SLOVENSKI STANDARD
SIST ES 201 873-4 V4.1.1:2009

01-september-2009

Metode za preskuġanje in specificiranje (MTS) - 3. razliį ica zapisa preskuġanja in krmilnih preskusov - 4. del: Operativna semantika TTCN-3

Methods for Testing and Specification (MTS) - The Testing and Test Control Notation version 3 - Part 4: TTCN-3 Operational Semantics

iTeh STANDARD PREVIEW
(standards.iteh.ai)

Ta slovenski standard je istoveten z: ES 201 873-4 Version 4.1.1

SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be01d994745/sist-es-201-873-4-v4-1-1-2009

ICS:

33.040.01	Telekomunikacijski sistemi na splošno	Telecommunication systems in general

SIST ES 201 873-4 V4.1.1:2009 **en**

2003-01.Slovenski inštitut za standardizacijo. Razmnoževanje celote ali delov tega standarda ni dovoljeno.

SIST ES 201 873-4 V4.1.1:2009

iTeh STANDARD PREVIEW
(standards.iteh.ai)

SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be01d994745/sist-es-201-873-4-v4-1-1-2009

SIST ES 201 873-4 V4.1.1:2009

ETSI ES 201 873-4 V4.1.1 (2009-06)

ETSI Standard

Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 4: TTCN-3 Operational Semantics

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be01d994745/sist-es-201-873-4-v4-1-1-2009

SIST ES 201 873-4 V4.1.1:2009

Reference
RES/MTS-00107-4 T3 ed411 OS

Keywords
interoperability, methodology, MTS, testing, TTCN

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

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be01d994744/sist-es-201-873-4-v4-1-1-2009

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

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

© European Telecommunications Standards Institute 2009.
All rights reserved.

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

SIST ES 201 873-4 V4.1.1:2009

Contents

Intellectual Property Rights 7
Foreword 7
1 Scope 8
2 References 8
2.1 Normative references 8
2.2 Informative references 8
3 Definitions and abbreviations 8
3.1 Definitions 8
3.2 Abbreviations 9
4 Introduction 9
5 Structure of the present document 9
6 Restrictions 10
7 Replacement of short forms 10
7.1 Order of replacement steps 11
7.2 Replacement of global constants and module parameters 11
7.3 Embedding single receiving operations into alt statements 11
7.4 Embedding stand-alone altstep calls into alt statements 12
7.5 Replacement of interleave statements 12
7.6 Replacement of trigger operations 25
7.7 Replacement of select-case statements 25
7.8 Replacement of simple break statements 27
7.9 Replacement of continue statements 27
7.10 Adding default parameters to disconnect and unmap operations without parameters 28
7.11 Adding default values of parameters 28
8 Flow graph semantics of TTCN-3 28
8.1 Flow graphs 29
8.1.1 Flow graph frame 29
8.1.2 Flow graph nodes 29
8.1.2.1 Start nodes 29
8.1.2.2 End nodes 29
8.1.2.3 Basic nodes 29
8.1.2.4 Reference nodes 30
8.1.2.4.1 OR combination of reference nodes 30
8.1.2.4.2 Multiple occurrences of reference nodes 30
8.1.3 Flow lines 31
8.1.4 Flow graph segments 31
8.1.5 Comments 32
8.1.6 Handling of flow graph descriptions 33
8.2 Flow graph representation of TTCN-3 behaviour 33
8.2.1 Flow graph construction procedure 33
8.2.2 Flow graph representation of module control 34
8.2.3 Flow graph representation of test cases 35
8.2.4 Flow graph representation of functions 35
8.2.5 Flow graph representation of altsteps 36
8.2.6 Flow graph representation of component type definitions 37
8.2.7 Retrieval of start nodes of flow graphs 38
8.3 State definitions for TTCN-3 modules 38
8.3.1 Module state 38
8.3.1.1 Accessing the module state 39
8.3.2 Entity states 40
8.3.2.1 Accessing entity states 41

