

7. Test methods

7.1 Introduction

The testing of a given OSI* protocol can require the use of several test methods, as systems under test can come in several configurations, and vary in terms of their ability to allow ways of producing effects applicable to a layer boundary.

This section first characterizes the features of the system under test which are to be taken into consideration, next defines the possible test methods in abstract terms, and finally provides guidance on their applicability to real systems.

7.2 Classification of real open systems and IUTs for conformance testing

7.2.1 Classification of systems under test

7.2.1.1 There is a relation between the test methods and the configurations of the real open systems to be tested. The appropriate test methods vary according to:

- a) the main function of the system (end-system or relay-system);
- b) which layers use OSI* protocols;
- c) whether the alternative of non-OSI* protocols is also available.

7.2.1.2 The following configurations of systems have been identified for the purposes of conformance testing, as illustrated in Figures 2 to 4. Configurations 1 to 3 are the basic configurations of systems under test (SUTs):

- a) Configuration 1: 7-layer open system (end-system)

These systems use OSI* Recommendation* protocols in all 7 layers.

- b) Configuration 2: Partial (N)-open system (end-system)

These systems use OSI* Recommendation* protocols in layers 1 to N.

- c) Configuration 3: Open relay-systems

These use OSI* protocols in layers 1 to 3 (Network relay-systems) or 1 to 7 (Application relay-systems).

7.2.1.3 Other configurations can be derived from the basic configurations.

A SUT can be a combination of basic configurations 1 and 2, allowing the alternative of using OSI* and non-OSI* protocols above layer N (see Figure 5).

7.2.2 Identification of the implementation under test (IUT)

An implementation under test (IUT) is that part of a real open system which is to be studied by conformance testing. It should be an implementation of one or more adjacent OSI* protocols.

IUTs can be defined for configurations 1 and 2 of SUTs as single-layer IUTs (one single layer of the SUT is to be tested), or as multi-layer IUTs (a set of any number of adjacent layers of the SUT to be tested in combination).

An IUT defined in an open relay-system will include at least the layer which provides the relay function.

When OSI* and non-OSI* protocols exist in a system, the IUT(s) will be defined for the OSI* mode(s) of operation. Testing non-OSI* protocols is outside the scope of this Recommendation.

Clients and test laboratories will agree on what part of the SUT will be considered to be the IUT.

7.3 Abstract testing methodology - general

Test methods need to refer to an abstract testing methodology, based upon the OSI reference model. Considering first end-systems (7-layer or partial (N)-open systems) and single layer IUTs within these systems, abstract test methods are described in terms of what outputs from the IUT are observed and what inputs to it can be controlled. More specifically, an abstract test method is described by identifying the points closest to the IUT at which control and observation are to be exercised.

The OSI* protocol Recommendations* define allowed behaviour of a protocol entity (i.e., the dynamic conformance requirements) in terms of the protocol data units (PDUs) and the abstract service primitives (ASPs) both above and below that entity. Thus the behaviour of an (N)-entity is defined in terms of the (N)-ASPs and (N-1)-ASPs (the latter including the (N)-PDUs).

If an IUT comprises more than one protocol entity, the required behaviour can be defined in terms of the ASPs above and below the IUT, including the PDUs of the protocols in the IUT.

The starting point for developing test methods is the conceptual testing architecture, illustrated in Figure 6. It is a "black-box" active testing architecture, based on the definition of behaviour required of the IUT.

The action of the tester shown in Figure 6 can either be applied locally, in which case there is a direct coupling within the system under test, or externally via a link or network. The two sets of interactions, above and below the IUT, can, in practice, be observed and controlled from several different points, locally or externally.

The possible points of control and observation (PCOs) are identified by three factors:

- a) whether it is the ASPs or PDUs which are observed and controlled;
- b) the layer identity of the ASPs or PDUs concerned;
- c) whether they are controlled and observed within the system under test or in a system remote from the system under test - if the latter then the ASPs are distinguished by the addition of a double-quote character(").

Possible PCOs within the SUT are illustrated in Figure 7(a). Possible PCOs, when the activity below the IUT is controlled and observed externally are illustrated in Figure 7(b). It can be seen from these figures that there is a multiplicity of possible PCOs in different layers, which offer different degrees of control and observation of IUT behaviour. This Recommendation makes a selection from this set of possible PCOs, defining a limited number of abstract test methods.

If control and observation below, or external from, the IUT is specified in terms of ASPs, it will include control and observation of the PDUs carried by those ASPs; but if it is specified in terms of PDUs (at layer N) then the ASPs (at layer N-1) are not considered to be controlled or observed.

It is assumed that the ASP activity below the IUT can at least be observable and controllable via the peer activity in a remote testing system - i.e., the corresponding ASP"s. Thus when the ASPs below the IUT are not controllable nor observable locally, conformance testing can be carried out externally, provided that the underlying service offered is sufficiently reliable for control and observation to take place remotely.

