ETSI EG 202 810 V1.1.1 (2010-03)

ETSI Guide

**Methods for Testing and Specification (MTS);
Automated Interoperability Testing;
Methodology and Framework**

Reference

DEG/MTS-00119 AutoIOP_Fwk

Keywords

interoperability, methodology, 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

Foreword

This ETSI Guide (EG) has been produced by ETSI Technical Committee Methods for Testing and Specification (MTS).

Introduction

The encouragement of the European Commission for the adoption and promotion of generic test frameworks for the validation of standards based on multiple stacks including middleware provides evidence that successful testing as well as interoperability is key factors to enable the use of new technologies providing all benefits attached to them including competitiveness and innovation. However, technologies are becoming more complex, collaborative, and inter-dependant. Therefore, methodologies and approaches for ensuring interoperability need to be innovative and consider new evolving challenges such as the distribution of components and their remote access in an embedded environment. This guide adapts and presents a solid and proven method to these new challenges.

The current and future e-communication market can be described as a convergent multimedia market with an increasingly complex structure. Within the present competitive environment, the risk of non-interoperability is increasing due to a fast evolution of technology and the use of non-open standards. The main purpose of standardization is to enable interoperability in a multi-vendor, multi-network, multi-service environment. The absence of interoperability should not be the reason why final services for which there is great demand do not come into being.

ETSI is very much aware of these developments and market demands. At ETSI, the inhibitors to interoperability that can be encountered during the standards development process are well known. A key part of this process is the development of test specifications for conformance and interoperability, and the provision of validation services based on many years of experience.

On one hand, testing the conformance of every single entity (protocol layer) in a complex system seems often prohibitively expensive. On the other hand, pure interoperability testing does not ensure that entities strictly adhere to standard specifications. For example, ETSI observed that 90 % of tests executed at an interoperability event showed that systems were interoperable. However, only 60 % of these executed tests showed reference points conforming to the standard used to link these systems together. In a production environment, this would undoubtedly lead to interoperability problems. Over the past few years, a combination of these two testing approaches has evolved. At ETSI, this approach is called interoperability testing with conformance checking in which key reference points are observed during the execution of end-to-end interoperability tests to evaluate if the message flow at these points conforms to the requirement stated in standards. So far, this approach has shown very promising results but has mainly applied with manual interoperability testing.

The focus of the present document is a generic methodology for automated interoperability testing with conformance checking of complex distributed systems. The proof of concept of this methodology has been done through a case study in the domain of IP Multimedia Subsystem (IMS). However, also other technologies such as IPv6, Robust Header Compression (ROHC), Health Level 7 messaging, grid, cloud, WiMax, IPTV, WiMedia, Voice over IP (VoIP) for air traffic control, and smart cards have been analyzed and considered in the writing of the present document.

1 Scope

The present document describes a methodology for automated interoperability testing for the validation of standards, e.g. in the context of interoperability events and for the validation of products against standards. It extends and complements the best working practices presented in EG 202 237 [i.1] which targets the certification of products based on interoperability testing.

The present document also proposes a generic framework for interoperability testing of distributed systems and includes guidance on, for example, access to reference points, the establishment and maintenance of test synchronisation, and the assignment of test verdicts.

2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific.

- For a specific reference, subsequent revisions do not apply.
- Non-specific reference may be made only to a complete document or a part thereof and only in the following cases:
    - if it is accepted that it will be possible to use all future changes of the referenced document for the purposes of the referring document;
    - for informative references.

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 indispensable for the application of the present document. For dated references, only the edition cited applies. For non-specific references, the latest edition of the referenced document (including any amendments) applies.

Not applicable.

2.2 Informative references

The following referenced documents are not essential to the use of the present document but they assist the user with regard to a particular subject area. For non-specific references, the latest version of the referenced document (including any amendments) applies.

[i.1] ETSI EG 202 237: "Methods for Testing and Specification (MTS); Internet Protocol Testing (IPT); Generic approach to interoperability testing".

[i.2] ETSI White Paper No.3 (Sophia-Antipolis, France, 2008): "Achieving Technical Interoperability - the ETSI Approach", Hans van der Veer and Anthony Wiles.

[i.3] ETSI EG 202 568: "Methods for Testing and Specification (MTS); Internet Protocol Testing (IPT); Testing: Methodology and Framework".

[i.4] ISO/IEC 9646 (parts 1 to 7): "Information technology - Open Systems Interconnection - Conformance testing methodology and framework".

[i.5] 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".

NOTE: Also published as ITU-T Recommendation series Z.140.

[i.6] ETSI ES 202 553: "Methods for Testing and Specification (MTS); TPLan: A notation for expressing Test Purposes".

[i.7] 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".

NOTE: Also published as ITU-T Recommendation series Z.140.

[i.8] S. Schulz: "Test suite development with TTCN-3 libraries", International Journal on Software Tools for Technology Transfer, 10(4), 327-36, Springer, 2008.

[i.9] ETSI TR 102 788: "Methods for Testing and Specification (MTS); Automated Interoperability Testing; Specific Architectures".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in EG 202 237 [i.1] and the following apply:

abnormal test termination: describes the result of executing an abstract test case after it has been prematurely terminated by the test system

abstract test case: complete and independent specification of the action required to achieve a specific test purpose, defined at the level of abstraction of a particular abstract test method, starting and ending in a stable testing state (ISO/IEC 9646-1 [i.4])

abstract test suite: test suite composed of abstract test cases (ISO/IEC 9646-1 [i.4])

base specification: specification of a protocol, abstract syntax, encoding rules, or information object

end-to-end testing: testing approach focused on verifying the correct behaviour of a set of interconnected systems by stimulating and observing the system functionalities from the end user's point of view

executable test case: realization of an abstract test case (ISO/IEC 9646-1 [i.4])

executable test suite: test suite composed of executable test cases (ISO/IEC 9646-1 [i.4])

fail verdict: test verdict given when the observed test outcome either demonstrates non-interoperability of equipment under test with respect to the end-to-end functionality on which the test case is focused, or demonstrates non-conformance with respect to at least one of the conformance requirement(s) on which the test purpose(s) associated with the test case is (are) focused, or contains at least one invalid test event, with respect to the relevant specifications

