CANADIAN THESES

THÈSES CANADIENNES

NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30.

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30.

THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED

LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE

Specification and Design of an OSI Standard Test Suite for X.25 DTEs

by

Biswajit Kanungo

A thesis
presented to the University of Ottawa
in fulfillment of the
thesis requirement for the degree of
Master of Science
in

Systems Science
(Department of Computer Science)
University of Ottawa
Ottawa, Ontario, Canada.

ISBN 0-315-36510-2

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

The University of Ottawa requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

ABSTRACT

The objective of Open Systems Interconnection (OSI) will not be achieved unless systems can be tested to show conformance to standards. The X.25 standard/ recommendation, is especially important since it is recommended by ISO and CCITT as the protocol standard for the lower three layers of the OSI Reference Model. This standard has matured over a number of years, and now provides several implementation options. The probability of interworking between different implementations of X.25 Data Terminating Equipments (DTEs), with implementations of Data Circuit Equipment on the network has to be increased through standard testing. Towards this objective ISO has undertaken work to produce standard X.25 DTE test suites.

This thesis is directed towards the design and specification of an OSI standard X.25 DTE conformance test suite. Using the test methods proposed in the ISO framework for conformance testing, we develop a methodology for designing a standard X.25 DTE test suite. We use the Tree Tabular Combined Notation (TTCN) to specify a standard abstract test suite for the Data Link layer of X.25.

In this thesis we propose a new approach for designing test cases based on the Finite State Machine model of the X.25 LAPB protocol specification. This test design method is applicable to real protocol implementations like the X.25 DTE. In this method we emphasise the use of Proper, Improper and Inopportune interactions in the abstract test suite design. The results indicate that this is a very practical and useful method for generating standard state transition tests.

ACKNOWLEDGEMENTS

I would like to express my very deep sense of gratitude to my thesis supervisor, Professor Robert L. Probert, for his advice, guidance and encouragement throughout the course of my study and research. He spent considerable amount of his precious time in directing this research and made available funds and resources for its execution.

I would also like to express my sincere thanks to Dr. Hasan Ural for his many useful suggestions during the course of this research. I also thankfully acknowledge the help provided by the members and colleagues of the Protocols Research Group, University of Ottawa, by their helpful suggestions.

I would like to thank Mr. Marc W. Hornbeek, Manager, Network Test Tools, Bell-Northern Research, for suggesting this area of research and providing valuable information during the course of this work.

Financial support from Bell-Northern Research and Natural Sciences and Engineering Research Council of Canada through research grants is gratefully acknowledged.

I would like to thank Mr. Henry Fan and Ricky Ma for the many stimulating discussions on this thesis topic. I also thank the Computer Science Department of the University of Ottawa for providing the necessary facilities for this research.

Finally, I would like to thank my wife, Jhuma Kanungo, for her support and encouragements during the crucial final phase of this work.

CONTENTS

LIST OF FIGURES

CHAPTER I
INTRODUCTION TO CONFORMANCE TESTING

1.1 INTRODUCTION

The X.25 protocol standard specifies recommendations for accessing a public packet-switched network. This standard defines the interface between Data Terminating Equipment (DTE) and Data Circuit-Terminating Equipment (DCE) in three layers, namely, Physical layer, Data Link layer, and Packet layer. This recommendation which was first adopted in 1976 has been revised both in 1980 and 1984. Details are in [CCITX25a,CCITX25b,CCITX25c]. Changes and additions resulted in a number of versions of X.25 as documented in [Barto84].

The X.25 Recommendations are defined by CCITT (1976, 1980, 1984), from the point of view of the DCE. This allows for various interpretations of the recommendations from the point of view of the DTE. The ISO X.25 DTE standards [ISO8208,ISO7776] also allow various implementation choices for the DTE. As a result, implementors of DTEs face a wide range of implementation choices [Sheri86]. For reliable communication between a DTE and a DCE, the DTE manufacturer's interpretation of X.25 should be compatible

with the network's implementation of X.25 (i.e., the DCE viewpoint).

One way of assessing the compatibility between a specific DTE and a particular network's DCE is to test the conformance of the DTE to the particular view of X.25 implemented in the network. However, the conformance of a DTE with a given network's implementation of X.25 does not guarantee conformance with implementations of X.25 in other networks due to variations in options and differences in interpretation of X.25 recommendations.

To eliminate the burden on DTE manufactures in testing the conformance of their DTE implementation with each network, ISO is currently working towards a standardized X.25 DTE conformance test suite [DP88821,DP88823]. In this chapter we will introduce the basic terminology of protocol testing and the concepts of conformance test standards. Then we introduce the abstract test specification language TTCN (Tree and Tabular Combined Notation) which is an interim standard notation for test suite specification. Finally, we give a summary of the main results of this thesis and an outline of the remaining chapters.

1.2 Basic OSI Reference Model Terminology

We shall be concerned with the testing of implementations of OSI standard X.25 protocols. We first review some of the basic terms and concepts of OSI. In the next section, we review the main ideas behind X.25 and relate them to the OSI concepts. For more detail see [Zimer80,PavDw84].

1.2.1 Service Layering

To reduce design complexity, data communication networks are organized as a hierarchy of LAYERS. The purpose of each layer is to offer certain services to the next higher layer so that layer-N of one machine could communicate to layer-N of another machine, using the service provided by layer N-1. The set of rules for this communication is collectively called the N-protocol. The International Organization for Standardization (ISO) has standardized a seven layer reference model for data communication networks. This is called the Open Systems Interconnection (OSI) reference model. The OSI reference model is shown in Figure 1.
The X.25 protocol standards specifies the rules for communication in the first three layers of the OSI reference model.

An (N)-Service Provider denotes a process offering some communication service (an (N)-Service) to some other processes ((N)-Service users).

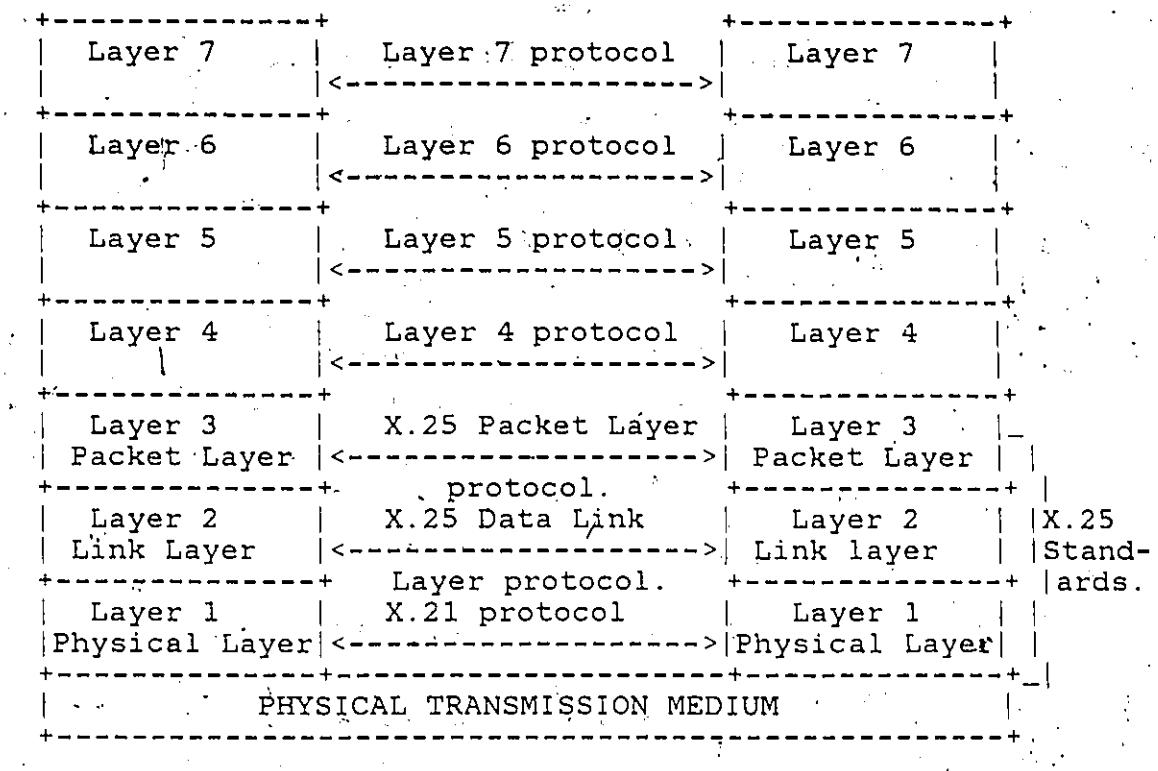


Figure 1: The OSI 7 Layer Reference Model

As shown in Figure 2, an (N)-ENTITY is an (N-1)-Service user which communicates with other (N)-Entities, by means of the (N-1)-Service Provider, to provide an enhanced service (the (N)-Service). The other (N)-Entities are often referred to as PEER ENTITIES, or simply peers.

Typically, (N)-Entities will be distributed over machines in different geographical locations. Protocol testing, the main subject of this thesis, means the testing of implementations of (N)-Entities. Conformance testing is the testing of implementations of (N) - Entities to show conformance to the N-protocol standard.

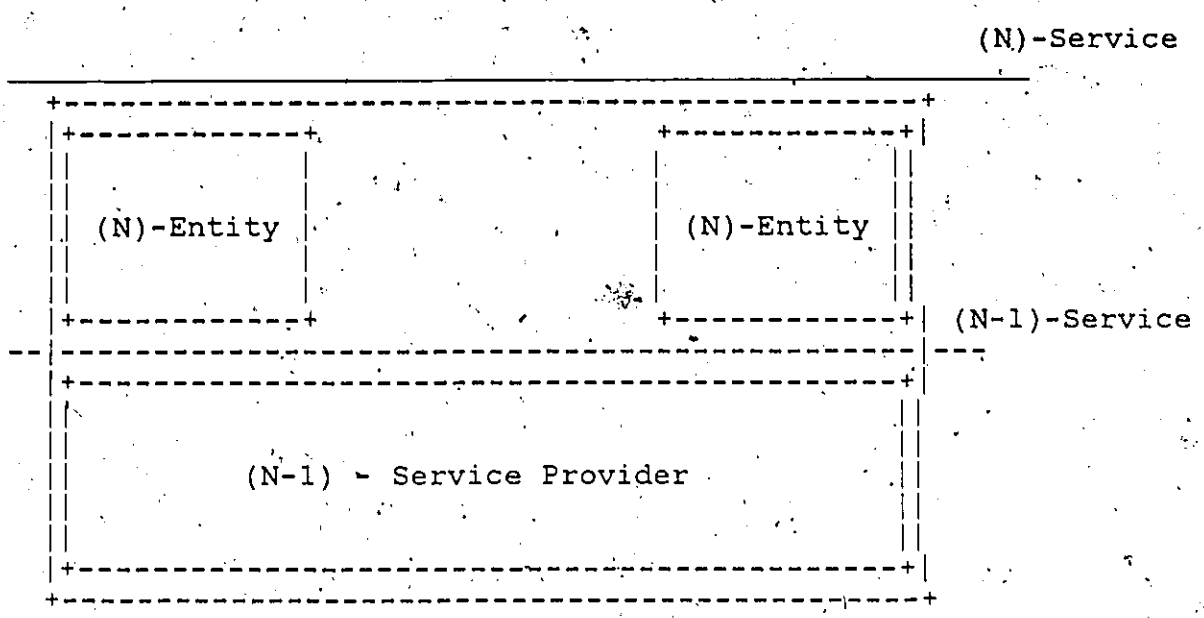


Figure 2: Protocol Entities and Service Layers

An IMPLEMENTATION UNDER TEST (IUT) is the part of an open system which implements a protocol and which is to be tested for conformance to that protocol. It is an implementation of one or more adjacent layers of protocols. The SYSTEM UNDER TEST (SUT) is the real open system in which the IUT resides [ISO909].

Three types of data and control interactions between layers need to be considered, namely Service Data Units (SDUs), Protocol Data Units (PDUs) and Abstract Service Primitives (ASPs).

1.2.1.1 SERVICE DATA UNITS

The abstract service provided by a layer can be expressed more concretely in terms of Data Units. These are called SERVICE DATA UNITS. (N)-Service users utilize (N)-SERVICE DATA UNITS ((N)-SDUs) to communicate with each other over the (N)-Service. They submit these, together with relevant control information, to an (N)-Entity. The (N)-Entity attempts to pass (N)-SDUs to a peer (N)-Entity by packaging them inside (N)-PROTOCOL DATA UNITS ((N)-PDUs) which are transferred between (N)-Entities inside (N-1)-SDUs. Refer to Figure 3, for an illustration of PDU and SDU exchanges.

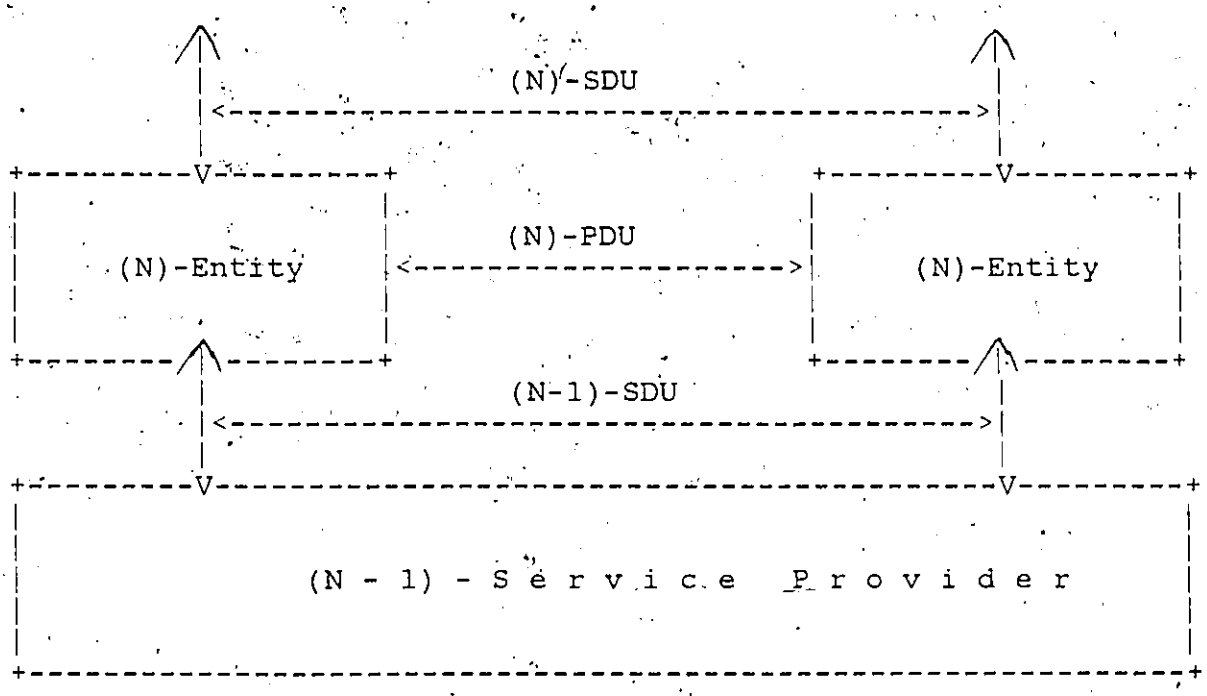


Figure 3: Relationship between PDU and SDU

1.2.1.2 PROTOCOL DATA UNITS

As has already been mentioned, (N)-PDUs are exchanged between (N)-Entities using the (N-1)-SDU transfer service of the (N-1)-Service Provider.

(N)-PDUs have the following attributes:-

- a type;
- (N)-Protocol control information being exchanged between communicating (N)-Entities;
- some types of (N)-PDU also contain part of an (N)-SDU, a whole (N)-SDU or several (N)-SDUs. The way in which successive (N)-SDUs are distributed over successive (N)-PDUs depends on the (N)-Protocol.

An **(N)-PROTOCOL** is an agreed upon procedure for transferring (N)-SDUs and control information between (N)-Entities by means of an underlying (N-1)-Service. The (N)-Protocol is usually expressed either in terms of (N)-SDUs at the N-Service interface and (N-1)-SDUs at the (N-1)-Service interface, or by the (N)-PDU exchange between the (N)-Entities.

In the case of **CONNECTION ORIENTED** protocols (e.g. X.25 switched virtual circuit) there is some initial dialogue in the protocol to establish a logical relationship (connection) between communicating (N)-users, and some final

dialogue (connection release) to dissolve that relationship.

1.2.1.3 ABSTRACT SERVICE PRIMITIVES

As shown in Figure 4, an (N)-Entity's behaviour can be specified in terms of interactions with its users above and interactions with the (N-1)-Service Provider below. These interactions are known as (N)-ABSTRACT SERVICE PRIMITIVES ((N)-ASPs) and (N-1)-ABSTRACT SERVICE PRIMITIVES respectively [PavDw84].

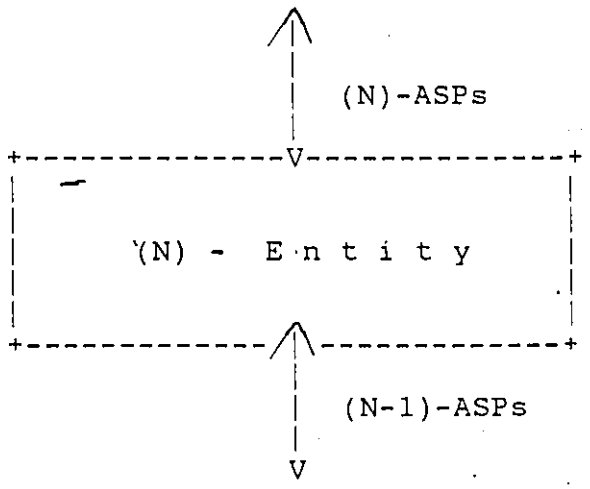


Figure 4: Abstract Service Primitives (ASPs)

These are called ABSTRACT because, for any given implementation, an ASP will represent a particular set of sequences of real interactions called service primitives,

for example, procedure calls or interprocess messages. The detailed characteristics of service primitives are left up to the IUT manufacturer.

