TR 101 279 V1.1.1 (1998-07)

Technical Report

Experiences of the application of SDL and CATG tools for the development of Abstract Test Suites (ATSs);

Reference

DTR/SPS-050174 (buc00ics.PDF)

Keywords

B-ISDN, DSS2, Methodology, SDL, Testing, TTCN, Validation

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

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 SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available **free of charge** from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://www.etsi.fr/ipr or http://www.etsi.org/ipr).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This ETSI Technical report (TR) has been produced by the Signalling Protocols and Switching (SPS) Technical Committee of the European Telecommunications Standards Institute (ETSI).

Introduction

This document adds complementary information to EG 201 022. It describes the experiences of the use of SDL as defined in Z.100 [3] and TTCN as defined in ISO/IEC 9646-3 [4] based **tools** for the development of Abstract Test Suites for B-ISDN DSS2 point-to-multipoint as defined in Q.2971 as modified by ETS 300 771-1 [1].

The tools studied were:

- SDT (Telelogic) for SDL simulation;
- ITEX (Telelogic) for TTCN simulation;
- Link and Autolink (Telelogic) for CATG;
- TTCgeN (Verilog) for CATG;
- TTCN Maker (INTOOLs project) for CATG.

1 Scope

The present document objectively documents the experimental use of various tools for Computer Aided Test Generation from the point of view of testing methodology. It is not intended that this document imply any comparison of the tools, nor is it intended that this document be used as a basis for ETSI recommending the use of one tool or another.

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 300 771-1 (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]".

[2] 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]".

[3] ITU-T Recommendation Z.100 (1994): "CCITT Specification and Description Language (SDL)".

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

[5] ITU-T Recommendation Z.120 (1993): "Message Sequence Chart".

[6] 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".

[7] 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 PIXIT proforma specification for the network".

[8] EG 201 022: "Broadband integrated Services Digital Network (B-ISDN); Digital Suscriber Signalling system 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".

3 Abbreviations

For the purposes of this TR, the following abbreviations apply:

ASN.1	Abstract Syntax Notation one
B-ISDN	Broadband ISDN
CATG	Computer-Aided Test Case Generation
DSS2	Digital Subscriber Signalling System two
INAP	Intelligent Networks Application Protocol
ISDN	Integrated Services Digital Network
IUT	Implementation Under Test
MSC	Message Sequence Chart
SDL	Specification and Description Language
TP	Test Purpose
TTCN	Tree and Tabular Combined Notation
UNI	User Network Interface

4 Use of tools for TP development and validation

4.1 Test Purpose development

None of the tools studied offered a real alternative to the intellectual processes that are applied when producing test purposes manually. The tools supported either:

- *semi-automated* techniques which rely on user interaction with a simulator to generate MSCs as defined in Z.120 [5] which express test purposes; or

- *fully-automated* techniques which systematically base test purposes on single state transitions.

While MSCs can be a useful complement to documenting test purposes they should not be regarded as a complete substitute for textual test purposes, which will often contain additional and necessary information not easily expressible in the MSC format (e.g. verdict assignment). Also, MSCs cannot cope with dynamic alternatives, as in this example from N-ISDN, User Side: Ensure that the IUT in Null call state U00, on receipt of a valid SETUP message with the sending complete information element, sends any of a CALL PROCEEDING, ALERTING or CONNECT message and enters the relevant call state Call Proceeding U09, Call Received U07 or Connect Request U08.

With fully-automated techniques there are two difficulties. In the first case, a single-state transition is not always an adequate expression of purpose where we may wish to express test purposes in terms of requirements not necessarily restricted to a single state transition (i.e., the level of granularity is too restrictive).

In the second case there is a potential for generating very many test purposes. Although the number of test purposes generated can limited by applying sets of criteria, these criteria are often arbitrary (e.g., maximum depth) and do not always bear relation to what a test engineer would normally call a ‘good’ test purpose. Finally, even if test purposes *are* generated automatically they must still be subject to a time-consuming manual review if we are to have full confidence in them.

