INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Q.755.2

(09/97)

SERIES Q: SWITCHING AND SIGNALLING

Specifications of Signalling System No. 7 – Signalling System No. 7 management

Transaction capabilities test responder

ITU-T Recommendation Q.755.2

(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
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.755.2

TRANSACTION CAPABILITIES TEST RESPONDER

Summary

The 1993 version of Recommendation Q.755 for Protocol Testers contained just the MTP Tester. It has now been split into a series of Recommendations.

Recommendation Q.755.2 defines the Transaction Capabilities (TC) Test Responder; it is an addition to the 1993 version of Recommendation Q.755.

A TC Test Responder present in the system under test allows the use of Abstract Test Suites for testing the defined TC functionality of that system, independently of the functionalities employed by the TC-Users actually present.

Source

ITU-T Recommendation Q.755.2 was prepared by ITU-T Study Group 11 (1997-2000) and was approved under the WTSC Resolution No. 1 procedure on the 12th of September 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.755.2

TRANSACTION CAPABILITIES TEST RESPONDER

(Geneva, 1997)

1 Scope

This Recommendation defines the Transaction Capabilities (TC) Test Responder, to be used as an aid when testing the TC of ITU Signalling System No. 7.

The Recommendation draws upon Recommendation Q.750 for architectural considerations of the relationship between the Test Responder and SS No. 7 management (OMAP) [9].

The Recommendation defines a simple and flexible Test Responder which enables the use of Abstract Test Suites (ATS) for Transaction Capabilities independently from the actual TC-Users which reside in a System Under Test (SUT).

No assumption is made about the actual implementation of the interface between this function and TC.

The availability of a standardized TC Test Responder has the following advantages:

- It makes it possible to write a unique Abstract Test Suite which can be executed against any TC implementation which resides in a system where the Test Responder is also implemented.
- It allows all the defined TC functionalities to be tested, irrespective of the subset of functionalities actually used by the TC-Users which are available when an equipment is under test.
- It helps to isolate faults during testing, since the proper response to a TC message or component will be independent of the proper execution of a real TC-User operation.
- It allows the TC stack to be tested before being delivered to a customer for supporting one or more particular TC-User applications.
- The Test Responder can even be used to perform stack-to-stack interoperability testing independently of any particular TC-User application.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; all users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

[1] ITU-T Recommendations X.680-X.683 (1994), *Information technology – Abstract Syntax Notation One (ASN.1).*

[2] ITU-T Recommendation X.690 (1994) | ISO/IEC 8825-1:1995, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER).*

[3] ITU-T Recommendations Q.711-Q.714 (1993), *Specifications of Signalling System No. 7: Signalling connection control part*.

[4] CCITT Recommendations Q.771-Q.775 (1988), *Specifications of Signalling System No. 7: Transaction Capabilities Application Part (TCAP).*

[5] ITU-T Recommendations Q.771-Q.775 (1993), *Transaction capabilities application part*.

[6] ITU-T Recommendation X.290 (1995), *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications – General concepts.*

ISO/IEC 9646-1:1994, *Information technology – Open Systems Interconnection – Conformance testing methodology and framework – Part 1: General concepts.*

[7] ITU-T Recommendation I.320 (1993), *ISDN protocol reference model.*

[8] CCITT Recommendation I.321 (1991), *B-ISDN protocol reference model and its application*.

[9] ITU Recommendation Q.750 (1993), *Overview of Signalling System No. 7 management.*

3 Symbols and abbreviations

This Recommendation uses the following abbreviations:

AE	Application Entity
APDU	Application Protocol Data Unit
ASE	Application Service Element
ASN.1	Abstract Syntax Notation One
ATS	Abstract Test Suite
DPC	Destination PC
GPC	Generator PC
IUT	Implementation Under Test
LME	Level Management Entity
LT	Lower Tester
MIB	Management Information Base
MSU	Message Signal Unit
MT	MTP Tester
MTP	(SS No. 7) Message Transfer Part
OPC	Originating PC
PC	Point Code
PDU	Protocol Data Unit
SAP	Service Access Point
SCCP	(SS No. 7) Signalling Connection Control Part
SP	Signalling Point
SS No. 7	Signalling System No. 7

SUT	System Under Test
TC	Transaction Capabilities
TMP	Test Management Protocol
TPC	Turn-around PC
TTCN	Tree and Tabular Combined Notation
UT	Upper Tester

4 General

The protocol tester may be used as an aid when testing the Transaction Capabilities (TC) of Signalling System No. 7 (SS No. 7), either when performing validation testing of an implementation or compatibility testing between two implementations. The testers' main function is simulation of an ordinary user part or application, as seen from TC for the generation of test traffic.