From the (N)-Entity's point of view, (N)-ASPs have the following attributes:-

- a type;
- items of (N)-Protocol control information of varying number and significance, exchanged between the (N+1)-Entity and the (N)-Entity. (as opposed to the (N)-Protocol control information in (N)-PDUs which is exchanged between (N)-Entities);
- some types of (N)-ASP also have an associated (N)-SDU which may not be interpreted by the (N)-Entity.

(N-1)-ASPs have the following attributes from the (N)-Entity's point of view:-

- a type;
- some items of (N-1)-Service Provider control information;
- some types also have an associated (N-1)-SDU containing one, or more (N)-PDUs. The permitted number of (N)-PDUs per (N-1)-SDU depends on the particular (N)-Protocol.

In the higher layers of the OSI reference model the ASPs can be clearly defined and are used for standard test

specifications. In the Data Link and Physical layers of X.25 the ASPs are not clearly defined and so PDUs are used for standard test specification.

1.2.2 X.25 Specification and Relation to OSI Reference Model

Effective utilization of data communication networks is possible through dynamic allocation of the transmission medium's bandwidth. For this purpose, information is exchanged as blocks of a limited size or packets. The Physical layer of X.25 deals with the hardware connection and the specifications for the electrical signals. This is presented in the X.21 and the X.21 bis. specifications. This layer is not within the scope of software testing. The Data Link layer of the OSI reference model provides the functions of link initialization, frame synchronization, flow control and error control. This is done in the LAPB (Link Access Procedure-B) of X.25. The CCITT X.25 Recommendations also specify the LAP procedure as optional, but this procedure will not be discussed in this thesis. The third layer of the OSI reference model is the Network Layer, and the Packet layer is a sublayer to this layer. The Packet layer supports multiple logical connections, called Permanent Virtual Circuits (PVC) and/ or Switched Virtual Calls (SVC). Each of these logical channel has an independent flow control and independent error recovery procedure. The DTE to DCE and DCE to DTE interconnections for X.25 are schematically shown in Figure 5.

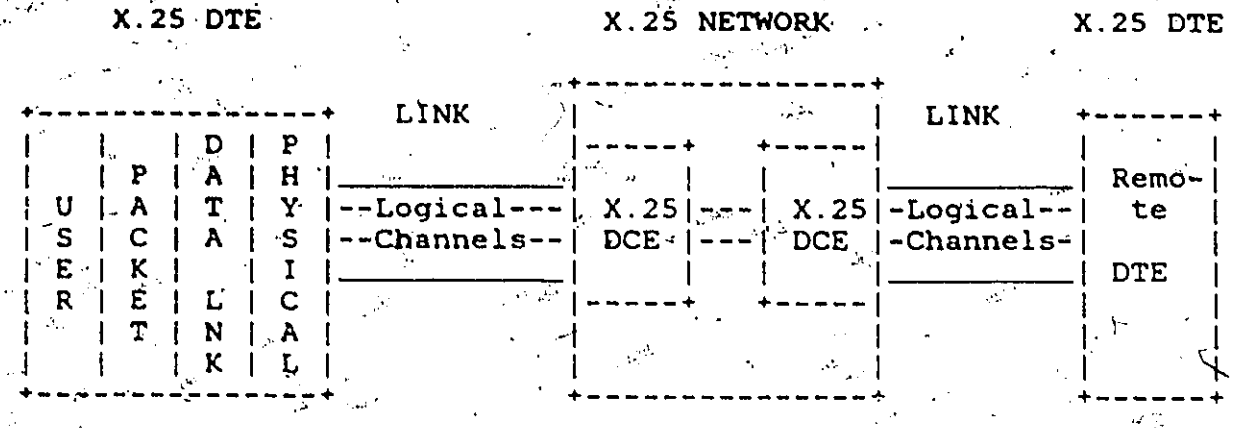


Figure 5: The X.25 Network Connections

The Packet Layer also provides various facilities like, closed user group (CUG), delivery confirmation, registration, etc.

Thus the X.25 standard specifies the basic communication service required for the network. The OSI reference model facilitates the interworking of different types of machines through the X.25 standards.

1.3 Standards for Conformance Testing

The objective of OSI is to promote interworking between different networks and machines by providing international standards and recommendations. This objective of OSI will not be completely achieved until systems can be tested in order to determine whether they conform to the relevant protocol standards. Towards this end the International

Organization for Standardization (ISO) undertook a project supervised by the technical committee 97, subcommittee 21 (TC97/ SC 21) in cooperation with CCITT subgroup VII question 47 (CCITT SG VII Q47). This working group of ISO TC97/ SC 21 has produced a document which is a proposed standard for Conformance Testing Methodology and Framework [ISO909]. This draft proposal (N909) is close to becoming an international standard and is a basic reference for this thesis [ISO909, Rayne85].

Simultaneously another subcommittee (SC6) has been working on a standard conformance test suite for the X.25 Recommendations. The purpose of the standard X.25 test suite is to ensure that the DTE's (Data Terminating Equipment) interpretation of X.25 is compatible with the DCE's (network equipment's) interpretation of X.25 [Sheri86]. Therefore, the purpose of this standard test suite is to increase the probability that different implementations are able to interwork.

Conformance testing is different from PERFORMANCE TESTING. In the latter case the performance of the IUT under varying conditions of load and network traffic are tested. Performance testing is outside the scope of this thesis.

The work of developing this standard X.25 DTE test suite is divided between two working groups (TC97/ SC6 WG1 and

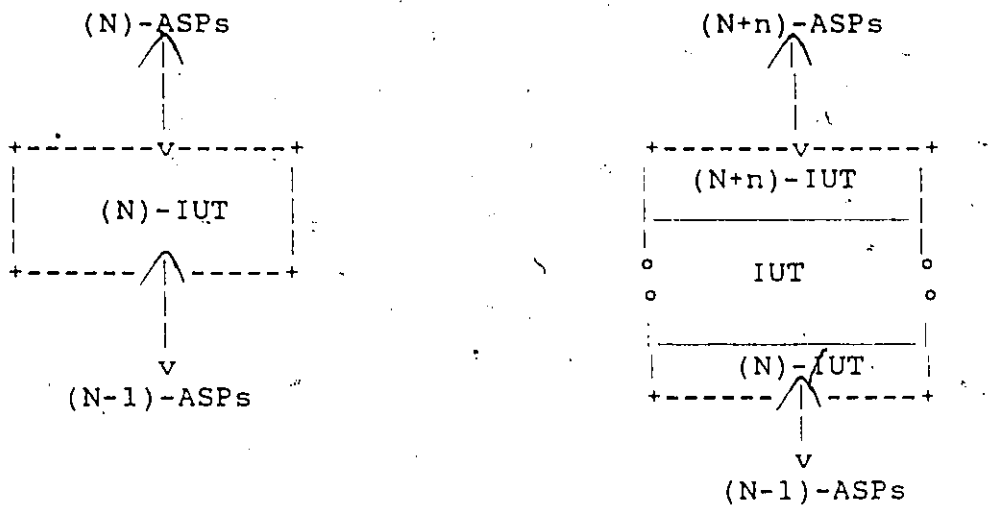
WG2). Working group 2 is entrusted with the X.25 packet level test suite [N3566,N3891] and working group 1 is for the X.25 Data Link Level test suite. The author of this thesis along with other members of the Protocols Research Group, University of Ottawa, has made contributions towards these standard X.25 test suites. The author of this thesis has submitted a TTCN test specification for the Data Link layer to TC97/ SC6 WG1 [CacSC686]. Presently it is under consideration by ISO and is the current official Canadian position on the standard X.25 Data Link Level test suite.

A standard conformance test consists of several steps. First, an appropriate test method is selected. This is followed by a Static conformance review, involving the verification of the SUT manufacturer's claims against the appropriate standards. Dynamic conformance testing may start after these two steps. Finally, analysis of the dynamic test results and the static review results are done to arrive at a verdict about the IUT's conformance to the standards [ISO909,Rayne85].

1.3.1 Abstract Test Methods

The abstract test method represents the testing architecture for each abstract conformance test suite. The abstract test method defines the points of control and observation for the purpose of designing the abstract conformance test suite. These points of observation and control are constrained by

the architecture of the SUT. For example, in X.25 DTE testing, it is common for the manufacturer to not provide access to the upper interface of the Data Link layer. The revised version (San Diego, April 1986) of the OSI Conformance Testing Methodology and Framework (N909) [ISO909] defines five general categories of abstract test methods. In each of these five categories, testing can be performed for a single layer (S), multiple layers (M), or a single layer embedded (E) in other layers. Figure 6, illustrates single and multi-layer IUTs.



(a) Single Layer IUT (b) Multi-layer IUT

Figure 6: Single and Multi-layer IUTs.

First we will define the five categories and then describe the three subcategories (i.e. S, M and E).

(1) Local Test Method (L): In this method the control and observation points are at the local service boundaries, above and below the entity (entities) under test (see Figure 7.

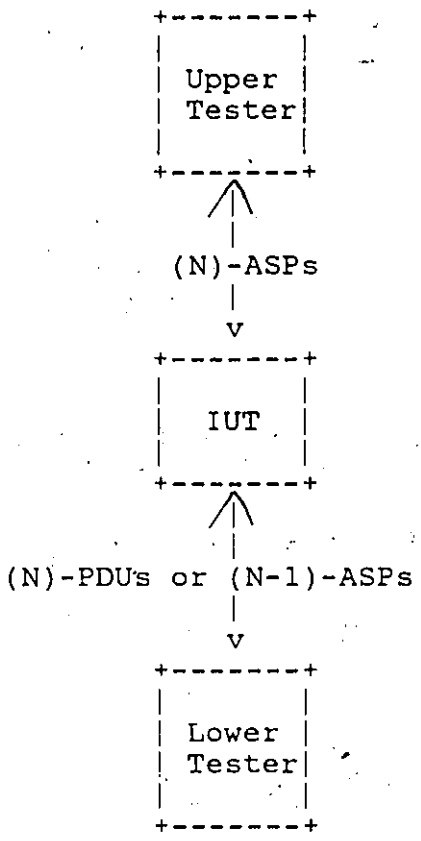


Figure 7: The LS Test Method.

The Local test method is a typical "in house" test method where both the service interfaces are accessible.

(2) Distributed Test Method (D): In this method the points of control and observation are at the service boundaries above the entity (entities) under test and

at the opposite side of the (N-1)-Service Provider from the (N)-Entity. The upper tester is at the upper service boundary and the lower tester is at the opposite side using the (N-1) service access point (see Figure 8).

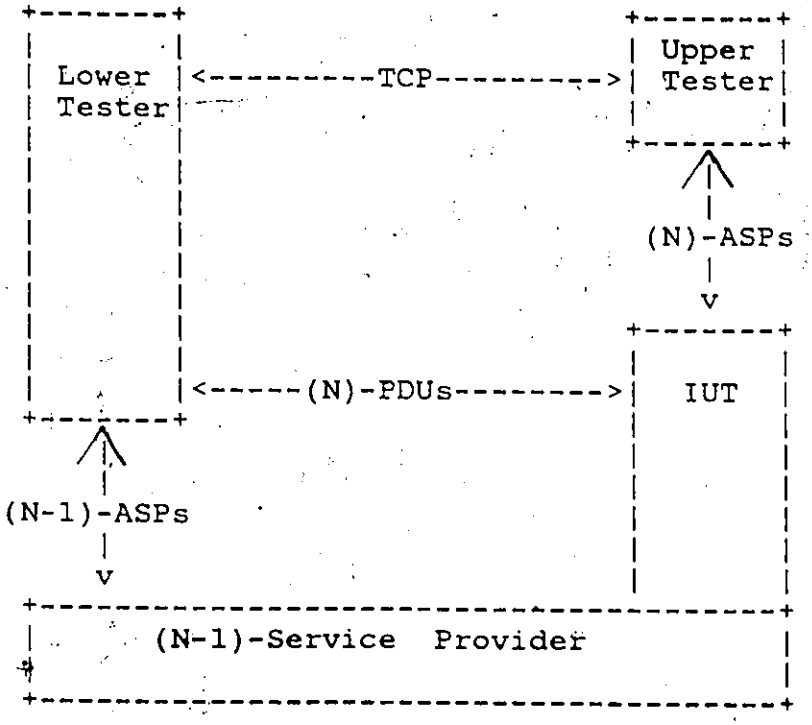


Figure 8: The DS Test Method

A Test Coordination Procedure (TCP) is used to coordinate test execution actions by the upper and lower testers. The essential distinguishing characteristics of this test method is that the points of observation and control are distributed over two different locations.

(3) The Coordinated Test Method (C): The Coordinated test method is an enhanced version of the Distributed 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. There is only a single point of control and observation at the opposite side of the (N-1)-Service Provider from the entity (entities) under test. This test method is shown in Figure 9.

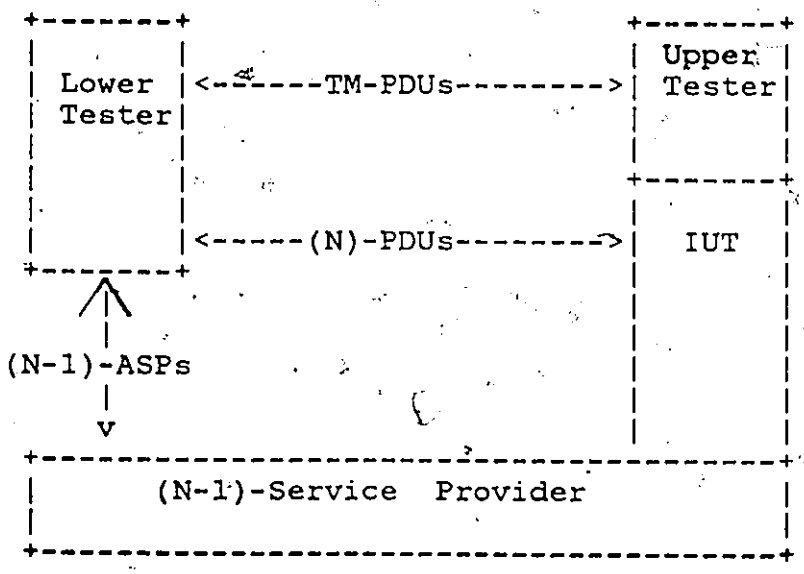


Figure 9: The CS Test Method

The distinguishing characteristic of this method is that there is only one SAP, although an upper tester is used. The upper tester interface to the IUT is inaccessible. In this case the upper tester is usually implemented by the IUT manufacturer.

(4) The Remote Test Method (R): In this method the points of observation and control are on the opposite side of the (N-1)-Service Provider from the entity (entities) under test. The test events are specified remotely in terms of ASP's and PDU's. Some requirements for test coordination may be implied or informally expressed in the abstract conformance test suite, but their realization would be implementation dependent (e.g. driver intervention from the IUT, etc.). This test method is illustrated in Figure 10. The test coordination procedure (TCP) and the upper tester are implementation dependent.

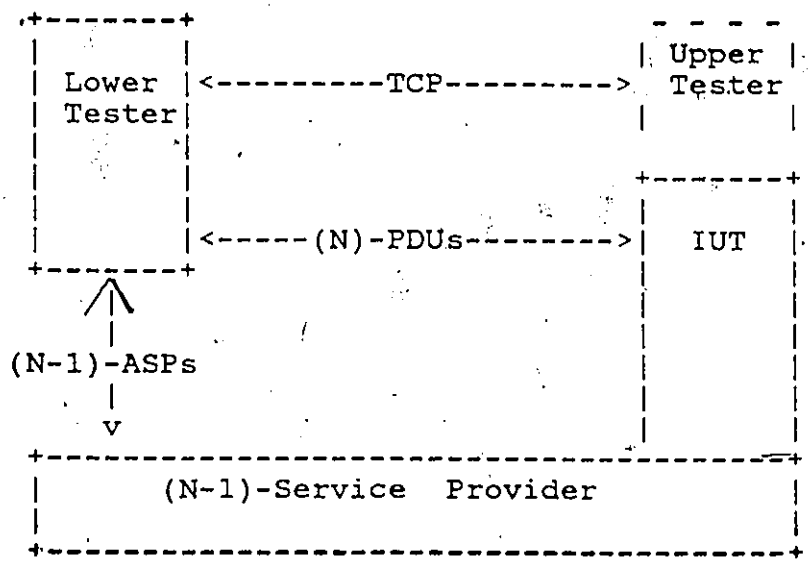


Figure 10: The RS Test Method

(5) Relay Test Method (Y): This test method is applicable while testing relay systems from one subnetwork to

another. There are two abstract test methods under this category:

(a) Loop - back Test Method (YL): In this method there are two points of observation and control on the same side of one subnetwork (see Figure 11). For connectionless protocols the PDUs are looped back within the second subnet and returned to the second point of observation and control. In connection - oriented protocols the looping back can be from the relay or the second subnetwork.

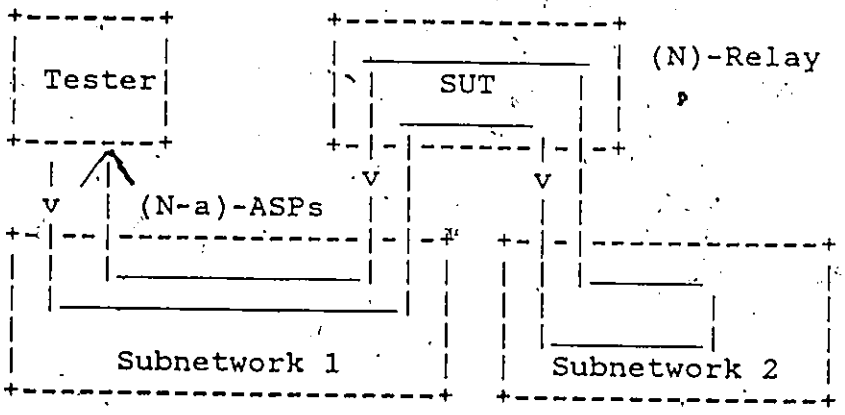


Figure 11: Loop - Back Test Method (YL)

(b) Transverse Test Method (YT): In this test method the points of observation and control are on opposite sides of the service provider from the (N)-Relay, one on each subnetwork (see Figure 12).

