Consortium Specification

Remote Database Access for TCP/IP

SQL Access Group

Consortium Specification

Remote Database Access for TCP/IP

ISBN: 1-872630-84-7
Document Number: J301

Published by X/Open Company Ltd., U.K.

Any comments relating to the material contained in this document may be submitted to:

SQL Access Group
Post Office Box 5559
Manchester, New Hampshire 03108
U.S.A.

Telephone: +1 603 434 0802
Fax: +1 603 437 1548

Contents

Chapter	**1**	**Overview**	**1**
Chapter	**2**	**Services**	**3**
	2.1	A-ASSOCIATE	4
	2.2	A-RELEASE	11
	2.3	A-ABORT	13
	2.4	A-P-ABORT	14
	2.5	P-DATA	15
Chapter	**3**	**Mapping onto TCP**	**17**
	3.1	Use of TCP Service	17
	3.1.1	TCP Service Options	17
	3.2	Mapper APDUs	18
	3.2.1	RDATCP APDU Syntax	18
	3.2.2	APDU Encoding	18
	3.3	Mapping Procedures	19
	3.3.1	A-ASSOCIATE Request Procedure	19
	3.3.2	A-ASSOCIATE Indication Procedure	20
	3.3.3	A-ASSOCIATE Response Procedure	22
	3.3.4	A-ASSOCIATE Confirm Procedure	22
	3.3.5	A-RELEASE Request Procedure	23
	3.3.6	A-RELEASE Confirm Procedure	24
	3.3.7	A-ABORT Request Procedure	24
	3.3.8	A-P-ABORT Indication Procedure	24
	3.3.9	P-DATA Request Procedure	25
	3.3.10	P-DATA Indication Procedure	25
	3.3.11	APDU Transmission	26
	3.3.12	APDU Reception	26
	3.3.13	TCP Connection Listening	27
Appendix	**A**	**OSI Primitives Mapping**	**29**
	A.1	X/Open XTI	29
	A.2	Berkeley System Distribution (BSD) Sockets	30
		Index	**31**

List of Tables

2-1	RDA-TCP Mapper Services	3
2-2	Application Context Names	4
2-3	Presentation Context Definition List Value for X/Open RDA Application-context	7
2-4	Presentation Context Definition List Value for ISO RDA Application-context	7
2-5	Presentation Context Definition Result List Value	8
2-6	A-ASSOCIATE Parameters	9
2-7	A-RELEASE Parameters	12
2-8	A-ABORT Parameters	13
2-9	A-P-ABORT Parameters	14
2-10	P-DATA Parameters	15
3-1	TCP Active OPEN Arguments	19
3-2	AREQ APDU Contents	20
3-3	A-ASSOCIATE Indication Parameters	21
3-4	ARSP APDU Contents	22
3-5	A-ASSOCIATE Confirm Parameters	23
3-6	A-RELEASE Confirm Parameters	24
3-7	A-P-ABORT Indication Parameters	25
3-8	P-DATA Indication Parameters	25
3-9	TCP SEND Arguments	26
3-10	TCP Passive OPEN Arguments	27
A-1	Mapping of OSI Primitives to XTI System Calls	29
A-2	Mapping of OSI Primitives to BSD Sockets System Calls	30

Preface

The SQL Access Group, founded in 1989, is a consortium of major vendors and users of database software and hardware. It is committed to the promotion of worldwide, multi-vendor interoperability and portability of applications across distributed, SQL-based relational database systems. This goal is manifested in three primary activities:

- Development of specifications, both for application programming interfaces, based on the existing SQL standard (ISO SQL), and for communication protocols, based on the existing Remote Database Access standard (ISO RDA).
- Validation of these specifications through implementation of prototypes on multiple platforms.
- Promotion of these specifications, both by making them available to others for general industry use and by submitting them for consideration by standards-writing bodies, including the International Standards Organization (ISO) and the American National Standards Institute (ANSI).

The SQL Access Group bases its work on existing or emerging standards, and supplements these by specification of the additional details and agreements necessary to ensure database interoperability and portability.

This Document

This document specifies SQL client-server interoperability in a TCP/IP environment. It endorses the ISO Remote Database Access (RDA) standards: ISO/IEC 9579-1 and ISO/IEC 9579-2; it specifies how to apply these standards to the TCP/IP environment. It applies to the RDA Basic application-context only and does not support the two-phase commit protocol used in the RDA Transaction Processing application-context. This specification should be used in conjunction with the X/Open Specification, **SQL Remote Database Access**.

The provisions of the X/Open Specification, **SQL Remote Database Access** apply except as modified by this specification.

Trade Marks

SQL Access® is a registered trade mark of the SQL Access Group in the U.S.A.

UNIX® is a registered trade mark of UNIX System Laboratories, Inc. in the U.S.A. and other countries.

X/Open™ and the ‘‘X’’ device are trade marks of X/Open Company Limited in the U.K. and other countries.

Referenced Documents

The following documents are referenced in this specification:

[1] RFC 793, Postel, J.B., Transmission Control Protocol, September 1981.

[2] RFC 1122, Braden, R.T., ed., Requirements for Internet Hosts — communication layers, October 1989, Internet.

[3] ISO/IEC 9579-1: —*, Information technology — Open Systems Interconnection — Remote Database Access — Part 1: Generic Model, Service and Protocol.

[4] ISO/IEC 9579-2: —*, Information technology — Open Systems Interconnection — Remote Database Access — Part 2: SQL Specialization.

[5] ISO 8649: 1988, Information processing systems — Open Systems Interconnection — Service definition for the Association Control Service Element.

[6] ISO 8822: 1988, Information processing systems — Open Systems Interconnection — Connection oriented presentation service definition.

[7] ISO 8824: 1990, Information technology — Open Systems Interconnection — Specification of Abstract Syntax Notation One (ASN.1).

[8] ISO 8825: 1990, Information technology — Open Systems Interconnection — Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1).

[9] RFC 1006, Rose, M.T. & Cass, D.E., ISO Transport Services on top of the TCP: Version 3, May 1987, Internet.

[10] X/Open CAE Specification, expected to be published in July 1993, SQL Remote Database Access (ISBN: 1-872630-98-7 C307).

[11] X/Open CAE Specification, January 1992, X/Open Transport Interface (XTI) (ISBN: 1-872630-29-4, XO/CAE/91/600 or C196).

* To be published (1993).

Chapter 1

Overview

This document specifies a model for supporting the Remote Database Access (RDA) Service, as specified in the X/Open **SQL Remote Database Access** specification, on top of the Transmission Control Protocol and Internet Protocol (TCP/IP). This model allows for the creation of an RDA service element in the TCP/IP environment without the overhead of an OSI protocol stack.

