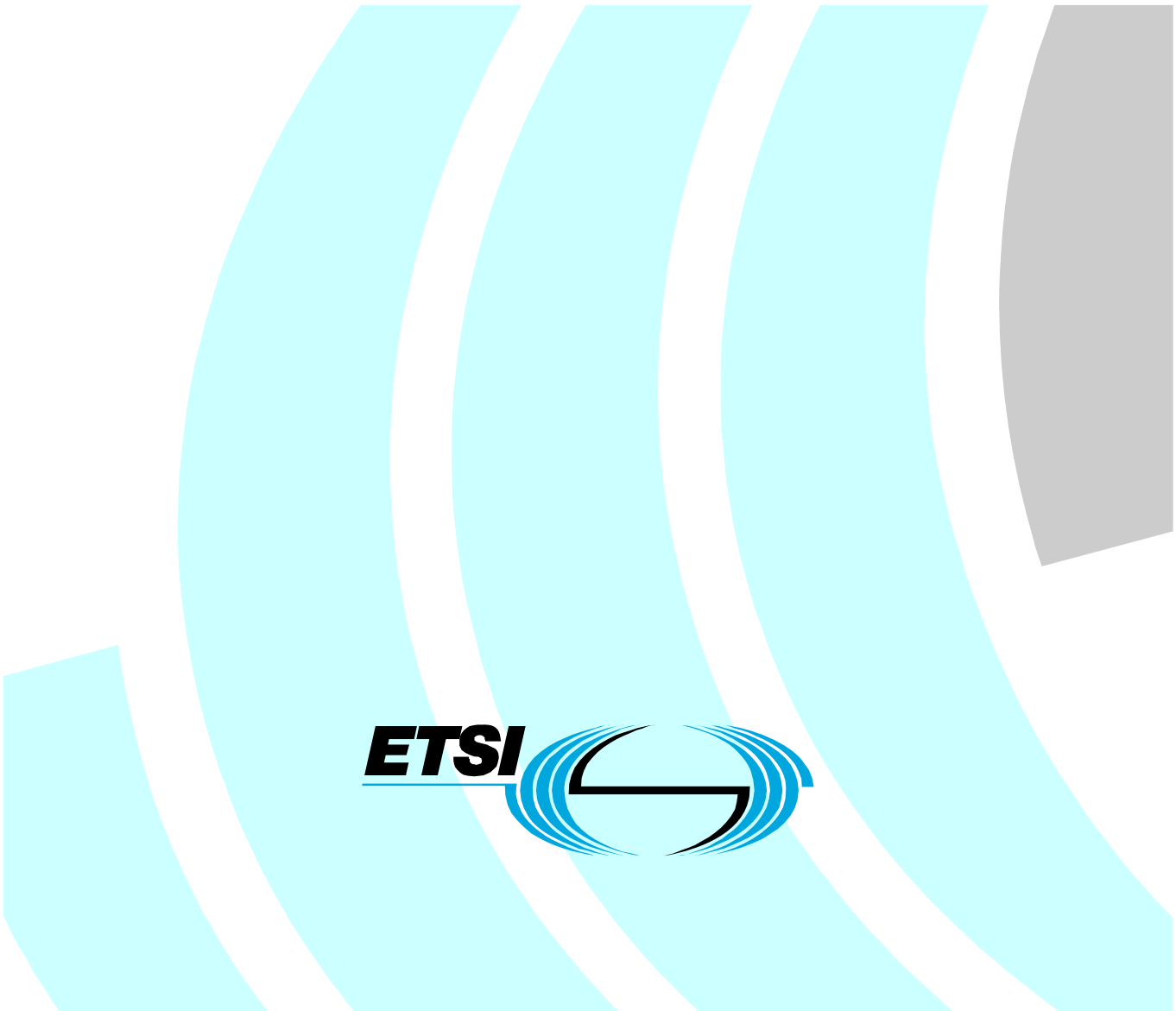
ETSI TS 102 995 V1.1.1 (2010-11)

Technical Specification

Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Proforma for TTCN-3 reference test suite



Reference
DTS/MTS-00115ED111 T3RefATS

Keywords
methodology, MTS, testing, TTCN, IXIT, ICS, 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, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Copyright Notification

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).

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 ETSI 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 Technical Specification (TS) has been produced by ETSI Technical Committee Methods for Testing and Specification (MTS).

The present document is the framework for a TTCN-3 Reference Test Suite.

1 Scope

The present document proposes a proforma for creating ICS, Test Purposes, and TTCN-3 test cases to assess compliance of TTCN-3 tools to TTCN-3 core language including also an estimate for integrating tool vendor tests into the framework. The present document has been developed to provide the essential structure, naming, style, etc. information to develop conformance test specifications for the TTCN-3 language. In the present document only the core language features, specified in ES 201 873-1 [1] have been considered but not the tool implementation (see [i.1] and [i.2]), language mapping (see [i.3], [i.4] and [i.5]) and language extension (see e.g. [i.6], [i.7]and [i.8]) aspects. The test notation used in the ATS attached in a zipped file is in TTCN-3 and it is part of the present document.

The following test specification - and design considerations can be found in the body of the present document:

- the overall test suite structure;
- the testing architecture;
- the test methods and port definitions;
- the test configurations;
- the design principles, assumptions, and used interfaces to the TTCN3 tester (System Simulator);
- TTCN styles and conventions;
- the language ICS (Implementation Conformance Statement);
- the partial Implementation eXtra Information for Testing (IXIT) proforma;
- the Test Suite Structure and Test Purposes (TSS&TP);
- the modules containing the TTCN-3 ATS.

Annex A provides the language ICS (Implementation Conformance Statement).

Annex B provides the Test Suite Structure and Test Purposes (TSS&TP).

Annex C provides the partial Implementation eXtra Information for Testing (IXIT) Proforma of the ATS.

Annex D provides the Testing and Test Control Notation (TTCN-3) part of the ATS.

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the reference document (including any amendments) applies.

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

NOTE: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

2.1 Normative references

The following referenced documents are necessary for the application of the present document.

[1] ETSI ES 201 873-1: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1: TTCN-3 Core Language".

[2] ETSI ES 201 873-10: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 10: TTCN-3 Documentation Comment Specification".

[3] ETSI TS 102 351: "Methods for Testing and Specification (MTS); Internet Protocol Testing (IPT); IPv6 Testing: Methodology and Framework".

[4] ISO/IEC 9646-1 (1992): "Information Technology - Open Systems Interconnection - Conformance Testing Methodology and Framework - Part 1: General concepts".

[5] ISO/IEC 9646-7 (1994): "Conformance testing methodology and framework - Part 7: Implementation Conformance Statement".

2.2 Informative references

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI ES 201 873-5: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 5: TTCN-3 Runtime Interface (TRI)".

[i.2] ETSI ES 201 873-6: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 6: TTCN-3 Control Interface (TCI)".

[i.3] ETSI ES 201 873-7: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 7: Using ASN.1 with TTCN-3".

[i.4] ETSI ES 201 873-8: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 8: The IDL to TTCN-3 Mapping".

[i.5] ETSI ES 201 873-9: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 9: Using XML schema with TTCN-3".

[i.6] ETSI ES 202 781: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; TTCN-3 Language Extensions: Configuration and Deployment Support".

[i.7] ETSI ES 202 784: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; TTCN-3 Language Extensions: Advanced Parameterization".