Recommendations I.320 [7] and I.321 [8] specify the ISDN protocol reference model to be used for N-ISDN and B-ISDN. User plane (U-plane), Control plane (C-plane) and Management plane (M-plane) are identified. The layering principles apply in each of these planes. The U-plane provides the user information flow transfer with associated controls. The C-plane handles the call and connection control information. The M-plane is divided into two portions, the Layer Management functions and the Plane Management functions. The Plane Management performs management functions related to a system as a whole; it provides coordination between all the planes and has no layered structure. The Layer Management plane contains Layer Management Entities (LME). Each of them provides management functions relating to resources and parameters residing in its own protocol entity. Layer Management handles the operation and maintenance information flows. The interface between adjacent layers within a plane and between LME and its associated layer have to be defined in terms of service primitives. The interface between the LMEs and Plane Management does not need to be specified; it is implementation dependent.

For SS No. 7, the **Level Management Entity** is defined by analogy with the LME of Recommendations I.320 and I.321. This is to account for the different positions of the boundaries between SS No. 7 lower levels and those of OSI (e.g. the upper part of the MTP is level 3 in SS No. 7, the SCCP is level 4, but both would be within layer 3 if the OSI model is strictly applied). For SS No. 7, the term "LME" is taken to mean "Level Management Entity".

Thus the TT (TC Test Responder) is within the LME of the application AE. The procedures, the messages and the TT substructure are described. The undefined primitives between the Plane Management (MIB) and the TT are only required to activate/deactivate the concerned testing functions (see Figure 1, which is a copy of Figure 5/Q.750).

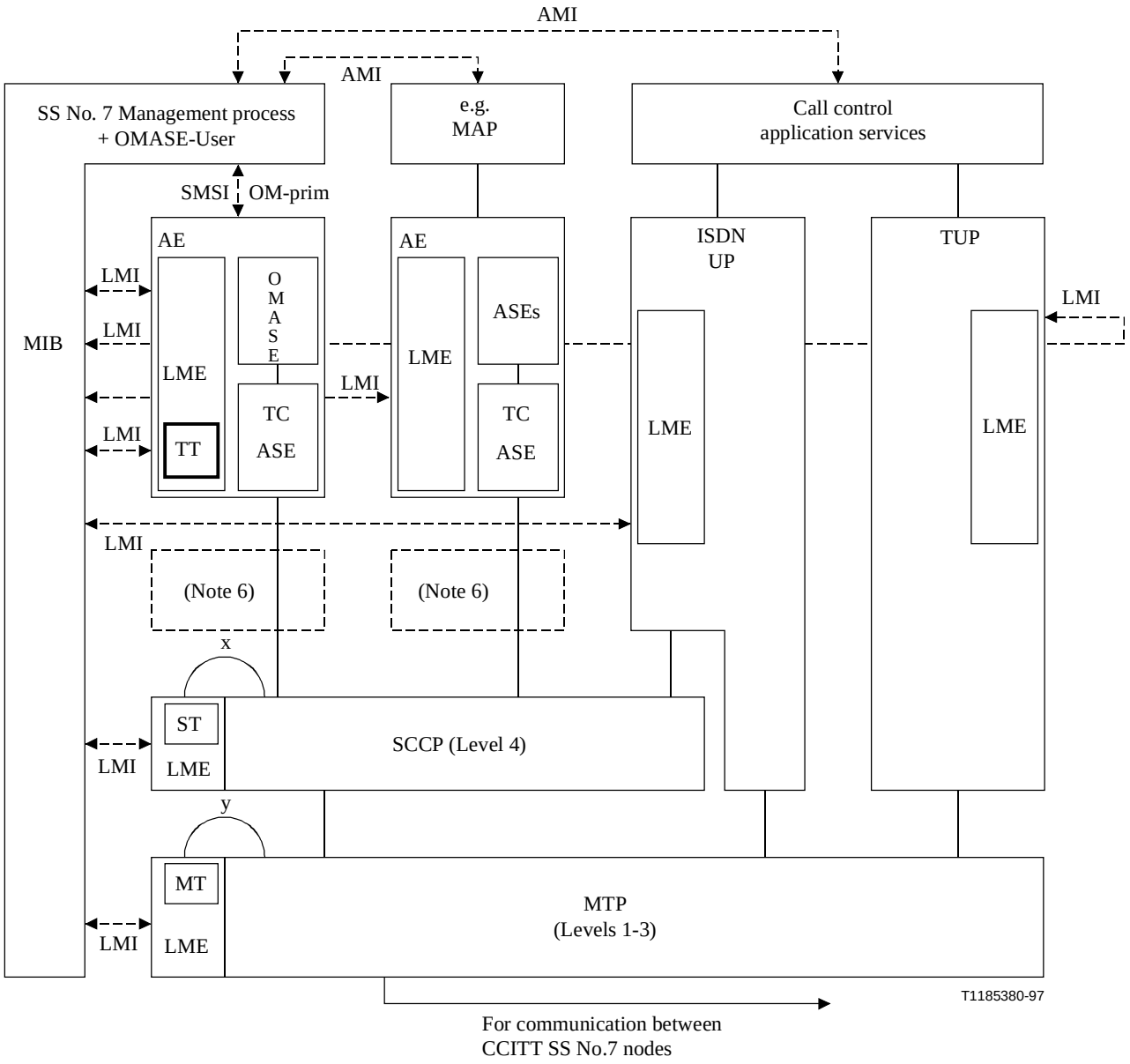

Figure 1/Q.755.2 – SS No. 7 management and internal configuration of an SP

For an explanation, see Figure 5/Q.750 [9].