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/40e5a84e-3d50-4645-8ca7-08e01d994745/sist-es-201-873-4-v4-1-1-2009

SIST ES 201 873-4 V4.1.1:2009

8.3.2.2 Data state and variable binding 43
8.3.2.3 Accessing data states 43
8.3.2.4 Timer state and timer binding 44
8.3.2.5 Accessing timer states 45
8.3.2.6 Port references and port binding 46
8.3.2.7 Accessing port references 47
8.3.3 Port states 47
8.3.3.1 Handling of connections among ports 48
8.3.3.2 Handling of port states 48
8.3.4 General functions for the handling of module states 49
8.4 Messages, procedure calls, replies and exceptions 50
8.4.1 Messages 50
8.4.2 Procedure calls and replies 50
8.4.3 Exceptions 51
8.4.4 Construction of messages, procedure calls, replies and exceptions 51
8.4.5 Matching of messages, procedure calls, replies and exceptions 51
8.4.6 Retrieval of information from received items 52
8.5 Call records for functions, altsteps and test cases 52
8.5.1 Handling of call records 52
8.6 The evaluation procedure for a TTCN-3 module 53
8.6.1 Evaluation phases 53
8.6.1.1 Phase I: Initialization 53
8.6.1.2 Phase II: Update 54
8.6.1.3 Phase III: Selection 54
8.6.1.4 Phase IV: Execution 54
8.6.2 Global functions 54

9 Flow graph segments for TTCN-3 constructs 55
9.1 Action statement 55
9.2 Activate statement 56
9.2a Alive component operation 56
9.2a.1 Flow graph segment <alive-comp-act> 58
9.2a.2 Flow graph segment <alive-comp-snap> 59
9.3 Alt statement 59
9.3.1 Flow graph segment <take-snapshot> 61
9.3.2 Flow graph segment <receiving-branch> 62
9.3.3 Flow graph segment <altstep-call-branch> 63
9.3.4 Flow graph segment <else-branch> 64
9.3.5 Flow graph segment <default-evocation> 65
9.4 Altstep call 66
9.5 Assignment statement 66
9.5a Break statements in altsteps 66
9.6 Call operation 67
9.6.1 Flow graph segment <nb-call-with-one-receiver> 69
9.6.1a Flow graph segment <nb-call-with-multiple-receivers> 69
9.6.2 Flow graph segment <nb-call-without-receiver> 71
9.6.3 Flow graph segment <b-call-without-duration> 71
9.6.4 Flow graph segment <b-call-with-duration> 72
9.6.5 Flow graph segment <call-reception-part> 73
9.6.6 Flow graph segment <catch-timeout-exception> 74
9.7 Catch operation 74
9.8 Check operation 75
9.8.1 Flow graph segment <check-with-sender> 76
9.8.2 Flow graph segment <check-without-sender> 77
9.9 Clear port operation 78
9.10 Connect operation 78
9.11 Constant definition 79
9.12 Create operation 80
9.13 Deactivate statement 81
9.13.1 Flow graph segment <deactivate-one-default> 82
9.13.2 Flow graph segment <deactivate-all-defaults> 82
9.14 Disconnect operation 83

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be01d994745/sist-es-201-873-4-v4-1-1-2009