Conclusion: Using SDL/MSC based tools as *aids* to the development and documentation of test purposes is useful and produces high-quality documentation of test purposes. The informal expression of TPs in textual format (using templates as is current ETSI practice) accompanied by the corresponding MSCs is especially effective. Tools are *not* suitable for automatic generation of TPs.

4.2 Test Purpose validation

Using tools to validate test purposes by simulation proved to be more successful. This was done by checking through simulation the behaviour described by each TP. The work was made easier by the implementation of a simple graphical user interface.

NOTE: EG 201 022 [8] describes the use of SDL simulation techniques to develop and validate TPs.

Once the system was setup it took about 8 man-days to validate 830 test purposes taken from ETS 300 771-5 [6]. The following table summarises 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: Errors found by the TP validation process

Errors in the test purposes (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 has the effect of not only validating the test purposes (an error rate of 8% was noted) but also the SDL model and, by implication, the standard itself. Due to the fact that all three components in the validation process (the standard, the SDL model and the test purposes) were produced by different parties we have confidence that the exercise was more than an academic.

Conclusion: The use of SDL simulation models can be very useful in the development and validation of test purposes. At this level they can also have the useful side-effect of validating the base standards.

5 Use of tools for Test Suite validation

The SDT/ITEX simulators were used to execute a manually written TTCN test suite for the network B-ISDN DSS2 against an enhanced SDL model.

The basis for the protocol simulation was the SDL model for test purpose validation. However, the test purpose validation model had only limited support of protocol data and could not cope with some special protocol situations. The following additions were necessary:

- detailed protocol data (messages, information elements etc.) descriptions in 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 data;
- 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., both network and user).

These tasks were completed in approx. 3 man-months.

NOTE - Much of the SDL work was done on a voluntary basis by Deutsche Telekom

Using the Telelogic ITEX tool, C-code was generated from the test suite. This code could not interwork with the SDT simulation mainly because the data format was not unique. In Tau version 3.11 it is not possible to interchange signals which contain ASN.1 sets with optional fields because there is no tag to distinguish these fields. Therefore the data coding defined in ETS 300 771-1 [1] and ETS 300 443-1 [2] had to be implemented and used as the interchange format between ITEX simulation and SDT simulation. It was necessary to write code for the encoding and decoding of protocol data on the ITEX side.

Other tasks needed to create an executable simulation were:

- provide PIXIT values;
- provide user defined test operations;
- provide protocol data encoding/decoding functions and other tool fixes.

The ITEX fixes that needed doing can be grouped as follows:

- data access errors (CHOICE values, SET fields, wrong type settings); and
- simulation run time errors (logging and scheduling).

The first kind of errors could be fixed in the generated code. The second kind of errors were fixed through a library update provided by Telelogic. These tasks were completed in 1 man-month. Through the provided encoding/decoding, the ITEX and SDT simulation could interwork. All test cases of the network side test suite were executed. Due to time constraints only a few user side test cases were executed. This task took approximately 1 ½ man-months. Through simulation errors in the TTCN specification as well as in the SDL specification were identified.

Table 2: Errors found in the TTCN test suite

Parameter order errors (parameters where not in the right order (these errors could have been detected by a better TTCN checker than ITEX)	33
Procedural errors (race conditions could have invalidated the test case)	12
Parameter value errors (parameters had wrong value)	42

Ignoring the parameter order errors the parallel simulation detected approximately 50 test cases which were erroneous in the 800+ test case test suite (6% error ratio). It is very unlikely that these errors would have been found by a manual review of the test suite.

Conclusion: by using the TTCN/SDL simulation techniques the quality of the TTCN test suite was significantly improved. However, it should be noted that building the SDL model and writing the encoder/decoder interface is probably not economically sensible unless the SDL model is to be used in other contexts (e.g., CATG, implementation etc.).

6 Use of tools for CATG

This section summarises the experiences of evaluating the following CATG tools:

- ITEX Link (Telelogic);
- ITEX Autolink (Telelogic);
- TTCgeN (Verilog);

TTCN Maker (INTOOLs project).

