ETSI TS 103 268-4 V1.1.1 (2017-04)

SmartM2M;
Smart Appliances Ontology and Communication Framework Testing;
Part 4: Abstract Test Suite (ATS) and Protocol Implementation eXtra Information for Testing (PIXIT)

Reference
DTS/SmartM2M-103 268-4_SAP_ATS

Keywords
ATS, IoT, M2M, PIXIT, Smart Appliance, 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

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 (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.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Smart Machine-to-Machine communications (SmartM2M).

The present document is part 4 of a multi-part deliverable covering Conformance test specifications for Smart Appliances Ontology and Communication Framework Testing, as identified below:

Part 1: "Testing methodology";

Part 2: "Protocol Implementation Conformance Statement (PICS) pro forma";

Part 3: "Test Suite Structure and Test Purposes (TSS & TP)";

Part 4: "Abstract Test Suite (ATS) and Protocol Implementation eXtra Information for Testing (PIXIT)".

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 contains the Abstract Test Suite (ATS) for SmartAppliances as defined in ETSI TS 103 264 [1] and ETSI TS 103 267 [2] in compliance with the relevant requirements and in accordance with the relevant guidance given in ISO/IEC 9646-7 [6].

The objective of the present document is to provide a basis for conformance tests for SmartAppliances equipment giving a high probability of inter-operability between different manufacturer's equipment.

The ISO standard for the methodology of conformance testing (ISO/IEC 9646-1 [9] and ISO/IEC 9646-2 [10]) as well as the ETSI rules for conformance testing (ETSI ETS 300 406 [11]) and the oneM2M Testing methodology ETSI TS 118 115 [5] are used as a basis for the test methodology.

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 http://docbox.etsi.org/Reference.

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

[1] ETSI TS 103 264: "SmartM2M; Smart Appliances; Reference Ontology and oneM2M Mapping".

[2] ETSI TS 103 267: "SmartM2M; Smart Appliances; Communication Framework".

[3] ETSI TS 118 104: "oneM2M; Service Layer Core Protocol Specification (oneM2M TS-0004)".

[4] ETSI TS 118 112: "oneM2M; Base Ontology (oneM2M TS-0012)".

[5] ETSI TS 118 115: "oneM2M; Testing Framework (oneM2M TS-0015)".

[6] ISO/IEC 9646-7: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 7: Implementation Conformance Statements".

[7] ISO/IEC 9646-6 (1994): "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 6: Protocol profile test specification".

[8] 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".

[9] ISO/IEC 9646-1 (1994): "Information technology -- Open Systems Interconnection -- Conformance testing methodology and framework -- Part 1: General concepts".

[10] ISO/IEC 9646-2 (1994): "Information technology -- Open Systems Interconnection -- Conformance testing methodology and framework -- Part 2: Abstract Test Suite specification".

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

[12] ETSI TS 103 268-2: "SmartM2M; Smart Appliances Ontology and Communication Framework Testing; Part 2: Protocol Implementation Conformance Statement (PICS) proforma".

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.

Not applicable.

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ETSI TS 118 115 [5], in ETSI TS 118 112 [4] and in ISO/IEC 9646-7 [6] apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in ETSI TS 118 115 [5], ETSI TS 103 264 [1], ETSI TS 118 112 [4] and the following apply:

AE	Application entity
APT	Abstract Protocol Tester
ATM	Abstract Test Method
ATS	Abstract Test Suite
BI	Invalid behaviour
BV	Valid behaviour
CoAP	Constrained Application Protocol
CSE	Common Service Entity
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
IPv6	Internet Protocol version 6
IUT	Implementation Under Test
MQTT	Message Queuing Telemetry Transport
MTC	Main Test Component
PA	Platform Adapter
PCTR	Protocol Conformance Test Report
PICS	Protocol Implementation Conformance Statement
PIXIT	Partial Protocol Implementation Extra Information for Testing
PTC	Parallel Test Component
PX	PiXit
SA	System Adapter
SAP	Service Access Point
SUT	System Under Test
TC	Test Case
TP	Test Purposes
TSS	Test Suite Structure
TTCN	Tree and Tabular Combined Notation
UT	Upper Tester

