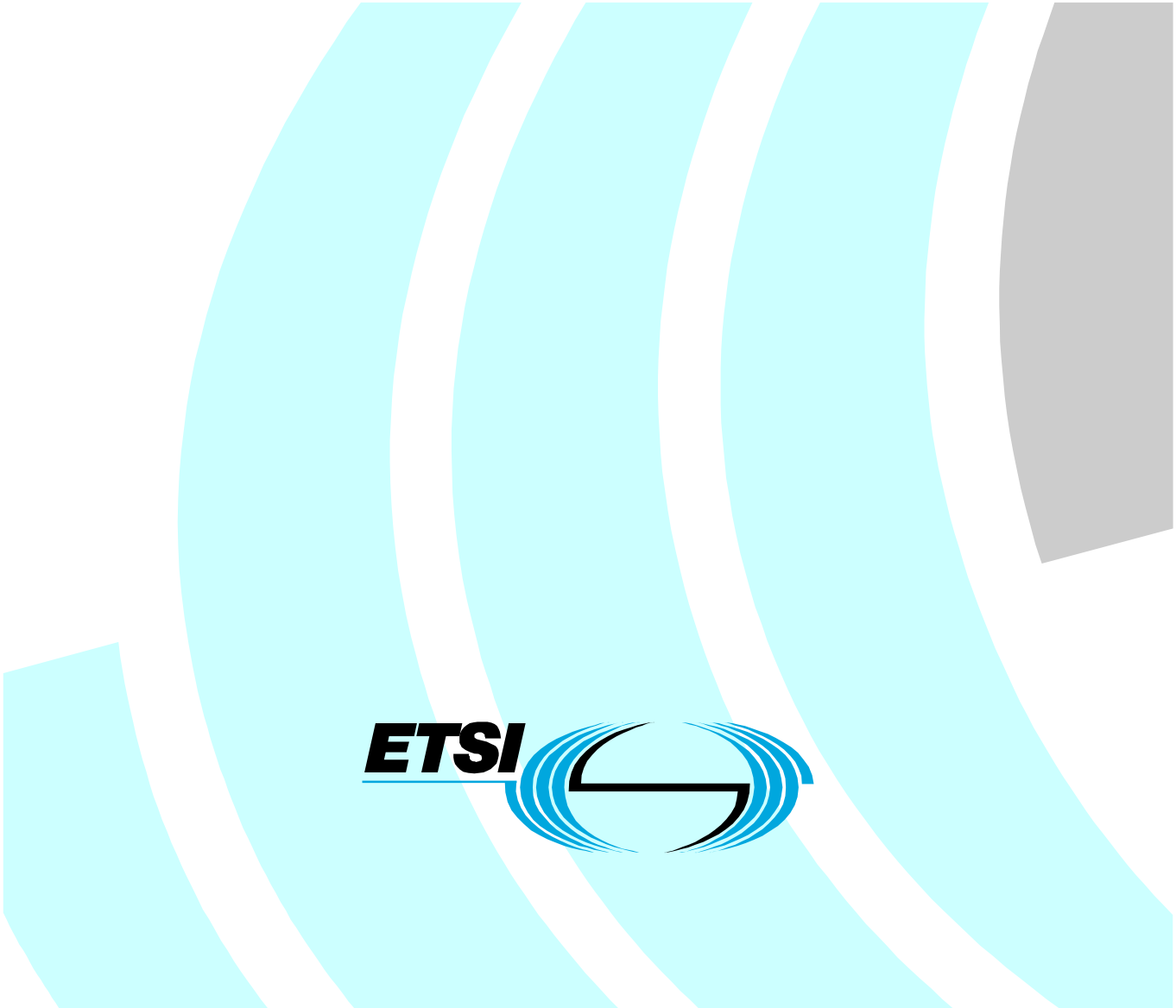
ETSI ES 202 912-2 V1.1.1 (2003-02)

ETSI Standard

Access and Terminals (AT);
Short Message Service (SMS) for PSTN/ISDN;
Test Suites for SMS User Based Solution;
Part 2: Test Suite Structure and Test Purposes (TSS&TP)
user side for Data Link Layer (DLL) Protocol 1



Reference
DES/AT-030014-02

Keywords
SMS, ISDN, PSTN, TSS&TP

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, send your comment to:
editor@etsi.org

Copyright Notification

No part may be reproduced except as authorized by written permission.
The copyright and the foregoing restriction extend to reproduction in all media.

© European Telecommunications Standards Institute 2003.
All rights reserved.

DECT™, **PLUGTESTS**™ and **UMTS**™ are Trade Marks of ETSI registered for the benefit of its Members.
TIPHON™ and the **TIPHON logo** are Trade Marks currently being registered by ETSI for the benefit of its Members.
3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

All published ETSI deliverables shall include information which directs the reader to the above source of information.

Foreword

This ETSI Standard (ES) has been produced by ETSI Technical Committee Access and Terminals (AT).

The present document is part 2 of a multi-part deliverable. Full details of the entire series can be found in part 1 [8].

1 Scope

The present document provides test suite structure and test purposes for testing Data Link layer in a Terminal Equipment implementing the Short Message Service (SMS) for PSTN/ISDN, UBS Protocol 1 according to ES 201 912 [1].

Basic ISDN or PSTN call procedures apply in order to establish a circuit-switched band connection between such Terminal Equipment and an SM-SC. Tests for these procedures are outside the scope of the present document. UBS1 terminals send and receive Data Link messages in the voice-band connection using the FSK signalling as defined in EN 300 659-2 [3] and ES 200 778-2 [6]. Tests for the FSK signalling are also outside the scope of the present document.

Terminal Equipment implementing the Short Message Service (SMS) for PSTN/ISDN according to UBS Protocol 1 are required to implement the Transfer Layer according to TS 100 901 [7]. Using the Remote Single Layer Embedded Test Method (see ISO/IEC 9646-2 [10]) for the UBS Protocol 1 Data Link layer , Transfer Layer messages appear here only as octet string parameters of Data Link layer messages. Tests for the Transfer Layer are not within the scope of the present document.

Figure 1 gives an overview of the reference architecture used for the UBS Protocol1 operation. Figure 2 shows the configuration used for testing.

ISO/IEC 9646-1 [9] and ISO/IEC 9646-2 [10] are used as the basis for the test specification methodology.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication and/or edition number or version number) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

[1] ETSI ES 201 912 (V1.1.1): "Access and Terminals (AT); Short Message Service (SMS) for PSTN/ISDN; Short Message Communication between a fixed network Short Message Terminal Equipment and a Short Message Service Centre".

[2] ETSI EN 300 659-1 (V1.3.1): "Access and Terminals (AT); Analogue access to the Public Switched Telephone Network (PSTN); Subscriber line protocol over the local loop for display (and related) services; Part 1: On-hook data transmission".

[3] ETSI EN 300 659-2 (V1.3.1): "Access and Terminals (AT); Analogue access to the Public Switched Telephone Network (PSTN); Subscriber line protocol over the local loop for display (and related) services; Part 2: Off-hook data transmission".

[4] ETSI ES 300 659-3 (V1.3.1): "Public Switched Telephone Network (PSTN); Subscriber line protocol over the local loop for display (and related) services; Part 3: Data link message and parameter coding".

[5] ETSI ES 200 778-1 (V1.3.1): "Access and Terminals (AT); Analogue access to the Public Switched Telephone Network (PSTN); Protocol over the local loop for display and related services; Terminal equipment requirements; Part 1: On-hook data transmission".

[6] ETSI ES 200 778-2 (V1.3.1): "Access and Terminals (AT); Analogue access to the Public Switched Telephone Network (PSTN); Protocol over the local loop for display and related services; Terminal equipment requirements; Part 2: Off-hook data transmission".

[7] ETSI TS 100 901 (V7.4.0): "Digital cellular telecommunications system (Phase 2+) (GSM); Technical realization of the Short Message Service (SMS) (GSM 03.40 version 7.4.0 Release 1998)".

[8] ETSI ES 202 912-1(V1.1.1): "Access and Terminals (AT); Short Message Service (SMS) for PSTN/ISDN; Test Suites for SMS User Based Solution; Part 1: Protocol Implementation Conformance Statement (PICS) proforma specification user side for Data Link Layer (DLL) Protocol 1".

[9] ISO/IEC 9646-1: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 1: General concepts".

[10] ISO/IEC 9646-2: "Information technology - Open systems interconnection - Conformance testing methodology and framework - Part 2: Abstract test suite specification".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

DL-Initiator: entity (SM-TE or SM-SC) initiating a data link connection to the peer entity by sending a DL establishing message (after the VB-connection has been established)

DL-Responder: entity (SM-TE or SM-SC) having received a DL establishing message from the peer entity (after the VB-connection has been established)

incoming VB-connection: VB-connection initiated by an SM-SC

inopportune behaviour (of the tester): the tester sends a message which is not expected by the IUT under the current circumstances (state)

invalid behaviour (of the tester): abbreviated form of "syntactically invalid behaviour", i.e. the tester sends a message where the presence or contents of one or more parameters or fields does not conform to the requirements

originator (of an SM): SM-TE sending an SM to another SM-TE

outgoing VB-connection: VB-connection initiated by an SM-TE

peer (entities): SM-TE and SM-SC for which a voice-band connection exists or is pending

SM-Call: incoming call for a SM-TE where the CLI contains the address of an SM-SC stored in the SM-TE

valid behaviour (tests): tester behaves according to the protocol

NOTE: Timeout tests normally belong to the valid tests.

VB-connection: voice-band connection between two peers

NOTE 1: A Voice-band connection is considered to be **completed** or **established**, when the basic call control procedures, performed according to the type of network access the SM-TE is connected to (i.e. PSTN or BRA ISDN or PRA ISDN), are completed and the voice-band connection is ready for FSK frame transfer.

NOTE 2: In order to establish an incoming VB-connection, the CLI information has to be previously provided to the terminal equipment. The way of providing CLI (e.g. by DTMF or FSK signalling and using a data transmission associated with ringing or not, etc.., in the case of PSTN) is out of the scope of the present document.