SIST ES 201 873-4 V4.1.1:2009

9.14.1 Flow graph segment <disconnect-one-par-pair> 83
9.14.2 Flow graph segment <disconnect-all> 85
9.14.3 Flow graph segment <disconnect-comp> 86
9.14.4 Flow graph segment <disconnect-port> 87
9.14.5 Flow graph segment <disconnect-two-par-pairs> 87
9.15 Do-while statement 88
9.16 Done component operation 89
9.17 Execute statement 91
9.17.1 Flow graph segment <execute-without-timeout> 92
9.17.2 Flow graph segment <execute-timeout> 93
9.17.3 Flow graph segment <dynamic-error> 94
9.18 Expression 94
9.18.1 Flow graph segment <lit-value> 95
9.18.2 Flow graph segment <var-value> 95
9.18.3 Flow graph segment <func-op-call> 96
9.18.4 Flow graph segment <operator-appl> 96
9.19 Flow graph segment <finalize-component-init> 97
9.20 Flow graph segment <init-component-scope> 97
9.20a Flow graph segment <init-scope-with-runs-on> 98
9.20b Flow graph segment <init-scope-without-runs-on> 98
9.21 Flow graph segment <parameter-handling> 99
9.22 Flow graph segment <statement-block> 99
9.23 For statement 100
9.24 Function call 101
9.24.1 Flow graph segment <value-par-calculation> 103
9.24.2 Flow graph segment <ref-par-var-calc> 103
9.24.3 Flow graph segment <ref-par-timer-calc> 104
9.24.3a Flow graph segment <ref-par-port-calc> 104
9.24.4 Flow graph segment <user-def-func-call> 105
9.24.5 Flow graph segment <predef-ext-func-call> 106
9.25 Getcall operation 106
9.26 Getreply operation 106
9.27 Getverdict operation 107
9.28 Goto statement 107
9.28a Halt port operation 108
9.29 If-else statement 108
9.29a Kill component operation 109
9.29a.1 Flow graph segment <kill-mtc> 111
9.29a.2 Flow graph segment <kill-component> 112
9.29a.3 Flow graph segment <kill-all-comp> 113
9.29b Kill execution statement 113
9.29b.1 Flow graph segment <kill-control> 114
9.29c Killed component operation 115
9.30 Label statement 116
9.31 Log statement 116
9.32 Map operation 117
9.33 Mtc operation 117
9.34 Port declaration 118
9.35 Raise operation 118
9.35.1 Flow graph segment <raise-with-one-receiver-op> 119
9.35.1a Flow graph segment <raise-with-multiple-receivers-op> 119
9.35.2 Flow graph segment <raise-without-receiver-op> 121
9.36 Read timer operation 122
9.37 Receive operation 123
9.37.1 Flow graph segment <receive-with-sender> 124
9.37.2 Flow graph segment <receive-without-sender> 125
9.37.3 Flow graph segment <receive-assignment> 126
9.38 Repeat statement 126
9.39 Reply operation 127
9.39.1 Flow graph segment <reply-with-one-receiver-op> 128
9.39.1a Flow graph segment <reply-with-multiple-receivers-op> 128
9.39.2 Flow graph segment <reply-without-receiver-op> 130

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be01d994745/sist-es-201-873-4-v4-1-1-2009

SIST ES 201 873-4 V4.1.1:2009

9.40 Return statement 130
9.40.1 Flow graph segment <return-with-value> 132
9.40.2 Flow graph segment <return-without-value> 133
9.41 Running component operation 134
9.41.1 Flow graph segment <running-comp-act> 135
9.41.2 Flow graph segment <running-comp-snap> 136
9.42 Running timer operation 137
9.43 Self operation 138
9.44 Send operation 138
9.44.1 Flow graph segment <send-with-one-receiver-op> 139
9.44.1a Flow graph segment <send-with-multiple-receivers-op> 139
9.44.2 Flow graph segment <send-without-receiver-op> 141
9.45 Setverdict operation 141
9.46 Start component operation 142
9.47 Start port operation 144
9.48 Start timer operation 144
9.48.1 Flow graph segment <start-timer-op-default> 145
9.48.2 Flow graph segment <start-timer-op-duration> 146
9.49 Stop component operation 146
9.49.1 Void 148
9.49.2 Flow graph segment <stop-alive-component> 148
9.49.3 Flow graph segment <stop-all-comp> 149
9.50 Stop execution statement 150
9.50.1 Void 151
9.51 Stop port operation 151
9.52 Stop timer operation 152
9.53 System operation 152
9.54 Timer declaration 153
9.54.1 Flow graph segment <timer-decl-default> 153
9.54.2 Flow graph segment <timer-decl-no-def> 154
9.55 Timeout timer operation 155
9.56 Unmap operation 156
9.56.1 Flow graph segment <unmap-all> 158
9.56.2 Flow graph segment <unmap-comp> 159
9.56.3 Flow graph segment <unmap-port> 160
9.57 Variable declaration 160
9.57.1 Flow graph segment <var-declaration-init> 161
9.57.2 Flow graph segment <var-declaration-undef> 162
9.58 While statement 162

