



INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.228

**OPEN SYSTEMS INTERCONNECTION
PICS PROFORMAS**

**RELIABLE TRANSFER: PROTOCOL
SPECIFICATION**

ITU-T Recommendation X.228

(Extract from the *Blue Book*)

NOTES

- 1 ITU-T Recommendation X.228 was published in Fascicle VIII.5 of the *Blue Book*. This file is an extract from the *Blue Book*. While the presentation and layout of the text might be slightly different from the *Blue Book* version, the contents of the file are identical to the *Blue Book* version and copyright conditions remain unchanged (see below).
- 2 In this Recommendation, the expression “Administration” is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

© ITU 1988, 1993

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

Recommendation X.228

RELIABLE TRANSFER: PROTOCOL SPECIFICATION¹⁾

(Melbourne, 1988)

The CCITT,

considering

(a) that Recommendation X.200 defines the Basic Reference Model of Open Systems Interconnection (OSI) for CCITT applications;

(b) that Recommendation X.210 defines the service conventions for describing the services of the OSI reference model;

(c) that Recommendation X.216 defines the Presentation Layer service;

(d) that Recommendation X.217 defines the Association Control service;

(e) that Recommendation X.218 defines the Reliable Transfer service;

(f) that there is a need for common Reliable Transfer support for various applications;

unanimously declares

that this Recommendation defines the Reliable Transfer protocol of Open Systems Interconnection for CCITT Applications as given in the Scope and Field of Application.

CONTENTS

0	<i>Introduction</i>
1	<i>Scope and Field of Application</i>
2	<i>References</i>
3	<i>Definitions</i>
4	<i>Abbreviations</i>
5	<i>Conventions</i>
6	<i>Overview of the Protocol</i>
7	<i>Elements of Procedure</i>
8	<i>Mapping to Used Services</i>
9	<i>Abstract Syntax Definition of APDUs</i>
10	<i>Conformance</i>

Annex A - State Transition Tables

Annex B - Differences between this Recommendation and Recommendation X.410-1984

Annex C - Summary of Assigned Object Identifier Values

¹⁾ Recommendation X.228 and ISO 9066-2 [Information processing systems - Text Communication - ReliableTransfer Part 2: Protocol Specification] were developed in close collaboration and are technically aligned.

0 Introduction

This Recommendation specifies the protocol for the services provided by an application-service- element - the Reliable Transfer Service Element (RTSE) - to provide for the Reliable Transfer of Application Protocol Data Units (APDUs) between open systems. This Recommendation is one of a set of Recommendations specifying the protocols for sets of application-service-elements commonly used by a number of applications.

Reliable Transfer provides an application-independent mechanism to recover from communication and end-system failure minimizing the amount of retransmission.

This Recommendation is technically aligned with ISO 9066-2.

1 Scope and Field of Application

This Recommendation specifies the protocol (abstract syntax) and procedures for the Reliable Transfer Service Element services (Recommendation X.218). The RTSE services are provided in conjunction with the Association Control Service Element (ACSE) services (Recommendation X.217) and the ACSE protocol (Recommendation X.227), and the presentation-service (Recommendation X.216).

The RTSE procedures are defined in terms of:

- a) the interactions between peer RTSE protocol machines through the use of ACSE and the presentation-service.
- b) the interactions between the RTSE protocol machine and its service-user.

This Recommendation specifies conformance requirements for systems implementing these procedures.

2 References

Recommendation X.200 - Reference Model of Open Systems Interconnection for CCITT Applications (See also ISO 7498)

Recommendation X.208 - Specification of abstract syntax notation (See also ISO 8824).

Recommendation X.209 - Specification of Basic Encoding Rules for the abstract syntax notation (See also ISO 8825).

Recommendation X.210 - Open Systems Interconnection Layer Service Definition Conventions (See also ISO/TR 8509).

Recommendation X.216 - Presentation Service Definition for Open Systems Interconnection for CCITT Applications (See also ISO 8822).

Recommendation X.217 - Association Control Service Definition for CCITT Applications (See also ISO 8649).

Recommendation X.218 - Reliable Transfer: Model and Service Definition (See also ISO 9066-1)

Recommendation X.219 - Remote Operations: Model, Notation and Service Definition (See also ISO 9072-1)

Recommendation X.227 - Association Control Protocol Specification for CCITT Applications (See also ISO 8650).

3 Definitions

3.1 Reference Model Definitions

This Recommendation is based on the concepts developed in Recommendation X.200 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) session-service;
- j) session-connection; and
- k) user-element
- l) two-way-alternate interaction
- m) transfer syntax.

3.2 *Service Conventions Definitions*

This Recommendation makes use of the following terms defined in Recommendation X.210:

- 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); and
- j) confirm (primitive).

3.3 *Presentation Service Definitions*

This Recommendation makes use of the following terms defined in Recommendation X.216:

- a) abstract syntax;
- b) abstract syntax name;
- c) presentation context;
- d) default context.

3.4 *Association Control Definitions*

This Recommendation makes use of the following terms defined in Recommendation X.217:

- a) application-association; association;
- b) application context;
- c) Association Control Service Element; and
- d) X.410-1984 mode.