Fundamentally, the model described in this specification consists of a service mapping, whereby the underlying communication services needed by RDA and its users are mapped onto the services of TCP/IP. Normally, in the OSI environment, these communication services are provided by the Association Control Service Element (ACSE) and the OSI Presentation Layer (as specified in [5] and [6]). However, in the model described, ACSE, the Presentation Layer and all lower layers are replaced by TCP/IP and a small mapping functionality.

Within the subsequent sections, this specification makes use of a conceptual 'mapper' layer to model the mapping functionality needed to adapt RDA to TCP/IP. This layer is referred to as the RDA-TCP mapper. The RDA-TCP mapper layer replaces ACSE and the Presentation Layer by providing the precise subset of their services needed to support RDA. Thus, it targets the minimal amount of functionality and overhead necessary to achieve the desired result.

The services of the RDA-TCP mapper layer are defined in terms of the ACSE and Presentation Layer services they replace and the restrictions upon those services that must be observed by the mapper's users. This model for the specification of the RDA to TCP/IP mapping has the following benefits:

- It defines the RDA-TCP mapper in terms familiar to those with an understanding of RDA and ACSE/Presentation.
- It preserves the nature of the relationship between RDA, its users and the layers they use to communicate, thus dovetailing cleanly with the RDA specification.
- In the cases where software architecture directly reflects the OSI model, the emulated ACSE and Presentation services should provide for easy adaptation to existing interfaces.

In describing the actual mapping functionality performed by the RDA-TCP mapper, the text makes use of the User/TCP Interface model, as defined in [1].

The SQL Access Group does not preclude the use of RFC 1006 [9] to use RDA over TCP/IP. The simple solution specified in this document may be more suitable for environments where the complexity of RFC 1006 is not desirable. Note that the protocol specified in this document and a protocol using RFC 1006 will not be identical and thus will not interoperate.

Chapter 2

Services

The services provided by the RDA-TCP mapper are a subset of those provided by ACSE and the Presentation Layer. In addition, the user of the RDA-TCP mapper (RDA and its service-user) is limited in the way in which it may make use of these services. These limitations serve to restrict the capabilities of the RDA-TCP mapper to the exact set of ACSE/Presentation services needed by an RDA entity.

Table 2-1 lists the services provided by the RDA-TCP mapper.

Table 2-1 RDA-TCP Mapper Services

Service	Purpose
A-ASSOCIATE	Association establishment
A-RELEASE	Normal association termination
A-ABORT and A-P-ABORT	Abnormal association termination
P-DATA	Data transfer

The following subsections describe each of the services of the RDA-TCP mapper in terms of the ACSE and Presentation Layer services they replace and the limitations incumbent upon their use. The semantics of these services and their associated parameters are defined in the ACSE and Presentation Layer service specifications ([5] and [6]). In addition to any limitations stated in the following subsections, all rules specified in [3] and [4], including those for the RDA Basic application-context, must also be observed.

This chapter uses object identifiers that are defined by other documents. For the **X/Open RDA** object identifiers, see [10]. For the **ISO RDA** and **ACSE** object identifiers, see [4].

The following abbreviations are used throughout the remainder of this document.

AE application-entity

AP application-process

APDU application-protocol-data-unit.

2.1 A-ASSOCIATE

The A-ASSOCIATE service is used to establish an association between peer RDA entities.

The A-ASSOCIATE service mimics that provided by the ACSE A-ASSOCIATE service with the following exceptions:

- When invoking the A-ASSOCIATE service, the user is limited to specifying the name of an RDA Basic application-context.
- Most of the Presentation and Session Layer parameters embedded within the A-ASSOCIATE service are restricted to a particular value or, if optional, are required not to be present.
- The A-ASSOCIATE service may not convey user information.

The parameters of the A-ASSOCIATE service are defined in terms of their corresponding ACSE parameters as follows:

Mode

Possible values: fixed (‘normal’).

This parameter specifies the mode of operation for the association. For all A-ASSOCIATE primitives in which it appears, the Mode parameter is required to have a fixed value specifying the normal mode of operation.

Application Context Name

Possible values: restricted

This parameter identifies the application-context to be used for the association. Since the RDA-TCP mapper supports the RDA Basic application-context only, an Application Context Name for the RDA TP application-context must not be used. There are two Application Context Names currently defined for the RDA Basic application-context.

Table 2-2 Application Context Names

Name	Value
ISO RDA	{iso(1) standard(0) rda(9579) part-2(2) basic-ac(2) version-1(1)}
X/Open RDA	{iso(1) national-member-body(2) uk(826) national(0) x-open(1050) data-management(3) x-open-rdasql(2) basic-ac(2) version-1(1)}

Calling AP Title, AE Qualifier, AP Invocation-identifier and AE Invocation-identifier

Possible values: any.

These parameters identify the requestor of the A-ASSOCIATE service. There are no restrictions on the values of these parameters. Their use and semantics are identical to those of the corresponding parameters within the ACSE A-ASSOCIATE primitive.

Called AP Title, AE Qualifier, AP Invocation-identifier and AE Invocation-identifier

Possible values: any.

These parameters identify the intended acceptor of the A-ASSOCIATE service. There are no restrictions on the values of these parameters. Their use and semantics are identical to those of the corresponding parameters within the ACSE A-ASSOCIATE primitive.

Responding AP Title, AE Qualifier, AP Invocation-identifier and AE Invocation-identifier

Possible values: restricted.

These parameters identify the actual acceptor of the A-ASSOCIATE service. For all A-ASSOCIATE primitives in which they appear, the values of these parameters are restricted such that the actual acceptor of the A-ASSOCIATE service never differs from the intended acceptor.

For the response primitive, the user must always specify the same values as that conveyed in the corresponding Called parameters of the indication primitive.

For the confirm primitive, the RDA-TCP mapper always returns the values specified by the user in the corresponding Called parameters of the request primitive.

User Information

Possible values: none.

The User Information parameter of the A-ASSOCIATE service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

Result

Possible values : fixed (‘accepted’).

The Result parameter indicates the result of using the A-ASSOCIATE service. For each A-ASSOCIATE primitive in which it appears, this parameter is required to have a fixed value specifying that the association was accepted.

Result Source

Possible values: fixed (‘ACSE service-user’).

The Result Source parameter identifies the creating source of the Result parameter. For each A-ASSOCIATE primitive in which it appears, the Result Source parameter must have a fixed value identifying the ACSE service-user.

Diagnostic

Possible values: none.

