ETSI GS MEC-DEC 032-3 V2.1.1 (2020-12)

Multi-access Edge Computing (MEC); API Conformance Test Specification; Part 3: Abstract Test Suite (ATS)

Disclaimer

The present document has been produced and approved by the Multi-access Edge Computing (MEC) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference
DGS/MEC-DEC32-3APIConformance

Keywords
API, conformance, MEC, testing

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

Contents

Intellectual Property Rights 4
Foreword 4
Modal verbs terminology 4
1 Scope 5
2 References 5
2.1 Normative references 5
2.2 Informative references 6
3 Definition of terms, symbols and abbreviations 6
3.1 Terms 6
3.2 Symbols 6
3.3 Abbreviations 6
4 ATS conventions 7
4.1 Introduction 7
4.2 Testing conventions 7
4.2.1 TTCN-3 Testing conventions 7
4.2.1.1 Testing states 7
4.2.1.1.1 Initial state 7
4.2.1.1.2 Final state 7
4.2.1.2 Message types - JSON definitions 7
4.2.2 Robot Testing conventions 7
4.3 Naming conventions 8
4.3.1 TTCN-3 Naming conventions 8
4.3.1.1 Introduction 8
4.3.1.2 General guidelines 8
4.3.2 Usage of Log statements 9
4.3.3 Robot Naming conventions 9
Annex A (normative): TTCN-3 Abstract Test Suite (ATS) 10
Annex B (normative): Robot Framework Abstract Test Suite (ATS) 11
Annex C (informative): Test System Execution 12
C.1 Information on execution of the TTCN-3 test suite 12
C.2 Information on execution of the Robot Framework test suite 12
Annex D (informative): Change History 13
History 14

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables 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 (https://ipr.etsi.org/).

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.

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.

Foreword

This Group Specification (GS) has been produced by ETSI Industry Specification Group (ISG) Multi-access Edge Computing (MEC).

The present document is part 3 of a multi-part deliverable. Full details of the entire series can be found in part 1 [i.7].

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

Based on the testing methodology guidelines and framework specified in ETSI GR MEC-DEC 025 [i.1], the present document specifies part 3 of a multi-part deliverable on conformance test specification. Part 3 (the present document) provides the Abstract Test Suites (ATS) in TTCN-3 [i.3] and the Robot Framework [i.5] for the MEC Application Enablement API specified in ETSI GS MEC 011 [2] and the MEC service APIs. The MEC service APIs in scope of the present document are specified in:

- ETSI GS MEC 012 [3];
- ETSI GS MEC 013 [4];
- ETSI GS MEC 014 [5];
- ETSI GS MEC 015 [6];
- ETSI GS MEC 016 [7];
- ETSI GS MEC 021 [8]; and
- ETSI GS MEC 029 [9].

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.

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

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

[1] ETSI GS MEC 001 (V2.1.1): "Multi-access Edge Computing (MEC) Terminology".

[2] ETSI GS MEC 011 (V2.1.1): "Multi-access Edge Computing (MEC); Edge Platform Application Enablement".

[3] ETSI GS MEC 012 (V2.1.1): "Multi-access Edge Computing (MEC); Radio Network Information API".

[4] ETSI GS MEC 013 (V2.1.1): "Multi-access Edge Computing (MEC); Location API".

[5] ETSI GS MEC 014 (V1.1.1): "Mobile Edge Computing (MEC); UE Identity API".

[6] ETSI GS MEC 015 (V1.1.1): "Mobile Edge Computing (MEC); Bandwidth Management API".

[7] ETSI GS MEC 016 (V2.1.1): "Multi-access Edge Computing (MEC); UE application interface".

[8] ETSI GS MEC 021 (V2.1.1): "Multi-access Edge Computing (MEC); Application Mobility Service API ".

[9] ETSI GS MEC 029 (V2.1.1): "Multi-access Edge Computing (MEC); Fixed Access Information API".

[10] ETSI GS MEC-DEC 032-2: "Multi-access Edge Computing (MEC); API Conformance Test Specification; Part 2: Test Purposes (TP)".

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 GR MEC-DEC 025 (V2.1.1) (06-2019): "Multi-access Edge Computing (MEC); MEC Testing Framework".

[i.2] ETSI ETS 300 406 (1995): "Methods for testing and Specification (MTS); Protocol and profile conformance testing specifications; Standardization methodology".

[i.3] ETSI ES 201 873-1 (V4.12.1): "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1: TTCN-3 Core Language".

[i.4] ETSI ES 201 873-11 (V4.8.1): "Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 11: Using JSON with TTCN-3".

[i.5] Robot Framework User Guide, v3.2.2.