3.5 *RTSE Service Definitions*

This Recommendation makes use of the following terms defined in Recommendation X.218:

- a) association-initiating-application-entity; association-initiator;
- b) association-responding-application-entity; association-responder;
- c) sending-application-entity; sender;
- d) receiving-application-entity; receiver;
- e) requestor;
- f) acceptor;

- g) Reliable Transfer Service Element;
- h) RTSE-user;
- i) RTSE-provider;
- j) ACSE-provider;
- k) monologue interaction;
- l) syntax-matching-services;
- m) Reliable Transfer;
- n) X.410-1984 mode; and
- o) normal mode.

3.6 *Reliable Transfer Protocol Specification Definitions*

For the purpose of this Recommendation the following definitions apply:

3.6.1 **reliable-transfer-protocol-machine:**

The protocol machine for the Reliable Transfer Service Element specified in this Recommendation.

3.6.2 **requesting-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the requestor of a particular Reliable Transfer Service Element service.

3.6.3 **accepting-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the acceptor for a particular Reliable Transfer Service Element service.

3.6.4 **sending-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the sender.

3.6.5 **receiving-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the receiver.

3.6.6 **association-initiating-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the association-initiator.

3.6.7 **association-responding-reliable-transfer-protocol-machine:**

The reliable-transfer-protocol-machine whose RTSE-user is the association-responder.

4 Abbreviations

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.

RTAB	RT-P-ABORT and RT-U-ABORT application-protocol-data-unit
RTORQ	RT-OPEN-REQUEST application-protocol-data-unit
RTOAC	RT-OPEN-ACCEPT application-protocol-data-unit
RTORJ	RT-OPEN-REJECT application protocol-data-unit
RTTR	RT-TRANSFER application-protocol-data-unit
RTTP	RT-TOKEN-PLEASE application-protocol-data-unit

4.3 Other Abbreviations

The following abbreviations are used in this Recommendation.

AE	application-entity
ACSE	Association Control Service Element
ASE	application-service-element
RTPM	reliable-transfer-protocol-machine
RT (or RTS)	Reliable Transfer
RTSE	Reliable Transfer Service Element

5 Conventions

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

M	presence is mandatory
U	presence is a RTSE service-user option
T	presence is a RTPM option
req	source is related request primitive
ind	sink is related indication primitive
resp	source is related response primitive
conf	sink is related confirm primitive
sp	source or sink is the RTPM

The structure of each RTSE APDU is specified in clause 9 using the abstract syntax notation of Recommendation X.208.

6 Overview of the Protocol

6.1 Service Provision

The protocol specified in this Recommendation provides the services defined in Recommendation X.218. These services are listed in Table 1/X.228.

TABLE 1/X.228

RTSE Service Summary

Service	Type
RT-OPEN	Confirmed
RT-CLOSE	Confirmed
RT-TRANSFER	Confirmed
RT-TURN-PLEASE	Non-confirmed
RT-TURN-GIVE	Non-confirmed
RT-P-ABORT	Provider-initiated
RT-U-ABORT	Non-confirmed

6.2 *Use of Services*

6.2.1 *ACSE Services*

The RTPM requires access to the A-ASSOCIATE, A-RELEASE, A-ABORT and A-P-ABORT services. This Recommendation assumes that the RTPM is the sole user of these services.

6.2.2 *Use of the Presentation-service*

The RTPM requires access to the P-ACTIVITY-START, P-DATA, P-MINOR-SYNCHRONIZE, P-ACTIVITY-END, P-ACTIVITY-INTERRUPT, P-ACTIVITY-DISCARD, P-U-EXCEPTION-REPORT, P-ACTIVITY-RESUME, P-P-EXCEPTION-REPORT, P-TOKEN-PLEASE and P-CONTROL-GIVE services. This Recommendation assumes that the RTPM is the sole user of the above services.

The RTPM requires access to local syntax-matching-services provided by the presentation-service provider. This syntax-matching-service consists of:

- an encoding service enabling the transformation from the local representation of an APDU value into an encoded-APDU-value of type OCTET STRING the value of which is the representation of the APDU value specified by the negotiated transfer syntax;
- a decoding service enabling the transformation from an encoded-APDU-value into the local representation of the APDU value.

If X.410-1984 mode or simple encoding is used by the Presentation Layer, the APDU value is encoded as ASN.1 type ANY. If full encoding is used by the Presentation Layer, the APDU value is encoded as ASN.1 type EXTERNAL. (For X.410-1984 mode, single encoding and full encoding see Recommendation X.226).

This Recommendation recognizes that the ACSE services require access to the P-CONNECT, P-RELEASE, P-U-ABORT and P-P-ABORT services. This Recommendation assumes that the ACSE and the RTPM are the sole users of any of the above, or of any other, presentation-services.

During the lifetime of an application-association, the underlying presentation-connections either use a single presentation context or multiple presentation contexts as a part of the presentation multiple defined context facility. The choice is determined by the use of the Single Presentation Context parameter of the RT-OPEN service as described in clause 8.1.1.1.3 and 8.1.1.1.4.

6.3 *Model*

The reliable-transfer-protocol-machine (RTPM) communicates with its service-user by means of primitives defined in Recommendation X.218. Each invocation of the RTPM controls a single application-association.

The RTPM is driven by RTSE service request and response primitives from its service-user, and by indication and confirm primitives from the ACSE services and the presentation-service. The RTPM, in turn, issues indication and confirm primitives to its service-user and request and response primitives on the used ACSE services or presentation-service.

The reception of an RTSE service primitive, or of an ACSE service primitive, or of a presentation-service primitive, and the generation of dependent actions are considered to be indivisible.

During the use of the RTSE services, the existence of both the association-initiating AE and the association-responding AE is presumed. How these AEs are created is beyond the scope of this Recommendation.

