INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Q.773

(06/97)

SERIES Q: SWITCHING AND SIGNALLING

Specifications of Signalling System No. 7 – Transaction capabilities application part

Transaction capabilities formats and encoding

ITU-T Recommendation Q.773

(Previously CCITT Recommendation)

ITU-T Q-SERIES RECOMMENDATIONS

SWITCHING AND SIGNALLING

SIGNALLING IN THE INTERNATIONAL MANUAL SERVICE	Q.1–Q.3
INTERNATIONAL AUTOMATIC AND SEMI-AUTOMATIC WORKING	Q.4–Q.59
FUNCTIONS AND INFORMATION FLOWS FOR SERVICES IN THE ISDN	Q.60–Q.99
CLAUSES APPLICABLE TO ITU-T STANDARD SYSTEMS	Q.100–Q.119
SPECIFICATIONS OF SIGNALLING SYSTEMS No. 4 AND No. 5	Q.120–Q.249
SPECIFICATIONS OF SIGNALLING SYSTEM No. 6	Q.250–Q.309
SPECIFICATIONS OF SIGNALLING SYSTEM R1	Q.310–Q.399
SPECIFICATIONS OF SIGNALLING SYSTEM R2	Q.400–Q.499
DIGITAL EXCHANGES	Q.500–Q.599
INTERWORKING OF SIGNALLING SYSTEMS	Q.600–Q.699
SPECIFICATIONS OF SIGNALLING SYSTEM No. 7	Q.700–Q.849
General	Q.700
Message transfer part (MTP)	Q.701–Q.709
Signalling connection control part (SCCP)	Q.711–Q.719
Telephone user part (TUP)	Q.720–Q.729
ISDN supplementary services	Q.730–Q.739
Data user part	Q.740–Q.749
Signalling System No. 7 management	Q.750–Q.759
ISDN user part	Q.760–Q.769
Transaction capabilities application part	**Q.770–Q.779**
Test specification	Q.780–Q.799
Q3 interface	Q.800–Q.849
DIGITAL SUBSCRIBER SIGNALLING SYSTEM No. 1	Q.850–Q.999
General	Q.850–Q.919
Data link layer	Q.920–Q.929
Network layer	Q.930–Q.939
User-network management	Q.940–Q.949
Stage 3 description for supplementary services using DSS 1	Q.950–Q.999
PUBLIC LAND MOBILE NETWORK	Q.1000–Q.1099
INTERWORKING WITH SATELLITE MOBILE SYSTEMS	Q.1100–Q.1199
INTELLIGENT NETWORK	Q.1200–Q.1999
BROADBAND ISDN	Q.2000–Q.2999

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

ITU-T RECOMMENDATION Q.773

TRANSACTION CAPABILITIES FORMATS AND ENCODING

Summary

This Recommendation has been revised for some corrections to the ASN.1 and tabular descriptions of TC messages and parameters. It also provides, in Annex A, a description of the TC using the ASN.1 notation defined in Recommendations Q.680 to Q.683 and some definitions from Recommendation X.880.

Source

ITU-T Recommendation Q.773 was revised by ITU-T Study Group 11 (1997-2000) and was approved under the WTSC Resolution No. 1 procedure on the 5th of June 1997.

FOREWORD

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

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

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

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

NOTE

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

INTELLECTUAL PROPERTY RIGHTS

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

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

CONTENTS

Recommendation Q.773

TRANSACTION CAPABILITIES FORMATS AND ENCODING

(Melbourne 1988; revised in 1993 and 1997)

1 Introduction

This Recommendation provides the format and encoding of Transaction Capabilities (TC) messages. These encoding rules are described in 4.1 and are based on a consistent subset of the encoding rules specified in Recommendation X.209 as contained in the *Blue Book*.

2 Description conventions

This Recommendation uses the Abstract Syntax Notation One (ASN.1) defined in Recommendation X.208 and the description method of other Q.700-Series Recommendations (tabular method). In the case of misalignment between the tabular and the ASN.1 description, the latter takes precedence over the tabular representation.