Table 3: Effort allocated to using each tool

Tool	Installation/Education	Test	Total (man-days)
ITEX Link	1	3	4
ITEX Autolink	2	4	6
TTCgen	1	3	4
TTCN Maker	2	3	5
			19

In general the major limitations of the tools were either

- that the tools could not handle complex SDL specifications of the kind that would be typically produced by ETSI; and/or
- that the tool worked but was cumbersome to use and did not offer greatly improved efficiency to an experienced test writer; and/or
- that the additional effort required to manually transform the raw TTCN output from the tool to a level of detail expected in an ETSI standard cancelled out the original benefit of using the tool in the first place.

Another drawback is that tools cannot generate tests for invalid behaviour (unless such behaviour is explicitly programmed into the SDL). In the B-ISDN DSS2 tests for the network side (point-to-multipoint) for example, out of a total of 800 test cases about 600 are for invalid behaviour. Thus, even if CATG were used it would only generate 200 of the necessary 800 test cases.

Conclusion: While CATG tools indicate future possibilities it was evident that for the purposes of B-ISDN testing they did not offer a faster alternative to manual development.

7 The ITEX link tool

ITEX Link is a semi-automatic CATG tool that allows the user to interactively build a test suite from an ITEX editor that is connected to a simulateble SDL model of the system under test. The tool transforms the SDL data to equivalent TTCN data automatically. TTCN SEND events are entered by the user, who must also specify the appropriate constraint. The tool automatically responds with the correct RECEIVE event (or set of events if there is the possibility of more than one) together with the correct constraint(s). In this manner the dynamic behaviour and corresponding PDU constraints are built-up to form the whole test case.

Link provides no support for the generation of test purposes. The manually produced test purposes are interpreted by the human user to guide the interactive development of the corresponding Test Cases.

Link generates correct TTCN test cases, but in the case of Q.2971 as modified by ETS 300 771-1 [1] the dynamic behaviour is reasonably simple, with a limited number of ‘generic’ behaviours. Most tests involve complex variations of message parameters over these generic behaviours. Because of the limited data modelling in the SDL many of these data variations cannot be generated by the tool. It was therefore decided that the tests would be produced more efficiently on a manual basis.

In the case where one has more complex behaviour (for example, in INAP) and where even the experienced test writer cannot anticipate all dynamic outcomes, a tool such as TTCN Link could provide valuable support.

Table 4 shows the extent of the TTCN generation offered by Link. The following items from this list are considered to be major deficiencies:

- link always generates completely flat constraints even if the ASPs (or PDUs) definitions are structured i.e., all structuring information is lost. This is particularly confusing if complex data structures are used;
- derived constraints cannot be used in synchronised mode;
- matching symbols and Test Suite Parameters cannot be used in synchronised mode, i.e., only explicit values may appear in constraints;
- Test Steps and Constraints cannot be parameterised in synchronised mode;
- concurrent TTCN is not supported.

NOTE: Table 4 also gives a general guide to what constitutes a resynchronisable edit (R) and one that is not resynchronisable (NR). The classification Limited Resynchronisation (LR) refers to the cases (e.g., addition of Test Case Variables) where these items may be declared in the ATS without affecting synchronisation but may not used in all circumstances (e.g., as values in constraints or the dynamic part).

Link is a well-integrated tool and is reasonably user-friendly. Facilities such as *Show_SDL* (which animates the path a particular Test Case dynamic behaviour takes through the SDL) and *Show_MSC* (which draws an MSC for any given Test Case behaviour sequence) were found to be quite useful.

The ability (not provided) to name the generated send constraints on the fly would be useful. Telelogic are aware of this request.

Link appears not to place restrictions on the scope of SDL used (i.e., SDL ‘92 is supported).

8 The ITEX Autolink tool

In this tool, test purposes must be developed in the form of MSCs using the SDT simulation and MSC tools. These test purposes are then input to Autolink which generates ‘raw’ TTCN from them. The raw TTCN must be postprocessed in order to make it suitable for standardisation. This post-processing is similar to that needed with the other tools:

- Major tasks:
  - splitting of the non-concurrent (i.e. interleaved) test cases produced by the tool into parallel TTCN behaviour trees and the addition of Configuration tables;
  - addition of missing information in the Constraints (certain parameters are not modelled in the SDL specification);
  - addition of several pre-ambles (the AAL was not fully modelled in the SDL specification);
  - addition of Test Suite Parameters;
  - parameterization of constraints and test steps.