The Diagnostic parameter of the A-ASSOCIATE service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

Calling Presentation Address

Possible values: restricted.

This parameter conveys the presentation-address of the calling RDA entity. For all A-ASSOCIATE primitives in which it appears, the value of the Calling Presentation Address parameter is restricted.

For the request primitive, the user must specify a Calling Presentation Address with the following contents:

- no Presentation Layer selector
- no Session Layer selector
- no Transport Layer selector
- a single network address containing the 32-bit IP address and the 16-bit port number of the calling RDA entity.

For the indication primitive, the RDA-TCP mapper returns the same value as that specified on the request primitive.

Called Presentation Address

Possible values: restricted.

This parameter conveys the presentation-address of the called RDA entity. For all A-ASSOCIATE primitives in which it appears, the value of the Called Presentation Address parameter is restricted.

For the request primitive, the user must specify a Called Presentation Address with the following contents:

- no Presentation Layer selector
- no Session Layer selector
- no Transport Layer selector
- a single network address containing the 32-bit IP address and the 16-bit port number of the called RDA entity

For the indication primitive, the RDA-TCP mapper returns the same value as that specified on the request primitive.

Responding Presentation Address

Possible values: restricted.

This parameter conveys the presentation-address of the responding RDA entity. For all A-ASSOCIATE primitives in which it appears, the value of this parameter is restricted such that the Responding Presentation Address never differs from the Called Presentation Address.

For the response primitive, the user must always specify the same value as that conveyed in the Called Presentation Address parameter of the indication primitive.

For the confirm primitive, the RDA-TCP mapper always returns the same value as that specified by the user in the Called Presentation Address parameter of the request primitive.

Presentation Context Definition List

Possible values: restricted.

The Presentation Context Definition List parameter identifies the presentation contexts to be used over the association. For each primitive in which it appears, this parameter shall have a value identifying two contexts: ACSE and RDA. The definition of this value is dependent on the Application Context Name specified in the A-ASSOCIATE primitive. See the following tables for the precise definition of this value. In the tables **ID** is the Presentation Context Identifier.

Table 2-3 Presentation Context Definition List Value for X/Open RDA Application-context

Context	ID	Syntax Name
ACSE	1	**Abstract Syntax Name** {joint-iso-ccitt(2) association-control(2) abstract-syntax(1) apdus(0) version(1)}
		Transfer Syntax Name {joint-iso-ccitt(2) asn1(1) basic-encoding(1)}
X/Open RDA	3	**Abstract Syntax Name** {iso(1) national-member-body(2) uk(826) national(0) x-open(1050) data-management(3) x-open-rdasql(2) abstract-syntax(1) version-1(1)
		Transfer Syntax Name {joint-iso-ccitt(2) asn1(1) basic-encoding(1)}

Table 2-4 Presentation Context Definition List Value for ISO RDA Application-context

Context	ID	Syntax Name
ACSE	1	**Abstract Syntax Name** {joint-iso-ccitt(2) association-control(2) abstract-syntax(1) apdus(0) version(1)}
		Transfer Syntax Name {joint-iso-ccitt(2) asn1(1) basic-encoding(1)}
ISO RDA	3	**Abstract Syntax Name** {iso(1) standard(0) rda(9579) part-2(2) abstract-syntax(1) version-1(1)
		Transfer Syntax Name {joint-iso-ccitt(2) asn1(1) basic-encoding(1)}

Presentation Context Definition Result List

Possible values: fixed.

The Presentation Context Definition Result List parameter indicates the acceptance or rejection of the presentation contexts proposed in the Presentation Context Definition List parameter. For each primitive in which it appears, this parameter shall have a fixed value declaring the acceptance of the proposed ACSE and RDA contexts. The precise definition of this value is as follows:

Table 2-5 Presentation Context Definition Result List Value

Context	Negotiation Result	Selected Transfer Syntax Name	Rejection Reason
ACSE	acceptance	{joint-iso-ccitt(2) asn1(1) basic-encoding(1)}	(no value)
RDA	acceptance	{joint-iso-ccitt(2) asn1(1) basic-encoding(1)}	(no value)

Default Presentation Context Name

Possible values: none.

The Default Presentation Context Name parameter of the A-ASSOCIATE service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

Default Presentation Context Result

Possible values: none.

The Default Presentation Context Result parameter of the A-ASSOCIATE service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

Quality of Service

Possible values: none.

The Quality of Service parameter of the A-ASSOCIATE service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

Presentation Requirements

Possible values: fixed (‘kernel’).

The Presentation Requirements parameter identifies the optional presentation functional units to be supported for the association. For each A-ASSOCIATE primitive, this parameter shall have a fixed value identifying only the presentation kernel functional unit.

Session Requirements

Possible values: fixed (‘kernel’ and ‘duplex’).

The Session Requirements parameter identifies the session functional units to be supported for the association. For each A-ASSOCIATE primitive, this parameter shall have a fixed value identifying only the session kernel and duplex functional units.

Initial Synchronization Point Serial Number

Possible values: none.

The Initial Synchronization Point Serial Number parameter of the A-ASSOCIATE service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

Initial Assignment of Tokens

Possible values: none.

The Initial Assignment of Tokens parameter of the A-ASSOCIATE service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

Session-connection Identifier

Possible values: none.

The Session-connection Identifier parameter of the A-ASSOCIATE service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

Table 2-6 summarises the parameters of the A-ASSOCIATE service and their restrictions.

Table 2-6 A-ASSOCIATE Parameters

Parameter Name	Request	Indication	Response	Confirm
Mode	F	F		
Application Context Name	M	M	M	M
Calling AP Title	M	M(=)		
Calling AE Qualifier	M	M(=)		
Calling AP Invocation-identifier	M	M(=)		
Calling AE Invocation-identifier	M	M(=)		
Called AP Title	U	C(=)		
Called AE Qualifier	U	C(=)		
Called AP Invocation-identifier	U	C(=)		
Called AE Invocation-identifier	U	C(=)		
Responding AP Title			C(R)	C(=)
Responding AE Qualifier			C(R)	C(=)
Responding AP Invocation-identifier			C(R)	C(=)
Responding AE Invocation-identifier			C(R)	C(=)
User Information	-	-	-	-
Result			F	F
Result Source				F
Diagnostic			-	-
Calling Presentation Address	M(R)	M(=)		
Called Presentation Address	M(R)	M(=)		
Responding Presentation Address			M(R)	M(=)
Presentation Context Definition List	M	M		
Presentation Context Definition Result List		F	F	F
Default Presentation Context Name	-	-		

Parameter Name	Request	Indication	Response	Confirm
Default Presentation Context Result			-	-
Quality of Service	-	-	-	-
Presentation Requirements	F	F	F	F
Session Requirements	F	F	F	F
Initial Synchronization Point Serial Number	-	-	-	-
Initial Assignment of Tokens	-	-	-	-
Session-connection Identifier	-	-	-	-

M A value must always be supplied for the parameter.