[i.8] ETSI ES 202 785: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; TTCN-3 Language Extensions: Behaviour Types".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ISO/IEC 9646-1 [4], ISO/IEC 9646-7 [5], ES 201 873-1 [1] (TTCN-3) and the following apply:

Abstract Test Case (ATC): complete and independent specification of the actions required to achieve a specific test purpose, defined at the level of abstraction of a particular Abstract Test Method, starting in a stable testing state and ending in a stable testing state

Abstract Test Method (ATM): description of how an IUT is to be tested, given at an appropriate level of abstraction to make the description independent of any particular realization of a Means of Testing, but with enough detail to enable abstract test cases to be specified for this method

Abstract Test Suite (ATS): test suite composed of abstract test cases

ICS proforma: document, in the form of a questionnaire, which when completed for an implementation or system becomes the ICS

Implementation Conformance Statement (ICS): statement made by the supplier of an implementation claimed to conform to a given specification, stating which capabilities have been implemented

Implementation eXtra Information for Testing (IXIT): statement made by a supplier or implementor of an IUT which contains or references all of the information related to the IUT and its testing environment, which will enable the test laboratory to run an appropriate test suite against the IUT

Implementation Under Test (IUT): implementation of one or more OSI protocols in an adjacent user/provider relationship, being part of a real open system which is to be studied by testing

IXIT proforma: document, in the form of a questionnaire, which when completed for the IUT becomes the IXIT

Means Of Testing (MOT): combination of equipment and procedures that can perform the derivation, selection, parameterization and execution of test cases, in conformance with a reference standardized ATS and can produce a conformance log

Point of Control and Observation (PCO): point within a testing environment where the occurrence of test events is to be controlled and observed, as defined in an Abstract Test Method

Pre-Test Condition: setting or state in the IUT which cannot be achieved by providing stimulus from the test environment

3.2 Abbreviations

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

ASP Abstract Service Primitive
NOTE: Exchanged between entities inside the TS or between the user of the ATS (operator) and the TS.

ATC Abstract Test Case
ATM Abstract Test Method
ATS Abstract Test Suite
EBNF Extended Backus Naur Form
ETS Executable Test Suite
ICS Implementation Conformance Statement
IUT Implementation under Test
IXIT Implementation eXtra Information for Testing
MOT Means Of Testing
SUT System Under Test
TC Test Case
TCI TTCN-3 Control Interface
TP Test Purpose
TRI TTCN-3 Runtime Interface
TS Test System
TSS Test Suite Structure
TSS&TP Test Suite Structure and Test Purposes
TTCN-3 Testing and Test Control Notation edition 3

4 Abstract Test Method (ATM)

This clause describes the ATM used to test the conformance of TTCN-3 tool implementations as described in part 1 of the TTCN-3 core language standard ES 201 873-1 [1]. In the ATM, we work on two levels:

- The TTCN-3 tool level. In TTCN-3 conformance tests, it is the TTCN-3 tool which is under test, i.e. the IUT. However, unlike in protocol conformance testing, it is not standardized how test inputs, i.e. TTCN-3 modules, are provided. Neither are there any standardized interfaces to monitor the reaction of the TTCN-3 tool to the test input. Outputs can only be observed indirectly by monitoring tool outputs such as tool specific command line information, graphical user interfaces, or test execution logs. The tool output is processed further in the tool output evaluation level in order to derive the tool conformance verdicts.

- The TTCN-3 tool output evaluation level. Here, the output of a TTCN-3 tool is indirectly observed, e.g. rejection of TTCN-3 code due to a compile-time error in a command line notification, logging of one or multiple test verdicts in a tool specific window or an execution trace. The observation is evaluated to assess the tool conformance as a result of stimulating the tool with the TTCN-3 modules. Compliance or support of the logging interface specified as part of the TTCN-3 Control Interface standard (TCI) is not required.

NOTE: The loading of the TTCN-3 modules and presentation of the output by the TTCN-3 tools is beyond the scope of the present document.

The ATS document contains the test inputs, i.e. TTCN-3 modules, for TTCN-3 tools do not automate the execution of TTCN-3 tool conformance tests. TTCN-3 tool conformance test decisions shall be made on the basis of expected outputs as specified in the test purposes provided in the documentation and as part of the documentation of TTCN-3 tests in the ATS. Three different tool output classifications for TTCN-3 inputs exist:

- Rejection as invalid, i.e. the TTCN-3 input is declared syntactically or semantically incorrect by the tool. This can either happen at compile-time or at runtime.

- Rejection to execute, i.e. an ETS is produced from the test input, but an execution does not take place.

- Execution with results, i.e. the compiled or interpreted TTCN-3 code is executed and different kinds of outputs are produced that can be subject of an evaluation, for example, a logged TTCN-3 test verdict in a test execution trace (none, pass, fail, inconc) in a file or the console output. The respective tool outputs must specify the expected execution results in order to be able to evaluate whether the conformance test is successful.

A TTCN-3 tool conformance test can attempt to trigger every kind of such outputs in a controlled way, i.e. a test input that is rejected as invalid does not imply a failing conformance test verdict, but instead results in a pass verdict for the conformance test if the test is designed to trigger the rejection. More generally: a TTCN-3 tool conformance test passes if the tool output corresponds to the expected output. The range of expected outputs is described by the tool output classification above.

5 The ATS development process

5.1 Requirements and test purposes

For each test purpose there is a table defined in Annex B. The requirements applicable to this TP are given by a reference to ES 201 873-1 [1]. There are no explicit formulations of requirements.

5.2 ATS structure

5.2.1 Test case grouping

The ATS structure defined in Table 1 is based on the structuring of Test Purposes in Annex B. The group names in columns 1 to 3 of Table 1 are those assigned in the ATS; they are based on the names provided in Annex B, but use the naming conventions defined for the ATS (see Clause 5.3.2.2). The test case identifier naming scheme differentiates between positive and negative tests as well as syntactical and semantics tests.

- Syntactical tests are tests that refer to Annex A of ES 201 873-1 [1]. They include pure syntactical tests and tests regarding the static semantics to the degree of detail that Annex A provides.
- Semantic tests are tests that refer to the checking of properties regarding the static and dynamic semantics of TTCN-3 according to the specific clauses of ES 201 873-1 [1].
- Positive tests are tests that shall work with a standards compliant TTCN-3 tool.
- Negative tests are tests that shall not work with a standards compliant TTCN-3 tool.

The test cases shall conform to the following correctness rules:

- Negative syntactic tests shall be correct with respect to the TTCN-3 EBNF and the static semantics of TTCN-3, but violate only one specific TTCN-3 EBNF rule or static semantic rule specified in Annex A of ES 201 873-1 [1]. They shall not produce an ETS.
- Positive syntactic tests shall be correct with respect to the TTCN-3 EBNF and the static semantics of TTCN-3. They may produce an ETS and if it contains a control-part or a test case, it should be executed.
- Negative semantic tests shall be correct with respect to the TTCN-3 EBNF and the static semantics of TTCN-3, but violate the semantics of one specific text clause of ES 201 873-1 [1]. They may produce an ETS. If an ETS is produced and if it contains a control-part or a test case, it should be executed.
- Positive semantic tests shall be correct with respect to the TTCN-3 EBNF, the static semantics of TTCN-3, and the respective text clauses of ES 201 873-1 [1]. They shall produce an ETS. If an ETS is produced and if it contains a control-part or a test case, it should be executed.

The test case identifiers and their group index do not imply the correct execution order of a TTCN-3 tool conformance test. Dependencies among test cases are expressed in the test purposes and by the structure of the ATS.

Table 1: Example ATS structure of positive tests

Group	Subgroup	Group Index
Basic language elements	Identifiers and keywords	Syn_0501_Identifier
	Identifiers and keywords	Sem_0501_Identifier
	Scope rules	Syn_0502_Scopes
	Scope rules	Sem_0502_Scopes
	Ordering of language elements	Syn_0503_Ordering
	Ordering of language elements	Sem_0503_Ordering
	Parameterization	Syn_0504_Parameterization
	Parameterization	Sem_0504_Parameterization
	Cyclic Definitions	Syn_0505_Cyclic
	Cyclic Definitions	Sem_0505_Cyclic
		Sem_0505_Cyclic

Table 2: Example ATS structure of negative tests

Group	Subgroup	Group Index
Basic language elements	Identifiers and keywords	NegSyn_0501_Identifier
	Identifiers and keywords Scope rules	NegSem_0501_Identifier NegSyn_0502_Scopes
	Scope rules Ordering of language elements	NegSem_0502_Scopes NegSyn_0503_Ordering
	Ordering of language elements Parameterization	NegSem_0503_Ordering NegSyn_0504_Parameterization
	Parameterization Cyclic Definitions	NegSem_0504_Parameterization NegSyn_0505_Cyclic
	Cyclic Definitions	NegSem_0505_Cyclic NegSem_0505_Cyclic

5.2.2 Test case identifiers

The test case names are built up according to the following scheme:

<"TC">"_"<Group index>"_"<TC number>

where:

a) double quotes (") are used to enclose literal strings;

b) <Group index> containing positive and negative syntactic and semantic test, refers to ES 201 873-1 [1] clause numbers and names;

c) <TC number> is a running 3-digit decimal number, starting in each subgroup path with "001".