NOTE: The above definition extends the original definition for this term provided in ISO/IEC 9646-1 [i.4].

inconclusive verdict: test verdict given when the observed test outcome is such that neither a pass nor a fail verdict can be given (ISO/IEC 9646-1 [i.4])

pass verdict: test verdict given when the observed test outcome gives evidence of all equipment under test interoperating for the end-to-end functionality on which the test case is focused, or conformance to the conformance requirement(s) on which the test purpose(s) of the test case is (are) focused, and when all the test events are valid with respect to the relevant specifications

NOTE: The above definition extends the original definition for this term provided in ISO/IEC 9646-1 [i.4].

test architecture: abstract description of logical entities as well as their interfaces and communication links involved in a test

test case: generic, abstract or executable test case (ISO/IEC 9646-1 [i.4])

test configuration: concrete instance of a test architecture defined on the basis of test components, ports and their connection

test description: informal description of a test case usually written in English prose

test event: indivisible unit of test specification at the level of abstraction of the specification (e.g. sending or receiving a single message) (ISO/IEC 9646-1 [i.4])

test log: human readable record of information produced as a result of a test campaign which is sufficient to record the observed test outcomes and to verify the assignment of the test result (including test verdicts)

test oracle: mechanism that determines a test verdict

test outcome: sequence of test events, together with associated data and/or parameter values, which occurred during test execution of a specific parameterised executable test case (ISO/IEC 9646-1 [i.4])

test purpose: description in English prose or TPLan [i.6] of a well defined objective of testing, focusing on either a specific end-to-end functionality, a single conformance requirement or a set of related conformance requirements as specified in the base specification (e.g. verifying the support of a specific value of a specific parameter)

NOTE: The above definition extends the original definition for this term provided in ISO/IEC 9646-1 [i.4].

test step: named subdivision of a test case, constructed from test events and/or other test steps (ISO/IEC 9646-1 [i.4])

test report: document that presents test results and other information relevant to a test

test result: test verdicts and associated information produced as a result of running a test case

test verdict: statement of "pass", "fail" or "inconclusive", specified in an abstract interoperability test case, concerning either the end-to-end interoperability of two or more Equipment Under Test (EUT) or conformance of an Implementation Under Test (IUT) with respect to that test case when it is executed

NOTE: The above definition extends the original definition for this term provided in ISO/IEC 9646-1 [i.4].

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ATS	Abstract Test Suite
E2E	End-to-End
ETS	Executable Test Suite
ETSI	European Telecommunications Standards Institute
EUT	Equipment Under Test
IPT	Internet Protocol Testing
IUT	Implementation Under Test
MTS	Methods for Testing and Specification
SUT	System Under Test
TC	Test Case
TD	Test Description
TP	Test Purpose
TTCN-3	Testing and Test Control Notation
UE	User Equipment

4 Basic concepts and guidelines

4.1 Interoperability of distributed systems

Interoperability is a key factor in the widespread commercial success of any given technology in the telecommunication sector. Interoperability fosters diversity as well as competition in a market. Vendors can achieve interoperability of their products only if they agree and implement a common set of open standards. However, standardization does not necessarily lead to interoperability. Standards have to be engineered for interoperability.

Generally, interoperability can be defined as "the ability of two systems to interoperate using the same communication protocol" [i.1] and [i.2]. In the case of distributed systems, this means that they provide their users a specified end-to-end (E2E) functionality. This implies that each component of a distributed system is able to exchange information with other components across standardized interfaces using the communication protocols and procedures in order to provide specified E2E functionality to the end user.

4.2 Interoperability testing of distributed systems

Systems that implement a set of standards need to be assessed for their interoperability with other systems. It is also necessary to demonstrate that they conform to these standards. At the standardization level, this in turn can be facilitated with the availability of open and validated test specifications.

Interoperability testing is the most intuitive way of confirming that two or more systems work together. A number of standardization organizations use interoperability testing events as a means to raise the status of a specification to the level of a standard. At ETSI, interoperability testing events, so called Plugtests™, are organized and executed to provide feedback to technical bodies on the maturity of a given technology and its underlying standards. Such events can only be successful if testing is based upon an agreed set of interoperability tests.

The purpose of interoperability test specifications is to assess that a communicating system can provide E2E functionality as defined in a specification, e.g. a set of standards. In the context of distributed systems, each system is called an Equipment Under Test (EUT) and the collection of all EUTs is called the System Under Test (SUT) as defined in EG 202 237 [i.1]. In order to verify the correctness of the protocol procedures and to provide a basis for fault analysis, interoperability test specifications can be combined with supplementary conformance checks when assessing E2E functionality. Such conformance checks are associated with standardized interfaces between different EUTs.

In addition, EG 202 237 [i.1] describes the basic concepts of interoperability testing, a means of describing test architectures and a process for developing and executing interoperability test specifications. Figure 1 depicts an example of an interoperability test setup combining E2E interoperability tests with conformance checks. The present document has been derived from the basic concepts and generic approach to interoperability testing which have been defined in EG 202 237 [i.1] for the purpose of using interoperability testing for product certification. It complements and extends these concepts by covering aspects related to the testing of distributed systems and the automation of interoperability testing for the purpose of validating standards and of validating products against standards.