VB-Initiator: entity (SM-TE or SM-SC) initiating a voice-band connection to the peer entity

VB-Responder: entity (SM-TE or SM-SC) having received a voice-band connection attempt from the peer entity

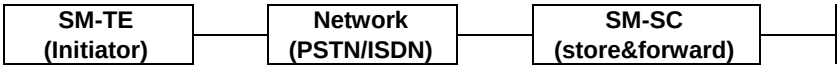
3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

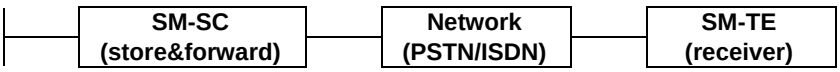
ASP	Abstract Service Primitive
ATS	Abstract Test Suite
CLI	Calling Line Identification (information)
CM	Connection Manager
DL	Data Link
DLL	Data Link Layer
DMI	Deliver Mode Identifier
DTMF	Dual Tone Multi-Frequency
FSK	Frequency Shift Keying
GSM	Global System for Mobile Communication
INV	Subgroup extension for "Invalid Behaviour" tests
ISDN	Integrated Services Digital Network
ISO	International Standard Organization
IUT	Implementation Under Test
LT	Lower Tester
LTS	Lower Test System (part of the Test System performing basic signalling procedures to establish a VB-connection and to transmit and receive FSK frames).
PDU	Protocol Data Unit
PICS	Protocol Implementation Conformance Statement
PIXIT	Protocol Implementation eXtra Information for Testing
PSTN	Public Switched Telephone Network
SM	Short Message(s)
SME	Short Message Entity
SMS	Short Message Service
SM-SC	Short Message Service Centre
SM-TE	Short Message Terminal Equipment
SUT	System Under Test
TF	Transfer (layer)
TP	Test Purpose
TSS	Test Suite Structure
TSS&TP	Test Suite Structure and Test Purposes
TTCN	Tree and Tabular Combined Notation
UBS	User Based Solution
UT	Upper Tester
VAL	Subgroup extension for "Valid Behaviour" tests
VB	Voice-band

4 Configuration assumed for the test specification

Figure 2 of ES 201 912 [1] shows the general principle of short message transfer, which consist in a SM submission phase and a SM delivery phase, which is repeated schematically in figure 1.



Submission phase



Delivery phase

Figure 1: Short Message transfer - General Principle

In the tests specified in the present document, the SM-TE will be tested as an SM originator as well as a SM receiver. No different test configurations are defined for the two different roles the SM-TE take in the two phases. The test configuration shown in figure 2 therefore reflects each of the two phases of figure 1, with some necessary additions.

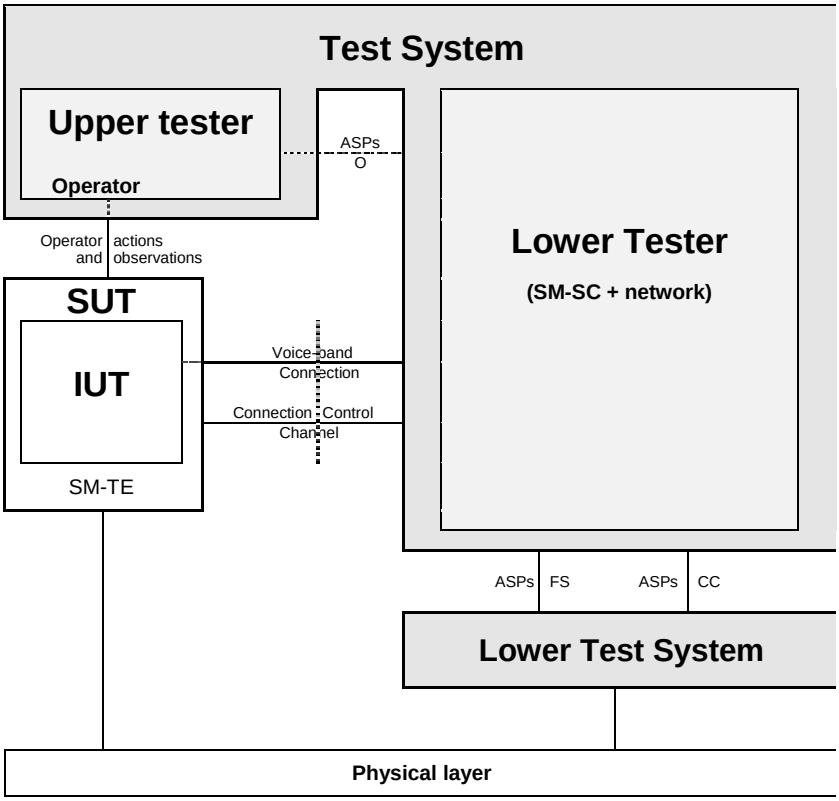


Figure 2: UBS1 test configuration

Explanations:

1) The SUT is an SM-TE.

2) The Test Method used here is the "Remote Single Layer Embedded" Test Method (see ISO/IEC 9646-1 [9]).

3) The IUT is the Data Link Layer implementation of the SM-TE.

4) The test system represents the network to which the IUT is attached, and the SM-SC. Correspondingly the SM-TE is physically connected to the test system as if it were connected to a real network.

5) The test system carries an Upper Tester and a Lower Tester. The Upper Tester is realized by an operator, controlling and observing the SUT. The test coordination between Lower Tester and the Upper Tester is performed via ASPs at PCO O.

6) Conceptually IUT and Lower Tester are connected by two channels:

   a) the voice-band channel, and

   b) the signalling control channel.

7) Virtually the Lower Tester communicates with the IUT via two PCOs:

   a) **FS**
   At this PCO the Lower Tester sends and receives FSK frames conceptually exchanged between SM-TE and SM-SC (which are transported over the voice-band channel).

   b) **CC**
   At this PCO the Lower Tester takes the actions of initiating and supervising connection control. The actions are realized by appropriate ASPs (see clause 6.3), while the actual transfer of signalling messages over the CC channel is not subject of this test specification.

   Although ringing signals are physically transported over the voice-band connection, they are conceptually associated to the CC channel in this testing scheme. This is done to keep the FS PCO free from any signals except for FSK frames conceptually exchanged between SM-TE and SM-SC.

   The same principle applies to the transfer of CLI: in the case where CLI has to be transported via FSK signalling, it is done via PCO CC. It is a matter for the test system to distinguish between these different ways of using FSK signalling.

5 Test purposes development

5.1 Introduction

A TP is defined for one or several conformance requirements to be tested. It is expected, that each TP will result in a test case keeping the same name, specified in an ATS based on the present document.

5.2 Source of test purpose specifications

The test purposes are based on the requirements made in ES 201 912 [1].

Requirements can be classified in "static" requirements and "dynamic" requirements. In the test purpose description tables of the present document, only the "dynamic" requirements of ES 201 912 [1] are referred to.

NOTE: The conformance requirements of ES 201 912 [1], "static" and "dynamic", are collected in ES 202 912-1 [8] (UBS1 PICS), but the PICS tables items are not referred to in the current document.

5.3 Restrictions and requirements not being tested

ES 201 912 [1] contains requirements for SM-TEs by direct specification and by reference to other standards. In particular, ES 201 912 [1] refers to the standards dealing with FSK signalling: EN 300 659-1 [2], EN 300 659-2 [3], EN 300 659-3 [4], ES 200 778-1 [5] and ES 200 778-2 [6].

It is not the intention of the present document to test Physical Layer aspects of FSK signalling, therefore there are no explicit tests except from those requirements related to FSK which are explicitly stated in ES 201 912 [1] (frame synchronization and delimitation, time interval requirements for frames).

According to ES 201 912 [1] SM-TEs are attached to PSTN or ISDN networks and perform basic call control procedures according to the type of the network access to establish a voice-band connection between the SM-TE and the SM-SC.

It is not within the scope of the present document to test any signalling associated with basic call control procedures. It is a matter of the test system implementing these tests to install and perform the appropriate procedures.

The initiation and supervision of these procedures is realized in the present document by specific ASPs (see clause 6.3).

According to ES 201 912 [1] the network, i.e. the tester implementing the tests specified here, has to provide the CLI of the SM-SC to the called SM-TE. When the network is a PSTN, CLI can be provided by DTMF signalling or by FSK signalling. It is also a matter of the test system to implement and use the method appropriate for the SM-TE and the network access type.

Only requirements that can be verified by inspection of the UBS Protocol 1 Data Link layer messages transferred over the voice-band channel are related to test purposes, in coordination with the ASPs necessary for terminal control via an operator and ASPs controlling voice-band connections, CLI provision and ringing.

TF layer contents of DLL messages are provided as Test Parameters of octet string type, i.e. it is a matter of parametrization of the executable test suite to provide valid TF layer contents for testing the DLL.

5.4 Grouping of test purposes

ES 201 912 [1] specifies requirements for frame synchronization, delimitation and time intervals between frames, as well as connection procedures at the DLL, which may be initiated by either side. The tests are grouped therefore in two main categories:

1) FSK Frame transfer related tests, and

2) DLL-connection-related tests.

The DLL-connection-related tests have been grouped into those for outgoing and for incoming connections. Within these groups, subgroups have been defined for the connection phases "establishment", "data transfer" and "release".

For each such subgroup "valid" tests have been specified. "Invalid" and "inopportune" tests have also been specified when appropriate.

Detailed information on the test groups and subgroups can be found in clause 6.2.

5.5 Test purpose naming convention

The current document is created together with other similar TSS&TP documents, dealing also with UBS Protocol 2 (UBS2) instead of UBS Protocol 1 (UBS1) and TL instead of DLL (see ES 201 912 [1]).

Therefore the Test purpose naming convention refers also to protocol and layer.

TP names are composed as follows:

Identifier := <Protocol>_<Layer>_<group structure>_<nn>

Where:

<Protocol> = **UBS1**, <Layer> = **DLL** and <nn> is a 2-digit sequential number, starting from **01** for each subgroup.

<group structure> is composed of sub-identifiers according to the subgroup structure of the group.

EXAMPLE: UBS1_DLL_OUT_EST_VAL_01

In this case **OUT** refers to group "Outgoing call", **EST** represents subgroup "establishment" and **VAL** refers to subgroup "Valid tests".

For details on groups and subgroups see table 1.

5.6 Method used for developing test purposes

Test purposes have only been developed for behaviour of the IUT which is visible at the FSK signalling interface. Following this basic principle, the development of the test purposes has been performed according to the following actions:

1) Using the PICS in ES 202 912-1 [8], identify all clauses of ES 201 912 [1] containing "dynamics" requirements and identify these requirements.

2) Determine the requirements not to be tested and identify them (see clause 5.3).

3) Define the test purposes structure (see clause 6.2).

4) Regarding that the structure is based on the DLL procedures defined in ES 201 912 [1] and that valid, invalid and inopportune test purposes have to be defined: identify the general requirements on invalid or inopportune messages and take them into account for all procedures, as far as applicable.