3 Abstract syntax description

3.1 TC-Messages

The following module defines the type of TC messages:

```
TCAPMessages { ccitt recommendation q 773 modules (2) messages (1) version2 (2) }

DEFINITIONS ::=

BEGIN

EXPORTS          OPERATION, ERROR, Component, InvokeId Type;

-- Transaction Portion fields

MessageType ::=  CHOICE {
                     unidirectional     [APPLICATION 1]  IMPLICIT Unidirectional,
                     begin              [APPLICATION 2]  IMPLICIT Begin,
                     end                [APPLICATION 4]  IMPLICIT End,
                     continue           [APPLICATION 5]  IMPLICIT Continue,
                     abort              [APPLICATION 7]  IMPLICIT Abort }

Unidirectional ::=  SEQUENCE        { dialoguePortion     DialoguePortion OPTIONAL,
                                      components          ComponentPortion }

Begin ::=           SEQUENCE        { otid                OrigTransactionID,
                                      dialoguePortion     DialoguePortion OPTIONAL
                                      components          ComponentPortion OPTIONAL }

End ::=             SEQUENCE        { dtid                DestTransactionID,
                                      dialoguePortion     DialoguePortion OPTIONAL,
                                      components          ComponentPortion OPTIONAL }
```

```
Continue ::=        SEQUENCE        { otid                  OrigTransactionID,
                                      dtid                  DestTransactionID,
                                      dialoguePortion       DialoguePortion OPTIONAL,
                                      components            ComponentPortion OPTIONAL }
Abort ::=           SEQUENCE        {
                                      dtid                  DestTransactionID,
                                    reason                  CHOICE
                                    { p-abortCause          P-AbortCause,
                                    u-abortCause    DialoguePortion } OPTIONAL
                                    }
-- NOTE – When the Abort Message is generated by the Transaction sublayer, a p-Abort Cause must be
-- present.The u-abortCause may be generated by the component sublayer in which case it is an ABRT
-- APDU, or by the TC-User in which case it could be either an ABRT APDU or data in some user-defined
-- abstract syntax.

DialoguePortion ::=     [APPLICATION 11]            EXTERNAL

-- The dialogue portion carries the dialogue control PDUs as value of the external data type.
-- The direct reference should be set to { ccitt recommendation q 773 as (1) dialogue-as (1) version (1) }
-- if structured dialogue is used and to { ccitt recommendation q 773 as (1) unidialogue-as (2) version (1) }
-- if unstructured dialogue is used or any user defined abstract syntax name when only user information
-- is carried (e.g. when user information is sent in a 1988 Abort message).

OrigTransactionID ::=   [APPLICATION 8]     IMPLICIT OCTET STRING (SIZE (1..4) )

DestTransactionID ::=   [APPLICATION 9]     IMPLICIT OCTET STRING (SIZE (1..4) )

P-AbortCause ::=        [APPLICATION 10]    IMPLICIT INTEGER {
                                unrecognizedMessageType (0),
                                unrecognizedTransactionID (1),
                                badlyFormattedTransactionPortion (2),
                                incorrectTransactionPortion (3),
                                resourceLimitation (4) }

-- COMPONENT PORTION. The last field in the transaction portion of the TCAP message is the Component Portion.
-- The Component Portion may be absent.

ComponentPortion ::=    [APPLICATION 12]    IMPLICIT SEQUENCE SIZE (1..MAX) OF Component

-- Component Portion fields

-- COMPONENT TYPE. Recommendation X.229 defines four Application Protocol Data Units (APDUs).
-- TCAP adds returnResultNotLast to allow for the segmentation of a result.

Component ::=       CHOICE {
                        invoke                  [1] IMPLICIT Invoke,
                        returnResultLast        [2] IMPLICIT ReturnResult,
                        returnError             [3] IMPLICIT ReturnError,
                        reject                  [4] IMPLICIT Reject,
                        returnResultNotLast     [7] IMPLICIT ReturnResult }

-- The Components are sequences of data elements.

Invoke ::=          SEQUENCE {
                        invokeID                InvokeIdType,
                        linkedID                [0] IMPLICIT InvokeIdType OPTIONAL,
                        operationCode           OPERATION,
                        parameter               ANY DEFINED BY operationCode OPTIONAL }
```

```
-- ANY is filled by the single ASN.1 data type following the keyword PARAMETER or the keyword ARGUMENT
-- in the type definition of a particular operation.

ReturnResult ::=    SEQUENCE {
                        invokeID                    InvokeIdType,
                        result                      SEQUENCE {
                            operationCode           OPERATION,
                            parameter               ANY DEFINED BY operationCode
                        } OPTIONAL
                    }

-- ANY is filled by the single ASN.1 data type following the keyword RESULT in the type definition
-- of a particular operation.

ReturnError ::=     SEQUENCE {
                        invokeID                    InvokeIdType,
                        errorCode                   ERROR,
                        parameter                   ANY DEFINED BY errorCode OPTIONAL }

-- ANY is filled by the single ASN.1 data type following the keyword PARAMETER in the type definition
-- of a particular error.

Reject ::=          SEQUENCE {
                        invokeID CHOICE {
                            derivable               InvokeIdType,
                            not-derivable           NULL },
                        problem CHOICE {
                            generalProblem          [0] IMPLICIT GeneralProblem,
                            invokeProblem           [1] IMPLICIT InvokeProblem,
                            returnResultProblem     [2] IMPLICIT ReturnResultProblem,
                            returnErrorProblem      [3] IMPLICIT ReturnErrorProblem } }

InvokeIdType ::=    INTEGER (–128..127)

-- OPERATIONS

-- Operations are specified with the OPERATION MACRO.
-- When an operation is specified, the valid parameter set, results, and errors for that operation are indicated.
-- Default values and optional parameters are permitted.

OPERATION MACRO     ::=

BEGIN
    TYPE NOTATION       ::=  Parameter Result Errors LinkedOperations
    VALUE NOTATION      ::=  value (VALUE CHOICE {
                                 localValue INTEGER,
                                 globalValue OBJECT IDENTIFIER } )
    Parameter           ::=  ArgKeyword NamedType | empty
    ArgKeyword          ::=  "ARGUMENT" | "PARAMETER"
    Result              ::=  "RESULT" ResultType | empty
    Errors              ::=  "ERRORS" "{"ErrorNames"}" | empty
    LinkedOperations    ::=  "LINKED" "{"LinkedOperationNames"}" | empty
    ResultType          ::=  NamedType | empty
    ErrorNames          ::=  ErrorList | empty
    ErrorList           ::=  Error | ErrorList "," Error
    Error               ::=  value (ERROR)
                                 -- shall reference an error value
                             | type  -- shall reference an error type
                                 -- if no error value is specified
```

```
        LinkedOperationNames  ::=  OperationList | empty
        OperationList         ::=  Operation | OperationList "," Operation
        Operation             ::=  value (OPERATION)
                                       -- shall reference an operation value
                                   | type  -- shall reference an operation type if
                                       -- no operation value is specified
        NamedType             ::=  identifier type | type
END

-- ERRORS

-- Errors are specified with the ERROR MACRO.
-- When an error is specified, the valid parameters for that error are indicated.
-- Default values and optional parameters are permitted.

ERROR MACRO           ::=

BEGIN
        TYPE NOTATION     ::=  Parameter
        VALUE NOTATION    ::=  value (VALUE CHOICE {
                                       localValue INTEGER,
                                       globalValue OBJECT IDENTIFIER } )
        Parameter  ::=  "PARAMETER" NamedType | empty
        NamedType  ::=  identifier type | type
END

-- PROBLEMS

GeneralProblem        ::=  INTEGER {    unrecognizedComponent (0),
                                        mistypedComponent (1),
                                        badlyStructuredComponent (2) }

InvokeProblem         ::=  INTEGER {    duplicateInvokeID (0),
                                        unrecognizedOperation (1),
                                        mistypedParameter (2),
                                        resourceLimitation (3),
                                        initiatingRelease (4),
                                        unrecognizedLinkedID (5),
                                        linkedResponseUnexpected (6),
                                        unexpectedLinkedOperation (7) }

ReturnResultProblem   ::=  INTEGER {    unrecognizedInvokeID (0),
                                        returnResultUnexpected (1),
                                        mistypedParameter (2) }

ReturnErrorProblem    ::=  INTEGER {    unrecognizedInvokeID (0),
                                        returnErrorUnexpected (1),
                                        unrecognizedError (2),
                                        unexpectedError (3),
                                        mistypedParameter (4) }

END -- TCAPMessages
```

3.2 Dialogue portion

3.2.1 Structured dialogue

The following module defines the type DialoguePDU whose values form the abstract syntax for the Dialogue APDUs used for the structured dialogue.

```
DialoguePDUs { ccitt recommendation q 773 modules (2) dialoguePDUs(2) version1 (1) }

DEFINITIONS ::=

BEGIN

EXPORTS          dialogue-as-id, DialoguePDU;

-- abstract syntax name for structured dialogue APDUs

dialogue-as-id OBJECT IDENTIFIER      ::=    { ccitt recommendation q 773 as (1)
                                               dialogue-as (1) version1 (1) }

DialoguePDU ::= CHOICE {
                                               dialogueRequest      AARQ-apdu,
                                               dialogueResponse     AARE-apdu,
                                               dialogueAbort        ABRT-apdu }

AARQ-apdu ::= [APPLICATION 0]          IMPLICIT SEQUENCE {
      protocol-version                 [0] IMPLICIT BIT STRING { version1 (0) }
                                             DEFAULT { version1 },
      application-context-name         [1] OBJECT IDENTIFIER,
      user-information                 [30] IMPLICIT SEQUENCE OF EXTERNAL
                                             OPTIONAL }

AARE-apdu ::= [APPLICATION 1]          IMPLICIT SEQUENCE {
      protocol-version                 [0] IMPLICIT BIT STRING { version1 (0) }
                                             DEFAULT { version1 },
      application-context-name         [1] OBJECT IDENTIFIER,
      result                           [2] Associate-result,
      result-source-diagnostic         [3] Associate-source-diagnostic,
      user-information                 [30] IMPLICIT SEQUENCE OF EXTERNAL
                                             OPTIONAL }

-- RLRQ PDU is currently not used.
-- It is included for completeness only.

RLRQ-apdu ::= [APPLICATION 2]          IMPLICIT SEQUENCE {
      reason                           [0] IMPLICIT Release-request-reason OPTIONAL,
      user-information                 [30] IMPLICIT SEQUENCE OF EXTERNAL
                                             OPTIONAL }

-- RLRE PDU is currently not used.
-- It is included for completeness only

RLRE-apdu ::= [APPLICATION 3]          IMPLICIT SEQUENCE {
      reason                           [0] IMPLICIT Release-response-reason OPTIONAL,
      user-information                 [30] IMPLICIT SEQUENCE OF EXTERNAL
                                             OPTIONAL }

ABRT-apdu ::= [APPLICATION 4]          IMPLICIT SEQUENCE {
      abort-source                     [0] IMPLICIT ABRT-source,
      user-information                 [30] IMPLICIT SEQUENCE OF EXTERNAL
                                             OPTIONAL }

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

Associate-result ::= INTEGER {         accepted (0),
                                       reject-permanent (1) }
```

```
Associate-source-diagnostic ::= CHOICE {
        dialogue-service-user       [1]    INTEGER {
                                           null (0),
                                           no-reason-given (1),
                                           application-context-name-not-supported (2) },
        dialogue-service-provider   [2]    INTEGER {
                                           null (0),
                                           no-reason-given (1),
                                           no-common-dialogue-portion (2) }
                                           }

-- Release-request-reason is currently not used.
-- It is included for completeness only.

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

-- Release-response-reason is currently not used.
-- It is included for completeness only.

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

3.2.2 Unstructured dialogue

The following module defines the type UnidialoguePDU whose values form the abstract syntax for the dialogue APDUs used for the unstructured dialogue.