During the use of the RTSE services, except RT-OPEN, the existence of an application-association between the peer AEs is presumed.

Note - Each application-association may be identified in an end system by an internal, implementation dependent mechanism so that the RTSE service-user, and the RTPM, and the ACSE service-provider can refer to it.

7 Elements of procedure

The RTSE protocol consists of the following elements of procedure:

- a) association-establishment
- b) association-release
- c) transfer
- d) turn-please
- e) turn-give
- f) error reporting:
 - f1) user-exception-report
 - f2) provider-exception-report
- g) error handling:
 - g1) transfer-interrupt
 - g2) transfer-discard
 - g3) association-abort
 - g4) association-provider-abort
- h) error recovery:
 - h1) transfer-resumption (for recovery from g1), or after successful h3) from g3) or g4))
 - h2) transfer-retry (for recovery from g2))
 - h3) association-recovery (for recovery from g3 or g4))
- i) abort:
 - i1) transfer-abort (recovery from g1) or g2) or g3) or g4) not possible)
 - i2) provider-abort (recovery from g1) or g2) or g3) or g4) not possible)
 - i3) user-abort.

In the following clauses, 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 RTSE service primitives and the APDUs involved and the used presentation-service.

Clause 8 describes how the service primitives are mapped on the ACSE services, and on the presentation-service.

7.1 Association-establishment

7.1.1 Purpose

The association-establishment procedure is used to establish an application-association.

7.1.2 APDUs used

The association-establishment procedures uses the RT-OPEN-REQUEST (RTORQ) APDU, the RT-OPEN-ACCEPT (RTOAC) APDU, and the RT-OPEN-REJECT (RTORJ) APDU.

Note - These APDUs are also used in the association-recovery procedure.

7.1.2.1 RTORQ APDU

The RT-OPEN REQUEST (RTORQ) APDU is used in the request to establish an application-association. The fields of the RTORQ APDU are listed in Table 2/X.228.

7.1.2.2 RTOAC APDU

The RT-OPEN-ACCEPT (RTOAC) APDU is used in the positive response to the request to establish an application-association. The fields of the RTOAC APDU are listed in Table 3/X.228.

TABLE 2/X.228

RTORQ APDU Fields

Field name	Presence	Source	Sink
Checkpoint-size	T	sp	sp
Window-size	T	sp	sp
Dialogue-mode	U	req	ind
User-data (Note 1)	U	req	ind
Session-connection-identifier (Note 2)	T	sp	sp
Application-protocol (Note 3)	U	req	ind

Note 1 - The User-data field is used solely in the association-establishment procedure.

Note 2 - The Session-connection-identifier field is used solely in the association-recovery procedure.

Note 3 - The Application-protocol field is used solely in the X.410-1984 mode.

TABLE 3/X.228

RTOAC APDU Fields

Field name	Presence	Source	Sink
Checkpoint-size	T	sp	sp
Window-size	T	sp	sp
User-data (Note 1)	U	resp	conf
Session-connection-identifier (Note 2)	T	sp	sp

Note 1 - The User-data field is used solely in the association-establishment procedure.

Note 2 - The Session-connection-identifier field is used solely in the association-recovery procedure.

7.1.2.3 RTORJ APDU

RT-OPEN-REJECT (RTORJ) APDU is used in the negative response to the request to establish an application-association. The fields of the RTORJ APDU are listed in Table 4/X.228.

TABLE 4/X.228

RTORJ APDU Fields

Field name	Presence	Source	Sink
Refuse-reason (Note 1)	T	sp	sp
User-data (Note 2)	U	resp	conf

Note 1 - The Refuse-reason field is used solely in the X.410-1984 mode.

Note 2 - The User-data field is used solely in the normal mode, and is not used in the association-recovery procedure.

7.1.3 Association-establishment procedure

This procedure is driven by the following events:

- an RT-OPEN request primitive from the requestor (association-initiator);
- an RTORQ APDU as user-data on an A-ASSOCIATE indication primitive;
- an RT-OPEN response primitive from the acceptor (association-responder);
- an A-ASSOCIATE confirm primitive that may contain either an RTOAC APDU, or an RTORJ APDU, or no APDU.

7.1.3.1 RT-OPEN request primitive

The requesting RTPM forms an RTORQ APDU from the parameter values of the RT-OPEN request primitive and its internal data. The RT-OPEN request primitive parameters, except user-data, are stored by the requesting RTPM for association-recovery. The requesting RTPM issues an A-ASSOCIATE request primitive also using information from the RT-OPEN request primitive. The RTORQ APDU is the user information parameter value of the A-ASSOCIATE request primitive.

The requesting RTPM waits for a primitive from the ACSE-provider and does not accept any other primitive from the requestor.

7.1.3.2 RTORQ APDU

If the application-association is not accepted by the ACSE-provider, no A-ASSOCIATE indication primitive is received by the accepting RTPM and no action takes place.

If the application-association is accepted by the ACSE-provider, the accepting RTPM receives the RTORQ APDU as the user information parameter on an A-ASSOCIATE indication primitive.

If any of the A-ASSOCIATE indication parameters or any of the RTORQ APDU fields is unacceptable to the accepting RTPM, or if the accepting RTPM is not able to accept the application-association, it forms and sends an RTORJ APDU with appropriate parameters from internal data. The accepting RTPM issues an A-ASSOCIATE response primitive. The RTORJ APDU is sent as the user information parameter of the A-ASSOCIATE response primitive. The application-association is not established. The accepting RTPM does not issue an RT-OPEN indication.