- Minor tasks
  - Re-formatting and re-structuring of the ASN.1 constraints for readability;
  - Re-naming of automatically named objects (for readability);
  - Addition of test steps (not strictly necessary, but reduces the size of the test suite and helps readability);
  - Addition of comments.

By using specially-written scripts to do some of the post-processing it is possible that the overhead could be reduced. There are no figures currently available to indicate what improvements might be expected.

The performance aspects of the Autolink were disappointing. A single test case of medium complexity took over 40 hours to generate. Simpler test cases took anything from ½ - 2 hrs. It is probable that with more powerful hardware (top of the range Sparc) that these times could be significantly reduced. The hardware used in this trial was a Sun Sparc 20 with 128 MB RAM and 256 MB swap.

Autolink appears not to place restrictions on the scope of SDL used (i.e., SDL '92 is supported).

Finally, it has not determined the quality of the CATG test cases with respect to execution in a real test system against a proper IUT. The lack of detail in some instances (due to lack of detail in the SDL model) may give cause for concern. The manually produced test suite compiled on a commercial ATM test system the first-time that it was tried. The ATS is also being evaluated in the field by several manufacturers of ATM equipment. Early feedback indicates that the test suite is executing well in a real environment.

9 The TTCgeN tool

TTCgeN generates test cases entirely automatically. Test Purposes are represented by MSCs. Tools are used to generate complete MSCs from partial MSCs, which have been created manually. These complete MSCs are then used as input then to derive complete MSCs. However the actual effort (e.g., finding suitable starting (partial) MSCs and then setting the boundary conditions to restrict the output of the MSC generator to reflect real situations

TTCgeN supports the test suite generation process as follows:

- for each test purpose MSC, the user binds it with the SDL model to generate a pair (model, observer) that defines the test system. An observer here is the tool-internal representation of the MSC;
- the tool then automatically runs this test system and generates TTCN test cases. TTCgeN is a batch command that takes as input the pair (model, observer), executes the co-simulation which may be seen as an exhaustive exploration of the behaviour of the SDL model strictly constrained by the test purpose observer, and derives the test cases from this reduced state graph;
- the result comes out as a 'testcase.mp' file that can be visualized using a TTCgeN facility, or that can be processed with a TTCN editor such as ITEX.

Table 5 shows the extent of the TTCN generation offered by TTCgeN. The following items from this list are considered to be major deficiencies:

- only the top-level of structured PDU (or ASP) definitions are translated. All sub-structure definitions must be created manually;
- only the top-level of structured PDU (or ASP) constraints are translated. All sub-structure constraints must be created manually;
- derived constraints cannot be generated;
- matching symbols and the use of Test Suite Parameters in constraints may not be generated, i.e., only explicit values appear in constraints;
- parameterised Test Steps and Constraints cannot be generated;
- timers are handled incorrectly;
- concurrent TTCN is not supported.

NOTE: Where *Same as Link* is used in table 5 all references to resynchronisation in table 1 should be ignored. The concept of synchronisation with the SDL is not relevant to TTCgeN.

It is clear that TTCgeN is a prototype and will need some tuning if it is to be useful product. The tool is not integrated with ObjectGEODE and the TTCgeN output must be input to a TTCN editor such as ITEX in order to continue with manual editing. This makes TTCgeN less easy to work with.

TTCgeN does not support SDL '92 and does not always handle signal parameters and abstract data types correctly, e.g., the type *Charstring* is ignored.

No system problems were encountered during this study.

10 The TTCN Maker tool

This tool generates both the test purposes and the test cases automatically. The generation is based on a single state transition. However, this is not always an adequate expression of purpose where we may wish to express test purposes in terms of requirements not necessarily restricted to a single state transition (i.e., the level of granularity is too restricted). Also, there is a potential for generating very many test purposes. The number of test purposes generated is limited by applying a set of criteria. These criteria are often arbitrary (e.g., maximum depth) and do not always bear relation to what a test engineer would normally call a ‘good’ test purpose. Finally, even if test purposes *are* generated automatically they must still be subject to a time-consuming manual review if one is to have full confidence in them.