5) The MSCs in annex A of ES 201 912 [1] are taken into account.

6) The method chosen for the presentation of the test purposes is described in clause 5.7. The method should enable an easy and systematic transition from the test purposes to a TTCN ATS.

7) When all test purposes have been defined, the "requirements references" are collected and checked with the list of all testable requirements, to ensure a suitable coverage.

5.7 Method used for test purpose description

A TP is described using a table as shown in the following example. The item names appearing in the left column of the example table are present in each TP table.

The right column of the example table contains *descriptive text* and example entries. The descriptive text is in italics, while example entries are in **bold text**.

The rows "Purpose", "Requirem. Ref." and "Selection Expr." contain normative contents, the rows "Preamble", "Test description", "Pass criteria" and "Postamble" contain just informative contents (the same information, expressed in the TTCN formalism, appear as normative in the .GR file related to the ATS document correspondent to this TSS&TP document).

	Test Purpose identifier like **UBS1_DLL_OUT_EST_VAL_01**
Purpose:	*<Textual description of the purpose to be achieved>*
Requirem. ref.:	*Reference to one or more clauses of ES 201 912 [1], containing a requirement which is tested in connection with the current test purpose. References to standards other than ES 201 912 [1] are added if appropriate. Since ES 201 912 [1] is the basic standard on which this test specification relies, the first paragraph of the "Requirem. ref." cell contains only clause numbers, without identifying the standard explicitly. Whenever a reference to another standards are necessary, a new paragraph is added, identifying the standard, followed by a colon and then clause numbers as for ES 201 912 [1].* **Example:** 5.3.1, A.1.1 EN 300 659-2 [3]: 5.1
Selection Expr.:	*When an entry is present here, it denotes a selection expression (see clause 6.6), specifying a* ***condition for the applicability*** *of this test purpose. If there is no entry, the test purpose is unconditionally applicable.* *Note that selection expressions can also be defined for a* ***whole group of test purposes****, i.e. outside a test purpose table.*
Preamble:	*Name of a preamble leading to a condition or state, where the test purpose can be verified.*
Test description:	*Sequence of events intending to lead to the verification of the test purpose.* *A TTCN-like notation is used (see clause 6.5).* *Note that unexpected events are not shown here.*
Pass criteria:	*Special indication of an event (or several events), an "ignore"-behaviour or specific received parameter value(s), being essential for the verification of the test purpose.* *This can be a copy of one or more lines of the "Test description", or can be a textual explanation.*
Postamble:	**None** *or name of a postamble leading to the IDLE condition of the IUT (no VB connection exists).*

6 Test purposes presentation and environment specification

6.1 Introduction

After the definition of the **Test Suite Structure**, a suitable environment has to be created in order to formulate the test purpose descriptions properly (i.e. referring to elements of this environment). Apart from this, the environment is specified to support a **systematic transition** from the test purposes to the TTCN ATS. This is e.g. taken into account in the naming conventions and "atomic phrases" used in the descriptions. "**Test parameters**" and other objects have been collected during the test purposes development, which will find their entry in the ATS, with few modifications, e.g. as **TS Parameters**.

6.2 Test Suite Structure (TSS)

The table 1 shows the structure of the UBS1 Data Link Layer Test Suite, as well as the TP group identifiers and the number of test purposes produced, per subgroup and in total.

Table 1: Test suite structure for testing the UBS1 layer 2 behaviour

Procedure/Group	Subgroup	Subgroup	Group identifier	Count
Frame transfer	Synchronization	Valid	UBS1_DLL_FRM_SYNC_VAL	2
	Timing intervals	Valid	UBS1_DLL_FRM_TIM_VAL	8
Outgoing call	Establishment	Valid	UBS1_DLL_OUT_EST_VAL	2
		Invalid	UBS1_DLL_OUT_EST_INV	4
	Data transfer	Valid	UBS1_DLL_OUT_DAT_VAL	6
		Invalid	UBS1_DLL_OUT_DAT_INV	9
	Release	Valid	UBS1_DLL_OUT_REL_VAL	5
Incoming call	Establishment	Valid	UBS1_DLL_INC_EST_VAL	1
		Invalid	UBS1_DLL_INC_EST_INV	2
	Data transfer	Valid	UBS1_DLL_INC_DAT_VAL	3
		Invalid	UBS1_DLL_INC_DAT_INV	7
	Release	Valid	UBS1_DLL_INC_REL_VAL	5
Total:				**54**

6.3 Abstract Service Primitives

Three classes of ASPs are defined, according to the **PCO**s at which they operate (see figure 2):

PCO: **ASP class,**

O: ASPs related to the operator (Upper Tester),

FS: ASPs for SM-related to the transmission and reception of FSK frames, and

CC: ASPs related to connection control.

Table 2 describes the general functions of the ASPs operating at the 3 PCOs. Detailed descriptions of the ASPs together with their parameters follow.

Table 2: List of ASPs

PCO	ASP Name	Direction	Description
O	OUTGOING_CALL_req	LT->UT	Make the SM-TE initiate a VB connection to the SM-SC in order to send an SM.
CC	INC_CALL_req	LT->LTS	Make the Lower Test System initiate the basic call control procedures for an incoming call, to establish a VB connection from the SM-SC to the SM-TE.
	INC_CALL_conf	LTS->LT	The Lower Test System confirms that the requested basic call control procedures for an incoming call have been successfully completed and the VB connection to the SM-TE is established, or that the connection attempt was not successful.
	OUTG_CALL_ind	LTS->LT	Indication that the SM-TE has initiated the basic call control procedures for an outgoing call.
	OUTG_CALL_resp	LT->LTS	Response to the Lower Test System that the basic call control procedures for an outgoing call have to be completed or refused.
	CALL_RELEASE_ind	LTS->LT	Indication from the Lower Test System that the VB connection has been released by the SM-TE.
	CALL_RELEASE_req	LT->LTS	Request the Lower Test System to release the VB connection.
FS	TRANSFER_req	LT->LTS	Request the Lower Test System to transfer an FSK frame to the IUT on the established VB connection.
	TRANSFER_ind	LTS ->LT	Indication that the IUT has sent an FSK frame to the tester on the established VB connection.

Tables 3 to 11 contain the descriptions of the ASPs used in the present document, including the ASP parameters (if any) and the kinds of values these may assume. No ASP parameter is optional.

Default values are indicated for most of the ASP parameters. The meaning of "Default value" is:

If the ASP is applied in a TP behaviour description and no value is indicated explicitly for an ASP parameter, then the application of the default value is assumed. Whenever a value is explicitly indicated for an ASP parameter, this value replaces the default value (at this instance of ASP application).

Table 3: OUTGOING_CALL_req ASP and its parameters

ASP Name: OUTGOING_CALL_req **PCO:** O **Direction:** LT->UT **Description:** Make the SM-TE initiate a VB connection to the SM-SC in order to send an SM (see note 1).		
Parameter	**Default value**	**Description**
SCADDR	TSPX_SC_ADDR	Address of the SM-SC to be called.
SMEID	TSPX_SME_ID	Subaddress of the calling SME.
LENIND	0	Length Indicator **0**: the length of the Payload (binary TL message) to be transferred is at most 176 octets long. **1**: the length of the Payload to be transferred is more than 176 octets long. (see note 2)
TFNUM	1	Number of SMs to be submitted during the same VB connection.
Comments: NOTE 1: For the purposes of the DLL tests the SM(s) to be submitted does not matter. NOTE 2: The extension mechanism is currently not used, since there are no valid TL messages with length > 176. This means that the ASP is currently only used with LENIND = 0.		

Table 4: INC_CALL_req ASP and its parameters

ASP Name: INC_CALL_req **PCO**: CC **Direction**: LT->LTS **Description**: Make the Lower Test System initiate the basic call control procedures for an incoming call, to establish a VB connection from the SM-SC to the SM-TE.		
Parameter	**Default value**	**Description**
CLDTE	TSPX_CLD_TE	Address of the SM-TE to be called.
SCADDR	TSPX_SC_ADDR	Address of the calling SM-SC.
SMEID	TSPX_SME_ID	Subaddress of the called SME.
DMI	0	Deliver Mode Identifier
Comments:		

Table 5: INC_CALL_conf ASP and its parameters

ASP Name: INC_CALL_conf **PCO**: CC **Direction**: LTS->LT **Description**: The Lower Test System confirms that the requested basic call control procedures for an incoming call have been successfully completed and the VB connection to the SM-TE is established, or that the connection attempt was not successful.		
Parameter	**Default value**	**Description**
ESTCONF	TRUE	**TRUE** if the requested VB connection to the SM-TE/SME has been successfully established. Otherwise **FALSE**.
Comments:		

Table 6: OUTG_CALL_ind ASP and its parameters

ASP Name: OUTG_CALL_ind **PCO**: CC **Direction**: LTS->LT **Description**: Indication that the SM-TE has initiated the basic call control procedures for an outgoing call.		
Parameter	**Default value**	**Description**
SCADDR	TSPX_SC_ADDR	Address of the called SM-SC.
SMEID	TSPX_SME_ID	Subaddress of the calling SME.
Comments:		

Table 7: OUTG_CALL_resp ASP and its parameters

ASP Name: OUTG_CALL_resp **PCO**: CC **Direction**: LT->LTS **Description**: Response to the Lower Test System that the basic call control procedures for an outgoing call have to be completed or refused.		
Parameter	**Default value**	**Description**
ESTRESP	TRUE	TRUE if the outgoing call is to be accepted, otherwise FALSE.
Comments:		

Table 8: CALL_RELEASE_ind ASP and its parameters

ASP Name: CALL_RELEASE_ind **PCO**: CC **Direction**: LTS->LT **Description**: Indication from the Lower Test System that the VB connection has been released by the SM-TE.		
Parameter	**Default value**	**Description**
Comments: No confirmation is issued by the tester. It is a matter of the test system to complete the release procedures. The LTS shall not issue this ASP when the tester has issued a CALL_RELEASE_req before on the same VB connection.		

Table 9: CALL_RELEASE_req ASP and its parameters

ASP Name: CALL_RELEASE_req **PCO**: CC **Direction**: LT->LTS **Description**: Request the Lower Test System to release the VB connection to the SM-TE.		
Parameter	**Default value**	**Description**
Comments: No confirmation is expected by the LT from the LTS. The LTS shall complete the release procedures when receiving this ASP. When this ASP is received by the LTS after VB release has been signalled by the SM-TE (release collisison), the LTS shall continue the VB release as if no CALL_RELEASE_req was received.		