5 TC Test Responder

5.1 Architecture

The TC Test Responder is a particular TC-User which can be implemented together with TC in any system, using several possible configurations.

The communication between the TC Test Responder which resides in a System Under Test (SUT) and a Test System relies on the use of a particular ASE, called the "TC Testing User" ASE. The TC Test Responder plays the role of the ASE supplier while the Test System plays the role of the ASE consumer.

The TC Testing User ASE can be implemented as the single component of an Application Entity (AE) with a particular subsystem number, or it can be combined with other application layer elements in which case it is selected by using any application-context-name starting with the following value:

{itu-t recommendation q 755 ac(5)}

The SCCP [3], allocates a subsystem number which may be used for addressing an AE which contains the Test Responder (e.g. when only *Blue Book* TC facilities are available).

According to OSI Methodology for Conformance Testing [6], the set of data units conveyed between the instances of Test Responder and Test System can be considered as an "in-band" Test Management Protocol (TMP), which can be used to support coordinated test methods between the Test Responder acting as an Upper Tester (UT) and the Lower Tester (LT) functionality of the Test System (see Figure 2).

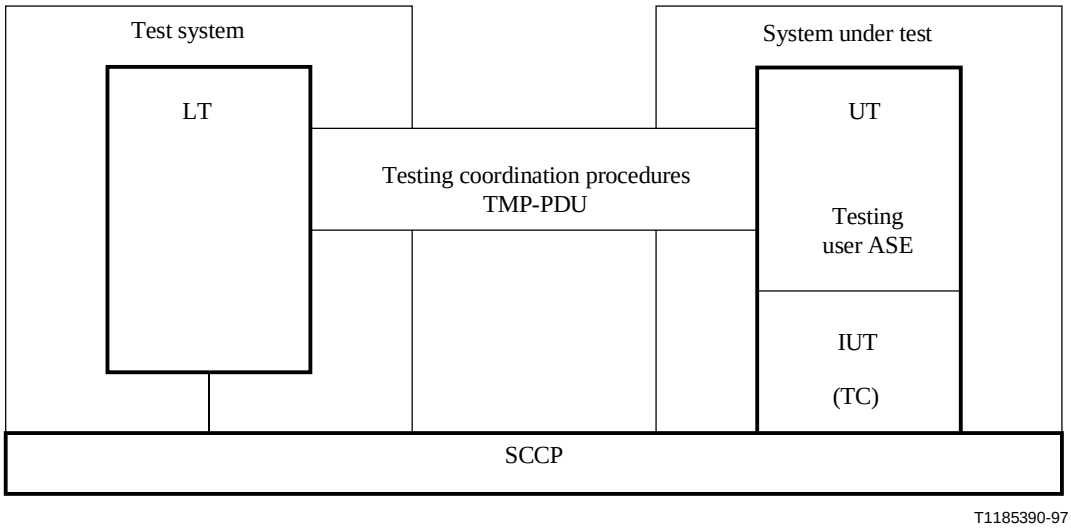

Figure 2/Q.755.2 – Arrangement for test architecture

The use of this Test Responder does not require any access to the TC service interface within the SUT, nor does it place any constraints on its implementation. This Test Responder does not provide means for testing the behaviour of a TC implementation in response to invalid behaviour of the local TC-User.

The "in-band" nature of the TMP implies that, although being under test, the TC service is reliable enough to carry such PDUs and deliver them to a User. In order to overcome potential difficulties due to a missing or unreliable TC service, the Testing User ASE procedures are defined in such a way that each TMP-PDU can be received in different types of messages or components.

The main purpose of the TMP is to allow a series of commands to be sent from the Test System to the Test Responder in order to provoke some behaviour in the TC Implementation Under Test (IUT). Each command is either a TC service primitive to be passed by the Test Responder to the TC under test or an indication that the Test Responder should wait for a subsequent event.

5.2 TC Testing User ASE

5.2.1 General principles

The TC Testing User ASE can be built on top of the 1988 version [4] or 1993 version [5] of TC[1]. This ASE simultaneously can handle several dialogues. It embodies the knowledge of seven operations, two of them can be invoked by the ASE consumer (i.e. the Test System), five of them can be invoked by the ASE supplier (i.e. the System Under Test). The two operations invoked by the test responder are defined as class 1 operations. However, this has no impact on the behaviour of the Test Responder.[2]

The Test Management Protocol Data Units (TMP-PDUs) can be conveyed in the operations' arguments, operations' result parameters, errors parameters and, when available, in the user information parameter of the dialogue portion.