Table 6 shows the extent of the TTCN generation offered by Link. The following items from this list are considered to be major deficiencies:

- it does not support many of the SDL ’92 constructs that are extensively used in the SDL model; and
- even if the tool performs well as a test generator it can only generate tests for one process at a time. In the SDL for Q.2971 as modified by ETS 300 771-1 [1], for even a simple testing situation of say two parties, at least nine concurrently executing processes are created. It is simply not feasible to generate separate pieces of TTCN for each of these processes and then merge them into a single test suite. Neither is it feasible to re-write the SDL as a single process as it is exactly the concurrent properties that are wishes to model.

The tool has good performance characteristics and appears robust.

11 CATG tool summaries

Table 4: TTCN Link summary

1	Test Purposes	Not generated	Test purposes must be produced manually before using the tool. They are interpreted by the human user as he/she interactively generates the Test Cases.	-
2	Test Structure Test Groups	Not generated	Structuring of the Test Suite into Test Groups must be done manually. The Test Suite indexes can be automatically generated by ITEX.	R
3	Concurrent TTCN	Not generated	Parallel test component declarations and the configuration declarations must be defined manually. Where an SDL specification defines behaviour at more than one PCO the events occurring at the different PCOs are interleaved in a TTCN Link Test Case. This means that that all behaviour trees must manually be split into their component trees, in our example this would mean two PTCs, one for A and one for B. Note that it should possible to write a script that converts the interleaved TTCN.MP to Concurrent TTCN. MTC activity such as CREATE and the specification of co-ordination between the PTCs has to be implemented manually.	LR NR NR
4	Type Definitions	**Generated**	*Newtypes* defined in the SDL (including *struct* types) are translated to corresponding ASN.1 types in TTCN.	LR
5	User Defined Ops	Not generated		LR
	Test Suite Params	Not generated	All Test Suite Parameters must be declared manually. Even when declared these parameters have no semantic connection to the SDL, i.e. TTCN Link does not recognise them. Test Suite Parameters should not be used as values in constraints if synchronisation with TTCN Link is required.	LR
6	Test Suite Constants Test Suite Variables Test Case Variables	Not generated	Same as for Test Suite Parameters.	LR
7	PCOs	**Generated**	All channels to the environment are treated as PCOs. Traffic on internal channels is not explicitly seen in the Test Case.	NR
8	Test Case Behaviour	**Generated** Semi-automatic	TTCN Link generates Test Cases semi-automatically. The user manually enters the desired *send* events (indicated in the Test Purpose) and the system automatically responds with the correct *receive* events. TTCN Link generates received alternatives according to the (reverse!) order in which the corresponding inputs appear in the SDL. This may be a problem in TTCN where the ordering of events in a single set of alternatives is significant.	NR
9	Timers	**Generated**	Timer operations (such as START and CANCEL) may be manually added to event lines. Timeouts are automatically generated (where applicable). It is not allowed to have Timer Operations on separate event lines.	LR
10	Assignments & Qualifiers	Not generated	Assignments and qualifiers (i.e. Boolean expressions) may be manually added to event lines but they do not have any semantic connection to the SDL, i.e. TTCN Link ignores them. It is not allowed to have Assignments or Qualifiers on separate event lines.	LR