4 Prerequisites and Test Configurations

4.1 Test Configurations

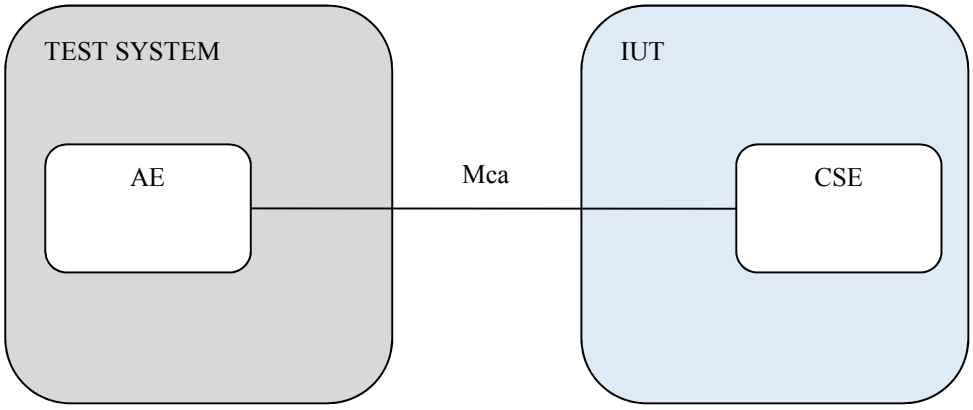

Figure 4.1-1: Test configuration 1 (CF01)

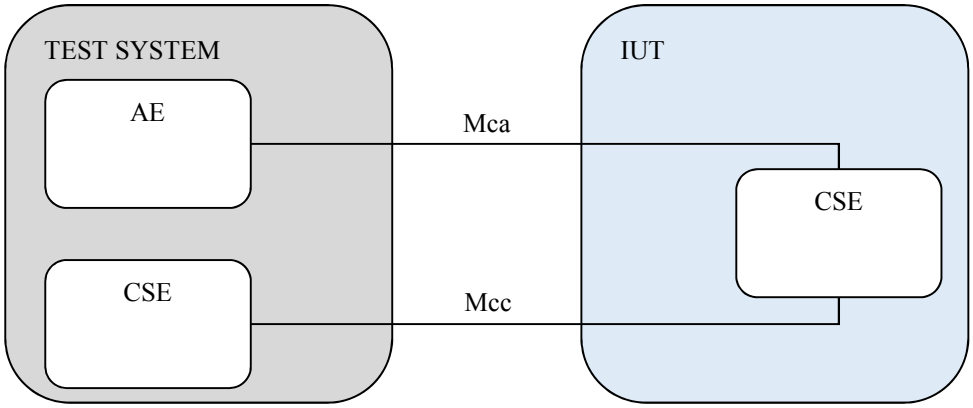

Figure 4.1-2: Test configuration 2 (CF02)

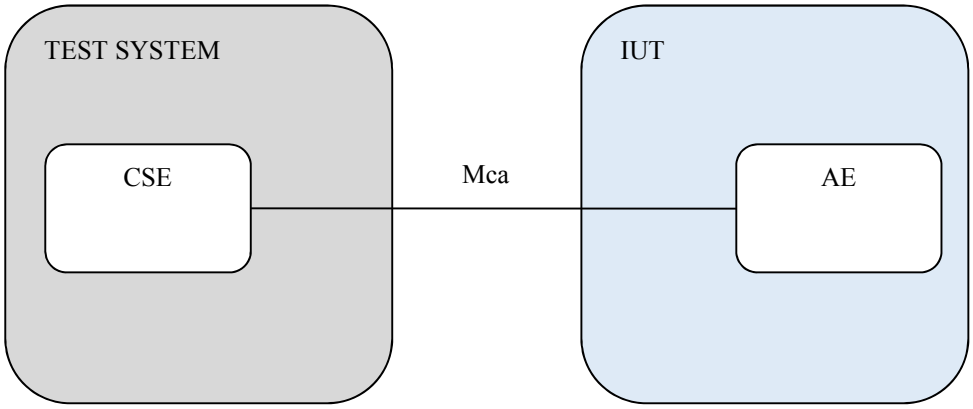

Figure 4.1-3: Test configuration 3 (CF03)

5 Abstract Test Method (ATM)

5.1 Abstract protocol tester

An abstract protocol tester (APT) is a process that provides behaviours for testing an IUT by emulating a peer IUT at the same layer, and enabling to address a single test objective.