EXAMPLE: TC_Syn_0501_Identifier_001

i) The example refers to a positive syntactical identifier and keyword test case.

ii) It is the first test case of this group/subgroup.

NOTE 1: This naming scheme corresponds to the TP identifiers and test case names as defined in Annex B.

NOTE 2: The TP identifier of TC_Syn_0501_Identifier_001 is TP_Syn_0501_Identifier_001.

5.3 ATS specification framework

5.3.1 ATS library

Table 3 shows the organization of the ATS as library of modules. In general, every test case shall be placed in a separate module. There are exceptions where it is essential to put more than one test case into a module, like tests related to scope or test cases for importing and visibility.

Table 3: Library of modules (proforma table)

Module Class	Module Id	Description
	Syn_0501_Identifier_001	Main TTCN-3 module with control-part.
	NegSyn_0501_Identifier_001	Main TTCN-3 module with control-part.

5.3.2 Use of TTCN-3

5.3.2.1 General

TTCN-3, as defined in ES 201 873-1 [1], is used as the ATS specification language.

A number of requirements have been identified for the development and production of the TTCN-3 specification for the ATS:

1) Top-down design.

2) A uniquely defined testing architecture and test method.

3) Uniform TTCN-3 style and naming conventions.

4) Human-readability.

5) The TTCN-3 specification shall be feasible, implementable, compilable, and maintainable.

6) Test cases shall be designed in a way to be easily adaptable, upwards compatible with the evolution of the base protocol and protocol interworking of future releases.

7) The test declarations, data structures, and data values shall be largely reusable.

8) Modularity and modular working method.

9) Minimizing the requirements of intelligence on the emulators of the lower testers.

10) Giving enough design freedom to the test equipment manufacturers.

Fulfilling these requirements should ensure the investment of the TTCN-3 implementation vendors and users of the ATS having stable testing means for a relatively long period.

5.3.2.2 TTCN-3 naming conventions

Like in other software projects using a programming language, the use of naming conventions supports or increases:

a) the readability;

b) the detection of semantic errors;

c) the shared work of several developers;

d) the maintainability.

The naming conventions applied to Reference Test suite ATS are based on the following underlying principles:

- when constructing meaningful identifiers, the general guidelines specified for naming in Clause 9 of TS 102 351 [3] should be followed;
- the names of TTCN-3 objects being associated with standardized data types (e.g. in the base protocols) should reflect the names of these data types as close as possible (of course not conflicting with syntactical requirements or other conventions being explicitly stated);
- the subfield names of TTCN-3 objects being associated with standardized data type should also be similar to corresponding element names in the base standards (be recognizable in the local context);
- in most other cases, identifiers should be prefixed with a short alphabetic string (specified in Table 4) indicating the type of TTCN-3 element it represents;
- prefixes should be separated from the body of the identifier with an underscore ("_");
- only test case names, module names, data type names, and module parameters should begin with an upper-case letter. All other names (i.e. the part of the identifier following the prefix) should begin with a lower-case letter.

Table 4 specifies the naming guidelines for each element of the TTCN-3 language indicating the recommended prefix and capitalization.

Table 4: TTCN-3 naming convention

Language element	Naming convention	Prefix	Example	Notes
Module	Use upper-case initial letter	*none*	IPv6Templates	
TSS grouping	Use all upper-case letters as specified in Clause 7.1.2.1.1	*none*	TP_RT_PS_TR	
Item group within a module	Use lower-case initial letter	*none*	messageGroup	
Data type	Use upper-case initial letter	*none*	SetupContents	
Message template	Use lower-case initial letter	m_	m_setupInit m_setupBasic	Note 1
Message template with wildcard or matching expression	Use lower-case initial letters	mw_	mw_anyUserReply	Note 2
Signature template	Use lower-case initial letter	s_	s_callSignature	
Port instance	Use lower-case initial letter	*none*	signallingPort	
Test component ref	Use lower-case initial letter	*none*	userTerminal	
Constant	Use lower-case initial letter	c_	c_maxRetransmission	
External constant	Use lower-case initial letter	cx_	cx_macId	
Function	Use lower-case initial letter	f_	f_authentication()	
External function	Use lower-case initial letter	fx_	fx_calculateLength()	
Altstep (incl. Default)	Use lower-case initial letter	a_	a_receiveSetup()	
Test case	Use numbering as specified in Clause 5.2.2	TC_	TC_COR_0009_47_ND	
Variable (local)	Use lower-case initial letter	v_	v_macId	
Variable (defined within a component)	Use lower-case initial letters	vc_	vc_systemName	
Timer (local)	Use lower-case initial letter	t_	t_wait	
Timer (defined within a component)	Use lower-case initial letters	tc_	tc_authMin	
Module parameter	Use all upper case letters	*none*	PX_MAC_ID	Note 3
Parameterization	Use lower-case initial letter	p_	p_macId	
Enumerated Value	Use lower-case initial letter	e_	e_syncOk	

NOTE 1: This prefix must be used for all template definitions which do *not* assign or refer to templates with wildcards or matching expressions, e.g. templates specifying a constant value, parameterized templates without matching expressions, etc.
NOTE 2: This prefix must be used in identifiers for templates which either assign a wildcard or matching expression (e.g. ?, *, value list, ifpresent, pattern, etc.) or reference another template which assigns a wildcard or matching expression.
NOTE 3: In this case it is acceptable to use underscore as a word delimiter.

5.3.2.3 TTCN-3 comment tags

Any TTCN-3 definition in the Test Suite Repository or Library should contain embedded comment tags, according to ES 201 873-10 [2]. These comment tags can be used by tools to extract information from the TTCN-3 code to create, for example, a HTML-based reference documentation.

Comment tags which cover one or more lines should be specified using block comments, as illustrated:

```
/* -----------------------
 * @desc This line of text is now identified as a description
 *       which covers multiple lines
 * -----------------------*/
```

Comments tags specified within a single line may be specified using line comments, as illustrated:

```
// @author John Doe
```

or:

```
/* @author John Doe */
```

Table 5 lists the tags that can be used in ETSI TTCN-3 test specifications with a short description of the intended use of each tag.

NOTE: Tools may also extract other information from the TTCN-3 code based, for example, on TTCN-3 keywords. The definition of that extraction is beyond the scope of the present document.

Table 5: TTCN-3 comment tags

Tag	Description
@author	This tag should be used to specify the names of the authors or an authoring organization which either has created or is maintaining a particular piece of TTCN-3 code.
@desc	This is probably the most import of all the tags. It should be used to describe the purpose of a particular piece of TTCN-3 code. The description should be concise yet informative and describe the function and use of the construct.
@remark	This tag may be used to add additional information, such as highlighting a particular feature or aspect not covered in the description.
@img	This tag may be used to associate images with a particular piece of TTCN-3 code.
@see	This tag may be used to refer to other TTCN-3 definitions in the same or another module.
@url	This tag should be used to associate references to external files or web pages with a particular piece of TTCN-3 code, e.g. a protocol specification or standard.
@return	This tag should only be used with functions. It is used to provide additional information on the value returned by the given function.
@param	This tag is used to document the parameters of parameterized TTCN-3 definitions.
@version	This tag is used to state the version of a particular piece of TTCN-3 code.
@purpose	This tag is used to state the purpose of a particular piece of TTCN-3 code.

