

PART 2: ABSTRACT TEST SUITE SPECIFICATION

Introduction

This part of the Recommendation provides a common approach to the specification of "OSI or related CCITT X-Series or T-Series" (hereafter abbreviated to "OSI*") conformance test suites at a level which is independent of the means of executing those test suites (hereafter called "abstract test suites"). This level of abstraction is suitable for standardization and facilitates the comparison of results produced by different organizations which run the corresponding executable test suites.

Section 1 recalls that there are requirements put on the OSI* protocol specifiers which have to be fulfilled before there can be an objective basis for the process of developing an abstract test suite. The need for consistent conformance sections and for PICS and PIXIT proformas in OSI* protocol Recommendations* is expressed.

Section 2 describes the process of developing an abstract test suite, including the design criteria to be used and guidance on its structure and coverage. The possible abstract test methods are defined and guidance is given to help the test suite specifier to decide which method(s) to use in the production of a particular test suite. The relationship between abstract test suites for different methods is provided by a generic test suite which is independent of the test method. A test notation is defined and requirements and guidance are given on its use for specifying both generic and abstract test cases. These include the subdivision of test cases into test steps and the assignment of verdicts to outcomes.

The test suite specifier is also required to provide information to the test realizers, particularly on the constraints governing test case selection and ordering.

Finally, guidance and requirements are given on test suite maintenance.

1. Scope and field of application

This part of the Recommendation specifies the requirements and provides guidance for the production of system-independent conformance test suites for one or more OSI* Recommendations*. In particular, it applies to the production of all conformance test suite Recommendations* for OSI* protocols.

It applies to the production of conformance test cases which check the adherence of an implementation to the relevant static and/or dynamic conformance requirements by controlling and observing protocol behaviour. The abstract test methods included in this Recommendation are in fact capable of being used to specify any test case which can be expressed abstractly in terms of control and observation of protocol data units, abstract service primitives and abstract local primitives. Nevertheless, for some protocols, test cases may be needed which cannot be expressed in these terms. The specification of such test cases is outside the scope of this Recommendation, although those test cases may need to be included in a Recommendation* for a conformance test suite.

Note - For example, some static conformance requirements related to an application service may require testing techniques which are specific to that particular application.

The production of conformance test suites for multi-peer or physical layer protocols is outside the scope of this Recommendation.

The relation between abstract test specification and formal description techniques is outside the scope of this Recommendation.

2. References

Recommendation X.200 - Reference model of open systems interconnection for CCITT applications. (See also ISO 7498.)

Recommendation X.214 - Transport service definition for open systems interconnection for CCITT applications. (See also ISO 8072.)

Recommendation X.224 - Transport protocol specification for open systems interconnection for CCITT applications. (See also ISO 8073.)

Recommendation X.210 - Open systems interconnection layer service definition conventions. (See also ISO TR 8509.)

Recommendation X.208 - Specification of Abstract Syntax Notation One (ASN.1). (See also ISO 8824.)

Recommendation X.209 - Specification of basic encoding rules for Abstract Syntax Notation One (ASN.1). (See also ISO 8825.)

Recommendation X.290/1 - OSI conformance testing methodology and framework for protocol Recommendations for CCITT applications - Part 1: General concepts.

ISO 8571-4 - Information processing systems - Open Systems Interconnection - File protocol specification.

3. Definitions

For the purposes of this part of the Recommendation, all the definitions in Part 1 of the Recommendation apply.

4. Abbreviations

For the purposes of this Recommendation the abbreviations given in section 4 of Part 1 of this Recommendation apply. The following abbreviations also apply to this Recommendation:

ASP : the abstract service primitive on the side of the service provider remote from the IUT

CM : coordinated multi-layer (test method)

CS : coordinated single-layer (test method)

CSE : coordinated single-layer embedded (test method)

DM : distributed multi-layer (test method)

DS : distributed single-layer (test method)

DSE : distributed single-layer embedded (test method)

FDT : formal description technique

LM : local multi-layer (test method)

LS : local single-layer (test method)

LSE : local single-layer embedded (test method)

RM : remote multi-layer (test method)

RS : remote single-layer (test method)

RSE : remote single-layer embedded (test method)

TTCN: tree and tabular combined notation

YL : loop-back (test method)

YT : transverse (test method).

5. Compliance

5.1 A protocol Recommendation* which complies with this part of this Recommendation shall satisfy all the requirements stated in section 1.

Note - Such compliance is a precondition for the protocol Recommendation* to be an effective basis for conformance testing of implementations.

5.2 An abstract test suite specification which complies with this part of this Recommendation

- a) shall be a conformance test suite;
- b) shall be specified in a test notation standardized by ISO or CCITT;
- c) shall satisfy all the requirements stated in section 2.

5.3 It is recommended that the test notation used be the Tree and Tabular Combined Notation (TTCN). If TTCN is used, the abstract test suite shall comply with all the requirements in Annex D.

Section 1 - Requirements on protocol specifiers

6. Conformance requirements in OSI* Recommendations*

6.1 Introduction

The meaning of conformance in OSI* is discussed in Part 1 of this Recommendation. It is necessary that there be an unambiguous and objective understanding of the conformance requirements of an OSI* protocol or transfer syntax Recommendation*, as a prerequisite to the production of an abstract test suite for that Recommendation*. This section states the requirements on protocol specifiers to ensure that there is such an understanding of the conformance requirements.

6.2 General requirements

6.2.1 A clear distinction shall be made between static and dynamic conformance requirements. To avoid ambiguity, they should be stated separately from one another.

6.2.2 It shall be clear what conformance to the Recommendation* means, in the sense of what shall be done, what is permitted but not mandatory, and what shall not be implemented, to conform to the Recommendation*.

6.2.3 It shall always be decidable whether an instance of communication conforms dynamically or not.

For example, one should be able to look at a record of PDU activity and decide whether it is valid.

6.2.4 Requirements on the need to produce and content of a PICS shall be stated separately from the requirements on the protocol implementation itself.

6.3 Conformance sections

6.3.1 Each OSI* protocol and transfer syntax Recommendation* shall include a conformance section, which shall be expressed clearly and unambiguously.