APTs used by the SmartAppliances test suite are described in figure 5.1-1. The test system will simulate valid and invalid protocol behaviour, and will analyse the reaction of the IUT.

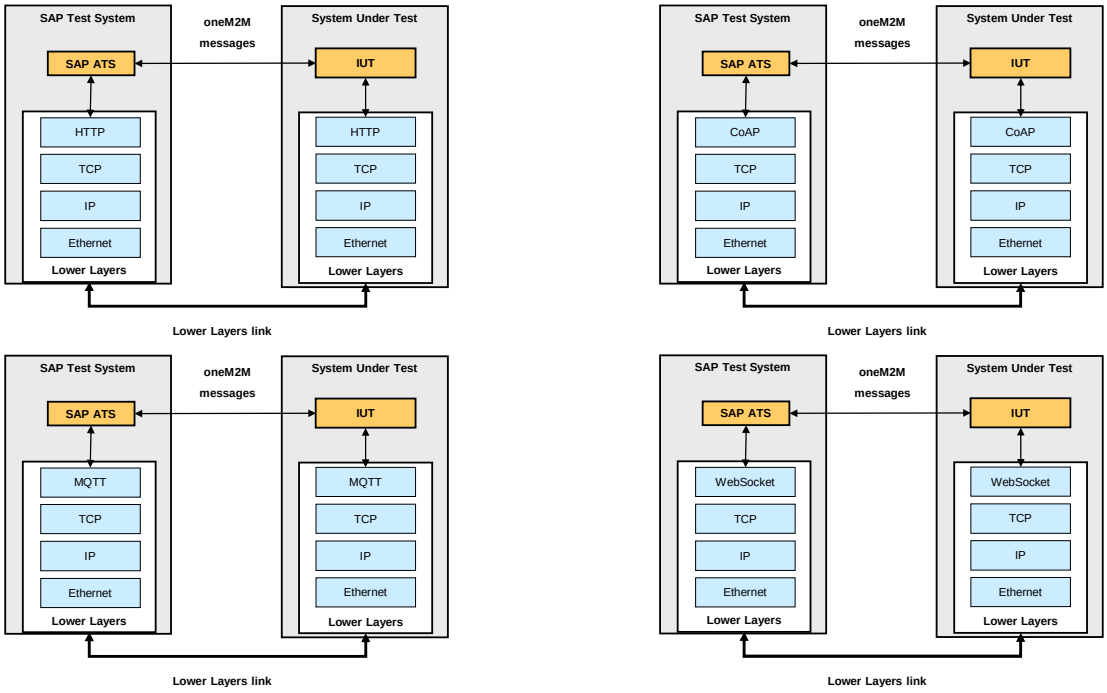

Figure 5.1-1: Abstract protocol testers – SmartAppliances

As figure 5.1-1 illustrates, the corresponding ATS needs to use lower layers to establish a proper connection to the system under test (SUT) over a physical link (Lower layers link). Three different lower layers have been specified corresponding to the binding protocols considered in oneM2M: HTTP, CoAP, MQTT or WebSocket.

5.2 Test Configuration

This test suite uses a unique test configuration in order to cover the different test scenarios. In this configuration, the tester simulates a CSE on Mca interface as defined in Test configuration 3 (CF03) in Figure 4.1-3.

5.3 Test architecture

The approach for the implementation of an Abstract Protocol Tester selected in SmartAppliances follows the recommendation of the oneM2M Testing Framework defined in ETSI TS 118 115 [5] where the TTCN-3 language and its architecture are recommended.

Following this recommendation the SmartAppliances tester architecture comprises a non-platform dependent Test Suite, and a platform dependent part.