If the A-ASSOCIATE indication primitive and the RTORQ APDU parameters are acceptable to the accepting RTPM, it issues an RT-OPEN indication primitive to the acceptor. The RT-OPEN indication parameter values are derived from the RTORQ APDU and the A-ASSOCIATE indication primitive parameter values.

The accepting RTPM waits for an RT-OPEN response primitive from the acceptor, or a primitive from the ACSE-provider.

7.1.3.3 *RT-OPEN response primitive*

When the accepting RTPM receives an RT-OPEN response primitive from the acceptor, the result parameter specifies whether the acceptor has accepted (value "accepted") or rejected the application-association.

If the application-association is accepted by the acceptor, the accepting RTPM forms an RTOAC APDU using the RT-OPEN response primitive parameters and internal data. The RT-OPEN response primitive parameters, except user data, are stored by the accepting RTPM for association-recovery.

The accepting RTPM issues an A-ASSOCIATE response primitive also using information from the RT-OPEN response primitive. The RTOAC APDU is sent as the user information parameter of the A-ASSOCIATE response primitive.

If the application-association is rejected by the acceptor, the accepting RTPM forms a RTORJ APDU using the RT-OPEN response primitive parameters and internal data. The accepting RTPM issues an A-ASSOCIATE response primitive also using information from the RT-OPEN request primitive. The RTORJ APDU is sent as the user information parameter of the A-ASSOCIATE response primitive. The application-association is not established.

7.1.3.4 *A-ASSOCIATE confirm primitive*

The requesting RTPM receives an A-ASSOCIATE confirm primitive. The following situations are possible:

- a) the application-association has been accepted by the acceptor;
- b) the accepting RTPM or the acceptor has rejected the application-association; or
- c) the ACSE service provider has rejected the application-association.

If the application-association was accepted by the acceptor, the A-ASSOCIATE confirm primitive result parameter has the value "accepted" and the RTOAC APDU is the value of the user information parameter of the A-ASSOCIATE confirm primitive. The requesting RTPM issues an RT-OPEN confirm primitive to the requestor. The result parameter has the value "accepted" and the user-data parameter contains the user-data parameter value of the RTOAC APDU. The other parameters of the RT-OPEN confirm primitive are derived from the A-ASSOCIATE confirm primitive.

If the application-association was rejected by either the acceptor or the accepting RTPM, the A-ASSOCIATE confirm primitive result parameter has one of the values "rejected..", the A-ASSOCIATE confirm primitive result source parameter has the value "ACSE service-user" and the RTORJ APDU is the value of the user information parameter of the A-ASSOCIATE confirm primitive. The requesting RTPM issues an RT-OPEN confirm primitive to the requestor. The result parameter has one of the values "rejected . . ." and the other parameter values are derived from the A-ASSOCIATE confirm primitive parameters and the RTORJ APDU. The application-association is not established.

If the application-association was rejected by the ACSE service-provider, the A-ASSOCIATE confirm primitive result parameter has one of the values "rejected ...", the A-ASSOCIATE confirm primitive result source parameter has either the value "ACSE service-provider" or "presentation service-provider". The user-data parameter of the RT-OPEN confirm primitive is absent and the application-association is not established. The other parameters of the RT-OPEN confirm primitive are derived from the A-ASSOCIATE confirm primitive.

7.1.4 Use of the RTORQ APDU fields

The RTORQ APDU fields are used as follows.

7.1.4.1 Checkpoint-size

The checkpoint-size field allows negotiation of the maximum amount of data (in units of 1024 octets) that may be sent between two minor synchronization points. A value of zero from the requesting RTPM invites the accepting RTPM to select checkpoint-size. If this field is absent, checkpoint-size zero is assumed.

7.1.4.2 Window-size

The window-size field allows negotiation of the maximum number of outstanding minor synchronization points before data transfer shall be suspended. If this field is absent, window-size 3 is assumed.

7.1.4.3 Dialogue-mode

This is the dialogue-mode parameter value from the RT-OPEN request primitive. It appears as the dialogue-mode parameter value of the RT-OPEN indication primitive.

The value of this field is either monologue, or two-way-alternate. If this field is absent, monologue is assumed.

7.1.4.4 User-data

This is the user-data parameter value from the RT-OPEN request primitive. It appears as the user-data parameter value of the RT-OPEN indication primitive.

The value of this field is transparent to the RTPM.

7.1.4.5 Session-connection-identifier

This field is used only in the association-recovery procedure.

7.1.4.6 Application-protocol

This field is used solely in the X.410-1984 mode. It is the application-protocol parameter value from the RT-OPEN request primitive. It appears as the application-protocol parameter value in the RT-OPEN indication primitive.

7.1.5 Use of the RTOAC APDU fields

The RTOAC APDU fields are used as follows.

7.1.5.1 Checkpoint-size

The checkpoint-size field allows negotiation of the maximum amount of data (in units of 1024 octets) that may be sent between two minor synchronization points. If the checkpoint-size in the RTORQ APDU is greater than zero, the accepting RTPM shall supply a value in the RTOAC APDU that is less than or equal to the value in the RTORQ APDU, else the accepting RTPM may select checkpoint-size. A value of zero from the accepting RTPM indicates that checkpointing will not be used. The value of this field becomes the agreed maximum value and governs both directions of transfer. If this field is absent, it is assumed that checkpointing will not be used.

