

**Methods for Testing and Specification (MTS);  
The Testing and Test Control Notation version 3;  
TTCN-3 Language Extensions:  
TTCN-3 Performance and Real Time Testing**

---

**iTeh STANDARD PREVIEW**  
(standards.iteh.ai)  
Full standard:  
<https://standards.iteh.ai/catalog/standards/sist/651e6fca-8ef4-414d-9aed-02754d064657/etsi-es-202-782-v1.1.1-2010-07>



---

Reference

DES/MTS-00113ed111 T3Ext\_Perf

---

Keywords

performance, 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

---

**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/chaicor/ETSI\\_support.asp](http://portal.etsi.org/chaicor/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 2010.  
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.

# Contents

Intellectual Property Rights .....	5
Foreword.....	5
1 Scope .....	6
2 References .....	6
2.1 Normative references .....	6
2.2 Informative references.....	6
3 Definitions and abbreviations.....	7
3.1 Definitions.....	7
3.2 Abbreviations .....	7
4 Package conformance and compatibility.....	7
5 Package concepts for the core language.....	8
5.1 The test system clock .....	8
5.1.1 Accessing the current test system time .....	8
5.1.2 The precision of the system time .....	9
5.2 Communication port types for real-time measurements.....	9
5.3 Measuring timing information for dedicated incoming communication events.....	9
5.3.1 Obtain the reception time for messages with the receive statement.....	10
5.3.2 Obtain the reception time for messages with the trigger statement .....	10
5.3.3 Obtain the reception time for procedure calls with getcall statement.....	11
5.3.4 Obtain the reception time for procedure replies with the getreply statement.....	12
5.3.5 Obtain the reception time for exceptions with the catch statement.....	12
5.4 The wait statement.....	13
6 TRI extensions for the package.....	14
6.1 triStartClock (TE → PA).....	14
6.2 triReadClock (TE → PA).....	14
6.3 triBeginWait (TE → PA).....	15
6.4 triEndWait (PA → TE).....	15
6.5 triEnqueueMsgRT (SA → TE).....	15
6.6 Communication Operations.....	16
6.6.1 triSendRT (TE → SA).....	16
6.6.2 triSendBCRT (TE → SA).....	17
6.6.3 triSendMCRT (TE → SA).....	17
6.6.4 triEnqueueMsgRT (SA → TE).....	18
6.6.5 triCallIRT (TE → SA) .....	18
6.6.6 triCallBCRT (TE → SA) .....	19
6.6.7 triCallMCRT (TE → SA) .....	20
6.6.8 triReplyRT (TE → SA).....	21
6.6.9 triReplyBCRT (TE → SA) .....	22
6.6.10 triReplyMCRT (TE → SA) .....	23
6.6.11 triRaiseRT (TE → SA) .....	24
6.6.12 triRaiseBCRT (TE → SA).....	24
6.6.13 triRaiseMCRT (TE → SA).....	25
6.6.14 triEnqueueCallIRT (SA → TE).....	25
6.6.15 triEnqueueReplyRT (SA → TE).....	26
6.6.16 triEnqueueExceptionRT (SA → TE).....	26
6.7 Definition of Interfaces .....	27
6.8 Changes for Java Language Mapping .....	27
6.8.1 Mapping of interface triCommunicationSART .....	27
6.8.2 Mapping of interface triCommunicationTERT.....	28
6.8.3 Mapping of interface triPlatformPART .....	28
6.8.4 Mapping of interface triPlatformTE .....	28

6.9	Changes for ANSI C Language Mapping.....	29
6.10	Changes for C++ Language Mapping .....	30
6.10.1	Mapping of interface triCommunicationSART .....	30
6.10.2	Mapping of interface triCommunicationTERT.....	31
6.10.3	Mapping of interface triPlatformPART .....	32
6.10.4	Mapping of interface triPlatformTERT .....	32
7	TCI extensions for the package .....	32
<b>Annex A (normative): BNF and static semantics .....</b>		<b>33</b>
A.1	Changed BNF Rules.....	33
A.2	New BNF Rules .....	33
<b>Annex B (informative): Bibliography.....</b>		<b>34</b>
History .....		35

**iTeh STANDARD PREVIEW**  
 (standards.iteh.ai)

Full standard:  
<https://standards.iteh.ai/catalog/standards/sist/651e6fca-8ef4-414d-9aed-02754d064657/etsi-es-202-782-v1.1.1-2010-07>

---

## 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), and is now submitted for the ETSI standards Membership Approval Procedure.

