5i’

FASCICLE VI.9

Recommendations Q.771 to Q.795

SPECIFICATIONS OF

SIGNALLING SYSTEM No. 7

Blanc

MONTAGE: PAGE 2 = PAGE BLANCHE

SECTION 1

TRANSACTION CAPABILITIES APPLICATION PART (TCAP)

Recommendation Q.771

FUNCTIONAL DESCRIPTION OF TRANSACTION CAPABILITIES

1 Introduction

1.1 *General*

Transaction Capabilities (TC) provide functions and protocols to a large variety of applications distributed over exchanges and specialized centres in telecommunication networks.

The support of TC by terminal equipments is for further study.

The term ‘‘Transaction Capabilities’’ refers to Application layer services and protocols, called Transaction Capabilities Application Part, or TCAP, plus any supporting Presentation, Session and Transport layers services and protocols, called the Intermediate Service Part, or ISP.

To date, only Signalling System No. 7 MTP plus SCCP have been considered as network layer service providers. However, any standard OSI Network Layer might be used in place of the MTP plus SCCP, provided that the requirements of the applications supported by TC (e.g. service and performance requirements) can be met. This area requires further study.

Figure 1/Q.771 shows the general structure of TC. It shows that the Transaction Capabilities Application Part (TCAP) forms a part of layer 7 of the OSI Reference Model. The remainder of layer 7 is referred to as a TC-user. The Intermediate Service Part (ISP) covers layers 4 to 6.

Figure 2/Q.771 illustrates the situation of TC in the No. 7 Signalling System.

1.2 *Contents of the Recommendations Series Q.771-Q.775*

Recommendation Q.771 contains a general description of the services provided by the Transaction Capabilities, and the service expected from the SCCP.

Recommendation Q.772 defines the Transaction Capabilities Information Elements, and their functions.

Recommendation Q.773 defines the formats and encoding used for the Transaction Capabilities Messages. Annex A specifies the protocol data units using the ASN.1 formal notation (Recommendations X.208/X.209).

Recommendation Q.774 specifies the Transaction Capabilities procedures. Annex A to this Recommendation contains SDL diagrams for TC.

Recommendation Q.775 contains guidelines and examples on how to define applications and their use of TC.

Figure 1/Q.771, p.

Figure 2/Q.771, p.

The present Recommendation contains both introductory information (chapters 1 and 2), and a detailed description (chapters 3 and 4), using primitives, of the services provided by TC. The reader interested in the first aspect only may read the first two chapters only; chapters 3 and on contain more detailed information.

1.3 *Objectives*

1.3.1 *Definition of Transaction Capabilities*

The overall objective of Transaction Capabilities is to provide the means for the transfer of information between nodes, and to provide generic services to applications, while being independent of any of these.

1.3.2 *Scope of Transaction Capabilities*

Transaction Capabilities in a Signalling System No. 7 network should be considered for use between:

1) exchanges

2) an exchange and a network service centre (e.g. data base, specialized facility unit, OA&M Centre).

3) network service centres.

The following applications have been recognized as TC-users:

— mobile service application (e.g. location of roamers)

— registration, activation and invocation of supplementary services involving specialized facility units (e.g. freephone service credit card service)

— non circuit control-related exchange of signalling information (e.g. closed user group, look-ahead procedure)

— operation and maintenance applications (e.g. query/response, bulk data transfer).

This list is not exhaustive.

These applications can be classified into two broad categories:

— real-time sensitive, with small amounts of data to be transferred

— less real-time sensitive, with possibly large amounts of data to be transferred.

A more precise definition of the boundary between these two categories requires further study. A given application is not compelled to belong to only one of these categories.

TC services offered to applications in the first category are based on a connectionless network service. They are introduced in § 2.3, and further described in chapter 3 of this Recommendation.

TC services offered to applications in the second category are based on a connection-oriented network service. They are introduced in § 2.4, and further described in chapter 4 of this Recommendation.

The mechanism for selecting a category is for further study.

2 Overview

2.1 *Terminology*

The following terms are used throughout the Q.77x Series of Recommendations and are defined in the Signalling System No. 7 glossary: class of operation; component correlation; component portion; dialogue; information element; Intermediate Service Part; linked operation; operation; reply; result; tag; transaction; Transaction Capabilities; Transaction Capabilities Application Part; transaction portion.

2.2 *Structure of TC*

2.2.1 *Architectural concepts*

The OSI protocol reference model (Recommendation X.200) is used to model TC.

From an end-user point of view, Transaction Capabilities for initially planned services lie within the Network layer of the OSI model. Provision of network layer services to end-users requires communication between TC-users at various network nodes; these intra-network communications may in turn be modelled using the 7-layer reference model of OSI.

TCAP is structured in two sub-layers:

— the component sub-layer, which deals with individual actions or data, called components

— the transaction sub-layer, which deals with the exchange of messages cotaining components between two TC-users.

This is illustrated by Figure 3/Q.771.

Figure 3/Q.771, p.

2.2.2 *Addressing issues*

When TC uses the Signalling System No. 7 network service, the addressing options supported by the SCCP are used.

When other network layer service providers are used, the addressing options supported by these providers will be used; further study on this area is required.

2.2.3 *Management aspects*

For further study.

2.2.4 *Alignment of TCAP with X.219 and X.229 (ROSE)*

The Component sub-layer of TCAP is in partial alignment with the capabilities of the Remote Operation Service Element (ROSE). The current status of TCAP and ROSE alignment is on the basis of protocol alignment, namely the X.229 protocol is contained within the TCAP component protocol. In addition, the Component sub-layer includes some extensions to ROSE. Service alignment on the primitive interface to TC/ROSE users is for further study.

The X.219 Remote Operation Service provides five classes of operations. Class 1 is synchronous, reporting both success and failure. Classes 2 to 5 are asynchronous and correspond to the TCAP operation classes 1 to 4. TCAP has not

adopted ROSE class 1 (synchronous), because the full-duplex mode of operation is used in TCAP. TC-users may use the TCAP operation class 1 in a synchronous mode if appropriate. Further details are given in Recommendation Q.775.

2.3 *TC Based on a Connectionless Network Service*

2.3.1 *Architecture*

This chapter defines a class of TC services based on a connectionless network service, in this case, no functionality is provided by the ISP, and TCAP interfaces directly with the SCCP, as represented on Figure 4/Q.771.

The class of TC services is selected by the TC-user on the basis of a Quality of Service parameter.

Figure 4/Q.771, p.

2.3.2 *Service Provided by the Component Sub-layer*

2.3.2.1 *Component*

A component consists of a request to perform an **operation** , or a **reply** .