There is no specific relation between a TMP-PDU and the type of message or component which is used to carry it.

NOTE – The upper service interface of the TC Testing User ASE is not standardized. However, such an interface could be made accessible locally to trigger a particular test scenario (e.g. for interoperability testing) or enable the test responder to report to some management function expiry of the timer T-Test.

5.2.2 Operations and errors

The ASN.1 module TC-Testing User defined in 5.4 contains the specification of the operations which can be invoked by the Test System or the Test Responder during communication.

The local values assigned to the following operations and errors are considered as default values. The actual local values used for these operations and errors should be treated as configuration parameters.

class1SupplierOperation
class2SupplierOperation
class3SupplierOperation
class4SupplierOperation
localConsumerError
localSupplierError

5.3 TC test management protocol

5.3.1 Test management protocol data units

There are three types of Test Management Protocol Data Units (TMP-PDU)[3]:

Test Init

The testInit PDU requests the Test Responder to initiate a test session and conveys a set of commands to be executed sequentially by the Test Responder. This PDU can only be sent by the test system.

---

[1] It is up to the test suite designer to write the test cases in such a way that the Test Responder does not request 1993 functionalities from a 1988 TC implementation.

[2] Whether or not the test responder reports an outcome for these operations depends on the commands received in the TMP-PDUs. This does not violate any rule as far as TC is concerned since the class of an operation is irrelevant to the TC residing at the side where the operation is executed.

[3] The need for a "TestEnd" PDU is for further study.

Test Continue

The testContinue PDU conveys a set of additional commands to be executed sequentially by the Test Responder. This PDU can only be sent by the test system.

Test Data Echo

The testDataEcho PDU conveys user data echoing the user data received with a particular command. This PDU can only be sent by the test responder.

The testInit and testContinue PDUs can be conveyed in the argument of the operations invoked by the Test System, in the result or error parameter returned by the Test System in response to an operation invoked by the Test Responder or in the user information parameter of the dialogue portion of the messages sent by the Test System.

The testDataEcho PDU can be conveyed in the argument of the operations invoked by the Test Responder, in the result or error parameter returned by the System Under Test in response to an operation invoked by the Test System, or in the user information parameter of the dialogue portion of the messages sent by the Test Responder.

5.3.2 Command structure

Each command sent from the Test System in a testInit or test Continue PDU indicates to the Test Responder that it should wait for an external event from the local TC or that it should issue a particular request primitive to the local TC.

Commands indicating to the Test Responder that it has to wait for an external event also indicate whether this event should occur in a particular dialogue or that this does not matter.

Commands to the Test Responder to issue a primitive comprise up to three items:

1) an indication of the type of request primitive to be passed to the local TC;
2) the reference to the dialogue over which the action should be taken. If this information is not explicitly provided, the Test Responder assumes that the command refers to the dialogue over which the TMP-PDU has been received. The dialogue reference is always chosen by the Test System. Unlike transaction Ids and dialogue Ids, the dialogue reference is a common reference shared by both sides of a dialogue;

   The Test Responder has to keep track of the one-to-one relation which exists between a dialogue-reference and the local TC dialogue-Id which has been agreed between TC and the Testing User ASE for this dialogue.
3) optionally, a user data parameter to be echoed as user data associated with the service primitive to be issued.

5.3.3 Abstract syntax

The ASN.1 module TC-TMP defined in 5.5 contains the specification of the Test Management Protocol data units.

When the TMP-PDUs are conveyed in the user information parameter of the dialogue portion, the following abstract-syntax-name is used as a direct-reference to identify the set of data values, each of which is a value of type

TMP-Protocol.TMP-PDU:

{itu-t recommendation q 755 as(4) tmp-pdus (1) version1(1)}

The applicable encoding rules are the basic encoding rules defined in Recommendation X.690 [2].

5.3.4 Procedures

5.3.4.1 Procedure at the Test System side

The Testing User ASE has limited error detection and error recovery capabilities. It is up to the designer of the abstract test suite to ensure that the sequence of commands which are sent to the Test Responder corresponds to a valid TC-User behaviour and does not conflict with the automatic procedures of this ASE.

It is up to the Test Suite designer to choose how a TMP-PDU is conveyed to the Test Responder (i.e. which message and which component, if any, etc.). The Test System can send a TMP-PDU in any type of message and component with the sole restriction that a testInit PDU can only be sent in a Begin message or in a Unidirectional message.

It is also the responsibility of the designer of the abstract test suite to ensure that the TMP-PDU will be delivered to the Test User ASE (i.e. it should be conveyed in a valid TC message or component).