```
UnidialoguePDUs { ccitt recommendation q 773 modules (2) unidialoguePDUs (3) version1 (1) }

DEFINITIONS ::=

BEGIN

EXPORTS                                  uniDialogue-as-id, UniDialoguePDU;

-- Abstract syntax name for unstructured dialogue APDUs

uniDialogue-as-id OBJECT IDENTIFIER      ::=    { ccitt recommendation q 773 as (1)
                                                  unidialogue-as (2) version1 (1) }

UniDialoguePDU ::= CHOICE { unidialoguePDU                    AUDT-apdu }

AUDT-apdu ::= [APPLICATION 0]            IMPLICIT SEQUENCE {
        protocol-version                 [0] IMPLICIT BIT STRING {version1 (0) }
                                                DEFAULT { version1 },
        application-context-name         [1] OBJECT IDENTIFIER,
        user-information                 [30] IMPLICIT SEQUENCE OF EXTERNAL
                                                OPTIONAL }

END -- UNIDialoguePDU
```

4 Message representation

A TCAP message is structured as a single constructor information element. It consists of a Transaction Portion which contains information elements used by the Transaction sublayer, and a Component Portion which contains information elements used by the Component sublayer related to components and, optionally, the Dialogue Portion which contains the Application Context and user information, which are not components. One of the Transaction Portion elements is called the Component Portion, and it contains the Component sublayer information elements. Each Component is a constructor information element.

4.1 Encoding rules

4.1.1 Specification of encoding rules

The encoding rules for TC messages are those described in Recommendation X.209 with the following restrictions:

– when the definite form is used for length encoding, a data value of length less than 128 octets must have the length encoded in the short form;

– when the long form is employed to code a length, the minimum number of octets shall be used to code the length field;

– OCTET STRING values and BIT STRING values must be encoded in a primitive form.

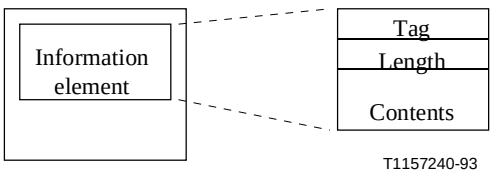
These encoding rules apply to TC messages as well as to the other abstract syntaxes defined in this Recommendation.

4.1.2 Overview of encoding rules

The following subclauses summarize the principles of these encoding rules. In case of misalignment between these subclauses and 4.1.1, the latter takes precedence.

4.1.2.1 General message structure

Each information element within a TCAP message has the same structure. An information element consists of three fields, which always appear in the following order. The Tag distinguishes one type from another and governs the interpretation of the Contents. The Length specifies the length of the Contents. The Contents is the substance of the element, containing the primary information the element is intended to convey. Figure 1 shows an overview of a TCAP message and an information element.



a) TCAP message **b) Information element**

Figure 1/Q.773 – Structure of TCAP message and information element

Each field is coded using one or more octets. Octets are labelled as shown in Figure 2. Bits in an octet are labelled as shown in Figure 3, with bit A the least significant.

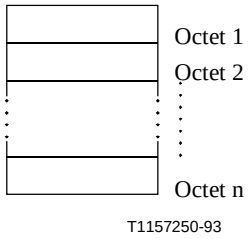


Figure 2/Q.773 – Octet labelling scheme

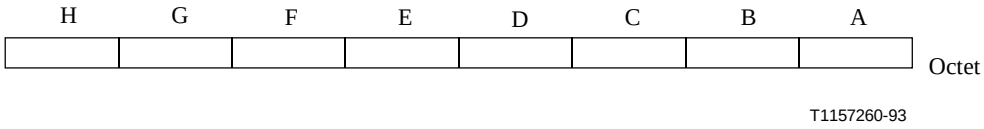
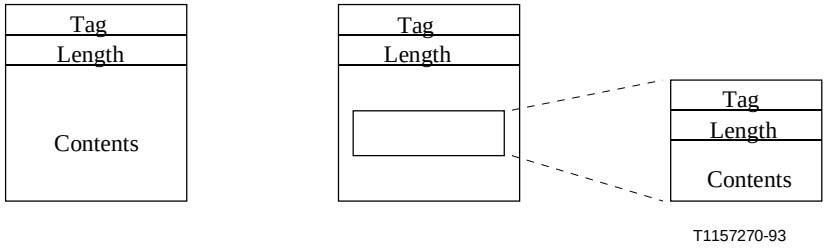


Figure 3/Q.773 – Bit labelling scheme

The contents of each element is either one value (Primitive) or one or more information elements (Constructor), as shown in Figure 4.

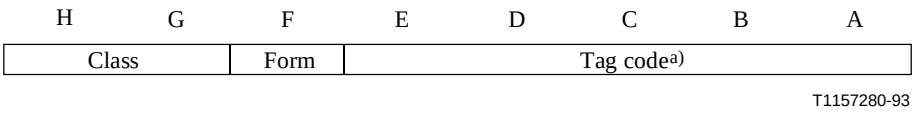


a) Primitive **b) Constructor**

Figure 4/Q.773 – Types of contents

4.1.2.2 Tag

An information element is first interpreted according to its position within the syntax of the message. The Tag distinguishes one information element from another and governs the interpretation of the Contents. It is one or more octets in length. The Tag is composed of "Class", "Form" and "Tag code", as shown in Figure 5.



a) The Tag code may be extended to the following octet(s) as discussed in 4.1.2.2.3.

Figure 5/Q.773 – Format of Tag

4.1.2.2.1 Tag class

All Tags use the two most significant bits (H and G) to indicate the Tag Class. These bits are coded as shown in Table 1.

Table 1/Q.773 – Coding of Tag class

Class	Coding (HG)
Universal	00
Application-wide	01
Context-specific	10
Private use	11

The universal class is used for Tags that are exclusively standardized in Recommendation X.208 and are application-independent types. Universal Tags may be used anywhere a universal information element type is used. The universal class applies across all CCITT Recommendations, i.e. across CCITT No. 7 ASEs, X.400 MHS, etc.

The Application-wide class is used for information elements that are standardized across all applications (ASEs) using Signalling System No. 7 TC, i.e. TC-Users.

The Context-specific class is used for information elements that are specified within the context of the next higher construction and take into account the sequence of other data elements within the same construction. This class may be used for tags in a construction, and the tags may be re-used in any other construction.

The Private Use class is reserved for information elements specific to a nation, a network or a private user. Such information elements are beyond the scope of the TC Recommendations.

The Tag codes of the Application-wide class not assigned in this Recommendation are reserved for future use.

4.1.2.2.2 Form of the element

Bit F is used to indicate whether the element is "Primitive" or "Constructor", as shown in Table 2. A primitive element is one whose structure is atomic (i.e. one value only). A constructor element is one whose content is one or more information elements which may themselves be constructor elements.

Both forms of elements are shown in Figure 4.

Table 2/Q.773 – Coding of element form

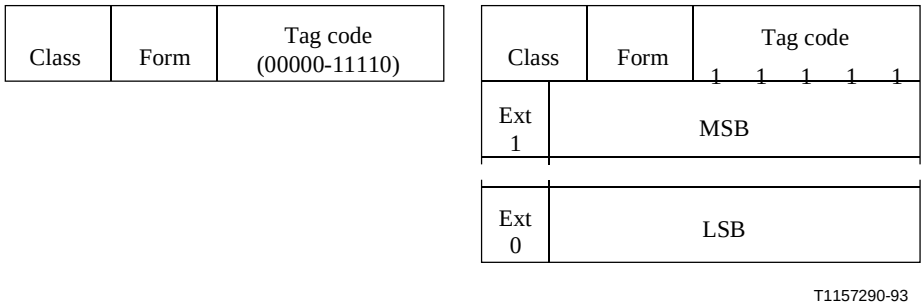
Element form	Coding (F)
Primitive	0
Constructor	1

4.1.2.2.3 Tag code

Bits A to E of the first octet of the Tag plus any extension octets represent a Tag code that distinguishes one element type from another of the same class. Tag codes in the range 00000 to 11110 (0 to 30 decimal) are provided in one octet.

The extension mechanism is to code bits A to E of the first octet as 11111. Bit H of the following octet serves as an extension indication. If bit H of the extension octet is set to 0, then no further octets for this tag are used. If bit H is set to 1, the following octet is also used for extension of the Tag code. The resultant Tag consists of bits A to G of each extension octet, with bit G of the first extension octet being most significant and bit A of the last extension octet being least significant. Tag code 31 is encoded as 0011111 in bits G to A of a single extension octet. Higher tag codes continue from this point using the minimum possible number of extension octets.

