EG 201 022 V1.1.1 (1998-06)

ETSI Guide

Broadband Integrated Services Digital Network (B-ISDN); Digital Subscriber Signalling No. two (DSS2) protocol; B-ISDN user-network interface layer 3 specification for point-to multipoint call/bearer control; Service Description Language (SDL) validation model

Reference

DEG/SPS-05128 (9go00icq.PDF)

Keywords

B-ISDN, basic, DSS2, broadband, network, SDL,
testing, TSS&TP, ISDN, validation, layer 3,
multipoint

ETSI

Postal address

F-06921 Sophia Antipolis Cedex - FRANCE

Office address

650 Route des Lucioles - Sophia Antipolis
Valbonne - 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

Internet

secretariat@etsi.fr
http://www.etsi.fr
http://www.etsi.org

Contents

Foreword

This ETSI Guide (EG) has been produced by ETSI Technical Committee Signalling Protocols and Switching (SPS).

Introduction

The present document describes and specifies two Specification and Description Language (SDL) models of the Broadband Integrated Digital Network (B-ISDN) Digital Subscriber System No 2 (DSS2) point-to-multipoint capability as defined in ITU-T Recommendation Q.2971 as modified by ETS 300 771-1 [2].

The models are fully simulateable on applicable SDL tools. The first model was used in the development and validation of TPs for B-ISDN DSS2 point-to-multipoint. The second model (an enhanced version of the first model) was used to validate the corresponding TTCN test cases by simulation. This model was also used as a testbed for Computer Aided Test Case Generation (CATG) tools.

The main body of the present document comprises a descriptive overview of the SDL models and the graphical user interfaces associated with them.

Annexes A and B are the respective SDL models in graphical format.

Annexes C and D are packages containing all the SDL source files for the respective models (including SDL.PR files, SDT source code, code for the graphical user interfaces, simulation files and installation instructions).

1 Scope

The present document describes and specifies two detailed SDL validation models of the B-ISDN DSS2 point-to-multipoint capability as defined in ITU-T Recommendation Q.2971 as modified by ETS 300 771-1 [2].

The models can also handle point-to-point calls as defined in ITU-T Recommendation Q.2931 as modified by ETS 300 443-1 [1] but they do not include narrowband/broadband interworking requirements.

The models have been used to develop and validate Test Purposes (TPs) and test cases for B-ISDN DSS2 point-to-multipoint capability. The models are not intended as a replacement of the SDL diagrams in ITU-T Recommendation Q.2971 as modified by ETS 300 771-1 [2].

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, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, subsequent revisions do apply.
- A non-specific reference to an ETS shall also be taken to refer to later versions published as an EN with the same number.

[1] ETS 300 443-1 (1996): "Broadband Integrated Services Digital Network (B-ISDN); Digital Subscriber Signalling System No. two (DSS2) protocol; B-ISDN user-network interface layer 3 specification for basic call/bearer control; Part 1: Protocol specification; [ITU-T Recommendation Q.2931 (1995), modified]".

[2] ETS 300 771-1 (1997): "Broadband Integrated Services Digital Network (B-ISDN); Digital Subscriber Signalling System No. two (DSS2) protocol; B-ISDN user-network interface layer 3 specification for point-to-multipoint call/bearer control; Part 1: Protocol specification; [ITU-T Recommendation Q.2971, modified]".

[3] ETS 300 771-4 (1998): "Broadband Integrated Services Digital Network (B-ISDN); Digital Subscriber Signalling System No. two (DSS2) protocol; B-ISDN user-network interface layer 3 specification for point-to-multipoint call/bearer control; Part 4: Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma specification for the user".

[4] ETS 300 771-5 (1998): "Broadband Integrated Services Digital Network (B-ISDN); Digital Subscriber Signalling System No. two (DSS2) protocol; B-ISDN user-network interface layer 3 specification for point-to-multipoint call/bearer control; Part 5: Test Suite Structure and Test Purposes [TSS&TP] specification for the network".

[5] ETS 300 771-6 (1998): "Broadband Integrated Services Digital Network (B-ISDN); Digital Subscriber Signalling System No. two (DSS2) protocol; B-ISDN user-network interface layer 3 specification for point-to-multipoint call/bearer control; Part 6: Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma specification for the network".

[6] ISO/IEC 9646-3 (1992): "Information technology - Open systems interconnection - Conformance testing methodology and framework - Part 3: The Tree and Tabular Combined Notation (TTCN)".

[7] ITU-T Recommendation Z.100 (1994): "CCITT specification and description language (SDL)".

[8] ITU-T Recommendation Z.105 (1994): "SDL Combined with ASN.1".

[9] ITU-T Recommendation Z.120 (1993): "Message sequence chart".