Starting a Test Case: At the beginning of each Test Case, the Test System sends a TestInit PDU in order to ensure that no resources associated with a previous test case are still active. This PDU includes a sequence of commands to be executed by the Test Responder and optionally the value of a watch-dog timer (T-Test). If no timer value is provided, an implementation value is selected[4].

Continuing a Test Case: If all the commands are not sent in the testInit PDU, the Test System can send further commands to the Test Responder using the testContinue PDU which also contains a sequence of commands to be executed by the Test Responder. A TC message can convey more than one testContinue PDU.

Ending a Test Case: There is no specific command nor TMP-PDU for ending a test case. It is up to the test designer to write the test case postamble in such a way that all active dialogues are closed.

5.3.4.2 Procedure at the Test Responder side

5.3.4.2.1 Generic rules

The Test Responder refuses any application-context-name whose object identifier value does not start with the following root:

{itu-t recommendation q 755 ac(5)}

When the Test Responder accepts a 1993 dialogue (i.e. a dialogue according to [5]), it uses the received application-context-name in the next dialogue handling primitive it issues.

When the Test Responder refuses a 1993 dialogue, it proposes the following application-context-name:

{itu-t recommendation q 755 ac(5) testing-ac (1) version1 (1)}

When requesting TC to send a Begin message or a Unidirectional message, the Test Responder uses as destination address the originating address of the message in which the TestInit PDU was received. If an application-context-name has to be provided by the Test Responder, it always uses the following value:

{itu-t recommendation q 755 ac(5) testing-ac (1) version1 (1)}

---

[4] Implementors may choose a very large value if the test responder is intended to be used for running background traffic or load testing.

If the Test Responder detects that it has been reached with a destination address which is not the one it knows from the configuration data, it forces TC to insert the address it knows in the first backward Continue message sent to the Test System.

During one dialogue, the following rules apply:

- If the first invoke Id used by the Test Responder has the value 0, then this value is incremented each time the Test Responder invokes an operation.
- If the invocation received by the test responder gives reason for a user reject to be generated, the Test Responder automatically requests TC to return a reject component (e.g. it rejects any operation which is not defined in the TC-Testing-User module) with the appropriate reject problem.
- The Test Responder always requests the return option when issuing a TC-BEGIN request primitive. In all other cases the setting of this parameter is an implementation option.
- When requesting the sending of partial results, the Test Responder always requests the sequencing option. In all other cases the setting of this parameter is an implementation option.
- The timer value provided by the Test Responder when invoking an operation is a configuration parameter.

5.3.4.2.2 Handling of TMP-PDUs

If an event is received without any TMP-PDU, the Test Responder does nothing.[5]

If an event is received with several TMP-PDUs, they are processed sequentially, starting with the ones included in the user information field of the dialogue portion (if any).

On receipt of a testInit PDU, the Test Responder releases all the active resources (e.g. it closes any active dialogue, releases the dialogue references, etc.)[6] and starts the T-Test timer. Then the commands are executed sequentially and the test responder waits for the next event or expiry of timer T-Test.

On Receipt of a testContinue PDU, the Test Responder executes sequentially the specified commands and waits for the next event or expiry of timer T-Test.

On receipt of a testDataEcho PDU, the Test Responder does nothing.

According to the generic rules, receipt of any other data value will lead to:

- a TC-user reject procedure if the invalid PDU value is received in an operation argument, a result parameter or an error parameter;
- a TC-user abort procedure if it is received in the user information field of the dialogue portion.

When the T-Test timer expires, all the resources associated with the test are automatically released.

The Test Responder interprets and executes sequentially each command received from the Test System, as described in 5.3.4.2.3 and 5.3.4.2.4.

---

[5] This includes receipt of a TC-NOTICE indication primitive which may can contain a returned TMP-PDU.

[6] How the resources are released is an implementation matter. However, this should not lead to the sending of any external message (e.g. END or ABORT message).

5.3.4.2.3 Execution of the wait command

The Test Responder waits for an incoming event from the IUT. When the incoming event has been received it executes the next command (if any) or waits for the next TMP-PDU.[7]

Conceptually, an incoming event corresponds to one or more related TC indication primitives. The latter case corresponds to a situation where the Test Responder receives a dialogue handling primitive indicating that components are present. In such a case, the Test Responder consumes all the component handling primitives until the "last component" parameter takes the value "TRUE". However, this does not place any constraint on the actual implementation of the interface between the Test responder and TC.

If the command is accompanied by an explicit dialogue reference, primitives whose dialogue identifier does not correspond to this reference are ignored (The Test Responder waits for a next event). Otherwise, the dialogue identifier value is not checked.