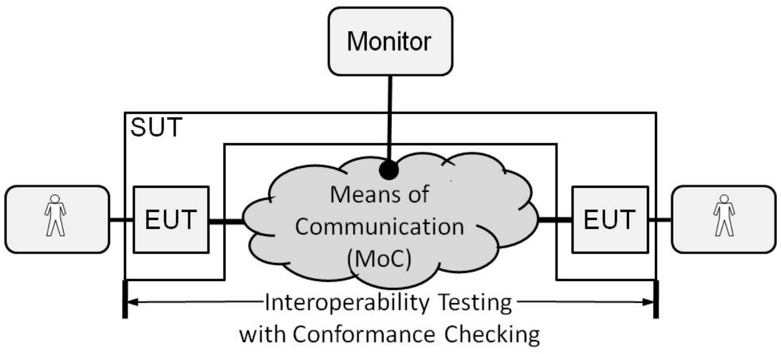
Monitor
SUT
EUT
Means of Communication (MoC)
EUT
Interoperability Testing with Conformance Checking

Figure 1: Example of interoperability testing with two EUTs and conformance checking

4.3 Verdicts

Automated interoperability testing of distributed systems produces a large amount of information on several interfaces. In order to manage this information and to speed up troubleshooting, two verdicts should be managed independently as described in the following.

- The E2E interoperability verdict which is based on the observation of SUT behaviour at its end points. It should assume the following values:
  - **pass:** the observed test outcome demonstrates that all EUTs interoperate when providing the E2E functionality required by the test;
  - **fail:** the observed test outcome demonstrates that some or all of the EUTs do not interoperate when providing the E2E functionality required by the test.
- The conformance verdict which is based on observation of protocol procedures and messages exchanged at interfaces between EUTs as shown in figure 2. It should assume the following values:
  - **pass:** the observed test outcome demonstrates conformance to the normative requirement(s) tested by the associated test case;
  - **fail:** the observed test outcome demonstrates non-conformance to at least one of the to the normative requirement(s) tested by the associated test case;
  - **inconclusive:** the observed test outcome is such that neither a pass nor a fail verdict can be given.

Table 1 shows different possible scenarios in verdict observations as well as their potential source of error. Note that verdicts alone do not allow direct identification of the cause of a test failure. Resolution of test failures is generally not trivial and may require troubleshooting.

Table 1: Example verdict scenarios

End-to-end Verdict	Conformance verdict	Interpretation
Pass	Pass	EUTs have been interoperating and communicating according to the standard.
Pass	Fail	EUTs have been interoperating but not communicating according to the standard. This may be due to a problem in the base standard, the test system, or one of the EUTs, e.g. a mandatory check should be optional.
Pass or Fail	Inconclusive	EUTs have been interoperating or not, all captured communication is according to the standard, but one interface with conformance checks has not been available for evaluation.
Fail	Fail	EUTs have not been interoperating and not communicating according to the standard. This may be due to a problem in the standard, the test system, or one of the EUTs.

Each test step in an interoperability test may lead to intermediate verdict. Inserting checkpoints in a test to capture intermediate verdicts for each significant test event can help to speed up the evaluation of the test execution results.

4.4 Automation

Reducing time to market for a new product, service, or architecture is a strategic requirement in a competitive environment, where it is important to introduce new services and technologies before or, at least not after competitors. Automation can significantly reduce the time for testing and avoids repetitive manual activity which is prone to error. Consequently, it leads to a considerable cost savings in terms of human expert resource as well as test bed usage, i.e. test equipment and EUT resource, acceptance costs both for suppliers and for customers, as well as manual interaction related to test execution, the analysis of trace and message contents and reporting.

A typical application of automated testing systems is related to regression testing. In this scenario, testing specialists are usually involved in repeating several times the same test suite consisting in a large number of tests. Nevertheless, automation also helps in settings such as interoperability events, e.g. to analyze interoperability execution traces of the same test for different vendor pairings.

A general prerequisite for test automation is the availability and accessibility of test interfaces used by the test specification to stimulate, observe or monitor the System Under Test (SUT).

4.4.1 Limitations

A cost/benefit analysis carried out prior to the development of test specifications can identify limitations and assist in determining a suitable degree of automation for interoperability test cases. This analysis should consider the topology of the distributed system, the protocols involved and their complexity, the stability of the specifications, and the resource required to develop a test system in terms of specialist manpower and hardware and software devices required.

The degree of test automation is often a compromise between testing requirements and feasibility. It may not be profitable, for example, to automate the entire testing process since the resources required to implement all or part of the tests may be prohibitively expensive. In addition, some interfaces may require significantly less effort to assess manually.

4.4.2 Degree of automation

The following types of interoperability testing activity may be automated:

- configuration of EUTs and of the interconnecting network;
- monitoring of relevant interfaces between EUTs;
- validation of EUT communication;
- simulation or emulation of (external) equipment;
- operation of all equipment involved in an interoperability test, e.g. EUTs;
- computation of test verdicts;
- test execution;
- generation of test reports.

The ability to automate these activities, i.e. the possible degree of automation, depends on the limitations described in the previous clause. The desired degree of automation depends on the context in which the tests are expected to be executed. For example, if tests are executed in the context of an interoperability event, it may not be desirable to emulate complex interfaces to the EUT as their emulation could lead to the introduction of interoperability problems in and might mask or overwhelm any errors that exist in the EUT. In such a case, the use of real equipment (whose operation could possibly be automated) is preferable. In practice, complete automation of interoperability testing is rarely achieved and automated test steps are usually complemented by a set of manually performed checks and actions.

Test systems should be adaptable to limitations of different equipment. It should be possible to control the interfaces monitored or automatically operated during a test execution since some interfaces may or may not be available for automation of an interoperability test. Similarly, it is desirable to control the general degree of automation during test execution. It should be possible to execute tests in a live mode where the operation of equipment and interface analysis is automated or in an offline mode where only interface analysis is automated.

4.5 Test case development

Automated interoperability testing implies the availability of abstract and executable test cases that specify interactions with one or more EUTs via at least one interface. This clause covers basic guidelines that should be followed in the development of software for automated interoperability tests. Not all of them are necessarily specific to the development of interoperability test specifications but are listed in this guide for completeness.

4.5.1 Requirements on test description content

A pre-requisite for the specification of executable test cases is the availability of unambiguous test descriptions. Such descriptions should capture informally all equipment required for a test, pre-conditions, equipment operation and observation, as well as optionally protocol messages (and their contents) or procedures to be checked between EUTs.