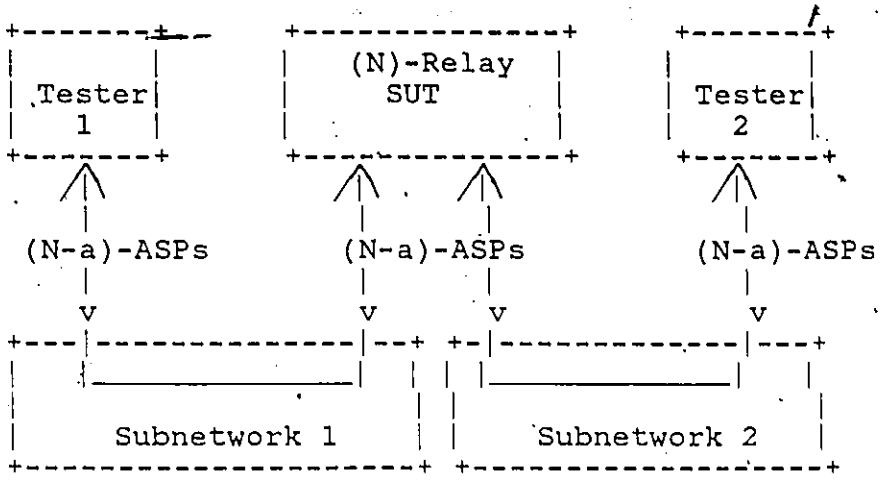


Figure 12: Transverse Test Method (YT)

Apart from the Relay Test Method, the other four test methods can be subclassified according to single (S), multiple (M) or embedded (E) layer IUT.

(S) Single-layer Test Methods (LS, DS, CS, RS): This test method is applicable for single layer IUTs. The test events are specified in terms of (N)-ASPs above the IUT and (N-1)-ASPs and (N)-PDUs below the IUT.

(M) Multi-layer Test Methods (LM, DM, CM, RM): In this test method all the layers of a multi-layer IUT is tested as a whole. The combined allowed behaviour of the multi-layer implementation must be known, without controlling or observing any of the inter-layer boundaries within the IUT.

(E) Embedded Test Methods (LSE, DSE, CSE, RSE): In this test method the testing of a layer is done embedded within a multi-layer IUT. The abstract test suite specifies 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+n), abstract test cases for testing layer (N+i) shall include the specification of the PDUs in layers (N+i+1) to (N+n) as well as those in layer (N+i). The DSE test method is shown in Figure 13.

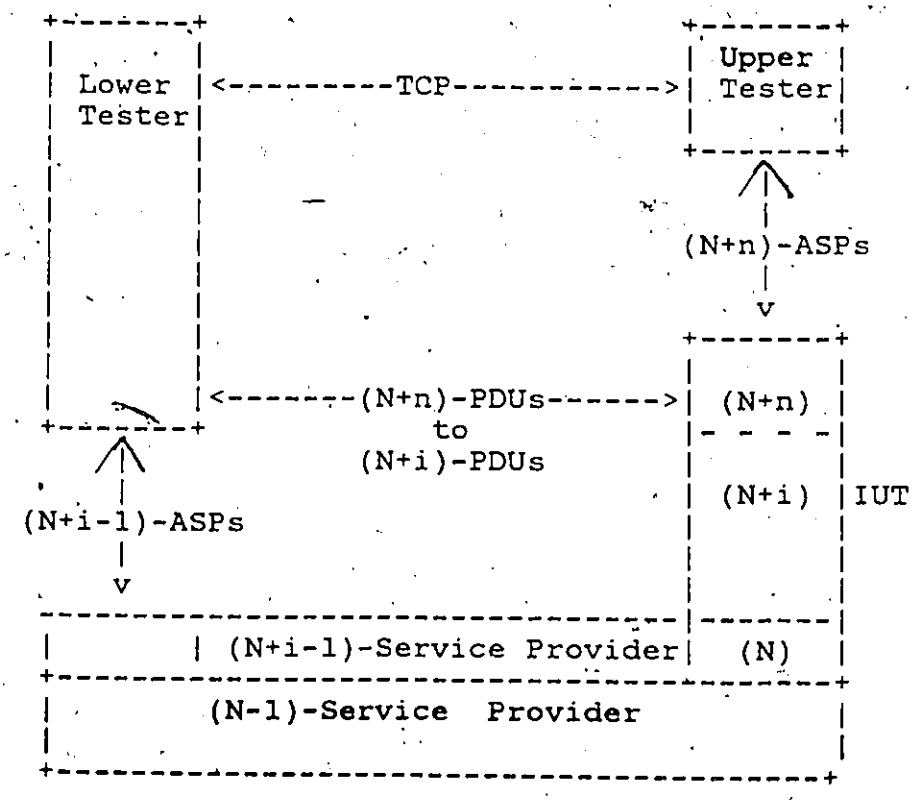


Figure 13: The DSE Test Method

1.3.2 Conformance Requirements and Conformance Statements

The most appropriate test method for a particular IUT is dependent on the specific protocol layer as well as the IUT's characteristics (service interface accessibility, etc.). Conformance statements of the IUT (IUT manufacturer's claims) are checked against the Conformance requirements during the static conformance review. At the end of this review, an appropriate test method is selected (depending on the Conformance statement) and further dynamic conformance testing may be undertaken. During testing only those tests that are appropriate to the IUT as claimed in the conformance statements, are allowed to execute.

The conformance requirements of an implementation claiming conformance to a standard/ recommendation are:-

(a) mandatory requirements - these must be implemented and satisfied during the testing of the IUT;

(b) conditional requirements - these must be satisfied during testing only when the conditions set out in the standard/ recommendation apply;

(c) options - these are protocol features in the standard/ recommendation that can be selected to suit the implementation. Any requirements on which the options depend or which depend on the options must be satisfied during testing.

For example, according to CCITT X.25 1984, essential facilities such as registration packets, are mandatory requirements; additional facilities such as delivery confirmation, can be either conditional requirements or options.

1.3.2.1 Static Conformance Requirement (SCR)

Provisions which define the requirements of sets of capabilities are placed in OSI standards/ recommendations to facilitate interworking. These provisions are called the static conformance requirements. The SCR forms the kernel set of requirements in the protocol standard/ recommendation to be implemented by a conforming IUT. An example of a proposed standard X.25 Static Conformance Requirements for the packet level can be found in [N639]. An example of a mandatory conformance requirement for X.25 DTE at the Data Link layer is the frame structure for modulo 8 operation of the send/ receive sequence numbers. The extended mode of operation (modulo 128) in LAPB is an option.

1.3.2.2 Protocol Implementation Conformance Statement (PICS)

The IUT is always to be tested against all mandatory requirements given in the SCR. In addition, the DTE manufacturer submits a Protocol Implementation Conformance Statement specifying in a tabular form the mandatory and optional features that have been implemented. In some

cases, the test organization may develop the PICS in consultation with the manufacturer. The PICS is used during testing to select tests appropriate to optional features. As well, it is checked for consistency against the SCR during the Static Conformance review.

The PICS should distinguish between the following categories of information:-

(a) Capabilities of the protocol implementation both at the broad level of grouping functional units and options, and also at the detailed level of ranges of protocol parameters and timer values supported. Capabilities which are not explicitly stated to be mandatory in the SCR are implicitly optional.

(b) Information related to the optional and conditional static conformance requirements for multi-layer dependencies. For example, Packet layer information is usually required during testing of the X.25 Data Link Layer.

(c) Other information which has to be specified (e.g. to assist testing) but which is not related to conformance requirements as such. For example, information regarding access to service boundaries, preferred test method, procedures for DTE initiated actions, etc.

An example of PICS for X.25 Data Link level is given and briefly explained in Appendix B. A proposal for X.25 Packet level PICS can be found in [N640].

1.3.3 Specification and Implementation of Tests

The conformance test method is an abstraction of the testing architecture. The test architecture is a concrete module level description of the test equipment required and their interconnections. Similarly, the test suite specification is an abstraction of the concrete test suite implementation. Therefore, the process of test suite implementation is the translation of the abstract test cases into a real executable test language. The details of the particular test implementation language depends on the testing equipment and the test driver.

1.3.4 Test Execution and Reporting

Test execution is the step in conformance testing which produces the final result (verdict) about the conformance of the IUT to the standard. In order to produce consistency of test results, the test execution step has to be specified in terms of the testing method used, test driver used, claims in the PICS and the characteristics of the SUT. For example, in X.25-DTE Data Link layer conformance testing the preparation before test execution will include checks on the physical connection (Physical layer). During test execution the test driver should detect and report any failure in the

Physical layer while the higher layers are being tested. This is important for assurance of consistency of test results for the the higher layers.

TEST REPORTING consists of summary reports and detailed error reports. The summary report of conformance testing is a combination of results obtained during the static and dynamic tests. The cutcome of each individual test case is analyzed to produce this summary report. The final verdict of conformance testing is obtained by comparing the summary results with the SCR and PICS.

The detailed report of conformance testing provides the detailed traces of each test case execution. These reports also contain diagnostic information. The detailed report is useful in determining the reasons for test failure or nonconformance.

1.4 TEST SPECIFICATION LANGUAGES

Abstract test suite is specified using an abstract test notation. In specifying test cases the use of a Formal Description Technique like LOTOS has been suggested [Brink86]. The ISO study committee on conformance (SC6 and SC21) has developed an interim test notation TTCN, pending agreement on the applicability of a more formal notation (FDT). The Tree Tabular Combined Notation (TTCN) has, in the meantime, gained wider acceptance and standard

conformance test suites have been specified using this notation [Kanun86,CasSi86].

The test notation TTCN serves the following purposes -

(a) to provide a common reference for assessing other notations;

(b) to promote a standard syntax and semantics for evaluation of tests and to examine the problems arising in test suite design;

(c) to provide a basis for the translation of tests into other notations or into executable languages;

(d) to be used to develop trial test suites.

A test suite contains test group which contain test subgroups, containing test cases. Test cases consist of test steps which consist of test events. In this hierarchy, TTCN is used to specify test cases and below. In this thesis TTCN is used to specify a comprehensive suite of test cases for the Data Link layer of X.25 DTE. A summary of the syntax and semantics of TTCN is presented in the next subsection. A more detailed description of TTCN can be found in Part2: Annex D to N909 (San Diego revision) [ISO909].

1.4.1 Tree Tabular Combined Notation (TTCN)

As stated above, TTCN is an abstract test specification language. Test suites written in TTCN has three parts:

a) Declarations - the alphabet of the events to be used in the test

b) Dynamic part - tables containing trees of behaviour

c) Constraints part - tables giving the ASP and PDU parameters.

The declarations contain lists for the service access points (SAPs), for example, [L] is SAP at the lower tester and [U] is SAP at the upper tester. Declaration tables are also used for ASP or PDUs and Timers. An example of a TTCN PDU declaration table is shown in Figure 14.

PDU Declaration		
PDU : DISC (DISConnect)	Restrictions on use : Always used with poll bit set	
Protocol Control Information		
Parameter Name	Range of values	Comments
ADdress (AD)	'H01, 'H03	
Poll bit (P)	'B0, 'B1	

Figure 14: Example TTCN Declaration table

In Figure 14, the declaration consists of the PDU name (DISC), restrictions on use which are significant for testing, and protocol control information. The protocol control information contains a list of PDU control parameters (address and poll bit setting) and the range of

acceptable values. Here the PDU address parameter may be 01 or 03 in hexadecimal. The poll bit may be 0 or 1 in binary.

The dynamic part contains tables having a heading for test reference and purpose followed by behaviour trees. Behaviour trees display successive events across the page and alternative events at the same indentation from the left (i.e., in the same column). An example of a dynamic TTCN table is shown in Figure 15.

Reference: Group/subgroup/ Test name Test ID : Test_case_id	Test Purpose: This is an example, showing temporal ordering of events.			Page 1 Of 1
Behaviour Description	Label	PDU/ASP Ref.	Comments	Results
Test-name [L, U] --------- L! <event_1> L! Start timer_1 U? <event_2> U? Otherwise	1	SO	send event receive	 Fail
U! <event_3> L! <event_4> ==> 1		S1	send go to 1	 Inconc.
L? <event_5> L! Cancel timer_1 L! Timeout timer_1				Pass Fail
Extended Comments: Temporal order of Events ---------->				

Figure 15: Example TTCN Dynamic Behaviour Table

Underlined tree names refer to the root of the current tree. A tree name may be followed by a list of SAP names enclosed in brackets, e.g., Test-name [L, U]. By convention, names of events to be initiated by a tester are prefixed by "!" and those possible for a tester to receive are prefixed by "?". These are event 'verbs' which are prefixed by the SAP name, for example, L? <event>, means "received at lower SAP the <event>". Events may be labeled in the "label" column. If an event is labeled, it is permitted to "go to" the choice among events at the level of the labeled event from a point lower in the tree (further to the right) to the labeled choice. This is done by using a "==>", for example, ==> <label>, mean "go to the event with the <label> above". A subtree itself can be used in place of an event name by prefixing the subtree name with a "+". For example, +L1STATE means an event denoted by the entire subtree "L1STATE". This provides modularization.

Curly brackets, ("{", "}") with a following integer enclosing a test step name or part of a tree are used to describe repeated behaviour. An event may be qualified by a Boolean expression by writing that expression in parenthesis ("(", ")") after the event name.

The "comments" column contains short remarks or references. The "results" column gives a verdict on the result of a test at all possible final events of a test.

This verdict is given with respect to the test purpose as PASS or INCONClusive or FAIL.

A number of predefined primitive events are provided for timer management. These are Start <timer_name>, Timeout <timer_name> and Cancel <timer_name>. In addition, the shorthand notation Timer <timer_name> specifies an inactivity timer, for example, waiting for some event to occur.

The 'Otherwise' predefined pseudo-event denotes any event at the SAP which may possibly be received by the tester, but for which no tree or subsequent behaviour is given.

The final section of the TTCN specifications is the constraints section. This section describes in detail the coding of parameters in PDUs and ASPs, indexed for reference in the dynamic behaviour section of the TTCN specification. Figure 16, is an example of a partial TTCN constraints table.

In Figure 16, the constraints on the DISC PDU is shown with reference to its use in the dynamic tables using the reference list names S0 and S1. In the usage of PDU DISC under the list name S0, the address is assigned the value of 03 hexadecimal and the poll bit is assigned a value of binary 0.

Constraints Declarations			
PDU DISC (DISConnect)			
List Name (Reference)	Field Name: Address	Field Name: P	Comments
S0	'H03	'B0	Address is value corresponding to 'A' in standard.
S1	'H03	'B1	

Figure 16: Example Constraints Table in TTCN

1.4.2 Use of Other Test Specification Techniques

Apart from TTCN specification technique, there are other specification techniques being considered by ISO as a standard test suite notation. LOTOS and Time Sequence Diagrams are two such notations under consideration by ISO.

1.4.2.1 LOTOS

LOTOS is a Formal Description Technique (FDT) used as a specification language for the description of communicating processes. In LOTOS, systems (for our purpose, test suites) are described in terms of behaviour of PROCESSES [Brink85]. A process communicates with its environment by means of interactions through GATES. For example, a service access point is a gate. The format of a process defined in LOTOS is:

```
PROCESS  <process-identifier> <parameter-part> :=
         <behaviour-expression>
ENDPROC
```

The choice operator ([]) is used in LOTOS to denote alternative choices of events. STOP is used to denote the termination of a process. EXIT is used to transfer control to the initial state of a successor process. The operator enabling operator (>>), is used for sequential composition. For example, B1>>B2 means process B1 terminates successfully and B2 begins. The notation N!CONreq means a connection request at gate (SAP) N and N_1!CONreq means connection request at gate (SAP) N-1.

Using this brief repertoire of LOTOS we now give an example of a process and a test case in LOTOS. A more detailed study can be found in [Brink85,Brink86].

The following is a specification of a process which sends a connection request, receives a connection confirmation and then enters the Data-phase or alternatively terminates the connection using a Disconnect request.

```
PROCESS Sender [CONreq, CONconf, DATAreq, DISreq] :=
            Connection-Phase [CONreq, CONconf]
         >> Data-Phase [DATAreq, DISreq]
ENDPROC
WHERE
     PROCESS Connection-Phase [CONreq, CONconf] :=
             CONreq ; CONconf ; EXIT
     ENDPROC
     PROCESS Data-Phase [DATAreq, DISreq] :=
             (   DATAreq ; Data-Phase [DATAreq, DISreq]
              [] DISreq ; STOP
     ENDPROC
ENDPROC
```

The following is an example test case in LOTOS. The purpose of this test is to test the successful entry into the Data transfer phase.

```
PROCESS test [N, N_1] : result
        := N! CONreq
         ; N_1! CONreq
           ; (   N_1! CONconf
              [] N_1! DISreq ; EXIT (Inconclusive)
             )
              ; N! CONconf
                ; N! DATAreq
                  ; N_1! DATAreq ; EXIT (Passed)
ENDPROC
```

In the above example, the SAP gates are declared as N and N_1 for the process test. This process returns a result as a parameter. The EXIT(Passed) is the point where the result value is returned as Passed. The indentation is used to show the sequence of events. However, this is done as a good programming style and is not a part of the LOTOS definition. Alternative choices, e.g. N_1! CONconf and N_1! DISreq, are at the same level of indentation. This is a very simple test case without complex protocol parameters or constraints. In the experience of the author, LOTOS is inadequate in specifying test cases with several parameters and PDU constraints.

