Final draft ETSI ES 203 022 V1.5.1 (2022-02)

iTeh STANDARD
PREVIEW

Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; TTCN-3 Language extension: Advanced Matching

(standards.iteh.ai)

ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71dc-4225-819c-99a7962f2494/etsi-es-203-022-v1-5-1-2022-02

Reference
RES/MTS-203022v151

Keywords
conformance, 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 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

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 prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

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

Notice of disclaimer & limitation of liability

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or other professional standard and applicable regulations.
No recommendation as to products and services or vendors is made or should be implied.
No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.
In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

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.

© ETSI 2022.
All rights reserved.

iTeh STANDARD PREVIEW
(standards.iteh.ai)
ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71dc-422[illegible]-5-1-2022-02

Contents

Intellectual Property Rights 5
Foreword 5
Modal verbs terminology 5
1 Scope 6
2 References 6
2.1 Normative references 6
2.2 Informative references 6
3 Definition of terms, symbols and abbreviations 7
3.1 Terms 7
3.2 Symbols 7
3.3 Abbreviations 7
4 Package conformance and compatibility 7
5 Package Concepts for the Core Language 8
5.0 General 8
5.1 Dynamic Matching 8
5.2 Templates with variable bindings 9
5.2.0 General 9
5.2.1 Value retrieval from matching 9
5.2.2 Declaring out parameters for template definitions 10
5.2.3 Matching template parameters and variables 12
5.3 Additional logical operators for combining matching mechanisms 13
5.3.0 General 13
5.3.1 Conjunction 13
5.3.2 Implication 14
5.3.3 Exclusion 15
5.3.4 Disjunction 15
5.4 Repetition 16
5.4.1 General 16
5.4.2 Repetition in record of and array 16
5.4.3 Repetition in string 18
5.4.4 Modifications to ETSI ES 201 873-1, clause 15.11 (Concatenating templates of string and list types) 19
5.4.4.0 General 19
5.4.4.1 Step 1 of modifications to ETSI ES 201 873-1, clause 15.11 19
5.4.4.2 Step 2 of modifications to ETSI ES 201 873-1, clause 15.11 19
5.4.5 Modifications to ETSI ES 201 873-6 20
5.4.5.1 Changes to clause 7.2.2.3.1 (The abstract data type `MatchingMechanism`) 20
5.4.5.2 Changes to clause 7.2.2.3.2 (The abstract data type `MatchingList`) 20
5.4.5.3 Changes to clause 8.3.2.17 (TciMatchingTypeType) 20
5.4.5.4 Changes to clause 8.3.3.1 (Type) 21
5.4.5.5 Changes to clause 9.5 (Data) 21
5.4.5.6 Changes to clause 10.5.2.16 (TciMatchingTypeType) 23
5.4.5.7 Changes to clause 10.5.3.19 (MatchingList) 23
5.4.5.8 Changes to clause 11.3.3.25 (MatchingMechanism) 23
5.4.5.9 Changes to clause 12.4.2.16 (TciMatchingTypeType) 24
5.4.5.10 Changes to clause B.4 (TCI-TL XML Schema for Templates) 24
5.5 Equality operator for the omit symbol and templates with restriction omit 24
5.5.0 General 24
5.5.1 Modifications to ETSI ES 201 873-1, clause 7 (Expressions) 25
5.6 Encodable Values 25
5.6.0 General 25
5.6.1 The **encvalue** Template Restriction 26
5.6.2 Encoding Mutation Annotation 26
5.6.2.0 Description 26
5.6.2.1 Modifications to ETSI ES 201 873-1, clause 22.2.1 (The Send operation) 27

iTeh STANDARD
PREVIEW
(standards.iteh.ai)
ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71dc-4225-819c-99a7962f2494/etsi-es-203-022-v1-5-1-2022-02