The following provides some basic guidelines on the usage of tags for specific TTCN-3 definitions:

- each TTCN-3 module should use the *@author*, *@version* and *@desc* tags;
- the *@desc* tag should be used with all TTCN-3 definitions. However, this should not be taken to the extreme. For example, it is probably not useful to tag literally every single constant or template declaration. It is left to the discretion of the writer to find the right level of use. At least all major constructs such as test cases and functions should have a comprehensive description:
  - when a TTCN-3 definition uses module parameters, it is also recommended to mention this explicitly in the description;
  - descriptions for behavioural constructs should mention if they set the test component verdict and also all known limitations of the construct;
  - descriptions for type definitions, e.g. component types, should mention if the type has been designed to be type compatible to another type or vice versa to be used as a basis for other type definitions.

- the @see tag should be used to make dependencies between TTCN-3 definitions which are described by a @desc tag more explicit in the documentation, e.g. if some TTCN-3 definition uses a module parameter then its TTCN-3 definition should be referenced to using a @see tag;

- where applicable, parameterized constructions such as functions, altsteps and templates should use the *@param* and *@return* tags. The @param tags should first list the parameter name and then a brief description of how this parameter is used by the construct;

- the *@url* tag should be used to refer to the specification from which the TTCN-3 definition was derived from, e.g. a type definition could refer to a particular RFC IETF page. In some cases it may be necessary to use the *@desc* tag instead for this purpose as documents often are hard to access internally, i.e. it may only be possible to specify a reference to a complete document but impossible to point to a very specific clause in the present document;

- the *@url* and *@img* tag may be used to link to relevant documentation such as Test Purposes or original requirements or even drawings of test configurations. Generally, the corresponding Test Purpose (in the TSS&TP) and to the corresponding Requirement (in the Requirements Catalogue) should be linked from the relevant TTCN-3 test case definition;

- the *@remark* tag may be used with any TTCN-3 definition. It should be used sparingly, e.g. possibly to indicate how a TTCN-3 definition should not be used.

- The @purpose tag should be used with test case or module definitions depending on which definition level is more suitable to describe the corresponding conformance test purpose.

5.4 ATS archive

Annex D contains the ATS archive (ts_102995v010101p0.zip file expanding to text files with TTCN-3 code).

Annex A (normative): Proforma for the ICS proforma

A.1 Instructions for completing the ICS proforma

A.1.1 Other information

More detailed instructions are given at the beginning of the different clauses of the ICS proforma.

The supplier of the implementation shall complete the ICS proforma in each of the spaces provided. If necessary, the supplier may provide additional comments separately in Clause A.4.

A.1.2 Purposes and structure

The purpose of this ICS proforma is to provide a mechanism whereby a TTCN-3 tool vendor of the TTCN-3 core language [1] may provide information about the implementation in a standardized manner.

The ICS proforma is subdivided into clauses for the following categories of information:

- instructions for completing the ICS proforma;
- identification of the implementation;
- ICS proforma tables (containing the global statement of conformance).

A.1.3 Conventions

The ICS proforma is composed of information in tabular form in accordance with the guidelines presented in ISO/IEC 9646-7 [5].

Item column

It contains a number that identifies the item in the table.

Item description column

It describes each respective item (e.g. parameters, timers, etc.).

Reference column

It gives reference to the TTCN-3 core language [1], except where explicitly stated otherwise.

Status column

The following notations, defined in ISO/IEC 9646-7 [5], are used for the status column:

m mandatory - the capability is required to be supported.

n/a not applicable - in the given context, it is impossible to use the capability. No answer in the support column is required.

o optional - the capability may be supported or not.

o.i qualified optional - for mutually exclusive or selectable options from a set. "i" is an integer which identifies a unique group of related optional items and the logic of their selection which is defined immediately following the table.

ci conditional - the requirement on the capability ("m", "o" or "n/a") depends on the support of other optional or conditional items. "i" is an integer identifying a unique conditional status expression that is defined immediately following the table. For nested conditional expressions, the syntax "IF ... THEN (IF ... THEN ... ELSE...) ELSE ..." shall be used to avoid ambiguities. If an ELSE clause is omitted, "ELSE n/a" shall be implied.

NOTE: Support of a capability means that the capability is implemented in conformance to the TTCN-3 core language [1].

Support column

The support column shall be filled in by the supplier of the implementation. The following common notations, defined in ISO/IEC 9646-7 [5], are used for the support column:

- Y or y supported by the implementation.
- N or n not supported by the implementation.
- N/A or n/a or "no answer required" (allowed only if the status is N/A, directly or after evaluation of a conditional status).

Values allowed column

This column contains the values or the ranges of values allowed.

Values supported column

The support column shall be filled in by the supplier of the implementation. In this column the values or the ranges of values supported by the implementation shall be indicated.

References to items

For each possible item answer (answer in the support column) within the ICS proforma, a unique reference exists. It is defined as the table identifier, followed by a slash character "/", followed by the item number in the table. If there is more than one support column in a table, the columns shall be discriminated by letters (a, b, etc.) respectively.

EXAMPLE: 5/4 is the reference to the answer of item 4 in Table 5.

A.2 Identification of the implementation

Identification of the Implementation under Test (IUT) and the system in which it resides - the System Under Test (SUT) should be filled in so as to provide as much detail as possible regarding version numbers and configuration options.

The product supplier information and client information should both be filled in if they are different.

A person who can answer queries regarding information supplied in the ICS should be named as the contact person.

A.2.1 Date of the statement

Date of the statement:	

A.2.2 Implementation under Test (IUT) identification

IUT name:	
IUT version:	

A.2.3 System under Test (SUT) identification

SUT name:	
Hardware configuration:	
Operating system:	

A.2.4 Product supplier

Name:	
Address:	
Telephone number:	
Facsimile number:	
Additional information:	

A.2.5 Client

Name:	
Address:	
Telephone number:	
Facsimile number:	
Additional information:	

A.2.6 ICS contact person

Name:	
Telephone number:	
Facsimile number:	
Additional information:	

A.3 ICS proforma tables

A.3.1 Global statement of conformance

	(Yes/No)
Are all mandatory capabilities implemented?	

NOTE: Answering "No" to this question indicates non-conformance to the TTCN-3 core language. Non-supported mandatory capabilities are to be identified in the ICS, with an explanation of why the implementation is non-conforming.

A.3.2 Basic language elements

Table A.1: Basic language elements

Item	Is the implementation able to...	Reference in ES 201 873-1	Status	Support
1	support case sensitive identifiers?	Clause 5.1	m	
2	support the nine basic scope units?	Clause 5.2	m	
3	support the scope rules?	Clause 5.2	m	
4	support uniqueness of identifiers?	Clause 5.2.2	m	
5	support arbitrary order of language elements?	Clause 5.3	m	
6	support formal parameters of kind value?	Clause 5.4.1.1	m	
7	support formal parameters of kind template?	Clause 5.4.1.2	m	