An operation is an action to be performed by the remote end. It may have associated parameters. An invocation of an operation is identified by a component ID; this allows several invocations of the same or different operations to be active simultaneously.

One or more replies may be sent to an operation.

The ability for TC-users to exchange components which are neither operation invocations, nor replies, is for further study.

Components are passed individually between a TC-user and the Component sub-layer. The originating TC-user may send several components to the Component sub-layer before these are transmitted (in a single message) to the remote end. Whenever several components are received in a single message, each one is delivered individually to the destination TC-user.

Components in a message are delivered to the remote TC-user in the same order as they are provided at the originating interface. The importance of the order is by prior agreement between the TC-users involved.

2.3.2.2 *Dialogue*

Successive components exchanged between two TC-users in order to perform an application constitute a dialogue. The Component sub-layer provides dialogue facilities, allowing several dialogues to run concurrently between two given TC-users.

Two kinds of facilities are provided: unstructured and structured.

2.3.2.2.1 *Unstructured dialogue*

TC-users send components that do not expect replies without forming an explicit association between themselves. This is referred to as the unstructured dialogue case. The implicit association always exists between the communicating TC-users. When one TC-user sends a unidirectional message to its peer, this indicates use of the unstructured dialogue facility. A TC-user may have any number of operations active at any given time, the maximum number is dependent on the unique invoke IDs available to it at any time.

When a TC-user is a receiver of a unidirectional message, if a protocol error is to be reported, it is also returned in a unidirectional message.

2.3.2.2.2 *Structured dialogue*

Alternatively, TC-users indicate the beginning, or the formation of an association, the continuation, and the end of a dialogue; this is referred to as a structured dialogue. Using a structured dialogue allows two TC-users to run several dialogues concurrently, each being identified by a particular dialogue ID. Each dialogue ID has a separate invoke ID name space, thus allowing duplication of invoke IDs in different dialogues. In sequence delivery of messages may be provided by means of application protocols, or by use of the appropriate class of service.

When using the structured dialogue service, the TC-user has to indicate one of the following three possibilities when sending a component to its peer entity:

i) a dialogue begins;

ii) a dialogue continues: full-duplex exchange of components is possible;

iii) a dialogue ends: the sending side will not send more components, nor will it accept any more components from the remote end.

2.3.2.3 *Component Correlation*

The Component sub-layer provides the following facilities:

a) association of operations and replies

The value of the invoke ID, which identifies an operation invocation without ambiguity, is returned in a reply to that invocation.

Four classes of operations are considered:

— class 1: both success and failure are reported

— class 2: only failure is reported

— class 3: only success is reported

— class 4: neither success, nor failure is reported.

The replies to an operation consist of one or more components. Where necessary, the TC-user provides segmentation of a successful result. In addition, any number of linked operations may be sent prior to the last component of the reply.

Any kind of component, except a reject component, may be rejected. Rejection of a result causes termination of the corresponding operation; rejection of a linked operation does not affect the linked-to operation.

A TC-user may cancel an operation which it has previously invoked. No reply for this invocation will be accepted afterwards.

The last component may be:

— a return result indicating success

— a return error indicating operation failure

— a reject indicating a syntax error.

b) abnormal situations handling

The Component sub-layer covers a number of abnormal situations in relation with a component:

— component reject: when the Component sub-layer receives a malformed component, or a component which violates the rules of exchange of operations and replies, it informs the TC-user(s)

— operation expiry: when the Component sub-layer detects that a class 1, 2 or 3 operation has not received a final reply after some amount of time (which depends on the operation), it releases the corresponding invoke ID and informs the TC-user. Note that this situation is abnormal only in the case of a class 1 operation. Application of this to class 4 operations is a local matter.

2.3.2.4 *Error handling*

When the Component sub-layer is informed of a situation which prevents it from providing the service expected by the TC-users, it will notify the TC-user, and may terminate the peding operations.

A TC-user may also decide to abort a dialogue, which puts an end to any pending operation.

2.3.3 *Service provided by the Transaction Sub-layer*

The Transaction sub-layer provides the capability for the exchange of components between TR-users. The transaction sub-layer also provides the capability to send transaction messages between peer TR-layer entities by means of the services provided by the lower layer network services. The only foreseen TS-user for the moment is the component sub-layer. Two types of service are provided:

2.3.3.1 *Unstructured dialogue*

There is no explicit initiation, or termination associated with an unstructured dialogue. The only facility provided to the TC-user is the capability to send one, or several components that do not expect replies (invocation of class 4 operations) grouped in a unidirectional message to the remote TR-user.

At the originating side, the TC-user indicates the components to be sent in a unidirectional message by means of primitives of the request type containing a unique dialogue ID. When the TC-user issues a TC-UNI request primitive with the same dialogue ID, all the components with the same dialogue ID are sent as user data to the transaction sub-layer by means of the TR-UNI primitive by the component sub-layer. At the transaction sub-layer message level, the unidirectional message does not contain any transaction ID thereby providing no association between messages of this type. The dialogue ID is used to send a group of components in a UNI message to a particular destination address.

2.3.3.2 *Structured dialogue*

The structured dialogue facility allows a TC-user to start a dialogue, exchange components within this dialogue, terminate it, or abort it.

Each TR-user identifies a transaction by a separate transaction ID. The following facilities are provided:

— transaction begin: the beginning of a transaction between two TR-users causes a transaction ID to be allocated to this transaction, and permits sending TR-user information to the destination TR-user. In response to transaction begin, the destination TR-user may continue the transaction, or end it.

— transaction continuation: allows full-duplex exchange of messages between TR-users inside a transaction.

— transaction end: release the associated transaction ID, and puts an end to the exchange of messages inside this transaction. Either of the TR-users may decide to end a transaction. There are three ways for the TR-user to terminate a transaction:

1) prearranged end: a convention exists between the TR-users; each of them may decide to terminate the transaction without having to inform the peer TR-user, which will take a similar decision on its own

2) basic end: it informs the peer TR-user, possibly sending TR-user information to it.

3) transaction abort: causes the abandonment of any message of the transaction for which transmission or delivery is pending, and ends the transaction. The reason for aborting the transaction is indicated to the remote TR-user.

— if, for some reason, no response of any kind is received to transaction begin, the Transaction sub-layer will eventually abort this transaction and inform the TR-user. This is a local option.

— transaction abort by TCAP: whenever one of a list of abnormal situations is detected, the Transaction sub-layer decides to abort the corresponding transaction and informs the TR-users.