5.6.2.2 Modifications to ETSI ES 201 873-1, clause 22.3.1 (The Call operation) 28
5.6.2.3 Modifications to ETSI ES 201 873-1, clause 22.3.3 (The Reply operation) 28
5.6.2.4 Modifications to ETSI ES 201 873-1, clause 22.3.5 (The Raise operation) 28
5.6.2.5 Modifications to ETSI ES 201 873-1, clause C.5.1 (The encoding function) 29
5.6.2.6 Modifications to ETSI ES 201 873-1, clause C.5.3 (The encoding to universal charstring function) 29
5.6.2.7 Modifications to ETSI ES 201 873-1, clause C.5.5 (The encoding to octetstring function) 30
5.7 Extension of the istemplatekind function 30
5.7.1 Modifications to ETSI ES 201 873-1, clause C.3.5 (Matching mechanism detection) 30
5.7.2 Modifications to ETSI ES 201 873-1, clause E.2.2.5 (Matching mechanism detection) 30

6 TCI Extensions for the Package 31
6.1 Extensions to clause 7.2.2.2.0 of ETSI ES 201 873-6, Basic rules 31
6.2 Extensions to clause 7.2.2.3.1 of ETSI ES 201 873-6, The abstract data type MatchingMechanism 33
6.3 Extensions to clause 7.2.2.3.2 of ETSI ES 201 873-6, The abstract data type MatchingList 33
6.4 Extensions to clause 7.2.2.3 of ETSI ES 201 873-6, The abstract data type MatchingMechanism 33
6.5 Extensions to clause 8 of ETSI ES 201 873-6, Java™ language mapping 34
6.6 Extensions to clause 9 of ETSI ES 201 873-6, ANSI C language mapping 37
6.7 Extensions to clause 10 of ETSI ES 201 873-6, C++ language mapping 38
6.8 Extensions to clause 12 of ETSI ES 201 873-6, C# language mapping 41
6.9 Extensions to clause 7.2.2.2.1 of ETSI ES 201 873-6, The abstract data type Value 43
6.10 Extensions to clause 7.2.2.4 of ETSI ES 201 873-6, Data types for complex TTCN-3 properties 43

Annex A (normative): BNF and static semantics 44

A.0 Additional TTCN-3 terminals 44
A.1 Modified TTCN-3 syntax BNF productions 44
A.2 Deleted TTCN-3 syntax BNF productions 45
A.3 Additional TTCN-3 syntax BNF productions 45

History 46

iTeh STANDARD
PREVIEW
(standards.iteh.ai)
ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71dc-4225-819c-99a7962f2494/etsi-es-203-022-v1-5-1-2022-02

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are 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 (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, 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.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM®** and the GSM logo are trademarks registered and owned by the GSM Association.

iTeh STANDARD PREVIEW (standards.iteh.ai)

Foreword

This final draft 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.

ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71de-4225-819e-99a790222494/etsi-es-203-022-v1-5-1-2022-02

The use of underline (additional text) and strike through (deleted text) highlights the differences between base document and extended documents.

The present document relates to the multi-part standard ETSI ES 201 873 covering the Testing and Test Control Notation version 3, as identified in ETSI ES 201 873-1 [1].

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document defines the support of advance matching 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 OMG 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

2.1 Normative 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 referenced document (including any amendments) applies.

iTeh STANDARD PREVIEW

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

(standards.iteh.ai)

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

ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71dc-4225-819c-99a7962f2494/etsi-es-203-022-v1-5-1-2022-02

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 201 873-4: "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: "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: "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 6: TTCN-3 Control Interface (TCI)".

2.2 Informative 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 referenced document (including any amendments) applies.

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

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

[i.2] 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.3] 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.4] 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 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI ES 201 873-1 [1], ETSI ES 201 873-4 [2], ETSI ES 201 873-5 [3] and ETSI ES 201 873-6 [4] apply.

3.2 Symbols

Void.

3.3 Abbreviations

iTeh STANDARD

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

PREVIEW
(standards.iteh.ai)

4 Package conformance and compatibility

ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71dc-4225-819c-99a7962f2494/etsi-es-203-022-v1.5.1-2022-02

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

`"TTCN-3:2017 Advanced Matching"` - 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 ETSI ES 201 873-1 [1] and ETSI ES 201 873-4 [2].