7.1.5.2 Window-size

This field is only used if checkpoint-size of the RTOAC APDU is greater than zero. The window-size field allows negotiation of the maximum number of outstanding minor synchronization points before data transfer shall be suspended. The accepting RTPM shall supply a value that is less than or equal to the value in the RTORQ APDU. This becomes the agreed maximum size and governs both directions of transfer. If this field is absent, window-size 3 is assumed.

7.1.5.3 *User-data*

This is the user-data parameter value from the RT-OPEN response primitive. It appears as the user-data parameter value of the RT-OPEN confirm service primitive.

The value of this field is transparent to the RTPM.

7.1.5.4 *Session-connection-identifier*

This field is used only in the association-recovery procedure.

7.1.6 *Use of the RTORJ APDU fields*

The RTORJ APDU fields are used as follows.

7.1.6.1 *Refuse-reason*

The refuse-reason field is only used in the X.410-1984 mode.

This field may contain one of the following values:

- rts-busy The accepting RTPM, or the acceptor, is so loaded that it cannot support a new application-association. The requesting RTPM should retry after a period of time. This value is either provided by the accepting RTPM, or is derived from the result parameter value "rejected (transient)" of the RT-OPEN response primitive from the acceptor. It appears as the result parameter value "rejected (transient)" of the RT-OPEN confirm primitive to the requestor.
- cannot recover This value is used only by the accepting RTPM in the association-recovery procedure if it is unable to accept an association-recovery.
- validation failure The acceptor does not recognize the requestor's credentials as being valid for the proposed application-association. This value is the user-data parameter value of the RT-OPEN response primitive from the acceptor. It appears as the user-data parameter value of the RT-OPEN confirm primitive to the requestor.
- unacceptable-dialogue-mode The acceptor does not accept the type of dialogue mode proposed for the application-association. This value is the user-data parameter value of the RT-OPEN response primitive from the acceptor. It appears as the user-data parameter value of the RT-OPEN confirm primitive to the requestor.

7.1.6.2 *User-data*

This field is only used in the normal mode:

This is the user-data parameter value of the RT-OPEN response primitive from the acceptor. It appears as the user-data parameter value of the RT-OPEN confirm primitive to the requestor.

The value of this field is transparent to the RTPM.

7.2 Association-release

7.2.1 Purpose

The association-release procedure is used for the normal release of an application-association by the association-initiator without loss of information in transit.

7.2.2 APDUs used

No APDUs are used in this procedure.

7.2.3 Association-release procedure

This procedure is driven by the following events:

- a) an RT-CLOSE request primitive from the requestor (association-initiator);
- b) an A-RELEASE indication primitive;
- c) an RT-CLOSE response primitive from the acceptor (association-responder);
- d) an A-RELEASE confirm primitive.

7.2.3.1 RT-CLOSE request primitive

The requestor may issue an RT-CLOSE request primitive only if it possesses the turn and if there is no outstanding RT-TRANSFER confirm primitive. When an RT-CLOSE request primitive is received from the requestor, the requesting (association-initiating) RTPM issues an A-RELEASE request primitive. The reason parameter of the A-RELEASE request primitive is the reason parameter of the RT-CLOSE request primitive. The user information parameter of the A-RELEASE request primitive is the user-data parameter of the RT-CLOSE request primitive.

Note - No RT-CLOSE request primitive parameters are present in X.410-1984 mode.

The requesting RTPM waits for a primitive from the ACSE service-provider and does not accept any other primitive from the requestor.

7.2.3.2 A-RELEASE indication primitive

The accepting RTPM receives the A-RELEASE indication primitive.

It issues an RT-CLOSE indication primitive to the acceptor. The RT-CLOSE indication parameter values are derived from the A-RELEASE indication primitive.

Note - No RT-CLOSE indication primitive parameters are present in X.410-1984 mode.

The RTPM waits for a primitive from the acceptor or the used service provider.

7.2.3.3 RT-CLOSE response primitive

When the accepting RTPM receives an RT-CLOSE response primitive, the accepting RTPM issues an A-RELEASE response primitive. The reason parameter of the A-RELEASE response primitive is the reason parameter of the RT-CLOSE response primitive. The user information parameter of the A-RELEASE response primitive is the user-data parameter of the RT-CLOSE response primitive. The result parameter value of the A-RELEASE response primitive is "affirmative".

Note - No RT-CLOSE response primitive parameters are present in X.410-1984 mode.

7.2.3.4 A-RELEASE confirm primitive

The requesting RTPM receives an A-RELEASE confirm primitive.

The requesting RTPM issues an RT-OPEN confirm primitive to the acceptor. The RT-OPEN confirm primitive parameter values are derived from the A-RELEASE confirm primitive.

Note - No RT-CLOSE confirm primitive parameters are present in X.410-1984 mode.

7.3 Transfer

7.3.1 Purpose

The transfer procedure is used to transfer an RTSE-user APDU from the requestor (sender) to the acceptor (receiver).

7.3.2 APDUs used

Each RTSE-user APDU, conveyed in an RT-TRANSFER request, constitutes an activity. For each application-association, at most one activity, or one interrupted activity awaiting resumption, may exist at a time.

The RTSE-user APDU value is transformed into the encoded-APDU-value and vice versa by means of the local syntax-matching services. The transfer procedure uses the RT-TRANSFER (RTTR) APDU. The transfer procedure supports segmenting and reassembling of the encoded-APDU-value into/from one or more RT-FR APDUs.