C The existence and values of the parameter are conditional on rules stated in the description of the parameter.

F A fixed value must always be supplied for the parameter.

U A value may be supplied for the parameter at the user's option.

- Use of the parameter is prohibited. No value may be supplied.

blank The parameter does not apply.

(=) The value of the parameter is equal to the value of the parameter indicated in the column immediately to the left.

(R) The range of values that may be supplied for the parameter is restricted.

2.2 A-RELEASE

The A-RELEASE service is used to effect an orderly termination of an existing association.

The A-RELEASE service mimics that of the ACSE A-RELEASE service with the following exceptions:

- The A-RELEASE service does not guarantee the delivery of any information in transit at the time the service is invoked.
- The A-RELEASE service cannot be refused by the acceptor and always completes successfully.
- Due to limitations in the underlying TCP service, the invocation of an A-RELEASE request primitive results in the reception of an A-P-ABORT indication by the peer entity, instead of an A-RELEASE indication as normal for the ACSE service. Thus, the ACSE A-RELEASE indication and A-RELEASE response primitives are not provided by the RDA-TCP mapper service.

  Note, however that, as in the ACSE service, the invocation of an A-RELEASE request primitive results in the generation of a corresponding A-RELEASE confirm.
- The A-RELEASE service may not convey any user information.

The parameters of the A-RELEASE service are defined in terms of their corresponding ACSE parameters as follows:

Reason

Possible values: none.

The Reason parameter of the A-RELEASE service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

User Information

Possible values: none.

The User Information parameter of the A-RELEASE service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

Result

Possible values: fixed (‘affirmative’).

The Result parameter is used by the acceptor to indicate if the request to release the association is acceptable. For all A-RELEASE primitives, this parameter is required to have a fixed value indicating that the association release is acceptable.

Table 2-7 summarises the parameters of the A-RELEASE service.

Table 2-7 A-RELEASE Parameters

Parameter Name	Request	Confirm
Reason	-	-
User Information	-	-
Result		F

F A fixed value must always be supplied for the parameter.
\- Use of the parameter is prohibited. No value may be supplied.
blank The parameter does not apply.

2.3 A-ABORT

The A-ABORT service is used by the requestor to cause the abnormal release of an existing association.

The A-ABORT service mimics that provided by the ACSE A-ABORT service with the following exceptions:

- The abort source is always the ACSE service-user.
- Due to limitations in the underlying TCP service, the invocation of an A-ABORT request primitive results in the reception of an A-P-ABORT indication by the peer entity, instead of an A-ABORT indication as normal for the ACSE service. Thus, the ACSE A-ABORT indication primitive is not provided by the RDA-TCP mapper service.
- The A-ABORT service may not convey user information.

The parameters of the A-ABORT service are defined in terms of their corresponding ACSE parameters as follows:

Abort Source

Possible values: none.

The Abort Source is always the ACSE service-user.

User Information

Possible values: none.

The User Information parameter of the A-ABORT service cannot be used. No value may be supplied by either the RDA-TCP mapper or its user.

Table 2-8 summarises the parameters of the A-ABORT service.

Table 2-8 A-ABORT Parameters

Parameter Name	Request
Abort Source	
User Information	-

- Use of the parameter is prohibited. No value may be supplied.

blank The parameter does not apply.

2.4 A-P-ABORT

The A-P-ABORT service is used to indicate the abnormal release of an existing association due to problems in the services below the RDA-TCP mapper.

The A-P-ABORT service provides identical functionality to the ACSE A-P-ABORT service.

The parameters of the A-P-ABORT service are defined in terms of their corresponding ACSE parameters as follows:

Provider Reason

Possible values: fixed (‘reason-not-specified’).

The Provider Reason parameter indicates the reason for the termination of the association. For the A-P-ABORT primitive in which it appears, this parameter is required to have the fixed value ‘reason-not-specified’.

Table 2-9 summarises the parameters of the A-P-ABORT service.

Table 2-9 A-P-ABORT Parameters

Parameter Name	Indication
Provider Reason	F

F A fixed value must always be supplied for the parameter.

2.5 P-DATA

The P-DATA service provides for the transmission of user information over an existing association.

The P-DATA service mimics that provided by the Presentation Layer P-DATA service, with the following exceptions:

- Only a single presentation data value, with no embedded presentation data values, may be transmitted.
- The presentation data value must be from the RDA presentation context.

The parameters of the P-DATA service are defined in terms of their corresponding Presentation parameters as follows:

User data

Possible values: restricted.

The User data parameter conveys the presentation data value being communicated. The possible values of this parameter are restricted as defined above.

Table 2-10 summarises the parameters of the P-DATA service.

Table 2-10 P-DATA Parameters

Parameter Name	Request	Indication
User data	M(R)	M(=)

M A value must always be supplied for the parameter.
(=) The value of the parameter is equal to the value of the parameter indicated in the column immediately to the left.
(R) The range of values that may be supplied for the parameter is restricted.

Chapter 3

Mapping onto TCP

The RDA-TCP mapper is responsible for mapping the services described in Chapter 2 onto the services provided by the underlying TCP. In describing the actual mapping functionality performed by the RDA-TCP mapper, the following description makes use of the User/TCP Interface model, as described in [1]. In mapping these services, the RDA-TCP mapper can be viewed as a finite state machine driven by the reception of events from the service-user (RDA) and from the TCP service-provider.

In this chapter, the mapping performed by the RDA-TCP mapper is described in terms of the TCP services used, the types of data exchanged between peer mapper layers and the procedures used for handling protocol events.

3.1 Use of TCP Service

The RDA-TCP mapper makes use of the following TCP services defined in [1]:

- Active OPEN
- Passive OPEN
- RECEIVE
- SEND
- CLOSE.

3.1.1 TCP Service Options

When using the TCP service, RDA-TCP mapper shall adhere to the following requirements regarding the use of TCP options:

TCP PUSH and URGENT Options

The RDA-TCP mapper shall not use the TCP PUSH or URGENT options when transmitting and receiving data.

Connection Orderly Release

The RDA-TCP mapper does not require the use of the TCP orderly release capability to ensure the delivery of data. The proper sequencing of RDA service requests — that is, issuing R-Terminate request and waiting for the confirm prior to issuing A-RELEASE — results in the delivery of all data prior to release of the association.