Following the existing ETSI testing methodology [i.3], the result of the interoperability test development process is a collection of interoperability test descriptions (TDs). A TD details test steps that should be followed in order to achieve the test purpose (TP) of each interoperability test. A TD might also include conformance TPs that are checked during the interoperability test. TDs are usually written in natural language and independent of a specific system implementation as well as proprietary interfaces. For automation of interoperability testing, at least one test case (TC) should be derived from each test description and should be written in a testing language (e.g. Testing and Test Control Notation (TTCN-3) [i.5]) or in a scripting language.

Whereas [i.3] allows the derivation of test case specifications directly from test purposes, this methodology for automated interoperability test development strongly recommends to specify test descriptions as an intermediate step. Test descriptions provide valuable and easily understandable documentation. Additional test documentation in interoperability testing is especially important because of the large number of different interfaces involved in a test as well as its basis on multiple EUTs.

In order to facilitate the specification of test cases an interoperability test description should include as a minimum:

- a unique test description ID;
- a concise summary of the test which should reflect the purpose of the interoperability test and enable readers to easily distinguish this test from any other test in the document;
- a list of references to the base specification section(s), use case(s), requirement(s), conformance test purpose(s) which are either used in the test or define the E2E functionality and conformance criteria being tested;
- a list of features and capabilities which are required to be supported by the SUT in order to execute this test;
- a list of all required equipment for testing and possibly also including (a reference to) an illustration of a test architecture or test configuration;
- a list of test specific pre-conditions that need to be met by the SUT including information about equipment configuration, i.e. precise description of the initial state of the SUT required for the execution of the test sequence;
- an ordered list of equipment operation and observations, i.e. an interoperability test sequence;
- in case of including communication assessment, a list of checks on the information exchanged between different EUTs; the level of detail of these checks depends on desired strength of conformance assessment - one extreme is to assess message field settings as specified in the standard.

4.5.2 Structure, style and documentation of test cases

As for any software development project, interoperability test suites should be specified using an agreed set of naming conventions that allow the identification of different testing constructs, enforce a proper use of modularization of test software [i.3], and be properly documented, e.g. using documentation tags [i.7]. In addition, it is recommended that interoperability test suite specification is based on test libraries [i.8]. Libraries can help to isolate common interoperability testing constructs and interface specific code so that the specification of test cases can be simplified. This also can significantly increase the readability, reusability, and maintainability of test software - especially across several test projects. An example application of these concepts can be found in [i.9]. Annex A provides an example specification of TTCN-3 interoperability testing library.

4.5.3 Test reporting

The test outcome of an interoperability test is usually a complex combination of information from multiple interfaces. An important aspect in automating interoperability testing is that test execution results are properly reported to parties which participate to the test or other parties such as the organizers of an interoperability testing event or independent evaluators of a test execution trace. A test report should provide more information than intermediate or final verdicts for each test.

In the context of standards validation, the quality of this information is important to provide proper feedback to the specification. During interoperability events, participants are generally eager to show that it was not their equipment that has caused the error which led to non-interoperability. Detailed test reports help to detect such errors.

NOTE: An error is not the cause of a failure but it leads to a failure. Finding the cause of a failure, i.e. the fault, is a non-trivial task that usually requires a detailed analysis of a test report.

For each test, especially for each failed or inconclusive test, a test report should include the following items and descriptions:

- the test step in the test description where the test case has failed;
- a description of the difference between the observed and the expected behaviour, i.e. the error or mismatch that leads to the test case failure;
- a reference to the section in a specification which contains the requirement that has been violated;
- the location or interface where the error was observed;
- the complete observation, e.g. received and expected messages.

EXAMPLE: In test TD_IMS_0015, test step 7, call flow message 40A the INVITE message sent by IMS Core Network to UE B the topmost Route header indicating the SIP URI of AS B and containing a Route header indicating the S-CSCF SIP URI of IUT is missing. The received and the expected messages are …

In order to be able to generate such detailed reports, two different approaches can be followed in the test system development process: The first approach leaves the responsibility to generate and attach reports to the verdicts to the test case developer. In the second approach, a report generator is implemented separately from the test system that generates reports by post-processing test execution logs from all interfaces.

The advantage of the first approach is that it provides more flexibility to find the location and origin of an error since the test case developer has a direct control of the code and the ability to transfer directly information into a report from any point in the code. In the second approach, an information loss may arise during the transformation of the reporting related information into a log format which becomes an intermediate format between the testing system and the log analyzer. However, an external log analyzer may have the ability to extract valuable relations between different parts of the logs. This is not possible at design time since all the information and logs are built during post-processing analysis.

4.6 Controllability of equipment with non standard interfaces

In practice, interoperability testing usually requires the operation of one or more equipment, e.g. an EUT or test equipment via non standardized interfaces. In a manual test execution, test operators operate this equipment in order to produce the behaviour required by the test.

The following scenarios need to be taken in consideration when trying to automate the control of equipment via non standardized interfaces:

- One or more equipment may need to be configured prior to an interoperability test execution. In some cases this configuration can be handled by the equipment operator once before executing the entire interoperability test suite or multiple tests. However, other cases may require a modification of equipment configuration(s) prior or during an interoperability test case execution.
- Frequently one or more equipment may need to be operated, i.e. stimulated and observed during a test case execution, e.g. in order to initiate a call from a phone, check that a phone is ringing, etc.

- As part of test configuration some test equipment or application support nodes, e.g. monitoring equipment, may need to be configured, e.g. to configure message filters on specific interfaces for a specific test.

5 Automated interoperability testing of distributed systems

5.1 A generic test framework for interoperability testing

Figure 2 depicts a generic framework for automated interoperability testing. In the following clauses, this framework and its entities are described.