— exception reporting: the Transaction sub-layer may report to TR-users abnormal situations of which it is notified by the underlying layer.

When the TR-user is the Component sub-layer:

a) there is a one-to-one mapping between a dialogue and a transaction,

b) a message may contain zero, one or more components, within the limits of the message size supported by the underlying layer.

2.4 *TC Based on a connection-oriented network service*

For further study.

3 Service provided by TC based on a connectionless network service

3.1 *Component Sub-layer*

3.1.1 *Overview of Component Sub-layer primitives*

Tables 1/Q.771 and 2/Q.771 give an overview of the primitives to/from the TC-users, and contain references to the sections of this Recommendation where these primitives are described in detail.

Table 1/Q.771 shows the TC-primitives relating to dialogue handling. The purpose of these primitives is to request or indicate facilities of the underlying (sub)-layer, in relation with message transmission or dialogue handling. When the Transaction Sub-layer is used to support the dialogue, these primitives map onto TR-primitives with the same generic name, as there is a one to one relationship between a dialogue and a transaction.

H.T. [T1.771]

TABLE 1/Q.771

Primitives for dialogue handling

Name	Type	Section
TC-UNI	Request Indication	3.1.2.2.1
TC-BEGIN	Request Indication	3.1.2.2.2.1
TC-CONTINUE	Request Indication	3.1.2.2.2.2
TC-END	Request Indication	3.1.2.2.2.3
TC-U-ABORT	Request Indication	3.1.2.2.2.3
TC-P-ABORT	Indication	3.1.4.2

Table 1/Q.771 [T1.771], p.

— TC-UNI: requests/indicates an unstructured dialogue.

— TC-BEGIN: begins a dialogue.

— TC-CONTINUE: continues a dialogue.

— TC-END: ends a dialogue.

Each of the previous primitives causes any component(s) previously passed on the interface for the referenced dialogue to be delivered to the remote end (except TC-END with prearranged end).

— TC-U-ABORT: allows a TC-user to terminate a dialogue abruptly, without transmitting any pending components.

— TC-P-ABORT: informs the TC-user that the dialogue has been terminated by the service provider (i.e. TC Transaction sub-layer) in reaction to a transaction abort by the Transaction sub-layer. Any pending components are not transmitted.

Table 2/Q.771 shows the TC-primitives for component handling. The main purpose of these primitives is to handle operations and replies; these primitives do not as such require facilities from the underlying (sub)-layer.

H.T. [T2.771]

TABLE 2/Q.771

Primitives for component handling

Name	Type	Section
TC-INVOKE	Request Indication	3.1.3.2
TC-RESULT-L	Request Indication	3.1.3.3
TC-RESULT-NL	Request Indication	3.1.3.3
TC-U-ERROR	Request Indication	3.1.3.4
TC-L-CANCEL	Indication	3.1.3.6
TC-U-CANCEL	Request	3.1.3.6
TC-L-REJECT	Indication	3.1.4.1
TC-R-REJECT	Indication	3.1.4.1
TC-U-REJECT	Request Indication	3.1.3.5

Table 2/Q.771 [T2.771], p.

— TC-INVOKE: invocation of an operation, which may be linked to another operation invocation

— TC-RESULT-L: only result or last part of the segmented result of a successfully executed operation

— TC-RESULT-NL: non-final part of the segmented result of a successfully executed operation

— TC-U-ERROR: reply to a previously invoked operation, indicating that the operation execution failed

— TC-L-CANCEL: informs the TC-user locally that an operation invocation is terminated due to a timeout condition

— TC-U-CANCEL: causes local termination of an operation invocation, as a consequence of a TC-user decision

— TC-L-REJECT: (local reject) informs the local TC-user that a Component sub-layer detected invalid component was received

— TC-R-REJECT: (remote reject) indicates that TCAP detected an invalid component

— TC-U-REJECT: rejection of a component by the TC-user, indicating a malformation which prevents the operation from being executed, or the reply from being understood

The various primitives associated with component and dialogue handling are described with their parameters. The following notation is used:

(M) indicates a mandatory parameter

(O) indicates an optional parameter

FS indicates that further study is required

A blank indicates that the parameter is not applicable

(=) indicates that the parameter must have the same value in a request primitive and in the corresponding indication primitive.

This notation applies throughout this Recommendation.

3.1.2 *Dialogue Handling*

Dialogue handling provides facilities for the exchange of components within a dialogue.

3.1.2.1 *Definition of Parameters*

This section defines the parameters used with the primitives associated with dialogue handling.

Address parameters: two address parameters are used: the ‘‘Destination Address’’ and the ‘‘Originating Address’’ parameters. These parameters identify respectively the destination and originating TC-user.

‘‘Components Present’’: indicates whether any components will be received; when no component is being transmitted, it indicates that the list is empty, other wise it indicates a sequence (see § 3.1.3.8) of components which are associated with the dialogue handling primitive. The ‘‘Components Present’’ parameter is used in primitives of the indication type only.

‘‘Dialogue ID’’: this parameter also appears in the component handling primitives, and is used to associate components with a dialogue. The same dialogue ID must be used within the same dialogue, or a unidirectional primitive. In a unidirectional primitive the same dialogue ID assures all components with the identical dialogue ID are blocked together in the same unidirectional message destined for the same destination address. For structured dialogues, the dialogue ID is used to identify all the components belonging to the same dialogue from the beginning of the dialogue to its end. The dialogue ID maps onto the IDs exchanged in the messages between a pair of nodes.

‘‘P-ABORT’’: contains information indicating the cause for which TCAP decides to abort a dialogue.

‘‘Parameters’’: contains the parameter(s) to be sent to the remote TC-user in association with an operation invocation, a reply, or a dialogue abort. This information is not analysed by TCAP.

‘‘Quality of Service’’: the TC-user indicates the acceptable quality of service. The default value of this parameter corresponds to the underlying service defined in § 3.4. Other Quality of service is for further study.

‘‘Termination’’: indicates which scenario is chosen by the TC-user for the end of the dialogue (prearranged or basic).

‘‘User Abort Information’’: the TC-user may include information related to a TC-user-initiated abort.

3.1.2.2 *Dialogue facilities*

The dialogue facilities allow a TC-user to exchange components with a peer TC-user to perform a distributed application. The unidirectional message facility may be used to send class 4 operation invocations and reports of protocol errors in these invocations from either TC-user using an unstructured dialogue. The structured dialogue facilities provide the capability to explicitly initiate a transaction, exchange components within the dialogue, terminate it, or abort it.