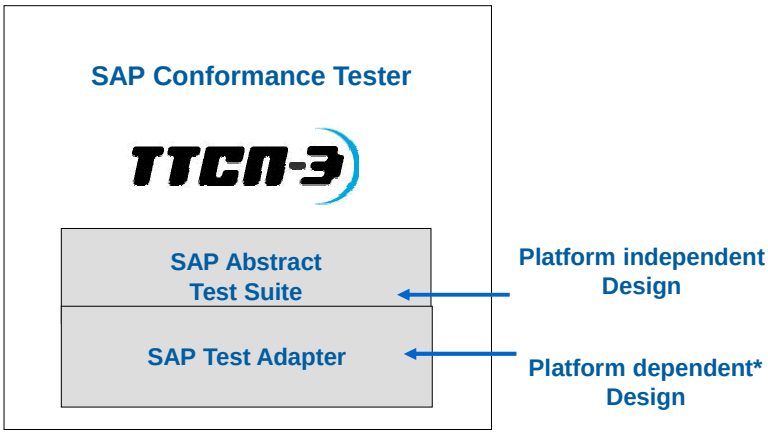

Figure 5.3-1: High level SmartAppliances Test Architecture

- **SmartAppliances TTCN-3 Abstract Test Suite:** the test suite is platform independent, and it is the cornerstone of the architecture. It allows a complete decoupling between the test suite and the rest of the test system. The test suite is composed of a complete set of test cases covering SmartAppliances requirements specified by ETSI TS 103 264 [1] and ETSI TS 103 267 [2].
- **SmartAppliances System Adapter:** this is the platform dependent part that includes adapters and codecs (out of the scope of the present document). This part of the architecture definition depends on the specific platform, operating system and test tool on which the tester is going to run.

However, it can be implemented in a semi-independent manner, which will minimize the dependency to those elements.

Figure 5.3-2 shows the SmartAppliances TTCN-3 test architecture design used for the SmartAppliances ATS. The Test Suite needs to interact with the System Adapter to implement the collection of TTCN-3 test cases that are intended to be used to test the SmartAppliances IUTs.

The SmartAppliances TTCN-3 test cases implement the test algorithms specified in the TSS & TP document ETSI ES 201 873-1 [8], including verdict logic that allows pass/fail diagnosis.

The test algorithms use the interfaces defined in ETSI TS 103 264 [1] and ETSI TS 103 267 [2] (mca, mcc) in order to:

1) control the test event to be sent towards the IUT; and

2) observe the test events received from the IUT.

In TTCN-3 these two interfaces have been implemented through a logical TTCN-3 concept called port (mcaPort and mccPort respectively) which allows oneM2M message primitives exchange with the IUT.