5.3.4.2.4 Execution of other commands

The Test Responder interprets the service type as follows:

- v1988uniReq: The Test Responder requests the IUT to send a 1988 Unidirectional message.
- v1993uniReq: The Test Responder requests the IUT to send a 1993 Unidirectional message.
- v1988beginReq: The Test Responder requests the IUT to send a 1988 Begin message.
- v1993beginReq: The Test Responder requests the IUT to send a 1993 Begin message including a dialogue portion.
- continueReq: The Test Responder requests the IUT to send a Continue message for the dialogue which corresponds to the dialogue-reference value included in the command.
- basicEndReq: The Test Responder requests the IUT to send an End message for the dialogue which corresponds to the dialogue-reference value included in the command.
- localEndReq: The Test Responder requests the IUT to terminate locally the dialogue which corresponds to the dialogue-reference value included in the command.
- uAbortReq: The Test Responder requests the IUT to abort the dialogue which corresponds to the dialogue-reference value included in the command. If the dialogue has been established using the 1993 procedure, the abort-reason is set to "user-specific".
- class1invokeReq: The Test Responder requests the IUT to invoke the class 1 operation defined in the module TC-Testing-User.
- class2invokeReq: The Test Responder requests the IUT to invoke the class 2 operation defined in the module TC-Testing-User.
- class3invokeReq: The Test Responder requests the IUT to invoke the class 3 operation defined in the module TC-Testing-User.
- class4invokeReq:The Test Responder requests the IUT to invoke the class 4 operation defined in the module TC-Testing-User.
- linkedInvokedReq: The Test Responder request the IUT to invoke the class 1 operation defined in the module TC-Testing-User as a linked operation to the oldest pending operation.
- resultNlReq: The Test Responder requests the IUT to send a result not last component in response to the oldest pending operation.

---

[7] The Test Responder always waits for events. The wait command is only required when there is a need to explicitly wait for an event before executing a command which has been sent previously.

- resultLReq: The Test Responder requests the IUT to send a result last component in response to the oldest pending operation.
- uErrorReq: The Test Responder requests the IUT to send an error component in response to the oldest pending operation. The error code is the single error code allowed by the operation definition.
- uCancelReq: The Test Responder requests the IUT to cancel the oldest pending operation.
- uRejectReq: The Test Responder requests the IUT to send a reject component indicating "invoke-problem: resource limitation" for the oldest pending operation.

NOTE 1 – In this context, a "1988 message" is a message defined in [4], a "1993 message" is one defined in [5].

NOTE 2 – For the TC-Testing-User module, see 5.4.

5.3.4.2.5 Data echo

When requesting a dialogue handling service from the 1993 TC (see [5]), and if a "to-be-echoed" element was present in the associated command received from the Test System, the Test Responder includes one or more testDataEcho TMP-PDUs as the value of the user information parameter. Each "testDataEcho" value is identical to the received "toBeEchoed" value. During the dialogue establishment phase or when the unstructured dialogue mode is used, the number of "testDataEcho" values provided by the Test Responder is a configuration parameter. Once the dialogue has been established, only one value can be sent.

When requesting a component handling service from TC, the Test Responder includes a testDataEcho TMP-PDU as user parameter (i.e. operation argument, result parameter, error parameter, etc.) if a "to-be-echoed" element was present in the associated command received from the Test System. If included, the "testDataEcho" value is identical to the received "toBeEchoed" value.

If the Test Responder receives some unknown information in the user information parameter of a dialogue handling primitives, it provides it unchanged in the user information parameter of the next dialogue handling primitive it issues.