3.1.2.2.1 *Unstructured dialogue*

There is no initiation or termination associated with an unstructured dialogue; the only facility provided is the request for transmission of one, or several components invoking class 4 operations or reporting protocol errors in these invocations, grouped in a message to the remote TC-user.

Components to be transmitted have been previously passed to the component sub-layer by means of component handling primitives of the ‘‘request’’ type.

The use of the unstructured dialogue facility is indicated by issuing a TC-UNI primitive, as described in Table 3/Q.771.

At the originating side, a TC-UNI request primitive is issued to request transmission to the remote TC-user of all the components which have been passed to the component sub-layer with the same dialogue ID.

At the receiving side, the destination TC-user is informed that one or more component(s) have been received by means of a TC-UNI indication primitive. The parameters in this primitive apply to all the components being received; these components will actually be delivered by means of component handling primitives of the indication type.

H.T. [T3.771]

TABLE 3/Q.771

TC-UNI Primitives

Parameter	Primitive: TC-UNI	
	Request	Indication
Quality of service	FS	
Destination address	M	M | ua)
Originating address	M | ua)	M (=)
Dialogue ID	M | ub)	
Components present	M	M (=)

a) This parameter may be implicitly associated with the access point at which the primitive is issued.

b) This parameter has only local significance.

Table 3/Q.771 [T3.771], p.

3.1.2.2.2 *Structured dialogue*

The structured dialogue facility allows a TC-user to start a dialogue, exchange components within this dialogue, terminate it, or abort it. It provides for Transaction IDs in the transaction messages that provide a unique association among the related transaction messages.

3.1.2.2.2.1 *Beginning of a dialogue*

A TC-user begins a new dialogue by issuing a TC-BEGIN request primitive. The purpose of this primitive is:

— to indicate to the Component sub-layer that a new dialogue starts, identified by the Dialogue ID parameter of the primitive;

— to request transmission of any component(s) previously passed to the Component sub-layer by means of component handling primitives of the ‘‘request’’ type with the same Dialogue ID.

A TC-BEGIN request primitive may be issued prior to passing any component to the Component sub-layer.

At the receiving side, the destination TC-user is informed that a new dialogue starts by means of a TC-BEGIN indication primitive. The presence of component(s) is indicated by the Components Present.

Table 4/Q.771 describes the TC-BEGIN primitives.

H.T. [T4.771]
TABLE 4/Q.771
TC-BEGIN Primitives

Parameter	Primitive: TC-BEGIN	
	Request	Indication
Quality of service	FS	FS
Destination address	M	M | ua)
Originating address	M | ua)	M (=)
Dialogue ID	M	M
Components present		M

a) This parameter may be implicitly associated with the access point at which the primitive is issued.

Table 4/Q.771 [T4.771], p.

3.1.2.2.2.2 *Dialogue continuation*

A TC-user indicates that it wants to continue a dialogue by issuing a TC-CONTINUE request primitive. This primitive requests transmission of any component(s) that have been passed to the Component sub-layer for this dialogue, since the last TC-BEGIN or TC-CONTINUE request primitive was issued for this dialogue.

At the receiving side, the TC-CONTINUE indication primitive indicates:

— that the dialogue may continue

— that components are being delivered (if the Components Present parameter does not indicate ‘‘empty’’).

Table 5/Q.771 describes the TC-CONTINUE primitives.

H.T. [T5.771]

TABLE 5/Q.771

TC-CONTINUE Primitives

Parameter	Primitive: TC-CONTINUE	
	Request	Indication
Dialogue ID	M	M
Components present		M

Table 5/Q.771 [T5.771], p.

3.1.2.2.2.3 *End of a dialogue*

Three scenarios are provided to TC-users to end a dialogue:

— prearranged end

— basic end

— abort by the TC-user.

Dialogue ending uses the TC-END request and indication primitives described in Table 6/Q.771. The TC-END request primitive indicates which scenario is being used for the dialogue.

H.T. [T6.771]

TABLE 6/Q.771

TC-END Primitives

Parameter	Primitive: TC-END	
	Request	Indication
Dialogue ID	M	M
Components present		M
Termination	M	

Table 6/Q.771 [T6.771], p.

a) prearranged end

In this scenario, TC-users have decided by prior arrangement when to end the dialogue: the effect of the TC-END primitive is purely local; no TC-END indication is used.

No component can be sent or received for the dialogue once the TC-END request primitive has been issued.

b) basic end

In this scenario, the ending causes transmission of any pending components at the side which initiates it. Note, however, that any components for which transmission would be pending in the reverse direction will not be delivered.

The basic scenario uses the TC-END primitives for two purposes:

— delivery of any component(s) that has been passed to the Transaction sub-layer, and for which transmission is pending

— indication that no more components will be exchanged for this dialogue in either direction.

c) abort of a dialogue by a TC-user

The TC-user has the ability to request immediate ending of a dialogue without taking into account any pending operation invocation (abort). When doing so, the TC-user may provide end to end information indicating the cause of the abort and diagnostic information; this information is transported by TCAP without analysis.

The TC-U-ABORT request and indication primitives are used to indicate abort by the TC-user; Table 7/Q.771 describes these primitives.

H.T. [T7.771]

TABLE 7/Q.771

TC-U-ABORT Primitives

Parameter	**Primitive: TC-U-ABORT**	
	Request	**Indication**
Dialogue ID	**M**	**M**
User abort information	**O**	**O (=)**

Table 7/Q.771 [T7.771], p.

3.1.3 *Component Handling*

3.1.3.1 *Definition of Parameters*

This section defines the parameters used with the primitives associated with component handling.

‘‘Class’’: see § 2.3.2.3.

‘‘Dialogue ID’’: relates components to a specific dialogue.

‘‘Invoke ID’’: identifies an operation invocation.

‘‘Linked ID’’: links an operation invocation to a previous operation invocation.

‘‘Error’’: contains information provided by the TC-user when an operation returns failure. This information is not analysed by TCAP.

‘‘Last Component’’: is used in primitives of the ‘‘indication’’ type only, to designate the last component of a message. Note that indication of the last part of the result of an operation is via the name of the primitive.

‘‘Operation’’: identifies the action to be executed by a TC-user on request of another TC-user.

‘‘Parameters’’: contains any parameters accompanying an operation, or provided in reply to an operation.

‘‘Problem Code’’: identifies the cause for rejecting a component.

‘‘Timeout’’: indicates the maximum lifetime of a component ID. It is used to handle cases where operations do not receive any expected reply.

3.1.3.2 *Operation Invocation*

An operation invocation is requested to the Component sub-layer by means of a TC-INVOKE request primitive. When this invocation is linked to a previous operation, the Linked ID parameter is used.