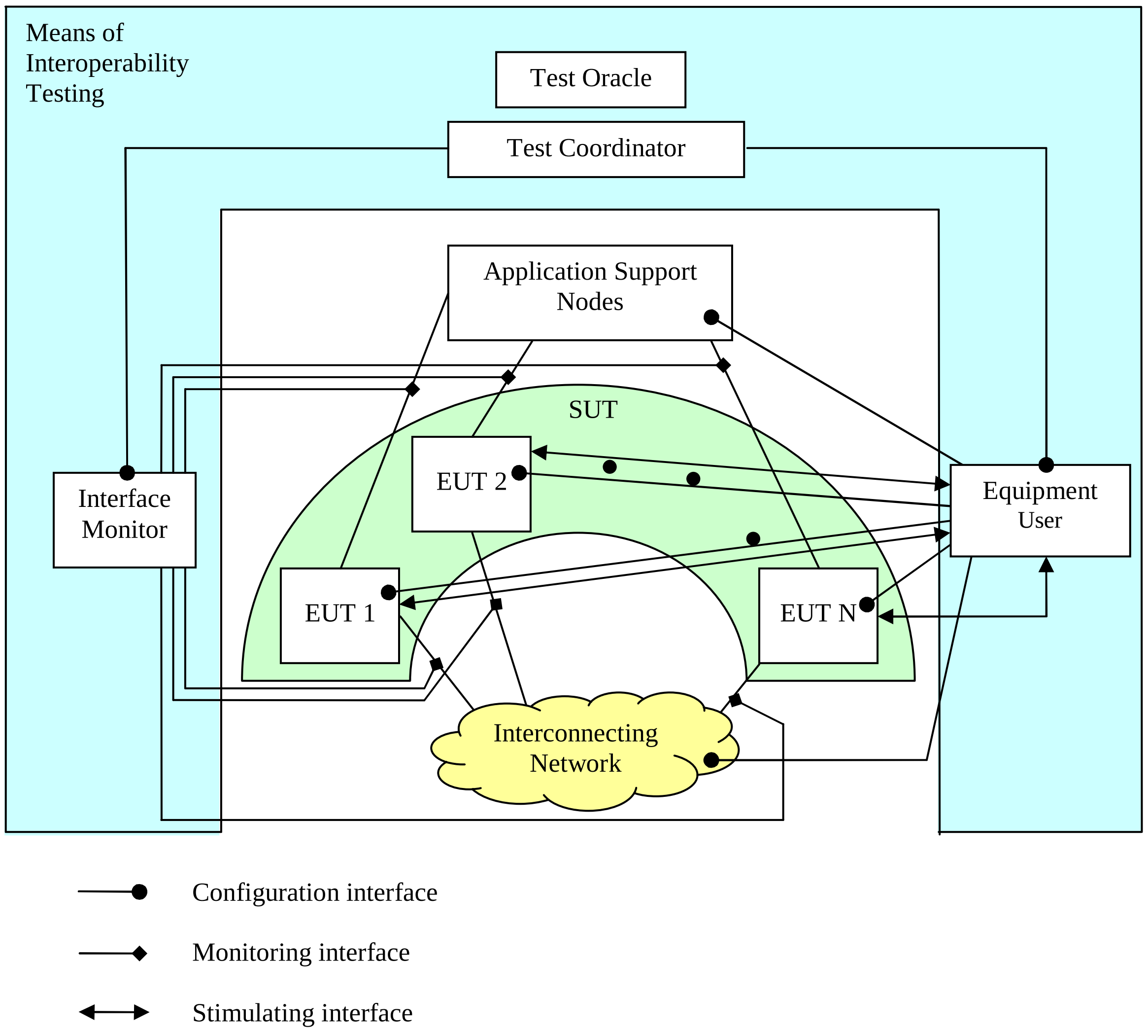

Figure 2: General framework for interoperability testing

5.1.1 System under test

In interoperability testing, the SUT constitutes two or more EUTs that are assessed for their interoperability. Each EUT is a group of one or more devices usually provided by the same vendor. The concrete device(s) included in an EUT depend(s) on the supplier's implementation as well as on the service or functionality needed.

5.1.2 Interconnecting network and application support nodes

Interconnecting network and application support nodes are entities essential for the interoperation of the tested equipment and are usually part of the service provided to the end user. They are, however, considered to be neither part of the SUT nor the means of testing. It is assumed that these devices have been previously tested and properly configured with an initial configuration. Their configuration can be changed during test execution.

Interconnecting network includes all the devices needed to provide the required network connections. At least it may be a simple wire. Application support nodes include all the devices involved in providing the service or functionality to the end user which are not object of the test, e.g. a data base with user data.

5.1.3 Means of interoperability testing

The means of interoperability testing includes logical entities that stimulate or control the SUT or configure the interconnecting network as well as the application support nodes. Different types of entities are:

- an equipment user entity that either acts like the end user of the service/functionality provided by an EUT or other equipment. In addition, it can configure an EUT or other equipment like the interconnecting network or application support nodes remotely;
- an interface monitor entity that listens to selected interfaces between EUTs, the interconnecting network and application support nodes. It checks relevant protocol procedures and messages;
- a test coordinator entity that coordinates the execution of all other entities of the means of testing which are assumed to be independent and parallel execution threads. It is in charge of controlling the overall execution, management of testing phases, and synchronization;
- a test oracle entity that collects information and manages E2E and conformance verdicts separately.

Depending on existing limitations as well as desired degree of automation, a specific instance of a means of interoperability testing may include only a subset of the described types of entities as well as multiple instances of each entity type. In addition, some tasks handled by different entities may be combined by a single execution thread, e.g. test coordinator and oracle.

5.2 A development process for an interoperability test system

The development of a specific interoperability test system is based on specific set of test descriptions, the generic interoperability testing framework as presented in the previous clause, the test architecture of the technology involved, and limitations which apply to the development of a specific test system. A typical development process for an interoperability test system is illustrated in figure 3. This process extends the activity "Write Test Cases" in the process for the general development of interoperability test specifications presented in [i.3]. Each step of the process is described in the following clauses.