A.3.3 Types and values

Table A.2: Types and values

Item	Is the implementation able to ...	Reference in ES 201 873-1	Status	Support
1	support basic type integer?	Clause 6.1.0	m	
2	support basic type float?	Clause 6.1.0	m	
3	support basic type boolean?	Clause 6.1.0	m	
4	support basic type verdicttype?	Clause 6.1.0	m	
5	support basic string type bitstring?	Clause 6.1.1	m	
6	support basic string type hexstring?	Clause 6.1.1	m	
7	support basic string type octetstring?	Clause 6.1.1	m	
8	support basic string type charstring?	Clause 6.1.1	m	
9	support basic string type universal charstring?	Clause 6.1.1	m	
10	support sub-typing of basic types: lists of values?	Clause 6.1.2.1	m	
11	support sub-typing of basic types: lists of types?	Clause 6.1.2.2	m	
12	support sub-typing of basic types: ranges?	Clause 6.1.2.3	m	
13	support sub-typing of basic types: infinite ranges?	Clause 6.1.2.3.1	m	
14	support sub-typing of basic types: string length restrictions?	Clause 6.1.2.4	m	
15	support sub-typing of basic types: pattern sub-typing of character string types?	Clause 6.1.2.5	m	
16	support sub-typing of basic types: mixing patterns, lists and ranges?	Clause 6.1.2.6.1	m	
17	support sub-typing of basic types: using length restrictions with other constraints?	Clause 6.1.2.6.2	m	
18	support structured type record?	Clause 6.2.1	m	
19	support structured type set?	Clause 6.2.2	m	
20	support records of single type?	Clause 6.2.3	m	
21	support sets of single type?	Clause 6.2.3	m	
22	support enumerated type?	Clause 6.2.4	m	
23	support union?	Clause 6.2.5	m	
24	support anytype?	Clause 6.2.6	m	
25	support array?	Clause 6.2.7	m	
26	support default type?	Clause 6.2.8	m	
27	support communication port type?	Clause 6.2.9	m	
28	support component type?	Clause 6.2.10.1	m	
29	support reuse of component types?	Clause 6.2.10.2	m	
30	support component references?	Clause 6.2.11	m	

Item	Is the implementation able to ...	Reference in ES 201 873-1	Status	Support
31	support addressing entities inside the SUT?	Clause 6.2.12	m	
32	support length sub-typing of record ofs?	Clause 6.2.13.1	m	
32	support length sub-typing of set ofs?	Clause 6.2.13.1	m	
33	support list sub-typing of structured types and anytype?	Clause 6.2.13.2	m	
34	support sub-typing of the iterated type of record ofs?	Clause 6.2.13.3	m	
35	support sub-typing of the iterated type of sets ofs?	Clause 6.2.13.3	m	
36	support component references?	Clause 6.2.11	m	
37	support component references?	Clause 6.2.11	m	
38	support component references?	Clause 6.2.11	m	
39	support component references?	Clause 6.2.11	m	
40	support type compatibility of non-structured types?	Clause 6.3.1	m	
41	support type compatibility of enumerated types?	Clause 6.3.2.1	m	
42	support type compatibility of record types?	Clause 6.3.2.2	m	
43	support type compatibility of record of types?	Clause 6.3.2.2	m	
44	support type compatibility of set types?	Clause 6.3.2.3	m	
45	support type compatibility of set of types?	Clause 6.3.2.3	m	
46	support type compatibility of union types?	Clause 6.3.2.4	m	
47	support type compatibility of anytype types?	Clause 6.3.2.5	m	
48	support compatibillty between sub-structures?	Clause 6.3.2.6	m	
49	support type compatibility of component types?	Clause 6.3.3	m	
50	support type synonym?	Clause 6.4	m	

A.3.4 Expressions

Table A.3: Expressions

Item	Is the implementation able to ...	Reference in ES 201 873-1	Status	Support
1	support precedence of operators?	Clause 7.1, Table 6	m	
2	support arithmetic operator addition?	Clauses 7.1 and 7.1.1, Table 5	m	
3	support arithmetic operator subtraction?	Clauses 7.1 and 7.1.1, Table 5	m	
4	support arithmetic operator multiplication?	Clauses 7.1 and 7.1.1, Table 5	m	
5	support arithmetic operator division?	Clauses 7.1 and 7.1.1, Table 5	m	
6	support arithmetic operator modulo?	Clauses 7.1 and 7.1.1, Table 5	m	
7	support arithmetic operator remainder?	Clauses 7.1 and 7.1.1, Table 5	m	
8	support string operator concatenation?	Clauses 7.1 and 7.1.2, Table 5	m	
9	support relational operator equal?	Clauses 7.1 and 7.1.3, Table 5	m	
10	support relational operator less than?	Clauses 7.1 and 7.1.3, Table 5	m	
11	support relational operator greater than?	Clauses 7.1 and 7.1.3, Table 5	m	
12	support relational operator not equal?	Clauses 7.1 and 7.1.3, Table 5	m	
13	support relational operator greater than or equal?	Clauses 7.1 and 7.1.3, Table 5	m	
14	support relational operator less than or equal?	Clauses 7.1 and 7.1.3, Table 5	m	
15	support logical operator logical not?	Clauses 7.1 and 7.1.4, Table 5	m	
16	support logical operator logical and?	Clauses 7.1 and 7.1.4, Table 5	m	
17	support logical operator logical or?	Clauses 7.1 and 7.1.4, Table 5	m	
18	support logical operator logical xor?	Clauses 7.1 and 7.1.4, Table 5	m	
15	support bitwise operator bitwise not?	Clauses 7.1 and 7.1.5, Table 5	m	
16	support bitwise operator bitwise and?	Clauses 7.1 and 7.1.5, Table 5	m	
17	support bitwise operator bitwise or?	Clauses 7.1 and 7.1.5, Table 5	m	
18	support bitwise operator bitwise xor?	Clauses 7.1 and 7.1.5, Table 5	m	
19	support shift operator shift left?	Clauses 7.1 and 7.1.6, Table 5	m	
20	support shift operator shift right?	Clauses 7.1 and 7.1.6, Table 5	m	
21	support rotate operator rotate left?	Clauses 7.1 and 7.1.7, Table 5	m	
22	support rotate operator rotate right?	Clauses 7.1 and 7.1.7, Table 5	m	

A.3.5 Modules

Table A.4: Modules

Item	Is the implementation able to ...	Reference in ES 201 873-1	Status	Support
1	support…?	Clause …	m	
…	…	…	m	

A.4 Additional information for ICS

This clause contains all additional comments provided by the supplier of the implementation.

Annex B (normative): Test Suite Structure and Test Purposes (TSS&TP)

B.1 Test Suite Structure (TSS)

The Test Suite Structure is in close alignment with ES 201 873-1 [1], containing:

a) positive syntactical tests (Table B.1);

b) positive semantical tests (Table B.1);

c) negative syntactical tests (Table B.2); and

d) negative semantical tests (Table B.2).

The execution order of the TTCN-3 tool conformance test cases is specified in the dependencies section of test purpose descriptions.

Table B.1: Test suite structure, positive tests

Basic language elements		
	Identifiers and keywords	TC_Syn_0501_Identifier_xxx
		TC_Sem_0501_Identifier_xxx
	Scope rules	TC_Syn_0502_Scopes_xxx
		TC_Sem_0502_Scopes_xxx
	Ordering of language elements	TC_Syn_0503_Ordering_xxx
		TC_Sem_0503_Ordering_xxx
	Parameterization	TC_Syn_0504_Parameterization_xxx
		TC_Sem_0504_Parameterization_xxx
	Cyclic Definitions	TC_Syn_0505_Cyclic_xxx
		TC_Sem_0505_Cyclic_xxx
Types and values		
	Basic types and values	TC_Syn _0601_BasicTypes_xxx
		TC_Sem _0601_BasicTypes_xxx
	Structured types and values	TC_Syn _0602_StructuredTypes_xxx
		TC_Sem _0602_StructuredTypes_xxx
	Type compatibility	TC_Syn _0603_TypeComp_xxx
		TC_Sem _0603_TypeComp_xxx
	Type synonym	TC_Syn _0604_TypeSynonym_xxx
		TC_Sem _0604_TypeSynonym_xxx
Expressions		TC_Syn _0700_Expressions_xxx
		TC_Sem _0700_Expressions_xxx
	Operators	TC_Syn _0701_Operators_xxx
		TC_Sem _0701_Operators_xxx
Modules		
	Definition of a module	TC_Syn _0801_DefModule_xxx
		TC_Sem _0801_DefModule_xxx
	Module definitions part	TC_Syn _0802_DefinitionsPart_xxx
		TC_Sem _0802_DefinitionsPart_xxx
	Module control part	TC_Syn _0803_ControlPart_xxx
		TC_Sem _0803_ControlPart_xxx
Port types, component types and test configurations		
	Communication ports	TC_Syn _0901_CommPorts_xxx
		TC_Sem _0901_CommPorts_xxx
	Test system interface	TC_Syn _0902_TestSystemInt_xxx
		TC_Sem _0902_TestSystemInt_xxx
Declaring constants		TC_Syn _1000_Constants_xxx
		TC_Sem _1000_Constants_xxx