Table 10: TRANSFER_req ASP and its parameters

ASP Name: TRANSFER_req **PCO**: FS **Direction**: LT->LTS **Description**: Request the Lower Test System to transfer an FSK frame to the IUT on the established VB connection.		
Parameter	**Default value**	**Description**
MS	80	Number of mark bits that shall be transferred by the Lower Test System in front of the DLL PDU (Mark signal).
PDU	-	DLL message to be transmitted, as specified in clause 6.4.1.
CS	TRUE	If TRUE, the PDU is transferred as an FSK frame with the correct checksum appended, as defined in clause 5.3.2.1 of ES 201 912 [1]. If FALSE, a checksum error is generated by the Lower Test System.
Comments:		

Table 11: TRANSFER_ind ASP and its parameters

ASP Name: TRANSFER_ind **PCO**: FS **Direction**: LTS ->LT **Description**: Indication that the IUT has sent a correct FSK frame to the tester on the established VB connection.		
Parameter	**Default value**	**Description**
PDU	-	Received DLL message as specified in clause 6.4.1. (see note)
Comments: NOTE: A received FSK frame is only passed to the LT by the LTS if the received "Mark signal" is in the range of 80 ± 25 mark bits, if the checksum is correct and if the contents of the "Message length" octet is consistent with the actual length of the received message (i.e. the length of the "Payload"). If an FSK frame is received by the LTS which is not correct with respect to one or more of these features, this will lead to a timeout in the test, because the message is not received by the LT. It is a matter of the test system to indicate errors related to these features.		

6.4 UBS1 DLL messages (PDUs)

6.4.1 List of UBS1 DLL messages

Table 12 lists the UBS1 DLL messages used in the present document:

Table 12: List of UBS1 Layer 2 messages

Message name	Purpose description
DLL_SMS_EST	Indicates that the Data Link Layer connection has been established.
DLL_SMS_DATA	Carries SM data, i.e. a TL message.
DLL_SMS_ACK	Carries a positive SM acknowledgement, i.e. a TL message.
DLL_SMS_NACK	Carries a negative SM acknowledgement, i.e. a TL message indicating an error.
DLL_SMS_REL	Indicates the release of the Data Link Layer connection.
DLL_SMS_ERROR	Indicates that a Data Link Layer error has occurred. An error cause is contained in the message.
DLL_SMS_UNKNOWN	Unknown message, i.e. a message not being defined in ES 201 912 [1]. This message is only be transmitted by the tester in order to create an error.
DLL_SMS_DATA_LONG	Message with the same type as DLL_SMS_DATA, but where the payload is longer than allowed (more than 176 octets). This message is only be transmitted by the tester in order to create an error.

6.4.2 How to interpret message parameters and their values

Table 13 shows the fields are defined for DLL messages.

Table 13: DLL message fields

Octet(s)	Subfield	Short field name	Description
Message Type octet	Extension bit	E	1 bit (most significant bit in Message Type octet). Used for DLL extension mechanism (segmentation and re-assembly of Payloads longer than 176 octets).
	Message type field	MT	7 bits (least significant bits in Message Type octet). Identifies the type of the DLL message.
Message length	-	ML	Length of Payload (in octets)
Payload	-	PL	Octet string of maximum length 176; optional, i.e. not present in all DLL messages

The short field names in table 13, e.g. **MT**, are the abbreviated field names as they appear in DLL messages occurring in test behaviour descriptions, when appropriate.

NOTE 1: Clause 5.3.2 of ES 201 912 [1] specifies also the "Mark signal" and the "Checksum". Both message elements do not appear in the message structure definition used in the present document, but are treated in the ASPs used for DLL message transmission and reception.

The following principle applies for messages to be transmitted by the tester:

- The **Message type field** is not shown explicitly, because it is identified by the message name.
- If the **Extension bit** is not shown explicitly, it is understood that the bit value "1" (no segmentation) is sent.
- The **Message length** is not shown explicitly, the contents of the Message length octet is the binary coding of the number of octets in the payload.
- If the **Payload** is not shown explicitly, it is omitted. If the Payload is shown explicitly, it refers to a Test Parameter (see table 14 on page 19) to be transmitted.

The following principle applies for messages to be received from the IUT:

- The **Message type field** is not shown explicitly, because it is identified by the message name.
- If the **Extension bit** is not shown explicitly, it is understood that the bit value "1" (no segmentation) has to be received.
- The **Message length** is not shown explicitly. The contents of the received Message length octet has to be the binary coding of the number of octets in the payload (otherwise the message is not passed to the LT by the LTS).
- If the **Payload** is not shown explicitly, it is assumed that no Payload has to be received in this message. If the Payload is shown explicitly, it either refers to a Test Parameter (see table 14 on page 19), or it explicitly indicates "OMIT", or it indicates one of the wildcards **Any** (?) or **AnyOrOmit** (*).

NOTE 2: The wildcards Any and AnyOrOmit and their symbols **?** and ***** have been borrowed from TTCN and have the same semantics:
? means that any non-empty value compatible with the type of the referred field is received,
***** means that either nothing is received, or any non-empty value compatible with the type of the referred field is received.

EXAMPLES:

!DLL_SMS_EST

A correct DLL_SMS_EST message is sent, without payload

!DLL_SMS_DATA (PL=TSPX_SMS_DELIVER_S1)

A syntactically correct DLL_SMS_DATA message is sent, with the Payload represented by Test Parameter 'TSPX_SMS_DELIVER_S1', which contains a default TL SMS_DELIVERY message.

?DLL_SMS_DATA (E=1, PL=?)

A syntactically correct DLL_SMS_DATA message is received, where the extension bit is set to "1" (i.e. "no segmentation"), and the Payload is any octet string of length up to 176 octets.

6.5 Behaviour notation

This clause describes the principles used when filling the "**Test description**" entry of the TP tables (see sample in clause 5.7) and the behaviour notation of preambles and postambles.

The notation used to describe the trees containing the required operations and signalling control primitives, is a TTCN-like notation, showing what is sent (character !) and received (character ?) by the tester (playing the role of the SM-SC).

ASPs are sent as shown in the following TTCN-like form:

O!OUTGOING_CALL_req

where **O** denotes the PCO used with the ASP, **!** denotes "transmission" and the **ASP name** follows.

Since default values have been defined for the ASP parameters, parameter values are only indicated when they differ from the default value.

EXAMPLE 1:

O!OUTGOING_CALL_req(TFNUM=2)

In this case 2 SMs are requested to be transmitted by the SM-TE during the VB connection (instead of default 1 SM to be transmitted).

The same principle applies for received ASPs, the symbol **!** being replaced by **?**.

EXAMPLE 2:

CC?OUTG_CALL_ind

In this case the LT receives an indication from the LTS that there is an incoming call from the SM-TE (being requested at the SM-TE as in example 1).

There is one important exception for the presentation of ASPs:

To increase the readability, ASP names **TRANSFER_req** and **TRANSFER_ind** are not shown, except when a DLL message is to be transmitted with an error in the "Mark signal" or the "Checksum".

DLL message names are directly used instead for transmission and reception. For examples see clause 6.4.2.

When the tester expects a reaction from the IUT following the transmission of a PDU or ASP, the start of an "**acknowledgement timer**" is assumed, but normally not indicated in the test description.

Notes are put into the behaviour descriptions whenever it appears to be necessary.

Preambles and Postambles:

The description of Preambles and Postamble starts with an **Objective** definition.

The behaviour description ends with the preamble or postamble name. Between start and end the notation is as described above for "**Test description**".

Each preamble shows the state from where it starts (idle or a different state reached by the execution of another preamble), then it shows the operations executed in this preamble and finally the state or configuration reached, using the notation described above.

Each test purpose description shows the state from where it starts by identifying a preamble.

6.6 Parametrization and selection

NOTE: During the TSS&TP development Test Parameters have been collected in table 14 and 15. Test Parameter names starting with "TSPX" are used for test parametrization and will correspond to PIXIT items, TS Parameter names starting with "TSPC" are used for selection and normally correspond to PICS items. Only Test Parameters referred to in the TP description tables appear here. It is assumed that the Test Parameters defined here will be transformed into TTCN "Test Suite Parameters" in an ATS basing on this TSS&TP, presumably in a one-to-one fashion.

Table 14 shows the Test Parameters that are necessary to parameterize the test descriptions to the necessary extent. They will normally appear as ASP parameter values or PDU field contents. It is also possible that they appear in [] brackets as qualifiers for different branches of behaviour (see clause 6.5).

Table 14 specifies the Test Parameter **name**, its **type** (normally a string, integer or Boolean type) and the **explanation** what it is used for.

Table 14: Test Parameters used for parametrization (associated with PIXIT items)

Test Parameter name	Type	Description
TSPX_CLD_TE	OCTETSTRING	Address of the destination SM-TE to be called by the SUT.
TSPX_SC_ADDR	OCTETSTRING	Address of the SM-SC to be called by the SUT and stored in the SUT.
TSPX_SME_ID	OCTETSTRING	Subaddress of an SME implemented in the SUT (referred to as SME1). This is the default SME subaddress.
TSPX_SMS_DELIVER_S1	OCTETSTRING	SMS_DELIVER TL message that is used as "default" SM to be delivered, and is transmitted as Payload in a DLL message.
TSPX_SMS_DELIVER_S2	OCTETSTRING	SMS_DELIVER TL message that is used as second message to be delivered on the same VB connection, and is transmitted as Payload in a DLL message.
TSPX_SMS_DELIVER_S3	OCTETSTRING	SMS_DELIVER TL message containing a TL error, to be delivered to the IUT. The message is expected to provoke a DLL_SMS_NACK message from the IUT.
TSPX_SMS_SUBMIT_REP_ACK	OCTETSTRING	SMS_SUBMIT_REPORT TL message that can be sent by the tester in a DLL_SMS_ACK message as a positive response upon a SM submitted by the SM-TE.
TSPX_SMS_SUBMIT_REP_NACK	OCTETSTRING	SMS_SUBMIT_REPORT TL message that can be sent by the tester in a DLL_SMS_NACK message as a negative response upon a SM submitted by the SM-TE.
TSPX_UNKNOWN_PL	OCTETSTRING	Payload to be transmitted in the DLL_SMS_UNKNOWN message.
TSPX_TOO_LONG_PL	OCTETSTRING	Payload longer than 176 octets, to be transmitted in the DLL_SMS_DATA_LONG message.
Comments:		

Table 15 shows the Test Parameters that are necessary to formulate the selection conditions as BOOLEAN expressions. Table 15 specifies the Test Parameter name (Boolean type), and the explanation when it is TRUE or FALSE. These Test Parameters normally correspond to optional PICS items.