TCP Keep-alives

This specification recommends the use of the TCP keep-alive service (as defined in [2]) to ensure the integrity of idle connections).

3.2 Mapper APDUs

The RDA-TCP mapper communicates with its peer using two sets of APDUs. During association establishment, the mappers exchange RDATCP APDUs to negotiate protocol version numbers and to communicate Application Layer addressing information. Once the association has been established, RDA APDUs (as defined by [3] and [4]) are communicated directly. In both cases, APDUs are conveyed as user data by the underlying TCP service.

3.2.1 RDATCP APDU Syntax

The ASN.1 syntax for the RDATCP APDUs is defined as follows:

```
SQL-ACCESS-RDA-TCP-IP DEFINITIONS ::=
BEGIN
  IMPORTS
    Application-context-name, AP-title, AE-qualifier,
    AP-invocation-identifier, AE-invocation-identifier
    FROM ACSE-1 { joint-iso-ccitt standard acse(8650) };

  RDATCP-apdu ::= CHOICE
  {
    areq  AREQ-apdu,
    arsp  ARSP-apdu
  }

  AREQ-apdu ::= [ PRIVATE 0 ] IMPLICIT SEQUENCE
  {
    protocol-version                [0] IMPLICIT BIT STRING
                                       { version1(0) } DEFAULT { version1 },
    application-context-name        [1] IMPLICIT Application-context-name,
    calling-AP-title                [2] IMPLICIT AP-title                 OPTIONAL,
    calling-AE-qualifier            [3] IMPLICIT AE-qualifier             OPTIONAL,
    calling-AP-invocation-identifier [4] IMPLICIT AP-invocation-identifier OPTIONAL,
    calling-AE-invocation-identifier [5] IMPLICIT AE-invocation-identifier OPTIONAL,
    called-AP-title                 [6] IMPLICIT AP-title                 OPTIONAL,
    called-AE-qualifier             [7] IMPLICIT AE-qualifier             OPTIONAL,
    called-AP-invocation-identifier  [8] IMPLICIT AP-invocation-identifier OPTIONAL,
    called-AE-invocation-identifier  [9] IMPLICIT AE-invocation-identifier OPTIONAL,
    implementation-information      [10] IMPLICIT Implementation-data      OPTIONAL
  }

  ARSP-apdu ::= [ PRIVATE 1 ] IMPLICIT SEQUENCE
  {
    protocol-version                [0] IMPLICIT BIT STRING
                                       { version1(0) } DEFAULT { version1 },
    implementation-information      [1] IMPLICIT Implementation-data      OPTIONAL
  }

  Implementation-data ::= GraphicString

END
```

3.2.2 APDU Encoding

All APDUs transmitted by the RDA-TCP mapper are encoded using the Basic Encoding Rules for ASN.1 ([8]). APDUs may be encoded using any of the encoding formats allowed by BER with the following exception: when encoding the outermost SET or SEQUENCE of an APDU, the BER definite length encoding format must be used.

3.3 Mapping Procedures

The following subsections describe the procedures used by the RDA-TCP mapper to map the services described in Chapter 2 onto the TCP service.

3.3.1 A-ASSOCIATE Request Procedure

An A-ASSOCIATE request primitive is invoked by the service-user to initiate an application-association.

The procedure for processing an A-ASSOCIATE request is as follows:

1. When an A-ASSOCIATE request primitive is invoked by the service user, the RDA-TCP mapper shall attempt to establish a TCP connection with the requested peer using the TCP Active OPEN command. The arguments of the TCP OPEN command shall be as follows:

Table 3-1 TCP Active OPEN Arguments

Argument Name	Value
Local port	(locally assigned TCP port)*
Foreign socket	(TCP port derived from Called Presentation Address parameter of A-ASSOCIATE request)
Active/Passive	ACTIVE
Timeout	(locally defined timeout duration; or FOREVER)*
Precedence	0
Security/compartment	(locally assigned security/compartment information)*
Options	NONE

2. The RDA-TCP mapper shall then await the completion of the TCP OPEN command.
3. If the TCP service fails to establish a connection, the RDA-TCP mapper shall generate an A-P-ABORT indication to the service-user as described in Section 3.3.8 on page 24.

* Source for value is implementation defined.

4. If a connection is successfully established, the RDA-TCP mapper shall encode an AREQ APDU and send it to the peer entity using the TCP SEND command. The contents of the AREQ APDU shall be:

Table 3-2 AREQ APDU Contents

AREQ APDU Field	Value
protocol-version	version1
application-context-name	(derived from A-ASSOCIATE request parameter)
calling-AP-title	(derived from A-ASSOCIATE request parameter)
calling-AE-qualifier	(derived from A-ASSOCIATE request parameter)
calling-AP-invocation-identifier	(derived from A-ASSOCIATE request parameter)
calling-AE-invocation-identifier	(derived from A-ASSOCIATE request parameter)
called-AP-title	(derived from A-ASSOCIATE request parameter)
called-AE-qualifier	(derived from A-ASSOCIATE request parameter)
called-AP-invocation-identifier	(derived from A-ASSOCIATE request parameter)
called-AE-invocation-identifier	(derived from A-ASSOCIATE request parameter)
implementation-information	1 through 255 characters

5. The RDA-TCP mapper shall then await the reception of an ARSP APDU by following the procedure described in Section 3.3.4 on page 22.

3.3.2 A-ASSOCIATE Indication Procedure

An A-ASSOCIATE indication primitive is generated by the RDA-TCP mapper to indicate a peer entity's desire to establish an application-association.

The procedure for generating an A-ASSOCIATE indication is as follows:

1. When the RDA-TCP mapper detects the existence of an incoming connection (via the completion of a TCP Passive OPEN command), it shall immediately await the reception of an AREQ APDU by issuing a TCP RECEIVE command.

   If, prior to the reception of the AREQ APDU, the TCP connection is aborted or closed by the peer entity, the RDA-TCP mapper shall close its end of the connection using the TCP CLOSE command and ignore the association attempt.

2. As an option, the RDA-TCP mapper may set a timer to ensure the timely reception of the AREQ APDU. If the timer expires prior to the reception of the APDU, the mapper should close its end of the connection using the TCP CLOSE command and ignore the association attempt.

3. When the AREQ APDU is received, the RDA-TCP mapper shall decode its contents as described in Section 3.2.2 on page 18.

   If an error is detected in the encoding of the AREQ APDU, the RDA-TCP mapper shall discard the APDU, close its end of the TCP connection using the TCP CLOSE command and ignore the association attempt.