3 Abbreviations

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

ASN.1	Abstract Syntax Notation one
B-ISDN	Broadband ISDN
CATG	Computer-Aided Test Case Generation
CR	Call Reference
CS-2	Capability Set No 2
DSS2	Digital Subscriber Signalling System No two
ER	Endpoint Reference
INAP	Intelligent Networks Application Protocol
IUT	Implementation Under Test
ISDN	Integrated Services Digital Network
GUI	Graphical User Interface
MSC	Message Sequence Chart
N-ISDN	Is it Network-ISDN
SDL	Specification and Description Language
TP	Test Purpose
TTCN	Tree and Tabular Combined Notation
TSS&TP	Test Suite Structure and Test Purposes
UNI	User Network Interface

4 General modelling principles in Specification and Description Language (SDL)

The methodologies of ETS 300 771-4 to 6 [3] to [5] define the use of SDL at various levels of detail, depending on whether the purpose of the model is for specification or validation (or other purposes which require a simulateable model). These are usually called a specification model and a validation model respectively.

4.1 Specification model

The purpose of a specification model is to clearly and precisely convey the (functional) requirements of the protocol (service or system) to a human reader, usually in the context of the text standard. This model is targeted towards readers who may not be SDL experts and who may not have access to SDL tools, i.e., it's importance is mainly as a paper document. The specification model can form the basis for the more detailed validation model if this is required at a later stage.

MTS methodology states that a specification model should use correct SDL syntax and static semantics, that is, it should conform to ITU-T Recommendation Z.100 [7]. It should attempt to be simple (without being trivial) by having a "descriptive feel" rather than a "programming feel". This model may not necessarily be a complete description of the protocol in question. It can, for example, concentrate on particular functional aspects of the protocol or perhaps only model data on a simple level.

There should be a clear relationship between the text version of the standard and the corresponding SDL diagrams. This can be achieved, for example by embedded cross references, naming procedures after clauses and by using identifiers for signals, conditions, variables etc. that have an obvious and unambiguous correspondence with the text standard.

4.2 Validation model

Because of its incompleteness it is not always possible to execute the specification model. The validation model complements the specification model with the necessary programming detail in order to make an executable specification. In the present document much of the extra detail is concerned with process management, especially the internal co-ordination and routing of signals to the correct processes. The validation model also includes extra, non-standard functionality such as a network application process.

When read by a non-SDL expert this additional code does not usually add to the intuitive understanding of the protocol. The true value of the validation model only becomes apparent when it is actually executed.

In general these executable models can be used for:

- development and evaluation of a protocol by simulation;
- validation of the standard by the application of specialized validation tools;
- Test Purpose (TP) development and validation;
- automated test case generation;
- education and demonstration purposes;
- a basis for other models (design, interworking).

4.3 The models specified in the present document

The present document describes two SDL validation models for B-ISDN DSS2 point-to-multipoint capability. The first model was used to develop and validate TPs for the DSS2 protocol described in ETS 300 771-5 [4]. The second model was used to validate the manually produced Tree and Tabular Combined Notation (TTCN) test suites described in ETS 300 771-6 [5] and ETS 300 771-4 [3]. The second model was also used as a testbed for Computer-Aided Test Case Generation (CATG) techniques and tools.

5 SDL model for the development and validation of TPs

This clause provides an introductory overview of the architecture and functionality of the TP development model. It is not intended to explain the detailed mechanisms of the model which are best understood by looking at the SDL specification itself. The model was developed and executed on the Telelogic tool, SDT 3.11. It has not been executed on other tools.

The graphical format of this model can be found in annex A. The full SDL specification can be found in annex C.

5.1 Basic building-blocks

The validation model is specified in object oriented SDL which allows us to define basic building-blocks (types and packages). These building blocks can easily be used to put together various configurations.

5.2 Communication paths

In the SDL model there is one main channel to the environment, called the User Network Interface (UNI) channel. This channel carries all the UNI messages to and from the environment, i.e., users of the network. The UNI channel is logically sub-structured into trunks, calls and parties. It may contain any number of trunks. A trunk may contain any number of calls and a call may contain any number of parties.

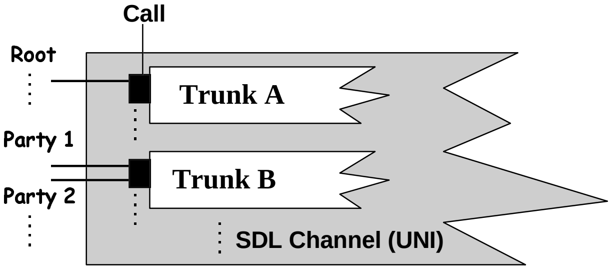
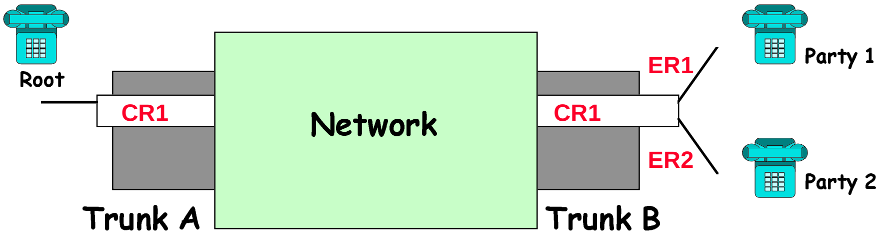


Figure 1: Conceptual model of the UNI

There is one trunk for each S_B/T_B or T_B reference point. A call corresponds to a Call Reference (CR) and a party corresponds to an Endpoint Reference (ER).