Table 15: Test Parameters used for selection

Test Parameter name	Description
TSPC_MORE_SMS_TX	TRUE if the SUT supports the possibility to send more than one SM within the same VB connection. Otherwise FALSE.
Comments:	

Table 16 shows the selection conditions formulated as BOOLEAN expressions. Table16 specifies the **Selection expression ID** or **name**, the BOOLEAN Expression, and a verbose description when a TP carrying this Selection Expressionit is applicable for execution with an IUT or not. In its simplest form, the BOOLEAN Expression just refers to a (BOOLEAN) Test Parameter associated with a PICS item. Combinations using BOOLEAN operators like **AND**, **OR** or **NOT** are also allowed.

Table 16: Selection expressions

Selection expression ID	Expression	Description
SelMoreSMs_TX	TSPC_MORE_SMS_TX	Selects a TP for testing if the SUT supports the possibility to send more than one SM within the same VB connection.
Comments:		

6.7 States

IUT States names are occasionally and informally used as in annex A of ES 201 912 [1].

6.8 Preambles

6.8.1 PRE_INIT

Objective: The preamble has the formal objective to initialize the terminal and test equipment before attempting the next VB connection and DLL connection. A particular behaviour is not associated with this preamble. It is assumed however, that at the end of this preamble no VB connection exists.

6.8.2 PRE_CONN_OUTG

Objective: Establish an outgoing VB connection to the SM-SC. Formal parameters are the Length Indicator requiring an unsegmented payload (TL message) (LengthInd) and the number of SMs to be transmitted during the VB connection (SmsNum).

NOTE: The extension mechanism is currently not used, since there are no valid TL messages with length > 176. This means that the ASP is currently only used with LENIND = 0.

PRE_CONN_OUTG(LengthInd, SmsNum)
+PRE_INIT
O!OUTGOING_CALL_req(LengthInd, SmsNum)
CC?OUTG_CALL_ind
CC!OUTG_CALL_resp
PRE_CONN_OUTG

6.8.3 PRE_CONN_OUTG_EST

Objective: Establish an outgoing VB connection to the SM-SC, followed by transmission of a DLL_SMS_EST message by the tester (the IUT is in state PENDING). Formal parameters are the Length Indicator requiring unsegmented payload (TL message) to be sent (LengthInd) and the number of SMs to be transmitted during the VB connection (SmsNum).

NOTE: The extension mechanism is currently not used, since there are no valid TL messages with length > 176. This means that the ASP is currently only used with LENIND = 0.

PRE_CONN_OUTG_EST(LengthInd, SmsNum)
+PRE_CONN_OUTG(LengthInd, SmsNum)
Wait T10+DELTA%
FS!DLL_SMS_EST
PRE_CONN_OUTG_EST

6.8.4 PRE_CONN_OUTG_DATA

Objective: Establish an outgoing VB connection to the SM-SC, followed by transmission of a DLL_SMS_EST message by the tester and a DLL_SMS_DATA message by the IUT (the IUT is in state RECEIVE). Formal parameters are the Length Indicator requiring unsegmented payload (TL message) to be sent (LengthInd) and the number of SMs to be transmitted during the VB connection (SmsNum).

NOTE: The extension mechanism is currently not used, since there are no valid TL messages with length > 176. This means that the ASP is currently only used with LENIND = 0.

PRE_CONN_OUTG_DATA(LengthInd, SmsNum)
+PRE_CONN_OUTG_EST(LengthInd, SmsNum)
  FS?DLL_SMS_DATA(E=1, PL=?)
PRE_CONN_OUTG_DATA

6.8.5 PRE_CONN_INC

Objective: Establish an incoming VB connection to the SM-TE. The Formal Parameter is the "Deliver Mode Identifier" (DelModeId) passed to the called SM-TE via CLI.

PRE_CONN_INC(DelModeId)
+PRE_INIT
CC!INC_CALL_req(DelModeId)
CC?INC_CALL_conf
PRE_CONN_INC

6.8.6 PRE_CONN_INC_EST

Objective: Establish an incoming VB connection to the SM-TE, and receive a DLL_SMS_EST message sent by the IUT (the IUT is in state RECEIVE)).

PRE_CONN_INC_EST(DelModeId)
+PRE_CONN_INC(DelModeId)
FS?DLL_SMS_EST
PRE_CONN_INC_EST

6.8.7 PRE_CONN_INC_DATA

Objective: Establish an incoming VB connection to the SM-TE, receive a DLL_SMS_EST message from the IUT and send a (first) DLL_SMS_DATA message (the IUT is in state SEND). Formal parameters are the "Deliver Mode Identifier" (DelModeId) passed to the called SM-TE via CLI, the extension bit to be used with the transmitted DLL_SMS_DATA message, and the Payload to be sent (Payload).

NOTE: The extension mechanism is currently not used, since there are no valid TL messages with length > 176. This means that the value of the extension bit is Ext = 1.

PRE_CONN_INC_DATA(DelModeId, Ext, Payload)
+PRE_CONN_INC_EST(DelModeId)
FS!DLL_SMS_DATA(Ext, PL=Payload)
PRE_CONN_INC_DATA

6.8.8 PRE_CONN_INC_DATA_ACK

Objective: Establish an incoming VB connection to the SM-TE, receive a DLL_SMS_EST message from the IUT, send a DLL_SMS_DATA message by the tester and receive a DLL_SMS_ACK message (positive acknowledgement) from the IUT (the IUT is in state RECEIVE). Formal parameters are the "Deliver Mode Identifier" (DelModeId) passed to the called SM-TE via CLI, the extension bit to be used with the transmitted DLL_SMS_DATA message, and the Payload to be sent (Payload).

NOTE: The extension mechanism is currently not used, since there are no valid TL messages with length > 176. This means that the value of the extension bit is Ext = 1.

PRE_CONN_INC_DATA_ACK(DelModeId, Ext, Payload)
+PRE_CONN_INC_DATA(DelModeId, Ext, Payload)
FS?DLL_SMS_ACK(PL=?)
PRE_CONN_INC_DATA_ACK

6.9 Postambles

6.9.1 General

The following postambles are used to ensure that all connections (DLL and VB) are released at the end of a TP. In some of the postambles the IUT is the initiator and in the other postambles the tester is the initiator (of the release).

In case the IUT does not release when it is expected to do so, it is assumed that the tester will ultimately initiate the release. However this is not shown.

In some cases the release can already occur in the TP body. In this case the use of the postamble is not intended to repeat release signalling.

6.9.2 POST_TESTER_RELEASE_VB

Objective: Release of the VB connection by the tester, independently of what the status of the DLL connection is and who is the originator of the call.

POST_TESTER_RELEASE_VB
CC!CALL_RELEASE_req
POST_TESTER_RELEASE_VB

6.9.3 POST_IUT_RELEASE_VB

Objective: Release of the VB connection by the IUT.

POST_IUT_RELEASE_VB
CC?CALL_RELEASE_ind
POST_IUT_RELEASE_VB

6.9.4 POST_TESTER_RELEASE_ALL

Objective: Release of the DLL connection and then the VB connection by the tester.

POST_TESTER_RELEASE_ALL
FS!DLL_SMS_REL
CC!CALL_RELEASE_req
POST_TESTER_RELEASE_ALL

6.9.5 POST_IUT_RELEASE_ALL

Objective: Release of the DLL connection and then the VB connection by the IUT.

POST_IUT_RELEASE_ALL
FS?DLL_SMS_REL
CC?CALL_RELEASE_ind
POST_IUT_RELEASE_ALL

7 Test purpose descriptions

7.1 Test purposes for Frame transfer/Synchronization (UBS1_DLL_FRM_SYNC)

	UBS1_DLL_FRM_SYNC_VAL_01
Purpose:	Verify that the SM-TE correctly receives a DLL message with a Mark Signal length of 55 Mark bits.
Requirements refs:	5.3.2.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(MS=55, PDU=DLL_SMS_EST, CS=TRUE) FS?DLL_SMS_DATA(E=1, PL=?)
Pass criteria:	DLL_SMS_DATA received, indicating correct receipt of DLL_SMS_EST
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_FRM_SYNC_VAL_02
Purpose:	Verify that the SM-TE correctly receives a DLL message with a Mark Signal length of 105 Mark bits.
Requirements refs:	5.3.2.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(MS=105, PDU=DLL_SMS_EST, CS=TRUE) FS?DLL_SMS_DATA(E=1, PL=?)
Pass criteria:	DLL_SMS_DATA received, indicating correct receipt of DLL_SMS_EST
Postamble:	POST_TESTER_RELEASE_ALL

