Final draft ETSI ES 201 873-1 V4.8.1 (2016-05)

Methods for Testing and Specification (MTS);

The Testing and Test Control Notation version 3;

Part 1: TTCN‑3 Core Language

**ETSI Standard**

Reference

RES/MTS-201873-1 T3ed481

Keywords

language, methodology, testing, TTCN-3

***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***

The present document can be downloaded from:
<http://www.etsi.org/standards-search>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the only prevailing document is the print of the Portable Document Format (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 <https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx>

If you find errors in the present document, please send your comment to one of the following services:
<https://portal.etsi.org/People/CommiteeSupportStaff.aspx>

***Copyright Notification***

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.
The content of the PDF version shall not be modified without the written authorization of ETSI.
The copyright and the foregoing restriction extend to reproduction in all media.

© European Telecommunications Standards Institute 2016.

All rights reserved.

**DECT**TM, **PLUGTESTS**TM, **UMTS**TM and the ETSI logo are Trade Marks of ETSI registered for the benefit of its Members.
**3GPP**TM and **LTE**™ are Trade Marks of ETSI 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.

### B.1.3.3 Permutation

*Permutation* is an operation for matching that shall be used only on values of **record of** and array types. *Permutation* is denoted by the keyword **permutation**. Expressions, templates and *AnyElement* and *AnyElementsOrNone* are allowed as permutation elements. *Permutation* elements shall obey the restrictions given below.

A permutation without *AnyElementsOrNone* in place of a single record of element means that any series of elements is acceptable provided that there is a one to one mapping between elements in the record of and in the permutation list such that each element matches its corresponding element in the permutation list.

*AnyElementsOrNone* used inside permutation (directly or via reference) replaces none or any number of elements within the segment of the record of value matched by permutation. The permutation matching is successful, if a subset of the elements in the record of matches the permutation list without the *AnyElementsOrNone*. If both permutation and *AnyElementsOrNone* are used in a record of template, they shall be evaluated jointly.

NOTE 1: *AnyElementsOrNone* used inside permutation has a different effect as *AnyElementsOrNone* used in conjunction with permutation as in the latter *AnyElementsOrNone* replaces consecutive elements only. For example, {**permutation**(1,2,\*)} is equivalent to ({\*,1,\*,2,\*},{\*,2,\*,1,\*}), while {**permutation**(1,2),\*} is equivalent to ({1,2,\*},{2,1,\*}).

NOTE 2: When *AnyElementsOrNone* is inside a permutation, a length attribute may be applied to *AnyElementsOrNone* to restrict the number of elements matched by *AnyElementsOrNone* (see also clause B.1.4.1).

Besides specifying all individual values, it is possible to add all elements of a **record of** or **set of** template into permutations using an **all from** clause.

***Restrictions***

a) Each individual member listed in the permutation shall be of the type replicated by the **record of** or array type.

b) The member type of the permutation and the member type of the template in the **all from** clause shall be compatible.

c) The template in the **all from** clause as a whole shall not resolve into a matching mechanism other than a *SpecificValue* (see clause B.1.1), and its elements may resolve to the matching mechanisms *SpecificValue*, *AnyElement* and *AnyElementsOrNone* only.

d) Should individual template members of the permutation resolve to an **all from** template clause, the template in the **all from** clause shall obeying to restriction c) above.

***Examples***

EXAMPLE 1:

 **type record of integer** MySequenceOfType;

 **template** MySequenceOfType mw\_myTemplate1 := { **permutation** ( 1, 2, 3 ), 5 };

 // matches any of the following sequences of 4 integers: 1,2,3,5; 1,3,2,5; 2,1,3,5;

 // 2,3,1,5; 3,1,2,5; or 3,2,1,5

 **template** MySequenceOfType mw\_myTemplate2 := { **permutation** ( 1, 2, ? ), 5 };

 // matches any sequence of 4 integers that ends with 5 and contains 1 and 2 at least once in

 // other positions

 **template** MySequenceOfType mw\_myTemplate3 := { **permutation** ( 1, 2, 3 ), \* };

 // matches any sequence of integers starting with 1,2,3; 1,3,2; 2,1,3; 2,3,1; 3,1,2 or 3,2,1

 **template** MySequenceOfType mw\_myTemplate4 := { \*, **permutation** ( 1, 2, 3 )};

 // matches any sequence of integers ending with 1,2,3; 1,3,2; 2,1,3; 2,3,1; 3,1,2 or 3,2,1

 **template** MySequenceOfType mw\_myTemplate5 := { \*, **permutation** ( 1, 2, 3 ),\* };

 // matches any sequence of integers containing any of the following substrings at any position:

 // 1,2,3; 1,3,2; 2,1,3; 2,3,1; 3,1,2 or 3,2,1

 **template** MySequenceOfType mw\_myTemplate6 := { **permutation** ( 1, 2, \* ), 5 };

 // matches any sequence of integers that ends with 5 and containing 1 and 2 at least once in

 // other positions

 **template** MySequenceOfType mw\_myTemplate7 := { **permutation** ( 1, 2, 3 ), \* **length** (0..5)};

 // matches any sequence of three to eight integers starting with 1,2,3; 1,3,2; 2,1,3; 2,3,1;

 // 3,1,2 or 3,2,1

 **template integer** mw\_myInt1 := (1,2,3);

 **template integer** mw\_myInt2 := (1,2,?);

 **template integer** mw\_myInt3 := ?;

 **template integer** mw\_myInt4 := \*;

 **template** MySequenceOfType mw\_myTemplate10 := { **permutation** (mw\_myInt1, 2, 3 ), 5 };

 // matches any of the sequences of 4 integers:

 // 1,3,2,5; 2,1,3,5; 2,3,1,5; 3,1,2,5; or 3,2,1,5;

 // 2,3,2,5; 2,2,3,5; 2,3,2,5; 3,2,2,5; or 3,2,2,5;

 // 3,3,2,5; 2,3,3,5; 2,3,3,5; 3,3,2,5; or 3,2,3,5;

 **template** MySequenceOfType mw\_myTemplate11 := { **permutation** (mw\_myInt2, 2, 3 ), 5 };

 // matches any sequence of 4 integers that ends with 5 and contains 2 and 3 at least once in

 // other positions

 **template** MySequenceOfType mw\_myTemplate12 := { **permutation** (mw\_myInt3, 2, 3 ), 5 };

 // matches any sequence of 4 integers that ends with 5 and contains 2 and 3 at least once in

 // other positions

 **template** MySequenceOfType mw\_myTemplate13 := { **permutation** (mw\_myInt4, 2, 3 ), 5 };

 // matches any sequence of integers that ends with 5 and containing 2 and 3 at least once in

 // other positions

 **template** MySequenceOfType mw\_myTemplate14 := { **permutation** (mw\_myInt3, 2, ? ), 5 };

 // matches any sequence of 4 integers that ends with 5 and contains 2 at least once in

 // other positions

 **template** MySequenceOfType mw\_myTemplate15 := { **permutation** (mw\_myInt4, 2, \* ), 5 };

 // matches any sequence of integers that ends with 5 and contains 2 at least once in

 // other positions

EXAMPLE 2:

 **type record of integer** RoI;

 **template** RoImw\_roI1 := {1, 2, \*};

 **template** RoImw\_roI2 := {**permutation**(0, **all from** mw\_roI1), 4, 5};

 // results in {permutation(0, 1, 2, \*), 4, 5}