6.3.2 Conformance sections shall distinguish between the following categories of information that they may contain:

- a) references to sections which state dynamic conformance requirements;
- b) static conformance requirements concerning the protocol implementation;
- c) static conformance requirements concerning multi-layer dependencies;
- d) what has to be stated in the PICS concerning (b) above;
- e) what has to be stated in the PICS concerning (c) above;
- f) what other information shall be provided (e.g. to assist testing) and whether this should be in the PICS or elsewhere.

6.3.3 In connection-oriented protocol Recommendations*, the conformance section should also include:

- a) the option to support either the initiation of a connection or the acceptance of a connection, or both;
- b) the requirement to be able to accept all correct sequences of PDUs received from peers, and respond with correct PDU sequences appropriate to the defined state of the connection;
- c) the requirement to be able to respond correctly to all incorrect sequences of PDUs received, the response being appropriate to the defined state of the connection.

6.4 Additional guidance for new protocol Recommendations*

It is recognized that although existing protocol Recommendations* can be improved by the progression of an addendum to add a PICS proforma and make the conformance section align with the requirements stated above, it is unrealistic to expect any greater improvement. However, new protocol Recommendations* should be developed using the additional guidance given in Annex B to make the task of understanding the conformance requirements easier and less prone to ambiguity.

7. PICS proformas

7.1 Requirements on PICS proformas

7.1.1 The specific requirements to be met by suppliers in respect of each PICS they provide shall normally be stated in the relevant protocol Recommendation*. The specification of these requirements shall include a PICS proforma. In exceptional circumstances, the PICS proforma may be found in the abstract test suite Recommendation* rather than in protocol Recommendation*; in particular, this applies when the PICS proforma has to cover different versions of the same protocol coming from both ISO and CCITT. In normal circumstances, the PICS proforma should be found in an annex to the protocol Recommendation* and referenced in the conformance section, and, if necessary, progressed as an addendum rather than as part of the original Recommendation*.

Note - A PICS for a specific protocol implementation will need to be accompanied by administrative and documentary information relating to the supplier, the system and its intended environment.

7.1.2 The PICS proforma shall be in the form of a questionnaire or checklist to be completed by the supplier or implementor of an implementation of the relevant OSI* protocol.

7.1.3 The PICS proforma shall cover all functions, elements of procedure, parameters, options, PDUs, timers, multi-layer dependencies and other capabilities identified in the protocol Recommendation*, including optional and conditional capabilities. It is desirable, where practicable, to include all mandatory features. There shall be a well-defined mapping between static conformance requirements and the PICS proforma and there shall be consistency between them.

7.1.4 The PICS proforma shall be preceded by a section that states:

"The supplier of a protocol implementation which is claimed to conform to this Recommendation shall complete the following Protocol Implementation Conformance Statements (PICS) proforma and shall provide the information necessary to identify uniquely both the supplier and the implementation."

7.1.5 The name, version and date of the relevant protocol Recommendation* shall be stated on each page of the PICS proforma.

7.1.6 The PICS proforma for a specific protocol shall contain:

- a) explanations of special symbols, abbreviations, special terms, together with appropriate references;
- b) explicit instructions for completing and interpreting the PICS;
- c) provision for mentioning any mandatory feature that has not been implemented, with the appropriate rationale;
- d) one or more tables (or other kinds of form as necessary) to be completed to state the capabilities of the implementation in detail, including:
 - 1) name of the feature, PDU type, timer, parameter, and other capabilities;
 - 2) a column stating whether each is mandatory, optional, negotiable, or conditional;
 - 3) wherever feasible, a column giving references to the relevant sections in the Recommendation;
 - 4) a column giving the permitted range or values, if appropriate;

- 5) a column to be filled in to give the supported values or range of values, if appropriate;
- 6) a column to be filled in to state if each capability has been implemented;
- e) the proforma shall give a clear indication of the preferred data types (e.g. number bases, string types, octets, bits, seconds, minutes, etc.) for responses.

7.1.7 The PICS proforma shall use the following abbreviations as appropriate, unless they conflict with the abbreviations used in the specific protocol Recommendation*:

m: mandatory

o: optional

c: conditional

n: negotiable (using the protocol)

x: exclusion of capability

-: not applicable

s: ability to send

r: ability to receive

7.2 Guidance on PICS proformas

Appendix III provides some general purpose examples to give guidance on the construction of PICS proformas.

Section 2 - Requirements on abstract test suite specifiers

8. Test suite production process

In order to present the requirements and general guidance for abstract test suite specification, it is useful to assume a normal form of the process of test suite production. This section describes the process in just such a normal form. Abstract test suite specifiers are not required to follow this normal form exactly, however, they are recommended to use a similar process involving the same steps, possibly in a different order.

For the purposes of this Recommendation, the abstract test suite production process is assumed to be as follows:

- a) study the relevant Recommendation(s)* to determine what the conformance requirements (including options) are which need to be tested, and what needs to be stated in the PICS (see section 9);
- b) decide on the test groups which will be needed to achieve the appropriate coverage of the conformance requirements and refine the test groups into sets of test purposes (see section 10);
- c) specify generic test cases for each test purpose, using some appropriate test notation (see

section 11);

- d) choose the test method(s) for which the complete abstract test cases need to be specified, and decide what restrictions need to be placed on the capabilities of the lower tester and (if appropriate to the chosen test method(s)) the upper tester and test coordination procedures (see section 12);
- e) choose the test notation for specifying the complete abstract test cases, and specify the complete abstract test cases, including the test step structure to be used (see section 13);
- f) specify the inter-relationships among the test cases and those between the test cases and the PICS and, as far as possible, the PIXIT, in order to determine the restrictions on the selection and parameterization of test cases for execution, and on the order in which they can be executed (see section 14);
- g) consider the procedures for maintaining the abstract test suite (see section 15).

The remainder of this section provides requirements and guidance which relate to each step in the above process.

9. Determining the conformance requirements and PICS

9.1 Introduction

Before an abstract test suite can be specified, the test suite specifier shall first determine what the conformance requirements are for the relevant Recommendation(s)* and what is stated in the PICS proforma concerning the implementation of those Recommendation(s)*.

9.2 Conformance requirements

Section 1 of this part of the Recommendation specifies the requirements to be met by protocol specifiers as a prerequisite to the production of an abstract test suite for a particular protocol.