Figure 6 shows the detailed format of the Tag code.



a) One octet format **b) Extended format**

Figure 6/Q.773 – Format of the Tag code

4.1.2.3 Length of the Contents

The Length of the Contents is coded to indicate the number of octets in the Contents. The length does not include the Tag nor the Length of the Contents octets.

The Length of the Contents uses the short, long or indefinite form. If the length is less than 128 octets, the short form is used. In the short form, bit H is coded 0, and the length is encoded as a binary number using bits A to G.

If the Length of the contents is greater than 127 octets, then the long form of the Length of the Contents is used. The long form Length is from 2 to 127 octets long. Bit H of the first octet is coded 1, and bits A to G of the first octet encode a number one less than the size of the Length in octets as an unsigned binary number whose MSB and LSB are bits G and A, respectively. The length itself is encoded as an unsigned binary number whose MSB and LSB are bit H of the second octet and bit A of the last octet, respectively. This binary number should be encoded in the fewest possible octets, with no leading octets having the value 0.

The indefinite form is one octet long and may (but need not) be used in place of the short or long form, whenever the element is a constructor. It has the value 10000000. When this form is employed, a special End-of-Contents (EOC) indicator terminates the Contents.

There is no notation for the end-of-contents indicator. Although considered part of the Contents syntactically, the end-of-contents indicator has no semantic significance.

The representation for the end-of-contents indicator is an element whose class is universal, whose form is primitive, whose ID Code has the value 0, and whose Contents is unused and absent:

EOC	Length	Contents
00(hex)	00(hex)	Absent

Figure 7 shows the formats of the Length field described above. The maximum value that may be encoded is constrained by the network message size limitations in the connectionless case. Limitations in the connection-oriented case are for further study.

0	Length of Contents MSB LSB

a) Short form

1	(Length of Field Size) – 1 MSB LSB
MSB	Length of Contents LSB

b) Long form

Constructor Element Tag
L = 10000000
Tag Length (Note) Contents
: :
Tag Length (Note) Contents
EOC Tag = 00000000
EOC Length = 00000000

c) Indefinite form

NOTE – The Length may take any of three forms: short, long, and indefinite.

Figure 7/Q.773 – Format of length field

4.1.2.4 Contents

The contents is the substance of the element and contains the information the element is intended to convey. Its length is variable, but always an integral number of octets. The contents is interpreted in a type-dependent manner, i.e. according to the tag value.

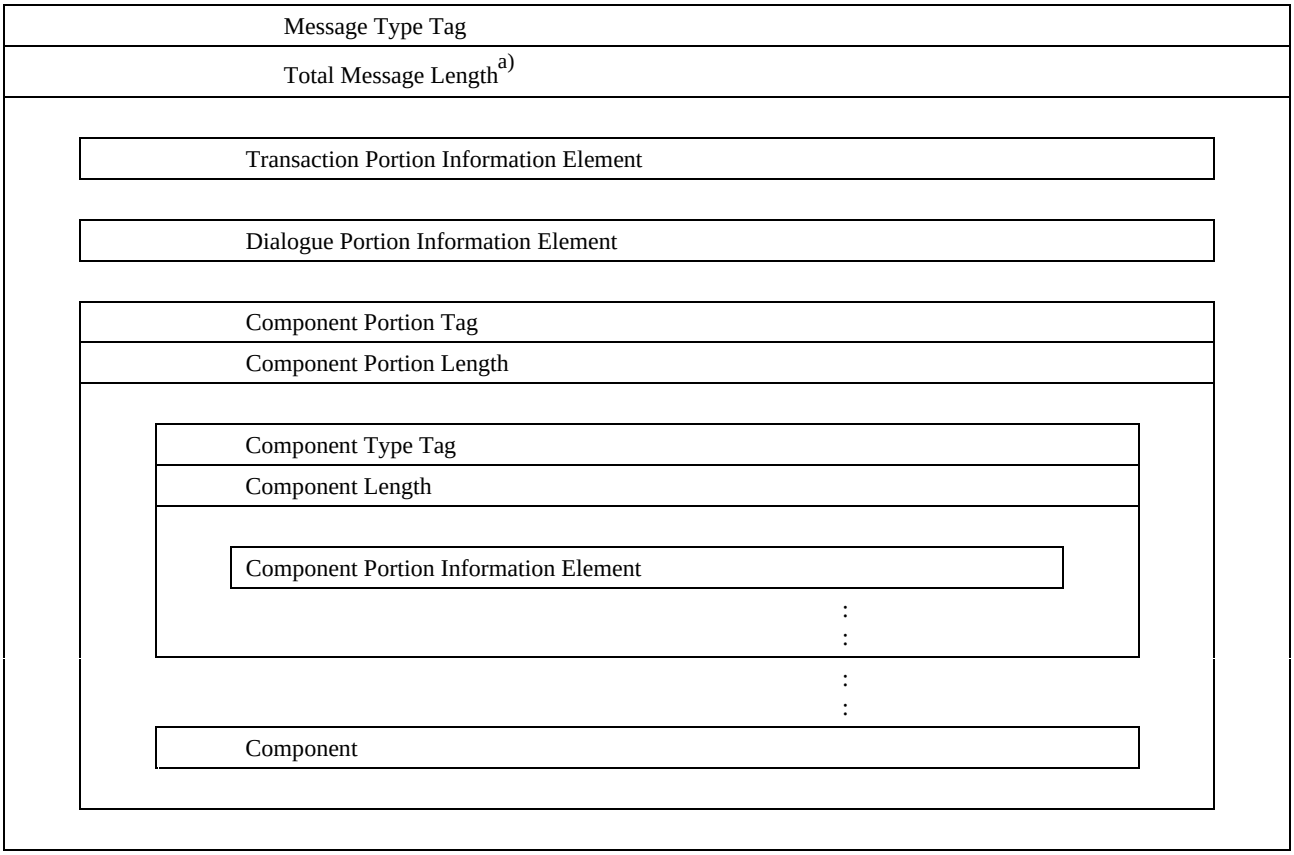
4.1.3 Transmission order

The following transmission order is assumed:

i) the first octet of a string of octets which result from the encoding of a TCAP message is first transmitted;

ii) the least significant bit of each octet is transmitted first.

Figure 8 shows the detailed TCAP message structure.



a) The user should be aware of total message length limitations when using TCAP in the SS No. 7 connectionless environment.

Figure 8/Q.773 – Detailed TCAP message structure

4.2 Message encoding

4.2.1 Transaction Portion

Transaction Portion information elements use the Application-wide class as defined in 4.1.2.2.1.

4.2.1.1 Structure of the Transaction Portion

The Transaction Portion fields for various message types are shown in Tables 3 to 7.

Table 3/Q.773 – Transaction Portion fields
Unidirectional message type

Element form	Fields of Transaction Portion	Mandatory Indication
Constructor	Message Type Tag Total message Length[a)]	Mandatory
Constructor	Dialogue Portion	Optional
Constructor	Component Portion Tag Component Portion Length	Mandatory
Constructor	One or more Components (Not a part of Transaction Portion) (Described in 4.2.2)	Mandatory

a) The user should be aware of total message length limitations when using TCAP in the SS No. 7 connectionless environment.

Table 4/Q.773 – Transaction Portion fields
Begin message type

Element form	Fields of Transaction Portion	Mandatory Indication
Constructor	Message Type Tag Total message Length[a)]	Mandatory
Primitive	Originating Transaction ID Tag Transaction ID Length Transaction ID	Mandatory
Constructor	Dialogue Portion	Optional
Constructor	Component Portion Tag Component Portion Length	Optional[b)]
Constructor	One or more Components (Not a part of Transaction Portion) (Described in 4.2.2)	Optional

a) The user should be aware of total message length limitations when using TCAP in the SS No. 7 connectionless environment.

b) The Component Portion Tag shall be present only if there are Components in the message.

Table 5/Q.773 – Transaction Portion fields
End message type

Element form	Fields of Transaction Portion	Mandatory Indication
Constructor	Message Type Tag Total message Length[a)]	Mandatory
Primitive	Destination Transaction ID Tag Transaction ID Length Transaction ID	Mandatory
Constructor	Dialogue Portion	Optional
Constructor	Component Portion Tag Component Portion Length	Optional[b)]
Constructor	One or more Components (Not a part of Transaction Portion) (Described in 4.2.2)	Optional

a) The user should be aware of total message length limitations when using TCAP in the SS No. 7 connectionless environment.

b) The Component Portion Tag shall be present only if there are Components in the message.

Table 6/Q.773 – Transaction Portion fields
Continue message type

Element form	Fields of Transaction Portion	Mandatory Indication
Constructor	Message Type Tag Total message Length[a)]	Mandatory
Primitive	Originating Transaction ID Tag Transaction ID Length Transaction ID	Mandatory
Primitive	Destination Transaction ID Tag Transaction ID Length Transaction ID	Mandatory
Constructor	Dialogue Portion	Optional
Constructor	Component Portion Tag Component Portion Length	Optional[b)]
Constructor	One or more Components (Not a part of Transaction Portion) (Described in 4.2.2)	Optional

a) The user should be aware of total message length limitations when using TCAP in the SS No. 7 connectionless environment.

b) The Component Portion Tag shall be present only if there are Components in the message.

Table 7/Q.773 – Transaction Portion fields
Abort message type

Element form	Fields of Transaction Portion	Mandatory Indication
Constructor	Message Type Tag Total message Length[a)]	Mandatory
Primitive	Destination Transaction ID Tag Transaction ID Length Transaction ID	Mandatory
Primitive	P-Abort Cause Tag P-Abort Cause Length P-Abort Cause	Optional[b)]
Constructor	Dialogue Portion	Optional[c)]