7.2 Test purposes for Frame transfer/ Timing intervals (UBS1_DLL_FRM_TIM)

	UBS1_DLL_FRM_TIM_VAL_01
Purpose:	Verify that the SM-TE sends a DLL_SMS_EST message not before T10min starting from the VB connection establishment completion.
Requirements refs:	5.3.1, 5.3.2.1, table 1
Selection Cond.:	
Preamble:	PRE_CONN_INC(DelModeId=0)
Test description:	Start T10min –TDELTA Timeout T10min FS?DLL_SMS_EST
Pass criteria:	DLL_SMS_EST received after timeout T10min
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_FRM_TIM_VAL_02
Purpose:	Verify that the SM-TE sends a DLL_SMS_DATA (SMS_SUBMIT) message not before T11min starting from the reception of a DLL_SMS_EST message.
Requirements refs:	5.3.1, 5.3.2.1, table 1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_EST(LengthInd=0, SmsNum=1)
Test description:	Start T11min – TDELTA Timeout T11min FS?DLL_SMS_DATA(E=1, PL=?)
Pass criteria:	DLL_SMS_DATA received after timeout T11min
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_FRM_TIM_VAL_03
Purpose:	Verify that the SM-TE sends a DLL_SMS_ACK (SMS_DELIVER_REPORT) message not before T11min starting from the reception of a correct DLL_SMS_DATA (SMS_DELIVER) message.
Requirements refs:	5.3.1, 5.3.2.1, table 1
Selection Cond.:	
Preamble:	PRE_CONN_INC_DATA(DelModeId=0, '1'B, TSPX_SMS_DELIVER_S1)
Test description:	Start T11min – TDELTA Timeout T11min FS?DLL_SMS_ACK(E=1, PL=?)
Pass criteria:	DLL_SMS_DATA received after timeout T11min
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_FRM_TIM_VAL_04
Purpose:	Verify that the SM-TE sends a DLL_SMS_NACK (SMS_DELIVER_REPORT) message not before T11min starting from the reception of a DLL_SMS_DATA (SMS_DELIVER) message with TL error.
Requirements refs:	5.3.1, 5.3.2.1, table 1
Selection Cond.:	
Preamble:	PRE_CONN_INC_DATA(DelModeId=0, '1'B, TSPX_SMS_DELIVER_S3)
Test description:	Start T11min – TDELTA Timeout T11min FS?DLL_SMS_NACK(E=1, PL=?)
Pass criteria:	DLL_SMS_NACK received after timeout T11min
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_FRM_TIM_VAL_05
Purpose:	Verify that the SM-TE sends a DLL_SMS_ERROR message not before T11min starting from the reception of a DLL_SMS_DATA (SMS_DELIVER) message with a DLL error.
Requirements refs:	5.3.1, 5.3.2.1, table 1, 5.3.2.2,, table 2
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST(DelModeId=0)
Test description:	FS!TRANSFER_req(MS=80, PDU=DLL_SMS_DATA(PL=TSPX_SMS_DELIVER_S1), CS=FALSE) Start T11min – TDELTA Timeout T11min FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	DLL_SMS_ERROR(PL='Wrong checksum') received after timeout T11min
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_FRM_TIM_VAL_06
Purpose:	Verify that the SM-TE sends a DLL_SMS_ERROR message not before T11min starting from the reception of a DLL_SMS_ACK message with a DLL error.
Requirements refs:	5.3.1, 5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(MS=80, PDU=DLL_SMS_ACK(PL=TSPX_SMS_SUBMIT_REP_ACK), CS=FALSE) Start T11min – TDELTA Timeout T11min FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	DLL_SMS_ERROR(PL='Wrong checksum') received after timeout T11min
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_FRM_TIM_VAL_07
Purpose:	Verify that the SM-TE retransmits the last DLL message previously sent not before timeout of T11min, starting from the reception of a DLL_SMS_ERROR message.
Requirements refs:	5.3.1, 5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ERROR(PL='Wrong checksum') Start T11min – TDELTA Timeout T11min FS?DLL_SMS_DATA(E=1, PL=?) (see note)
Pass criteria:	DLL_SMS_DATA message retransmitted and received after timeout T11min
Postamble:	POST_TESTER_RELEASE_ALL
	NOTE: The payload contained in the DLL_SMS_DATA message shall be the same as the payload of the DLL_SMS_DATA message transmitted in the preamble.

	UBS1_DLL_FRM_TIM_VAL_08
Purpose:	Verify that the SM-TE sends the DLL_SMS_REL message not before T11min starting from the reception of the last DLL message.
Requirements refs:	5.3.1, 5.3.2.1, table 1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK(PL=TSPX_SMS_SUBMIT_REP_ACK) Start T11min – TDELTA Timeout T11min FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after timeout T11min
Postamble:	POST_TESTER_RELEASE_ALL

7.3 Test purposes for Outgoing call/Establishment (UBS1_DLL_OUT_EST)

	UBS1_DLL_OUT_EST_VAL_01
Purpose:	Verify that the SM-TE, once the VB connection has been established (to submit a SM and not to collect it), accepts the DLL_SMS_EST message as first DLL message (i.e. it does not send a DLL_SMS_ERROR message).
Requirements refs:	5.3.2.1, table 1, A.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_EST Wait to ensure that no DLL_SMS_ERROR is received (see note)
Pass criteria:	No DLL_SMS_ERROR is received upon the DLL_SMS_EST message.
Postamble:	POST_TESTER_RELEASE_ALL
	NOTE: A DLL_SMS_DATA message is accepted.

	UBS1_DLL_OUT_EST_VAL_02
Purpose:	Verify that the SM-TE, in case it does not receive any DLL message before the expiry of a timer starting from the VB connection establishment, sends a DLL_SMS_REL message.
Requirements refs:	5.3.2.1, table 1, A.1.7
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	Start Establish-timer(TSPX_EST_TIMEOUT + TDELTA) FS?DLL_SMS_REL
Pass criteria:	FS?DLL_SMS_REL
Postamble:	POST_IUT_RELEASE_VB

	UBS1_DLL_OUT_EST_INV_01
Purpose:	Verify that the SM-TE sends a DLL_SMS_ERROR (error cause='wrong checksum') message after receiving a DLL_SMS_EST message with a wrong checksum.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(MS=80, PDU=DLL_SMS_EST, CS=FALSE) FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	FS?DLL_SMS_ERROR(PL='Wrong checksum')
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_OUT_EST_INV_02
Purpose:	Verify that the SM-TE sends a DLL_SMS_ERROR (error cause='Wrong message length' or 'Wrong checksum') message after receiving a DLL_SMS_EST message with a wrong message length.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_EST(E=1, ML=1, PL=OMIT) FS?DLL_SMS_ERROR(PL='Wrong message length') Or FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	DLL_SMS_ERROR(PL='Wrong message length') or DLL_SMS_ERROR(PL='Wrong checksum') received
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_OUT_EST_INV_03
Purpose:	Verify that the SM-TE sends a DLL_SMS_ERROR (error cause='unknown message type') message if it receives as first DLL message a DLL message with an unknown message type.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_UNKNOWN FS?DLL_SMS_ERROR(PL='Unknown message type')
Pass criteria:	FS?DLL_SMS_ERROR(PL='Unknown message type')
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_OUT_EST_INV_04
Purpose:	Verify that the SM-TE, in case of a 'collect SM' call, sends a DLL_SMS_REL message if it does not receive any DLL message before the expiry of a timer starting from the VB connection establishment.
Requirements refs:	5.3.2.1, table 1
Selection Cond.:	
Preamble:	PRE_INIT
Test description:	CC!INC_CALL_req(DelModeId=2) CC?INC_CALL_conf(ESTCONF=FALSE) CC?OUTG_CALL_ind CC!OUTG_CALL_resp Start Establish-timer(TSPX_EST_TIMEOUT - TDELTA) Timeout Establish-timer FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after timeout of Establish-timer.
Postamble:	POST_TESTER_RELEASE_ALL

7.4 Test purposes for Outgoing call/Data transfer (UBS1_DLL_OUT_DAT)

	UBS1_DLL_OUT_DAT_VAL_01
Purpose:	Verify that the SM-TE sends a DLL_SMS_DATA (SMS_SUBMIT) message after receiving the DLL_SMS_EST message.
Requirements refs:	5.3.2.1, table 1, A.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_EST(LengthInd=0, SmsNum=1)
Test description:	FS?DLL_SMS_DATA(E=1, PL=?)
Pass criteria:	DLL_SMS_DATA message received
Postamble:	POST_IUT_RELEASE_ALL

	UBS1_DLL_OUT_DAT_VAL_02
Purpose:	Verify that the SM-TE after sending a DLL_SMS_DATA (SMS_SUBMIT) message accepts a DLL_SMS_ACK (SMS_SUBMIT_REPORT) message (i.e. it doesn't send a DLL_SMS_ERROR message).
Requirements refs:	5.3.2.1, table 1, A.1.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK(PL=TSPX_SMS_SUBMIT_REP_ACK) Wait to ensure that no DLL_SMS_ERROR is received (see note)
Pass criteria:	No DLL_SMS_ERROR is received
Postamble:	POST_IUT_RELEASE_ALL
	NOTE: A DLL_SMS_REL message, possibly followed by VB release, is accepted.

	UBS1_DLL_OUT_DAT_VAL_03
Purpose:	Verify that the SM-TE after sending a DLL_SMS_DATA (SMS_SUBMIT) message accepts a DLL_SMS_NACK (SMS_SUBMIT_REPORT) message (i.e. it does not send a DLL_SMS_ERROR message).
Requirements refs:	5.3.2.1, table 1, A.1.2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_NACK(PL=TSPX_SMS_SUBMIT_REP_NACK) Wait to ensure that no DLL_SMS_ERROR is received
Pass criteria:	No DLL_SMS_ERROR is received
Postamble:	POST_IUT_RELEASE_ALL
	NOTE: A DLL_SMS_REL message, possibly followed by VB release, is accepted.

	UBS1_DLL_OUT_DAT_VAL_04
Purpose:	Verify that the SM-TE after sending a DLL_SMS_DATA (SMS_SUBMIT) message accepts a DLL_SMS_ERROR message (i.e. it does not send a DLL_SMS_ERROR message).
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2, A.1.3
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ERROR(PL='Unspecified error cause') Wait to ensure that no DLL_SMS_ERROR is received
Pass criteria:	No DLL_SMS_ERROR is received
Postamble:	POST_IUT_RELEASE_ALL
	NOTE: A DLL_SMS_REL message, possibly followed by VB release, is accepted.

	UBS1_DLL_OUT_DAT_VAL_05
Purpose:	Verify that in case of more than one SM to send, the SM-TE sends a new DLL_SMS_DATA (SMS_SUBMIT) message after receiving a DLL_SMS_ACK (SMS_SUBMIT_REPORT) message in response to the previous DLL_SMS_DATA (SMS_SUBMIT) message.
Requirements refs:	5.3.2.1, table 1
Selection Cond.:	SelMoreSMs_TX
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=2)
Test description:	FS!DLL_SMS_ACK(PL=TSPX_SMS_SUBMIT_REP_ACK) FS?DLL_SMS_DATA(E=1, PL=?) FS!DLL_SMS_ACK(PL=TSPX_SMS_SUBMIT_REP_ACK)
Pass criteria:	Two DLL_SMS_DATA messages received
Postamble:	POST_IUT_RELEASE_ALL

	UBS1_DLL_OUT_DAT_VAL_06
Purpose:	Verify that in case of more than one SM to send, the SM-TE, having sent a first DLL_SMS_DATA (SMS_SUBMIT) message and received a DLL_SMS_ACK (SMS_SUBMIT_REPORT) message, accepts a DLL_SMS_ACK (SMS_SUBMIT_REPORT) message in response to a new DLL_SMS_DATA message sent (i.e. it does not send a DLL_SMS_ERROR message).
Requirements refs:	5.3.2.1, table 1
Selection Cond.:	SelMoreSMs_TX
Preamble:	PRE_CONN_OUTG_EST(LengthInd=0, SmsNum=2)
Test description:	FS!DLL_SMS_ACK(PL=TSPX_SMS_SUBMIT_REP_ACK) FS?DLL_SMS_DATA(E=1, PL=?) FS!DLL_SMS_ACK(PL=TSPX_SMS_SUBMIT_REP_ACK) Wait to ensure that no DLL_SMS_ERROR is received
Pass criteria:	Two DLL_SMS_DATA messages received and no DLL_SMS_ERROR message received.
Postamble:	POST_IUT_RELEASE_ALL
	NOTE: A DLL_SMS_REL message is accepted.