1.4.2.2 Time Sequence Diagrams (TSD)

Time Sequence Diagrams are informal description techniques for showing sequence of PDU exchanges between the tester and the IUT over time. An exchange of PDUs at a given SAP is shown as a directed arrow with the PDU name (e.g. SABM---------->). The service interface is shown as a vertical line. The events are ordered along this line (service interface), downwards in ascending order of time. Figure 17 is an example of a X.25 Data Link Layer test case in TSD. More examples can be found in [DP88823].

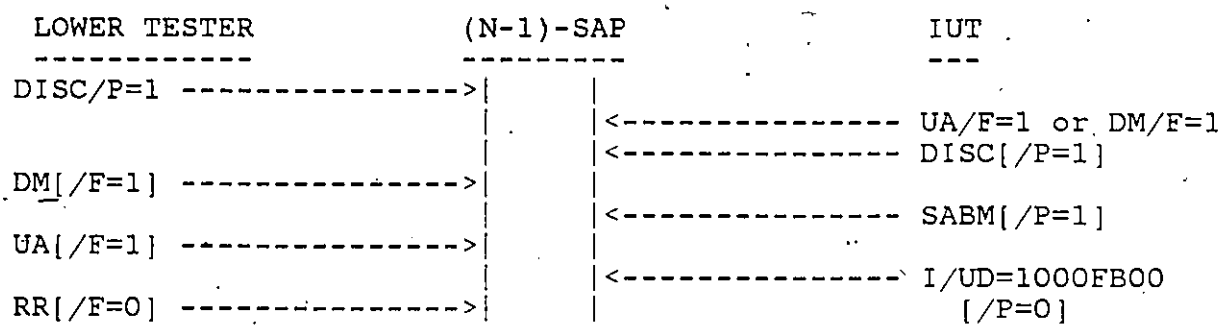


Figure 17: Time Sequence Diagram.. Initialization Step for Information Transfer Phase

In the example time sequence diagram of Figure 17, the link set-up dialogue for an active DTE is followed by an information frame by the IUT. The I-frame sent is actually a restart packet, which is acknowledged by the tester. The drawback of this notation is that the choices between alternative possible events can not be specified. This notation is also inadequate for specifying timer management

functions, like starting or cancelling a timer or timing out on a response.

1.5 Thesis Objectives

The objective of this thesis is to investigate the available test methods for X.25 DTE conformance testing. Using the proposed standard for conformance testing (N909), we develop a FSM directed test case design methodology. The main goal in this design process is to efficiently specify real X.25-DTE conformance test cases in an abstract test notation (TTCN). This work is proposed to ISO as a standard X.25 DTE conformance test suite.

1.6 Summary of Main Results

The main results of this investigation are as follows:-

(a) The Finite State Machine (FSM) representation of X.25-DTE is useful for test suite design and development.

(b) The original contribution of the author in the design of FSM directed test cases is the three step state transition specification of tests. The three steps are Initialization, Evaluation and Verification steps.

(c) The test language TTCN was found to be flexible as well as expressive as a test notation. The author along with members of the Protocols Research Group, University of

Ottawa, has made original contributions to ISO/SC21 for the development of TTCN as a test language, and to ISO/SC6 to illustrate the usefulness of TTCN as a test suite specification language.

(d) The TTCN Data Link layer conformance test suite for CCITT X.25 (1984) and ISO IS7776 is partially listed in Appendix C to Appendix N of this thesis. This suite is the first complete test suite to be developed according to international standards for conformance testing and the first specified in a standard test specification language (TTCN).

1.7 Outline of Thesis

In Chapter 2 we will discuss the design process for conformance test suites with examples from the design of our standard X.25 Data Link layer test suite. In Chapter 3 the FSM representation of X.25 LAPB protocol will be discussed and an augmented state transition table will be derived for specifying the state transition tests. The three step method for transition tests will be formulated in Chapter 4 and in the conclusion a comparison of this method with the W.P method will be made.

CHAPTER II

X.25 DTE CONFORMANCE TEST SUITE DESIGN

Based on the requirements and objectives of conformance testing introduced in the previous chapter, we now present the test suite design process. Theory of protocol testing is yet in the nascent stage, but an attempt will be made in this chapter to illustrate some of our findings using CSP (Communicating Sequential Processes) [Hoare85], as the basic formalism. Issues relating to test coverage and some test methods will also be discussed in this chapter.

Three related design activities of interest are software design, protocol software design, and test software design. We now briefly describe the last two activities.

In protocol software design the emphasis is on the service requirements of each layer. The software for the protocol layers are usually designed and implemented bottom-up from a specification of the lower layer service provider. The service specifications are derived directly from the protocol standards, in this case, X.25 standards [CCITX25c, ISO7776, ISO8208]. In addition, the testing and maintenance requirements for the software are taken into account.

The design of test software consists of two major stages, namely architectural design and detailed design. The architectural design is the preliminary design stage and involves the selection and refinement of the most appropriate test method. Completing the architectural design for protocol conformance testing consists of deciding on the test environment, defining service access points and choosing a method for observation and control at the upper and lower testers. On this basis the Remote, Distributed, Coordinated and Local test architectures have evolved as possible test methods for X.25 conformance testing [DP88821,Rayne85,N909]. During the architectural design stage, the emphasis is on the abstract representation of the system as relationships between modules. The detailed design stage involves transforming the system structure proposed by the architectural design stage into a procedural description of a software system.

2.1 Test Suite Design Process

The above discussion was on general software design and the use of hierarchical design decomposition techniques. We apply these principles to the design of a standard X.25 DTE conformance test suite.

The test suite design process is a series of refinements. The OSI test reference model states : 'only the external behaviour of Open Systems is retained as the standard of

behaviour of real Open Systems.' [ISO909]. Hence, the internal details of the system under test (SUT) are invisible to the tester. Only the external behaviour is specified in the standards. Thus external behaviour requirements form the basis for the test suite requirements. The steps in the standard test suite design process are given in Figure 18, namely:

(1) formulate test requirements,

(2) formally represent the specification of the IUT. In our case this is a FSM state diagram of the X.25 Data Link layer protocol;

(3) derive the FSM state transition tables augmented by proper, improper and inopportune PDU receptions at the DTE;

(4) derive formal abstract test suite specification (typically in TTCN);

(5) derive executable test suite implementation.

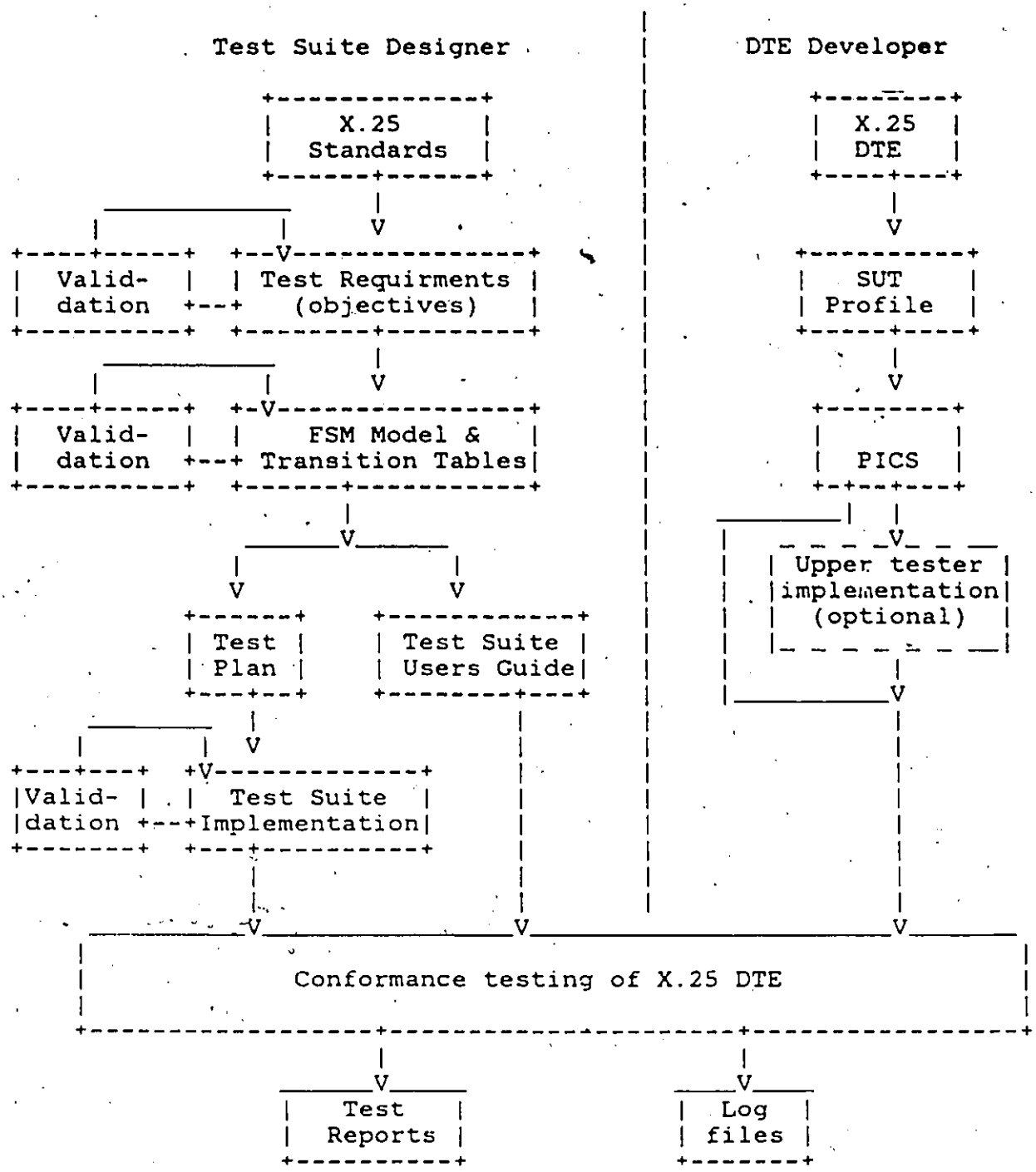


Figure 18: Conformance Test Suite Development. The correspondence between test suite development and the testing processes.

Formulating the test suite requirements is the first step in the design of a standard X.25 DTE test suite. The test suite requirements comprise the set of objectives to be realized by the test suite designer. From a system testing viewpoint [Myers79], the objective is to compare the behaviour of the System Under Test to its original objectives as specified in the protocol standards. But these standards also allow several implementation choices for the SUT. This suggests that the set of test design objectives be decomposed into (i) tests for mandatory requirements and (ii) tests for optional requirements.

The test suite requirement can be viewed as an abstraction of the actual test plan. The test plan specifies the test suite in terms of individual groups of tests and their functional requirements. After the formulation and validation of the test suite requirements, the second step is the selection of a protocol specification technique by which the behaviour of the SUT can be formally specified. In our case the Finite State Machine (FSM) specification technique for the X.25 protocol has been adopted. The State Transition Table is derived using the FSM model of the X.25 protocol. Figure 18 on page 41, shows the interrelation between these stages of test suite development, as well as a division of roles of test designer and DTE Developer.

Following the specification of the state transition tables, the test plan is the fourth step in the design process. An example of a standard X.25 test plan can be found in the U of Ottawa, X.25 DTE Conformance Test Plan [Lamon86b]. The test plan uses formal (TTCN) or informal notation to describe the test cases under each test group. The test plan also partitions the test groups according to functional modules.

The fifth and final step is the test suite implementation and execution. The implementation of the test suite from TTCN to an executable language is a straightforward translation process. The implementation is checked against a standard conforming IUT (reference DTE). The execution of the standard test suite against a IUT produces the final test results.

2.2 Structure of Standard Test Suite

The test suite structure can be viewed from several angles. A test event is the smallest element of the test suite structure. A test event is an indivisible unit of test specification at the lowest level of abstraction of the specification (e.g. sending or receiving a single test PDU). A test step is a subdivision of a test used to modularize test cases and are constructed from test events. For example, a sub-tree of DTE behaviour is a test step which may be shared by more than one abstract test case. At the

next level is a test case which is constructed from test steps to achieve a specific test purpose. A set of related test cases form a test group. A test subgroup is a subset of a test group. A test suite is the the largest member of this hierarchical structure. A test suite is a complete set of test groups necessary to perform conformance testing for an IUT or a layer within an IUT.

Another way of defining a test suite structure is in order of application during execution of the test suite [Hornb86]. Accordingly, the test suite is structured as indicated in Figure 19. The primary decomposition in the case of X.25 test suite is according to layer - i.e., a data link layer group and a packet layer group.

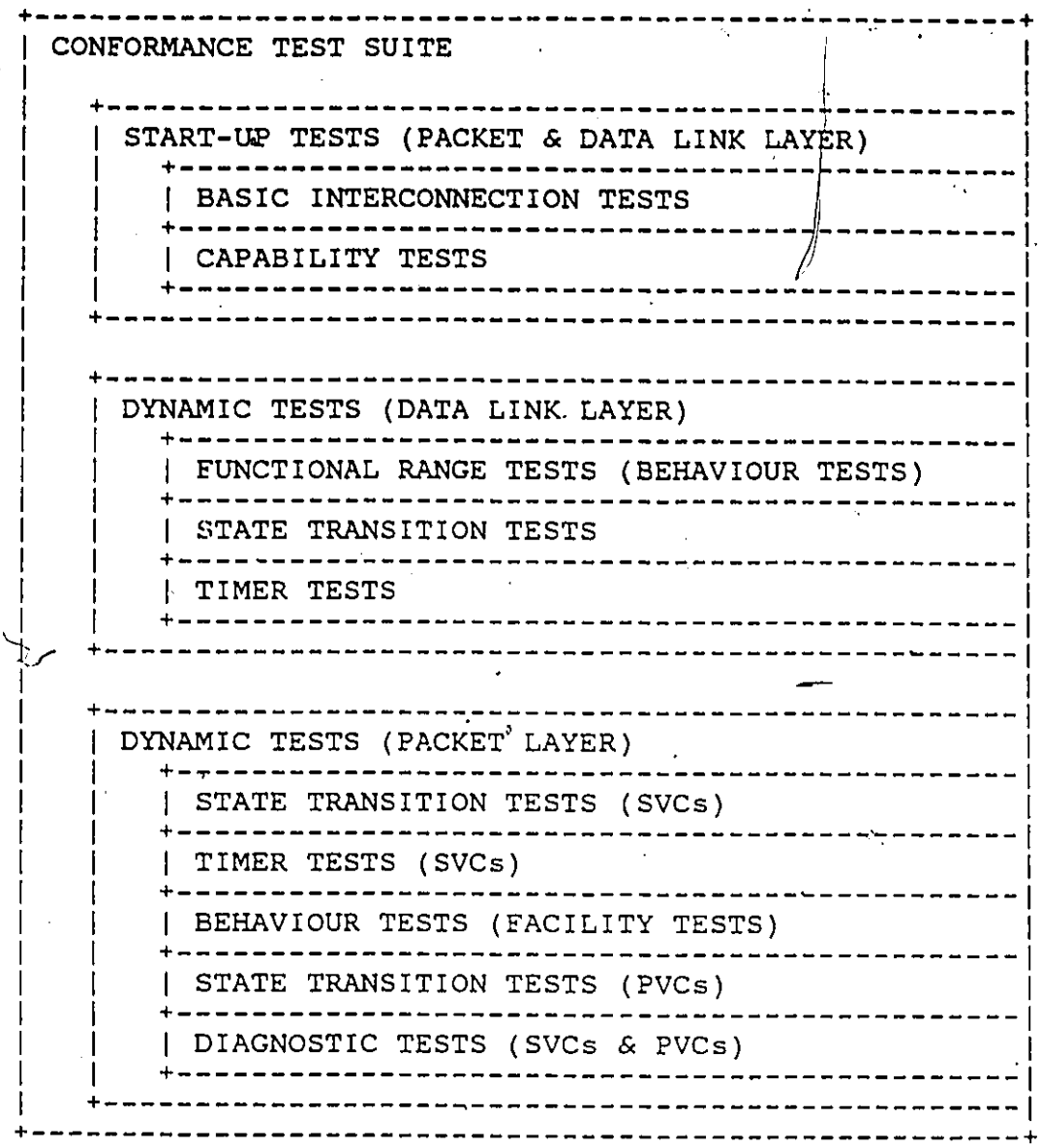


Figure 19: Test Suite Structure

2.2.1 Basic Interconnection Testing

This test group is designed to detect severe cases of nonconformance before trying to perform thorough testing. These tests provide limited testing of the main features in

a standard/ recommendation, to establish that there is sufficient conformance for interconnection to be possible. Basic interconnection tests are standardized in a separate test suite, which can be used on its own or together with a conformance test suite.

2.2.2 Capability Tests

This test group checks for the presence or absence of optional features/ facilities and also checks their validity with respect to the SCR and the PICS. The Capability Tests enable efficient selection of behaviour tests to be made for a particular IUT. Tests for capabilities which are optional in the standard/ recommendation and are not claimed to be supported by the IUT are automatically cancelled during subsequent test sessions.

2.2.3 Functional Range Testing

These tests are a subgroup of the Behaviour Test Group. These tests are intended to provide as thorough testing of an implementation as is practical, over the full range of dynamic conformance requirements specified in a standard/ recommendation for various protocol parameters and timers.

2.2.4 State Transition Tests

In our specification of conformance tests, the finite state model is used. The State Transition tests provide a thorough testing of an implementation to cover every possible transition of the IUT from one state into a new state as

defined according to a state transition table representation of the X.25 protocol (see Appendix A for state transition table at the data link level). The state transition tests are designed to place the IUT in a particular state, apply the test PDU, verify the observed response, and confirm that the IUT has entered the expected next state. In cases where the correct response is no response, the test case attempts to verify that no response was made by the IUT.

The State Transition tests are a subgroup to Behaviour Tests and in the case of X.25 Packet level these tests are further subdivided into subgroups for Switched Virtual Calls (SVCs) and Permanent Virtual Circuits (PVCs). Tests in each of these subgroups consist of tests for responses to all proper PDUs, tests for IUT response to improper PDUs, and tests for IUT response to inopportune PDUs (see section 3.2 item 1 for definition of these terms).