a) The user should be aware of total message length limitations when using TCAP in the SS No. 7 connectionless environment.

b) P-Abort Cause shall be present when the Abort is generated by the transaction sublayer.

c) The Dialogue Portion is optional, and may only be present when the Abort is generated by the TC-User.

4.2.1.2 Message Type Tag

This field consists of one octet and is mandatory for all TCAP messages. Message Type tags are coded as shown in Table 8.

Table 8/Q.773 – Coding of message type Tag

Message type	H	G	F	E	D	C	B	A
Unidirectional	0	1	1	0	0	0	0	1
Begin	0	1	1	0	0	0	1	0
(reserved)	0	1	1	0	0	0	1	1
End	0	1	1	0	0	1	0	0
Continue	0	1	1	0	0	1	0	1
(reserved)	0	1	1	0	0	1	1	0
Abort	0	1	1	0	0	1	1	1

4.2.1.3 Transaction ID tags

Two types of Transaction IDs, i.e. Originating Transaction ID and Destination Transaction ID, may be used. Zero, one or two ID information elements are required depending upon the Message type used. Table 9 depicts this relationship.

Table 9/Q.773 – Transaction ID(s) in each message type

Message type	Originating ID	Destination ID
Unidirectional	No	No
Begin	Yes	No
End	No	Yes
Continue	Yes	Yes
Abort	No	Yes

The Originating and Destination Transaction ID Tags are coded as shown in Table 10.

Table 10/Q.773 – Coding of Transaction ID Tags

	H	G	F	E	D	C	B	A
Originating Transaction ID Tag	0	1	0	0	1	0	0	0
Destination Transaction ID Tag	0	1	0	0	1	0	0	1

The length of a Transaction ID is 1 to 4 octets.

4.2.1.4 P-Abort Cause tag

The P-Abort Cause tag is coded as shown in Table 11.

Table 11/Q.773 – Coding of P-Abort Cause Tag

	H	G	F	E	D	C	B	A
P-Abort Cause Tag	0	1	0	0	1	0	1	0

The P-Abort cause values are coded as shown in Table 12.

Table 12/Q.773 – Coding of P-Abort Cause values

P-Abort Cause	H	G	F	E	D	C	B	A
Unrecognized Message Type	0	0	0	0	0	0	0	0
Unrecognized Transaction ID	0	0	0	0	0	0	0	1
Badly Formatted Transaction portion	0	0	0	0	0	0	1	0
Incorrect Transaction Portion	0	0	0	0	0	0	1	1
Resource Limitation	0	0	0	0	0	1	0	0

4.2.1.5 Dialogue Portion tag

The Dialogue Portion Tag is coded as shown in Table 13.

Table 13/Q.773 – Coding of Dialogue Portion Tag

	H	G	F	E	D	C	B	A
Dialogue Portion Tag[a)]	0	1	1	0	1	0	1	1

[a)] The presence of this tag indicates the presence of the dialogue APDUs, which are described in 4.2.3.

4.2.1.6 Component Portion tag

The Component Portion Tag is coded as shown in Table 14.

Table 14/Q.773 – Coding of Component Portion Tag

	H	G	F	E	D	C	B	A
Component Portion Tag	0	1	1	0	1	1	0	0

4.2.2 Component Portion

The Component Portion, when present, consists of one or more Components. The Components are based on, and extended from, the Remote Operations Service Element (ROSE) Application Protocol Data Units (APDUs) of Recommendation X.229 as indicated in clause 3/Q.772.

4.2.2.1 Component type tag

Each Component is a sequence of information elements. The Component types, as defined for TCAP, have the structure indicated in Tables 15 to 18.

Table 15/Q.773 – Invoke component

Element form	Invoke component	Mandatory Indication
Constructor	Component Type Tag Component Length	M
Primitive	Invoke ID Tag Invoke ID Length Invoke ID	M
Primitive	Linked ID Tag Linked ID Length Linked ID	O
Primitive	Operation Code Tag Operation Code Length Operation Code	M
Primitive/ constructor	Parameter Tag Parameter Length Parameters	O

Table 16/Q.773 – Return Result (Last) and Return Result (Not Last) components

Element form	Return Result (Last) and Return Result (Not Last) components	Mandatory Indication
Constructor	Component Type Tag[a)] Component Length	M
Primitive	Invoke ID Tag Invoke ID Length Invoke ID	M
Constructor	Sequence Tag Sequence Length	O[b)]
Primitive	Operation Code Tag Operation Code Length Operation Code	O[b)]
Primitive/ constructor	Parameter Tag Parameter Length Parameters	O[b)]

a) ROSE has only one APDU called Return Result which is the same as the TCAP Return Result (Last). See 3.1/Q.772.

b) Omitted when no information elements are included in the parameters. The Operation information element is required in order to resolve an ambiguity in the Presentation Layer (or any encoding/decoding process) usage of the ASN.1 representation for the "ANY DEFINED BY" in the ROSE APDU, hence it is necessary to include this in the Return Result when any result parameter is returned. This serves as a reference point from which to examine the bit stream for representation transformation, if required.

Table 17/Q.773 – Return Error Component

Element form	Return Error Component	Mandatory Indication
Constructor	Component Type Tag Component Length	M
Primitive	Invoke ID Tag Invoke ID Length Invoke ID	M
Primitive	Error Code Tag Error Code Length Error Code	M
Primitive/ constructor	Parameter Tag Parameter Length Parameters	O

Table 18/Q.773 – Reject component

Element form	Reject component	Mandatory Indication
Constructor	Component Type Tag Component Length	M
Primitive	Invoke ID Tag[a)] Invoke ID Length Invoke ID	M
Primitive	Problem Code Tag Problem Code Length Problem Code	M
[a)] If the Invoke ID is not available, Universal Null (see Table 21) with length = 0 should be used.		

The parameter tag should be described by the ASN.1 data type that follows the keywords ARGUMENT/PARAMETER or RESULT, as appropriate, in the corresponding OPERATION or ERROR MACRO definition referred to, respectively, by the operation or error code.

Subclause 4.2.2.4 and Table 23 define the Sequence and Set tags.

The Component Type Tag is coded context-specific, constructor as indicated in Table 19.

The format of a Return Result (Not Last) is identical to that of a Return Result (Last).

Table 19/Q.773 – Component Type Tag

Component Type Tag	H	G	F	E	D	C	B	A
Invoke	1	0	1	0	0	0	0	1
Return Result (Last)	1	0	1	0	0	0	1	0
Return Error	1	0	1	0	0	0	1	1
Reject	1	0	1	0	0	1	0	0
(reserved)	1	0	1	0	0	1	0	1
(reserved)	1	0	1	0	0	1	1	0
Return Result (Not Last)	1	0	1	0	0	1	1	1

4.2.2.2 Component ID tag

The term Component ID refers to the Invoke ID or the Linked ID. The Component ID tag is coded as shown in Table 20.

Table 20/Q.773 – Coding of Component ID Tag

	H	G	F	E	D	C	B	A
Invoke ID	0	0	0	0	0	0	1	0
Linked ID[a)]	1	0	0	0	0	0	0	0
[a)] This tag differs from the Invoke ID, which is coded as a universal INTEGER, in order to distinguish it from the following tag (Operation Code) which is also coded as a universal INTEGER.								

The length of a Component ID is 1 octet.

An Invoke Component has one or two Component IDs: an Invoke ID, and if it is desired to associate the Invoke with a previous Invoke, then the Linked ID is provided in addition to the Invoke ID.

Return Result and Return Error Components have one Component ID, called an Invoke ID which is the reflection of the Invoke ID of the Invoke Component to which they are responding.