	UBS1_DLL_OUT_DAT_INV_01
Purpose:	Verify that the SM-TE sends again the DLL_SMS_DATA (SMS_SUBMIT) message after receiving a DLL_SMS_ERROR message in response to the previous DLL_SMS_DATA (SMS_SUBMIT) message.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, A.1.3
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ERROR(E=1, PL='Unspecified error cause') FS?DLL_SMS_DATA(E=1, PL=?)
Pass criteria:	DLL_SMS_DATA(E=1, PL=?) retransmitted
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_OUT_DAT_INV_02
Purpose:	Verify that the SM-TE sends the DLL_SMS_REL message after receiving twice a DLL_SMS_ERROR message in response to two consecutives DLL_SMS_DATA (SMS_SUBMIT) messages.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, A.1.5
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ERROR(E=1, PL='Unspecified error cause') FS?DLL_SMS_DATA(E=1, PL=?) FS!DLL_SMS_ERROR(E=1, PL='Unspecified error cause') FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_DATA(E=1, PL=?) retransmitted once, then FS?DLL_SMS_REL
Postamble:	POST_IUT_RELEASE_VB

	UBS1_DLL_OUT_DAT_INV_03
Purpose:	Verify that the SM-TE sends the DLL_SMS_ERROR (error cause='wrong checksum') message after receiving a DLL_SMS_ACK (SUBMIT_REPORT) message with a wrong checksum.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2, A.1.4
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(MS=80, PDU= DLL_SMS_ACK(PL=TSPX_SMS_SUBMIT_REP_ACK), CS=FALSE) FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	FS?DLL_SMS_ERROR(PL='Wrong checksum')
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_OUT_DAT_INV_04
Purpose:	Verify that the SM-TE sends the DLL_SMS_ERROR (error cause='Wrong message length' or 'Wrong checksum') message after receiving a DLL_SMS_ACK (SUBMIT_REPORT) message with a wrong message length.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2, A.1.4
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK(E=1, ML= Length(TSPX_SMS_SUBMIT_REP_ACK) + 1, PL=TSPX_SMS_SUBMIT_REP_ACK) FS?DLL_SMS_ERROR(PL='Wrong message length') Or FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	DLL_SMS_ERROR(PL='Wrong message length') or DLL_SMS_ERROR(PL='Wrong checksum') received
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_OUT_DAT_INV_05
Purpose:	Verify that the SM-TE sends the DLL_SMS_ERROR (error cause='wrong checksum') message after receiving a DLL_SMS_NACK (SUBMIT_REPORT) message with a wrong checksum.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(MS=80, PDU= DLL_SMS_NACK(PL=TSPX_SMS_SUBMIT_REP_NACK), CS=FALSE) FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	FS?DLL_SMS_ERROR(PL='Wrong checksum')
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_OUT_DAT_INV_06
Purpose:	Verify that the SM-TE sends the DLL_SMS_ERROR (error cause='Wrong message length' or 'Wrong checksum') message after receiving a DLL_SMS_NACK (SUBMIT_REPORT) message with a wrong message length.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_NACK(E=1, ML= Length(TSPX_SMS_SUBMIT_REP_NACK) - 1, PL=TSPX_SMS_SUBMIT_REP_NACK) FS?DLL_SMS_ERROR(PL='Wrong message length') Or FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	DLL_SMS_ERROR(PL='Wrong message length') or DLL_SMS_ERROR(PL='Wrong checksum') received
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_OUT_DAT_INV_07
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_DATA (SMS_SUBMIT), sends a DLL_SMS_ERROR (error cause='Wrong message length' or 'Wrong checksum') message after receiving a DLL_SMS_ERROR message with a wrong message length.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ERROR(E=1, ML=0, PL='Unspecified error cause') FS?DLL_SMS_ERROR(PL='Wrong message length') Or FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	DLL_SMS_ERROR(PL='Wrong message length') or DLL_SMS_ERROR(PL='Wrong checksum') received
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_OUT_DAT_INV_08
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_DATA (SMS_SUBMIT), sends a DLL_SMS_ERROR (error cause='Wrong checksum') message after receiving a DLL_SMS_ERROR message with a wrong checksum.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!TRANSFER_req(MS=80, PDU= DLL_SMS_ERROR(PL='Unspecified error cause'), CS=FALSE) FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	FS?DLL_SMS_ERROR(PL='Wrong checksum')
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_OUT_DAT_INV_09
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_DATA (SMS_SUBMIT), sends the DLL_SMS_ERROR (error cause='unknown message type') message after receiving a DLL message with an unknown message type.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_UNKNOWN(E=1, PL=TSPX_UNKNOWN_PL) FS?DLL_SMS_ERROR(PL='Unknown message type')
Pass criteria:	FS?DLL_SMS_ERROR(PL='Unknown message type')
Postamble:	POST_TESTER_RELEASE_ALL

7.5 Test purposes for Outgoing call/Release (UBS1_DLL_OUT_REL)

	UBS1_DLL_OUT_REL_VAL_01
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_DATA (SMS_SUBMIT), sends a DLL_SMS_REL message after receiving a DLL_SMS_ACK (SMS_SUBMIT_REPORT) message, in case there are no other SMs to submit.
Requirements refs:	5.3.2.1, table 1, A.1.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK(E=1, PL=TSPX_SMS_SUBMIT_REP_ACK) FS?DLL_SMS_REL
Pass criteria:	FS?DLL_SMS_REL
Postamble:	POST_IUT_RELEASE_VB

	UBS1_DLL_OUT_REL_VAL_02
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_DATA (SMS_SUBMIT), sends a DLL_SMS_REL message after receiving a DLL_SMS_NACK (SMS_SUBMIT_REPORT) message, in case there are no other SMs to submit.
Requirements refs:	5.3.2.1, table 1, A.1.2
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_NACK(E=1, PL=TSPX_SMS_SUBMIT_REP_NACK) FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received
Postamble:	POST_IUT_RELEASE_VB

	UBS1_DLL_OUT_REL_VAL_03
Purpose:	Verify that the SM-TE releases the VB connection after sending the DLL_SMS_REL message.
Requirements refs:	5.3.2.1, table 1, A.1
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK(E=1, PL=TSPX_SMS_SUBMIT_REP_ACK) FS?DLL_SMS_REL CC?CALL_RELEASE_ind
Pass criteria:	VB connection released
Postamble:	None

	UBS1_DLL_OUT_REL_VAL_04
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_DATA message and not receiving any DLL message in response, sends the DLL_SMS_REL message after the expiry of the T12 timeout starting after the sending of the DLL_SMS_DATA message.
Requirements refs:	5.3.2.1, table 1, 5.3.2.3, A.1.8
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	Start T12 – TDELTA Timeout T12 FS?DLL_SMS_REL
Pass criteria:	FS?DLL_SMS_REL
Postamble:	POST_IUT_RELEASE_VB

	UBS1_DLL_OUT_REL_VAL_05
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_ERROR message and not receiving any DLL message in response, sends the DLL_SMS_REL message after the expiry of the T12 timeout starting after the sending of the DLL_SMS_ERROR message.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2, 5.3.2.3
Selection Cond.:	
Preamble:	PRE_CONN_OUTG_DATA(LengthInd=0, SmsNum=1)
Test description:	FS!DLL_SMS_ACK(E=1, ML = Len(TSPX_SMS_SUBMIT_REP_ACK) – 1, PL=TSPX_SMS_SUBMIT_REP_ACK) FS?DLL_SMS_ERROR(PL='Wrong message length') Start T12 – TDELTA Timeout T12 FS?DLL_SMS_REL
Pass criteria:	FS?DLL_SMS_REL
Postamble:	POST_IUT_RELEASE_VB

7.6 Test purposes for Incoming call/Establishment (UBS1_DLL_INC_EST)

	UBS1_DLL_INC_EST_VAL_01
Purpose:	Verify that the SM-TE sends a correct DLL_SMS_EST message after the VB connection establishment initiated by the SM-SC.
Requirements refs:	5.3.2.1, table 1, A.2 (except A.2.1)
Selection Cond.:	
Preamble:	PRE_CONN_INC(DelModeId=0)
Test description:	FS?DLL_SMS_EST
Pass criteria:	DLL_SMS_EST received
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_INC_EST_INV_01
Purpose:	Verify that the SM-TE retransmits the DLL_SMS_EST message after receiving a DLL_SMS_ERROR message in response to the previous DLL_SMS_EST message.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST(DelModeId=0)
Test description:	FS!DLL_SMS_ERROR(E=1, PL='Wrong checksum') FS?DLL_SMS_EST
Pass criteria:	DLL_SMS_EST retransmitted
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_INC_EST_INV_02
Purpose:	Verify that the SM-TE sends the DLL_SMS_REL message after receiving twice the DLL_SMS_ERROR message in response to two consecutives DLL_SMS_EST messages.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST(DelModeId=0)
Test description:	FS!DLL_SMS_ERROR(E=1, PL='Wrong checksum') FS?DLL_SMS_EST FS!DLL_SMS_ERROR(E=1, PL='Wrong checksum') FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL received after second DLL_SMS_ERROR message.
Postamble:	POST_IUT_RELEASE_VB