10 Lists of operational semantic components 163
10.1 Functions and states 163
10.2 Special keywords 164
10.3 Flow graphs of TTCN-3 behaviour descriptions 165
10.4 Flow graph segments 165

History 168

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6ba01d994745/sist-es-201-873-4-v4-1-1-2009

SIST ES 201 873-4 V4.1.1:2009

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

The present document is part 4 of a multi-part deliverable. Full details of the entire series can be found in part 1 [1].

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be01d994745/sist-es-201-873-4-v4-1-1-2009

SIST ES 201 873-4 V4.1.1:2009

1 Scope

The present document defines the operational semantics of TTCN-3. The present document is based on the TTCN-3 core language defined in ES 201 873-1 [1].

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific.

- For a specific reference, subsequent revisions do not apply.
- Non-specific reference may be made only to a complete document or a part thereof and only in the following cases:
  - if it is accepted that it will be possible to use all future changes of the referenced document for the purposes of the referring document;
  - for informative references.

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.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

2.1 Normative references

SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e5a84e-5d56-4645-8ca7-6be01d994743/sist-es-201-873-4-v4-1-1-2009

The following referenced documents are indispensable for the application of the present document. For dated references, only the edition cited applies. For non-specific references, the latest edition of the referenced document (including any amendments) applies.

[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".

2.2 Informative references

The following referenced documents are not essential to the use of the present document but they assist the user with regard to a particular subject area. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Not applicable.

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 201 873-1 [1] apply.

SIST ES 201 873-4 V4.1.1:2009

3.2 Abbreviations

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

BNF	Backus-Nauer Form
MTC	Master Test Component
SUT	System Under Test
TTCN	Testing and Test Control Notation

4 Introduction

This clause defines the meaning of TTCN-3 behaviour in an intuitive and unambiguous manner. The operational semantics is not meant to be formal and therefore the ability to perform mathematical proofs based on this semantics is very limited.

This operational semantics provides a state oriented view on the execution of a TTCN module. Different kinds of states are introduced and the meaning of the different TTCN-3 constructs is described by:

1) using state information to define the preconditions for the execution of a construct; and

2) defining how the execution of a construct will change a state.

The operational semantics is restricted to the meaning of behaviour in TTCN-3, i.e. functions, altsteps, test cases, module control and language constructs for defining test behaviour, e.g. **`send`** and **`receive`** operations, **`if-else`**-, or **`while`**- statements. The meaning of some TTCN-3 constructs is explained by replacing them with other language constructs. For example, **`interleave`** statements are short forms for series of nested **`alt`** statements and the meaning of each **`interleave`** statement is explained by its replacement with a corresponding series of nested alt statements.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

In most cases, the definition of the semantics of a language is based on an abstract syntax tree of the code that shall be described. This semantics does not work on an abstract syntax tree but requires a graphical representation of TTCN-3 behaviour descriptions in form of flow graphs. A flow graph describes the flow of control in a function, alt step, test case or the module control. The mapping of TTCN-3 behaviour descriptions onto flow graphs is straightforward.

SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be01d994745/sist-es-201-873-4-v4-1-1-2009

NOTE: The mapping of TTCN-3 statements onto flow graphs is an informal step and is not defined by using the BNF rules in ES 201 873-1 [1]. The reason for this is that the BNF rules are not optimal for an intuitive mapping because several static semantic rules are coded into BNF rules in order to allow static semantic checks during the syntax check.