A corresponding TC-INVOKE indication primitive is used to indicate operation activation to the destination TC-user.

Table 8/Q.771 shows the primitives associated with operation invocation.

H.T. [T8.771]

TABLE 8/Q.771

Operation invocation primitives

Parameter	Primitive: TC-INVOKE	
	Request	Indication
Dialogue ID	M	M | ua)
Class	M	
Invoke ID	M	M (=)
Linked ID	O	O (=)
Operation	M	M (=)
Parameters	O	O (=)
Last component		M
Timeout	M	

a) Mandatory except for invocation of class 4 operation received in a unidirectional message.

Table 8/Q.771 [T8.771], p.

3.1.3.3 *Report of success*

Success is reported to indicate that an operation (of class 1 or 3) has been executed by the remote TC-user. The operation is identified in the Invoke ID parameter. Several replies may be used to report success. The following primitives are used:

— TC-RESULT-L indicates the only or last segment of a result

— TC-RESULT-NL indicates a segment of a result (with more segments to follow)

There is no limitation on the number of segments.

The TC-RESULT-L and TC-RESULT-NL primitives are described in Table 9/Q.771. A primitive of the ‘‘request’’ type is used to pass a result from the TC-user to the Component sub-layer; a primitive of the ‘‘indication’’ type is used to deliver this result to the TC-user.

H.T. [T9.771]

TABLE 9/Q.771

Report of success primitives

Parameter	Primitive { {	
Dialogue ID	M	M
Invoke ID	M	M (=)
Parameters	O	O (=)
Last component		M

Table 9/Q.771 [T9.772], p.

3.1.3.4 *Report of failure*

A TC-user receiving a (class 1 or 2) operation which it cannot execute, though it ‘‘understands’’ it, will issue a TC-U-ERROR request primitive, indicating the reason of the failure (Error parameter). The corresponding operation is identified by the Invoke ID parameter.

The TC-user which invoked this operation is informed by a TC-U-ERROR indication primitive.

Table 10/Q.771 describes the TC-U-ERROR primitives.

H.T. [T10.771]

TABLE 10/Q.771

Report of failure primitives

Parameter	Primitive: TC-U-ERROR	
	Request	Indication
Dialogue ID	M	M
Invoke ID	M	M (=)
Error	M	M (=)
Parameters	O	O (=)
Last component		M

Note — Report of failure is a final reply.

Table 10/Q.771 [T10.771], p.

3.1.3.5 *Reject by the TC-User*

A TC-user may reject any component (except a reject component) generated by its peer entity, which it considers incorrect. The cause for the rejection is indicated in the Problem Code parameter; separate parameters are available for the rejection of individual component types.

Any rejection of an invocation or a result terminates the operation. When a linked operation is rejected, the linked-to operation is not affected.

A TC-user rejects a component by means of the TC-U-REJECT request primitive, and is informed of rejection by the remote TC-user by means of the TC-U-REJECT indication primitive. These primitives are described by Table 11/Q.771.

H.T. [T11.771]
TABLEAU 11/Q.771
User rejection primitives

Parameter	Primitive: TC-U-REJECT	
	Request	Indication
Dialogue ID	M	M | ua)
Invoke ID	M	M (=)
Problem code	M	M (=)
Last component		M

a) Mandatory except for rejection of invocation of class 4 operation received in a unidirectional message.

Table 11/Q.771 [T11.771], p.

3.1.3.6 *Cancel of an Operation*

The cancel facility terminates the corresponding operation invocation. It can be requested either by the TC-user, or by the Component sub-layer. In both cases, it has only local effect: no notification is sent to the remote end.

The Component sub-layer uses the cancel facility to inform the TC-user that the timer associated with a class 1, 2 or 3 operation has expired; the TC-L-CANCEL indication primitive is used for this purpose. The timer is run for all classes, but the reporting for class 4 operations is a local matter.

The TC-user uses the TC-U-CANCEL request primitive to inform the local Component sub-layer of a cancel decision. No component is sent.

Table 12/Q.771 describes the TC-CANCEL primitives.

H.T. [T12.771]

TABLE 12/Q.771

TC-CANCEL Primitives

Parameter	Primitive	
	TC-L-CANCEL indication	TC-U-CANCEL request
Dialogue ID	M	M
Invoke ID	M	M

Table 12/Q.771 [T12.771], p.

3.1.3.7 *Grouping of Components inside a Message*

A sequence of components is obtained by passing one or several components with a given Dialogue ID to the Component Sub-layer between two successive requests for transmission (TC-BEGIN, TC-CONTINUE or TC-END request primitives), or before the first one (TC-BEGIN request), using the same Dialogue ID, or the only request for transmission (i.e. TC—UNI).

At the originating side, a list of components is delimited by TC-UNI, TC-BEGIN, TC-CONTINUE or TC-END request primitives.

At the destination side, a sequence of components starts with a primitive indicating transmission; its end is indicated by the ‘‘Last Component’’ parameter of the primitives which deliver components to a TC-user. The ‘‘Components Present’’ parameter in the transmission primitive indicates whether the sequence is empty, or not.

Note — Components grouped inside a message are delivered to the remote end in the same order as they are provided by the TC-user at the originating end.

3.1.4 *Abnormal situations*

3.1.4.1 *Reject of a Component by the Component sub-layer*

When detecting that a received component is invalid, the Component sub-layer notifies the local TC-user by means of the TC-L-REJECT indication primitive. This primitive indicates the cause of the reject (Problem Code parameter) with sufficient information to make the retention of the failed component superfluous: whenever possible the Component Type and Component ID are indicated; otherwise a ‘‘general problem’’ cause is indicated. This information is passed to the TC-user, and also retained in the Component sub-layer which uses it to form a reject component.

Any type of component can be rejected. When the component to be rejected is itself identified as a reject component, rejection is purely local; when the rejected component is identified as an invoke or a result, the whole corresponding operation is considered as terminated; when it is a linked operation, this linked operation is terminated, but the linked-to operation is not affected.

When informed of a Component sub-layer reject, the local TC-user may decide to continue the exchange of components. If so, the remote TC-user is informed through the reject component sent when the local TC-user issues the next dialogue handling primitive.

If the Component sub-layer generated reject combined with accumulated components from the TC-user exceeds the message length limitations, then the TC-user, being aware of the reject component, must initiate two dialogue handling primitives. The Component sub-layer, also being aware of the length problem, will send all the components, except the reject, with the first primitive. The reject will be sent with the next dialogue handling primitive together with any further components provided by the TC-user.

Table 13/Q.771 describes the primitives used in relation with TCAP component rejection.