This figure shows a simple configuration that could be used for developing conformance tests.

Note that the UNI channel carries both Trunk A and Trunk B.

Figure 2: An example configuration for testing

5.3 The network

The Network comprises three main components, namely:

- DSS2 protocol (p-to-mp, Network Side);
- Network Application;
- Test Purpose Handler.

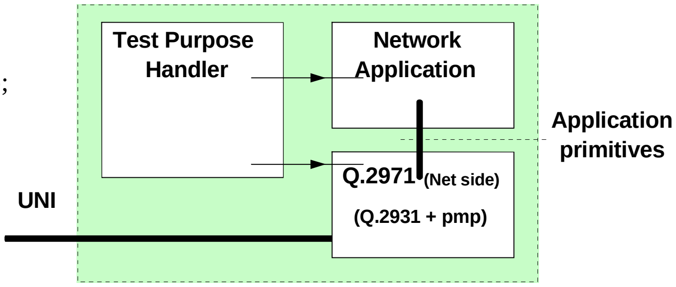


Figure 3: The Network functional blocks

5.4 The DSS2 protocol

This component models the B-ISDN for DSS2, network side according to ITU-T Recommendation Q.2931 as modified by ETS 300 443-1 [1] except for those aspects related to N-ISDN interworking. The model also includes full point-to-multipoint capabilities according to ITU-T Recommendation Q.2971 as modified by ETS 300 771-1 [2]. The protocol is divided into four functional blocks

- Call Control
- Party Control
- Co-ordination
- Reset

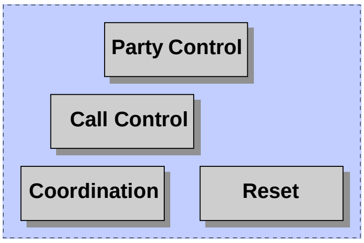
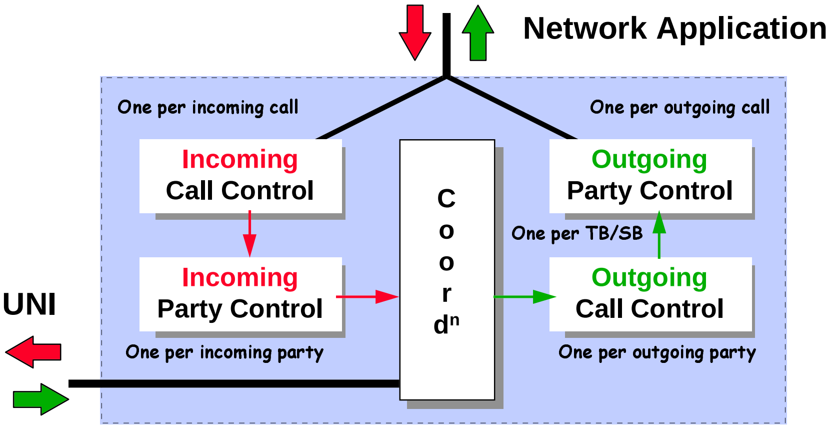


Figure 4: The DSS2 protocol functional blocks

5.4.1 Call and party control

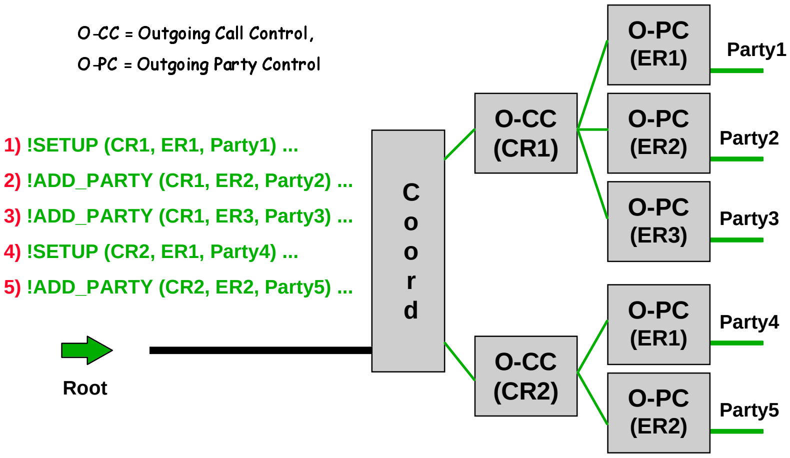
The present document separates call and party control into incoming and outgoing calls, as illustrated in figure 5.



Note that we are using accepted terminology by associating direction with the user. Thus, when we say outgoing call we mean outgoing from the user's point of view and not from the network's point of view.

Figure 5: Incoming and outgoing calls

One co-ordination process is created for each trunk. One call control process (either ingoing or outgoing, depending on the direction of the call) is created for each call and one party process is created for each party.