5.4 TC test responder operations and errors

The following module defines the TC-Testing-User ASE in term of which operations can be invoked by the Consumer (Test System) and the Supplier (System Under Test).

```
TC-Testing-User {itu-t recommendation q 755 modules(0) testing-user(1) version1(1)}

DEFINITIONS ::=
BEGIN

IMPORTS

OPERATION, ERROR
        FROM TCAPMessages {itu-t recommendation q 773 modules(2) messages(1) version2(2)}

APPLICATION-SERVICE-ELEMENT
        FROM Notation-Extensions {joint-iso-ccitt remote-operations(4) notation-extension(2)}

TMP-PDU
        FROM TC-TMP {itu-t recommendation q 755 modules(0) tmp(2) version1(1)}
;
```

```
-- application-context-names

ac-id  OBJECT IDENTIFIER ::= {itu-t recommendation q 755 ac(5) }
testing-ac-id OBJECT IDENTIFIER ::= {ac-id  testing-ac(1) version1(1)}

-- ase
Testing-User-ASE APPLICATION-SERVICE-ELEMENT
CONSUMER INVOKES  {
                  localConsumerOperation,
                  globalConsumerOperation
                  }
-- consumer is the test system
SUPPLIER INVOKES  {
                  class1SupplierOperation,
                  class2SupplierOperation,
                  class3SupplierOperation,
                  class4SupplierOperation,
                  globalSupplierOperation
                  }
-- supplier is the test responder

::= {itu-t recommendation q 755 ase(3) testing-user(1) version1(1)}


LocalConsumerOperation ::= OPERATION
ARGUMENT          TMP-PDU
RESULT            TMP-PDU
ERRORS            {localSupplierError}


GlobalConsumerOperation ::= OPERATION
ARGUMENT          TMP-PDU
RESULT            TMP-PDU
ERRORS            {globalSupplierError}


Class1SupplierOperation ::= OPERATION
ARGUMENT          TMP-PDU
RESULT            TMP-PDU
ERRORS            {localConsumerError}
LINKED            {localConsumerOperation}

Class2SupplierOperation ::= OPERATION
ARGUMENT          TMP-PDU
ERRORS            {localConsumerError}
LINKED            {localConsumerOperation}

Class3SupplierOperation ::= OPERATION
ARGUMENT          TMP-PDU
RESULT            TMP-PDU

Class4SupplierOperation ::= OPERATION
ARGUMENT          TMP-PDU

GlobalSupplierOperation ::= OPERATION
ARGUMENT          TMP-PDU
RESULT            TMP-PDU
ERRORS            {globalConsumerError}
LINKED            {globalConsumerOperation}
```

```
ConsumerError ::= ERROR
PARAMETER               TMP-PDU

SupplierError ::= ERROR
PARAMETER               TMP-PDU


localConsumerOperation LocalConsumerOperation ::= localValue : 0

globalConsumerOperation GlobalConsumerOperation ::=
            globalValue : {itu-t recommendation q 755 operations(1) consumer(1)}

class1SupplierOperation   Class1SupplierOperation ::= localValue : 1
class2SupplierOperation   Class2SupplierOperation ::= localValue : 2
class3SupplierOperation   Class3SupplierOperation ::= localValue : 3
class4SupplierOperation   Class4SupplierOperation ::= localValue : 4
globalSupplierOperation   Class1SupplierOperation ::=
                          globalValue : {itu-t recommendation q 755 operations(1) supplier(2)}

localConsumerError        ConsumerError ::= localValue : 1
globalConsumerError       ConsumerError ::=
                          globalValue : {itu-t recommendation q 755 errors(2) consumer(1)}

localSupplierError        SupplierError ::= localValue : 2
globalSupplierError       SupplierError ::=
                          globalValue : {itu-t recommendation q 755 errors(2) supplier(2)}


END
```

5.5 Test management protocol data units

The following module defines the Test Management Protocol data units.

```
TC-TMP {itu-t recommendation q 755 modules(0) tmp(2) version1(1)}

DEFINITIONS IMPLICIT TAGS ::=

BEGIN

TMP-PDU ::=  CHOICE
            {
        testInit            [0] TestInit,
        testContinue        [1] CommandSequence,
        testDataEcho        [2] UserData
            }


CommandSequence ::=     SEQUENCE SIZE(0..maxNbOfCommands)  OF TestCommand

maxNbOfCommands  INTEGER ::= 30

TestInit ::= SEQUENCE
        {
        timeout             INTEGER (1..127) OPTIONAL-- T-Test (unit is 30 sec)
        commands            CommandSequence,
        ...
        }
```

```
UserData ::=  CHOICE{
        simple          OCTET STRING (SIZE(0..maxUserDataLength)),
        complex         [0] ABSTRACT-SYNTAX.&Type
                        }

maxUserDataLength INTEGER ::= 2048

TestCommand ::=  CHOICE
                {
                wait            [0] DialogueReference,
                action          [1] ActionInfo
                }

DialogueReference ::= CHOICE
{
        unspecified             NULL,
        dialogue                INTEGER (0..255)
}

ActionInfo ::= SEQUENCE
{
        service                 ServiceType,
        dialogueReference       DialogueReference       DEFAULT  unspecified : NULL,
        to-be-echoed            UserData      OPTIONAL,
        ...
}

ServiceType ::= ENUMERATED
                        {
                                v1988uniReq (10),
                                v1993uniReq (11),
                                v1988beginReq (12),
                                v1993beginReq (13),
                                continueReq (14),
                                basicEndReq (15),
                                localEndReq (16),
                                uAbortReq (17),
                                class1invokeReq (21),
                                class2invokeReq (22),
                                class3invokeReq (23),
                                class4invokeReq (24),
                                linkedInvokeReq (25),
                                resultNlReq (26),
                                resultLReq (27),
                                uErrorReq (28),
                                uCancelReq (29),
                                uRejectReq (30),
                                ...
                        }
```

-- abstract syntax name for TMP-PDUs

```
tmp-pdus-as OBJECT IDENTIFIER ::= {itu-t recommendation q 755 as(4) tmp-pdus(1) version1(1)}

END
```

ANNEX A

Example of use of the TC test responder

This annex illustrates how some of the test purposes defined in Recommendation Q.787 can be implemented using the TC Test Responder.