An encoded-APDU-value is transferred as a single RTTR APDU if checkpointing is not used. Otherwise, the encoded-APDU-value is transferred as a series of RTTR APDUs, the maximum size (i.e. number of octets forming the RTTR APDU value) of each being the negotiated checkpoint-size. The concatenation of the RTTR APDU values is the encoded-APDU-value.

The fields of the RTTR APDU are listed in Table 5/X.228.

TABLE 5/X.228

RTTR APDU Fields

Field name	Presence	Source	Sink
User-data-part	M	req	ind/conf

7.3.3 Transfer procedure

This procedure is driven by the following events:

- an RT-TRANSFER request primitive from the requestor (sender);
- a P-ACTIVITY-START indication primitive, followed by one or more RTTR APDUs as user-data of P-DATA indication primitives each, except the last, followed by a P-MINOR-SYNCHRONIZE indication primitive;
- a P-MINOR-SYNCHRONIZE confirm primitive;
- a P-ACTIVITY-END indication primitive;
- a P-ACTIVITY-END confirm primitive;
- a transfer time-out.

7.3.3.1 *RT-TRANSFER request primitive*

If the requesting RTPM possesses the turn and receives a RT-TRANSFER request from the requestor, the requesting RTPM transforms the RTSE-user APDU value into the encoded-APDU-value by means of the encoding service of the local syntax-matching services.

The requesting RTPM issues a P-ACTIVITY-START request primitive and may start transmitting the first RTTR APDU in a P-DATA request primitive immediately after the P-ACTIVITY-START request primitive is issued, since the latter service is not a confirmed service.

The maximum RTTR APDU size will have been negotiated during the association-establishment procedure. The requesting RTPM shall submit, in P-DATA request primitives, RTTR APDUs that conform to that agreement. Checkpoints may only be inserted if a checkpoint-size greater than zero was negotiated during the association-establishment procedure.

If a transferred RTTR APDU is not the last in a series of RTTR APDUs used to transfer a single encoded-APDU-value, the requesting RTPM inserts a checkpoint by issuing a P-MINOR-SYNCHRONIZE request primitive. The requesting RTPM uses only the "explicit confirmation expected" type of minor synchronization. The requesting RTPM may issue further P-DATA request primitives and P-MINOR-SYNCHRONIZE request primitives unless the agreed window-size has been reached.

If the RTTR APDU is the only one, or the last in a series of RTTR APDUs used to transfer a single encoded-APDU-value, the requesting RTPM issues a P-ACTIVITY-END request primitive.

Consecutive P-DATA request primitives shall not be issued, and all data transfer shall take place within an activity.

7.3.3.2 *P-ACTIVITY-START indication primitive, RTTR APDUs, and P-MINOR-SYNCHRONIZE indication primitives*

The accepting RTPM receives a P-ACTIVITY-START indication primitive, indicating the start of transfer of a RTSE-user APDU. The accepting RTPM receives an RTTR APDU as user-data of a P-DATA indication primitive.

If the RTTR APDU is not the last in a series of RTTR APDUs used to transfer a single encoded-APDU-value, the accepting RTPM receives a P-MINOR-SYNCHRONIZE indication primitive. If the accepting RTPM has secured the RTTR APDU, it issues a P-MINOR-SYNCHRONIZE response primitive.

7.3.3.3 *P-MINOR-SYNCHRONIZE confirmed primitive*

When the requesting RTPM receives a P-MINOR-SYNCHRONIZE confirm primitive, it assumes that the accepting RTPM has secured the encoded-APDU-value APDU up to that point.

The requesting RTPM may issue further P-DATA request primitives and P-MINOR-SYNCHRONIZE request primitives unless the agreed window-size has been reached. The window is advanced when a P-MINOR-SYNCHRONIZE confirm primitive is received by the requesting RTPM.

When a complete encoded-APDU-value has been transmitted, the requesting RTPM issues a P-ACTIVITY-END request primitive.

7.3.3.4 *P-ACTIVITY-END indication primitive*

A P-ACTIVITY-END indication primitive indicates to the accepting RTPM that a complete encoded-APDU-value has been transferred. The accepting RTPM transforms the encoded-APDU-value into the RTSE-user APDU value by means of the decoding service of the local syntax-matching-services.

If the accepting RTPM has secured the complete RTSE-user APDU, it issues an RT-TRANSFER indication primitive to the acceptor, and issues a P-ACTIVITY-END response primitive.

The accepting RTPM records the session-connection-identifier and the activity identifier of the last RTSE-user APDU which it completely secured for association-recovery purposes.

7.3.3.5 *P-ACTIVITY-END confirm primitive*

An activity end is an implicit major synchronization point and once successfully confirmed by means of an P-ACTIVITY-END confirm primitive, it indicates to the requesting RTPM that the RTSE-user APDU has been secured by the accepting RTPM. The requesting RTPM may then delete the transferred RTSE-user APDU.

When the requesting RTPM receives the P-ACTIVITY-END confirm primitive, it issues an RT-TRANSFER confirm primitive with a result parameter value of "APDU-transferred" to the requestor.

7.3.3.6 *Transfer time-out*

If an APDU has not been transferred within the time specified in the transfer-time parameter of the RT-TRANSFER request primitive (that is, the requesting RTPM has not received the P-ACTIVITY-END confirm primitive), the requesting RTPM performs the transfer-discard procedure followed by the transfer-abort procedure.