O-CC = Outgoing Call Control,
O-PC = Outgoing Party Control
1) !SETUP (CR1, ER1, Party1) ...
2) !ADD_PARTY (CR1, ER2, Party2) ...
3) !ADD_PARTY (CR1, ER3, Party3) ...
4) !SETUP (CR2, ER1, Party4) ...
5) !ADD_PARTY (CR2, ER2, Party5) ...
Root
C
o
o
r
d
O-CC (CR1)
O-CC (CR2)
O-PC (ER1)
O-PC (ER2)
O-PC (ER3)
O-PC (ER1)
O-PC (ER2)
Party1
Party2
Party3
Party4
Party5

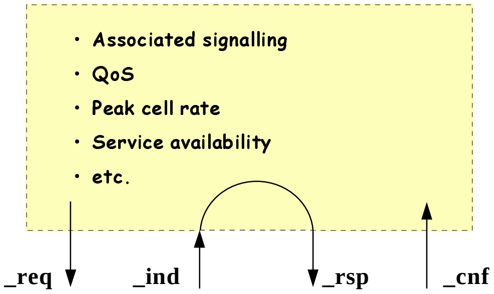
In this example five parties are set-up, three on call CR1 and two on call CR2. Note that Endpoint References (ER1 etc.) need only be unique within a call.

Note that the first call is created by sending a SETUP. Subsequent parties are created using ADD_PARTY.

Figure 6: Creation of processes

5.5 The network application

The network application needs to be present in order to provide full simulation capabilities. This process takes care of all the interactions that can occur at the network/application interface. These interactions are not standardized in the main body of ITU-T Recommendation Q.2971 as modified by ETS 300 771-1 [2] but we follow the convention where they are treated as service primitives modelled as SDL signals (e.g., Setup_req, Alerting_ind etc.).



Depending on conditions set by the user the application will perform the necessary responses to any given sequence of network interactions.

Typical conditions are support of QoS, availability of services or peak cell rates and support for associated/non-associated signalling

Figure 7: The network application

5.6 The lean data model

One of the main difficulties when building an SDL validation model is to decide the level of detail to describe data, such as the content of the UNI messages. There are two reasons for this:

- SDL is not good at modelling data, for example it is difficult to model message parameters that are optional or where choices are offered. This situation should improve when SDL tools support ASN.1 as defined in ITU-T Recommendation Z.105 [8];
- when data is fully modelled the specification can become very complex, without actually increasing the functional behaviour exhibited by the model.

The present document uses a lean data model where:

- only parameters essential to modelling dynamic data requirements are included in the UNI messages. This means, for example, that "transparent" parameters and internal message contents dependencies are not modelled (e.g., if p2 present then omit p3);
- other protocol requirements are modelled symbolically in the TP data base.

5.6.1 Parameters carried by UNI messages

The following parameters are carried by the appropriate UNI messages:

- signal type identifier;
- call reference and endpoint reference;
- called party number;
- current state and endpoint state;
- cause.

5.6.2 Symbolic TP data

A typical example of symbolic data is the case of "missing mandatory information element". Without symbolic values the SDL model would need to be programmed to analyse the entire content of all messages in order to determine whether or not a mandatory information element is missing. In a symbolic model such properties are represented as conditions of the system. We can, for example, simply set a condition Mandatory_IE_Missing to TRUE and the model will exhibit behaviour as if a mandatory information element was missing without actually having to build or analyse the message in detail.

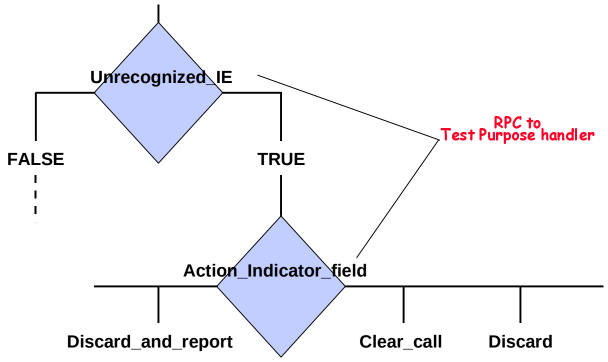
Other examples of the symbolic values in the TP data base are Unrecognized_IE, Message_too_short, Service_not_supported, VPCI_VCI_not_available etc. The model currently supports approximately 40 such symbolic values.

5.6.3 Consequences of the lean data model

This technique leads to a much simpler model (both to build and to understand) which still exhibits the full (functional) behaviour of the protocol. It also provides a clear relation to the text standard. However, in the case of test case generation, for example, the lean data model will not produce complete message values, or constraints as they are called in TTCN as defined in ISO/IEC 9646-3 [6]. These constraints shall be completed manually in the TTCN test suite. It is a question of trade-off, that is, either putting the work into the SDL model or into the TTCN test suite.

5.7 The TP handler

Many of the requirements in a (text) standard can be represented in an SDL model as decisions. In the present document the conditions of the decisions are (remote) procedure calls. The names of the procedures have a clear and consistent relation to the corresponding text, as illustrated in figure 8.