In practice, early OSI* Recommendations* might not contain a clear specification of all the relevant conformance requirements. In particular, the static conformance requirements might be badly specified or even omitted. In such cases, the test suite specifier shall contribute to the production of an amendment or addendum to the relevant Recommendation(s)* to clarify the conformance requirements. If, however, an abstract test suite has to be produced in advance of the conformance requirements being clarified in the relevant Recommendation(s)*, then the test suite specifier shall adopt the short-term solution given in Annex C and state clearly in the test suite Recommendation* what the implications of this are (i.e. what is assumed to be mandatory, what conditional and what the conditions are, and what is assumed to be optional).

9.3 PICS proforma

If no PICS proforma is included in a relevant protocol Recommendation*, the test suite specifier shall provide a PICS proforma to be processed as an addendum to that Recommendation*, or in exceptional circumstances (as discussed in § 7.1.1) as an annex to the abstract test suite Recommendation*.

Note - Progressing an addendum to an existing protocol Recommendation* may well be faster than progressing the abstract test suite Recommendation* because the PICS proforma is likely to be less controversial than the test suite and therefore is likely to require fewer revisions before a final text can be agreed.

10. Test suite structure

10.1 Basic requirements

An abstract test suite shall comprise a number of test cases. The test cases shall be grouped into test groups, if necessary nested. The structure shall be hierarchical in the sense that an item at a lower level shall be completely contained within a higher level item. The structure need not, however, be strictly hierarchical in the sense that any one test case may occur in more than one test suite or test group. Similar test groups may occur in more than one higher level test group or test suite.

The abstract test suite specifier shall ensure that a subset of the test purposes of each abstract test suite is concerned with capability testing, and another subset is concerned with behaviour testing. This need not lead to distinct test cases for behaviour and capability testing because it may be possible to use the same test steps for both a behaviour test purpose and for a capability test purpose. The test suite specifier shall provide an explanation of how the test purposes are derived from or relate to the protocol Recommendation*. The test suite specifier shall also provide a summary of the coverage achieved by the test suite.

10.2 The test group structure

In order to ensure that the resulting abstract test suite provides adequate coverage of the relevant conformance requirements, the test suite specifier is advised to design the test suite structure in terms of nested test groups in a top down manner (see Figure 1).

There are many ways of structuring the same test suite into test groups; no one way is necessarily right and the best approach for one test suite may not be appropriate for another test suite. Nevertheless, the test suite specifier shall ensure that the test suite includes test cases for whichever of the following categories is relevant:

- a) capability tests (for static conformance requirements);
- b) behaviour tests of valid behaviour;
- c) behaviour tests of syntactically invalid behaviour;
- d) behaviour tests of inopportune behaviour;
- e) tests focusing on PDUs sent to the IUT;
- f) tests focusing on PDUs received from the IUT;
- g) tests focusing on interactions between what is sent and received;
- h) tests related to each protocol mandatory feature;
- i) tests related to each optional feature which is implemented;
- j) tests related to each protocol phase;
- k) variations in the test event occurring in a particular state;
- l) timing and timer variations;

- m) PDU encoding variations;
- n) variations in values of individual parameters;
- o) variations in combinations of parameter values.

This list is not exhaustive; additional categories might be needed to ensure adequate coverage of the relevant conformance requirements for a specific test suite. Furthermore, these categories overlap one another and it is the task of the test suite specifier to put them into an appropriate hierarchical structure.

The following structure is an example of a single-layer test suite, provided for guidance:

- A. Capability tests
 - A.1 Mandatory features
 - A.2 Optional features said by the PICS to be supported
- B. Behaviour tests: response to valid behaviour by peer implementation
 - B.1 Connection establishment phase (if relevant)
 - B.1.1 Focus on what is sent to the IUT
 - B.1.1.1 Test event variation in each state
 - B.1.1.2 Timing/timer variation
 - B.1.1.3 Encoding variation
 - B.1.1.4 Individual parameter value variation
 - B.1.1.5 Combination of parameter values
 - B.1.2 Focus on what is received from the IUT
 - substructured as B.1.1
 - B.1.3 Focus on interactions
 - substructured as B.1.1
 - B.2 Data transfer phase
 - substructured as B.1
 - B.3 Connection release phase (if relevant)
 - substructured as B.1
- C. Behaviour tests: response to syntactically invalid behaviour by peer implementation
 - C.1 Connection establishment phase (if relevant)

C.1.1 Focus on what is sent to the IUT

- C.1.1.1 Test event variation in each state
- C.1.1.2 Encoding variation of the invalid event
- C.1.1.3 Individual invalid parameter value variation
- C.1.1.4 Invalid parameter value combination variation

C.1.2 Focus on what the IUT is requested to send

C.1.2.1 Individual invalid parameter values

C.1.2.2

Invalid combinations of parameter values

C.2 Data transfer phase

- substructured as C.1

C.3 Connection release phase (if relevant)

- substructured as C.1

D. Behaviour tests: response to inopportune events by peer implementation

D.1 Connection establishment phase (if relevant)

D.1.1 Focus on what is sent to the IUT

- D.1.1.1 Test event variation in each state
- D.1.1.2 Timing/timer variation
- D.1.1.3 Special encoding variations
- D.1.1.4 Major individual parameter value variations

D.1.1.5 Variation in major combination of parameter values

D.1.2 Focus on what is requested to be sent by the IUT

- substructured as D.1.1

D.2 Data transfer phase

- substructured as D.1

D.3 Connection release phase (if relevant)

- substructured as D.1

If the test suite is to cover more than one layer, then a single-layer test suite structure such as this could be replicated for each layer concerned. In addition, a correspondingly detailed structure could be produced for testing the capabilities and behaviour of multiple layers taken as a whole, including the interaction between the activities in adjacent layers.

10.3 Test purposes

The test suite specifier shall ensure that, for each test case in an abstract test suite, there is a clear statement of the test purpose. It is suggested that these test purposes should be produced as the next refinement of the test suite, after its structure in terms of test groups has been defined. The test purposes could be produced directly from studying those sections in the relevant Recommendation(s)* which are appropriate to the test group concerned. For some test groups, the test purposes might be derivable directly from the protocol state table; for others, they might be derivable from the PDU encoding definitions or the descriptions of particular parameters, or from text which specifies the relevant conformance requirements. Alternatively, the test suite specifier could employ a formal description of the protocol(s) concerned and derive test purposes from that by means of some automated method.