The present document relates to the multi-part standard covering the Testing and Test Control Notation version 3, as identified below:

- ES 201 873-1 [1]: "TTCN-3 Core Language";
- ES 201 873-2 [i.1]: "TTCN-3 Tabular presentation Format (TFT)";
- ES 201 873-3 [i.2]: "TTCN-3 Graphical presentation Format (GFT)";
- ES 201 873-4 [2]: "TTCN-3 Operational Semantics";
- ES 201 873-5 [3]: "TTCN-3 Runtime Interface (TRI)";
- ES 201 873-6 [4]: "TTCN-3 Control Interface (TCI)";
- ES 201 873-7 [i.3]: "Using ASN.1 with TTCN-3";
- ES 201 873-8 [i.4]: "The IDL to TTCN-3 Mapping";
- ES 201 873-9 [i.5]: "Using XML schema with TTCN-3";
- ES 201 873-10 [i.6]: "TTCN-3 Documentation Comment Specification".

---

# 1 Scope

The present document defines the real time and performance testing support package of TTCN-3. TTCN-3 can be used for the specification of all types of reactive system tests over a variety of communication ports. Typical areas of application are protocol testing (including mobile and Internet protocols), service testing (including supplementary services), module testing, testing of CORBA based platforms, APIs, etc. TTCN-3 is not restricted to conformance testing and can be used for many other kinds of testing including interoperability, robustness, regression, system and integration testing. The specification of test suites for physical layer protocols is outside the scope of the present document.

TTCN-3 packages are intended to define additional TTCN-3 concepts, which are not mandatory as concepts in the TTCN-3 core language, but which are optional as part of a package which is suited for dedicated applications and/or usages of TTCN-3.

While the design of TTCN-3 package has taken into account the consistency of a combined usage of the core language with a number of packages, the concrete usages of and guidelines for this package in combination with other packages is outside the scope of the present document.

---

# 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 (V4.2.1): "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1: TTCN-3 Core Language".
- [2] ETSI ES 201 873-4 (V4.2.1): "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 4: TTCN-3 Operational Semantics".
- [3] ETSI ES 201 873-5 (V4.2.1): "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 5: TTCN-3 Runtime Interface (TRI)".
- [4] ETSI ES 201 873-6 (V4.2.1): "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 6: TTCN-3 Control Interface (TCI)".
- [5] ISO/IEC 9646-1: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework; Part 1: General concepts".

## 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 201 873-2: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 2: TTCN-3 Tabular presentation Format (TFT)".

- [i.2] ETSI ES 201 873-3: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 3: TTCN-3 Graphical presentation Format (GFT)".
- [i.3] ETSI ES 201 873-7 (V4.2.1): "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 (V4.2.1): "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 (V4.2.1): "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 201 873-10 (V4.2.1): "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 10: TTCN-3 Documentation Comment Specification".

---

## 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], ES 201 873-4 [2], ES 201 873-5 [3], ES 201 873-6 [4] and ISO/IEC 9646-1 [5] apply.

### 3.2 Abbreviations

For the purposes of the present document, the abbreviations given in ES 201 873-1 [1], ES 201 873-4 [2], ES 201 873-5 [3], ES 201 873-6 [4] and ISO/IEC 9646-1 [5] apply.

---

## 4 Package conformance and compatibility

The package presented in the present document is identified by the package tag:

"TTCN-3:2010 Real Time and Performance Testing" - to be used with modules complying with the present document.

For an implementation claiming to conform to this package version, all features specified in the present document shall be implemented consistently with the requirements given in the present document and in ES 201 873-1 [1], ES 201 873-4 [2], ES 201 873-5 [3] and ES 201 873-6 [4].

The package presented in the present document is compatible to:

- ES 201 873-1 (V4.2.1) [1]
- ES 201 873-4 (V4.2.1) [2]
- ES 201 873-5 (V4.2.1) [3]
- ES 201 873-6 (V4.2.1) [4]
- ES 201 873-7 (V4.2.1) [i.3]
- ES 201 873-8 (V4.2.1) [i.4]
- ES 201 873-9 (V4.2.1) [i.5]
- ES 201 873-10 (V4.2.1) [i.6]

If later versions of those parts are available and should be used instead, the compatibility to the package presented in the present document has to be checked individually.

## 5 Package concepts for the core language

Real-time systems have to respect special requirements for timing. Often functional requirements are directly connected to the timing of the messages and procedure calls. Thus, checking the message values and the message order is not sufficient here. A test component must be able to check whether a message has been received in time and must be able to control the timing for the stimulation.