2.2.5 Timer Tests

These test group determines whether or not the IUT times out in specific, normal operating situations in such a way as to conform to the X.25 standard/ recommendation for DTEs.

2.3 Test Consideration: Nondeterminism in IUT Behaviour

Protocol specifications allow the DTE developer some implementation choices which can be viewed as resulting in nondeterministic IUT behaviour. For example, the active or passive nature of IUT [Sheri86] may be characterized by nondeterministic OR of test events during link setup and disconnect at the Data Link Layer [Kanun86] of X.25 implementations. Other similar examples of nondeterministic IUT behaviour can be found in [Bochm82].

Communicating Sequential Processes (CSP) by C. A. R. Hoare, deals with the mathematical approach to the study of concurrent and communicating processes. In CSP the nondeterministic OR (||) can be used to define a process which exhibits a range of possible behaviours; however, the environment of the process does not have any ability to influence or even observe the selection between the alternatives [Hoare85].

Hoare in CSP [Hoare85] and Milner in CCS [Milne80] have formally defined such nondeterministic behaviour for communicating systems. Hoare defines it as follows:

Definition : If P and Q are processes, and || is a nondeterninistic or operator, then we define -

P || Q (meaning, P "non-deterministic or" Q),

to denote a process which behaves either like P or like Q, where the selection between them is made, without the

knowledge or control of the external environment. In the case of protocol testing, P and Q are two distinct behaviour trees and the external environment is the test suite used for conformance testing. Hoare points out that nondeterminism is useful for maintaining a high level of abstraction in description of the behaviour of physical systems and machines.

In the testing of IUT behaviour, the process P or Q can be atomic events or even entire test cases. There exists nondeterminism in the occurrence and sequencing of possible valid events in a test. All such valid behaviour, over which the tester has no control and which are selected by the IUT via some internal event [Milne80] or arbitrarily by the system as a whole, have to be specified and represented in the test cases. For example, at the X.25 Data Link level, the possible link reset by the IUT on entering the error state is decided by some internal event transparent to the tester. The tester only observes the outcome of this internal event. So, the choice here is between observing the continuation of the error state or receiving PDU for Link reset.

IUT behaviour for a particular test execution or for a deterministic test case, can be represented as a trace of events. A trace is a possible sequence of events in a test case. Hoare has defined trace as follows -

Definition : A trace of the behaviour of a process is a finite sequence of symbols recording the actual events in which the process has engaged up to some moment in time.

For example, to say that P and Q have potential traces

<a, b> and <a, c>

respectively, means that these are possible valid sequence of events in these respective processes.

A process can be pictured as a tree with unlabelled nodes and the events as branches [Milne80]. Nondeterministic OR can be represented in a picture by a node from which emerges two or more unlabelled branches [Hoare85] or branches with the same label [Milne80]. Figure 20 a, b, c are all pictorial descriptions of nondeterministic processes represented as event trees, where P and Q are sub-trees.

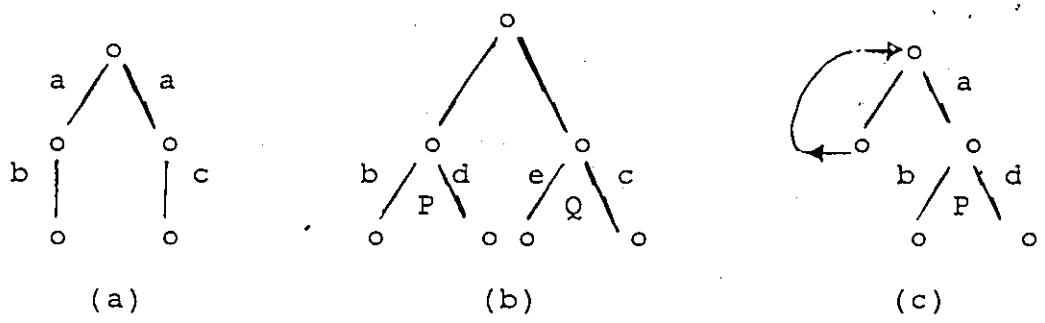


Figure 20: Nondeterministic processes

It is obvious that while (a) and (b) in Figure 20 represent finite trees, (c) is an example of an infinite or potentially divergent tree [Hoare85]. We do not allow

infinite nondeterministic paths in test case design. If such paths are present, it is considered to be an error in the test case implementation, because it leads to a test case which may never terminate. All test cases should terminate after a finite time. Such cases of divergence are regarded as errors in process calculli [Hoare85,Milne80].

2.3.1 Laws of Nondeterministic Processes

The laws for nondeterministic processes have been well formulated in process algebra by Hoare and Milner. They are:-

Law 1:	P || P = P	(idempotency)
Law 2:	P || NULL = P	(nullity)
Law 3:	P || Q = Q || P	(commutativity)
Law 4:	P || (Q || R) = (P || Q) || R	(associativity)
Law 5:[1]	P . (Q || R) = P . Q || P . R	(distributivity)

The last law of distributivity states that the nondeterministic OR between Q and R is prefixed by the process P. This law is not valid for recursive processes. In fact, Milners CCS does not allow distributivity. For example in LOTOS [Brink85], which is based on CCS, the trees in Figure 21, model distinct behaviours -

The reason is that when the environment (or the tester in our case) wants to perform event A , it does not resolve the choice in process Q. A nondeterministic selection is made internally in the process Q between A with possible future

[1] This law is used only in CSP and not in CCS.

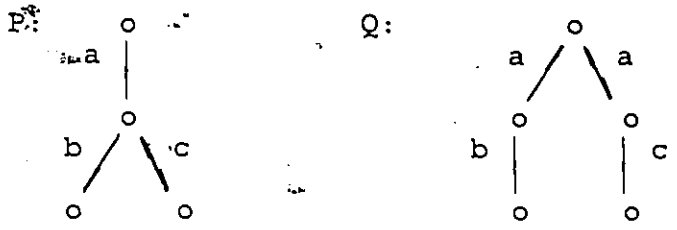


Figure 21: Tree behaviour for nondeterministic processes

B or A with possible future C. In process P the nondeterministic selection between B and C is made by the process P after the event A has occured and not before. Both P and Q have identical traces, viz. {ab, ac} but Q can deadlock whereas P cannot under the given set of traces. The deadlock in Q will occur if the left branch is chosen by the process on input a, then input c is applied or if the right branch was chosen on input a, then b is applied. Thus P and Q are considered to model different behaviour.

Because CCS and LOTOS reject the distributive law, it can no longer take languages (sets of strings) as the behaviour of machines. Hence, observation and comparison of traces is considered insufficient to test the behaviour of the IUT. In CCS and LOTOS the laws of Observation Equivalence are used to prove equivalence (symbol is $\approx$) between processes -

$p \approx q$ is always true (assume)

$p \overset{}{\approx}_{k+1} q$ iff $\forall s \in \Lambda^*$ such that:

(i) if $p \overset{s}{\Longrightarrow} p'$ then for some q', $q \overset{s}{\Longrightarrow} q'$ and $p' \overset{}{\approx}_{k} q'$;

(ii) if $q \stackrel{s}{\Longrightarrow} q'$ then for some p', $p \stackrel{s}{\Longrightarrow} p'$ and $q' \approx_k p'$;

and $p \approx q$ iff $\forall\, k >= 0$, $p \approx_k q$,

i.e. $\approx = \bigcap_k \approx_k$.

This concept of observation equivalence is not an equivalence relationship in the classical sense. For example, $p \approx q$ and and $q \approx r$ does not necessarily imply $p \approx r$. Informally this would mean that to prove absolute observation equivalence, we need to investigate all possible behaviours of P and Q under the s - experiment. Now, since s ranges over Λ^* ($s \in \Lambda^*$), this would mean exhaustive testing for all possible behaviour. Such testing is not practical [RapWe85].

In our test suite design we use observation equivalence in a much weaker sense. We consider nondeterminism in IUT behaviour, but the range of our s ranges over some typical values in Λ^*.

2.4 Testing Architecture and Test Method

The general architecture for OSI conformance testing was presented in Chapter I. The abstract test architecture is closely related to the test suite design and also to the protocol layer under test. The abstract architecture defines the points of observation and control as a first step in the test design process.

The (N - 1) abstract service primitives (ASP) or the (N) protocol data units (PDU), can be observed and controlled by the tester from several different points, directly or remotely. But all these different architectures are not applicable in every test environment. The ISO recommended architecture for X.25 testing is the Remote architecture (RS or RM) or the coordinated architecture [ISO909]. In some cases, where the upper interface is accessible, the distributed ('D') architecture could also be used. However, in most instances the 'D' architecture is not applicable to X.25 testing, because direct or indirect control of (N)-ASPs may not be possible. Moreover, the (N)-ASPs are not clearly defined for the X.25 services in the standards [ISO7776,ISO8208,CCITX25c]. To avoid test synchronization problems we have used the Remote and Coordinated test methods for out standard X.25 conformance test suite.

The question of TEST SYNCHRONIZATION is one of the major issues in selecting the test architecture and designing the test suite. In earlier work on test synchronization by Sarikaya [SarBo82b], the remote testing method was used where the test responder had no knowledge of the lower tester implementation. In their testing architecture, the lower tester and the test responder are distributed over two sites and are only synchronized through interactions with the IUT. However, later the ISO methodology [ISO909] established the 'C' and 'D' architectures in which the test

responder (or upper tester) and the lower tester have explicit knowledge of the events at the other's SAP. In the latter case synchronization problems may arise when one of the service interfaces are inaccessible or hidden. In the coordinated single layer architecture the test suite can be designed to have no synchronization problems if the following conditions are satisfied -

(a) The communication between the tester and test responder can be guaranteed over lossless FIFO channels.

(b) For all send events at the either the tester or the test responder, the sending entity sends a synchronization message to the other entity. These are the test management PDUs (TM-PDUs).

In our design of X.25 DTE conformance test suite, we have used the remote architecture with synchronization messages for test starting operations (in the DTE initiated test cases) or the coordinated architecture with the use of an additional test link to exchange the TM-PDUs [Lamon86a]. Figure 22 and Figure 23, shows the test architecture for X.25 testing.

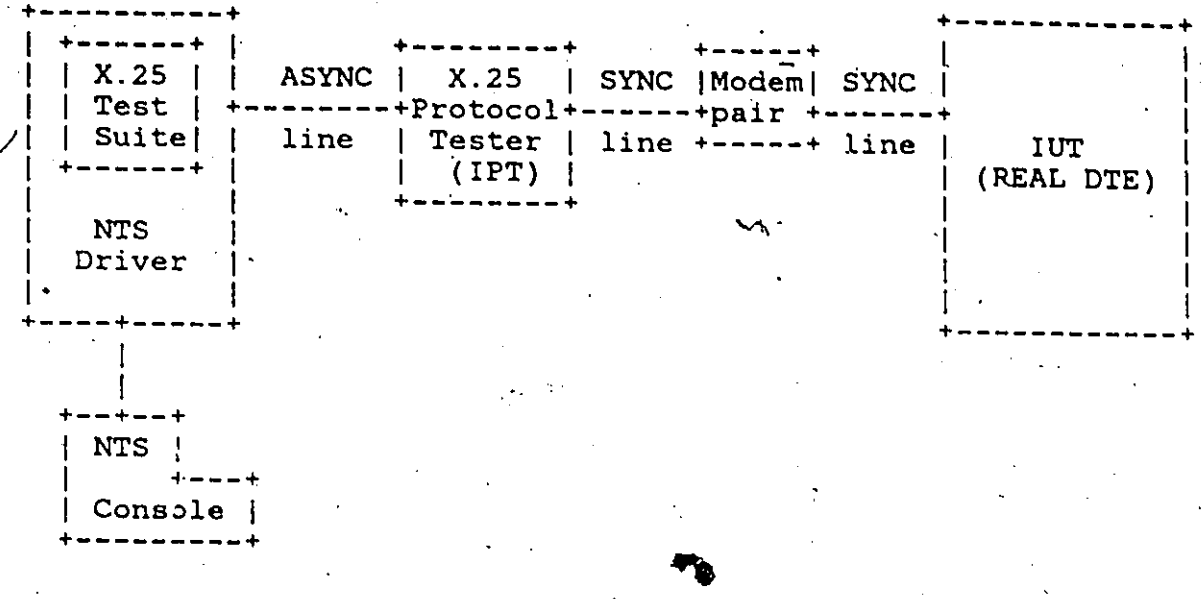


Figure 22: Remote Test Architecture

2.4.1 Test Implementation and Execution

The design and specification of X.25 DTE conformance test suite is followed by actual implementation and execution. The test suite is implemented in a proprietary testing language developed for the Network Testing System (NTS), supplied by Bell-Northern Research (BNR) [Hornb86]. The test suite validation is done by running the suite against both reference and real DTEs over a testing facility at the University of Ottawa running NTS.

The Network Test System (NTS) provides all the facilities for developing and executing X.25 test cases. The NTS includes a X.25 Interactive Protocol Tester (IPT), as shown in Figure 22 and Figure 23 and a Test Language Driver (TLAN).

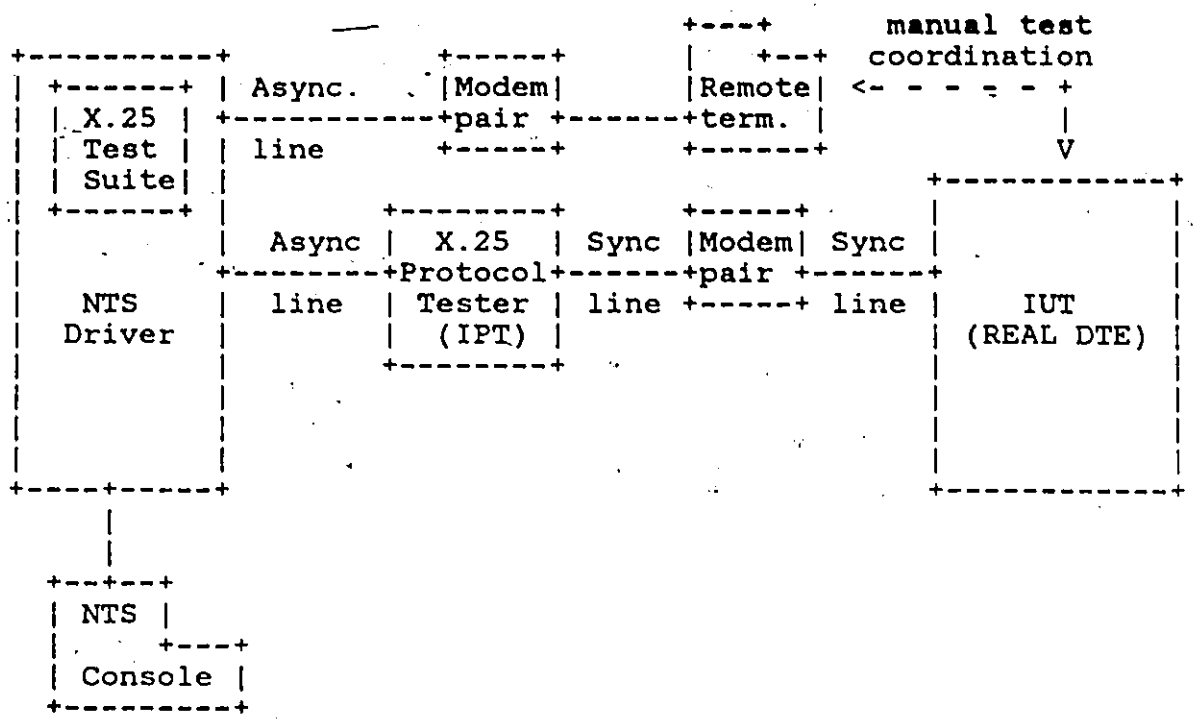


Figure 23: Coordinated Test Architecture

The IPT runs on Northern Telecom's DPN, Data Network System[2] Access Module, while TLAN runs on a Microvax[3] computer. This system has evolved from an older protocol test system developed by Bell Northern Research and used to test the DATAPAC network [Weir78].

The capability of IPT's X.25 data link and packet layer interactive commands, combined with TLAN's programming power and system management facilities, is a system well suited to X.25 DTE test suite development. Test cases coded in TLAN

---

[2] DPN is a trademark of Northern Telecom Limited.

[3] Microvax is a trademark of Digital Equipment Corporation.

drive the IPT to send test protocol data units (PDUs) to the IUT in a controlled fashion to an initial condition appropriate for each particular test case. PDUs received from the DTE are decoded by the IPT and forwarded to the TLAN driver for checking and processing. Correctness of behaviour is judged on response received from the IUT according to the programmed logic of the TLAN test case. If unexpected, syntactically invalid, or inopportune responses are observed, a test case is reported to have failed, indicating nonconformity to the test case purpose. If no test case during the execution of the entire test suite fails, then the DTE is declared to be in full conformance with the test suite.

In the test implementation and execution of X.25 test suite, the correctness of behaviour is judged solely on PDU exchanges at the DTE-to-DCE interface. Many test cases involve actions initiated by the user layers above the IUT. For example, in DTE switched virtual call initiation cannot be controlled by stimulus at the DTE-to-DCE interface. To test such cases involve actions to be initiated by the DTE. The test tool provides a user-prompting terminal which helps coordinate DTE-initiated actions. When a test case requires the IUT to initiate an action, the test tool writes an user-prompt message at the remote terminal (see Figure 23 on page 57). The DTE operator performs the requested action and acknowledges the prompt. TLAN then checks the IUT response and reports the test result automatically.

NTS has a set of test management facilities. Profile and subroutine files can bind hundreds of test cases into a single test suite. NTS runs each test case in the profile sequentially. Test case execution is retried several times in case of failures. During the execution, report lines are written on a file and trace data is included for failed test cases. The outcome of each test case and a summary report is generated automatically during test suite execution. Failed test case trace data can be analyzed to decide what further action should be taken.