5 Structure of the present document

The present document is structured into four parts:

1) The first part (see clause 6) describes restrictions of the operational semantics, i.e. issues related to the semantics, which are not covered by the present document.

2) The second part (see clause 8) defines the meaning of TTCN-3 short cut and macro notations by their replacement with other TTCN-3 language constructs. These replacements in a TTCN-3 module can be seen as pre-processing step before the module can be interpreted according to the following operational semantics description.

3) The third part (see clause 9) describes the operational semantics of TTCN-3 by means of flow graph interpretation and state modification.

4) The fourth part (see clause 10) specifies the mapping of the different TTCN-3 statements onto flow graph segments, which provide the building blocks for flow graphs representing functions, alt steps, test cases and module control.

SIST ES 201 873-4 V4.1.1:2009

6 Restrictions

The operational semantics only covers behavioural aspects of TTCN-3, i.e. it describes the meaning of statements and operations. It does not provide:

a) A semantics for the data aspects of TTCN-3. This includes aspects like encoding, decoding and the usage of data imported from non-TTCN-3 specifications.

b) A semantics for the grouping mechanism. Grouping is related to the definitions part of a TTCN-3 module and has no behavioural aspects.

c) A semantics for the **`import`** statement. The import of definitions has to be done in the definitions part of a TTCN-3 module. The operational semantics handles imported definitions as if they are defined in the importing module.

7 Replacement of short forms

Short forms have to be expanded by the corresponding complete definitions on a textual level before this operational semantics can be used for the explanation of TTCN-3 behaviour.

TTCN-3 short forms are:

- lists of module parameter, constant and variable declarations of the same type and lists of timer declarations;
- stand-alone receiving operations;
- stand-alone altsteps calls;
- **`trigger`** operations;
- missing **`return`** and **`stop`** statements at the end of function and test case definitions;
- missing **`stop`** execution statements;
- **`interleave`** statements;
- **`select-case`** statements;
- **`break`** and **`continue`** statements;
- **`disconnect`** and **`unmap`** operations without parameters; and
- default values of missing actual parameters.

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be01d994745/sist-es-201-873-4-v4-1-1-2009

In addition to the handling of short forms, the operational semantics requires a special handling for module parameters, global constants, i.e. constants that are defined in the module definitions part, and pre-processing macros. All references to module parameters, global constants and pre-processing macros shall be replaced by concrete values. This means, it is assumed that the value of module parameters, global constants and pre-processing macros can be determined before the operational semantics becomes relevant.

NOTE 1: The handling of module parameters and global constants in the operational semantics will be different from their handling in a TTCN-3 compiler. The operational semantics describes the meaning of TTCN-3 behaviour and is not a guideline for the implementation of a TTCN-3 compiler.

NOTE 2: The operational semantics handles parameters of and local constants in test components, test cases, functions and module control like variables. The wrong usage of local constants or **`in`**, **`out`** and **`inout`** parameters has to be checked statically.

SIST ES 201 873-4 V4.1.1:2009

7.1 Order of replacement steps

The textual replacements of short forms, global constants and module parameters have to be done in the following order:

1) replacement of lists of module parameter, constant, variable and timer declarations with individual declarations;
2) replacement of global constants and module parameters by concrete values;
3) replacement of all **`select-case`** statements by equivalent nested **`if-else`** statements;
4) embedding stand-alone receiving operations into **`alt`** statements;
5) embedding stand-alone altstep calls into **`alt`** statements;
6) expansion of **`interleave`** statements;
7) replacement of all **`trigger`** operations by equivalent **`receive`** operations and **`repeat`** statements;
8) adding **`return`** at the end of functions without **`return`** statement, adding **`self.stop`** operations at the end of testcase definitions without a stop statement;
9) adding **`stop`** at the end a module control part without stop statement;
10) expansion of **break** statements;
11) expansion of **continue** statements;
12) adding default parameters to **`disconnect`** and **`unmap`** operations without parameters; and
13) adding default values of parameters.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

NOTE: Without keeping this order of replacement steps, the result of the replacements would not represent the defined behaviour.

SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be01d994745/sist-es-201-873-4-v4-1-1-2009

7.2 Replacement of global constants and module parameters

Constants declared in the module definitions part are global for module control and all test components that are created during the execution of a TTCN-3 module. Module parameters are meant to be global constants at run-time.

All references to global constants and module parameters shall be replaced by the actual values before the operational semantics starts the interpretation of the module. If the value of a constant or module parameter is given in form of an expression, the expression has to be evaluated. Then, the result of the evaluation shall replace all references of the constant or module parameter.

7.3 Embedding single receiving operations into alt statements

TTCN-3 receiving operations are: **`receive`**, **`trigger`**, **`getcall`**, **`getreply`**, **`catch`**, **`check`**, **`timeout`**, and **`done`**.

NOTE: The operations **`receive`**, **`trigger`**, **`getcall`**, **`getreply`**, **`catch`** and **`check`** operate on ports and they allow branching due to the reception of messages, procedure calls, replies and exceptions. The operations **`timeout`** and **`done`** are not real receiving operations, but they can be used in the same manner as receiving operations, i.e. as alternatives in **`alt`** statements. Therefore, the operational semantics handles **`timeout`** and **`done`** like receiving operations.

A receiving operation can be used as stand-alone statement in a function, an altstep or a test case. The **`timeout`** operation can also be used as stand-alone statement in module control. In such a case the receiving operation as considered to be shorthand for an **`alt`** statement with only one alternative defined by the receiving operation. For the operational semantics an **`alt`** statement in which the receiving statement is embedded shall replace all stand-alone occurrences of receiving operations.

SIST ES 201 873-4 V4.1.1:2009

EXAMPLE:

```
// The stand-alone occurrence of
     :
    MyCL.trigger(MyType:?);
     :

// shall be replaced by
     :
    alt {
        []  MyCL.trigger (MyType:?) { }
    }
     :

// or
     :
    MyPTC.done;
     :

// shall be replaced by
     :
    alt {
        []  MyPTC.done { }
    }
     :
```

7.4 Embedding stand-alone altstep calls into alt statements

TTCN-3 allows calling altsteps like functions in functions, altsteps, test cases and module control. The meaning of a stand-alone call of an altstep is given by an **alt** statement with one branch only that calls the altstep. The **alt** statement is responsible for the snapshot that is evaluated within the altstep and for the invocation of the default mechanism if none of the alternatives in the altstep can be chosen.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

NOTE: An altsteps used in module control can only include alternatives with **timeout** operations and an **else** branch.

SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-5d56-4645-8ca7-6be0d994745/sist-es-201-873-4-v4-1-1-2009

EXAMPLE:

```
// The stand-alone occurrence of
     :
    myAltstep(MyPar1Val);
     :

// shall be replaced by
     :
    alt {
        []  myAltstep(MyPar1Val) { }
    }
     :
```

7.5 Replacement of interleave statements

The meaning of an **interleave** statement is defined by its replacement by a series of nested **alt** statements that has the same meaning. The algorithm for the construction of the replacement for an **interleave** statement is described in this clause. The replacement shall be made on a syntactical level.

Within an **interleave** statement it is not allowed:

1) to use the control transfer statements **for**, **while**, **do-while**, **goto**, **activate**, **deactivate**, **stop**, **repeat** and **return**;

2) to call altsteps;

3) to call user-defined functions which include communication operations;

4) to guard branches of the **interleave** statement with Boolean expressions; and

5) to specify **else** branches.

SIST ES 201 873-4 V4.1.1:2009

Due to these restrictions, all not mentioned stand-alone statements (e.g. assignment, **log**, **send** or **reply**), *blocking* **call** operations and the compound statements **interleave**, **if-else** and **alt** can be used within an **interleave** statement.