5.7.2 Information element errors

When a message other than a RELEASE ... is received which has one or more unexpected information elements, *unrecognised information elements* or information elements with unrecognized contents, the receiving entity ... follow the procedures described in a), b), c), d) or e)

...

a) Action indicator field = clear call

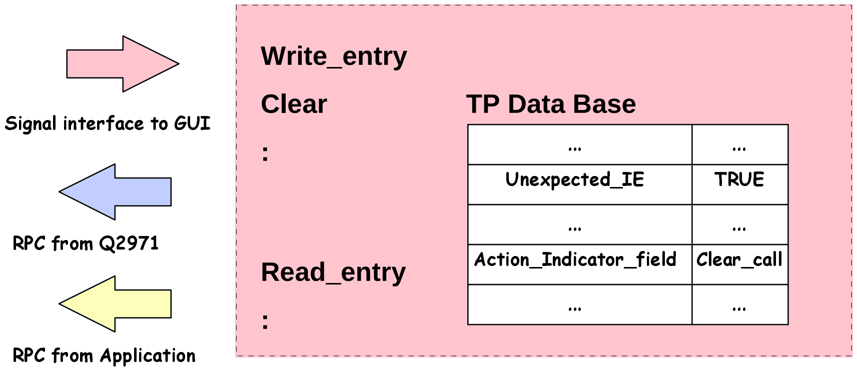
If the *action indicator field* is equal to *"clear call"* the call shall be cleared ... with cause No. 99 ...

:

b) ...

Figure 8: Requirements modelled as decisions

The TP handler allows the user to set the various conditions. When the procedure corresponding to a data base entry is invoked, either from the network or the application, they will return the value defined in the data base. Note that TP conditions are associated with individual UNI messages, thus, for example, we can specify an ALERTING as having an unexpected IE while all other messages will be considered not to have an unexpected IE.



...	...
Unexpected_IE	TRUE
...	...
Action_Indicator_field	Clear_call
...	...

*In this example the condition **Unexpected_IE** has been set to **TRUE** and (default is **FALSE**) and the value of the **Action_Indicator_field** has been set to **Clear_call**.*

Writing to the TP data base can be done directly over the signal interface or via the GUI.

Figure 9: The TP Handler

5.8 The Graphical User Interface (GUI) for the TP model

To demonstrate the applicability of the model to a wider audience the executable version of the present document has been complemented with a simple graphical user interface or GUI.

Although the GUI has been used to develop and validate the Test Suite Structure and Test Purposes (TSS&TP) for ETS 300 771-1 [2] it is also suitable as a tool to evaluate the protocol itself. The advantage is that the GUI provides simulation of the protocol by *someone who has little or even no knowledge of SDL*.

The GUI provides four dialogues which allow the user to:

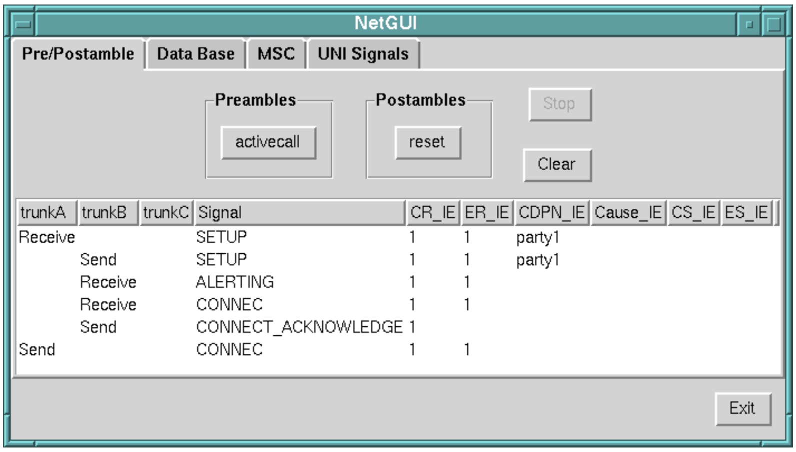
- execute pre- and postambles;
- set different TP conditions;
- set up calls and parties and to send UNI messages over these connections;
- create Message Sequence Charts (MSCs) files that can be displayed by the SDT tool.

For simplicity the GUI allows a maximum of three trunks (TrunkA, TrunkB and TrunkC). Within each trunk we may have a maximum of 10 calls (numbered CR0 - CR9) and within each call we may have up to 10 parties (numbered ER0 - ER9 and named Party0 - Party9). By default, the root party is considered to be ER0/Party0.

5.8.1 Running pre- and postambles

The pre/postamble dialogue provides a number of predefined preambles which are used to drive the network into certain combinations of states. For example the preamble n10p7p3 drives the network into call control state N10 with party1 in party control state P7 and party2 in party control state P3.