4. The RDA-TCP mapper shall then verify the protocol-version component present in the AREQ APDU. The protocol version chosen for the association shall be the highest version supported by both entities.

   If no protocol versions are supported in common, the RDA-TCP mapper shall discard the AREQ APDU, close its end of the TCP connection using the TCP CLOSE command and ignore the association attempt.

5. If a common protocol version is found, the RDA-TCP mapper shall then generate a A-ASSOCIATE indication to the service user. The A-ASSOCIATE indication shall contain the following parameters:

Table 3-3 A-ASSOCIATE Indication Parameters

Parameter Name	Value
Mode	normal
Application Context Name	(derived from application-context name field in AREQ APDU)
Calling AP Title	(from calling-AP-title field in AREQ APDU)
Calling AE Qualifier	(from calling-AE-qualifier field in AREQ APDU)
Calling AP Invocation-identifier	(from calling-AP-invocation-identifier field in AREQ APDU)
Calling AE Invocation-identifier	(from calling-AE-invocation-identifier field in AREQ APDU)
Called AP Title	(from called-AP-title field in AREQ APDU)
Called AE Qualifier	(from called-AE-qualifier field in AREQ APDU)
Called AP Invocation-identifier	(from called-AP-invocation-identifier field in AREQ APDU)
Called AE Invocation-identifier	(from called-AE-invocation-identifier field in AREQ APDU)
Calling Presentation Address	(derived from result of TCP Passive OPEN command)
Called Presentation Address	(derived from locally assigned TCP port)
Presentation Context Definition List	(derived from application-context name field in AREQ APDU as defined in Table 2-3 and Table 2-4 on page 7)
Presentation Context Definition Result List	(as defined in Table 2-5 on page 8)
Presentation Requirements	kernel
Session Requirements	kernel and duplex

3.3.3 A-ASSOCIATE Response Procedure

An A-ASSOCIATE response primitive is invoked by the service-user to indicate its willingness to accept an application-association. If the service-user is not willing to accept the application-association, it shall invoke A-ABORT, as described in Section 3.3.7 on page 24.

The procedure for processing an A-ASSOCIATE response primitive is as follows:

1. When an A-ASSOCIATE response primitive is invoked by the service-user, the RDA-TCP mapper shall encode an ARSP APDU and send it to the peer entity using the TCP SEND command. The contents of the ARSP APDU shall be:

Table 3-4 ARSP APDU Contents

ARSP APDU Field	Value
protocol-version	version1
implementation-information	1 through 255 characters

2. Upon sending the ARSP APDU, the application-association is considered established.

3.3.4 A-ASSOCIATE Confirm Procedure

An A-ASSOCIATE confirm primitive is generated by the RDA-TCP mapper to indicate the success of an association establishment attempt.

The procedure for generating an A-ASSOCIATE confirm is as follows:

1. After sending an AREQ APDU (as described in Section 3.3.1 on page 19), the RDA-TCP mapper shall await the reception of a corresponding ARSP APDU by performing a TCP RECEIVE command.
2. As an option, the RDA-TCP mapper may set a timer to ensure the timely reception of the ARSP APDU. If the timer expires prior to the reception of the APDU, the mapper should generate an A-P-ABORT indication according to the procedure described in Section 3.3.8 on page 24.
3. When the ARSP APDU is received, the contents of the APDU shall be decoded as per Section 3.2.2 on page 18.

   If an error is detected in the encoding of the APDU, the RDA-TCP mapper shall discard the APDU and generate an A-P-ABORT indication according to the procedure described in Section 3.3.8 on page 24.

4. If the ARSP APDU is successfully decoded, the RDA-TCP mapper shall generate an A-ASSOCIATE confirm primitive to the service-user. The parameters of the A-ASSOCIATE confirm shall be:

Table 3-5 A-ASSOCIATE Confirm Parameters

Parameter Name	Value
Application Context Name	(same as Application Context Name parameter of A-ASSOCIATE request)
Responding AP Title	(same as Called AP Title parameter of A-ASSOCIATE request)
Responding AE Qualifier	(same as Called AE Qualifier parameter of A-ASSOCIATE request)
Responding AP Invocation-identifier	(same as Called AP Invocation-identifier parameter of A-ASSOCIATE request)
Responding AE Invocation-identifier	(same as Called AE Invocation-identifier parameter of A-ASSOCIATE request)
Result	accepted
Result Source	ACSE service-user
Responding Presentation Address	(same as Called Presentation Address parameter of A-ASSOCIATE request)
Presentation Context Definition Result List	(as defined in Table 2-5 on page 8)
Presentation Requirements	kernel
Session Requirements	kernel and duplex

5. Upon generating the A-ASSOCIATE confirm, the association is considered established.

3.3.5 A-RELEASE Request Procedure

An A-RELEASE request primitive is invoked by the service user to release an existing application-association.

The procedure for processing an A-RELEASE request primitive is as follows:

1. When an A-RELEASE request primitive is invoked by the user, the RDA-TCP mapper shall close its end of the TCP connection using the TCP CLOSE command.
2. The RDA-TCP mapper shall then generate an A-RELEASE confirm primitive as specified in Section 3.3.6 on page 24.

3.3.6 A-RELEASE Confirm Procedure

An A-RELEASE confirm primitive is generated by the RDA-TCP mapper to indicate the successful termination of an application-association.

The procedure for generating an A-RELEASE confirm primitive is as follows:

1. After an A-RELEASE request primitive has been received and processed as described in Section 3.3.5 on page 23, the RDA-TCP mapper shall immediately generate an A-RELEASE confirm primitive back to the service-user. The parameters of the A-RELEASE confirm shall be:

Table 3-6 A-RELEASE Confirm Parameters

Parameter Name	Value
Result	affirmative

2. After the A-RELEASE confirm primitive has been generated, the application-association shall be considered terminated.

3.3.7 A-ABORT Request Procedure

An A-ABORT request primitive is invoked by the service user to terminate immediately an existing application-association.

The procedure for processing an A-ABORT request is as follows:

1. When an A-ABORT request primitive is invoked by the service user, the RDA-TCP mapper shall close its end of the TCP connection using the TCP CLOSE command.
2. The application-association shall then be considered terminated.

3.3.8 A-P-ABORT Indication Procedure

An A-P-ABORT indication primitive is generated by the RDA-TCP mapper to indicate the abnormal termination of an existing application-association.

The procedure for generating an A-P-ABORT indication primitive is as follows:

1. When any one of the following events occurs the mapper shall generate an A-P-ABORT indication primitive to the service-user:
   - the RDA-TCP mapper is unable to establish a TCP connection using the TCP Active OPEN command
   - the RDA-TCP mapper detects that the TCP connection has disappeared (such as when using the TCP SEND or RECEIVE commands).
   - an unknown or invalid APDU is received.