The package presented in the present document is compatible to:

- ETSI ES 201 873-1 [1], version 4.9.1;
- ETSI ES 201 873-4 [2], version 4.6.1;
- ETSI ES 201 873-5 [3], version 4.8.1;
- ETSI ES 201 873-6 [4], version 4.9.1;
- ETSI ES 201 873-7 [i.1];
- ETSI ES 201 873-8 [i.2];
- ETSI ES 201 873-9 [i.3];
- ETSI ES 201 873-10 [i.4].

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

5.0 General

This package defines advanced matching mechanisms for TTCN-3, i.e. new matching mechanisms which go beyond the matching mechanisms defined in ETSI ES 201 873-1 [1]. This package realizes the following concepts:

- Dynamic matching: allows to specify matching in a function-like fashion, i.e. statement blocks and functions can be used to define matching.
- Templates with variable bindings: ease the retrieval field values of received messages and signatures. For this the "->" symbol is used to denote a value assignment to a variable or an **out** parameter in the scope of a template definition in case of a successful template matching.
- Additional logical operators: conjunction, implication, exclusion and disjunction allow the combination matching mechanisms for advanced matching.
- Repetition: allows to match repetitions of a sub-sequence templates inside values of a certain type.
- Restrictions for the omit symbol and templates with restriction omit are relieved allowing omit symbols and templates with restriction omit as operands for the equality operator.

5.1 Dynamic Matching

iTeh STANDARD PREVIEW (standards.iteh.ai)

A dynamic matching is a special matching mechanism. Similar to other matching mechanisms, it can be considered as a Boolean function that indicates successful matching for the value to be matched by returning the value **`true`** and unsuccessful matching by returning the value **`false`**.

Syntactical Structure

@dynamic (*StatementBlock* | *FunctionRef*)

ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71dc-4225-819c-99a7962f2494/etsi-es-203-022-v1-5-1-2022-02

Semantic Description

The *StatementBlock* shall return a value of type **`boolean`**. The value to be matched is referenced by the special keyword **`value`**. When applying this matching mechanism to a value, the *StatementBlock* is executed and if and only if the execution returns **`true`**, the dynamic matching function matches. Unsuccessful matching shall return **`false`**.

A dynamic matching function can only be used in the context of a template, the **`value`** expression inside the *StatementBlock* shall have the same type as the whole template.

The notation **`@dynamic`** *FunctionRef* denotes a shorthand for the special case **`@dynamic { return`** *FunctionRef***`(value) }`** where *FunctionRef* is a reference to a Boolean function with the first parameter compatible with the template's type. If the function contains more than one parameter, all parameters following the first one shall have a default value. The type of the first formal parameter of the referenced function provides the type context, if this template's place of usage does not provide a type context already.

Restrictions

a) The dynamic matching syntax shall only be used when a type context (as defined in clause 3 of ETSI ES 201 873-1 [1]) is provided.

b) The *StatementBlock* shall compute a value of type **`boolean`**.

c) The *StatementBlock* shall be deterministic and side-effect free and follow the restrictions of clause 16.1.4 of ETSI ES 201 873-1 [1].

d) The *StatementBlock* shall not use variables that are declared outside of the *StatementBlock*.

e) The *StatementBlock* shall not use **`inout`** or **`out`** parameters.

f) Only if the dynamic matching syntax appears on the right-hand-side of a parameterized template definition, the formal **in** parameters of that definition may be referenced inside the *StatementBlock*. All other formal **in** parameters shall not be used by the *StatementBlock*.

g) The *FunctionRef* shall not have a **runs on** clause and the *StatementBlock* shall not use any behaviour that has a **runs on** clause.

EXAMPLE:

```
type record of integer Numbers;
template Numbers mw_sorted := @dynamic { // value is of type Numbers
    for (var integer v_i := 1; v_i < lengthof(value); v_i := v_i + 1)  {
        if (value[v_i-1] > value[v_i]) { return false }
    }
    return true;
}   // mw_sorted(v_recInt) matches all values of type Numbers
    // if elements of v_recInt do not break an ascending order

type record Coordinate { float x, float y };
external function @deterministic fx_distance(Coordinate p_a, Coordinate p_b) return float;
template Coordinate mw_closeTo(Coordinate p_origin := { 0.0, 0.0 }, float p_maxDistance := 1.0)
    // access to in parameters is allowed
    := @dynamic { return fx_distance(p_origin, value) <= p_maxDistance; };
    // mw_closeTo(c,d) matches all values of type Coordinate
    // which have maximum distance of d from Coordinate c

external function @deterministic fx_isPrime(integer p_x) return boolean;
:
p.receive(@dynamic fx_isPrime)
// is the same as p.receive(integer:@dynamic { return fx_isPrime(value) })
```

iTeh STANDARD PREVIEW (standards.iteh.ai)