Currently only one postamble is defined. This postamble resets the whole system. Additional pre- and postambles can easily be defined and added to the GUI.



trunkA	trunkB	trunkC	Signal	CR_IE	ER_IE	CDPN_IE	Cause_IE	CS_IE	ES_IE
Receive			SETUP	1	1	party1			
	Send		SETUP	1	1	party1			
	Receive		ALERTING	1	1				
	Receive		CONNEC	1	1				
	Send		CONNECT_ACKNOWLEDGE	1					
Send			CONNEC	1	1				

In this example the preamble ***activecall*** *has driven the network into states N10 and P7 (one party only).*

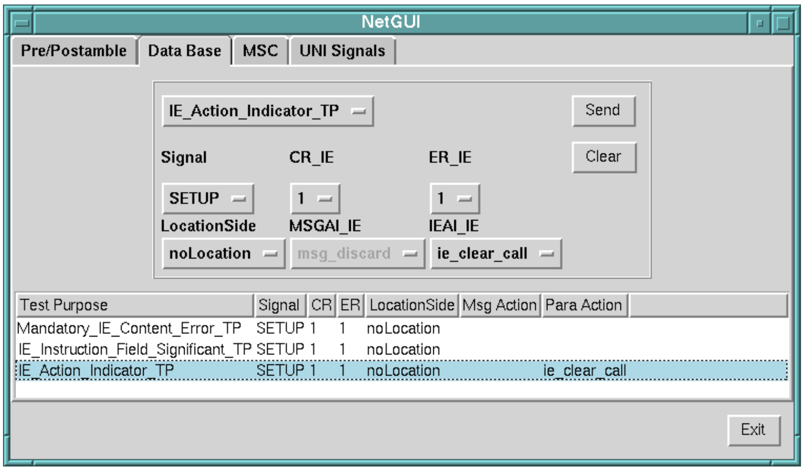
Note also, that because this is an environment for test purpose development the Network is assumed to be the Implementation Under Test (IUT). Thus, on the trunks (in this case TrunkA) the directions Receive and Send are from the IUT's point of view.

Figure 10: Setting preambles with the GUI

5.8.2 Setting TP conditions

The Data Base dialogue interfaces to the TP handler to allow the easy set-up of TP conditions. Note that the conditions are associated with individual UNI messages. Any number of the pre-defined TP conditions (of which there are currently about 40) can be associated with any message. There is no restriction on which and how many messages may be associated to TP conditions.

Finally, the settings need not be static, TP conditions on any particular message may be redefined during execution of the test body.



IE_Action_Indicator_TP

Signal: SETUP

CR_IE: 1

ER_IE: 1

LocationSide: noLocation

MSGAI_IE: msg_discard

IEAI_IE: ie_clear_call

Test Purpose	Signal	CR	ER	LocationSide	Msg Action	Para Action
Mandatory_IE_Content_Error_TP	SETUP	1	1	noLocation		
IE_Instruction_Field_Significant_TP	SETUP	1	1	noLocation		
IE_Action_Indicator_TP	SETUP	1	1	noLocation		ie_clear_call

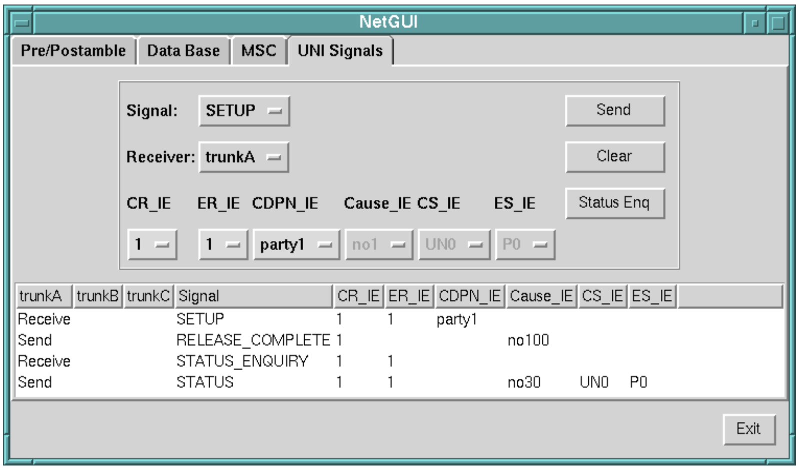
This example sets the conditions for the following (partial) TP:

Ensure that the IUT in N0 and P0, on receipt of a SETUP message (Mandatory information element with contents error, IE instruction field flag = "IE instruction field significant" and IE action indicator = clear call), sends ...... etc.

Figure 11: Setting TP conditions with the GUI

5.8.3 Sending UNI messages

Once the data base has been set-up and once the preamble has been run the UNI Signals dialogue is used to send/receive other UNI messages. In the case of TP development/validation these would be the messages comprising the test body.