To facilitate the analysis of TEST REPORTS , the test plan document is used along with the PICS and SCR. The test plan contains the precise description of each test case in the test suite. The failure of a test is analyzed against the test purpose and the PICS. For a more detailed analysis, the test specification (in TTCN) can be checked against the test purpose. All this is done to determine the reason for the failure of any test case. Note, however, that a test case failure does not necessarily imply that the IUT is nonconforming.

2.4.2 TEST COVERAGE CONSIDERATIONS

One of the important considerations in test suite design is the Coverage considerations. This relates to the effectiveness of the test suite or the testing method, in discovering errors in the implementation. In general, test

coverage criteria [Myers79] can be applied to White-box Testing or Black-box Testing. Although conformance testing is considered to be Black-box testing, some of the criteria of White-box testing also apply here.

In White-box testing, knowledge of the source code is available to the test suite designer. The coverage criteria for White-box testing has been given by Myers [Myers79] as:

(a) Statement coverage (node coverage) - This involves writing enough test cases so that every statement is executed at least once. Statement coverage is a weak criteria, because executing every statement does not necessarily mean checking the program logic, or covering all the decision conditions (predicates). Moreover, appropriate test data for executing all the program statements may not be available i.e., there may exist nonexecutable statements.

(b) Path coverage - This involves writing enough test cases to cover all the program paths at least once. But, Path coverage is not a viable testing goal. This is because most practical programs have a very large number of feasible paths. Particularly, programs with loops may have restrictively large number of distinct paths [RapWe85].

(c) Decision coverage - Decision coverage, also called Branch coverage, is a stronger coverage than Statement coverage. In this criteria enough test

cases are written so that each 'true' and 'false' outcome of each decision is executed at least once. Another way of stating is that each branch direction must be traversed at least once. It has been proved [Myers79] that Decision coverage satisfies Statement coverage. An extension of this criteria is the Multi-way Decision coverage where the goal is to exercise each possible outcome of all decisions at least once and invoke each point of entry at least once.

In the case of protocol conformance testing the source code of the implementation is in general not available to the test designer. What is available, however, is the detailed program specification in the form of the protocol standards and the PICS. Some of the White-box testing criteria can be applied once the IUT is modeled as a FSM. The states of the FSM are the nodes and the transitions are the branches of the program under test. The states of the FSM are functional groups of code in the IUT and may possibly have multiple entry point. But the details of these are transparent to the test suite designer. In this case the states are nodes with converging transitions [Gill62] and the behaviour of the IUT at different state's entry point can be nondeterministic. As we have pointed out earlier, our test cases take into account nondeterminism in IUT behaviour. It is noted, however, that not all cases of nondeterminism are as a result of unknown entry points.

In Black-box testing the internal details or the source code of the implementation are not available to the test suite designer. Only the functional specification in terms of input - output relationship is observable as the external behaviour of the system. Myers [Myers79] has treated Black-box testing on the basis of 'equivalence class' partitioning.

(d) Equivalence class - Input domain is partitioned into equivalence classes such that one can reasonably assume that a test of a representative value of each class is equivalent to a test of any other value. Equivalence classes may overlap.

From the above criteria of Black-box testing the following goals for testing can be derived:

- Identify interesting conditions to be tested.
- Form a minimal set of test conditions by using equivalence partitioning.
- Testing time is best spent in different equivalence partitions so:
  - Identify the equivalence class
  - Define the test cases.

Identifying equivalence classes is largely a heuristic process. Study of data communication protocols show that the set of proper, improper and inopportune PDUs in a given state is an equivalence class (see section 3.2 item 1 for

definition of these terms). Particular PDU parameters, for example, the PDU length or the sequence numbers also form equivalence classes. The following heuristics can be applied to test the functional range of these parameters [Myers79,ISO909,Rayne85]:

(a) If the input parameter is a range of values, e.g. "the retry counter can be from 1 to 10" then identify two classes:

1 < retry count < 10	valid case.
10 < retry count	invalid case

(b) If the input specifies a set of values, e.g. at the Data Link layer of X.25 for the modulo 8 operation, the address can be {01 03}, then form two equivalent classes {01}, {03} valid and {more than 03} invalid.

(c) If there is a range of values for a particular parameter than test one value inside and one value outside this range. For example in checking the window size in X.25 this criteria could be applied.

(d) Boundary value analysis are those test cases that focus on the edge of the input - output set. This test criteria is applied in testing the maximum PDU size. See for example, Appendix D, functional range test for Data Link Layer N1 bits (maximum frame size).

(e) Equivalence partitioning in the case of the Timer tests is done over the functional range of the timer.

That is the valid range of the timer is one class, and at least one timer value test is performed corresponding to the invalid class. This equivalence partition test verifies that the timer implemented in the IUT has the appropriate time out value, and is functional over the range of this time value.

Here we have presented the basic test coverage criteria for conformance testing. A similar list is given in the test requirements for X.25 DTE conformance test suite which which was generated by the author of this thesis [Kanun85][4]. The concept of equivalence classes is useful in X.25 DTE testing for partitioning the dynamic test groups by the type of the PDU exchanged. Branch coverage is a useful coverage criteria for the state transition tests for X.25 DTEs.

2.5 Other Test Methods

The present trend in conformance testing has been influenced by experiences in protocol testing over the past few years [Rayn82,SarBo82a,UraPr83,RafAn83,Rayne85].

Sarikaya and Bochmann [SarBo82a], reported their experience in test sequence generation for protocols modelled as finite state machines. Their method of test sequence generation was derived from techniques used in the area of testing sequential circuits [Kohav78], and

---

[4] Part of the BNR DTE Conformance test suite project undertaken at the U of Ottawa.

experiences in testing software design modelled by finite state machines [Chow78]. Thus, the following methods of test sequence generation were applied by Sarikaya and Bochmann:-

TRANSITION TOURS - A test sequence is called a transition tour starting with the initial state if it covers all the transitions in the state table at least once. The machine under test is assumed to be minimal, strongly connected and fully specified (these terms are defined in [Gill62]). This method detects errors in output function but does not detect all transfer errors (errors of next state function). The applicability of transition tours for practical protocols is further limited by its inability to adequately handle FSMs with loops (i.e. in cases where the initial state and the final state are the same and can be reached by a sequence of transitions in a loop).

THE W-METHOD - This method is based on finding a characterization set of a minimal finite state machine called a W-set [Gill62]. The W-set is an input sequence that can distinguish between the behaviour of every pair of states in the FSM. A characterising sequence is a sequence of interactions which can be used to verify an unique state reached in the FSM. A characterising sequence or diagnosing experiment belongs to the W-set. P is any set of input sequence such that for each transition from state Ai to

state Aj on input x, there are input sequences p and x.p in P such that p takes the machine from the initial state into state Ai. This means that P consists of all PARTIAL PATHS in the FSM [Chow78]. The test sequence generated by the W-method consists of the concatenation of P and W, i.e. P.W is the test sequence.

The assumption about the machine under test is that the machine is minimal, has a fixed initial state and is strongly connected. This method lends itself to more practical application than the transition tour method, but the main draw back is that the W-set is not guaranteed to exist for incompletely specified machines. For Remote and Distributed test methods the W-method may have synchronization problems [SarBo82b]. The P.W sequence is applied starting with the initial state, hence a return to the initial state is necessary after each sequence. If the sequence consists of an improper PDU exchange leading to an error state then there must be at least one transition from the error state to the initial state, otherwise the test method will deadlock.

CHECKING SEQUENCES (DS SEQUENCE METHOD) - The method of Checking Sequence is designed for machines that have a DISTINGUISHING SEQUENCE (DS). An input sequence for which the output sequence produced by the machine is different for each initial state is called a distinguishing sequence.

The **CHECKING SEQUENCE METHOD** is the generation of a test sequence from the concatenation of

(a) the initial sequence to bring to initial state;

(b) the DS sequence for each initial state;

(c) the transition checking sequence which checks all individual transitions in the state table.

It is assumed that the finite state machine under test is strongly connected and has a distinguishing sequence. The study by Sarikaya and Bochmann shows that there exists no unique DS sequence for the X.25 protocol [SarBo82b]. Hence the checking sequence method is not applicable for the X.25 protocol.

On the basis of these methods for test sequence generation some tools for automatic test sequence generation have been developed. For example, the tool by Rafiq and Ansart (VADILOC and GAST) [RafAn83] can generate test sequences for the above mentioned three methods if the FSM is specified. Another approach for user guided test sequence generation uses PROLOG based logic specification for the FSM and then generates test sequences from the specifications [UraPr83]. Our method of test case generation, though not automatic at this time, uses a different approach from the above mentioned methods. Our method will be discussed in Chapters III and IV.

CHAPTER III

FSM ORIENTED TEST SPECIFICATION METHODOLOGY

There are a number of approaches to specifying protocols, two of these are: the state machine approach [Bochm78,Bochm80,Zafir80a,ZafBr83], and the high-level programming language approach [Hugas84,BraJo78,CheHo81]. Brand and Zafiropulo [ZafBr83] find the FSM model for describing protocols the least powerful, but relatively easy to analyze. In the FSM model of describing communication protocols, the system including all the component processes and interconnecting channels is modelled using states and transitions. Each pair of communicating processes is connected by a full-duplex FIFO channel (the channel is modeled by a separate FSM). There are no assumptions about the time a process spends in a state or the time that the message spends in a channel. The number of distinguishable states in a state machine for most practical protocols increases drastically with any increase in complexity of the system. To reduce the total number of distinguishable states in the FSM model, the hybrid model using programming language concept, such as conditional execution, has been introduced [Bochm82]. Another technique to deal with large FSM models is to group states according to protocol phases.

This latter technique is used in this chapter for specifying the X.25 Data Link layer LAPB protocol.

3.1 Overview of FSM Test Specification

In this thesis, we attempt to formalize and augment the DTE's perspective of X.25 Data Link level. State Diagrams for the Packet level of X.25 can be found in [CCITX25c]. The formalism adapted for this attempt is based on the finite state transition model and is particularly useful in generation of state transition tests as part of a conformance test suite for X.25 DTE conformance testing. As a first step, we define the state transition diagrams for the LAPB protocol. Next, we discuss the use of a validation tool to check the connectivity and completeness properties of our finite state machine model. Then we specify the state transition tables as an aid to test specification. The state transition tests are derived using the augmented state transition tables. This is done using a three step method of test specification. This three step method is discussed in the next chapter.

3.2 State Transition Diagram for X.25 Data Link Layer

The state transition diagrams shown in Figure 24, Figure 25, and Figure 26, specify the behaviour of a DTE at the data link level of X.25 and are based on the 1984 version of X.25 [CCITX25c, ISO7776].