Whatever method is used to derive the test purposes, the test suite specifier shall ensure that they provide an adequate coverage of the conformance requirements of the Recommendation(s)* concerned. There shall be at least one test purpose related to each distinct conformance requirement.

In addition, it is possible to give guidance on the meaning of "adequate coverage" with reference to the above example. In order to express this, a shorthand notation will be used: the letter "x" will represent all appropriate values for the first digit in the test group identifier, and similarly "y" for the second digit, so that B.x.y.1 stands for B.1.1.1, B.1.2.1, B.1.3.1, B.2.1.1, B.2.2.1, B.2.3.1, B.3.1.1, B.3.2.1 and B.3.3.1. With this notation, a minimum "adequate coverage" for the above example is considered to be as follows:

- a) for capability test groups (A.1, A.2):
 - 1) at least one test purpose per relevant feature, class or subset;
 - 2) at least one test purpose per relevant PDU type and each major variation of each such type, using "normal" or default values for each parameter;
- b) for test groups concerned with test event variation in each state (B.x.y.1, C.x.1.1, D.x.y.1):
 - at least one test purpose per relevant state/event combination;
- c) for test groups concerned with timers and timing (B.x.y.2, D.x.y.2):
 - 1) at least one test purpose concerned with the expiry of each defined timer;
 - 2) at least one test purpose concerned with very rapid response for each relevant type of PDU;
 - 3) at least one test purpose concerned with very slow response for each relevant type of PDU;
- d) for test groups concerned with encoding variations (B.x.y.3, C.x.1.2, D.x.y.3):
 - at least one test purpose for each relevant kind of encoding variation per relevant PDU type;
- e) for test groups concerned with valid individual parameter values (B.x.y.4, D.x.y.4):
 - 1) for each relevant integer parameter, test purposes concerned with the boundary values and one randomly selected mid-range value;

- 2) for each relevant bitwise parameter, test purposes for as many values as practical, but not less than all the "normal" or common values;
- 3) for other relevant parameters, at least one test purpose concerned with a value different from what is considered "normal" or default in other test groups;
- f) for test groups concerned with invalid individual parameter values (C.x.1.3, C.x.2.1):
 - 1) for each relevant integer parameter, test purposes concerned with invalid values adjacent to the allowed boundary values plus one other randomly selected invalid value;
 - 2) for each relevant bitwise parameter, test purposes for as many invalid values as practical;
 - 3) for all other relevant types of parameter, at least one test purpose per parameter;
- g) for test groups concerned with combinations of parameter values (B.x.y.5, C.x.1.4, C.x.2.2, D.x.y.5):
 - 1) at least one test purpose per "critical" value pair;
 - 2) at least one test purpose per pair of interrelated parameters to test a random combination of relevant values.

11. Generic test case specification

11.1 Introduction

The test suite specifier is recommended to specify a generic test suite, particularly if there is an intention to produce more than one abstract test suite.

A generic test suite shall consist of one generic test case for each test purpose. Each generic test case will be a refinement of its test purpose, which can be used as a common root of corresponding abstract test cases for different test methods.

If a generic test suite is produced in advance of abstract test suites, then it will be a useful step in the design process. If the generic test suite is produced after the production of at least one abstract test suite, then it will provide a means of relating different test suites to one another and analyzing where there may be gaps in their coverage.

11.2 Description of generic test cases

A generic test case consists of a test description of an "initial state" [of the test body] and a specification of the test body in a standardized test notation. The "initial state" includes not only the protocol state, but also any necessary information concerning the state of the SUT and the testing environment.

The test body is that part of a test case in which verdicts related to the test purpose are assigned to foreseen outcomes. The test body:

- a) shall be defined in the style of either the Remote or Distributed test method;

Note - Generic test cases may use the full expressive power of these methods (see § 12.3.3 for details), including the use of abstract local primitives.

b) shall assign verdicts to all possible outcomes; all outcomes yielding "pass" verdicts shall be explicitly identified, whereas all outcomes yielding "fail" or "inconclusive" verdicts shall be either identified or categorized (which may include categorization of default verdicts);

c) shall be described using a standardized test notation; the Tree and Tabular Combined Notation (TTCN) (defined in Annex D) is recommended.

11.3 Relation of generic to abstract test cases

For a particular abstract test method there may be many abstract test cases that can be derived from a single generic test case. A major difference between an abstract test case and a generic test case is that the abstract test case includes specifications of a preamble and a postamble. The preamble starts from a chosen stable state and leads to the required initial state of the test body. The postamble starts from the end of the test body and returns to a chosen stable state.

The preamble and postamble may be realized in different ways depending on the degree of control and observation provided by the test method used, or on the variety of different possible stable states which the derived abstract test case can start from and end in. These abstract test cases are simply different ways of achieving the same test purpose.

In addition, the test body of an abstract test case may be different from the corresponding generic test case if the test method used for the abstract test case is different from that used for the generic test case.

If a generic test suite is produced, it shall be used as the means of relating corresponding abstract test suites for different abstract test methods.

12. Abstract test methods

12.1 Introduction

Each abstract test suite shall be specified in terms of the control and observation of events in accordance with one of the abstract test methods defined in this section. The chosen test method determines the points at which control and observation shall be specified and the categories of events which shall be used (e.g. (N-1)-ASPs, (N)-PDUs).

12.2. General specification of the abstract test methods

12.2.1 End-system IUTs

For IUTs within end-system SUTs there are four categories of abstract test methods, 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.

For generality, the test methods are described with reference to an IUT in which the highest layer is numbered "N_t" (for "top") and in which the lowest layer protocol to be tested is numbered "N_b" (for "bottom"). The IUT may implement protocols in layers lower than "N_b", but these are not of interest in the test method descriptions. The description applies to single-layer IUTs by taking N_t to be equal to N_b.

12.2.2 Abstract local primitives

Abstract test suite specifiers may, if necessary, define a set of abstract local primitives (ALPs) to be used in specifying the test suite. An ALP is an abbreviation for a description of control and/or observation to be performed by the upper tester, which cannot be described in terms of ASPs but which initiate events or state changes defined within the relevant protocol Recommendation(s)*. ALPs may be used with any abstract test method for end- system IUTs, but, for simplicity, will not be shown in any of the figures which illustrate those methods.

Any test case which uses an ALP shall be optional in the abstract test suite.