The A-P-ABORT indication shall have the following parameters:

Table 3-7 A-P-ABORT Indication Parameters

Parameter Name	Value
Provider Reason	reason-not-specified

2. If the TCP connection is still Active, then the RDA-TCP mapper shall close its end of the TCP connection using the TCP CLOSE command.
3. The application-association shall then be considered terminated.

3.3.9 P-DATA Request Procedure

A P-DATA request primitive is invoked by the service-user to transmit an RDA APDU to the peer entity.

The procedure for processing a P-DATA request is as follows:

1. When a P-DATA request primitive is invoked by the service-user, the RDA-TCP mapper shall encode the RDA APDU contained within the User data parameter and send the encoded APDU using the TCP SEND command.

3.3.10 P-DATA Indication Procedure

A P-DATA indication primitive is generated by the RDA-TCP mapper to indicate the reception of an RDA APDU from the peer entity.

The procedure for generating a P-DATA indication is as follows:

1. When an application-association has been established, the RDA-TCP mapper shall continuously await the reception of RDA APDUs by performing TCP RECEIVE commands.
2. When an RDA APDU is received, the RDA-TCP mapper shall decode the APDU as specified in Section 3.2.2 on page 18.

   If an error is detected in the encoding of the APDU, the RDA-TCP mapper shall discard the APDU and generate an A-P-ABORT indication according to the procedure described in Section 3.3.8 on page 24.
3. After successfully decoding the APDU, the RDA-TCP mapper shall generate a P-DATA indication primitive to the service-user with the following parameters:

Table 3-8 P-DATA Indication Parameters

Parameter Name	Value
User data	(decoded RDA APDU)

3.3.11 APDU Transmission

APDUs are transmitted using the TCP SEND command. When invoking this command, the RDA-TCP mapper shall supply the following arguments:

Table 3-9 TCP SEND Arguments

Argument Name	Value
Local Connection Name	(Connection identifier returned by TCP OPEN command)
Buffer address	(Address of data buffer containing encoded APDU)
Byte count	(Number of octets in APDU encoding)
PUSH flag	FALSE
URGENT flag	FALSE

3.3.12 APDU Reception

The RDA-TCP mapper uses the TCP RECEIVE command to receive incoming APDUs. Due to the stream-oriented nature of the TCP service, a single APDU may be received over several invocations of the RECEIVE command. However, because of the encoding restrictions described in Section 3.2.2 on page 18, the RDA-TCP mapper may use the following simple procedure to recover the boundaries of an individual APDU:

1. Invoke the TCP RECEIVE command to receive the tag octet of the APDU. This octet identifies the type of APDU being communicated (either an RDATCP-apdu or RDA-apdu).
2. Receive the second octet of the APDU to determine its length.
3. If the high-order bit of the second octet is 0, the length of the APDU is conveyed in the value of the octet.
4. If the high-order bit of the second octet is 1, the length of the APDU is conveyed in one or more subsequent octets, in most significant to least significant order. The number of subsequent octets containing the length is specified by the remaining bits in the second octet.
5. Once the type and length of the APDU have been received, the RDA-TCP mapper can allocate a data buffer of an appropriate size and repeatedly invoke the TCP RECEIVE command until the entire APDU has been received.

3.3.13 TCP Connection Listening

When supporting RDA server entities, the RDA-TCP mapper shall listen for incoming TCP connections by repeatedly invoking the TCP Passive OPEN command. When invoking this command, the RDA-TCP mapper shall supply the following arguments:

Table 3-10 TCP Passive OPEN Arguments

Argument Name	Value
Local port	(locally assigned TCP port)*
Foreign socket	NONE
Active/Passive	PASSIVE
Timeout	FOREVER
Precedence	0
Security/compartment	(locally assigned security/compartment information)*
Options	NONE

* Source for value is implementation defined.

Appendix A

OSI Primitives Mapping

This appendix describes the mapping of OSI primitives to two common TCP implementations. It is an informative annex. The following tables are for illustration only and are not conformance requirements for this specification.

A.1 X/Open XTI

Table A-1 describes the mapping of OSI primitives to XTI system calls.

Table A-1 Mapping of OSI Primitives to XTI System Calls

OSI Primitives	XTI System Calls
A-ASSOCIATE request	Client: *t_open*( ) + *t_bind*( ) + *t_connect*( ) + *t_rcvconnect*( ) + *t_snd*(AREQ-apdu)
A-ASSOCIATE indication	Server: *t_open*( ) + *t_bind*( ) + *t_listen*( ) + *t_accept*( ) + *t_rcv*(AREQ-apdu)
A-ASSOCIATE response	*t_snd*(ARSP-apdu)
A_ASSOCIATE confirm	*t_rcv*(ARSP-apdu)
P-DATA request	*t_snd*(RDA-apdu)
P-DATA indication	*t_rcv*(RDA-apdu)
A-RELEASE request	*t_close*( )*
A-RELEASE confirm	*t_close*( )
A-ABORT request	*t_close*( )
A-P-ABORT indication	T_DISCONNECT event returned from *t_look*( )
to clean up:	*t_unbind*( )

* The A-RELEASE services are mapped to a single XTI system call (*t_close*( )) as advised in [11] because not all XTI implementations support an 'orderly' release mechanism. For further details, see reference [11].

A.2 Berkeley System Distribution (BSD) Sockets

Table A-2 describes the mapping of OSI primitives to BSD Sockets system calls.

Table A-2 Mapping of OSI Primitives to BSD Sockets System Calls

OSI Primitives	BSD Sockets System Calls
A-ASSOCIATE request	Client: *socket*( ) + *connect*( ) + *send*(AREQ-apdu)
A-ASSOCIATE indication	Server: *socket*( ) + *bind*( ) + *listen*( ) + *accept*( ) + *recv*(AREQ-apdu)
A-ASSOCIATE response	*send*(ARSP-apdu)
A_ASSOCIATE confirm	*recv*(ARSP-apdu)
P-DATA request	*send*(RDA-apdu)
P-DATA indication	*recv*(RDA-apdu)
A-RELEASE request	*close*( )
A-RELEASE confirm	eof on *recv*( ), error on *send*( )
A-ABORT request	*shutdown*( )
A-P-ABORT indication	eof on *recv*( ), error on *send*( )