The Reject Component uses as its Invoke ID, the Invoke ID in the Component being rejected. If this ID is unavailable (e.g. due to mutilation of the message undetected by lower layers), then the Invoke ID tag is replaced with a universal NULL tag (which always has length = 0) as shown in Table 21.

If an Invoke containing both Invoke and Linked IDs is being rejected, only the Invoke ID is used in the Reject Component.

Table 21/Q.773 – Coding of NULL tag

	H	G	F	E	D	C	B	A
NULL tag	0	0	0	0	0	1	0	1

4.2.2.3 Operation Code tag

Each operation is assigned a value to identify it. Operations can be classified as local or global operations. A local operation code follows an Operation Code Tag and Operation Code length. The Operation Code Tag is coded as shown in Table 22.

The Global Operation Code is coded as described in Recommendation X.208.

Table 22/Q.773 – Coding of Operation Code Tag

	H	G	F	E	D	C	B	A
Local Operation Code Tag	0	0	0	0	0	0	1	0
Global Operation Code Tag	0	0	0	0	0	1	1	0

4.2.2.4 Parameter tag

The Parameter tag shall be any valid ASN.1 tag, depending on the type of the parameter supplied. It can indicate either a primitive or a constructor element and refer to any of the defined tag classes.

When the parameter element is a collection of several information elements, the associated data type shall be derived from the Sequence, SequenceOf, Set or SetOF types (see Table 23).

Table 23/Q.773 – Coding of Sequence and Set Tags

	H	G	F	E	D	C	B	A
Sequence Tag	0	0	1	1	0	0	0	0
Set Tag	0	0	1	1	0	0	0	1

4.2.2.5 Error Code tag

Each error is assigned a value to identify it. Errors can be classified as local or global errors. A local error code follows the Error Code Tag and Error Code Length. The Error Code Tag is coded as shown in Table 24.

The Global Error Code is coded as an Object Identifier, which is described in Recommendation X.208.

Table 24/Q.773 – Coding of Error Code Tag

	H	G	F	E	D	C	B	A
Local Error Code Tag	0	0	0	0	0	0	1	0
Global Error Code Tag	0	0	0	0	0	1	1	0

4.2.2.6 Problem Code

The Problem Code consists of one of the four elements General Problem, Invoke Problem, Return Result Problem or Return Error Problem. The tags for these elements are coded as shown in Table 25. Their values are shown in Tables 26 to 29.

Table 25/Q.773 – Coding of Problem Type Tags

Problem Type	H	G	F	E	D	C	B	A
General Problem	1	0	0	0	0	0	0	0
Invoke	1	0	0	0	0	0	0	1
Return Result	1	0	0	0	0	0	1	0
Return Error	1	0	0	0	0	0	1	1

Table 26/Q.773 – Coding of General Problem

	H	G	F	E	D	C	B	A
Unrecognized Component [a)]	0	0	0	0	0	0	0	0
Mistyped Component [a)]	0	0	0	0	0	0	0	1
Badly Structured Component [a)]	0	0	0	0	0	0	1	0
[a)] TCAP Components are equivalent to ROSE APDUs.								

Table 27/Q.773 – Coding of Invoke Problem

	H	G	F	E	D	C	B	A
Duplicate Invoke ID	0	0	0	0	0	0	0	0
Unrecognized Operation	0	0	0	0	0	0	0	1
Mistyped Parameter[a)]	0	0	0	0	0	0	1	0
Resource Limitation	0	0	0	0	0	0	1	1
Initiating Release[b)]	0	0	0	0	0	1	0	0
Unrecognized Linked ID	0	0	0	0	0	1	0	1
Linked Response Unexpected	0	0	0	0	0	1	1	0
Unexpected Linked Operation[c)]	0	0	0	0	0	1	1	1

a) TCAP Invoke parameter is equivalent to ROSE Invoke argument.

b) ROSE uses "Initiator releasing" as only the initiator of the underlying association may release it. In TCAP, either entity may release the association.

c) ROSE refers to a linked operation as a child operation.

Table 28/Q.773 – Coding of Return Result Problem

	H	G	F	E	D	C	B	A
Unrecognized Invoke ID	0	0	0	0	0	0	0	0
Return Result Unexpected	0	0	0	0	0	0	0	1
Mistyped Parameter[a)]	0	0	0	0	0	0	1	0

a) TCAP Return Result parameter is equivalent to ROSE Return Result.

Table 29/Q.773 – Coding of Return Error Problem

	H	G	F	E	D	C	B	A
Unrecognized Invoke ID	0	0	0	0	0	0	0	0
Return Error Unexpected	0	0	0	0	0	0	0	1
Unrecognized Error	0	0	0	0	0	0	1	0
Unexpected Error	0	0	0	0	0	0	1	1
Mistyped Parameter	0	0	0	0	0	1	0	0

4.2.3 Dialogue Portion

The Dialogue Portion when present consists of one dialogue control protocol data unit or user information. The dialogue portion is type of EXTERNAL. The direct-reference element of the EXTERNAL type shall indicate the following abstract syntax name:

{ ccitt recommendation q 773 as (1) dialogue-as (1) version1 (1) }

if structured dialogue is used and:

{ ccitt recommendation q 773 as (1) unidialogue-as (2) version1 (1) }

if unstructured dialogue is used, the data value shall be one of the defined dialogue control protocol data units.

User information when present shall be identified by a user defined abstract syntax name.

4.2.3.1 Dialogue Control PDUs

The dialogue portion when present consists of one of the following dialogue control protocol data units. See Tables 30 to 62.

Table 30/Q.773 – Dialogue Portion

Element form	Dialogue Portion	Mandatory Indication
Constructor	Dialogue Portion Tag Dialogue Portion Length	Mandatory
Constructor	External Tag External Length	Mandatory
Constructor	Structured or unstructured dialogue	Mandatory

Table 31/Q.773 – Dialogue Portion Tag

	H	G	F	E	D	C	B	A
Dialogue Portion Tag[a)]	0	1	1	0	1	0	1	1
[a)] [APPLICATION 11]								

Table 32/Q.773 – External Tag

	H	G	F	E	D	C	B	A
External Tag[a)]	0	0	1	0	1	0	0	0
[a)] [UNIVERSAL 8]								

Table 33/Q.773 –Structured Dialogue

Element form	Structured Dialogue	Mandatory Indication
Primitive	Object Identifier Tag Object Identifier Length Dialogue-as-ID value	Mandatory
Constructor	Single-ASN.1-type Tag[a) b)] Single-ASN.1-type Length	Mandatory
Constructor	Dialogue PDU	Mandatory

[a)] This definition comes from the definition of the EXTERNAL type (see Recommendation X.208).

[b)] The use of the single-ASN.1-type constructor is only one possible encoding.

Table 34/Q.773 – Unstructured Dialogue

Element form	Unstructured Dialogue	Mandatory Indication
Primitive	Object Identifier Tag Object Identifier Length Unidialogue-as-ID value	Mandatory
Constructor	Single-ASN.1-type Tag[a) b)] Single-ASN.1-type Length	Mandatory
Constructor	Unidirectional Dialogue PDU	Mandatory

a) This definition comes from the definition of the EXTERNAL type (see Recommendation X.208).

b) The use of the single-ASN.1-type constructor is only one possible encoding.

Table 35/Q.773 – Object Identifier Tag

	H	G	F	E	D	C	B	A
Object Identifier Tag[a)]	0	0	0	0	0	1	1	0

a) [UNIVERSAL 6]

Table 36/Q.773 – Unidialogue-As-ID Value

	H	G	F	E	D	C	B	A
{ccitt recommendation	0	0	0	0	0	0	0	0
q	0	0	0	1	0	0	0	1
773 (X'305)[a)]	1	0	0	0	0	1	1	0
	0	0	0	0	0	1	0	1
as(1)	0	0	0	0	0	0	0	1
unidialoguePDU(2)	0	0	0	0	0	0	1	0
version1(1)}	0	0	0	0	0	0	0	1

a) Coding as defined in 22.2/X.209.

Table 37/Q.773 – Dialogue-As-ID Value

	H	G	F	E	D	C	B	A
{ccitt recommendation	0	0	0	0	0	0	0	0
q	0	0	0	1	0	0	0	1
773 (X'305)[a)]	1	0	0	0	0	1	1	0
	0	0	0	0	0	1	0	1
as(1)	0	0	0	0	0	0	0	1
dialoguePDU(1)	0	0	0	0	0	0	0	1
version1(1)}	0	0	0	0	0	0	0	1

a) Coding as defined in 22.2/X.209.

Table 38/Q.773 – Dialogue Request (AARQ-apdu)

Element form	Dialogue Request (AARQ-apdu)	Mandatory Indication
Constructor	Dialogue Request Tag Dialogue Request Length	Mandatory