5.2 Templates with variable bindings

5.2.0 General

The possibilities to retrieve field values of received messages and signatures in ETSI ES 201 873-1 [1] are restricted and cumbersome. To overcome this situation, this clause implements the definition of templates with variable bindings that store the actual value matched by a template instance if the matching of all containing templates is successful. This feature is implemented by using the "->" symbol for denoting a value assignment to a variable or an **out** parameter in the scope of a template definition in case of a successful template matching. Such a value assignment can syntactically be specified at all places where a template matching mechanism or a template reference is used in a template definition or an inline template.

ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71de-4225-8104-9fc7962f3494/etsi-es-203-022-v1-5-1-2022-02

5.2.1 Value retrieval from matching

In case of a successful template match, the value which matches the template can be assigned to a variable. This can be specified by using the "->" symbol.

Syntactical Structure

TemplateInstance **"->"** *VariableRef*

Semantic Description

If a template successfully matches a value and contains a value retrieval assignment for a *TemplateInstance*, then, if during the matching process that *TemplateInstance* and all its containing *TemplateInstances* contribute to the successful template match, the part of the value the *TemplateInstance* was matching is assigned to the *VariableRef* referenced in the value retrieval assignment.

EXAMPLE:

```
template integer mw_t1(out integer p)
         := (?, (1..3) -> p); // mw_t1 always matches because of the first list element,
                              // and since the (1..3) does not contribute to the successful match
                              // the out parameter p is not assigned a value
template integer mw_t2(out integer p_small, out integer p_big)
         := ((1..3) -> p_small, (3..5) -> p_big)
```

```
// due to the preference of (1..3) over (3..5),
// matching mw_t2 to number in (1..3) will cause only p_small to be assigned,
// while matching it to number in (4..5) will cause only p_big to be assigned
```

Restrictions

Additional to the restrictions given in clause 5.2.3, the following restrictions apply:

a) If *TemplateInstance* describes the matching of a mandatory element in a template definition, *VariableRef* shall refer to a variable of the same type as the mandatory element.

b) If *TemplateInstance* describes the matching of an optional element in a template definition, *VariableRef* shall refer to a template variable of the same type as the optional element. In case of a successful matching, the matching value or omit shall be assigned to the template variable denoted by *VariableRef*.

c) Value retrieval shall not be used in special places as described in clause 16.1.4 of ETSI ES 201 873-1 [1] with the exception of the matching parts of receiving operations. If value retrieval is used in the matching part of a receiving operation, the assignment to the variables of all the templates is done once the whole matching part matches and all other conditions for the selection of the alternative are met.

d) Value retrieval shall not be applied to templates of set of types or any of their direct or indirect elements, elements of a permutation matching mechanism or any of their direct or indirect elements.

NOTE: TTCN-3 makes no assumptions about the value of the variable or template variable denoted by *VariableRef* My in case of an unsuccessful match.

5.2.2 Declaring out parameters for template definitions

iTeh STANDARD PREVIEW (standards.iteh.ai)

The retrieval of field values in case of successful matching from a structured value can be specified by means of **`out`** parameters of template definitions.

Syntactical Structure

The syntactical structure for global and local template definitions does not change:

template [*restriction*] [**@fuzzy**] *Type TemplateIdentifier* ["(" *TemplateFormalParList* ")"]
[**modifies** *TemplateRef*] ":=" *TemplateBody*

ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71dc-4225-819c-99a7962f2494/etsi-es-203-022-v1-5-1-2022-02

Only the static semantics restriction for formal template parameters change in such a way that formal template parameters shall evaluate to **`in`** or **`out`** parameters.

