When the protocol negotiates the syntax of its User data, the scope of the negotiation is completely compatible between ACSE and Presentation.

- if the negotiation only specifies the concrete syntax, then it is only specifying the concrete syntax of the encapsulated APDUs;
- if the negotiation specifies the full P-context, then it is specifying the syntax of the PCI and User information and the User data.

It is important to note that if a complete application protocol is designed and implemented with the full P-context, but is later used to encapsulate another protocol with ACSE as User data, no difficulty is encountered. The syntax negotiation applies to the User data, which now simply happens to be APCI. If the embedded protocol uses ACSE to negotiate the syntax of its User data, then that negotiation takes precedence with respect to the User data over the negotiation of the lower level, just as scoping does in a programming language. That instantiation of ACSE will negotiate the syntax of the subsequent User data. If the underlying protocol is ACSE, it could automatically revert to negotiating only concrete syntax, or might continue with the full P-context form and simply offer fewer abstract and concrete syntaxes in its negotiation. If the underlying protocol is Presentation, then fewer abstract and concrete syntaxes will be offered consistent with the lesser scope of the negotiation.

User data may be encoded in one of two ways: the Simple or Full encoding used in Presentation. The simple and full encoding forms are used for true User data and would generally be used when encapsulating complete protocols. When the syntax negotiation facilities of A-ALTER-CONTEXT are used, it applies only to this User data. The rationale for this is that ACSE and its context negotiation facilities are part of the application protocol. The syntax of the application protocol is known to itself (and if negotiated, was negotiated by the level/layer below this one). The syntax that is not known and that may need to be negotiated, is the syntax of the User data.

---

11) Not only is this architecturally consistent, but it also facilitates the re-use of such modules while keeping the implementation strategy straightforward.

E.3.4 ASO-context

In an ASO specification, the service of the ASO, the control function, and the use of the supporting service are defined. (The child ASOs and their service definitions are referenced in the specification, but are defined elsewhere.) Thus, when an ASO-association is created, all of this information is known and is referenced in the protocol by the use of the ASO-naming facilities. However, an ASO may have multiple ASO-associations extant at the same time, each being used for potentially different purposes. Thus, it is necessary to negotiate at establishment time which ASO-association is being used for what purpose along with any other information associated with its use.

NOTE – There is a rather subtle distinction to be made here. ASO-associations with different protocols would be between distinct child ASOs. In this case, ASO-naming would be sufficient to establish the ASO-association and to distinguish its use. ASO-context is required when the same protocol is being operated over multiple ASO-associations according to some semantics important to the application. One might call it "semantic splitting" as opposed to the performance-related splitting found in the lower layers. Another view might be that ASO-naming can distinguish types of associations, i.e. have the same protocol operating over them, while ASO-context is necessary to assign uses to instances of the same type.

The ASO-context determines those aspects of the shared state that cannot be known at specification time and are known only at association establishment time (or may be changed during the lifetime of the association). Thus, the scope of an ASO-context is an ASO-association. The ASO-context is determined by the CF of the communicating ASOs[12)] and the role of the association. The ASO-context specification may also include an indication of certain policies and parameter values or ranges to be used on this association that are relevant to the application. The ASO-context has no direct bearing on the ASO-contexts of other ASO-associations that may be created by the child ASOs.

Changing the ASO-context may in rare circumstances also require a change in the syntax negotiated on the ASO-association. It is the responsibility of the CF to ensure that the appropriate syntax is also negotiated on the ASO-association. Since the ASO-context can only be modified after association establishment with ACSE, such changes will only affect the Presentation context as established by the Presentation Layer when:

1) this is the outermost ASO; and
2) the change in ASO-context affects the syntax of the Protocol-Control-Information or the User-information.

Changes in ASO-context or syntax to application protocols on higher level ASO-associations will have no affect on the negotiated presentation-context.

E.3.5 Naming and Addressing in the Application Layer

ALS provides a structure that allows the construction of distributed applications from a set of building blocks or modules. These modules, called ASOs or ASEs, are used to construct a complete application protocol to provide the establishment and data transfer phases of the protocol. An ASO consists of a combination of ASEs and ASOs, and a control function (CF), which moderates the interactions of the component ASEs and ASOs with the service provided and the supporting service. These ASO/ASEs can be embedded to any depth to create higher level associations which may have a scope greater than a single application association.