Signal: SETUP

Receiver: trunkA

CR_IE: 1

ER_IE: 1

CDPN_IE: party1

Cause_IE: no1

CS_IE: UN0

ES_IE: P0

trunkA	trunkB	trunkC	Signal	CR_IE	ER_IE	CDPN_IE	Cause_IE	CS_IE	ES_IE
Receive			SETUP	1	1	party1			
Send			RELEASE_COMPLETE	1			no100		
Receive			STATUS_ENQUIRY	1	1				
Send			STATUS	1	1		no30	UN0	P0

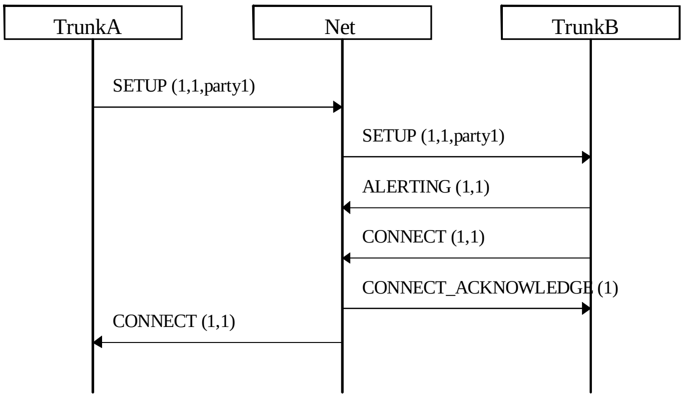
In this example an attempt to set-up a connection with a SETUP that has an erroneous mandatory information element results in the IUT sending a RELEASE_COMPLETE with cause value 100.

A STATUS_ENQUIRY is sent to the IUT to ascertain that it is in the correct state, i.e., N0 and P0.

Figure 12: sending messages with the GUI

5.8.4 Creating MSCs

The MSC dialogue allows the user to instruct the GUI to log input/output as an MSC file that can be displayed by the SDT tool in graphical form according to ITU-T Recommendation Z.120 [9].



This example is the MSC trace of the preamble described in section 8.1

Figure 13: A typical MSC

5.9 Validation of TPs by simulation

Using the GUI described in the previous clauses it was possible to validate the manually-produced TPs specified in ETS 300 771-5 [4] according to the process illustrated in figure 14.

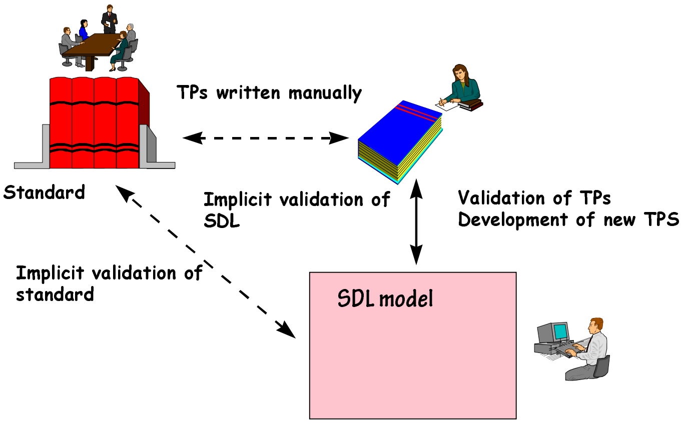


Figure 14: Validation of TPs by simulation

Once the system was setup it took about 8 man-days to validate over 800 TPs. The following table summarizes the results of this process (this does not include the time needed to develop the SDL model (about 2 man-months), or the time needed to build the graphical user interface (about 2 man-weeks)).

Table 1: TP errors found by the validation process

Errors in the TPs (missing or too many parameters, incorrect messages etc.)	51
Errors found in the Q.2931 and Q.2971	19
Errors (bugs) found in the SDL model	20

It is worth noting that this process had the effect of not only validating the TPs (an error rate of 8 % was found in the TPs) but also the SDL model and the standard itself.

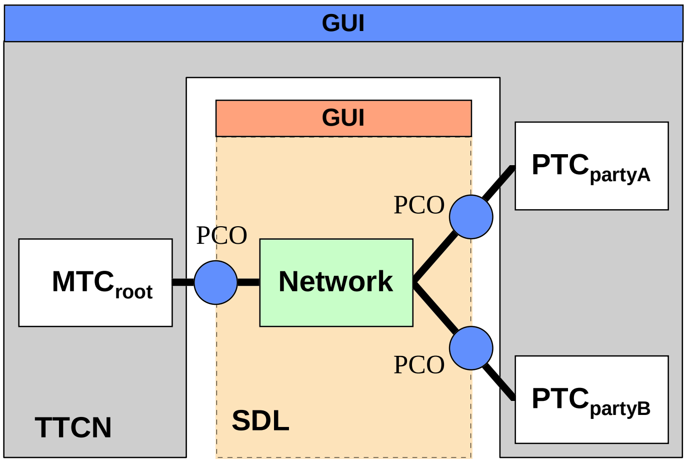
6 SDL model for the validation of TTCN test suites

The test suite validation model was developed from the TP validation model. It was developed and executed on the Telelogic tool, SDT 3.11. It has not been executed on other tools. The TTCN simulation tool used was ITEX 3.11.

The graphical format of this model can be found in annex B. The full SDL specification can be found in annex D.

6.1 Enhancements necessary to the TP model

The SDL model was equipped with a fixed number of channels to the environment to allow interfacing of the SDL simulation with the TTCN and CATG tools (which required a fixed number of access points). This architecture is illustrated in figure 15.