Declaring variables		TC_Syn _1100_Variables_xxx
		TC_Sem _1100_Variables_xxx
	Value variables	TC_Syn _1101_ValueVariables_xxx
		TC_Sem _1101_ValueVariables_xxx
	Template variables	TC_Syn _1102_TemplVariables_xxx
		TC_Sem _1102_TemplVariables_xxx
Declaring timers		TC_Syn _1200_Timers_xxx
		TC_Sem _1200_Timers_xxx
Declaring messages		TC_Syn _1300_Messages_xxx
		TC_Sem _1300_Messages_xxx
Declaring procedure signatures		TC_Syn _1400_Signatures_xxx
		TC_Sem _1400_Signatures_xxx
Declaring templates		TC_Syn _1500_Templates_xxx
		TC_Sem _1500_Templates_xxx
	Declaring message templates	TC_Syn _1501_MessageTempl_xxx
		TC_Sem _1501_MessageTempl_xxx
	Declaring signature templates	TC_Syn _1502_SignatureTempl_xxx
		TC_Sem _1502_SignatureTempl_xxx
	Global and local templates	TC_Syn _1503_GlobLocalTempl_xxx
		TC_Sem _1503_GlobLocalTempl_xxx
	In-line Templates	TC_Syn _1504_InlineTempl_xxx
		TC_Sem _1504_InlineTempl_xxx
	Modified templates	TC_Syn _1505_ModifiedTempl_xxx
		TC_Sem _1505_ModifiedTempl_xxx
	Referencing elements of templates or template fields	TC_Syn _1506_RefTempl_xxx
		TC_Sem _1506_RefTempl_xxx
	Template matching mechanisms	TC_Syn _1507_TemplMatching_xxx
		TC_Sem _1507_TemplMatching_xxx
	Template Restrictions	TC_Syn _1508_TemplRestr_xxx
		TC_Sem _1508_TemplRestr_xxx
	Match Operation	TC_Syn _1509_Match_xxx
		TC_Sem _1509_Match_xxx
	Valueof Operation	TC_Syn _1510_ValueOf_xxx
		TC_Sem _1510_ValueOf_xxx
Functions, altsteps and testcases		
	Functions	TC_Syn _1601_Functions_xxx
		TC_Sem _1601_Functions_xxx
	Altsteps	TC_Syn _1602_Altsteps_xxx
		TC_Sem _1602_Altsteps_xxx
	Test cases	TC_Syn _1702_TestCases_xxx
		TC_Sem _1702_TestCases_xxx
Basic program statements		
	Assignments	TC_Syn _1901_Assignments_xxx
		TC_Sem _1901_Assignments_xxx
	The If-else statement	TC_Syn _1902_IfElse_xxx
		TC_Sem _1902_IfElse_xxx
	The Select case statement	TC_Syn _1903_SelectCase_xxx
		TC_Sem _1903_SelectCase_xxx
	The For statement	TC_Syn _1904_For_xxx
		TC_Sem _1904_For_xxx
	The While statement	TC_Syn _1905_While_xxx
		TC_Sem _1905_While_xxx
	The Do-while statement	TC_Syn _1906_DoWhile_xxx
		TC_Sem _1906_DoWhile_xxx
	The Label statement	TC_Syn _1907_Label_xxx
		TC_Sem _1907_Label_xxx
	The Goto statement	TC_Syn _1908_Goto_xxx
		TC_Sem _1908_Goto_xxx
	The Stop execution statement	TC_Syn _1909_Stop_xxx
		TC_Sem _1909_Stop_xxx
	The Return statement	TC_Syn _1910_Return_xxx
		TC_Sem _1910_Return_xxx
	The Log statement	TC_Syn _1911_Log_xxx
		TC_Sem _1911_Log_xxx

	The Break statement	TC_Syn _1912_Break_xxx
		TC_Sem _1912_Break_xxx
	The Continue statement	TC_Syn _1913_Continue_xxx
		TC_Sem _1913_Continue_xxx
	Statement block	TC_Syn _1914_StatementBlock_xxx
		TC_Sem _1914_StatementBlock_xxx
Statement and operations for alternative behaviours		
	The Alt statement	TC_Syn _2002_Alt_xxx
		TC_Sem _2002_Alt_xxx
	The Repeat statement	TC_Syn _2003_Repeat_xxx
		TC_Sem _2003_Repeat_xxx
	The Interleave statement	TC_Syn _2004_Interleave_xxx
		TC_Sem _2004_Interleave_xxx
	Default Handling	TC_Syn _2005_Default_xxx
		TC_Sem _2005_Default_xxx
Configuration Operations		
	Connection Operations	TC_Syn _2101_Connections_xxx
		TC_Sem _2101_Connections_xxx
	Test Component Operations	TC_Syn _2102_CompOperations_xxx
		TC_Sem _2102_CompOperations_xxx
Communication operations		
	Message-based communication	TC_Syn _2202_MessageBased_xxx
		TC_Sem _2202_MessageBased_xxx
	Procedure-based communication	TC_Syn _2203_ProcedureBased_xxx
		TC_Sem _2203_ProcedureBased_xxx
	The Check operation	TC_Syn _2204_Check_xxx
		TC_Sem _2204_Check_xxx
	Controlling communication ports	TC_Syn _2205_ControllingPorts_xxx
		TC_Sem _2205_ControllingPorts_xxx
	Use of any and all with ports	TC_Syn _2206_AnyAll_xxx
		TC_Sem _2206_AnyAll_xxx
Timer operations		
	The timer mechanism	TC_Syn _2301_TimerMechanism_xxx
		TC_Sem _2301_TimerMechanism_xxx
	The Start timer operation	TC_Syn _2302_StartTimer_xxx
		TC_Sem _2302_StartTimer_xxx
	The Stop timer operation	TC_Syn _2303_StopTimer_xxx
		TC_Sem _2303_StopTimer_xxx
	The Read timer operation	TC_Syn _2304_ReadTimer_xxx
		TC_Sem _2304_ReadTimer_xxx
	The Running timer operation	TC_Syn _2305_RunningTimer_xxx
		TC_Sem _2305_RunningTimer_xxx
	The Timeout operation	TC_Syn _2306_Timeout_xxx
		TC_Sem _2306_Timeout_xxx
Test verdict operations		
	The Verdict mechanism	TC_Syn _2401_VerdictMechanism_xxx
		TC_Sem _2401_VerdictMechanism_xxx
	The Setverdict operation	TC_Syn _2402_SetVerdict_xxx
		TC_Sem _2402_SetVerdict_xxx
	The Getverdict operation	TC_Syn _2403_GetVerdict_xxx
		TC_Sem _2403_GetVerdict_xxx
External actions		TC_Syn _2500_Action_xxx
		TC_Sem _2500_Action_xxx
Module control		
	The Execute statement	TC_Syn _2601_Execute_xxx
		TC_Sem _2601_Execute_xxx
	The Control part	TC_Syn _2602_Control_xxx
		TC_Sem _2602_Control_xxx

Specifying attributes		
	The Attribute mechanism	TC_Syn _2701_AttribMechanism_xxx
		TC_Sem _2701_AttribMechanism_xxx
	The With statement	TC_Syn _2702_WithAttrib_xxx
		TC_Sem _2702_WithAttrib_xxx
	Display attributes	TC_Syn _2703_DisplayAttrib_xxx
		TC_Sem _2703_DisplayAttrib_xxx
	Encoding attributes	TC_Syn _2704_EncodeAttrib_xxx
		TC_Sem _2704_EncodeAttrib_xxx
	Variant attributes	TC_Syn _2705_VariantAttrib_xxx
		TC_Sem _2705_VariantAttrib_xxx
	Extension attributes	TC_Syn _2706_ExtensionAttrib_xxx
		TC_Sem _2706_ExtensionAttrib_xxx
	Optional attributes	TC_Syn _2707_OptionalAttrib_xxx
		TC_Sem _2707_OptionalAttrib_xxx