H.T. [T13.771]

TABLE 13/Q.771

Component sub-layer rejection primitive

Parameter	**Primitive**	
	TC-L-REJECT indication	**TC-R-REJECT indication**
Dialogue ID	**M**	**M | ua)**
Invoke ID	**O**	**O**
Problem code	**M**	**M**
Last component		**M**

a) Mandatory except for rejection of invocation of a class 4 operation received in a unidirectional message.

Table 13/Q.771 [T13.771], p.

3.1.4.2 *Dialogue abort*

Due to an abnormal situation, an underlying (sub-)layer may decide to abort the association between users; the dialogue has then to be aborted. All associated operations are terminated, and the TC-users are notified by means of TC-P-ABORT indication primitives. The P-abort parameter contains the cause for which it was decided to abort the dialogue.

The Component sub-layer does not decide on dialogue abort.

Table 14/Q.771 describes the TC-P-ABORT primitive.

H.T. [T14.771]

TABLE 14/Q.771

Primitive for TCAP Abort

Parameter	Primitive TC-P-ABORT indication
Dialogue ID	M
P-abort	M

Table 14/Q.771 [T14.771], p.

3.1.5 *Component states and state transition diagrams*

For a given component ID, component correlation takes place only at the side which originates the operation; for this ID, component states and state transition diagrams are defined at this side only. The other side just reflects the value of the component ID in an Invoke or a Linked ID.

The following states are defined:

— Idle: no activity associated with the component ID

— Operation Pending: an operation has been passed to the Component sub-layer, but no request for transmission has been issued

— Operation Sent: an operation has been transmitted to the remote end, but no result has been received

— Wait for Reject: the result has been received; TCAP is waiting for its possible rejection by the TC-user

— Reject pending: reject of the result has been requested by the TC-user, but no request for transmission has been issued.

State transition diagrams are defined for the four classes of operations.

Note 1 — Each of these diagrams corresponds to one component ID: the one indicated in the Invoke ID parameter; linked operations do not alter the state machine of the linked-to operation.

Note 2 — TC-END request or indication primitives, TC-U-ABORT request or indication primitives, or the TC-P-ABORT indication primitive cause return to the ‘‘Idle’’ state of any component ID associated with the dialogue. Corresponding transitions are not represented on the diagrams.

Figure 5/Q.771, p.18

Figure 6/Q.771, p.19

Figure 7/Q.771, p.20

Figure 8/Q.771, p.21

3.1.6 *Mapping of Component sub-layer onto Transaction sub-layer*

When mapping the Component sub-layer onto the Transaction sub-layer, a one to one mapping exists between a dialogue and a transaction explicity in the case of a structured dialogue, or implicitly in the case of an unstructured dialogue. It follows that there is a one to one relationship between dialogue handling primitives of the Component sub-layer and transaction handling primitives in the Transaction sub-layer; similar generic names have been chosen for the primitives to reflect this. The component handling primitives of the Component sub-layer have no counterpart in the Transaction sub-layer.

The correspondence between the two sub-layers is further described in Recommendation Q.774.

3.2 *Transaction Sub-layer*

3.2.1 *Overview of Transaction Sub-layer primitives*

Table 15/Q.771 gives an overview of the primitives between the TR users and the Transaction sub-layer. A detailed description of these primitives and their parameters is given in the next sections. For each primitive, Table 15/Q.771 indicates the relevant section.

H.T. [T15.771]

TABLE 15/Q.771

Primitives for the transaction sub-layer

Name	Type	Section
TR-UNI	Request indication	3.2.2
TR-BEGIN	Request indication	3.2.3
TR-CONTINUE	Request indication	3.2.4
TR-END	Request indication	3.2.5
TR-U-ABORT	Request indication	3.2.5.3
TR-P-ABORT	Indication	3.2.6.1

Table 15/Q.771 [T15.771], p.

Definition of the parameters :

‘‘Quality of Service’’: the TR-user indicates the preferred quality of service. This is for further study.

‘‘Destination Address’’: identifies the destination TR-user.

‘‘Originating Address’’: identifies the originating TR-user.

‘‘P-abort’’: indicates the cause of the abort of a transaction by TCAP.

‘‘Reason’’: indicates the nature of an abnormal situation.

‘‘Transaction ID’’: a transaction is identified by a separate transaction ID at each end.

‘‘Termination’’: identifies the termination scenario chosen for the transaction (prearranged or basic).

‘‘User Abort Information’’: information related to a TR-user abort.

‘‘User Data’’: contains the information to be passed between TR-users.

3.2.2 *Information Transfer In Unstructured Dialogue*

Information may be sent from one TR—user to another TR-user without establishing an explicit association. In this case, the transaction sub-layer considers that there is no relationship among messages transmitted by this means.

The corresponding primitives are the TR-UNI request and indication primitives, described in Table 16/Q.771.

H.T. [T16.771]
TABLE 16/Q.771
TR-UNI Primitives

Parameter	Primitive: TR-UNI	
	Request	Indication
Quality of service	FS	—
Destination address	M	M | ua)
Originating address	M | ua)	M (=)
User data	M	M (=)

a) This parameter may be implicitly associated with the access point at which the primitive is issued.

Table 16/Q.771 [T16.771], p.

3.2.3 *Transaction begin*

The transaction begin facility starts a transaction between two TR-users. This may be accompanied by the transfer of TR-user information (called user data in the following).

In order to begin a transaction, a TR-user issues the TR-BEGIN request primitive.

At the destination side, the TR-BEGIN indication primitive is used to inform the destination TR-user of the beginning of a transaction, and to deliver any accompanying user data.

Table 17/Q.771 describes the transaction begin primitives.

H.T. [T17.771]

TABLE 17/Q.771

Primitives for transaction begin

Parameter	Primitive: TR-BEGIN	
	Request	Indication
Quality of service	FS	FS
Destination address	M	M | ua)
Originating address	M | ua)	M (=)
Transaction ID	M	M
User data	O	O (=)

a) This parameter may be implicitly associated with the access point at which the primitive is issued.

Table 17/Q.771 [T17.771], p.

Figure 9/Q.771 shows the transaction state transitions during transaction begin. The following states are introduced:

— Idle (I): the transaction does not exist

— Init Sent (IS): the transaction just started at the originating side

— Init Received (IR): the transaction just started at the destination side.

Figure 9/Q.771 [T18.771], p. (traiter comme tableau MEP)

3.2.4 *Transaction continuation*

Transaction continuation allows two TR-users to exchange messages in both directions inside a transaction. The TR-CONTINUE primitives are used for this purpose. They are described by Table 18/Q.771.