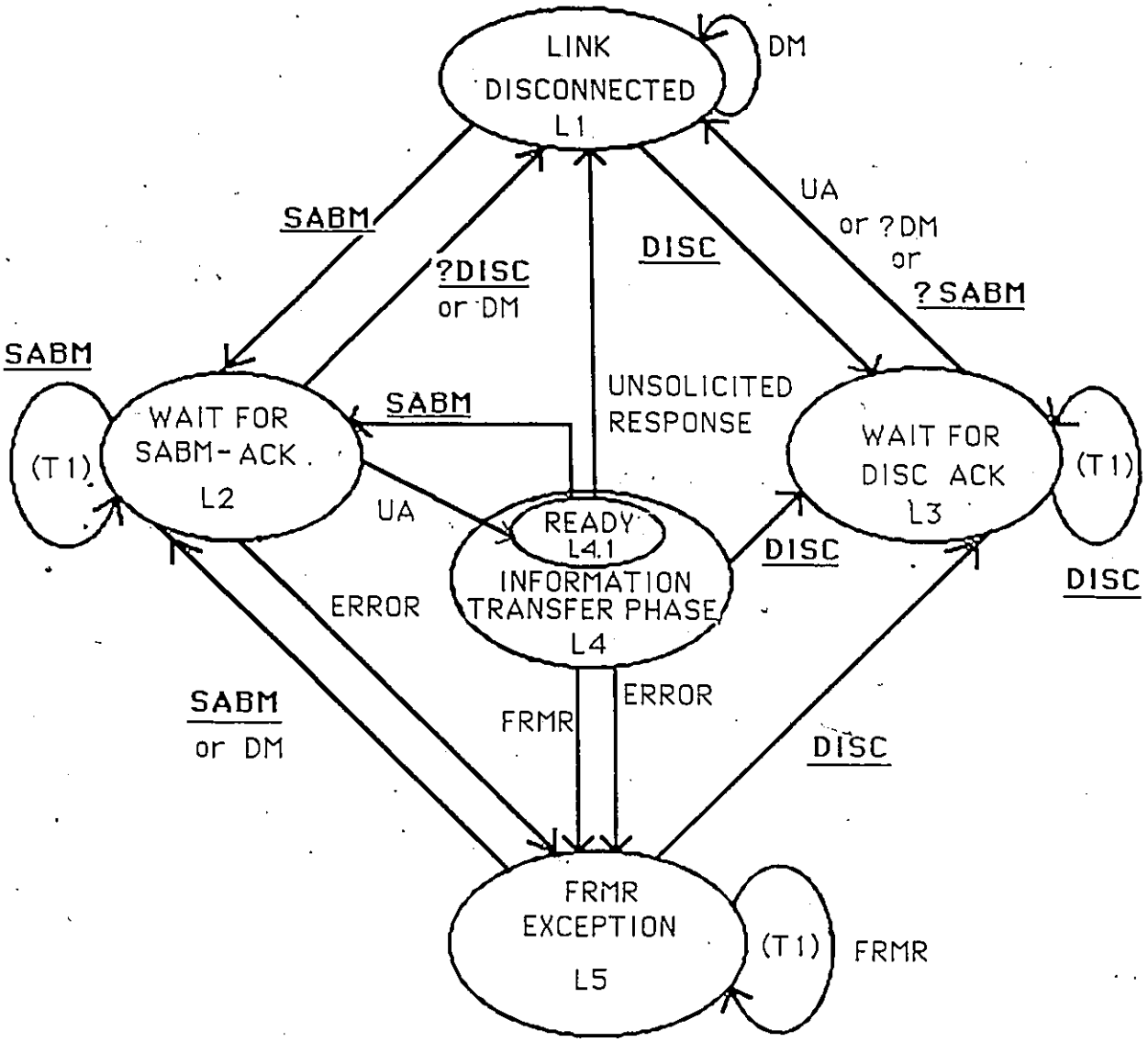


Figure 24: LAPB Link Set-up and Disconnect Phase

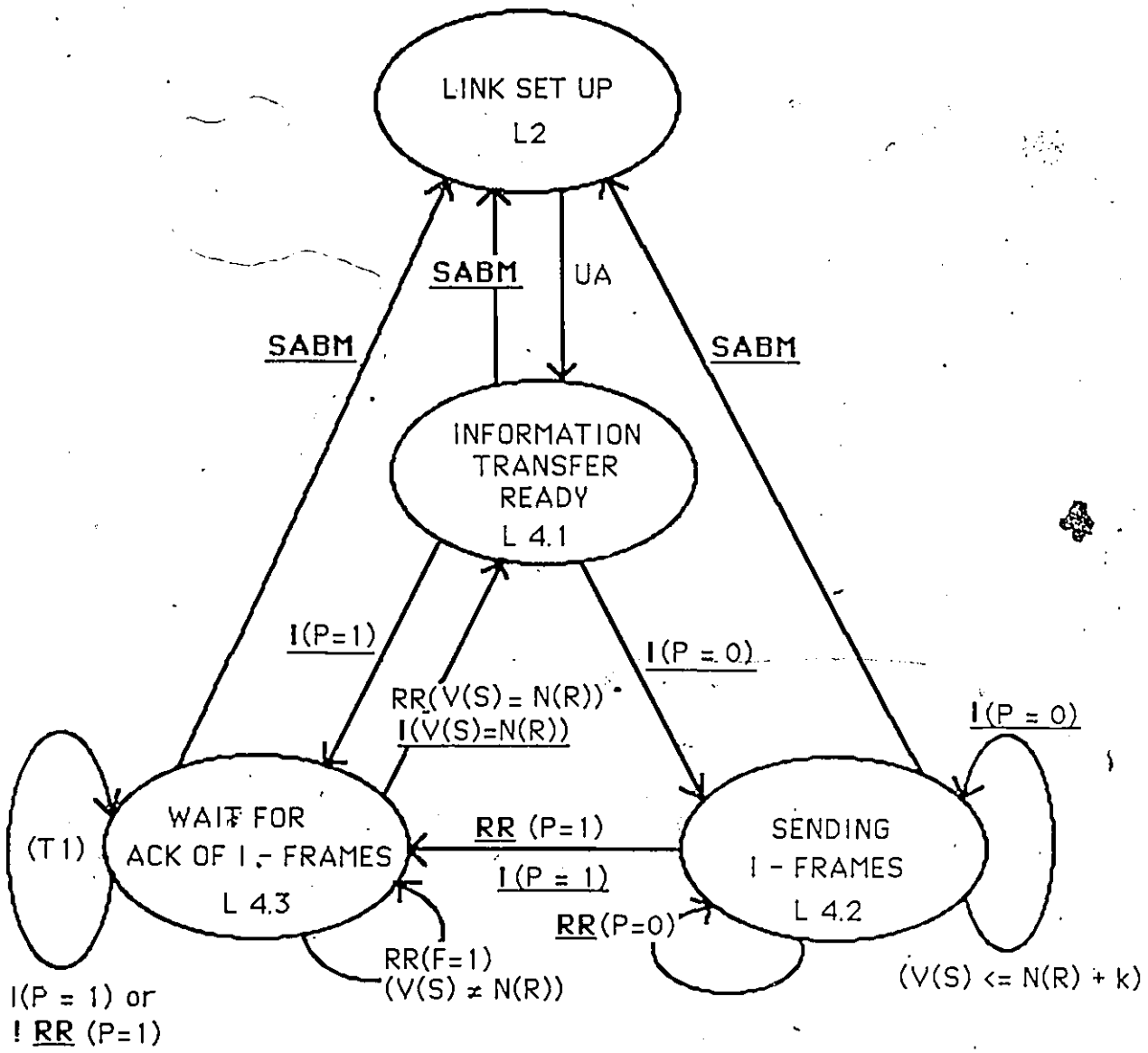


Figure 25: LAPB Information Transfer Phase

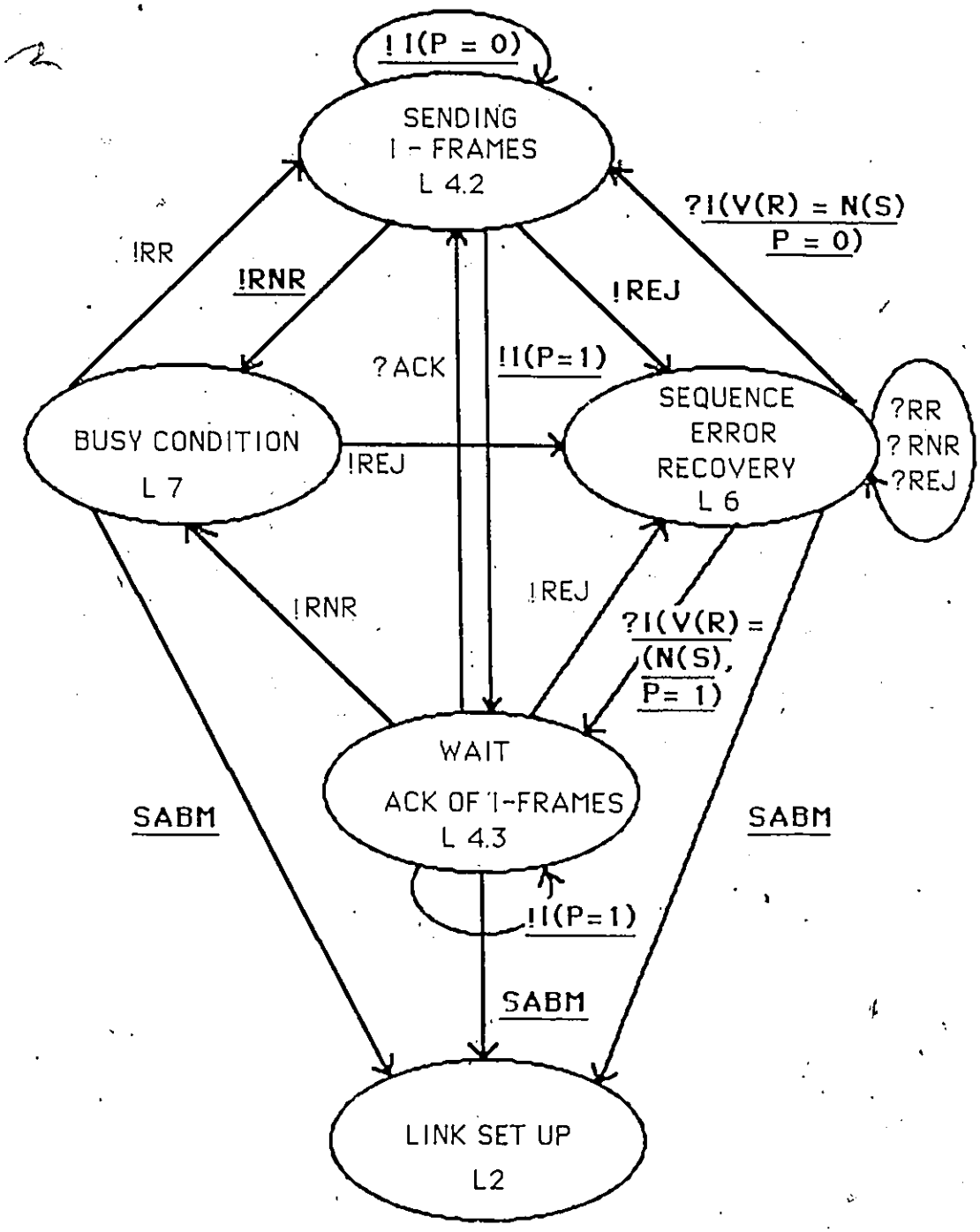


Figure 26: LAPB Error Recovery Phase

State transition diagrams have been used for specifying the related HDLC class of protocols [BocCh77] and the physical layer in X.21 [Gouda85], among others. During the derivation of test specifications from these X.25 state transition diagrams, the following cautions should be observed:

1) DTE AND DCE STATE - the DCE states are not shown separately, i.e., only the DTE states are shown. The DCE in this case is the tester, perhaps at a remote site, as in the remote single/multi-layer testing architecture [Rayne85,ISO909]. The tester follows the DCE state model only in the 'ideal case', that is, for generating proper protocol data units (PDU's) which are syntactically correct and are acceptable at the current state of the data link layer. Note that improper (syntactically incorrect) and inopportune (syntactically correct but having arrived at an irrelevant state) PDUs are not shown in the state transition diagrams. However, they are part of the input set from the tester to the implementation under test (IUT).

The tester can send improper PDUs to the IUT to verify its response to such PDUs. The inopportune PDUs are implicit in the state transition diagrams in the sense that only proper PDUs are shown at each transition. For X.25 DTEs the allowed responses to improper PDUs is dependent on the state of the DTE. These responses are specified in the actions (A)

row of the state transition table. Inopportune PDUs are discarded in all the X.25-DTE states, and in some cases it is followed by a link reset. The expected output from the IUT at any particular time is a subset of all proper PDUs defined by the FSM representing the funnctionality of the IUT. Reception at the tester of any PDU outside the range of the expected outputs corresponds to a test failure for the IUT.

2) GUARDS ON THE STATE TRANSITIONS - the state transitions in this model assume direct coupling as in [Bochm80], for local interactions of an IUT through the N or N-1 interface with the service user or lower service provider, respectively. The transitions may be enabled by an enabling predicate. For example, the predicate (V(S)<=N(R)+k) represents the condition that the send state variable is within the window, and the action I(P=0),(V(S)<=N(R)+k) would mean 'send an I-frame with the poll-bit not set (i.e., P=0) provided the frame sequence number is within the window'. Similarly, transitions in the event of a time out (shown as T1 in Figure 24 on page 70) are expressed with a suitable enabling predicate. The data link layer uses command or response PDUs. These are distinguished using underlined or normal type font, respectively. The '?' symbol is used to denote a reception and '!' is used for a transmission. When neither is specified this denotes both a reception and transmission.

3.3 Validation of Protocol FSM Representation

The next step involves validation of our FSM model for the protocol, to check its consistency and other properties. Several tools are now available for the validation of protocol specifications, for example [RafAn83,RaMor85,Vuong86]. These tools are based on reachability analysis of the state machine or Petri Net [Berth82] representations. Reachability analysis is performed to validate the protocol against potential errors like state deadlocks, unspecified receptions and nonexecutable interactions [Zafir80b]. In this thesis our aim for validation is not to find errors in the X.25 protocol but to check the consistency of our FSM model for the Data Link layer LAPB protocol with respect to the X.25 standards/ recommendations which are expressed in natural language. This FSM model was generated from the 1984 version of the X.25 standards/ recommendations [CCITX25c,ISO7776].

The tool that is used for validating our FSM model is VADILOC [RafAn83]. In this tool the user describes FSM via the state transitions. The syntax of this description is as follows :

<Event From-state Predicate To-state Action>

Here the Event is the test event, i.e. PDU send/ receive or timer event. From-state is the initial state for the transition and To-state is the final state. The predicate is a single condition under which the transition can be executed. When the predicate is absent the notation [] is used to specify the absence of any condition for the execution of that transition. The action represents the action performed by the execution of the transition. In the case of sending or receiving a PDU the action is the same as the event. The entire FSM is specified using a data base of such transitions.

Using this method the FSM shown in Figure 24 on page 70, is validated. To check the FSM's properties the following commands were used:

-> START-STATE STOP-STATE

This command checks the existance of a path from the start state to the stop state. This is used to check strong connectivity of the FSM.

>> STATE-1 STATE-2 ... STATE-N

This command checks the reachability of all states starting from the initial state. This property verifies that all states are reachable from the initial state.

<< STATE-1 .... STATE-N

This command checks if there are states from which it is not possible to reach other states. These states represent potential omissions in the FSM specification.

All these properties were successfully verified for our FSM of Figure 24 on page 70. However, the following property to detect deadlocks and nonexecutable transitions could not be checked because of a limitation of the tool. The tool fixes an upper bound on the channel capacity of two. This is very low and unrealistic for practical protocols. The command used is:

& STATE-P1 STATE-P2 N

where State-P1 and State-P2 are the initial states of two communicating processes and N is the maximum channel bound. For protocols like X.25 Data Link layer there exist loops. Thus, a low bound on the channel is unrealistic for the purpose of detecting deadlocks.

This tool also had the drawback of having to specify each predicate as a single condition and not an expression of the type If <condition[s]> Then Action, etc. Another drawback of test sequence generation via VADILOC and GAST is the requirement of the construction of a FSM for the DCE, if it is different in behaviour from the DTE. The current version of VADILOC does not consider the communication between two dissimilar FSMs, i.e. the DCE and the DTE are considered to

have similar behaviour if used for test generation by GAST. Thus, this tool was useful in verifying the connectivity properties but not for detecting state deadlocks.

3.4 Augmentation of State Transition Tables for Test Specification

The next step is to convert the state diagrams into the augmented state transition table (shown in Appendix A). The columns in the state transition table contain the initial states of the IUT. The states are labeled L1 to L7 and correspond to the states shown in the state diagrams. Note that state L4 can be subdivided into three inter-related states as in Figure 25 on page 71. However, to promote clarity in the state transition table, the information transfer state is shown in only one column. The rows correspond to the input PDU types (Proper, Improper and Inopportune PDUs). Each entry in the table gives the expected response of the IUT as a 5-tuple:

$$F = (A, S, D, T, C)$$

where, A = appropriate action(s) to be taken by IUT as specified in [CCITX25c, ISO7776],

S = next state after taking the appropriate action(s)

D = diagnostic code (if any) in case of a frame reject condition

T = test identification for easy reference and automated test selection

C = test group corresponding to expected test behaviour. The test group is categorized as (P) proper, (I) improper or (O) inopportune.

The actions can be further specified using options (e.g., ACT1 and ACT2) to distinguish between different types of IUT implementations. All such possible actions are considered to be valid IUT behaviours.

Multiple appropriate actions are specified in the action field. For example, active DTE's perform the Link Set-up procedure as soon as they enter the disconnected state L1. Prior to link set up an active DTE's may initiate link disconnection by transmitting a DISC(P=1) and going to state L3, or it might transmit an unsolicited DM(F=0) response to request the DCE to initiate link set up, or initiate link set up on its own by transmitting a SABM(P=1). All these possible alternatives are represented by 'ACT 1' in the state table. In this case, it is up to the implementor to specify all details of the link set up procedure in the protocol implementation conformance claim (PICS) (see Appendix B for details).

The normal link resetting procedure is shown as 'ACTION 2' in the state transition tables (Appendix A). The DTE has the option of resetting the link by transmitting a DISC(P=1) and going to state L3 or by transmitting a SABM(P=1) and going to state L2 or it may ask the DCE to initiate link

resetting by transmitting DM(F=0) and change state to L1. The test cases are designed to handle all these possible link resetting procedures.

The test identifier in the state transition table provides a unique test case identification to allow selection of this test during a test session. Since the test case corresponds to a transition from a given state, the state label appears as the first field of the test identifier. The remaining portion of the identifier consists of the protocol label, category of the test, and the test number.

The following example illustrates the use of the state transition table with reference to column one (L1), row one (DISC) of Appendix A. Consider the entry shown in Figure 27. The tester sends a DISC PDU with the poll bit set (P=1) to the DTE which is known to be in state L1. The action performed by the DTE is either a DM(F=1) response or ACT1 (as described earlier), both are allowed by the protocol. The next state is L1 if a DM response was sent or L2 or L3 depending on ACT1 (in the case of active DTEs). The test unique identifier is L1LAPB101 and the test category is (P) proper.

In the next chapter we describe in detail the design method used for generating the test cases. The test steps in the test cases are then expressed in the abstract test specification notation TTCN.

PDU FROM TESTER	PDU AT DTE	TYPE	STATE L1 (DISCONNECTED)
DISC	DISC(P=1)	A S D T C	DM(F=1)/ACT1 L1 / - - L1LAPB101 P

Figure 27: Example of test table entries

CHAPTER IV

DETAILED TEST SUITE SPECIFICATION

In this chapter we will develop a test case specification methodology based on the Finite State Machine model and the State Transition Table developed in the previous chapter. Using this method we generate detailed test specification in TTCN from the State Transition Tables. Since TTCN is an abstract test specification language, it is necessary to further translate the test suite into executable test implementation language. This translation from TTCN to a executable test implementation language is usually quite straightforward. For a discussion of the corresponding executable X.25 test suite, see [Horn86].

This method is applicable for deriving the part of the standard conformance test suite known as State Transition Tests, from the State Transition Table specifications. Each entry in the State Transition Table (the A,S,D,T,C tuple) translates into an unique transition test in our test suite. The transition tests are grouped by initial state and there are seven such groups. Each group is divided into subgroups of proper, improper and inopportune test cases. The Dynamic tests part of the standard conformance test suite for the

X.25 Data Link layer also include tests for the verification of the T1 and T2 timers. These tests are not specified by the state transition tables. The design considerations discussed in chapter II under test coverage are also used in these timer test cases.

4.1 Deriving Dynamic Tests from State Transition Tables

The State Transition Tables represent valid behaviour of the IUT for a given test PDU sent from the tester. All valid alternative behaviours are specified. For example, ACT1 in the State Transition Table represents all valid link set-up procedures. Active DTEs may initiate the link set up procedure as soon as they enter the disconnected state L1. Prior to link set up, some active DTEs may initiate link disconnection by sending a DISC(P=1) PDU and go to L3 state. The IUT may transmit an unsolicited DM(F=0) PDU to request the DCE (in this case tester) to initiate link set up. Alternatively, the DTE may initiate link set up on its own by transmitting a SABM(P=1). All these alternatives are represented by 'ACT1' in the State Transition Table. The TTCN test specifications implements these alternatives as possible valid IUT actions.

Test cases are derived from the state transition tables (Appendix A) using a three step procedure:

(1) Test Initialization Step

(2) Evaluation Step

(3) Verification Step

4.2 Test Initialization Step

Test Initialization step consists of bringing the DTE to the required test state. This step assumes that the FSM describing the behaviour of the DTE is strongly connected [Gill 62]. However, it should be noted that some DTE states like busy state (L7) or waiting DISC acknowledgement state (L3) cannot be guaranteed to be reached by any actions directed from the tester. In order to perform tests in these states, a DTE initiated test initialization step is required. How the DTE performs this initialization is implementation dependent. Alternatively, an upper tester (test responder) may be used as described in the DS (Distributed Single Layer) testing method [ISO 909]. The DS method provides greater control over the testing process at the cost of possible synchronization problems [SarBo82b]. We have adopted the RS testing method as outlined in [ISO 8882/1] The reasons behind the choice of this test method has already been discussed in chapter 2. The initialization step can be decomposed into two substeps -

(a) transfer to the root of the testing tree [Chow78], by use of a transfer sequence (reset) and,

(b) starting at the root, find a partial path "P" to the given state.

(a) The root of the testing tree is the disconnected state (L1) and the transfer sequence consists of [!DISC . (?UA | ?DM)]. Here concatenation (.) and OR (|) have the

usual significance. This transfer sequence becomes a TTCN sub-tree as illustrated in test step L1STATE in Figure 28, for initializing to the Disconnected State (L1 State).

```
L1STATE [L]
--------
    { L! DISC
        L! Start Timer T1
          L? Timeout T1 N2                    1

        L? UA (AD=3 & F=1)
          L! Cancel T1

        L? DM (AD=3 & F=1)
          L! Cancel T1

        L? Otherwise
            ==>1
```

Figure 28: TTCN Initialization Step for L1 State

(b) Once the IUT is at the disconnected state (L1), a minimal partial path, "P", can be chosen to bring the IUT to the required state. An algorithm for building a testing tree and deriving the partial paths in the tree is given in [Chow78]. For the L1 state the partial path "P" is an empty sequence.

4.3 Evaluation Step

The evaluation test step consists of performing a simple transition test, or timer test (transitions with time out predicate), or window rotation test (transition with state variable predicate). During the evaluation test step, the input sequence to the IUT in a given state can be from the

set of proper, improper or inopportune PDUs. Testing all the proper interactions is considered mandatory [ISO9091]. Testing improper or inopportune test cases can not be exhaustive. A higher priority is given for testing typical situations rather than testing for boundary value cases. The evaluation step consists of a single interaction (e.g. execution of a DTE state transition). This interaction consists of:

- sending PDU as test stimulus,
- verifying acceptable response (including no response if allowed in the state transition table specification).

4.4 Verification Step

The verification test step consists of verifying the IUT's state after the transition performed in the evaluation step. The next state is specified in the state table and the verification step reduces to a diagnosing experiment [Gill62]. A typical diagnosing experiment is shown in Figure 29, under subtree L5CHK.

```
L5CHK
-----
   L! RRC
      L! Start Timer T1
         L? FRMR (AD=3 & F=1)
            L! Cancel T1          Pass

         L? Timeout Timer T1      Fail
         L? Otherwise             Fail
```

Figure 29: Diagnosing Experiment for L5 State

In Figure 29, the RRC command PDU forces a FRMR response from the IUT. This is characteristic of the L5 state (Frame Reject State).

It was observed that the test cases obtained by the above method were very effective in judging the IUT's conformance to the protocol without requiring a very large test suite (e.g. total number of test cases were around 300). A similar test suite was also developed for the packet level of X.25 [Lamon86a].