12.2.3 The local test methods

Abbreviation: L

In this method:

- a) the abstract test suite is specified in terms of control and observation of (N_b-1) -ASPs and (N_b) to (N_t) -PDUs;
- b) the abstract test suite is also specified in terms of control and observation of (N_t) -ASPs;
- c) the abstract test suite may also be specified in terms of control and observation by the upper tester of abstract local primitives (ALPs);
- d) the method requires access to the lower and upper boundaries of the IUT and a mapping between the specified ASPs and their realization within the SUT;
- e) the requirements for the test coordination procedures are specified in the abstract test suite but no assumption is made regarding their realization;
- f) the upper and lower testers are required to achieve control and observation of the specified ASPs and the required test coordination procedures. They are assumed to be integrated into the SUT.

This method is illustrated in Figure 1.

12.2.4 External test methods

12.2.4.1 The distributed test method

Abbreviation: D

In this method:

- a) the abstract test suite is specified in terms of control and observation of (N_b-1) -ASP"s and (N_b) to (N_t) -PDUs;
- b) the abstract test suite is also specified in terms of control and observation of (N_t) -ASPs; the method requires access to the upper boundary of the IUT and a mapping between the (N_t) -ASPs and their realization within the SUT;
- c) the abstract test suite may also be specified in terms of control and observation by the upper tester of ALPs;

- d) the requirements for the test coordination procedures are specified in the abstract test suite but no assumption is made regarding their realization;
- e) the upper tester is required to achieve control and observation of the specified (N_t)-ASPs and to achieve the effects of the required test coordination procedures; no other assumptions are made;
- f) the lower tester is required to achieve control and observation of specified (N_b)-ASP"s and specified PDUs and to achieve the required test coordination procedures.

This method is illustrated in Figure 2.

FIGURE 1/X.290, PART 2 FIGURE 2/X.290 - PART 2

The local test method

The distributed test method

FIGURE 3/X.290, PART 2

FIGURE 4/X.290, PART 2

The coordinated test method

The remote test method

12.2.4.2 The coordinated test method

Abbreviation: C

In this method:

- a) the abstract test suite is specified in terms of control and observation of (N_b-1) -ASP"s, (N_b) to (N_t) -PDUs and test management PDUs (TM-PDUs);
- b) (N_t) -ASPs need not be used in the specification of the abstract test suite; no assumption is made about the existence of an upper boundary of the IUT;
- c) the requirements for the test coordination procedures are specified in the abstract test suite by means of a standardized test management protocol;
- d) TM-PDUs may be defined which correspond to ALPS;
- e) the upper tester is required to implement the test management protocol and achieve the appropriate effects on the IUT;
- f) the lower tester is required to achieve control and observation of specified (N_b) -ASP"s and specified PDUs (including TM-PDUs).

This method is illustrated in Figure 3.

12.2.4.3 The remote test method

Abbreviation: R

In this method, provision is made for the case where it is not possible to observe and control the upper boundary of the IUT. Also in this method:

- a) the abstract test suite is specified in terms of control and observation of (N_b-1) -ASP"s and (N_b) to (N_t) -PDUs;
- b) (N_t) -ASPs are not used in the specification of the abstract test suite; no assumption is made about the existence of an upper boundary of the IUT;
- c) the abstract test suite may also be described in terms of control and observation of ALPs within the SUT;
- d) some requirements for test coordination procedures may be implied or informally expressed in the abstract test suite but no assumption is made regarding their feasibility or realization;
- e) abstractly the SUT needs to carry out some upper tester functions to achieve whatever effects of test coordination procedures and whatever control and/or observation of the IUT are implied, informally expressed or described by ALPs in the abstract test suite for a given protocol; these functions are not specified nor are any assumptions made regarding their feasibility or realization;
- f) the lower tester is required to achieve control and observation of specified (N_b) -ASP"s and specified PDUs and should attempt to achieve the implied or informally expressed test coordination

procedures in accordance with the relevant information in the PIXIT.

This method is illustrated in Figure 4.

12.2.5 Single-layer, multi-layer and embedded variants

Each category of test methods has a variant which can be applied to single-layer IUTs (abbreviation: S), and another which can be applied to multi-layer IUTs (abbreviation: M), when the set of adjacent layers is to be tested in combination (as a whole).

For a multi-layer IUT in which the protocols are to be tested layer by layer, an embedded variant of the test methods has been defined (abbreviation: E).

If control and observation are applied to a means of access to the upper boundary of the entities under test within SUT, then the test methods are normal (and no E is added to the abbreviated name). If, however, control and observation are applied through one or more OSI* layer entities above the entities under test, the test methods are called embedded (and an E is appended to the abbreviated name).

Names of particular variants of the test methods shall be formed as follows:

For example, DSE is the abbreviation for the "distributed single embedded" test method.

12.2.6 Open relay-systems

For open relay-systems, loop-back and transverse abstract test methods are defined. They are given the abbreviated names: YL and YT, respectively.

12.3 Single-layer test methods for single-layer IUTs in end-systems

For single-layer test methods, the abstract model of the IUT is called the (N)-entity under test.

12.3.1 The Local Single-layer test method

The Local Single-layer (LS) abstract test method defines the points of control and observation as being at the service boundaries above and below the (N)-entity under test. The test events are specified in terms of (N)-ASPs above the IUT, and (N-1)-ASPs and (N)-PDUs below the IUT, as shown in Figure 5. In addition, ALPs may be used as test events. Abstractly, a lower tester is considered to observe and control the (N-1)-ASPs and (N)-PDUs while an upper tester observes and controls the (N)-ASPs and ALPs. The requirements to be met by test coordination procedures used to coordinate the realizations of the upper and lower testers are defined in the abstract test suites, although the test coordination procedures themselves are not.

12.3.2 The Distributed Single-layer test method

The Distributed Single-layer (DS) abstract test method defines the points of control and observation as being at the service boundaries above the (N)-entity under test and above the (N-1)-Service Provider at the SAP remote from the (N)-entity under test. The test events are specified in terms of (N)-ASPs above the IUT and (N-1)-ASP"s and (N)-PDUs remotely, as shown in Figure 6. In addition, ALPs may be used as test events. Abstractly, lower and upper testers are again considered to observe and control the behaviour at the respective points. The requirements to be met by the test coordination procedures are again defined in the abstract test suites, although the procedures themselves are not.