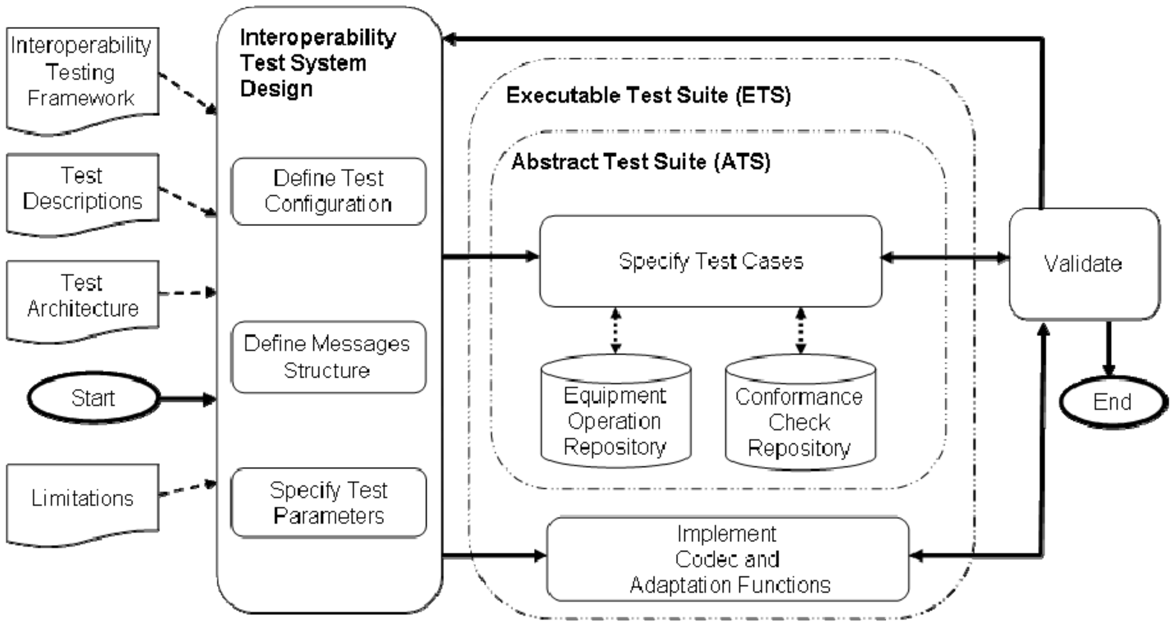
Interoperability Testing Framework
Test Descriptions
Test Architecture
Start
Limitations
Interoperability Test System Design
Define Test Configuration
Define Messages Structure
Specify Test Parameters
Executable Test Suite (ETS)
Abstract Test Suite (ATS)
Specify Test Cases
Equipment Operation Repository
Conformance Check Repository
Implement Codec and Adaptation Functions
Validate
End

Figure 3: The development process for an interoperability test system

5.2.1 Interoperability test system design

This phase leads to the definition of all structural information that is needed to define the test entities and their behaviour as well as their communication between themselves and the SUT.

5.2.1.1 Define test configuration

The first step in the design of a test system is the definition of the test configuration which is a concrete instance of the generic framework introduced in clause 5.1. Its definition is based on observations and analysis of the test architecture. This task consists of defining all test entity structure and their communication links. Instances derived from these structures may be used to act as one or more types of the test entities described in clause 5.1.3 and at the same time reproduce the behaviour specified in the test description. Such test entity instances can be used to configure or operate a specific EUT, to monitor relevant interfaces, or to configure application support nodes or the interconnecting network. It is recommended to dedicate one test entity instance per EUT or logical interface to be monitored. This increases reusability of test behaviour across a test suite and makes test case specification less dependent on the availability of interfaces. Annex B describes two different approaches for the definition of test entity instances for monitoring interfaces.

In addition to equipment user and interface monitor test entities, one separate entity should be dedicated to act as the test coordinator and test oracle. It should create and supervise the execution of all other test entity instances running in a test case to synchronize the instances on identified synchronization points, and to manage and resolve E2E as well as conformance verdicts.

When defining the test entity instances, the designer should pay attention to the logical interface topology of the technology to be tested as well as its configuration. For example, one logical interface may map to multiple IP connections in a SUT. Similarly, communication channels associated with one logical interface may be distributed via different physical carriers.

5.2.1.2 Define message structure

The second step in interoperability test system design consists of the definition of messages structures for all test interfaces based on their respective protocol specification. This step is not necessary in the case that protocol specifications standardize the abstract syntax of messages, e.g. using ASN.1.

Proprietary interfaces should be defined as an abstract set of simple commands with parameters that can be mapped to different proprietary interfaces. An example for such a command could be "Initiate a VoIP call" with a parameter "user identifier". Command definitions should enable a unique association with a specific equipment operation but at the same time be independent of a specific realization. For example, the previous command should not be defined as "Press the green button on the phone" which would restrict the action of making a call to have a keypad available on a terminal.

5.2.1.3 Specify test parameters

The ability to parameterize a test case specification is a basic requirement to ensure its adaptability to different execution environments or SUTs without modification of executable code. Parameterization of test cases needs to be addressed at several levels:

- **EUT information parameters** allow to update test cases EUT information like identifiers, addresses, and port numbers quickly for equipment from different vendors;
- **message parameters** allow to adapt test cases to specific needs of a testing environment or EUT, e.g. specific user identities to be used, a specific identifier to be used for a non-existing user;
- **timing parameters** allow to modify timeouts for no activity or other timing like call duration in test cases;
- **interface availability parameters** allow to adapt test cases quickly to the lack of automatic accessibility of certain EUT interfaces prior to the execution of a test;
- **test session pairing parameters** allow to keep test case specifications independent of specific EUT configuration information;
- **test execution parameters** allow a customized selection of test cases to be executed as part of a test suite, e.g. to skip tests which are not applicable due to the lack of support of a given functionality by one of the EUTs. They can also be used to run a test suite in different modes or configurations, e.g. live versus offline mode.

5.2.2 Specify test cases

A test case is a sequence of elementary actions and checks being executed on different test entities. Some of this actions and checks can be grouped. Typically these groups of elementary actions and checks are called test steps. Similarly to subroutines and functions in conventional programming languages, it is useful to collect test steps in a test step repository and reuse them across a test suite. Test step definitions should be parameterized to improve their potential for reuse.