Table B.2: Test suite structure, negative tests

Basic language elements		
	Identifiers and keywords	TC_NegSyn_0501_Identifier_xxx
		TC_NegSem_0501_Identifier_xxx
	Scope rules	TC_NegSyn_0502_Scopes_xxx
		TC_NegSem_0502_Scopes_xxx
	Ordering of language elements	TC_NegSyn_0503_Ordering_xxx
		TC_NegSem_0503_Ordering_xxx
	Parameterization	TC_NegSyn_0504_Parameterization_xxx
		TC_NegSem_0504_Parameterization_xxx
	Cyclic Definitions	TC_NegSyn_0505_Cyclic_xxx
		TC_NegSem_0505_Cyclic_xxx
Types and values		
	Basic types and values	TC_NegSyn _0601_BasicTypes_xxx
		TC_NegSem _0601_BasicTypes_xxx
	Structured types and values	TC_NegSyn _0602_StructuredTypes_xxx
		TC_NegSem _0602_StructuredTypes_xxx
	Type compatibility	TC_NegSyn _0603_TypeComp_xxx
		TC_NegSem _0603_TypeComp_xxx
	Type synonym	TC_NegSyn _0604_TypeSynonym_xxx
		TC_NegSem _0604_TypeSynonym_xxx
Expressions		TC_NegSyn _0700_Expressions_xxx
		TC_NegSem _0700_Expressions_xxx
	Operators	TC_NegSyn _0701_Operators_xxx
		TC_NegSem _0701_Operators_xxx
Modules		
	Definition of a module	TC_NegSyn _0801_DefModule_xxx
		TC_NegSem _0801_DefModule_xxx
	Module definitions part	TC_NegSyn _0802_DefinitionsPart_xxx
		TC_NegSem _0802_DefinitionsPart_xxx
	Module control part	TC_NegSyn _0803_ControlPart_xxx
		TC_NegSem _0803_ControlPart_xxx
Port types, component types and test configurations		
	Communication ports	TC_NegSyn _0901_CommPorts_xxx
		TC_NegSem _0901_CommPorts_xxx
	Test system interface	TC_NegSyn _0902_TestSystemInt_xxx
		TC_NegSem _0902_TestSystemInt_xxx
Declaring constants		TC_NegSyn _1000_Constants_xxx
		TC_NegSem _1000_Constants_xxx
Declaring variables		TC_NegSyn _1100_Variables_xxx
		TC_NegSem _1100_Variables_xxx
	Value variables	TC_NegSyn _1101_ValueVariables_xxx
		TC_NegSem _1101_ValueVariables_xxx
	Template variables	TC_NegSyn _1102_TemplVariables_xxx
		TC_NegSem _1102_TemplVariables_xxx

Declaring timers		TC_NegSyn _1200_Timers_xxx
		TC_NegSem _1200_Timers_xxx
Declaring messages		TC_NegSyn _1300_Messages_xxx
		TC_NegSem _1300_Messages_xxx
Declaring procedure signatures		TC_NegSyn _1400_Signatures_xxx
		TC_NegSem _1400_Signatures_xxx
Declaring templates		TC_NegSyn _1500_Templates_xxx
		TC_NegSem _1500_Templates_xxx
	Declaring message templates	TC_NegSyn _1501_MessageTempl_xxx
		TC_NegSem _1501_MessageTempl_xxx
	Declaring signature templates	TC_NegSyn _1502_SignatureTempl_xxx
		TC_NegSem _1502_SignatureTempl_xxx
	Global and local templates	TC_NegSyn _1503_GlobLocalTempl_xxx
		TC_NegSem _1503_GlobLocalTempl_xxx
	In-line Templates	TC_NegSyn _1504_InlineTempl_xxx
		TC_NegSem _1504_InlineTempl_xxx
	Modified templates	TC_NegSyn _1505_ModifiedTempl_xxx
		TC_NegSem _1505_ModifiedTempl_xxx
	Referencing elements of templates or template fields	TC_NegSyn _1506_RefTempl_xxx
		TC_NegSem _1506_RefTempl_xxx
	Template matching mechanisms	TC_NegSyn _1507_TemplMatching_xxx
		TC_NegSem _1507_TemplMatching_xxx
	Template Restrictions	TC_NegSyn _1508_TemplRestr_xxx
		TC_NegSem _1508_TemplRestr_xxx
	Match Operation	TC_NegSyn _1509_Match_xxx
		TC_NegSem _1509_Match_xxx
	Valueof Operation	TC_NegSyn _1510_ValueOf_xxx
		TC_NegSem _1510_ValueOf_xxx
Functions, altsteps and testcases		
	Functions	TC_NegSyn _1601_Functions_xxx
		TC_NegSem _1601_Functions_xxx
	Altsteps	TC_NegSyn _1602_Altsteps_xxx
		TC_NegSem _1602_Altsteps_xxx
	Test cases	TC_NegSyn _1702_TestCases_xxx
		TC_NegSem _1702_TestCases_xxx
Basic program statements		
	Assignments	TC_NegSyn _1901_Assignments_xxx
		TC_NegSem _1901_Assignments_xxx
	The If-else statement	TC_NegSyn _1902_IfElse_xxx
		TC_NegSem _1902_IfElse_xxx
	The Select case statement	TC_NegSyn _1903_SelectCase_xxx
		TC_NegSem _1903_SelectCase_xxx
	The For statement	TC_NegSyn _1904_For_xxx
		TC_NegSem _1904_For_xxx
	The While statement	TC_NegSyn _1905_While_xxx
		TC_NegSem _1905_While_xxx
	The Do-while statement	TC_NegSyn _1906_DoWhile_xxx
		TC_NegSem _1906_DoWhile_xxx
	The Label statement	TC_NegSyn _1907_Label_xxx
		TC_NegSem _1907_Label_xxx
	The Goto statement	TC_NegSyn _1908_Goto_xxx
		TC_NegSem _1908_Goto_xxx
	The Stop execution statement	TC_NegSyn _1909_Stop_xxx
		TC_NegSem _1909_Stop_xxx
	The Return statement	TC_NegSyn _1910_Return_xxx
		TC_NegSem _1910_Return_xxx
	The Log statement	TC_NegSyn _1911_Log_xxx
		TC_NegSem _1911_Log_xxx
	The Break statement	TC_NegSyn _1912_Break_xxx
		TC_NegSem _1912_Break_xxx
	The Continue statement	TC_NegSyn _1913_Continue_xxx
		TC_NegSem _1913_Continue_xxx
	Statement block	TC_NegSyn _1914_StatementBlock_xxx
		TC_NegSem _1914_StatementBlock_xxx