Semantic Description

The semantics of the value retrieval using templates with `out` parameters can be described by means of the **`match`** operation:

```
// Given the record type definition
type record MyRecordType {
    integer     field1,
    boolean     field2
}

// the constant definition
const MyRecordType c_myRecord := {
    field1 := 7,
    field2 := true
}

// the template definition
template MyRecordType mw_myTemplate1 := {
    field1 := 7,
    field2 := ?
}

// the same template definition with an out parameter matchinstring
template MyRecordType mw_myTemplate2 (out boolean p_matchingBool):= {
    field1 := 7,
    field2 := ? -> p_matchingBool
}
```

```
    // and the variable declarations
    var boolean v_a, v_b;

    // then the effect of
    v_a := match (c_myRecord, mw_myTemplate2(v_b));

    // is identical to
    if (match (c_myRecord, mw_myTemplate1) {
        v_a := true;
        v_b := c_myRecord.field2        // i.e., true
    }
    else {
        v_a:= false;
    }
```

Restrictions

Additional to the restrictions given in clause 5.2.3, the following restrictions apply:

a) *TemplateInstance* with `out` parameters shall not be used as templates of `set of` types or any of their direct or indirect elements.

b) *TemplateInstance* with `out` parameters shall not be used as elements of a permutation matching mechanism or any of their direct or indirect elements. *TemplateInstance* with `out` parameters shall not be used in special places as described in clause 16.1.4 of ETSI ES 201 873-1 [1] with the exception of the matching parts of receiving operations. If *TemplateInstance* with `out` parameters are used in the matching part of a receiving operation, the assignment to the actual `out` parameter variables of all the templates is done only once the whole matching part matches and all other conditions for the selection of the alternative are met.

c) Every formal `out` parameter of a template with `out` parameters shall be referenced at most once inside the template body.

NOTE: In certain cases a template may match without assigning a value to a declared `out` parameter. In such cases, the boundness of the actual `out` parameter has to be checked even after a successful match before using it.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

EXAMPLE 1:

ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71dc-4225-819c-99a7962f2494/etsi-es-203-022-v1-5-1-2022-02

```
type union Tree {
    integer leaf,
    Branch branch
}
type record Branch {
    Tree left optional,
    Tree right optional
}

template Branch mw_branch(out omit Tree p_left, out omit Tree p_right) := {
    left := * -> p_left,
    right := * -> p_right
}
template Tree mw_treebranch(out omit Tree p_left, out omit Tree p_right) :=
    { branch := mw_branch(p_left, p_right) }
template Tree mw_leaf(out integer p_value, in template integer p_expected := ?) :=
    { leaf := p_expected -> p_value }

// compute the sum of absolute values of all leafs in the given tree
function f_absSum(omit Tree p_tree) return integer {
    var omit Tree v_left, v_right;
    var integer v_value;
    if (ispresent(p_tree)) {
        select (p_tree) {
            case (mw_treebranch(v_left, v_right)) {
                    return f_absSum(v_left) + f_absSum(v_right);
                }
            case (mw_leaf(v_value, (0 .. infinity)) { return v_value; }
            case (mw_leaf(v_value) { return -v_value; }
        }
    }
    else {
        return 0; }
}
```

EXAMPLE 2:

```
var Tree v_tree;
…
template Tree mw_anyTree(out integer p_value,
                                    out omit Tree p_left, out omit Tree p_right) :=
    (mw_leaf(p_value), mw_treebranch(p_left, p_right));

// extraction of leaf or branch from v_tree (any value, if present)
if (match(v_tree, mw_anyTree(v_value, v_left, v_right)) {
    if (isbound(v_value)) {
        // v_value might not have been assigned => needs check
    }
    else if (isbound(v_left)) {
        // v_left and v_right might not have been assigned
        // => needs check
    }
}
```

EXAMPLE 3: Receiving with multiple out parameter templates.

```
signature S(out Tree p_tree) return Tree;

alt {
    [match(v_tree, mw_leaf(v_value)] p.receive { }
    // NOT allowed to use template with out parameter in alt guard
[] p.getreply(S:{ mw_leaf(v_value) } value mw_treebranch(v_left, v_right)) {
    // if either mw_leaf or mw_treebranch does not match, v_value, v_left and v_right
    // all remain unchanged
    // it is equivalent with the following:
    // [] p.getreply(S:{mw_leaf(v_value1)} value mw_treebranch(v_left1, v_right1)) {
    //  v_left := v_left1;
    //  v_right := v_right1;
    //   v_value := v_value1;
    // }
    // where v_value1, v_left1 and v_right1 are all internal unbound variables of the
    // appropriate ty
```

iTeh STANDARD PREVIEW
(standards.iteh.ai)