a) Test 2.1.6: Valid functions, User Cancel

Test Responder/TC Interface	Message flow	Test System
	BEGIN [Invoke (localConsumerOperation, 1)] <----------	
TC-Begin-Ind TC-Invoke-Ind(1)		
TC-Invoke-Req (0) TC-ContinueReq	CONTINUE[Invoke (class1SupplierOperation, 0)] ---------->	
TC-U-Cancel-Req (0)		
	CONTINUE[Return-Result-L (0)] <----------	
TC-L-Reject-Ind (0)		
TC-End-Req		
	END[Reject (0)]	

The localConsumerOperation argument is defined as follows:

```
testInit : {
        timeOut 30,
        commands
                {
                action : {service class1InvokeReq},
                action : {service continueReq},
                action : {service uCancelReq},
                wait    : {unspecified : NULL},
                action : {service basicEndReq}
                }
        }
```

b) Test 2.1.2.1.1: Valid Functions, Linked Operations, Class 1 original operation, IUT as sender

Test Responder/TC Interface	Message flow	Test System
	BEGIN [Invoke(localConsumerOperation, 1)] <----------	
TC-Begin-Ind TC-Invoke-Ind(1)		
TC-Invoke-Req (0) TC-Continue-Req		
	CONTINUE[INV(class1SupplierOperation, 0)] ---------->	
	CONTINUE[Invoke(localConsumerOperation, 2)] <----------	
TC-Continue-Ind TC-Invoke-Ind (2)		
TC-Result-L(2) TC-Continue-Req		
	CONTINUE[Return-Result-L(2)]	
	END[Return-Result-L(0)]	
TC-End-Ind TC-Result-L(0)		

The first localConsumerOperation invocation argument is defined as follows:

```
testInit : {
        timeOut 30,
        commands {
                action : {service class1InvokeReq},
                action : {service continueReq},
                wait    : {unspecified : NULL}
                 }
        }
```

The second localConsumerOperation invocation argument is defined as follows:

```
testContinue : {
                action : {service resultLReq},
                action : {service continueReq},
                wait    : {unspecified : NULL},
                 }
        }
```

c) Test 1.1.2.2.1.1-3: Valid Functions, Clearing after Continue message, IUT Abort by TR-User

Test Responder/TC Interface	Message flow	Test System
	BEGIN [Invoke(localConsumerOperation, 1)] <----------	
TC-Begin-Ind (0) TC-Invoke-Ind (1)		
TC-Begin-Req (1)		
	BEGIN ---------->	
	CONTINUE <----------	
TC-Continue-Ind (1)		
TC-U-Abort-Req(1)		
	ABORT ---------->	
TC-End-Req(local,0)		

The localConsumerOperation invocation argument is defined as follows:

```
testInit : {
        timeOut 30,
        commands {
                action : {service v1988beginReq, dialogueReference value :1},
                wait    : {dialogueReference value :1},
                action : {service uAbortReq, dialogueReference value :1},
                action : {service localEndReq, dialogueReference value : 0}
                }
        }
```

ANNEX B

Implementing loops using the TC Test Responder

This annex illustrates the use of the TC Test Responder for implementing loops. The test case is such that both the Test System and the Test Responder keep continuously opening dialogues which are immediately ended by the other entity.

Test Responder/TC Interface	Message flow	Test System
	BEGIN (OTID = X1, testInit) <----------	
TC-Begin-Ind (0)		
TC-Begin-Req (1)		
	BEGIN (OTID = Y1) ---------->	
TC-End-Req (0)	END (DTID =X1) ---------->	
	END (DTID = Y1) <----------	
	BEGIN (OTID = X2, testContinue) <----------	
TC-Begin-Ind (2)		
TC-Begin-Req (3)		
	BEGIN (OTID = Y2) ---------->	
TC-End-Req (2)	END (DTID =X2) ---------->	
	END (DTID = Y2) <----------	
...		

The first BEGIN message sent from the Tester to the Test Responder includes a testInit PDU defined as follows:

```
testInit : {
        timeOut 30,
        commands {
                action : {service v1988beginReq, dialogueReference value: 1},
                action : {service basicEndReq, dialogueReference value: 0},
                wait    : {dialogueReference value: 1}
                }
        }
```

The subsequent BEGIN messages include a testContinue PDU defined as follows:

```
testContinue : {
                action : {service v1988beginReq, dialogueReference value: i+1},
                action : {service  basicEndReq, dialogueReference  value: i},
                wait : {dialogueReference value : i+1}
                }
        }
```

where i is incremented from 1 to the number of desired loops N.

When i = N, the test case can be terminated by sending a testContinue PDU without the v1988beginReq command. In this example there exist at most two dialogues at a time. However for load testing purposes, the test case could be rewritten in such a way that the Tester does not send any END message and the TMP-PDUs (except the one which terminates the test case) do not contain any "basicEndRequest" in the list of commands.

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