There are two basic forms of ASO: those that define a protocol for both the establishment and data transfer phases and those that define modules for the data transfer phase. ACSE is a standard ASE for providing the establishment phase. ALS only requires it to be used if the ASO in which it is contained is expected to operate as the outermost ASO, i.e. as an application-entity. However, using a non-standard ASE as the establishment mechanism requires the Application Layer subsystem to be able to parse more than one kind of establishment PDU and/or would require directory-objects unique to that ASO to be supported, thus impairing re-usability and increasing the complexity of the application infrastructure required.

To understand the use of names in ALS, it is first important to understand what needs to be named, when those names are needed and then how those names are used to establish ASO-associations. The ASO structure is a tree of ASOs with the application-entity being the root ASO. The ALS naming structure is a hierarchical name that reflects the ASO structure of the AE by extending in a backward compatible manner the Application naming defined by ITU-T Rec. X.650 | ISO/IEC 7498-3, such that Application Process Titles and Application Entity Titles are the roots of the ASO-name. (The reader should be familiar with both ITU-T Rec. X.650 | ISO/IEC 7498-3 and ITU-T Rec. X.207 | ISO/IEC 9545, especially 5.9 and Table 1.)

ASO-associations may be established only to ASOs which have ASEs or ASOs that provide both the establishment and data transfer phases. To be addressable, an ASO must have an ASE/ASO that performs the establishment phase. In other words, only nodes in the tree of ASOs with an establishment ASE/ASO are addressable. Applications that use the

12) This implies the set of the child ASEs and ASOs constituting the parent ASO.

Nested Connection functional unit must terminate on the same AE as the application-association. Applications using the Higher Level Association functional unit are not subject to this constraint and may have a broader scope than a single application-association.

ALS recognizes two basic elements that must be distinguishable for communications: ASOs and ASO-invocations. This is analogous to the (N)-entities and (N)-entity-invocations in the lower layers. Like (N)-entity-invocations, ASOIs have a single (N)-association. ASO-associations are initiated to either ASOs or ASOIs and established to ASOIs. As in the lower layers, an establishment request may address an ASO [i.e. (N)-entity in the lower layers] or an ASOI [an (N)-entity-invocation], but once the association (connection) is established, the association (connection) is to an ASOI [an (N)-entity-invocation]. In most cases, an application will establish a new ASO-association to a peer ASO which will create a new instantiation of an ASOI. Establishing an ASO-association to an existing ASOI would be used most often when recovering a previously lost association, or when the higher level ASO-association had remained established and its supporting ASO-association has been released and is now being re-established.

NOTE – This can be done with ACSE, but the capability must be incorporated into the CF of the ASO.

As an aid in understanding, the elements that must be named within the Application Layer Structure have direct analoguous with the naming in the lower layers:

Lower Layer Concept	Is named by	Corresponds to
(N)-entity (N)-entity-invocation	(N)-address (N)-connection-endpoint-id	ASO-name ASOI-identifier

(N)-SAPS identify the bindings between (N)-entities and (N + 1)-entities, while (N)-connection-endpoint-identifiers identify (N)-entity-invocations. The ASO-name identifies the ASO and is analogous to the (N)-address, while the ASOI-identifier is analogous to the (N)-connection-endpoint-identifier of the lower layers. In the lower layers, it is assumed that (N)-entity-invocation addressing is local to a system and is consequently not supported by the Directory. Similarly, ASOI naming information is not inherently supported by the Directory. Therefore, applications that make use of ASOI addressing must either provide the support for it or define application specific directory object classes to support it.