Thus, a test language has to provide means to measure time, to specify time points and time spans, to control the timing of the stimulation, and to calculate and compare time values. Moreover the test execution engine has to ensure that the specified actions (time measurement, timed stimulation) are executed correctly with respect to the required precision.

To fulfil the requirements for testing real time system we define the following TTCN-3 core language extensions.

- A test system wide available test system clock, that allows the measurement of time during test case execution.
- Means to directly and precisely access the time points of the relevant interaction events between the test system and the system under test.

Real-time measurements at ports require additional resources (e.g. functionality that monitor ports and collect timestamps that describe the reception time of messages, calls, replies or exceptions) that may slow down the test execution. In order to avoid unnecessary delays at ports, such resources may only be provided when needed. An additional **real-time** clause for ports shall indicate the need for real-time measurement at a port.

### 5.1 The test system clock

In RT TTCN-3 time progress is measured with a test system clock. The clock is initialized (set to 0.0) at the beginning of each test case execution and is available during the complete test run in each component. The clock values are represented as float values. The system clock and the already available TTCN-3 timer mechanisms are synchronized with respect to time progress.

#### 5.1.1 Accessing the current test system time

The current value of the test system clock by means of the symbol **now**. The **now** symbol is used as a TTCN-3 expression that yields the current test system clock value in seconds. The test system clock value is represented by means of a **float** number. The symbol **now** can be applied in each expression inside of testcase definitions and function definitions. It is not allowed for the TTCN-3 control part and in guard conditions of alt branches.

EXAMPLE 1:

```
// Use of now to retrieve the actual time
var float myTimePoint := now;
```

EXAMPLE 2:

```
// Use of now to retrieve the send time of a message
var float sendTimePoint;
// ...
p.send(m);
sendTimePoint:= now;
```

EXAMPLE 3:

```
// Measuring time progress
var float startTime;
startTime:= now;
p.send(m1);
// ...
p.receive(m2);
if(now-startTime >= 10.0){...};
```



*Syntactical Structure:*

```
OpCall ::= ConfigurationOps | VerdictOps | TimerOps | TestcaseInstance |
          FunctionInstance | TemplateOps | ActivateOp | NowOperation
```

```
NowOperation ::= NowKeyword
```

```
NowKeyword ::= "now"
```

## 5.1.2 The precision of the system time

The requirements on the overall precision of the test system clock can be specified by means of the stepsize annotation. The stepsize annotation is allowed for modules only and can be used to state the minimal necessary precision for time measurement provided by the test system clock. The precision is defined by means of a charstring value that represents a decimal number which states the smallest necessary time distance in seconds that is measureable by the test system clock. A concrete test system has to fulfil the requirements given by the stepsize annotation to be adequate for the execution of the respective test case definitions. When a test system is not adequate for the test case execution the user shall be informed, at least test run shall end with an error verdict.

EXAMPLE:

```
// specifies the requirement on a necessary precision of a millisecond
module myModule{
...
} with {stepsize "0.001"};
```

In case of module imports with different stepsize annotation the test system has to respect the stepsize annotation with the highest precision.

## 5.2 Communication port types for real-time measurements

This package extends the port type definition of message-based and procedure-based ports with a **realtime** clause. Ports facilitate communication between test components and between test components and the test system interface.

Only instances of ports with a realtime clause shall be used for real-time measurements. This means, the redirection operator `-> timestamp` shall only be used by receiving operations (i.e. the operations **receive**, **trigger**, **getcall**, **getreply** and **catch**) applied to ports with a **realtime** clause.

*Syntactical Structure*

Message-based port:

```
type port PortTypeIdentifier message [realtime] "{"
  { ( in | out | inout ) { MessageType [ "," ] }+ ";" }
}"
```

Procedure-based port:

```
type port PortTypeIdentifier procedure [realtime] "{"
  { ( in | out | inout ) { Signature [ "," ] }+ ";" }
}"
```

## 5.3 Measuring timing information for dedicated incoming communication events

Testing real time systems requires exact timing information that relates directly to the communication (reception and distribution of messages and procedure calls) between the test system and the system under test. The timing information that can be obtained by the **now** symbol or the TTCN-3 timer construct is related to the logical structure of the test program, thus it allows the measurement on TTCN-3 statement level. Time measurement on TTCN-3 statement level may be affected by blocked queues, decoding and matching procedures. It is not exact with respect to the real timing of the reception and disposal of messages and procedure calls at the interface between the test system and the SUT.