NOTE 1: Blocking **call** operations and **if-else** statements can be treated like stand-alone statements if they have no embedded **alt** statements. In case of embedded **alt** statements, the alternatives contribute to the **interleave** statement and need a special handling. For simplicity, the algorithm below does not distinguish between these two cases.

NOTE 2: Non-blocking **call** operations are also allowed in interleave statements, they are considered to be stand-alone statements.

The algorithm described in this clause only works for **interleave** statements without embedded **interleave** statements. In case of an **interleave** statement that has embedded **interleave** statements, the embedded **interleave** statements have to be replaced before the algorithm can be applied.

NOTE 3: Due to restrictions 1 to 5, it is always possible to find finite replacements for nested embeddings of **interleave** statements.

The replacement algorithm works on a graph representation of an interleave statement and transforms it into a semantically equivalent tree structure describing a series of nested **alt** statements. For this, a graph representation of stand-alone statements, the compound statements **if-else**, *blocking* **call**, **alt** and **interleave** is needed.

A stand-alone statement is described by a node with the statement as inscription. A sequence of stand-alone statements is described by a set of nodes connected by a flow lines. This is shown in figure 1.

iTeh STANDARD PREVIEW
(standards.iteh.ai)
SIST ES 201 873-4 V4.1.1:2009
https://standards.iteh.ai/catalog/standards/sist/46e3a84e-...-6be01d994745/sist-es-201-873-4-v4-1-1-2009

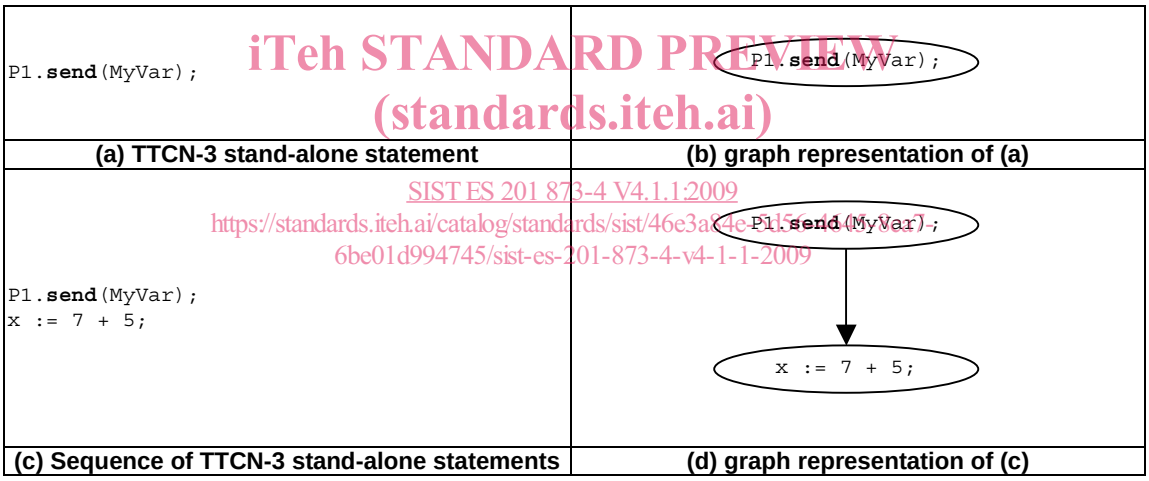

Figure 1: Graph representation of TTCN-3 stand-alone statements

The graph representation of an **if-else** statement is shown in figure 2. An **if-else** statement is represented by an IF node with two flow lines connected to the first statement in the two alternatives. An **if-else** statement without ELSE branch is represented in the same manner, if there are statements following the **if-else** statement. In this case the flow line representing the *else* branch is connected to the first statement following the **if-else** statement. An **if-else** statement without ELSE branch and without following statements is represented by an IF node with one flow line only.

NOTE 4: The inscriptions on the flow lines in figure 1 are introduced for readability purposes only. The algorithm only uses the relation expressed by the flow line and not the inscription.