11	REPEAT GOTO	Not generated	There is no concept of repeated behaviour in TTCN Link.	NR
12	Test Steps	Not generated	Test Steps may be created by manually copying behaviour from Test Cases into Test Steps. However, TTCN Link requires that the attached Test step appear on a separate line and may not be an alternative among other alternatives, e.g., the following is not allowed: A? PDU1 + TestStep1	R
13	Default Behaviour	**Generated** (Limited)	TTCN Link generates a single default behaviour for the whole test suite comprising ?OTHERWISE (for each PCO) and a general ?TIMEOUT. More complex defaults must be created manually.	R
14	Parametrisation	Not generated	Test steps cannot be parameterised. Constraints cannot be parameterised.	NR NR
15	ASP definitions	**Generated**	All SDL signals on channels to the environment are translated to ASN.1 ASPs (See also PDUs).	
16	PDU definitions	**Generated**	If it is required to express the ATS in terms of PDUs rather than ASPs then the ASP definitions must be manually cut and pasted into the PDU definitions in ITEX. If it is required that the PDUs are carried in ASPs then the ASPs must be defined manually and all send/receive events updated accordingly, possibly through the use of aliases. Because signal parameters in SDL are not named TTCN Link uses dummy names (e.g., integer1, charstring2) to identify names of top-level ASP parameters. These must be changed manually in both the ASP/PDU definitions and constraints.	R NR NR
17	Constraints	**Generated** (Limited)	Constraints are expressed as ASN.1 values. Unfortunately TTCN Link flattens all constraints. That is, any structuring of ASPs/PDUs is not reflected in the constraint. If it is required to have structured constraints the structuring must be re-created manually after the generation process. During generation of send events the system will prompt for the required input. This may either be a new constraint, which can be defined on-the-fly or an existing constraint. Received constraints are automatically generated by the system. They are given generated names which will usually mean that a manual re-naming must be done.	NR R
18	Matching symbols	Not generated	Only explicit values may appear in constraints. TTCN matching mechanisms such as ranges and wildcards may not be used if synchronisation with TTCN Link is to be maintained.	NR
19	Derived constraints	Not generated	Derived constraints are not generated. This must be done manually if required. Note that a bug in ITEX does not allow the use of multiple REPLACE or OMIT in an ASN.1 derived constraint(!).	NR
20	Aliases	Not generated	Aliases are not generated. This must be done manually if required.	NR
21	Verdicts	Not generated	The only verdicts assigned by TTCN Link are FAIL on the ?OTHERWISE and INCONC on the ?TIMEOUT in the default behaviour generated by TTCN Link. PASS and other verdicts (including preliminary) results are not generated, these must be added manually.	R
22	Comments	Not generated	Comments in the SDL are not reflected in the TTCN. All comments must be added manually.	R

Table 5: TTCgeN summary

1	Test Purposes	**Generated** (Limited)	Test purposes are in the form of system level MSCs. The MSCs were generated semi-automatically from incomplete MSCs using the ObjectGEODE simulator.
2	Test Structure Test Groups	Not generated	Same comment as for TTCN Link
3	Concurrent TTCN	Not generated	Same comment as for TTCN Link
4	Type Definitions	Not generated	*Newtypes* defined in the SDL (including *struct* types) are not translated.
5	User Defined Ops	Not generated	Same comment as for TTCN Link
	Test Suite Params	Not generated	Same comment as for TTCN Link
6	Test Suite Constants Test Suite Variables Test Case Variables	Not generated	Same comment as for TTCN Link
7	PCOs	Not Generated	PCOs are not generated. This needs to be fixed.
8	Test Case Behaviour	**Generated**	Once the Test Purposes as MSCs have been created with the simulator, TTCgeN runs fully automatically in the form of a batch command. The output is standard TTCN-IS MP format on a Test Case basis. These MP files can then be (manually) merged into a Test Suite and input to ITEX or TTCN export.
9	Timers	**Generated**	Timer operations (such as START and CANCEL) added later automatically. However, TTCgeN appears to generate a START Timer operation for all Send events and a corresponding CANCEL Timer for all Receive events. This needs to be fixed.
10	Assignments & Qualifiers	Not generated	Same comment as for TTCN Link
11	REPEAT GOTO	Not generated	Same comment as for TTCN Link
12	Test Steps	Not generated	Same comment as for TTCN Link
13	Default Behaviour	**Generated** (Limited)	Same comment as for TTCN Link
14	Parametrisation	Not generated	Same comment as for TTCN Link
15	ASP definitions	**Generated**	If it is required to express the ATS in terms of ASPs rather than PDUs then the PDU definitions must be manually cut and pasted into the ASP definitions in ITEX. If it is required that the PDUs are carried in ASPs then the ASPs must be defined manually and all send/receive events updated accordingly, possibly through the use of aliases.
16	PDU definitions	**Generated**	All SDL signals on channels to the environment are translated to TTCN PDUs (See also ASPs).
17	Constraints	**Generated** (Limited)	Constraints are expressed in tabular (not ASN.1) format. Because sub-types are not translated, only the top-level constraints are generated.
18	Matching symbols	Not generated	Same comment as for TTCN Link
19	Derived constraints	Not generated	Same comment as for TTCN Link
20	Aliases	Not generated	Same comment as for TTCN Link
21	Verdicts	**Generated** (Limited)	TTCgeN assigns the following verdicts: PASS for the matching SUT behaviours, FAIL for the impossible SUT behaviours, INCONC for the unexpected SUT behaviours Preliminary verdict results are not generated and must be added manually
22	Comments	Not generated	Same comment as for TTCN Link