It is possible that the ASP activity above the IUT might not be controllable nor observable, in which case this activity is said to be hidden.

SUTs are not required to provide access to layer boundaries. However, the possible provision of such access and the possible positions of such boundaries with respect to the layers of the IUT are factors to be taken into consideration in the definition of the test methods, which may take advantage of this access to define the test suites in terms of the corresponding ASPs. It does not matter whether the accessible boundaries are accessed via service access points (SAPs) or via some other PCOs.

Figure 8 shows examples of IUTs, with respect to layer boundary accessibility.

Note - In addition, a conformance test suite Recommendation* may define "abstract local primitives". These are used to specify control and observation of events or states which are referred to in the protocol Recommendation* but which are internal to the IUT and which cannot be expressed in terms of ASPs. They are abbreviations for text descriptions of control and observation to be performed by the upper tester.

Similar consideration apply to relay systems (see Part 2 of the Recommendation for details).

FIGURE 8/X.290, PART 1

Examples of IUT configurations

7.4 Abstract testing functions

The definition of abstract test methods requires that the PCOs be distributed over two abstract testing functions, the lower and upper testers.

The lower tester is the abstraction of the means of providing, during test execution, control and observation at the appropriate PCO either below the IUT or remote from the IUT, as defined by the chosen abstract test method. Thus, it is the testing function related to the control and observation of the lower boundary of the IUT. If the action of the tester is local to the SUT, the lower tester will take the place of the lower part of the SUT. If the action of the tester is external to the SUT, the lower tester will rely on the (N-1)-Service, provided jointly by the lower tester itself, a link and the SUT.

The upper tester is the abstraction of the means of providing, during test execution, control and observation of the upper service boundary of the IUT and of any relevant abstract local primitive.

There is a need for cooperation between the upper tester and the lower tester; the rules for such cooperation are called the test coordination procedures.

The test methods will vary in the way that they specify the test coordination procedures. In some cases it is possible to define a test management protocol to provide the coordination between the upper and lower testers. In other cases it is only possible to describe the requirements to be met by the test coordination procedures without specifying what mechanisms might be used to realize them.

7.5 Overview of abstract test methods

7.5.1 End-system IUTs

For the IUTs defined within end-system SUTs (configurations 1 and 2 in Figures 2 and 3) four categories of abstract test methods are defined, one local, and three external ones which assume the lower tester is located remotely from the SUT and connected to it by a link or network.

7.5.2 The local test methods

The local abstract test methods define the PCOs as being at the service boundaries above and below the IUT. The test events are specified in terms of the ASPs above the IUT and the ASPs and PDUs below the IUT, as illustrated in Figure 9(a). Abstractly, a lower tester is considered to observe and control the ASPs and PDUs below the IUT, while an upper tester observes and controls the ASPs above the IUT. Requirements to be met by test coordination procedures used to coordinate the realizations of the upper and lower testers are defined in the abstract conformance test suites, although the test coordination procedures themselves are not.

7.5.3 External test methods

The external test methods use control and observation of the ASPs below the IUT by means of a lower tester separated from the SUT, together with control and observation of the ASPs above the IUT. Three different categories of external abstract test methods are defined, which are referred to as distributed, coordinated, and remote. They vary according to the level of requirement or standardization put on the test coordination procedures, on the access to the layer boundary above the IUT, and on the requirements on an upper tester. They are illustrated in Figures 9(b), (c) and (d).

The coordinated test method requires that the test coordination procedures used to coordinate the realization of the upper and lower testers be achieved by means of test management protocols. The other two methods do not make any assumptions about the realization of the test coordination procedures.

The distributed and coordinated test methods require specific functions from the upper tester above the IUT. The remote method does not.

The distributed method requires access to the upper boundary of the IUT. The other two methods do not.

7.5.4 Variants of end-system test methods

Each category of test methods has variants which can be applied to single-layer IUTs or to multi-layer IUTs. For multi-layer IUTs in which the protocols are to be tested layer by layer, embedded variants can be used.

All abstract test methods for end-systems are fully specified in section 8 of Part 2 of this Recommendation, including their single-layer, multi-layer and embedded variants where applicable.

7.5.5 Relay-system IUTs

For open relay-systems, two test methods are defined, loop-back and transverse. These are fully specified in section 8 of Part 2 of this Recommendation. FIGURE 9/X.290, PART 1

Overview of abstract test methods

7.6 Applicability of test methods to real open systems

The architecture and stage of development of a real open system determines the applicability of test methods to it.

Local test methods are useful for systems under development, when their architecture permits the isolation of an IUT, be it single-layer or multi-layer.