5.2.3 Matching template parameters and variables

ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90fcde-71dc-4225-819e-99a7962f2494/etsi-es-203-022-v1-5-1-2022-02

Matching template parameters and variables can be safely used for value retrieval during matching. Such parameters and variables include the @match modifier in their declaration.

NOTE: Storing template instances which are binding local variable references to formal out parameters or directly to value redirection as normal variables or parameters could lead to situations where the content of the variable is used though the variables that are bound to its content are not in existence anymore. The notion of matching templates allows to distinguish normal templates from templates (potentially) incorporating value retrieval and eliminates the issue with non-existent target variables.

Syntactical Structure

```
var template [ restriction ]
[ (@lazy | @fuzzy) [ @deterministic ] ]
[ @match ]
  Type { VarIdentifier [ ArrayDef ] ":=" TemplateBody }+ [ ";" ]

[ in | inout | out ] template [ Restriction ]
[ ( @lazy | @fuzzy ) [@deterministic] ]
[ @match ]
Type ValueParIdentifier [ArrayDef] [ ":=" ( TemplateInstance | "-" ) ]
```

Semantic Description

Templates are categorized as matching templates if they have at least one of the following properties:

a) They are a matching mechanism with a value retrieval.

b) They are an instance of a template with at least one out parameter.

c) They are a reference to a template variable declared with the @match modifier.

d) They are a reference to a formal parameter declared with the @match modifier.

e) They contain a matching template as one of their fields or elements or part thereof.

Matching templates can be only used in assignments, as actual parameters, in template body notations, in the match operation, in the matching parts of receiving operations and case clauses of the select operation.

Restrictions

a) When used in a formal template parameter declaration, the `@match` modifier is allowed to be present only in *in* parameter declaration. Using the `@match` modifier in *out* or *inout* parameter declaration shall cause an error.

b) If a matching template is assigned to a variable, the variable shall have the `@match` modifier.

c) If a matching template is used as an actual parameter, the corresponding formal parameter shall have the `@match` modifier.

d) Variables with the `@match` modifier shall never be partially initialized. Referencing fields or elements both on the left hand side and on the right hand side of such variables is not allowed.

e) Actual parameters of formal parameters with the `@match` modifier shall be completely initialized.

f) Matching templates shall not be present in a return or raise statement.

g) Matching templates shall not be present in sending operations.

iTeh STANDARD PREVIEW (standards.iteh.ai)

5.3 Additional logical operators for combining matching mechanisms

5.3.0 General

This clause defines the additional logical operators conjunction, implication, exclusion and disjunction for matching mechanisms. Their usage allows the combination matching mechanisms for advanced matching.

ETSI ES 203 022 V1.5.1 (2022-02)
https://standards.iteh.ai/catalog/standards/sist/0a90f6de-71dc-4225-819c-99a7962f2494/etsi-es-203-022-v1-5-1-2022-02

5.3.1 Conjunction

The conjunction matching mechanism is used when a value shall fulfil several distinct conditions.

Syntactical Structure

conjunct "(" { (*TemplateInstance* | **all from** *TemplateInstance*) [","] } ")"

Semantic Description

The conjunction matching mechanism can be used for matching values of all types.

A template specified by the conjunction matching mechanism matches the corresponding value if and only if the value matches all templates listed in the template list.

Besides specifying individual templates, it is possible to add all elements of an existing **`record of`** or **`set of`** template into a conjunction matching mechanism using an **`all from`** clause.

Restrictions

a) The type of the conjunction matching mechanism and the member type of the template in the **`all from`** clause shall be compatible.

b) The template in the **`all from`** clause as a whole shall not resolve into a matching mechanism (i.e. its elements may contain any of the matching mechanisms or matching attributes with the exception of those described in the following restriction).