Index

A-ABORT 3, 22
  parameters 13
A-ABORT indication 29-30
A-ABORT request 29-30
A-ASSOCIATE 3
  parameters 9
A-ASSOCIATE confirm 29-30
A-ASSOCIATE indication 29-30
A-ASSOCIATE request 29-30
A-ASSOCIATE response 29-30
A-P-ABORT 22, 25
  parameters 14
A-RELEASE 3
  parameters 12
A-RELEASE confirm 29-30
A-RELEASE request 29-30
abnormal release of association 13
Abort Source 13
abstract syntax name 7
  ACSE 7
  RDA 7
accept( ) 30
acceptability result 11
acceptance is mandatory 5
ACSE 1, 3
  presentation context 7
Active/Passive 19
address, network 6
AE 3
affirmative result 11
AP 3
APDU 3
  boundary 26
  delimiting 26
  encoding 18
  invalid 24
  length octet 26
  mapper 18
  reception 26
  tag octet 26
  transmission 26
  unknown 24
Application Context Name 4
Application Layer 18
application-context 4
  ISO RDA 4, 7
  X/Open RDA 4, 7
application-context-name 18
application-entity 3
application-process 3
application-protocol-data-unit 3
AREQ APDU 20, 22
AREQ-apdu 29-30
ARSP APDU 22-23
ARSP-apdu 29-30
ASN.1 for RDATCP APDU 18
A-ABORT 13
  request procedure 24
A-ASSOCIATE 4
  acceptor 5
  confirm procedure 22
  indication procedure 20
  request procedure 19
  response procedure 22
A-P-ABORT 11, 14, 19
  indication procedure 24
A-RELEASE 11
  confirm procedure 24
  refusal 11
  request procedure 23
basic application-context 4
basic encoding rules 18
bind( ) 30
BSD sockets 30
Called AE Invocation-identifier 5
Called AE Qualifier 5
Called AP Invocation-identifier 5
Called AP Title 5
Called Presentation Address 6
called-AP-invocation-identifier 18
called-AP-qualifier 18
called-AP-title 18
Calling AE Invocation-identifier 4
Calling AE Qualifier 4

Calling AP Invocation-identifier 4
Calling AP Title.....4
Calling Presentation Address.....6
calling-AE-invocation-identifier.....18
calling-AE-qualifier.....18
calling-AP-invocation-identifier.....18
calling-AP-title 18
capability of mapper 3
close( ) 30
confirm primitive 5
connect( ) 30
connection orderly release.....17
creation of RDA service element 1
Default Presentation Context Name.....8
Default Presentation Context Result 8
definite length encoding.....18
delivery not guaranteed 11
Diagnostic 5
embedded values 15
encoding 18
event-driven state machine.....17
finite state machine 17
Foreign socket.....19
guaranteed delivery 11
highest common protocol version 20
implementation-information.....18
inability to connect.....24
information in transit 11
Initial Assignment of Tokens 9
Initial Synchronization Point Serial Number.....9
interface model.....1
IP address.....6
keep-alive.....17
kernel.....8
listen( ) 30
listening on TCP connection 27
Local port 19
mapper.....1
  limitation on user.....3
  services 3
mapper APDU 18
mapping 1
mapping onto TCP.....17
mapping procedure.....19
Mode.....4
model
  benefits.....1
  overview **1**
name
  abstract syntax 7
  transfer syntax.....7
negotiation.....18
network address.....6
object identifier 4
  abstract syntax 7
  transfer syntax.....7-8
option, TCP service 17
Options.....19
orderly release.....17
orderly termination.....11
OSI
  easy adaptation 1
  existing interface.....1
outermost SET or SEQUENCE.....18
P-DATA 3
  parameters 15
P-DATA indication.....29-30
P-DATA request.....29-30
peer entity 11
port number.....6
Precedence 19
presentation context
  ACSE 7-8
  RDA.....7-8, 15
Presentation Context Definition List 7
Presentation Context Definition Result List 8
presentation data value 15
presentation functional unit.....8
  kernel 8
Presentation Layer.....1, 3, 15
  mapping overview.....1
  selector.....6
  subset.....1
Presentation Layer parameter 4
Presentation Requirements 8
protocol events, handling.....17
protocol version.....18
  how chosen 20
protocol-version.....18
Provider Reason.....14
PUSH option 17
P-DATA 15
  indication procedure.....25
  request procedure.....25
Quality of Service 8
RDA
  for TCP/IP overview.....**1**
RDA APDU.....18, 25
RDA Basic application-context.....3-4
RDA presentation context.....7
RDA TP application-context.....4
RDA-apdu 26, 29-30

RDA-TCP mapper 1, 3
RDATCP 18
RDATCP APDU.....18
RDATCP-apdu.....26
Reason.....11
receiving APDU.....26
recv( ).....30
release of association, abnormal 13
Responding AE Invocation-identifier 5
Responding AE Qualifier 5
Responding AP Invocation-identifier 5
Responding AP Title.....5
Responding Presentation Address.....6
response primitive 5
Result.....5, 11
Result Source.....5
RFC 1006.....1
Security/compartment.....19
selector 6
semantic definition of services.....3
send( ).....30
service element creation.....1
service option, TCP 17
service provided by mapper.....3
service-provider.....17
service-user.....5, 17
services, mapping 1
session functional unit 8
  duplex.....8
  kernel 8
Session Layer
  selector.....6
Session Layer parameter.....4
Session Requirements 8
Session-connection Identifier.....9
shutdown( ).....30
socket( ) 30
state machine, finite 17
TCP
  mapping.....17
  stream-oriented nature.....26
TCP CLOSE.....25
TCP CLOSE 17, 20, 23-24
TCP connection error 24
TCP connection listening.....27
TCP OPEN.....19, 26
  Active.....17, 19, 24
  Passive 17, 20-21, 27
TCP OPEN argument
  Active/Passive 19
  Foreign socket 19
  Local port.....19
  Options.....19
  Precedence.....19
  Security/compartment 19
  Timeout 19
TCP RECEIVE.....17, 20, 22, 24, 26
TCP SEND 17, 20, 22, 24-26
TCP service option 17
TCP/IP 1
terminate association.....11
Timeout.....19
transfer syntax name.....7
  ACSE 7-8
  RDA 7-8
Transport Layer
  selector.....6
t_accept( ) 29
t_bind( ) 29
t_close( ).....29
t_connect( ) 29
t_listen( ).....29
t_look( ) 29
t_open( ).....29
t_rcv( ).....29
t_rcvconnect( ).....29
t_snd( ).....29
t_snddis( ).....29
t_unbind( ) 29
URGENT option 17
User data.....15, 25
User Information.....5, 11, 13
user information.....4, 11, 13
User/TCP Interface model 17
version
  how chosen 20
  of protocol.....18
XTI.....29