Primitive	Protocol Version Tag Protocol Version Length Protocol Version	Optional[b)]
Constructor	Application Context Name Tag Application Context Name Length	Mandatory
Primitive	Object Identifier Tag Object Identifier Length Application Context Name[a)]	Mandatory
Constructor	User Information Tag User Information Length User Information	Optional

a) Coded as an OBJECT IDENTIFIER value (see Recommendation X.209).

b) If this information element is not included, its default value will be version 1.

Table 39/Q.773 – Dialogue Response (AARE-apdu)

Element form	Dialogue Response (AARE-apdu)	Mandatory Indication
Constructor	Dialogue Response Tag Dialogue Response Length	Mandatory
Primitive	Protocol Version Tag Protocol Version Length Protocol Version	Optional[b)]
Constructor	Application Context Name Tag Application Context Name Length	Mandatory
Primitive	Object Identifier Tag Object Identifier Length Application Context Name[a)]	Mandatory
Constructor	Result Tag Result Length INTEGER Tag INTEGER Length Result	Mandatory
Constructor	Result Source Diagnostic Tag Result Source Diagnostic Length Result Source Diagnostic	Mandatory
Constructor	User Information Tag User Information Length User information	Optional

a) Coded as an OBJECT IDENTIFIER value (see Recommendation X.209).

b) If this information element is not included, its default value will be version 1.

Table 40/Q.773 – Dialogue Abort (ABRT-apdu)

Element form	Dialogue Abort (ABRT-apdu)	Mandatory Indication
Constructor	Dialogue Abort Tag Dialogue Abort Length	Mandatory
Primitive	Abort Source Tag Abort Source Length Abort Source	Mandatory
Constructor	User Information Tag User Information Length User Information	Optional

Table 41/Q.773 – Dialogue Tag

	H	G	F	E	D	C	B	A
Dialogue Request Tag	0	1	1	0	0	0	0	0
Dialogue Response Tag	0	1	1	0	0	0	0	1
Dialogue Abort Tag	0	1	1	0	0	1	0	0

Table 42/Q.773 – Protocol Version Tag

	H	G	F	E	D	C	B	A
Protocol Version Tag	1	0	0	0	0	0	0	0

Table 43/Q.773 – Application Context Name Tag

	H	G	F	E	D	C	B	A
Application Context Name Tag	1	0	1	0	0	0	0	1

Table 44/Q.773 – User Information Tag

	H	G	F	E	D	C	B	A
User Information Tag	1	0	1	1	1	1	1	0

Table 45/Q.773 – Abort Source Tag

	H	G	F	E	D	C	B	A
Abort Source Tag	1	0	0	0	0	0	0	0

Table 46/Q.773 – Result Tag

	H	G	F	E	D	C	B	A
Result Tag	1	0	1	0	0	0	1	0

Table 47/Q.773 – Result Source Diagnostic Tag

	H	G	F	E	D	C	B	A
Result Source Diagnostic Tag	1	0	1	0	0	0	1	1

Table 48/Q.773 – Protocol Version

	H	G	F	E	D	C	B	A
Version 1(0)[a) b)]	0	0	0	0	0	1	1	1
	1	0	0	0	0	0	0	0

a) Coding of BIT STRING value (See Recommendation X.209).

b) This table represents a possible encoding for the value of the Protocol Version Parameter sent by a TC-implementation conforming to this Recommendation.

Table 49/Q.773 – User Information

Element form	User Information	Mandatory Indication
Constructor	EXTERNAL Tag EXTERNAL Length	Mandatory
Primitive	OBJECT IDENTIFIER Tag OBJECT IDENTIFIER Length direct reference[a)]	Optional
Primitive	INTEGER Tag INTEGER Length indirect reference[b)]	Optional
Primitive	Object Descriptor Tag Object Descriptor Length data value descriptor[c)]	Optional
Constructor/ Primitive	Encoding (as single ASN.1-type, octet-aligned or arbitrary)	Mandatory

a) Encoded as OBJECT IDENTIFIER value (see Recommendation X.209).

b) Encoded as INTEGER value (see Recommendation X.209).

c) Encoded as ObjectDescriptor value (see Recommendation X.209).

Table 50/Q.773 – Single ASN.1-Type

Element form	Single ASN.1-Type	Mandatory Indication
Constructor	Single ASN.1-Type Tag Single ASN.1-Type Length	Mandatory
Constructor/ Primitive	Any kind of ASN.1-type	Mandatory

Table 51/Q.773 – Octet-Aligned

Element form	Octet-Aligned	Mandatory Indication
Constructor/ Primitive	Octet-Aligned Tag Octet-Aligned Length Octet-Aligned [a)]	Mandatory
[a)] Coded as a OCTET STRING value (see Recommendation X.209).		

Table 52/Q.773 – Arbitrary

Element form	Arbitrary	Mandatory Indication
Constructor/ Primitive	Arbitrary Tag Arbitrary Length Arbitrary [a)]	Mandatory
[a)] Coded as a BIT STRING value (see Recommendation X.209).		

Table 53/Q.773 – Encoding Tags

	H	G	F	E	D	C	B	A
Single ASN.1-Type Tag	1	0	1	0	0	0	0	0
Octet-Aligned Tag	1	0	X	0	0	0	0	1
Arbitrary Tag	1	0	X	0	0	0	1	0

Table 54/Q.773 – Result Value

	H	G	F	E	D	C	B	A
Accepted	0	0	0	0	0	0	0	0
Reject-permanent	0	0	0	0	0	0	0	1

Table 55/Q.773 – Dialogue Service User Diagnostic

Element form	Dialogue Service User Diagnostic	Mandatory Indication
Constructor	Dialogue Service User Tag Dialogue Service User Length	Mandatory
Primitive	Integer Tag Integer Length Dialogue Service User Diagnostic Value	Mandatory

Table 56/Q.773 – Dialogue Service Provider Diagnostic

Element form	Dialogue Service Provider Diagnostic	Mandatory Indication
Constructor	Dialogue Service Provider Tag Dialogue Service Provider Length	Mandatory
Primitive	Integer Tag Integer Length Dialogue Service Provider Diagnostic Value	Mandatory

Table 57/Q.773 – Dialogue Service Diagnostic Tag

	H	G	F	E	D	C	B	A
Dialogue Service Provider Tag	1	0	1	0	0	0	1	0
Dialogue Service User Tag	1	0	1	0	0	0	0	1

Table 58/Q.773 – Dialogue Service User Diagnostic Value

	H	G	F	E	D	C	B	A
Null	0	0	0	0	0	0	0	0
No reason given	0	0	0	0	0	0	0	1
Application Context Name not supplied	0	0	0	0	0	0	1	0

Table 59/Q.773 – Dialogue Service Provider Diagnostic Value

	H	G	F	E	D	C	B	A
Null	0	0	0	0	0	0	0	0
No reason given	0	0	0	0	0	0	0	1
No common dialogue portion	0	0	0	0	0	0	1	0

Table 60/Q.773 – Abort Source

	H	G	F	E	D	C	B	A
Dialogue Service User	0	0	0	0	0	0	0	0
Dialogue Service Provider	0	0	0	0	0	0	0	1

Table 61/Q.773 – Unidirectional Dialogue (AUDT-apdu)

Element form	Unidirectional Dialogue (AUDT-apdu)	Mandatory Indication
Constructor	Unidirectional Dialogue Tag Unidirectional Dialogue Length	Mandatory
Primitive	Protocol Version Tag Protocol Version Length Protocol Version	Optional[b)]
Constructor	Application Context Name Tag Application Context Name Length	Mandatory
Primitive	Object Identifier Tag Object Identifier Length Application Context Name[a)]	Mandatory
Constructor	User Information Tag User Information Length User Information	Optional

a) Coded as an OBJECT IDENTIFIER value (see Recommendation X.209).

b) If this information element is not included, its default value will be version 1.

Table 62/Q.773 – Unidirectional Dialogue Tag

	H	G	F	E	D	C	B	A
Unidirectional Dialogue Tag	0	1	1	0	0	0	0	0

ANNEX A

The following module defines the parametrized type TCMessage which provides a basis for the definition of an abstract syntax containing TCAP messages. This type is parametrized in that the values conveyed in the component portion depend on the set of operations which can be invoked and the set of operations for which a response may be generated.

The definition of the TC PDUs will lead to identical encodings to those derived using the BER from the abstract syntax in the main body of this Recommendation.