The Transaction sub-layer does not provide segmentation/reassembly or flow control.

State transitions associated with the continuation of a transaction are represented on Figure 10/Q.771, where state A (Active) indicates that the transaction was accepted by the remote end, and the transaction can be used to exchange messages in both directions.

H.T. [T19.771]

TABLE 18/Q.771

Transaction Continuation Primitives

Parameter	Primitive: TR-CONTINUE	
	Request	Indication
Transaction ID	M	M
User Data	O	O (=)

Table 18/Q.771 [T19.771], p.

Figure 10/Q.771, p.

3.2.5 *Transaction End*

Three facilities are provided to a TR-user to end a transaction:

— prearranged end

— basic end

— abort.

The first two facilities use the TR-END primitives; the Termination parameter indicates which option is selected. The TR-END primitives are described by Table 19/Q.771.

The last facility uses the TR-U-ABORT primitives described by Table 20/Q.771.

H.T. [T20.771]

TABLE 19/Q.771

TR-END primitives

Parameter	Primitive: TR-END	
	Request	Indication
Transaction ID	M	M
Termination	M	
User data	O	O (=)

Table 19/Q.771 [T20.771], p.

3.2.5.1 *Prearranged end*

When prearranged end has been selected, the procedure is purely local. Each TR-user may decide to end the transaction at any point in time, regardless of the current transaction state. The TR-END request primitive only is used: the remote TR-user is not informed, and should request transaction end on its own.

The User Data parameter should not be present in this case.

Figure 11/Q.771 shows the transaction state transitions for prearranged end of a transaction. The states are those defined in 3.2.3 and 3.2.4 above.

Figure 11/Q.771, p.

3.2.5.2 *Basic end*

When basic termination has been selected, the TR-user requests the end of the transaction by issuing the TR-END request primitive indicating this option; the primitive may then contain User Data which is sent to the peer entity.

At the destination side, the TR-END indication primitive is used to inform the TR-user of the end of the transaction, and deliver any accompanying User Data.

Figure 12/Q.771 shows the transaction state transitions for the basic end of transaction. The states are those defined in §§ 3.2.3 and 3.2.4 above.

Figure 12/Q.771, p.

3.2.5.3 *Transaction Abort by the TR-user*

A TR-user may request the abort of a transaction at any moment; it uses for this purpose the TR-U-ABORT request primitive, which may optionally contain the cause of the abort, and/or additional end to end information. This information is contained in the User Abort Information parameter: it is transmitted without analysis to the peer entity. Any messages of the transaction for which transmission is pending are discarded.

A TR-user is informed of the decision of its peer entity to abort the transaction by means of the TR-U-ABORT indication primitive.

TR-U-ABORT primitives are described by Table 20/Q.771.

H.T. [T21.771]

TABLE 20/Q.771

TR-U-ABORT Primitives

Parameter	Primitive: TR-U-ABORT	
	Request	Indication
Transaction ID	M	M
User Abort Information	O	O (=)

Table 20/Q.771 [T21.771], p.

3.2.6 *Abnormal situations*

3.2.6.1 *Abort by the Transaction Sub-layer*

The abort facility may be invoked by the Transaction sub-layer in reaction to abnormal situations. The possible reasons for such a decision are indicated in Recommendation Q.774.

Transaction abort causes the abandonment of any message of the transaction for which transmission is pending.

Transaction abort is made by means of the TR-P-ABORT indication primitive described by Table 21/Q.771.

H.T. [T22.771]

TABLE 21/Q.771

Transaction sub-layer abort primitive

Parameter	Primitive TR-P-ABORT indication
Transaction ID	M
P-abort	M

Table 21/Q.771 [T22.771], p.

Figure 13/Q.771 shows the state transitions for transaction abort. The states are those defined in §§ 3.2.3 and 3.2.4 above.

Figure 13/Q.771 [T23.771], p. (traiter comme tableau MEP)

3.3 *Services provided by the ISP*

No additional service is provided by the ISP when the TC-service is based on a connectionless network service.

3.4 *Services assumed from the connectionless network layer*

In the Signalling System No. 7 environment, the services assumed from the SCCP are those defined in Recommendation Q.711, § 2.2 (SCCP Connectionless Services, class 0 or class 1).

Relations of TC with the SCCP management require further study.

4 Service provided by TC based on a connection-oriented network service

For further study.

Recommendation Q.772

TRANSACTION CAPABILITIES INFORMATION ELEMENT DEFINITIONS

1 General

This Recommendation describes the individual information elements and parameters used within Transaction Capabilities messages. The encoding and formatting of these elements are shown in Recommendation Q.773.

The meaning of each information element is described in general terms.

For TC based upon a connectionless network service, the current TC is equivalent to the Transaction Capabilities Application Part (TCAP).

The TCAP message format consists of two parts, namely the transaction portion and the component portion. Information in the component portion concerns individual operations and their replies. The transaction portion contains protocol control information for the transaction sub-layer.

For a more detailed analysis of the architecture, see Figure 3/Q.771, and associated text.

2 Transaction portion

The transaction portion of a TC message may contain the following information elements, viz:

2.1 *Message type*

Five types of messages are defined for the transaction portion as follows:

2.1.1 *Unidirectional*

This message is used when there is no need to establish a transaction with another peer TR-User.

2.1.2 *Begin*

This message is used to initiate a transaction with another peer TR-User.

2.1.3 *End*

This message is used to terminate a transaction with another peer TR-User.

2.1.4 *Continue*

This message is used to complete the establishment of a transaction and to continue an established transaction.

2.1.5 *Abort*

This message is used to terminate a transaction following an abnormal situation detected by the transaction sub-layer (the service provider), or to abort a transaction by the TR-User (the service user).

2.2 *Transaction IDs*

Transaction IDs are independently assigned by each of the two nodes communicating via a transaction, enabling each node to uniquely identify the transaction and associate the entire contents of the message with that particular transaction. There are two types of Transaction IDs, viz:

2.2.1 *Originating Transaction ID*

The Originating Transaction ID is assigned by the node sending a message, and is used to identify the transaction at that end.

2.2.2 *Destination Transaction ID*

The Destination Transaction ID identifies the transaction at the receiving end. The first Originating Transaction ID value received is reflected as the Destination Transaction ID value.

2.3 *P-Abort Cause*

This is used when the transaction sub-layer aborts a transaction.

P-Abort cause definitions are as follows:

2.3.1 *Unrecognized Message Type*

The message type is not one of those defined in §§ 2.1.1 to 2.1.5 above.