If during the transfer-discard procedure, the requesting RTPM does not receive a P-ACTIVITY-DISCARD confirm primitive within a (locally specified) reasonable time, the requesting RTPM performs the transfer-abort procedure followed by the provider-abort procedure.

7.4 *Turn-please*

7.4.1 *Purpose*

The turn-please procedure is used by a receiver (requestor) to request the turn from the sender (acceptor).

7.4.2 *APDUs used*

The turn-please procedure uses the RT-TURN-PLEASE (RTTP) APDU.

The fields of the RTTP APDU are listed in Table 6/X.228.

TABLE 6/X.228

RTTP APDU Fields

Field name	Presence	Source	Sink
Priority	U	req	ind

7.4.3 *Turn-please procedure*

This procedure is driven by the following events:

- an RT-TURN-PLEASE request primitive from the requestor;
- an RTTP APDU as user-data of a P-TOKEN-PLEASE indication primitive.

7.4.3.1 *RT-TURN-PLEASE request primitive*

If the requesting RTPM does not possess the turn and receives an RT-TURN-PLEASE request from the requestor, the requesting RTPM issues a P-TOKEN-PLEASE request primitive. If the priority parameter is present in the RT-TURN-PLEASE request primitive, and RTTP APDU is formed from the parameter value and transferred as user-data of the P-TOKEN-PLEASE request primitive. This procedure may be performed either inside or outside an activity.

7.4.3.2 *RTTP APDU*

If the accepting RTPM receives a P-TOKEN-PLEASE indication primitive, the accepting RTPM issues an RT-TURN-PLEASE indication primitive to the acceptor. If an RTTP APDU is transferred as user-data of the P-TOKEN-PLEASE indication primitive, the RT-TURN-PLEASE indication primitive parameter is present and derived from the RTTP APDU.

7.4.4 *Use of the RTTP fields*

The RTTP APDU fields are used as follows.

7.4.4.1 *Priority*

This is the priority parameter value of the RT-TURN-PLEASE request primitive. It appears as the priority parameter value of the RT-TURN-PLEASE indication primitive.

The value of this field is transparent to the RTPM.

7.5 *Turn-give*

7.5.1 *Purpose*

The turn-give procedure is used by a sender (requestor) to give the turn to the receiver (acceptor). The requestor becomes the receiver and the acceptor becomes the sender.

7.5.2 *APDUs used*

No APDUs are used in this procedure.

7.5.3 *Turn-give procedure*

The turn-give procedure is driven by the following events:

- a) an RT-TURN-GIVE request primitive;
- b) a P-CONTROL-GIVE indication primitive.

7.5.3.1 *RT-TURN-GIVE request primitive*

If the requesting RTPM possesses the turn and receives an RT-TURN-GIVE request primitive from the requestor, it issues a P-CONTROL-GIVE request primitive and becomes the receiving RTPM. This may be done only outside an activity.

7.5.3.2 *P-CONTROL-GIVE indication primitive*

If the accepting RTPM receives a P-CONTROL-GIVE indication primitive, the accepting RTPM issues an RT-TURN-GIVE indication primitive to the acceptor, and issues a P-CONTROL-GIVE response primitive. The accepting RTPM becomes the sending RTPM.

7.6 *Error reporting*

7.6.1 *User-exception-report*

7.6.1.1 *Purpose*

The user-exception-report procedure is used by the receiving RTPM to report an error situation to the sending RTPM.

7.6.1.2 *APDUs used*

No APDUs are used in this procedure.

7.6.1.3 *User-exception-report procedure*

This procedure is driven by the following events:

- a) a receiving RTPM problem;
- b) a P-U-EXCEPTION-REPORT indication primitive.

7.6.1.3.1 *Receiving RTPM problem*

If the receiving RTPM detects a problem, it issues a P-U-EXCEPTION-REPORT request primitive and starts a local recovery timer. Depending on the severity of the detected error, the value of the reason parameter of the P-U-EXCEPTION-REPORT request primitive is as follows:

- a) In severe problem situations, the value "receiving ability jeopardized" is used.
- b) In exceptional circumstances, the receiving RTPM may have to delete a partially received RTSE-user APDU, even though some minor synchronization points have been confirmed. In this case, the value "unrecoverable procedure error" is used.
- c) If the receiving RTPM is not willing to complete a transfer procedure, the value "non-specific error" is used.
- d) If the sending RTPM resumes a transfer procedure already finished by the receiving RTPM (see clause 7.8.1.3.2), the value "sequence error" is used.
- e) For all other less severe error situations, the value "local SS-user error" is used.

7.6.1.3.2 *P-U-EXCEPTION-REPORT indication primitive*

If the sending RTPM receives a P-U-EXCEPTION-REPORT indication primitive, it performs one of the following procedures depending on the reason parameter value of the P-U-EXCEPTION-REPORT indication primitive:

- a) With a value "receiving ability jeopardized", the transfer-abort procedure followed by the provider abort procedure are performed.
- b) With a value "unrecoverable procedure error", the transfer-discard procedure followed by transfer-retry procedure are performed.
- c) With a value "non-specific error", the transfer-discard procedure followed by the transfer-abort procedure are performed.
- d) With a value "sequence error", the transfer-discard procedure is performed and the requesting RTPM issues an RT-TRANSFER confirm primitive with a result parameter value of "APDU-transferred" to the requestor and the transfer procedure is finished.

- e) With a value "local SS-user error" and at least one confirmed checkpoint in the transfer procedure, the transfer-interrupt procedure followed by the transfer-resumption procedure are performed. If no checkpoint was confirmed in the transfer procedure, the transfer-discard procedure followed by the transfer-retry procedure are performed.