7.7 Test purposes for Incoming call/Data transfer (UBS1_DLL_INC_DAT)

	UBS1_DLL_INC_DAT_VAL_01
Purpose:	Verify that the SM-TE sends a DLL_SMS_ACK (SMS_DELIVER_REPORT) message after receiving a correct DLL_SMS_DATA (SMS_DELIVER) message.
Requirements refs:	5.3.2.1, table 1, A.2
Selection Cond.:	
Preamble:	PRE_CONN_INC_DATA(DelModeId=0, '1'B, TSPX_SMS_DELIVER_S1)
Test description:	FS?DLL_SMS_ACK(E=1, PL=?)
Pass criteria:	DLL_SMS_ACK received
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_INC_DAT_VAL_02
Purpose:	Verify that the SM-TE sends a DLL_SMS_NACK message after receiving a DLL_SMS_DATA (SMS_DELIVER) message with a TL error.
Requirements refs:	5.3.2.1, table 1, A.2.4
Selection Cond.:	
Preamble:	PRE_CONN_INC_DATA(DelModeId=0, '1'B, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_NACK(E=1, PL=?)
Pass criteria:	DLL_SMS_NACK received
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_INC_DAT_VAL_03
Purpose:	Verify that the SM-TE, having acknowledged a DLL_SMS_DATA (SMS_DELIVER) message with the DLL_SMS_ACK (SMS_DELIVER_REPORT) message, acknowledges a new DLL_SMS_DATA (SMS_DELIVER) message with a new DLL_SMS_ACK (SMS_DELIVER_REPORT) message.
Requirements refs:	5.3.2.1, table 1, A.2.3
Selection Cond.:	
Preamble:	PRE_CONN_INC_DATA_ACK(DelModeId=0, '1'B, TSPX_SMS_DELIVER_S1)
Test description:	FS!DLL_SMS_DATA(E=1, PL=TSPX_SMS_DELIVER_S2) FS?DLL_SMS_ACK(E=1, PL=?)
Pass criteria:	Second DLL_SMS_DATA message acknowledged by the IUT.
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_INC_DAT_INV_01
Purpose:	Verify that the SM-TE sends a DLL_SMS_ERROR message (error cause='wrong checksum') after receiving a DLL_SMS_DATA (SMS_DELIVER) message with a wrong checksum.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2, A.2.5
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST(DelModeId=0)
Test description:	FS!TRANSFER_req(MS=80, PDU=DLL_SMS_DATA(PL=TSPX_SMS_DELIVER_S1), CS=FALSE) FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	DLL_SMS_ERROR(PL='Wrong checksum')
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_INC_DAT_INV_02
Purpose:	Verify that the SM-TE sends a DLL_SMS_ERROR message (error cause='wrong message length' or 'wrong checksum') after receiving a DLL_SMS_DATA (SMS_DELIVER) message with a wrong message length.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2, A.2.5
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST(DelModeId=0)
Test description:	FS!TRANSFER_req(MS=80, PDU=DLL_SMS_DATA(E=1, ML=Len(TSPX_SMS_DELIVER_S1) + 1, PL=TSPX_SMS_DELIVER_S1), CS=TRUE) FS?DLL_SMS_ERROR(PL='Wrong message length') Or FS?DLL_SMS_ERROR(PL='Wrong checksum')
Pass criteria:	DLL_SMS_ERROR(PL='Wrong message length') or DLL_SMS_ERROR(PL='Wrong checksum') received
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_INC_DAT_INV_03
Purpose:	Verify that the SM-TE sends a DLL_SMS_ERROR message (error cause='extension mechanism not supported') after receiving a DLL_SMS_DATA (SMS_DELIVER) message with the extension bit='0'.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2, A.2.5
Selection Cond.:	SEL_NOT_DLL_EXT_RX
Preamble:	PRE_CONN_INC_EST(DelModeId=0)
Test description:	FS!DLL_SMS_DATA(E=0, PL=TSPX_SMS_DELIVER_S1) FS?DLL_SMS_ERROR(PL='Extension mechanism not supported')
Pass criteria:	DLL_SMS_ERROR(PL='Extension mechanism not supported') received
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_INC_DAT_INV_04
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_EST message, sends a DLL_SMS_ERROR message (error cause='unknown message type') after receiving a DLL message with an unknown message type.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, table 2
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST(DelModeId=0)
Test description:	FS!DLL_SMS_UNKNOWN(E=1, PL=TSPX_UNKNOWN_PL) FS?DLL_SMS_ERROR(PL='Unknown message type')
Pass criteria:	DLL_SMS_ERROR(PL='Unknown message type') received
Postamble:	POST_TESTER_RELEASE_ALL

	UBS1_DLL_INC_DAT_INV_05
Purpose:	Verify that the SM-TE sends again the DLL_SMS_ACK (SMS_DELIVER_REPORT) message after receiving a DLL_SMS_ERROR message in response to a DLL_SMS_ACK (SMS_DELIVER_REPORT) message.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, A.2.6
Selection Cond.:	
Preamble:	PRE_CONN_INC_DATA_ACK(DelModeId=0, '1'B, TSPX_SMS_DELIVER_S1)
Test description:	FS!DLL_SMS_ERROR(PL='Wrong checksum') FS?DLL_SMS_ACK(PL=?)
Pass criteria:	DLL_SMS_ACK retransmitted with the same payload as in the preamble.
Postamble:	POST_TESTER_RELEASE_ALL
	NOTE: The DLL_SMS_ACK message shall contain the same payload as the one in the preamble.

	UBS1_DLL_INC_DAT_INV_06
Purpose:	Verify that the SM-TE sends the DLL_SMS_REL message after receiving twice the DLL_SMS_ERROR message in response to two consecutives DLL_SMS_ACK (SMS_DELIVER_REPORT) messages.
Requirements refs:	5.3.2.1, table 1, 5.3.2.2, A.2.8
Selection Cond.:	
Preamble:	PRE_CONN_INC_DATA_ACK(DelModeId=0, '1'B, TSPX_SMS_DELIVER_S1)
Test description:	FS!DLL_SMS_ERROR(PL='Wrong checksum') FS?DLL_SMS_ACK(PL=?) FS!DLL_SMS_ERROR(PL='Wrong checksum') FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL received after second DLL_SMS_ERROR message.
Postamble:	POST_TESTER_RELEASE_VB

	UBS1_DLL_INC_DAT_INV_07
Purpose:	Verify that the SM-TE, in case it receives a DLL_SMS_DATA (SMS_DELIVER) message with a payload larger than 176 octets, sends a DLL_SMS_NACK (SMS_DELIVER_REPORT) message or a DLL_SMS_ERROR('Wrong message length' or 'Wrong checksum') message.
Requirements refs:	5.3.2.1, 5.3.2.2
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST(DelModeId=0)
Test description:	FS!DLL_SMS_DATA_LONG(E=1, PL=TSPX_TOO_LONG_PL) FS?DLL_SMS_NACK(E=1, PL=?) or FS?DLL_SMS_ERROR('Wrong message length') or FS?DLL_SMS_ERROR('Wrong checksum')
Pass criteria:	DLL_SMS_NACK or DLL_SMS_ERROR(PL='Wrong checksum') or DLL_SMS_ERROR(PL='Wrong checksum') received.
Postamble:	POST_TESTER_RELEASE_ALL

7.8 Test purposes for Incoming call/Release (UBS1_DLL_INC_REL)

	UBS1_DLL_INC_REL_VAL_01
Purpose:	Verify that the SM-TE releases the VB connection after the reception of the DLL_SMS_REL message.
Requirements refs:	5.3.2.1, table 1, A.2
Selection Cond.:	
Preamble:	PRE_CONN_INC_DATA_ACK(DelModeId=0, '1'B, TSPX_SMS_DELIVER_S1)
Test description:	FS!DLL_SMS_REL CC?CALL_RELEASE_ind
Pass criteria:	CC?CALL_RELEASE_ind
Postamble:	None

	UBS1_DLL_INC_REL_VAL_02
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_EST message and not received any DLL message in response, sends the DLL_SMS_REL message after the expiry of the T12 timeout starting after the sending of the DLL_SMS_EST message.
Requirements refs:	5.3.2.1, table 1, 5.3.2.3
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST(DelModeId=0)
Test description:	Start T12 – TDELTA Timeout T12 FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after Timeout T12.
Postamble:	POST_IUT_RELEASE_VB

	UBS1_DLL_INC_REL_VAL_03
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_ACK (SMS_DELIVER_REPORT) message and not receiving any DLL message in response, sends the DLL_SMS_REL message after the expiry of the T12 timeout starting after the sending of the DLL_SMS_ACK (SMS_DELIVER_REPORT) message.
Requirements refs:	5.3.2.1, table 1, 5.3.2.3
Selection Cond.:	
Preamble:	PRE_CONN_INC_DATA_ACK(DelModeId=0, '1'B, TSPX_SMS_DELIVER_S1)
Test description:	Start T12 – TDELTA Timeout T12 FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after Timeout T12.
Postamble:	POST_IUT_RELEASE_VB

	UBS1_DLL_INC_REL_VAL_04
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_NACK (SMS_DELIVER_REPORT) message and not receiving any DLL message in response, sends the DLL_SMS_REL message after the expiry of the T12 timeout starting after the sending of the DLL_SMS_NACK (SMS_DELIVER_REPORT) message.
Requirements refs:	5.3.2.1, table 1, 5.3.2.3
Selection Cond.:	
Preamble:	PRE_CONN_INC_DATA(DelModeId=0, '1'B, TSPX_SMS_DELIVER_S3)
Test description:	FS?DLL_SMS_NACK(E=1, PL=?) Start T12 – TDELTA Timeout T12 FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after Timeout T12.
Postamble:	POST_IUT_RELEASE_VB

	UBS1_DLL_INC_REL_VAL_05
Purpose:	Verify that the SM-TE, having sent a DLL_SMS_ERROR message upon receiving a DLL_SMS_DATA message with a wrong checksum, and having not received any DLL message in response, sends a DLL_SMS_REL message after the expiry of the T12 timeout starting after the sending of the DLL_SMS_ERROR message.
Requirements refs:	5.3.2.1, table 1, 5.3.2.3
Selection Cond.:	
Preamble:	PRE_CONN_INC_EST(DelModeId=0)
Test description:	FS!TRANSFER_req(MS=80, PDU=DLL_SMS_DATA(E=1, PL=TSPX_SMS_DELIVER_S1), CS=FALSE) FS?DLL_SMS_ERROR(E=1, PL='Wrong checksum') Start T12 – TDELTA Timeout T12 FS?DLL_SMS_REL
Pass criteria:	DLL_SMS_REL message received after Timeout T12.
Postamble:	POST_IUT_RELEASE_VB

Annex A (informative): Bibliography

ETSI TS 100 942 (V7.0.0): "Digital cellular telecommunications system (Phase 2+) (GSM); Point-to-Point (PP) Short Message Service (SMS) support on mobile radio interface (GSM 04.11 version 7.0.0)".

ISO/IEC 9646-7: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 7: Implementation Conformance Statements".

History

Document history		
V1.1.1	December 2002	Membership Approval Procedure MV 20030207: 2002-12-10 to 2003-02-07
V1.1.1	February 2003	Publication