For lower layers (1-3) where it may be unrealistic to specify observation and control of (N-1)-ASPs, the lower tester observation and control shall be specified in terms of (N)-PDUs and, when necessary, changes in the state of the underlying connection.

Note - For example, the state of the underlying connection could be changed by setting up a new connection, or resetting or closing an existing connection.

The observation and control to be performed by the lower tester can optionally be specified in terms of (N)-ASP"s where this will reduce the size of the test case specification without loss of required precision.

12.3.3 The Coordinated Single-layer test method

The Coordinated Single-layer (CS) abstract test method is an enhanced version of the DS method, using a standardized upper tester and the definition of a test management protocol to realize the test coordination procedures between the upper and lower testers. The same standardized upper tester and test management protocol are not necessarily applicable to all test suites which use the coordinated method.

Standardized upper testers and test management protocols are applicable to a particular standardized abstract test suite for the coordinated test method and may not be applicable to other abstract test suites for the coordinated method.

There is only a single point of control and observation, above the (N-1)-Service Provider at the SAP remote from the (N)-entity under test. Test events are specified in terms of (N-1)-ASP"s, (N)-PDUs and TM-PDUs, as shown in Figure 7.

For lower layers (1-3) where it may be unrealistic to specify observation and control of (N-1)-ASP"s, the observation and control shall be specified in terms of TM-PDUs, (N)-PDUs and, when necessary, changes in the state of the underlying connection.

Concerning the test management protocol:

- a) the test management protocol shall be implemented within the SUT directly above the abstract service boundary at the top of the IUT;
- b) the IUT shall not be required to interpret TM-PDUs, only pass them to and from the upper tester;
- c) a test management protocol is only defined for testing a particular protocol and so does not need to be independent of the underlying protocol;
- d) verdicts on test cases shall not be based on the ability of the SUT to exhibit any ASP or parameter of an ASP at the upper service boundary of the IUT, since this would contradict the definition of the coordinated method in that the upper service boundary of the IUT is not a point of control and observation in this method. However, it is recommended that the test management protocol be defined separately from the abstract test suites(s) in order to facilitate the task of the implementor of an upper tester. This definition (as with the definition of any OSI* protocol defined by ISO or CCITT) can refer to the ASPs of its underlying service (i.e. the ASPs at the upper service boundary of the IUT);
- e) TM-PDUs which correspond to ALPs are optional and there is no requirement for the upper tester to support them.

FIGURE 5/X.290, PART 2

FIGURE 6/X.290, PART 2

The LS test method

The DS test method

FIGURE 7/X.290, PART 2

FIGURE 8/X.290, PART 2

The CS test methodThe RS test method12.3.4 The Remote Single-layer test method

The Remote Single-layer (RS) abstract test method defines the point of control and observation as being above the (N-1)-Service Provider at the SAP remote from the (N)-entity under test. The test events are specified in terms of the (N-1)-ASP"s and (N)-PDUs remotely, as shown in Figure 8. In addition, ALPs may be used as test events. Some requirements for test coordination procedures may be implied or informally expressed in the abstract test suites, but no assumptions shall be made regarding their feasibility or realization.

For the lower layers (1-3) where it is unrealistic to specify observation and control of (N-1)-ASP"s, the observation and control shall be specified in terms of (N)-PDUs and when necessary the state of the underlying connection.

In addition, in order to overcome the lack of specification of behaviour above the (N)-entity under test, where necessary the required behaviour of the system under test shall be specified in terms of the (N-1)-ASP"s or (N)-PDUs which need to be observed by the lower tester. This form of implicit specification shall be taken to mean "do whatever is necessary within the system under test in order to provoke this required behaviour".

Note - Such implicit specification is necessary with this test method for any test case which requires an IUT initiated event which cannot be initiated by means of an ALP. Since ALPs may only be defined if the same effect cannot be achieved by ASPs, then any PDU which can be initiated by an ASP needs implicit specification to allow it to be initiated in this test method.

However, it is possible that some of the test cases in the abstract test suite cannot be executed (e.g. transmission and maintenance of busy conditions, transmission of consecutive unacknowledged Data PDUs, etc.). In such instances, it is left to the test laboratory and client to negotiate the method by which these tests can be accomplished.

Even with such implicit specification of control of the IUT, in this method it is possible to specify control but not observation above the IUT, except through the use of ALPs. This is a major difference between this and the DS and CS test methods.

12.4 Multi-layer test methods for multi-layer IUTs (LM, DM, CM, RM)

Multi-layer testing, when the combined allowed behaviour of the multi-layer implementation is known, involves testing all the layers of a multi-layer IUT as a whole, without controlling or observing any of the inter-layer boundaries within the IUT.

In the Local Multi-layer (LM) test method the points of observation and control are the service boundaries directly above and below the IUT. The test events shall be specified in terms of the (N_i)-ASPs and ALPs above the IUT and the (N-1)-ASPs and (N) to (N_i)-PDUs below the IUT.

In the Distributed Multi-layer (DM) test method the points of observation and control are at the service boundary above the IUT and above the (N-1)-Service Provider at the SAP remote from the IUT. The test events shall be specified in terms of the (N_i)-ASPs and ALPs above the IUT and the (N-1)-ASP"s and (N) to (N_i)-PDUs remotely.

In the Coordinated Multi-layer (CM) test method the point of observation and control is above the (N-1)-Service Provider at the SAP remote from the IUT. The test events shall be specified in terms of the (N-1)-ASP"s, the

(N) to (N_i)-PDUs and the TM-PDUs. The test management protocol shall be designed to operate over the (N_i)-Service, where (N_i) is the highest layer in the IUT.

In the Remote Multi-layer (RM) test method the point of observation and control is above the (N-1)-Service Provider at the SAP remote from the IUT. The test events shall be specified in terms of the (N-1)-ASP"s and the (N) to (N_i)-PDUs, ALPs and the implicit specification of the control of the SUT behaviour when necessary. Some requirements for test coordination procedures may be implied or informally expressed, but no assumptions shall be made regarding their feasibility or realization.