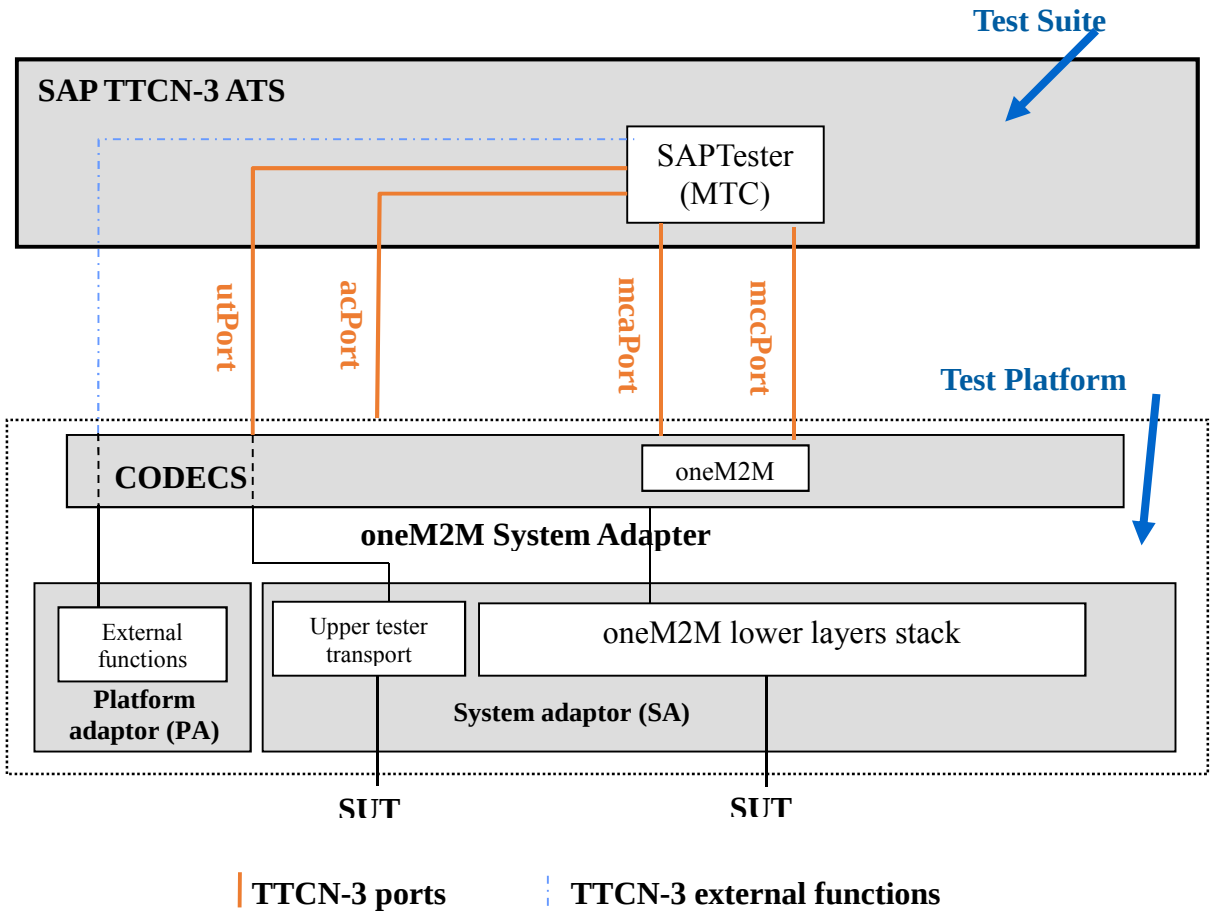

Figure 5.3-2: SmartAppliances Test Architecture

The oneM2M primitive messages have been mapped into TTCN-3 structure. Through this mapping, the TTCN-3 is able to build and send these messages, as well as receive them via the mcaPort and mccPort.

Additionally, the test cases are able to control and configure the test platform through a dedicated port called acPort.

To build up a tester, the test platform shall be also developed (out of scope). This test platform is composed of three adaptation layers:

- PA (Platform Adapter) layer functionality implements the communication between the TTCN-3 modules and external elements that constitute the test tool such as timers and external functions. The External functions are a powerful resources supported by TTCN-3 language. An External function is a function declared at the TTCN-3 level but implemented at the native level.

- SA (System Adapter) layer functionality is divided into two modules:

  - oneM2M lower layers stack module implements the communication with the IUT and carries out the oneM2M primitives messages sent to or received from the IUT. This module is based on TCP or UDP depending on the binding supported by the IUT. The binding is a system adapter parameter.

  - Upper Tester Transport module implements functions that enable triggering specific actions or behaviour on the IUT.

- CODECS layer is the part of the tester to encode and decode messages between the TTCN-3 abstract internal data representation and the format required by the related base standard which the IUT understands. Several CODECS are required in SmartAppliances tester to cope with the bindings considered in oneM2M (HTTP, CoAP, MQTT) and the serialization methods (xml, json).

5.4 Ports and ASPs (Abstract Services Primitives)

5.4.1 Introduction

The SmartAppliances ATS implements four ports:

- The mcaPort
- The mccPort
- The acPort
- The utPort

5.4.2 mcaPort and mccPort

These ports are used to send and receive the following message sets;

- Request Primitives messages in accordance with ETSI TS 118 104 [3].
- Response Primitives messages in accordance with ETSI TS 103 267 [2].

Two primitives are currently defined for this port:

1) MsgOut - to send oneM2M messages to the IUT. Depending on the IUT to be tested:

   a) If the IUT is an AE, these messages are sent by the tester which is associated with the CSE role through the mcaPort.

   b) If the IUT is a CSE, these messages are sent by the tester when it plays the AE role through the mcaPort or sent by the tester when it plays the CSE role through the mccPort.