Once the bindings between the peer ACPMs (creating an association) and the supporting service are established, the naming parameters on the APDUs can be reduced in their scope. For data PDUs to be routed to the appropriate ASO-invocation, the scope need only identify the APDU within the scope of the parent ASO and ASOI. Here too the analogy with the lower layers is helpful:

Lower Layer Concept	Corresponds to
(N)-protocol-id (N)-connection-id	ASO-qualifier ASOI-id

The modular structure of ALS and the re-use of ASOs will cause ASOs to not use globally unambiguous ASO-names or ASO-titles in their APDUs, but will use names relative to the context in which they are used, i.e. names which are unambiguous in that context. Thus, it is likely that shorter relative ASO-names (or an ASO-qualifier) and relative ASOI-tags (or a ASOI-id) will be used in many cases. When APDUs are sent on ASO-associations, the minimal information that must be included will depend on the ASO-structure. For an APDU to be deliverable to the intended ASOI, the identifiers required in any APDUs will depend on the ASO structure as follows:

An ASO with	Requires	To be unambiguous in the scope of
Single child in a single parent Multiple child ASOs in a single parent Single ASOI in a child ASO Multiple ASOIs in a child ASO	No identifiers An ASO-qualifier No identifiers An ASOI-id	The parent ASO The parent ASO The child ASO That child ASO

where all ASOs are complete protocols.

In general, one would expect these to occur in applications with decreasing frequency as one moves down the list. Note that it follows from the definition of Application Entity, i.e. it is the outermost ASO, that no ASO-qualifier is ever required on the APDUs of this protocol, but may be present if such an ASO is designed to be re-used as an embedded ASO. In general, the appropriate identifiers must be present on APDUs in the data transfer phase if there may be multiple ASOs or ASOIs to be instantiated at the same time. However, this constraint can be relaxed such that no identifiers are present on the APDUs for the first association, if the CF is designed to recognize and maintain the distinction. The CF can distinguish the first association as the null case, and then the subsequent associations by having unambiguous naming fields present. However, this does require the CF to duplicate some functionality of ACSE and is only recommended when multiple ASOs or ASOIs are rare.

With these preliminaries, it is now possible to consider the behaviour of ACSE in resolving names and initiating new associations. To do that, let us first review how connection establishment works in the lower layers.

To initiate a connection in the lower layers, the (N + 1)-layer invokes an (N)-Connect-Request service primitive with the destination (N)-address with which it wishes to establish a connection as well as other parameters providing information relevant to the nature of the connection and perhaps some User data in the form of an (N)-SDU:

Connect-Request(called-(N)-address, calling-(N)-address, other, (N)-SDU)

The (N)-layer recognizes the primitive as a request for a new instance of communication and creates a new (N)-entity-invocation, i.e. a new protocol state machine. This new (N)-entity-invocation is given the Connect-Request service primitive and interprets the parameters of the Connect-Request primitive [other than the (N)-SDU]. The (N)-entity-invocation must first determine where it will send the Connect-Request (N)-PDU it is about to generate. It invokes the (N)-directory-function giving it the destination (N)-address that was provided in the service primitive. [See ITU-T Rec. X.650 | ISO/IEC 7498-3, clause 10, for a description of these (N)-directory-functions.] The (N)-directory function returns a destination (N – 1)-address and source (N – 1)-address. The (N)-entity-invocation constructs a Connect-Request (N)-PDU with the (N)-PCI being formed as a function of the parameters of the Connect Request service primitive[13] and the (N)-User-Data is formed by the direct mapping of the (N)-SDU to the User data field of the (N)-PDU[14]. The addressing information is (N)-PAI in this (N)-PDU, it will designate the destination of the (N)-PDU.

Based on the information provided in the initial Connect-Request Service primitive and the results of the (N)-directory function, the (N)-entity-invocation generates a (N – 1)-Connect Service primitive to the source (N – 1)-address with the (N)-PDU generated by the (N)-entity-invocation as the (N – 1)-SDU:

Connect-Request(called-(N – 1)-address, calling-(N – 1)-address, other, (N – 1)-SDU)

And so it goes down through the layers.