12.5 Single-layer testing of multi-layer IUTs or SUTs (embedded methods)

In embedded single-layer test methods testing is specified for a single-layer within a multi-layer IUT, including the specification of the protocol activity in the layers above the one being tested but without specifying control or observation at service boundaries within the multi-layer IUT. Thus in a multi-layer IUT from layer (N) to (N_i), abstract test cases for testing layer (N_i) shall include the specification of the PDUs in layers (N_i+1) to (N_i) as well as those in layer (N_i).

The Local Single-layer Embedded (LSE) test method uses the same points of control and observation as the LM test method for the same set of layers. The test events shall also be specified in the same terms as for the LM test method. The difference is that the LSE test method concentrates on a single-layer at a time, whereas the LM test method tests the multi-layer IUT as a whole.

In the Distributed Single-layer Embedded (DSE) test method for layer (N_i) within a multi-layer IUT from layer (N) to (N_i) the points of observation and control are at the service boundary above the IUT and above the (N_i-1)-Service Provider at the SAP remote from the IUT, as illustrated in Figure 9(a). The test events shall be specified in terms of (N_i)-ASPs and ALPs above the IUT and (N_i-1)-ASP"s and (N_i) to (N_i)-PDUs remotely.

Note - For the top layer in the multi-layer IUT, (N_i), this method is the same as the DS test method.

The Coordinated Single-layer Embedded (CSE) test method uses features of both the CM and DSE test methods. The test events shall be specified in terms of (N_i-1)-ASP"s, (N_i) to (N_i)-PDUs and TM-PDUs and the test management protocol shall be designed to operate over the (N_i)-Service. This is illustrated in Figure 9(b).

The Remote Single-layer Embedded (RSE) test method uses the same point of control and observation as the RS test method for the same layer, but differs from the RS test method in that (N_i+1) to (N_i)-PDUs shall be specified in test cases for layer (N_i).

Successive use of a single-layer embedded test method (from layer (N) to (N_i)) is called incremental testing of a multi-layer IUT.

The DSE/CSE/RSE methods are defined for a single layer under test in a multi-layer IUT. This does not mean that there cannot be accessible service boundaries within the multi-layer IUT, just that no such boundaries are used in the test methods. Thus, all layers between the layer under test and the highest layer for which PDUs are expressed as test events in the abstract test suite shall be regarded as being part of the multi-layer IUT.

Note - DME/CME/RME test methods could theoretically be defined to test multiple layers as a whole within a larger multi-layer IUT, but in order to avoid excess complexity, they are not detailed in this Recommendation.

12.6 Relay test methods

There are two abstract test methods for relay system testing:

- a) the loop-back test method (YL): used for testing a relay system from one subnetwork.

This test method is illustrated in Figure 10.

For this test method there are two points of observation and control on one subnetwork at SAPs remote from the (N)-Relay. For connection-oriented protocols, it requires that the two test connections are looped together on the far side of the (N)-Relay, but it is not specified whether this looping is performed within the (N)-Relay or in the second subnetwork. For connectionless protocols, it requires that the PDUs are looped back within the second subnetwork and addressed to return to the second point of observation and control.

- b) the transverse test method (YT): used for testing a relay system from two subnetworks.

This test method is illustrated in Figure 11.

In this test method there are two points of observation and control, one on each subnetwork, at SAPs remote from the (N)-Relay.

FIGURE 9/X.290, PART 2

The embedded test methods

FIGURE 10/X.290, PART 2

Loop-back test method (YL)

FIGURE 11/X.290, PART 2

Transverse test method (YT)

12.7 Choice of test method

12.7.1 Introduction

Before an abstract test suite can be defined, it is necessary to study all the environments in which the protocol is likely to be tested and establish accordingly the abstract test method(s) to be used for the production of one or more abstract test suites.

Abstract test suite specifiers shall place a requirement in the abstract test suite Recommendation* defining which abstract test method(s) shall be supported as a minimum by any organization claiming to provide a comprehensive testing service for the protocol(s) in question.

12.7.2 IUT environments

There is a relation between the test methods and the configurations of the real open systems to be tested.

Section 7.2, Part 1 of this Recommendation gives a full account of the classification of systems and IUTs.

When choosing a test method, the test suite specifiers should identify, if they have not already done so, whether the test suites they plan to produce are for IUTs which

- a) are single or multi-layer;
- b) belong to end or relay systems;

- c) belong to complete or partial systems;
- d) belong to fully open or mixed systems;
- e) have service boundaries accessible or not;
- f) are special purpose (i.e. to be used by a single application) or general purpose (i.e. to be used by several applications).

12.7.3 Applicability of the abstract test methods

The possibility of developing an abstract test suite for a given abstract test method will depend on the protocol(s) being considered, together with the results of the study described in subsection 12.7.2. This applies to the possibility of developing test suites for a given combination of methods (e.g. incremental) or a given variant of a method (e.g. embedded).

Some considerations concerning the applicability of the methods to different layers are discussed in Appendix I of Part 1 of this Recommendation.

One or more appropriate abstract test methods shall be selected for the protocol being considered.

Priorities should be assigned to the various test methods which have been identified, according to the configurations which are most likely to be found in real systems.

12.7.4 Illustrative examples

Appendix IV provides an illustrative example of the choice of abstract test methods for given protocols.

12.8 Test coordination procedures

For effective and reliable execution of conformance tests, some set of rules is required for the coordination of the test process between the lower tester and the upper tester. The general objective of these rules is to enable the lower tester to control remotely the operation of the upper tester or vice versa, in ways necessary to run the test suite selected for the IUT.

These rules lead to the development of test coordination procedures to achieve the synchronization between the lower tester and the upper tester and the management of information exchanged during the testing process. The details and how these effects are achieved are closely related to the characteristics of the SUT, as well as the external test methods.

For each abstract test suite using the coordinated, distributed or local test methods, a standard set of test coordination procedures should be developed. This is because those procedures depend on the functionality of the upper tester and definitions of test cases.

In the case of the coordinated test methods (CS, CSE, CM) the test coordination procedures are realized by the standardization of a test management protocol. The test management protocol needs to be able to convey requests to the IUT to achieve the effect of a service primitive or ALP and to convey back to the lower tester the record of observations of effects equivalent to the occurrence of service primitives or ALPs. The upper tester should be an implementation of the test management protocol. It will be necessary to add test cases to the abstract test suite for the purpose of testing that the upper tester conforms to the requirements of the test management protocol specification. Such test cases do not contribute to the conformance assessment of the IUT.