RT TTCN-3 introduces a mechanism to store the arrival time of messages, procedure calls at system adapter level. The time points of message reception are automatically registered by the system adapter, communicated to the test executable and stored with the message. The timing information can be retrieved directly at the communication statements by means of the redirection operator `-> timestamp`.

The existing redirections for `getcall`, `getreply`, `receive`, `trigger`, `catch`, and `check` operations are extended by an optional clause `timestamp`. A redirect specification of the form:

```
-> timestamp VariableRef
```

specifies the redirection of the time point, which has been measured at message, procedure call, reply or exception arrival to a given float variable. The redirection is processed when the respective communication statement matches.

### Restrictions

The redirection operator `-> timestamp` shall only be used by receiving operations (i.e. the operations `receive`, `trigger`, `getcall`, `getreply` and `catch`) applied to ports with a `realtime` clause.

## 5.3.1 Obtain the reception time for messages with the receive statement

The existing redirections for `receive` are extended by an optional clause `"timestamp VariableRef"`. A `receive` statement that holds a `timestamp` clause and that is executed successfully (i.e. it matches a message) allocates the given variable with the reception time of the matched message.

EXAMPLE 1:

```
p.receive(t) -> timestamp myTime;
// yields the reception time of a message
if (myTime>MAX) {setverdict(fail);}
```

EXAMPLE 2:

```
interleave{
  [ ] FrontOut.receive(ON) -> timestamp f_actv{
    if (f_actv>MAX) {setverdict(fail);}
  };
  [ ] RearOut.receive(ON) -> timestamp r_actv{
    if (r_actv>MAX) {setverdict(fail);}
  };
}
```

*Syntactical Structure:*

```
( Port | any port ) "." receive ["(" TemplateInstance ")"] [ from AddressRef ]
```

```
[ -> [ value VariableRef ] [ sender VariableRef ] [ timestamp VariableRef ] ]
```

[Note: If several redirect specifications (such as `value VariableRef`, `sender VariableRef`) are used, they have to be separated by a comma (this is not expressed in the above syntax schemas).]

## 5.3.2 Obtain the reception time for messages with the trigger statement

The existing redirections for `trigger` are extended by an optional clause `"timestamp VariableRef"`. A `trigger` statement that holds a `timestamp` clause and that is executed successfully (i.e. it matches a message) allocates the given variable with the reception time of the matched message.

EXAMPLE 1:

```
p.trigger(t) -> timestamp myTime;
// yields the reception time of a message
if (myTime>MAX) {setverdict(fail);}
```

EXAMPLE 2:

```
interleave{
  [ ] FrontOut.trigger(ON) -> timestamp f_actv{
    if(f_actv>MAX){setverdict(fail);}
  };
  [ ] RearOut.trigger(ON) -> timestamp r_actv{
    if(r_actv>MAX){setverdict(fail);}
  };
}
```

*Syntactical Structure:*

( Port | **any port** ) "." **trigger** [ "(" TemplateInstance ")" ] [ **from** AddressRef ]

[ -> [ **value** VariableRef ] [ **sender** VariableRef ] ] [ **timestamp** VariableRef ] ]

### 5.3.3 Obtain the reception time for procedure calls with getcall statement

The existing redirections for getcall are extended by an optional clause "**timestamp** VariableRef". A getcall statement that holds a timestamp clause and that is executed successfully (i.e. it matches an incoming call) allocates the given variable with the reception time of the matched message.

EXAMPLE 1:

```
p.getcall(proc: {m}) -> timestamp myTime;
// yields the reception time of the message call matched by m
if(myTime>MAX){setverdict(fail);}
```

EXAMPLE 2:

```
alt{
  [ ] p.getcall(proc: {m1}) -> timestamp f_actv {
    if(f_actv>MAX){setverdict(fail);}
  };
  [ ] p.getcall(proc: {m2}) -> timestamp r_actv {
    if(f_actv>MAX){setverdict(fail);}
  };
}
```

*Syntactical Structure:*

( Port | **any port** ) "." **getcall** [ "(" TemplateInstance ")" ] [ **from** AddressRef ]

[ "->" [ **param** "(" { VariableRef ":" ParameterIdentifier } "," ] |

{ VariableRef | NotUsedSymbol } "," ]

)" ]

[ **sender** VariableRef ]

[ **timestamp** VariableRef ]

]