For the specification of test steps the following aspects should be taken into consideration:

- selection of sequence of actions and checks to be isolated;
- specification message contents to be sent and to expect;
- guarding against deadlock or no activity with timers;
- handling of exceptions and other unexpected test events;
- assignment of applicable test verdict, i.e. E2E or conformance.

For each test step a state machine needs to be defined. The detail of this state machine specification, in particular for the management of the exception and faults, depends on the accuracy decided according to the testing objectives and constraints.

Which actions and checks to combine in a test step sequence is mainly determined from testing experience. Test descriptions however are a good source for the identification of such test steps since they also essentially specify sequences of abstract actions and checks. The design of test steps based on test descriptions and conformance test purposes has also other benefits:

- alignment of terminology and identifier naming of test case specification and test description;
- easier maintenance of test steps in case of test description changes;

- improved readability of interoperability test execution traces.

Each test case specification can be split into the following phases:

- specification of the preamble that is the preliminary set of steps required to advance the SUT to the state described in the pre-test conditions of a TD, e.g. to configure the SUT and the other network elements involved in the test, register users, etc;
- specification of the sequence of steps, i.e. actions and checks for each test entity instance needed to achieve the interoperability TP;
- specification of the postamble that is the set of steps to return the SUT and the other network elements to their initial state.

The specification of test coordinator behaviour should include the creation and management of test entities required for a specific test as well as the management of applicable overall test verdicts, i.e. E2E and conformances based on the execution of other test entities.

In order to ensure proper handling of abnormal test termination, the following issues have to be considered during test case specification:

- a test case should release the test system resources (e.g. interfaces, users, memory) prior to its termination. It should use general purpose functions to manage unpredictable events and it should stop a test case execution within a specified time limit in case the test system receives no input during a test;
- a test case should attempt to return the SUT to its initial configuration if possible before executing the next test case to minimize the impact of the abnormal test termination on the following test case executions.

In interoperability testing two types of test step repositories can be derived:

- the equipment operation repository containing all actions required to configure or operate a specific EUT, or to configure application support nodes or the interconnecting network;
- the conformance check repository containing the conformance checks applied to the monitored interfaces.

Test case specification completes the Abstract Test Suite (ATS) specification in test system development. Tests are written in a formal manner but are still not executable. In the final step, it is needed to implement codec and adaptation functions in order to complete the executable test system.

5.2.3 Implement codec and adaptation function

To complete interoperability test system development, the implementation of codec and adaptation functions is required. These functions provide a number of services necessary for interoperability test execution:

- conversion of messages exchanged with the SUT from abstract to transfer protocol syntax and vice versa;
- transport of encoded messages via test interfaces to the relevant EUT in the SUT and vice versa;
- communicating information between the test entities with other test equipment, application support nodes and the interconnecting network, e.g. protocol analyzers, jamming equipment, etc;
- filtering of message traffic related to a specific logical monitored interface from traffic capture;
- map abstract operation of abstract EUT interfaces to concrete, proprietary interfaces , e.g. for configuring EUTs;
- initiate and control test case execution;
- handling a meaningful presentation of test execution results to test operators.

Through compilation and linking the ATS is put together with these functions and an executable code Executable Test Suite (ETS) is created which can be executed against the SUT.

5.2.4 Validation

During the validation step, the test system is connected to a SUT. If a SUT is not available, the test system can be tested in a back to back configuration or with a SUT emulator. The purpose of this phase is to assure that the test system reproduces at its interface faithfully the behaviour specified for each test description.

Validation realizes a feedback loop to the previous steps and should be used to correct the test steps library, test cases or adaptation functions.

Annex A: TTCN-3 library for interoperability testing

The TTCN-3 core (text) representation corresponding to this library is contained in the ASCII files (LibIot_TestConfiguration.**ttcn**, LibIot_TestInterface.**ttcn**, LibIot_TypesAndValues.**ttcn**, LibIot_Functions.**ttcn**, LibIot_PIXITS.**ttcn** contained in archive eg202810v010101p0.**zip**) which accompanies the present document

Annex B: Interface versus entity based approach for realizing the interoperability testing framework

So far two different approaches have been identified for using interface monitor entities within the general test framework for interoperability testing as introduced in clause 5.1:

- An interface approach where each interface monitor test entity is directly associated with an interface or connection between two EUTs; this approach more straight forward to comprehend from the architecture point of view and more flexible when it comes to dealing with (unexpected lack of) EUT interface availability.