And eventually on the end system specified by the dest-(N – 1)-address an indicate primitive is invoked:

Connect-Indication(called-(N – 1)-address, calling-(N – 1)-address, other, (N)-SDU)

The (N)-subsystem may have to inspect the protocol-id in the (N)-PDU contained in the (N)-SDU (if there is more than one protocol operating in this layer) to determine the type of (N)-entity the (N)-PDU should be delivered. [Depending on how the concepts are mapped to the implementation, resource allocation decisions will be made by the (N)-subsystem or the (N)-entity.] The primitive is delivered to that (N)-entity which creates a new (N)-entity-invocation and passes it the information from the indication primitive and the SDU. The (N)-entity-invocation invokes an (N)-directory-function with the dest-(N – 1)-address and the src-(N – 1)-address as well as local information. The (N)-directory-function returns the appropriate (N)-address to which the (N)-entity-invocation should be bound (if not already) and to which the service primitives it invokes to its service user are destined. The (N)-entity-invocation then makes the appropriate state transitions and, if no errors are detected, invokes a CONNECT indication primitive to the layer above:

Connect-Indication(called-(N)-address, calling-(N)-address, other, (N)-SDU)

Fundamentally, ALS works exactly the same way. However, rather than having (N)-addresses, the elements of an application-entity are addressed by ASO-names. The names for both ASOs and ASOIs are interpreted as a hierarchical sequence of component names that reflect the ASO structure and are used by ACSE to establish higher level ASO-associations with corresponding ASOs. Since ASOs may be built for re-use, they may be used in contexts other than as an AE or other as embedded ASOs in the context for which they were originally designed. The ability to use this flexibility is in part provided by the naming. Embedded ASOs may be passed relative ASO-names and invoke

---

13) It is important to remember that (N)-entities (and their invocations) are only able to understand the parameters of the service primitives and the (N)-PCI (is why they are called protocol-control-information and used to be interface-control-information). (N)-entities are not able to interpret either SDUs or (N)-user-data.

14) The contents of the SDU might be fragmented across subsequent (N)-PDUs.

supporting services based on relative ASO-names. A relative ASO-name consists of any contiguous subsequence of ASO-qualifiers in an ASO-name. The embedded ASO is, in effect, designed based on assumptions about the structure in which it is embedded, but not the specific ASOs (i.e. with specific names) in which they may operate. The ASO's relative name may not be the same, but only established at the time the CF of the outer ASO is written. As previously noted, ASO-names and ASOI-tags are not permanently assigned to an ASO, but are parameters that characterize its "location" in the ASO structure of the application-entity. ASO-qualifiers and ASOI-identifiers are not globally assigned or unambiguous. However, ASO-names and ASOI-tags are globally unambiguous.

For an ASO to initiate communication, a request is invoked by a user to an outer ASO, ultimately this comes from the Application Process to the Application-entity. This request may or may not have the form of an A-ASSOCIATE, but does provides sufficient information that the outer ASO can derive the necessary relative ASO-name(s) to cause inner ASOs to create ASO associations. Requests of this kind are propagated down through the ASO structure until they are issued to an ASO whose CF requires that an ASO-association be created. From information passed in this request, the CF is able to construct and generate an A-ASSOCIATE request service primitive with a relative address appropriate for the structure this ASO expects:

A-ASSOCIATE(called-rel-ASO-name, calling-rel-ASO-name, other, User-information)

The ASO-name information will most often represent a relative ASO-name. Most of these relative ASO-names will consist of a single element of the sequence, but may consist of more. The ACPM must process the information provided in the A-ASSOCIATE request. Now as in the lower layers, the ACPM must determine where to send the AARQ it is about to generate. The ACPM locally invokes the functionality of its CF to determine where to deliver this PDU. The CF invokes the Application Title Directory Facility to provide the ASO-names (and possibly ASOI-tags) of the supporting ASO. The ASO then forms an AARQ PDU according to the ACSE protocol specification. The (relative) called and calling ASO-names are placed in the appropriate parameters of the AARQ. The AARQ is mapped to the User-Information parameter. The ACPM now invokes an IA-BIND-REQUEST using the naming information to indicate that it needs a supporting association to the ASO indicated by the name returned by the Directory Facility.

The action taken by the supporting service can have one of three forms depending on the source and destination title parameters in the IA-BIND-REQUEST:

1) if the naming parameter is a Presentation-address, the ASOI creates a new Presentation Connection by invoking a P-CONNECT request service primitive.