External test methods are useful for testing complete or partial end-systems which can attach to telecommunications networks.

Coordinated test methods apply where it is possible to implement a standardized test management protocol in an upper tester in the SUT, above the IUT.

Remote test methods apply when it is possible to make use of some functions of the SUT to control the IUT during testing, instead of using a specific upper tester.

Distributed test methods apply when it is necessary to allow complete freedom for the implementation of the test coordination procedures between the SUT and the lower tester, but to specify in detail the control and observation requirements at both ends.

Single-layer test methods are the most appropriate methods for testing the majority of the protocol conformance requirements.

Multi-layer test methods will be used when conformance to true multi-layer dynamic conformance requirements is to be tested.

Embedded test methods permit the application of single-layer testing to all layers of a multi-layer IUT.

For 7-layer open systems, the preferred methods are the incremental use of appropriate external single-layer embedded methods with the following PCOs:

- a) the upper interface of the application layer as provided by the 7-layer open system, when applicable;
- b) successively, each SAP (or corresponding PCO if there is no SAP as such) below the protocol which is the focus of the testing, as controlled and observed in the external lower tester, starting from the lowest protocol of the IUT and working upwards.

7.7 Applicability of the test methods to OSI* protocols and layers

The test methods defined in this Recommendation apply to all layers, with the exception of the Physical-layer and the Media Access Control protocols which are outside the scope of this Recommendation. Appendix I of this Recommendation provides guidance on the applicability of test methods to all other layers.

8. Test suites

8.1 Structure

Test suites have a hierarchical structure (see Figure 10) in which an important level is the test case. Each test case has a narrowly defined purpose, such as that of verifying that the IUT has a certain required capability (e.g. the ability to support certain packet sizes) or exhibit a certain required behaviour (e.g. behave as required when a particular event occurs in a particular state).

Within a test suite, nested test groups are used to provide a logical ordering of the test cases. Test groups may be nested to an arbitrary depth. They may be used to aid planning, development, understanding or execution of the test suite.

Test cases are modularized by using named subdivisions called test steps. Each test case comprises at least one test step: the ordering of events covered by the test purpose ("test body"). It may include further test steps to put the IUT into the state required for the test body to start from (the "preamble") or return to a quiescent state after the test body has finished (the "postamble").

For practical reasons, common test steps may be grouped together into test step libraries. Test step libraries may be structured into nested sets of test steps, to any depth of nesting. Test step libraries may be associated with the whole test suite or with a particular test group or test case.

Furthermore, all test steps consist of an ordering of other test steps and/or test events (e.g. the transfer of a single PDU or ASP to or from the IUT). All test steps are, therefore, equivalent to an ordering of test events (after expansion of the inner test steps).

FIGURE 10/X.290, PART 1

Test suite structure

8.2 Generic, abstract and executable test cases

8.2.1 A generic test case is one which:

- a) provides a refinement of the test purpose;
- b) specifies that all sequences of test events (paths) in the test body which correspond to verdicts of "pass", "fail" and "inconclusive", using a specialized notation;
- c) is used as the common root of corresponding abstract test cases for different abstract test suites for the same protocol;
- d) includes a description of the initial state in which the test body should start, in lieu of a preamble;
- e) need not describe the postamble;
- f) is specified using the style of either the Remote or Distributed Single-layer test methods.

8.2.2 An abstract test case is derived from a generic case and the relevant protocol specification; it:

- a) specifies the test case in terms of a particular test method;
- b) adds a more precise specification for sequences of events which are only described informally in the generic test case;
- c) adds the sequences of test events required to achieve the objectives of the preamble and postamble of the generic test case using a specialized notation.

8.2.3 An executable test case is derived from an abstract test case, and is in a form which allows it to be run on a real tester for testing a real implementation.

8.2.4 The terms generic, abstract and executable are used to describe test suites, which comprise generic, abstract and executable test cases, respectively.

8.2.5 A generic test suite has the coverage of the set or a superset of all possible abstract test suites for a particular protocol.

9. Relationships between concepts and roles

Figure 11 is a pictorial representation of the relation between the various Recommendations and the processes of producing generic, abstract and executable test suites and test reports.

Part 2 concerns the production of testable protocol Recommendations and abstract test suite Recommendations. Part 1 provides general concepts and definitions.

Note - Other aspects of the conformance assessment process, such as executable test derivation, preparation of the IUT, PICS and PIXIT by the client and test laboratory role are for further study.

10. Compliance

In this Recommendation, "compliance" refers to meeting the requirements specified by the Recommendation. This word is used as an attempt to eliminate confusion between compliance to this Recommendation and conformance of a protocol implementation to protocol Recommendations.

This part of the Recommendation contains no compliance requirements.

FIGURE 11/X.290, PART 1

Relationships between concepts and roles