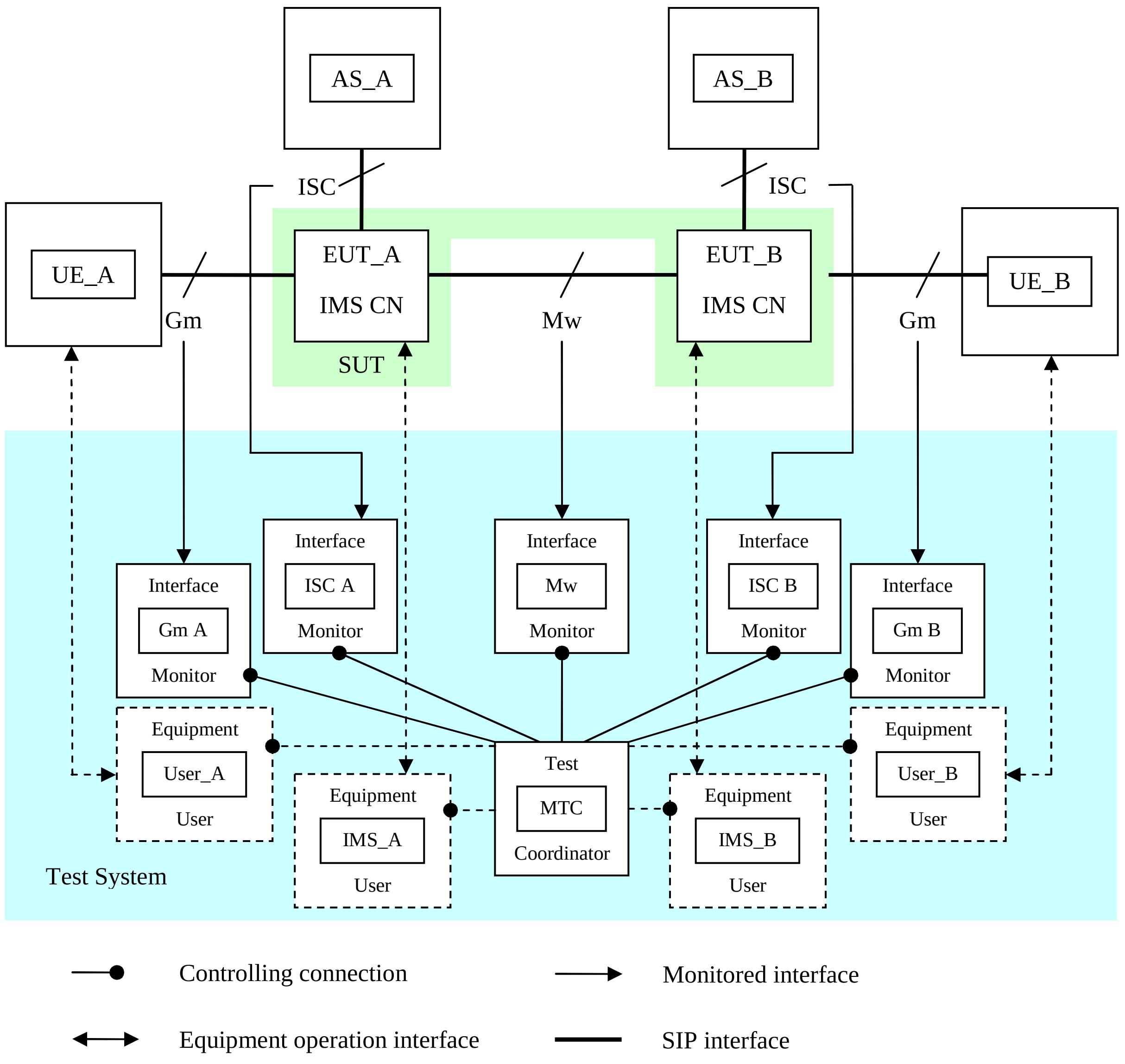

Figure B.1: Illustration of interface based approach based on IMS interoperability testing

- An entity approach where the monitored interfaces are grouped into higher level testing entities mirroring all EUTs, i.e. the complete architecture of the SUT. This approach allows mirroring message exchanges between EUTs as part of the test execution trace, i.e. as part of the test execution the message exchange specified in test descriptions is produced.

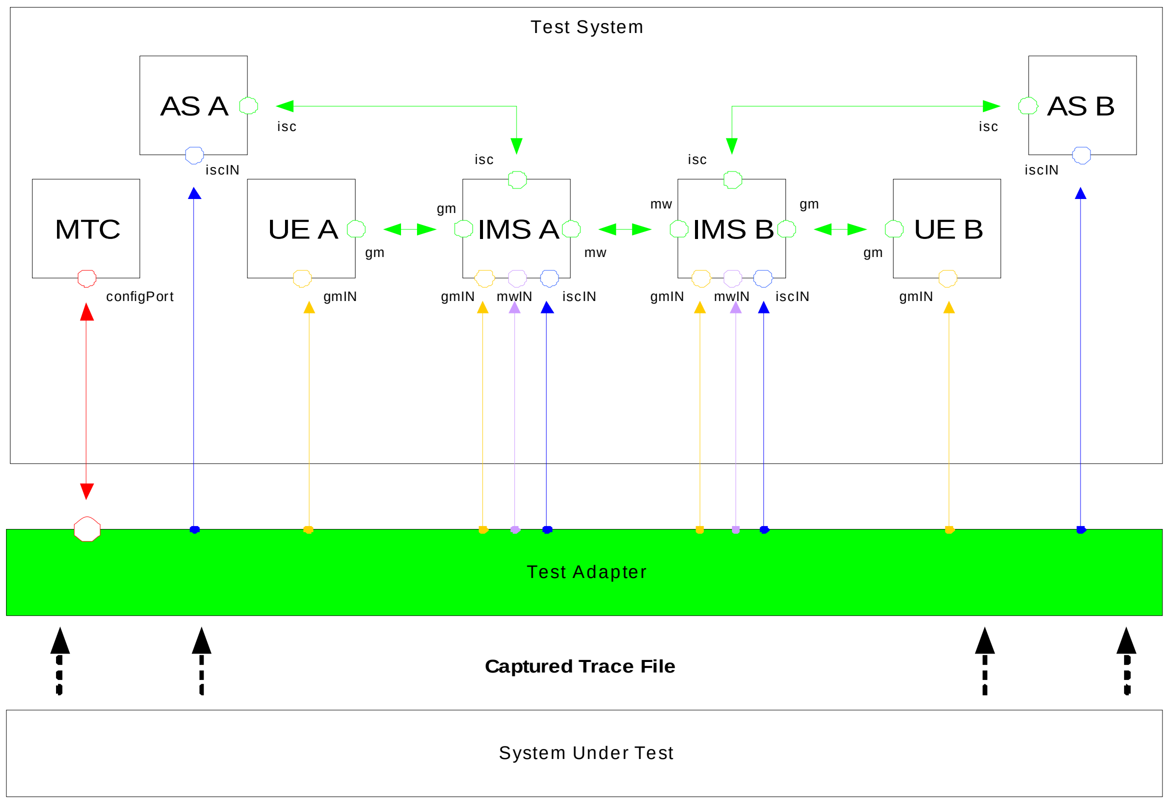

Figure B.2: Illustration of entity based approach based on IMS interoperability testing

Annex C: Bibliography

- ISO/IEC 7948-1: "Information technology - Open Systems Interconnection - Basic Reference Model: The Basic Model".

History

Document history		
V1.1.1	January 2010	Membership Approval Procedure MV 20100326: 2010-01-25 to 2010-03-26
V1.1.1	March 2010	Publication