This primitives contain other parameters that permit to dynamically configure the test adapter for every single sending. These parameters are:

- Host: IP address of the IUT
- XML Namespace: The XML namespace to use for XML serialization
- Protocol binding: the chosen binding (MQTT, WebSocket, CoAP or HTTP)
- Serialization: the chosen serialization method (JSON, XML or CBOR)

2) MsgIn - to receive oneM2M messages from the IUT. Depending on the IUT to be tested:

   a) If the IUT is an AE, these messages are received by the tester which is associated with the CSE role through the mcaPort.

   b) If the IUT is a CSE, these messages are received by the tester when it plays the CSE role through the mccPort or received by the tester when it plays the AE role through the mcaPort.

5.4.3 utPort

The utPort has been included in the SmartAppliances ATS in order to be able to stimulate the IUT and receive extra information from IUT upper layers.

The utPort is not used in the current implementation and is provided for future extension.

5.4.4 acPort

The acPort has been included in the SmartAppliances ATS in order to be able to control and configure the test adapter for specific cases.

The acPort is not used in the current implementation and is provided for future extension.

5.5 External functions

The following external functions are defined:

- fx_base64ToBitstring
  - Description: Converts a base64-encoded blob into bitstring
  - Parameters:
    - p_base64 - Base64-encoded blob to be converted
  - Returns: bitstring corresponding to the de-encoded version of the base64 blob

6 Untestable Test Purposes

No untestable Test Purpose.

7 ATS Conventions

7.1 Introduction

The ATS conventions are intended to give a better understanding of the ATS but they also describe the conventions made for the development of the ATS. These conventions shall be considered during any later maintenance or further development of the ATS.

The ATS conventions contain two clauses, the naming conventions and the implementation conventions. The naming conventions describe the structure of the naming of all ATS elements. The implementation conventions describe the functional structure of the ATS.

To define the ATS, the guidelines of ETSI TS 118 115 [5] were considered.

7.2 Testing conventions

7.2.1 Testing states

7.2.1.1 Initial state

All test cases start with the function f_preamble_XYZ. This function brings the IUT in an "initialized" state by performing some actions such as registration of AE, creation of auxiliary access control policy resource, creation of additional needed resources.

7.2.1.2 Final state

All test cases end with the function f_postamble_XYZ. This function brings the IUT back in an "idle" state which means deletion of all created resources being used by the test case so that next test case execution is not disturbed.

As necessary, further actions may be included in the f_postamble functions.

7.3 Naming conventions

7.3.1 Introduction

This test suite follows the naming convention guidelines provided in ETSI TS 118 115 [5].

7.3.2 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 7.3.2-1) indicating the type of TTCN-3 element it represents;
- suffixes should not be used except in those specific cases identified in table 7.3.2-1;
- 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 7.3.2-1 specifies the naming guidelines for each element of the TTCN-3 language indicating the recommended prefix, suffixes (if any) and capitalization.

Table 7.3.2-1: TTCN-3 generic naming conventions

Language element	Naming convention	Prefix	Example identifier
Module	Use upper-case initial letter	none	SmartAppliances_Templates
Group within a module	Use lower-case initial letter	none	messageGroup
Data type	Use upper-case initial letter	none	SetupContents
Message template	Use lower-case initial letter	m_	m_setupInit
Message template with wildcard or matching expression	Use lower-case initial letters	mw_	mw_anyUserReply
Signature template	Use lower-case initial letter	s_	s_callSignature
Port instance	Use lower-case initial letter	none	signallingPort
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_receiveSetup()
Test case	Use ETSI numbering	TC_	TC_COR_0009_47_ND
Variable (local)	Use lower-case initial letter	v_	v_macId
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_DOOROPEN
Module parameters for other parameters	Use all upper case letters	PX_	PX_TESTER_STATION_ID
Formal Parameters	Use lower-case initial letter	p_	p_macId
Enumerated Values	Use lower-case initial letter	e_	e_syncOk

7.3.3 SmartAppliances specific TTCN-3 naming conventions

Next to such general naming conventions, table 7.3.3-1 shows specific naming conventions that apply to the SmartAppliances TTCN-3 ATS.