Statement and operations for alternative behaviours		
	The Alt statement	TC_NegSyn _2002_Alt_xxx
		TC_NegSem _2002_Alt_xxx
	The Repeat statement	TC_NegSyn _2003_Repeat_xxx
		TC_NegSem _2003_Repeat_xxx
	The Interleave statement	TC_NegSyn _2004_Interleave_xxx
		TC_NegSem _2004_Interleave_xxx
	Default Handling	TC_NegSyn _2005_Default_xxx
		TC_NegSem _2005_Default_xxx
Configuration Operations		
	Connection Operations	TC_NegSyn _2101_Connections_xxx
		TC_NegSem _2101_Connections_xxx
	Test Component Operations	TC_NegSyn _2102_CompOperations_xxx
		TC_NegSem _2102_CompOperations_xxx
Communication operations		
	Message-based communication	TC_NegSyn _2202_MessageBased_xxx
		TC_NegSem _2202_MessageBased_xxx
	Procedure-based communication	TC_NegSyn _2203_ProcedureBased_xxx
		TC_NegSem _2203_ProcedureBased_xxx
	The Check operation	TC_NegSyn _2204_Check_xxx
		TC_NegSem _2204_Check_xxx
	Controlling communication ports	TC_NegSyn _2205_ControllingPorts_xxx
		TC_NegSem _2205_ControllingPorts_xxx
	Use of any and all with ports	TC_NegSyn _2206_AnyAll_xxx
		TC_NegSem _2206_AnyAll_xxx
Timer operations		
	The timer mechanism	TC_NegSyn _2301_TimerMechanism_xxx
		TC_NegSem _2301_TimerMechanism_xxx
	The Start timer operation	TC_NegSyn _2302_StartTimer_xxx
		TC_NegSem _2302_StartTimer_xxx
	The Stop timer operation	TC_NegSyn _2303_StopTimer_xxx
		TC_NegSem _2303_StopTimer_xxx
	The Read timer operation	TC_NegSyn _2304_ReadTimer_xxx
		TC_NegSem _2304_ReadTimer_xxx
	The Running timer operation	TC_NegSyn _2305_RunningTimer_xxx
		TC_NegSem _2305_RunningTimer_xxx
	The Timeout operation	TC_NegSyn _2306_Timeout_xxx
		TC_NegSem _2306_Timeout_xxx
Test verdict operations		
	The Verdict mechanism	TC_NegSyn _2401_VerdictMechanism_xxx
		TC_NegSem _2401_VerdictMechanism_xxx
	The Setverdict operation	TC_NegSyn _2402_SetVerdict_xxx
		TC_NegSem _2402_SetVerdict_xxx
	The Getverdict operation	TC_NegSyn _2403_GetVerdict_xxx
		TC_NegSem _2403_GetVerdict_xxx
External actions		TC_NegSyn _2500_Action_xxx
		TC_NegSem _2500_Action_xxx
Module control		
	The Execute statement	TC_NegSyn _2601_Execute_xxx
		TC_NegSem _2601_Execute_xxx
	The Control part	TC_NegSyn _2602_Control_xxx
		TC_NegSem _2602_Control_xxx

Specifying attributes		
	The Attribute mechanism	TC_NegSyn _2701_AttribMechanism_xxx
		TC_NegSem _2701_AttribMechanism_xxx
	The With statement	TC_NegSyn _2702_WithAttrib_xxx
		TC_NegSem _2702_WithAttrib_xxx
	Display attributes	TC_NegSyn _2703_DisplayAttrib_xxx
		TC_NegSem _2703_DisplayAttrib_xxx
	Encoding attributes	TC_NegSyn _2704_EncodeAttrib_xxx
		TC_NegSem _2704_EncodeAttrib_xxx
	Variant attributes	TC_NegSyn _2705_VariantAttrib_xxx
		TC_NegSem _2705_VariantAttrib_xxx
	Extension attributes	TC_NegSyn _2706_ExtensionAttrib_xxx
		TC_NegSem _2706_ExtensionAttrib_xxx
	Optional attributes	TC_NegSyn _2707_OptionalAttrib_xxx
		TC_NegSem _2707_OptionalAttrib_xxx

B.2 Test Purposes (TP)

B.2.1 Introduction

For each test requirement a Test Purpose (TP) is defined. Test purposes shall be defined in a dedicated test purpose document as well as with TTCN-3 documentation tags in each test case of the ATS. Both documentations shall convey the same information for each test purpose.

B.2.1.1 Test purpose naming convention

The test purpose naming scheme corresponds to the test case identifier naming scheme (see Clause 5.2.2) and vice-versa.

B.2.1.2 Source of test purpose definition

B.2.1.3 Test purpose structure

The test purpose structure is according to the test suite structure (TSS).

B.2.2 Test purpose format

In the following, examples for tabular test purpose descriptions are shown that shall be defined in the test purpose document. This representation is a direct mapping of the contents of the document tags in the ATS (such as @purpose, @remark, or @verdict). The tabular descriptions are presented along with their corresponding TTCN-3 documentation tag equivalent. The test purpose reference shall be provided in a machine-readable format.

Test Purpose Id	TP_NegSyn_0501_Identifier
Reference	ES 201 873-1 [1], Clause 5.1
ICS	None
Dependencies	None
Summary	Ensure that when the IUT loads a module containing an identifier named with a keyword then the module is rejected
Expected Output	Rejection as invalid
Notes	

A corresponding TTCN-3 module addressing TP_NegSyn_0501_Identifier is the following:

```
/*************************************************************
 ** @author   STF 380
 ** @version  0.0.1
 ** @purpose  1:5.1, Ensure that when the IUT loads a module containing an
              identifier named with a keyword then the module is rejected.
 ** @verdict  pass Rejection as invalid
 *************************************************************/

module NegSyn_0501_Identifier_001 {

type component GeneralComp {
}

testcase TC_NegSyn_0501_Identifier_001() runs on GeneralComp {
    var integer component := 1;
}

control{
    execute(TC_NegSyn_0501_Identifier_001());
}

}
```

Test Purpose Id	TP_Syn_0501_Identifier
Reference	ES 201 873-1 [1], Clause 5.1
ICS	None
Dependencies	None
Summary	Ensure that the IUT handles the identifiers case sensitively.
Expected Output	TTCN-3 verdict “pass”
Notes	

A corresponding TTCN-3 module for TP_Syn_0501_Identifier is the following:

```
/*************************************************************
 ** @author   STF 380
 ** @version  0.0.1
 ** @purpose  1:5.1, Ensure that the IUT handle the identifiers case sensitively.
 ** @verdict  pass TTCN-3 verdict
 *************************************************************/
module Syn_0501_Identifier_001 {

type component IdComp {
    const integer c_int := 0;
}

testcase TC_Syn_0501_Identifier_001() runs on IdComp {
    const integer C_INT := 1;
    if (c_int == 0){
        setverdict(pass);
    }
    else {
        setverdict(fail);
    }
}

control{
    execute(TC_Syn_0501_Identifier_001());
}

}
```

Annex C (normative): Partial IXIT proforma

C.1 Introduction

This partial IXIT proforma contained in the present document is provided for completion, when the related Abstract Test Suite is to be used against the Implementation under Test (IUT).

The completed partial IXIT will normally be used in conjunction with the completed ICS, as it adds precision to the information provided by the ICS.

C.2 IXIT items

Table C.1: Example: IXIT elements

Item	Module Parameter	Description	Type	Value

Annex D (informative): TTCN-3 library modules

D.1 The ATS in TTCN-3 core (text) format

The TTCN-3 library modules have been produced using the Testing and Test Control Notation (TTCN-3) according to ES 201 873-1 [1].

The TTCN-3 core (text) representation corresponding to this ATS is contained in several ASCII files contained in archive ts_102995v010101p0.zip which accompanies the present document.

Annex E (informative): Bibliography

- ETSI EG 202 106 (V2.1.1): "Methods for Testing and Specification (MTS); Guidelines for the use of formal SDL as a descriptive tool".
- ISO/IEC 9646-2 (1994): "Conformance testing methodology and framework - Part 2: Abstract Test Suite Specification".
- ISO/IEC 9646-3 (1992): "Conformance testing methodology and framework - Part 3: The Tree and Tabular Combined Notation".
- ISO/IEC 9646-3/DAM 1 (1992): "Conformance testing methodology and framework - Part 3: The Tree and Tabular Combined Notation; Amendment 1: TTCN extensions".
- ISO/IEC 9646-5 (1994): "Conformance testing methodology and framework - Part 5: Requirements on test laboratories and clients for the conformance assessment process".
- ETSI ES 201 873-4: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 4: TTCN-3 Operational Semantics".

Annex F (informative): Change history

Date	WG Doc.	CR	Rev	CAT	Title / Comment	Current Version	New Version
10/2010					Ready for approval	0.0.4	

History

Document history		
V1.1.1	November 2010	Publication