2.3.2 *Unrecognized transaction ID*

A transaction ID has been received for which a transaction does not exist at the receiving node.

2.3.3 *Badly formatted transaction portion*

The transaction portion of the received message does not conform to the X.209 encoding rules as outlined in Recommendation Q.773, § 3.

2.3.4 *Incorrect transaction portion*

The elemental structure within the transaction portion of the received message, does not conform to the rules for the transaction portion defined in Recommendation Q.773 § 5.

2.3.5 *Resource limitation*

Sufficient resources are not available.

2.4 *User abort information*

This is used to pass User Specified Information by the TR-User when it aborts a transaction.

2.5 *Component portion*

This contains the component portion. When the component portion is empty this information element is not present.

3 Component Portion

The Component Portion contains the following types of information elements. They are delivered to the user at the receiving end in the same order in which they were received from the user at the originating end.

3.1 *Component type*

There are five types of component that may be present in the Component Portion of a TC message. The four Protocol Data Units (PDUs) defined in Recommendation X.229 are used, viz:

H.T. [T1.772]

TCAP component	X.229 PDU
Invoke	ROIV
Return result (last)	RORS
Return error	ROER
Reject	RORJ

Table [T1.772], p.

The remaining component type - Return Result (Not Last) - is defined by TCAP.

These component types are defined as follows:

3.1.1 *Invoke*

The invoke component requests that an operation be performed. It may be linked to another operation invocation previously sent by the other end.

3.1.2 *Return result (Not Last)*

When TC uses a connectionless Network Service, it may be necessary for the TC-User to segment the result of an operation. In this case this component is used to convey each segment of the result except the last, which is conveyed in a Return Result (Last) component.

3.1.3 *Return result (Last)*

The Return Result (Last) component reports successful completion of an operation. It may contain the last/only segment of a result.

3.1.4 *Return error*

The Return Error component reports that an operation has not been successfully completed.

3.1.5 *Reject*

The Reject component reports the receipt and rejection of an incorrect component, other than a Reject component. The possible causes for rejecting a component are defined by the Problem Code element in § 3.8.

3.2 *Invoke ID*

An Invoke ID is used as a reference number to identify uniquely a request for an operation. It is present in any reply to an Invoke component (Return Result, Return Error or Reject), enabling the reply to be correlated with the invoke.

3.3 *Linked ID*

A Linked ID is included in an invoke component by a node when it responds to an operation invocation with a linked operation invocation. The node receiving the Linked ID uses it for correlation purposes, in the same way that it uses the invoke ID in Return Result, Return Error and Reject components.

3.4 *Operation code*

The Operation Code element indicates the precise operation to be invoked, and is present in an invoke component type. The operation may be a local operation or a global operation. A local operation can be used in one ASE only. The same global operation can be used in several different ASEs.

The actual operation codes, the definition of the operations and their associated parameters, are defined in relevant ASE specifications. The component sub-layer does not set or examine the operation code value, nor which parameters are present, nor the parameter values.

3.5 *Set (of parameters)*

The Set element is used to contain a set of parameters accompanying a component. It is required in the case of more than one parameter being included in a component. The parameters themselves are defined in relevant ASE specifications.

3.6 *Sequence (of parameters)*

The Sequence element is used similarly to the Set element, except that a specific sequence of parameters is included in the component.

3.7 *Error code*

The Error Code element contains the reason why an operation cannot be completed successfully. It is present only in a Return Error component. As with operations, errors may be local or global.

These errors and associated parameters are defined in relevant ASE specifications.

3.8 *Problem code*

The Problem code element contains the reason for the rejection of a component, and one such element is present in a Reject component. Four problem code elements are defined, viz:

3.8.1 *General problem*

This element contains one of the problem codes which apply to the component sub-layer in general, and which do not relate to any specific component type. All of these are generated by the component sub-layer. They are:

3.8.1.1 *Unrecognized component*

The component type is not recognized as being one of those defined in § 3.1.

3.8.1.2 *Mistyped component*

The elemental structure of a component does not conform to the structure of that component as defined in Recommendation Q.773 § 6.

3.8.1.3 *Badly structured component*

The contents of the component do not conform to the encoding rules defined in Recommendation Q.773 § 3.

3.8.2 *Invoke problem*

This element contains one of the problem codes which relate only to the invoke component type. They are:

3.8.2.1 *Duplicate invoke ID*

The invoke ID is already in use by a previously invoked operation. This code is generated by the TC-User.

3.8.2.2 *Unrecognized operation*

The operation code value is not one of those used by the ASE. This code is generated only by the TC-User.

3.8.2.3 *Mistyped parameter*

A parameter tag is not one of those associated with the operation invoked. This code is generated only by the TC-User.

3.8.2.4 *Resource limitation*

Sufficient resources are not available to perform the operation requested. This code is generated by the TC-User.

3.8.2.5 *Initiating release*

The operation requested cannot be invoked as the dialogue is about to be released. This code is generated only by the TC-User.

3.8.2.6 *Unrecognized linked ID*

The linked ID does not correspond to a previously invoked operation. This code is generated only by the component sub-layer.

3.8.2.7 *Linked response unexpected*

The operation referred to by the linked ID is not an operation for which linked invokes are allowed. This code is generated only by the TC-User.

3.8.2.8 *Unexpected linked operation*

The linked operation is not one of those that the operation referred to by the linked ID allows. This code is generated only by the TC-User.

3.8.3 *Return result problem*

This element contains one of the problem codes which relate only to the return result component type. They are:

3.8.3.1 *Unrecognized invoke ID*

No operation with the specified invoke ID is in progress. This code is generated by the component sub-layer.

3.8.3.2 *Return result unexpected*

The invoked operation does not report success. This code is generated by the component sub-layer.

3.8.3.3 *Mistyped parameter*

A parameter tag is not one of those associated with the outcome of the operation. This code is generated only by the TC-User.

3.8.4 *Return error problem*

This element contains one of the problem codes which relate only to the return error component type. They are:

3.8.4.1 *Unrecognized invoke ID*

No operation with the specified invoke ID is in progress. This code is generated by the component sub-layer.

3.8.4.2 *Return error unexpected*

The invoked operation does not report failure. This code is generated by the component sub-layer.

3.8.4.3 *Unrecognized error*

The reported error is not one of those defined for the ASE. This code is generated by the TC-User.

3.8.4.4 *Unexpected error*

The received error is not one of those which the invoked operation may report. This code is generated by the TC-User.

3.8.4.5 *Mistyped parameter*

A parameter tag is not one of those associated with the outcome of the operation. This code is generated only by the TC-User.