When defining test cases for the local and distributed test methods, the test suite specifier shall record any constraints on the upper tester and/or test coordination procedures which may be necessary.

13. Specification of abstract test suites

13.1 Test cases

13.1.1 The abstract test suite specifier shall select an appropriate standardized notation in which to define the abstract test cases. The Tree and Tabular Combined Notation (TTCN), defined in Annex D, is defined for this purpose.

13.1.2 Once the test notation and test method have been chosen, then the generic test cases can be expanded into abstract test cases. There are two main kinds of change required to convert a generic test case into an abstract test case. The first is to express the test body in terms of control and observation required by the test method, and, if relevant, include a description of the synchronization needed between upper and lower testers. The second kind of change is to specify the preamble and postamble.

13.1.3 Specification of preambles and postambles may result in more than one test step for each. The preamble shall start in a stable state and progress to the desired state. Conversely, the postamble shall progress from the final state of the test body to a stable state. A small number of stable states shall be defined for the protocol concerned, always containing as a minimum the "idle" state. Each abstract test case shall be capable of being run on its own and shall therefore include test steps to start the preamble from the "idle" state and to end the postamble in the "idle" state.

However, other stable starting and final states for an abstract test case can be useful when efficient concatenation of abstract test cases is required.

Furthermore, in designing the test step structure within the abstract test cases, the test suite specifier can benefit from using the same test steps in several abstract test cases.

13.1.4 In converting from generic test cases to abstract test cases, the test suite specifier shall ensure that the initial state for the test body is preserved, the pass paths through the test body are preserved and the assignment of verdicts to outcomes remains consistent.

In order to maintain consistency, the following conditional requirements apply when assigning verdicts to outcomes:

- a) if the behaviour of the preamble and the postamble are valid then the verdict assigned to a particular outcome shall be the same as that assigned to the corresponding outcome in the generic test case;
- b) if the preamble results in the initial state of the test body not being reached, although there is no invalid behaviour, then the verdict shall be inconclusive;
- c) if the preamble results in the initial state of the test body not being reached, because of invalid behaviour, then the verdict shall be "fail but test purpose inconclusive" ("fail type 3");
- d) if the postamble exhibits invalid behaviour and the generic test case (or test body) verdict is "pass" or "inconclusive", then the verdict shall be "fail but test purpose passed" ("fail type 2") or "fail but test purpose inconclusive" ("fail type 3"), respectively;
- e) if the generic test case (or test body) verdict is "fail" then invalid behaviour in the postamble

shall not change the verdict ("fail type 1").

13.1.5 The test suite specifier shall also ensure that each abstract test case explicitly identifies all the outcomes assigned the verdict "pass" and identifies or categorizes all the remaining foreseen outcomes, assigning to each individual outcome or category a verdict "fail" or "inconclusive". All unforeseen outcomes in the test body shall be assigned either

- a) the verdict "fail"; or
- b) the verdict "inconclusive".

The test suite specifier shall ensure that the application of a) or b) is consistent throughout the abstract test suite. If a) is chosen, then any unforeseen outcome in the preamble shall be assigned the verdict "fail but test purpose inconclusive" ("fail type 3"), and any unforeseen outcome in the postamble shall be treated as a protocol violation, leading to the appropriate type of fail verdict.

If b) is chosen, then any unforeseen outcome in the preamble shall be assigned the verdict "fail but test purpose inconclusive" ("fail type 3"), and any unforeseen outcome in the postamble shall be assigned the appropriate type of fail verdict.

13.2 Test suites

An abstract test suite specification comprises a set of test cases and test steps. Preceding the test cases themselves shall be the following information:

- a) abstract test suite name, date of origin and version number;
- b) names (and version numbers) of the protocol Recommendation(s)* for which test cases are provided;
- c) names (and version numbers) of the service Recommendation(s)* whose primitives are specified as being observed;
- d) name (and version number) of the Recommendation* defining the test notation, or a reference to an annex for the same if it is not standardized elsewhere;
- e) name of target test method;
- f) description of the coverage of the test suite; for example, the functional subsets of the protocol Recommendation(s)* that are tested;
- g) description of the structure of the test suite, in terms of test groups and their relationship to the protocol Recommendation(s)*:
- h) description of the test coordination procedures (if applicable in the test method);
- i) statement of which test cases are optional and which mandatory for conformance to the abstract test suite Recommendation*;
- j) information to assist the test realizer and test laboratory in their use of the abstract test suite Recommendation* (see section 14).

14. Use of an abstract test suite specification

The abstract test suite specifier shall provide information in the abstract test suite Recommendation* to assist the test realizer and test laboratory in their uses of the test suite. This information shall include, but is not limited to, the following:

- a) a mapping of the abstract test cases to the PICS proforma entries to determine whether or not each abstract test case is to be selected for the particular IUT; the mapping should be specified in a suitable notation for Boolean expressions;
- b) a mapping of the abstract test cases to the PIXIT proforma entries, to the extent that they are known by the abstract test suite specifier, to parameterize the test suite for the particular IUT and to determine which selected test cases cannot be run with the particular IUT.

The test suite specifier shall define a partial PIXIT proforma. This shall contain a list of all the ALPs used in the test suite (or test management protocol) and a list of all parameters for which the test suite requires values. If any of the required parameter values will be found in the PICS, the PIXIT proforma entry for each such parameter shall reference the corresponding entry in the PICS proforma.

Note - Other aspects of the PIXIT proforma are for further study.

- c) a list of the abstract test cases in the order that shall be used in the Protocol Conformance Test Report (PCTR), together with any information which shall be preserved in the PCTR on the status of each test case; this is a contribution to the development of a PCTR proforma;
- d) any restrictions that there may be on the order in which the test cases may be executed;
- e) reference to the description of the test coordination procedures (if applicable in the chosen test method);
- f) any necessary timing information.

15. Test suite maintenance

Once an abstract test suite has been specified and is in use, it can be expected that errors and omissions in it will be detected by those who are using the test suite. The abstract test suite specifier shall in such circumstances progress the updating of the test suite via the relevant rapid amendment procedures.

In addition, from time-to-time, changes will be made to the protocol Recommendation(s)* to which the test suite relates. The abstract test suite specifier shall ensure that the test suite is updated as soon as possible after changes to the relevant protocol Recommendation* have been ratified.