Table 6: TTCN Maker summary

1	Test Purposes	**Generated**	Test purposes are based on single state transitions. Directives such as hiding of certain signals can be used to limit the test purposes.
2	Test Structure Test Groups	**Generated**	Directives can be applied to generate test group paths. Like the test purposes these are state and transition oriented. The Test Suite indexes can be automatically generated by ITEX.
3	Concurrent TTCN	Not generated	Same comment as for TTCN Link
4	Type Definitions	Not generated	Same comment as for TTCN Link
5	User Defined Ops	Not generated	Same comment as for TTCN Link
	Test Suite Params	Not generated	Same comment as for TTCN Link
6	Test Suite Constants Test Suite Variables Test Case Variables	**Generated**	These are derived from constants and variables that appear in the SDL. Directives can be used to indicate whether or not an SDL constant/variable shall be reflected in the TTCN.
7	PCOs	**Generated**	All system channels are treated as potential PCOs. A directive can be used to hide PCOs, if wished.
8	Test Case Behaviour	**Generated**	Once the directives have been set the tool generates TTCN.MP V8.3. The tool is very fast.
9	Timers	**Generated**	NOAC Timer operations (START and CANCEL) are added automatically.
10	Assignments & Qualifiers	Not generated	Same comment as for TTCN Link
11	REPEAT GOTO	Not generated	Same comment as for TTCN Link
12	Test Steps	Not generated	The contents of test steps for preambles and check-state sequences are not generated. These must be added manually.
13	Default Behaviour	**Generated** (Limited)	Same comment as for TTCN Link
14	Parametrisation	**Generated**	Constraints are parameterised.
15	ASP definitions	Not generated	If it is required to express the ATS in terms of ASPs rather than PDUs then the PDU definitions must be manually cut and pasted into the ASP definitions in ITEX. If it is required that the PDUs are carried in ASPs then the ASPs must be defined manually and all send/receive events updated accordingly, possibly through the use of aliases.
16	PDU definitions	**Generated**	All SDL signals on channels to the environment are translated to TTCN PDUs (See also ASPs). The tool also generates SDL Timeouts as PDUs. As these are not considered to be signals in a ‘real’ protocol these PDUs need to be deleted.
17	Constraints	**Generated**	The tool generates a base constraint for each PDU. Individual constraints are built using the TTCN REPLACE mechanism.
18	Matching symbols	Not generated	Same comment as for TTCN Link
19	Derived constraints	**Generated**	See 17
20	Aliases	Not generated	Same comment as for TTCN Link
21	Verdicts	**Generated**	TTCN Maker assigns PASS verdicts in the body. FAIL is assigned in the default behaviour and INCONC is associated with the NOAC timeouts.
22	Comments	Not generated	Same comment as for TTCN Link

12 Errors/ambiguities found in Q.2931 and Q.2971

One of the beneficial side-effects of this work number discovered a number of errors/ambiguities in Q.2931 as modified by ETS 300 443-1 [2] and Q.2971 as modified by ETS 300 771-1 [1]. A list of these errors was submitted to WG SPS 5 for further submission to ITU-T.

History

Document history		
V1.1.1	July 1998	Publication

ISBN 2-7437-2347-5
Dépôt légal : Juillet 1998