Table 7.3.3-1: oneM2M specific TTCN-3 naming conventions

Language element	Naming convention	Prefix	Example identifier
oneM2M Module	Use upper-case initial letter	SmartAppliances_	SmartAppliances_Testcases
Module containing oneM2M types	Use upper-case initial letter	SmartAppliances_Types	SmartAppliances_Types
Module containing types and values	Use upper-case initial letter	SmartAppliances_TypesAndValues	SmartAppliances_TypesAndValues
Module containing Templates	Use upper-case initial letter	SmartAppliances_Templates	SmartAppliances_Templates
Module containing test cases	Use upper-case initial letter	SmartAppliances_Testcases	SmartAppliances_Testcases
Module containing functions	Use upper-case initial letter	SmartAppliances_Functions	SmartAppliances_Functions
Module containing external functions	Use upper-case initial letter	SmartAppliances_ExternalFunctions	SmartAppliances_ExternalFunctions
Module containing components, ports and message definitions	Use upper-case initial letter	SmartAppliances_TestSystem	SmartAppliances_TestSystem
Module containing module parameters	Use upper-case initial letter	SmartAppliances_Pixits	SmartAppliances_Pixits

7.3.4 Usage of Log statements

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

- The TTCN-3 test case or function identifier in which the log statement is defined.
- One of the categories of log: INFO, WARNING, ERROR, TIMEOUT, NONE.
- Free text.

EXAMPLE 1:
```
log("f_utInitializeIut: INFO: IUT initialized");
```

Furthermore, the following rules are applied too:

- All TTCN-3 setverdict statements are combined (as defined in TTCN-3 - ETSI ES 201 873-1 [8]) with a log statement following the same above rules (see example 2).

EXAMPLE 2:
```
setverdict(pass, "TC_SAP_SAREF_BV_001: Received correct message");
```

7.3.5 Test Case (TC) identifier

Table 7.3.5-1: TC naming convention

Identifier:	**TC_<root>_<gr>_<sgr>_<x>_<nn>**		
	<root> = root	SAP	SmartAppliances
	<gr> = group	SAREF	
	<sgr> = subgroup		
	<x> = type of testing	BV	Valid Behaviour tests
		BI	Invalid Behaviour tests
	<nn> = sequential number		001 to 999

EXAMPLE: TP identifier: TP/SAP/SAREF//BV/001
TC identifier: TC_SAP_SAREF_BV_001

Annex A (normative): Partial PIXIT pro forma

A.0 The right to copy

Notwithstanding the provisions of the copyright clause related to the text of the present document, ETSI grants that users of the present document may freely reproduce the Partial PIXIT pro forma in this annex so that it can be used for its intended purposes and may further publish the completed Partial PIXIT.

A.1 Introduction

The PIXIT Pro forma is based on ISO/IEC 9646-6 [7]. Any additional information which may be needed can be found in this international standard document.

A.1 Identification summary

Table A.1-1

PIXIT Number:	
Test Laboratory Name:	
Date of Issue:	
Issued to:	

A.2 ATS summary

Table A.2-1

Protocol Specification:	
Protocol to be tested:	
ATS Specification:	
Abstract Test Method:	

A.3 Test laboratory

Table A.3-1

Test Laboratory Identification:	
Test Laboratory Manager:	
Means of Testing:	
SAP Address:	

A.4 Client identification

Table A.4-1

Client Identification:	
Client Test manager:	
Test Facilities required:	

A.5 SUT

Table A.5-1

Name:	
Version:	
SCS Number:	
Machine configuration:	
Operating System Identification:	
IUT Identification:	
PICS Reference for IUT:	
Limitations of the SUT:	
Environmental Conditions:	

A.6 Protocol layer information

A.6.1 Protocol identification

Table A.6.1-1

Name:	
Version:	
PICS References:	SmartM2M TS 103 268-2 [12]

A.7 PIXIT items

Each PIXIT item corresponds to a Module Parameter of the ATS.

Table A.7-1: SmartAppliance Pixits

Id	Identifier	Type	Description
Configuration			
1	PX_DEVICE_URI	charstring	The base URI of the SmartAppliance

Annex B (informative): TTCN-3 library modules

B.1 Electronic annex, zip file with TTCN-3 code

The TTCN-3 library modules are contained in archive ts_103268004v010101p0.zip which accompanies the present document.

This ATS has been produced using the Testing and Test Control Notation (TTCN) according to ETSI ES 201 873-1 [8].

History

Document history		
V1.1.1	April 2017	Publication