```
TCAPMessages {ccitt recommendation q 773 modules(2) messages(1) version3(3)}

DEFINITIONS IMPLICIT TAGS ::=
BEGIN
-- EXPORTS everything

-- Transaction Portion fields.
IMPORTS

ROS{} FROM

Remote-Operations-Generic-ROS-PDUs {joint-iso-itu-t remote-operations(4) generic-ROS-PDUs(6) version1(0)}

OPERATION FROM
```

```
Remote-Operations-Information-Objects {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}

TCMessage {OPERATION: Invokable, OPERATION: Returnable} ::=CHOICE
{
 unidirectional     [APPLICATION 1] Unidirectional {{Invokable}, {Returnable}},
 begin              [APPLICATION 2] Begin {{Invokable}, {Returnable}},
 end                [APPLICATION 4] End {{Invokable}, {Returnable}},
 continue           [APPLICATION 5] Continue {{Invokable}, {Returnable}},
 abort              [APPLICATION 7] Abort
}

Unidirectional {OPERATION : Invokable, OPERATION: Returnable} ::= SEQUENCE
{
                    dialoguePortion         DialoguePortion     OPTIONAL
                    components              ComponentPortion    {{Invokable},{Returnable}}
}

Begin {OPERATION : Invokable, OPERATION: Returnable} ::= SEQUENCE
{
 otid               OrigTransactionID,
 dialoguePortion    DialoguePortion     OPTIONAL,
 components         ComponentPortion {{Invokable}, {Returnable}} OPTIONAL
}

End {OPERATION : Invokable, OPERATION: Returnable}::= SEQUENCE
{
 dtid               DestTrasactionID,
 dialoguePortion    DialoguePortion     OPTIONAL,
 components         ComponentPortion {{Invokable}, {Returnable}} OPTIONAL
}

Continue {OPERATION : Invokable, OPERATION: Returnable}::= SEQUENCE
{
otid                OrigTransactionID,
 dtid               DestTransactionID,
 dialoguePortion    DialoguePortion     OPTIONAL,
 components         ComponentPortion {{Invokable}, {Returnable}} OPTIONAL
}

Abort::=            SEQUENCE
                    {
                     dtid           DestTransactionID,
                     reason         CHOICE
                                    {
                                    p-abortCause        P-AbortCause
                                    u-abortCause        DialoguePortion
                                    }       OPTIONAL
                    }
-- NOTE – When the Abort Message is generated by the Transaction sublayer, a p-Abort Cause may be
-- present. The u-abortCause may be generated by the component sublayer in which case it is an ABRT
-- APDU, or by the TC-User in which case it could be either an ABRT APDU or data in some user-defined
-- abstract syntax.

DialoguePortion ::=         [APPLICATION 11] EXPLICIT EXTERNAL
-- The dialogue portion carries the dialogue control PDUs as value of the external data type. The direct
-- reference should be set to {ccitt recommendation q 773 as(1) dialogue-as(1) version1(1)} if structured
-- dialogue is used and to {ccitt recommendation q 773 as(1) unidialogue-as(2) version1(1)} if unstructured
-- dialogue is used.
```

```
OrigTransactionID ::=     [APPLICATION 8] OCTET STRING(SIZE(1..4))

DestTransactionID ::=     [APPLICATION 9] OCTET STRING(SIZE(1..4))

P-AbortCause ::=          [APPLICATION 10] INTEGER {
                              unrecognizedMessageType (0),
                              unrecognizedTransactionID (1),
                              badlyFormattedTransactionPortion (2),
                              incorrectTransactionPortion (3),
                              resourceLimitation (4) } (0..127)
-- COMPONENT PORTION. The last field in the transaction portion of the TCAP message is the
-- component portion. The component portion may be absent.

ComponentPortion {OPERATION : Invokable, OPERATION: Returnable} ::=

                          [APPLICATION 12] SEQUENCE SIZE (1..MAX) OF
                                                  Component {{Invokable}, {Returnable}}
-- Component Portion fields
-- Recommendation X.880 defines four Application Protocol Data Units (APDUs) for invoking
-- operations, returning results or error, and for the rejection of invalid PDUs.
-- TCAP adds returnResultNotLast to allow for the segmentation of a result.

Component {OPERATION : Invokable, OPERATION: Returnable} ::= CHOICE

                          {
                          basicROS                ROS {TCInvokeldSet, {Invokable}, {Returnable}},
                          returnResultNotLast      [7] returnResult < ROS {{Returnable}}
                          }

TCInvokeldSet INTEGER ::={-128..127}
END -- TCAPMessages
```

NOTE – The parametrized type ROS{} defined in Recommendation X.880 represents the four basic ROS PDUs: invoke, return result, return error and reject.

The information object class OPERATION defined in Recommendation X.880 is the replacement for the OPERATION MACRO.

Invokable and returnable are two sets of operations.

ITU-T RECOMMENDATIONS SERIES

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