7.6.2 *Provider-exception-report*

7.6.2.1 *Purpose*

If the presentation service-provider detects an unexpected situation during an activity, not covered by other services, a P-P-EXCEPTION-REPORT indication primitive is issued to both RTPMs.

7.6.2.2 *APDUs used*

No APDUs are used in this procedure.

7.6.2.3 *Provider-exception-report procedure*

This procedure is driven by the following events:

- a) a P-P-EXCEPTION-REPORT indication primitive.

7.6.2.3.1 *P-P-EXCEPTION-REPORT indication primitive*

The receiving RTPM receives a P-P-EXCEPTION-REPORT indication primitive.

If the sending RTPM receives a P-P-EXCEPTION-REPORT indication primitive, it may perform one of the following procedures:

- a) if at least one checkpoint was confirmed in the transfer procedure, the transfer-interrupt procedure followed by the transfer-resumption procedure; or,
- b) if no checkpoint was confirmed in the transfer procedure, the transfer-discard procedure followed by the transfer-retry procedure; or,
- c) the transfer-abort procedure followed by the provider-abort procedure.

7.7 *Error handling*

7.7.1 *Transfer-interrupt*

7.7.1.1 *Purpose*

The transfer-interrupt procedure is used by the sending RTPM to handle a less severe (than those handled by the other error handling procedures) error situation during the transfer procedure, if at least one checkpoint was confirmed during the transfer procedure.

7.7.1.2 *APDUs used*

No APDUs are used in this procedure.

7.7.1.3 *Transfer-interrupt procedure*

This procedure is driven by the following events:

- a) a sending RTPM problem;
- b) a P-ACTIVITY-INTERRUPT indication primitive;
- c) a P-ACTIVITY-INTERRUPT confirm primitive.

7.7.1.3.1 *Sending RTPM problem*

If the sending RTPM detects a less severe problem and at least one checkpoint was confirmed during the transfer procedure, it issues a P-ACTIVITY-INTERRUPT request primitive with one of the following reason parameter values:

- a) "non-specific error", if the problem was indicated by an error reporting procedure;
- b) "local SS-user error", if the problem is a local sending RTPM problem.

7.7.1.3.2 *P-ACTIVITY-INTERRUPT indication primitive*

If the receiving RTPM receives a P-ACTIVITY-INTERRUPT indication primitive, it issues a P-ACTIVITY-INTERRUPT response primitive and starts a local recovery timer.

7.7.1.3.3 *P-ACTIVITY-INTERRUPT confirm primitive*

If the sending RTPM receives a P-ACTIVITY-INTERRUPT confirm primitive, it starts the transfer-resumption procedure.

7.7.2 *Transfer-discard*

7.7.2.1 *Purpose*

The transfer-discard procedure is used by the sending RTPM to escape from a more severe (than those handled by the transfer-interrupt procedure) error situation, or a less severe error situation if no checkpoint was confirmed, during the transfer procedure.

7.7.2.2 *APDUs used*

No APDUs are used in this procedure.

7.7.2.3 *Transfer-discard procedure*

This procedure is driven by the following events:

- a) a sending RTPM problem;
- b) a P-ACTIVITY-DISCARD indication primitive;
- c) a P-ACTIVITY-DISCARD confirm primitive.

7.7.2.3.1 *Sending RTPM problem*

If the sending RTPM detects a more severe problem, or a less severe problem if no checkpoint was confirmed during the transfer procedure, it issues a P-ACTIVITY-DISCARD request primitive with one of the following Reason parameter values:

- a) "non-specific error", if the problem was indicated by an error reporting procedure;
- b) "local SS-user error", or "unrecoverable procedural error", if the problem is a local sending RTPM problem.

7.7.2.3.2 P-ACTIVITY-DISCARD indication primitive

If the receiving RTPM receives a P-ACTIVITY-DISCARD indication primitive, it issues a P-ACTIVITY-DISCARD response primitive. The receiving RTPM deletes all knowledge and contents of the associated RTSE-user APDU so far received.

If the receiving RTPM has already issued an RT-TRANSFER indication primitive, it performs the association-abort procedure. The abort-reason field value of the RTAB APDU is "transfer-completed". In this case the sending RTPM ends the transfer procedure with a positive RT-TRANSFER confirm primitive and the association-recovery procedure is performed.

7.7.2.3.3 P-ACTIVITY-DISCARD confirm primitive

The receipt of a P-ACTIVITY-DISCARD confirm primitive by the sending RTPM signifies the completion of the transfer-discard procedure.

7.7.3 Association-abort

7.7.3.1 Purpose

The association-abort procedure is used by the RTPMs to handle the most severe error situations. This procedure can be performed between an RT-TRANSFER request primitive and its corresponding RT-TRANSFER confirm primitive.

7.7.3.2 APDUs used

The association-abort procedure uses the RT-ABORT (RTAB) APDU. The fields of the RTAB APDU are listed in Table 7/X.228.

Note - The RTAB APDU is also used by the provider-abort and the user-abort procedure.

TABLE 7/X.228

RTAB APDU Fields

Field name	Presence	Source	Sink
Abort-reason	T	sp	sp
Reflected-parameter	T	sp	sp
User-data	U	req	ind

7.7.3.3 Association-abort procedure

This procedure is driven by the following events:

- an RTPM-abort;
- an RTAB APDU.