This figure shows the configuration for validating the Network side of the protocol. As the configuration for the User is symetrical the same SDL model can be used for both sides.

The two GUIs in this architecture have not been custom built (as in the TP model) but are the standard GUI of the SDT and ITEX tools.

Figure 15: Architecture for Test Case validation by simulation

Other necessary additions to the original model were:

- all data, messages and information elements were fully defined using ASN.1;
- provision of detailed protocol data checking functions in SDL;
- provision of functions which generate appropriate signals with detailed data contents in reaction to protocol activities;
- provision of an encoding/decoding of protocol messages and information elements (this encoding is not supplied by the SDT and ITEX tools);
- provision of a full functional call processing to simulate switch behaviour, define a mechanism how to provide user and network side simulations in one SDL model with minimal maintenance effort;
- the user side simulation (i.e., the model now included both network and user behaviour).

This work took about 3 man-months to complete.

6.2 Validation of test cases by simulation

Validation by simulation means executing a test suite against a simulated Implementation Under Test (IUT). In our case the SDL model was the IUT and the test suite was the manually produced test suite of ETS 300 771-6 [5]. It involved executing each test case against the SDL model and (manually) checking that the expected results (verdicts) were achieved.

While this form of validation is not as effective as execution against a real IUT it gave some effective results. The following errors in the TTCN specification (network side) were identified:

Table 2: Test Case errors found by the validation process

Parameter order errors (parameters where not in the right order)	33
Procedural errors (race conditions which have invalidated the test case)	12
Parameter value errors (parameters had wrong value)	42

Ignoring the parameter order errors the validation process detected approximately 50 test cases which were erroneous in the 830 test case test suite (i.e., a 6 % error rate). These errors had not been detected by previous manual reviews of the test suite.

NOTE: In these statistics we have ignored the parameter order errors which could have been detected by a more powerful syntax/static semantic checker.

7 Conclusions

Using SDL/MSC based simulation tools as *aids* to the development, documentation and validation of TPs is useful and produces high-quality TPs. The informal expression of TPs in textual format (using templates as is current ETSI practice) accompanied by the corresponding MSCs is especially effective. Because the SDL model is an abstraction of the real protocol it is relatively economical to build. At this level the tools can also have the useful side-effect of helping to develop and validate the corresponding base standards.

By using TTCN/SDL simulation tools the quality of TTCN test suites can be significantly improved. However, it should be noted that building an SDL model of the necessary complexity is probably not economically sensible to do *unless* the SDL model is to be used in other contexts (e.g., CATG, design, pre-implementation, etc.).

The experiments with CATG tools made it evident that for the purposes of the B-ISDN DSS2 CS2 test suites the current tools did not offer a faster alternative to manual development. However, improvements in the tools have been made since this study was performed. It is possible that the next generation of CATG can at least partially assist in the development of test suites. The extent to which this can be done will largely depend on the detail in the relevant SDL model.

NOTE: CATG has been successfully used by ETSI in other applications, namely the generation of test cases for Intelligent Networks Application Protocol (INAP).

Annex A: SDL TP validation model - GR format

Annex A is the graphical rendition of the SDL TP validation model of ETS 300 771-1 [2] combined with ETS 300 443-1 [1] (Q.2931 + Q.2971, network).

This annex is available in a Portable Document Format file (annex_a.pdf contained in archive 9go00icq.lzh) which accompanies the present document.

Annex B:
SDL Test Case validation model - GR format

Annex B is the graphical rendition of the SDL test case validation model of ETS 300 771-1 [2] combined with ETS 300 443-1 [1] (Q.2931 + Q.2971, network and user).

This annex is available in a Portable Document Format file (annex_b.pdf contained in archive 9go00icq.lzh) which accompanies the present document.

Annex C:
SDL TP validation model - source files

Annex C is a package of the SDL source code files for the SDL TP validation model of ETS 300 771-1 [2] combined with ETS 300 443-1 [1] (Q.2931 + Q.2971, network).

This annex is available in electronic format only. It includes the SDL.PR files, SDT source files, code for the Graphical User Interface, simulation code and installation instructions. The SDL source code in annexe C is intended for use with the Telelogic tool SDT version 3.1.

These files are contained in an archive file (annex_c.zip contained in archive 9go00icq.lzh) which accompanies the present document.

Annex D:
SDL Test Case validation model - source files

Annex D is a package of the SDL source code files for the SDL test case validation model of ETS 300 771-1 [2] combined with ETS 300 443-1 [1] (Q.2931 + Q.2971, network and user).

This annex is available in electronic format only. It includes the SDL.PR files, SDT and ITEX source files, simulation code and installation instructions. The SDL source code in annex D is intended for use with the Telelogic tool SDT version 3.1.

These files are contained in an archive file (annex_d.taz contained in archive 9go00icq.lzh) which accompanies the present document.

History

Document history		
V1.1.1	March 1998	Membership Approval Procedure MV 9822: 1998-03-31 to 1998-05-29
V1.1.1	June 1998	Publication

ISBN 2-7437-2217-7
Dépôt légal : Juin 1998