NOTE: http://robotframework.org/robotframework/3.2.2/RobotFrameworkUserGuide.html

[i.6] ETSI EG 202 810 (V1.1.1) (03-2010): "Methods for Testing and Specification (MTS); Automated Interoperability Testing; Methodology and Framework".

[i.78] ETSI GS MEC-DEC 032-1: "Multi-access Edge Computing (MEC); API Conformance Test Specification; Part 1: Test Requirements and Implementation Conformance Statement (ICS)".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI GS MEC 001 [1] apply.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in ETSI GS MEC 001 [1] and the following apply:

ATS	Abstract Test Suite
ICS	Implementation Conformance Statement
IUT	Implementation Under Test
PICS	Protocol Implementation Conformance Statement
TTCN	Testing and Test Control Notation

4 ATS conventions

4.1 Introduction

The abstract test suites specified in the present document provide conformance tests for MEC API servers according to the ETSI MEC specifications.

In order to promote adoption and ease deployment of MEC technology, the present document targets a broad set of communities involved in the specific context of MEC. This fundamental and specific goal of MEC is addressed by requiring application of best practices for testing, most relevant for the telecommunication and software engineering communities.

In this regard, the Test Purposes specified in ETSI MEC-DEC 032-2 [10] have been the basis for the development of two Abstract Test Suites, formalized in two different languages for the testing domain: TTCN-3 [i.3] and Robot Framework [i.5].

The quality and coherency between the two ATS are sustained by the common Test Purposes and by a set of conventions used by the contributors. The conventions applied are described in the following clauses of the present document.

4.2 Testing conventions

4.2.1 TTCN-3 Testing conventions

4.2.1.1 Testing states

4.2.1.1.1 Initial state

All test cases start with the function f_cf_01_http_up. This function connects the Test System to the IUT and activates all the default messages processing for error handling.

As necessary, further actions may be included in the f_cf_01_http_up function.

4.2.1.1.2 Final state

All test cases end with the function f_cf_01_http_down. This function disconnects the Test System and stops all default messages processing for error handling.

As necessary, further actions may be included in the f_cf_01_http_down function.

4.2.1.2 Message types - JSON definitions

JSON definitions from MEC APIs are not directly imported in TTCN-3 but they are implemented within the respect of the JSON import method specified in ETSI ES 201 873-11 [i.4].

4.2.2 Robot Testing conventions

Test system configuration is implemented via configuration file name "variables.txt", distributed among individual folders.

Test steps are defined as Robot Framework Keywords either in the common "GenericKeywords.robot" file or within individual test cases files.

PICS are modelled as Tags in the Robot Framework language and can be therefore flagged at runtime to select the set of tests for execution.

4.3 Naming conventions

4.3.1 TTCN-3 Naming conventions

4.3.1.1 Introduction

This test suite follows the naming convention guidelines provided in the ETSI ETS 300 406 [i.2].

4.3.1.2 General guidelines

The naming convention is based on the following underlying principles:

- in most cases, identifiers should be prefixed with a short alphabetic string (specified in table 4.3.1.2-1) indicating the type of TTCN-3 element it represents;
- suffixes should not be used;
- prefixes and suffixes should be separated from the body of the identifier with an underscore ("_");

EXAMPLE 1: `c_sixteen, t_wait`.

- only module names, data type names and module parameters should begin with an upper-case letter. All other names (i.e. the part of the identifier following the prefix) should begin with a lower-case letter;
- the start of second and subsequent words in an identifier should be indicated by capitalizing the first character. Underscores should not be used for this purpose.

EXAMPLE 2: `f_initialState`.

Table 4.3.1.2-1 specifies the naming guidelines for each element of the TTCN-3 language indicating the recommended prefix, suffixes (if any) and capitalization.

Table 4.3.1.2-1: ETSI TTCN-3 generic naming conventions

Language element	Naming convention	Prefix	Example identifier
Module	Use upper-case initial letter	none	AtsMec_AppEnablementAPI_TestCases
Group within a module	Use lower-case initial letter	none	app_saq
Data type	Use upper-case initial letter	none	Headers
Message template	Use lower-case initial letter	m_	m_security_info
Message template with wildcard or matching expression	Use lower-case initial letters	mw_	mw_end_point_uris
Signature template	Use lower-case initial letter	s_	s_callSignature
Port instance	Use lower-case initial letter	none	httpPort
Test component instance	Use lower-case initial letter	none	userTerminal
Constant	Use lower-case initial letter	c_	c_maxRetransmission
Constant (defined within component type)	Use lower-case initial letter	cc_	cc_minDuration
External constant	Use lower-case initial letter	cx_	cx_macId
Function	Use lower-case initial letter	f_	f_authentication()
External function	Use lower-case initial letter	fx_	fx_calculateLength()
Altstep (incl. Default)	Use lower-case initial letter	a_	a_cf_01_http_notif_down ()
Test case	Use ETSI numbering	TC_	TC_MEC_SRV_APPSAQ_001_OK