2) if the source ASO-name names the ASO which is interpreting this IA-BIND-REQUEST service primitive, then the ASO has two choices:

   a) it may create a new ASO-association to support this request; or

   b) it may map the request for supporting service to an existing ASO-association, assuming the supporting service is designed to provide the re-use of the instance of service;

3) if the source relative ASO-name designates an ASO other than the immediate parent of the requesting ASO, the parent ASO maps this request directly to its parent ASO by generating an IA-BIND-REQUEST with the same naming parameters.

This latter case represents the situation where the relative ASO-name in the AARQ of the requesting ACSE contains more than one element, i.e. it addresses the ASO in which it occurs and one or more outer ASOs. In this case, the immediate parent is required to maintain a local mapping and generates an IA-BIND-REQUEST with identical parameters to the next outer ASO. The immediate parent is not allowed to map the requested ASO-association onto an existing ASO-association of its own, but must pass it through. However, the ASO named in the IA-BIND-REQUEST may map it to an existing ASO-association or create a new ASO-association at its discretion.

To illustrate this algebra of addressing operations, consider four ASOs each containing ACSE with A containing B which contains B, which contains C, which contains D communicating with four embedded ASOs, W, X, Y, and Z respectively:

Case 1: An A-ASSOCIATE request is invoked to the ACSE in A with the source and destination names A and W respectively. It generates an AARQ(W, A, other) and invokes an IA-BIND-REQUEST (called-P-addr, calling-P-addr, other, user-data) which is mapped to a P-CONNECT request with the appropriate P-addresses for this application-entity.

Case 2: An A-ASSOCIATE request is invoked to the ACSE in D with the source and destination names D and Z respectively. It generates an AARQ(D, Z, other) and invokes an IA-BIND-REQUEST(C, Y, other, User data), where the User data contains the AARQ. This is mapped to an A-ASSOCIATE with the identical parameters. The ACSE has the option of mapping this to an existing ASO-association or to create an new association between C and Y.

Case 3: An A-ASSOCIATE request is invoked to the ACSE in D with the source and destination names C/D and Y/Z respectively. It generates an AARQ(C/D, Y/Z, other, User data) and invokes an IA-BIND-REQUEST(B, X, other, user-data), where the User data contains the AARQ. ASO C gets the IA-BIND-REQUEST and notes that it addresses its parent. It creates a local mapping and invokes an IA-BIND-REQUEST(B, X, other, user-data). ASO B receives this primitive and now acts according to Case 2 above.

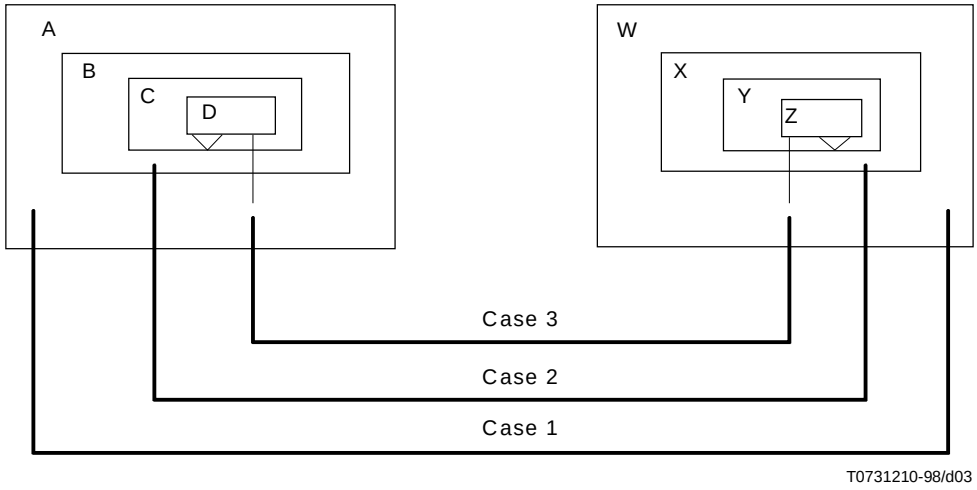

Figure E.3 – Examples of setting up higher level associations

ITU-T RECOMMENDATIONS SERIES

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