Language element	Naming convention	Prefix	Example identifier
Variable (local)	Use lower-case initial letter	v_	v_headers
Variable (defined within a component type)	Use lower-case initial letters	vc_	vc_systemName
Timer (local)	Use lower-case initial letter	t_	t_wait
Timer (defined within a component)	Use lower-case initial letters	tc_	tc_authMin
Module parameters for PICS	Use all upper case letters	PICS_	PICS_APP_ENABLEMENT_API_SUPPORTED
Module parameters for other parameters	Use all upper case letters	PX_	PX_APP_INSTANCE_ID
Formal Parameters	Use lower-case initial letter	p_	p_headers
Enumerated Values	Use lower-case initial letter	e_	e_success

4.3.2 Usage of Log statements

All TTCN-3 log statements use the following format using the same order:

- Three asterisks.
- The TTCN-3 test case or function identifier in which the log statement is defined.
- One of the categories of log: INFO, WARNING, ERROR, PASS, FAIL, INCONC, TIMEOUT.
- Free text.
- Three asterisks.

EXAMPLE 1:
```
log("*** " & testcasename() & ": INFO: Registration for
notification succeed ***");
```

NOTE: The INCONC category of log refer to the case of inconclusive test verdict as defined in ETSI EG 202 810 [i.6] and ETSI ES 201 873-1 [i.3], i.e. "test verdict given when the observed test outcome is such that neither a pass nor a fail verdict can be given".

Furthermore, the following rules are applied for the all ATS:

- Log statements are used in the body of the functions, so that invocations of functions are visible in the test logs.
- All TTCN-3 setverdict statements are combined (as defined in ETSI ES 201 873-1 [i.3]) with a log statement following the same above rules (see example 2).

EXAMPLE 2:
```
setverdict(pass, "*** " & testcasename() & : PASS: IUT
successfully responds with a ServiceInfoList ***").
```

4.3.3 Robot Naming conventions

None applicable.

Annex A (normative): TTCN-3 Abstract Test Suite (ATS)

The TTCN-3 MEC API Conformance Test Suite has been developed using the Testing and Test Control Notation (TTCN) according to ETSI ES 201 873-1 [i.3].

This test suite has been compiled error-free using a third party TTCN-3 compiler. The TTCN-3 library modules, which form parts of the present document, are accessible from the ETSI Forge at https://forge.etsi.org/rep/mec/gs032p3-ttcn-test-suite/tree/v2.1.1.

Annex B (normative): Robot Framework Abstract Test Suite (ATS)

The Robot Framework MEC API Conformance Test Suite has been developed using the Robot Framework [i.5].

This test suite has been validated error-free using the official Robot Framework Python distribution. The Robot Framework test definitions, keyword definitions and configuration files, which form parts of the present document, are accessible from the ETSI Forge at https://forge.etsi.org/rep/mec/gs032p3-robot-test-suite/tree/v2.1.1.

Annex C (informative): Test System Execution

C.1 Information on execution of the TTCN-3 test suite

The detailed information needed to compile and run the TTCN-3 test suite is available at https://forge.etsi.org/rep/mec/gs032p3-ttcn-test-suite/blob/v2.1.1/README.md.

The documentation provided suggests possible approach by means of virtualized solutions to install and setup the TTCN-3 environment.

C.2 Information on execution of the Robot Framework test suite

The execution of the Robot Framework test suite comprises three main steps:

1) installation and configuration of the required software, in particular the prerequisites for the Robot Framework;

2) configuration of the IUT specifics (e.g. ICS selection);

3) execution of the test against the IUT.

The detailed information needed to compile and run the Robot Framework test suite is available at https://forge.etsi.org/rep/mec/gs032p3-robot-test-suite/blob/v2.1.1/README.md.

Annex D (informative): Change History

Date	Version	Information about changes
July 2019	V0.0.1	Initial proposal: MECDECODE(19)000015.
Nov 2019	V0.0.2	Incorporated MECDECODE(19)000076.
Nov 2019	V0.0.3	Move to "stable draft" after editHelp! review.
Nov 2020	V0.0.4	Incorporated MECDECODE(20)000057r3, which is considered sufficient to move the document to "final draft".

History

Document history		
V2.1.1	December 2020	Publication