4.5 Additional Standard Conformance Tests

These tests are required to cover instances of dynamic behaviour that are not covered by the state transition tests. The timer tests for the Data Link layer are instances of such test cases. This is because the actual time required for a transition is not adequately represented in the FSM model. The 1984 version of X.25 standards/ recommendations specify four timers - T1, T2, T3, T4. The timers T1 and T2 are mandatory whereas the other two are optional. Tests are included in the standard test suite to check the functional range of timers T1 and T2 (see Appendix D). As well, tests are included to verify that proper actions (e.g. retransmission of the last transmitted PDU) are performed if the timer T1 is allowed to time out. The latter tests are grouped as time-out tests. Finally, tests are also included to check the maximum retry count parameter

(N2 count) and the maximum frame size (N1 bits) for the X.25 Data Link layer. These test cases are about fifteen percent of the entire test suite, rest are the state transition tests.

4.6 Environmental Issues

In order to achieve the ultimate objectives of conformance testing, the overall verdict concerning conformance of an IUT must be independent of the testing environment in which testing takes place. There are a large number of factors to be considered to ensure independence of conformance test results from the test environment used; some of the more important ones are:

(a) Flexibility in the abstract test case specifications to take into account environmental factors. For example, the test driver's IPT may be capable of handling only a given maximum PDU size (because of buffer restrictions). Tests with longer PDUs must be cancelled in this environment, and the reason for cancellation should be reported.

(b) Flexibility in the specification of the test drivers which are used to run the necessary test suite. This means that an actual test driver (NTS environment in our case) should be flexible enough to account for different environmental conditions. The environmental constraint in our case (NTS) can be, for example, the

maximum number of ASIM ports handled by the driver. Also the test configuration used constrains the choice of abstract test method.

(c) Careful specification of the procedures to be followed in determining how the contents of the PICS are used. This is done, abstractly, by having standard PICS proforma and concretely, by encoding the PICS into test procedure guards. In the case of the NTS driver, the PICS are coded directly into the SUT file for automatic reference by the test suite during execution.

(d) Careful specification of the procedures to be followed by the test operators. An example is the procedure for manual test coordination in the Coordinated test method [Lamon86b].

CHAPTER V

CONCLUSIONS AND SUGGESTIONS FOR FURTHER STUDY

In this chapter we conclude this thesis on the specification and design of OSI standard X.25 test suites by assessing the effectiveness of our method for test case design by comparing it with another approach for generating test sequences. We discuss the limitations of some test notations. Finally, we suggest areas of further research in conformance testing.

5.1 BENEFITS OF METHOD

The standard test suite design method presented in this thesis has the following benefits:

(1) The FSM model for protocol specification is well developed and there are many tools for its verification. Hence, test suites can be correctly developed using the validated FSM specification for the protocol.

(2) The three step method for deriving state transition tests is easy to use and considers the important cases of proper, improper and inopportune test stimulus.

(3) Each test step in this method easily translates into a TTCN test sub-tree. Hence, the standard X.25 test suite can be easily specified in a standard test notation (TTCN).

(4) The abstract test specification can be easily translated from TTCN to an executable test language.

5.2 Limitations of Method

This method for test suite design and specification is based on the finite state machine model. This model was chosen as it is well understood and many tools are available for validating such models. Moreover, the X.25 standard suggests the use of finite state machine models and the state tables at the Packet level. Hence the same model is considered to be suitable at the Data Link level. However, translation of the X.25 specification from the informal description (English) to the FSM specification poses difficulties, such as:

(1) There is no known algorithm for determining the number of states required in the finite state machine so that it is minimal, completely specified and is a true translation of the informal description. To reduce the complexity of the model the number of states have to be kept within limits. This has been done by predicates and separation of protocol phases (modularization).

(2) There is a possibility that the number of states in the FSM model would be large. In that case, the state transition table will become very large and testing all transitions for proper, improper and inopportune PDUs will become almost impossible. Some other design method may, in that case, become necessary.

(3) The timing in executing a transition is not clear in the model. For this reason precise specification of concurrent events may become difficult to represent. Synchronization of test events is possible only if the variations due to transition execution delays are ignored.

(4) In the case of multiple transitions between the same pair of states, and the simultaneous occurance of the events corresponding to these transitions, it is not known which event will take precedence. This is resolved by assigning a priority of events. In this work the priority is:
send event > receive event > timer event.

These are some of the drawbacks of the FSM model and result in limitations in the test method used in this thesis. The number of states in the FSM determines the state transition table from which the transition tests are derived. The state transition table and the entries in it will become unmanageably complex if the number of states were to increase. The transition execution time can become critical if the test cases are to be synchronized (between upper and lower tester). In the case of simultaneous occurance of a send and receive event (over full duplex channels) at the tester or the IUT, an arbitrary choice has to be made in processing the events, and this may lead to spurious test failure in some cases.

There are other limitations to this method which are not due to the FSM model. For example, there is no guarantee that the execution of a transition will detect errors in the implementation. Many protocol violations may occur when multiple transitions are executed in some sequence. In some FSM models it may not be possible to determine an unique verification step. In the case of some hyper-active implementations the initialization step may become very complex. The choice of improper PDUs is based on testing experience, and in some test cases these may be inappropriate.

5.3 Limitations of Test Notations

Presently there is no standard test notation. There is an interim test notation (TTCN), and other formal/ informal notations under consideration by ISO. Many of the FDTs under consideration were not originally designed for the purpose of specifying test cases. A FDT (like LOTOS) is very abstract in its description of test cases and it is difficult to translate it into an executable test case.

5.3.1 TTCN Test Suites

Some of the main limitations of TTCN test suites are as follows:-

(1) TTCN is more informal than the other FDTs and it's defination is flexible. It is possible that different test suite developers produce different interpretations

of the abstract test specifications in TTCN because it is less formal.

(2) The TTCN test notation is new and therefore not yet widely used. There are indications that it will be used, but perhaps only by test suite developers. Under such circumstances, the development of TTCN will be slower than the other FDTs.

(3) In the process of gaining confidence in TTCN, there should be both theory and tools to support the notation. At this time there are no tools for the use of TTCN and the theory of conformance testing is under development.

The overall summary of the evaluation of TTCN as a test notation is that it satisfies all the criteria of an effective test specification language, but its complete development will take time (as much time as the other FDTs like LOTOS.!).

5.3.2 Informal Notations

The informal notations are used in test specification because other more formal notations have not yet stabilized. Informal notations are least effective in specifications because they are open to various interpretations. For example, Data Link layer test cases in time sequence diagrams [NO48,X3S3] are very informal and do not express test cases clearly.

The main limitations of informal notations are the lack of clarity, precision and standardization. The time sequence diagrams (TSD) are informal test notations which may have multiple interpretations. The declaration of PDUs and parameters are not precise in the time sequence diagrams and are potentially confusing for test suite developers. Timer functions can not be accurately expressed in TSD. Behaviours at more than one service access points can not be properly shown in TSD. Overall, TSD is unsuitable as a test notation. For these reasons TTCN was proposed to ISO, and has been accepted as an interim standard.

5.4 Comparison with W-Method of Test Case Generation

In this section we will consider the automated generation of test sequences by the W-method and compare it with the manually generated TTCN test cases.

The automated test sequence generator GAST (Generateur Automatique de Sequences de Test) is used in an experimental run on the Link Set-up FSM of Figure 24 on page 70. The use of GAST must be preceeded by the use of VADILOC [RafAn83]. GAST can use three different techniques - Transition Tour (TT), Checking Sequence (DS) and W-method. The DS sequence does not exist for X.25, so the W.P method is the best choice for using GAST on X.25. The limitations of the W.P method are as follows:-

(a) If the same object is used as an input event and as an action, it means that the event is an element of communication (PDU) between two communicating automata. (Refer to Chapter 3, section 3.3, for syntax of transitions in VADILOC, GAST). The result of this interpretation is that a timeout action after sending or receiving a PDU can not be adequately expressed in the transition base.

(b) While the DS and TT methods produce a single input sequence, the W-method produces several sequences each starting from the initial state. The transfer sequence to the initial state is to be added to the W.P sequence as many times as there are test sequences. However, the transfer sequence is not generated automatically by GAST.

(c) The run time required by the W-method is large even for small FSM models and hence expensive. The method also generates a lot of redundant test sequences in this time.

(d) The success of the W-method depends to a large extent on the specification of the transition base used by this tool. The transition base must include specifications for all improper and inopportune PDUs, if these are to appear in the test cases.

(e) Additional validation via VADILOC would require the construction of a new FSM for the DCE.

5.5 Suggestions for Further Research

From the above discussion it is clear that the manual method used in this thesis is superior to the automatically generated test cases. But then, automated tools are increasingly becoming sophisticated. It is a question of time before these methods become comparable to the manual method. Since protocols are specified by extended FSM, the regular techniques for testing simple FSMs do not necessarily apply to practical protocols. The techinique used in this thesis applies to X.25 protocol which is today a very important protocol in data communication. The augmented state table developed in this thesis has been adopted by ISO SC6 as a standard for LAPB X.25 conformance testing.

5.5.1 Suggestion for TTCN Tools

Many new tools are required for aiding test suite design and specification. These tools must be capable of handling abstract test specifications as well as automatically generating test cases from the protocol specifications. Some of the tools on which work is in progress in the Protocols Research Group, University of Ottawa are:

- TLAN to TTCN Translator for the Data Link and Packet layers.
- TTCN to TLAN Translator for X.25.
- TTCN Editor, which is a tool for converting a 'flat' representation of the events into a properly indented paragraphed representation.

Once these tools are available, they can be integrated into a test tool design aids environment. In this environment the tools can be selected from a menu. The test cases can be automatically classified and saved in an abstract test suite within the proper group or subgroup. The TTCN translator can be made layer independent by implementing a seperate tool for generating PDU lists and TTCN Declaration Tables interactively by the user of the tool. Subsequently, the TTCN translator can be called using the TTCN Declaration Tables as the PDU database for the particular layer.

5.5.2 Completeness of Test Suites

The complete X.25 test suite should include the following (not included at present):

(1) Test Cases for simultaneous multiple connections at the Packet level. For example, the behaviour of the DTE when one connection is set up on one logical channel and another connection is disconnected on another logical channel.

(2) Multi-link procedure (MLP) at the frame level will require optional test cases to be executed when support of MLP is claimed.

(3) For DTEs conforming to CCITT X.25 LAP (1984), optional tests are required in the Data Link layer test suite.

(4) Additional optional tests at the Data Link layer are required to support the EXTENDED (MODULO 128) MODE of operation.

(5) Optional tests for testing the DATAGRAM MODE of operation is required at the Packet level.

This thesis work has focused on the design and specification of X.25-DTE (1984) conformance testing , a realistic, practical application of protocol testing. The contributions made in this work have received international attention. Very few such approaches have been tried on real protocols. Thus, this thesis is a significant contribution to work in this area.

Appendix A
State Transition Tables

PDU FROM TESTER	PDU AT DTE	TYPE	STATE L1 (DISCONNECT)	STATE L2 (SABM SENT)	STATE L3 (DISC SENT)	STATE L4 (INFO.)	STATE L5 (FRMR SENT)	STATE L6 (REJ SENT)	STATE L7 (RNR SENT)
DISC	DISC(P=1)	ASDTC	DM(F=1)/ACT1 L1 / - - L1LAPB101 P	DM(F=1)@ L1 - L2LAPB101 P	UA(F=1)@ L3 - L3LAPB101 P	UA(F=1) L1 - L4LAPB101 P	UA(F=1) L1 - L5LAPB101 P	UA(F=1) L1 - L6LAPB101 P	UA(F=1) L1 - L7LAPB101 P
	DISC(P=0)	ASDTC	DM(F=0)/ACT1 L1 / - - L1LAPB102 P	DM(F=0)@ L1 - L2LAPB102 P	UA(F=0)@ L3 - L3LAPB102 P	UA(F=0) L1 - L4LAPB102 P	UA(F=0) L1 - L5LAPB102 P	UA(F=0) L1 - L6LAPB102 P	UA(F=0) L1 - L7LAPB102 P
SABM	SABM(P=1)	ASDTC	UA(F=1)/ACT1 L4 / - - L1LAPB103 P	UA(F=1)@ L2 - L2LAPB103 P	DM(F=1)@ L3 - L3LAPB103 P	UA(F=1) L4 - L4LAPB103 P	UA(F=1) L4 - L5LAPB103 P	UA(F=1) L4 - L6LAPB103 P	UA(F=1) L4 - L7LAPB103 P
	SABM(P=0)	ASDTC	UA(F=0)/ACT1 L4 / - - L1LAPB104 P	UA(F=0)@ L2 - L2LAPB104 P	DM(F=0)@ L3 - L3LAPB104 P	UA(F=0) L4 - L4LAPB104 P	UA(F=0) L4 - L5LAPB104 P	UA(F=0) L4 - L6LAPB104 P	UA(F=0) L4 - L7LAPB104 P

PDU FROM TESTER	PDU AT DTE	TYPE	STATE L1 (DISCONNECT)	STATE L2 (SABM SENT)	STATE L3 (DISC SENT)	STATE L4 (INFO.)	STATE L5 (FRMR SENT)	STATE L6 (REJ SENT)	STATE L7 (RNR SENT)
DM	DM(F=1)	ASDTC	DISCARD/ACT1 L1 / - - L1LAPB315 0	DISCARD L1 - L2LAPB105 P	DISCARD L1 - L3LAPB105 P	ACTION2 - - L4LAPB301 0	ACTION2 - - L5LAPB315 0	ACTION2 - - L6LAPB301 0	ACTION2 - - L7LAPB301 0
	DM(F=0)	ASDTC	ACTION1 - - L1LAPB105 P	DISCARD L2 - L2LAPB315 0	DISCARD L3 - L3LAPB315 0	ACTION2 - - L4LAPB105 P	SABM / DISC (P=1)/ (P=1) L2 / L3 - L5LAPB105 P	ACTION2 - - L6LAPB105 P	ACTION2 - - L7LAPB105 P
UA	UA(F=1)	ASDTC	DISCARD/ACT1 L1 / - - L1LAPB316 0	DISCARD L1 - L2LAPB106 P	DISCARD L1 - L3LAPB106 P	ACTION2 - - L4LAPB302 0	DISCARD L5 - L5LAPB316 0	ACTION2 - - L6LAPB302 0	ACTION2 - - L7LAPB302 0
	UA(F=0)	ASDTC	DISCARD/ACT1 L1 / - - L1LAPB317 0	DISCARD L2 - L2LAPB316 0	DISCARD L3 - L3LAPB316 0	ACTION2 - - L4LAPB303 0	DISCARD L5 - L5LAPB317 0	ACTION2 - - L6LAPB303 0	ACTION2 - - L7LAPB303 0

PDU FROM TESTER	PDU AT DTE	TYPE	STATE L1 (DISCONNECT)	STATE L2 (SABM SENT)	STATE L3 (DISC SENT)	STATE L4 (INFO.)	STATE L5 (FRMR SENT)	STATE L6 (REJ SENT)	STATE L7 (RNR SENT)
NON OCTET FRAMES	I FRAME P=1 WITH INFO=1	A S D T C	DM(F=1)/ACT1 L1 / - - L1LAPB201 |	DISCARD L2 - L2LAPB201 |	DISCARD L3 - L3LAPB201 |	DISCARD L4 - L4LAPB201 |	FRMR(F=1) L5 - L5LAPB201 |	DISCARD L6 - L6LAPB201 |	DISCARD L7 - L7LAPB201 |
SHORT FRAMES	FRAME WITH ADDR FIELD=03 LESS THAN 32 BITS	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB202 |	DISCARD L2 - L2LAPB202 |	DISCARD L3 - L3LAPB202 |	DISCARD L4 - L4LAPB202 |	DISCARD L5 - L5LAPB202 |	DISCARD L6 - L6LAPB202 |	DISCARD L7 - L7LAPB202 |
LONG FRAMES	I FRAME WITH MORE THAN MAXIMUM FRAME SIZE	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB203 |	DISCARD L2 - L2LAPB203 |	DISCARD L3 - L3LAPB203 |	FRMR L5 Y=1,C/R=0 L4LAPB203 |	DISCARD L5 - L5LAPB203 |	FRMR L5 Y=1,C/R=0 L6LAPB203 |	FRMR L5 Y=1,C/R=0 L7LAPB203 F
FCS ERROR FRAMES	U FRAME WITH FCS ERROR CONTROL FIELD=0F	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB204 |	DISCARD L2 - L2LAPB204 |	DISCARD L3 - L3LAPB204 |	DISCARD L4 - L4LAPB204 |	DISCARD L5 - L5LAPB204 |	DISCARD L6 - L6LAPB204 |	DISCARD L7 - L7LAPB204 |

PDU FROM TESTER	PDU AT DTE	TYPE	STATE L1 (DISCONNECT)	STATE L2 (SABM SENT)	STATE L3 (DISC SENT)	STATE L4 (INFO.)	STATE L5 (FRMR SENT)	STATE L6 (REJ SENT)	STATE L7 (RNR SENT)
UN-DEFINED RESPONSE	CONTROL FIELD=FF	A S D T C	DISCARD/ACT1 L1 / - L1LAPB205	DISCARD L2 - L2LAPB205	DISCARD L3 - L3LAPB205	FRMR L5 W=1,C/R=1 L4LAPB205	DISCARD L5 - L5LAPB205	FRMR L5 W=1,C/R=1 L6LAPB205	FRMR L5 W=1,C/R=1 L7LAPB205
UN-DEFINED COMMAND	CONTROL FIELD=SARM	A S D T C	DISCARD/ACT1 L1 / - L1LAPB206	DISCARD L2 - L2LAPB206	DISCARD L3 - L3LAPB206	FRMR L5 W=1,C/R=0 L4LAPB206	FRMR L5 - L5LAPB206	FRMR L5 W=1,C/R=0 L6LAPB206	FRMR L5 W=1,C/R=0 L7LAB206
DISC WITH INFO. FIELD	DISC (INF=11)	A S D T C	DISCARD/ACT1 L1 / - L1LAPB207	DISCARD L2 - L2LAPB207	DISCARD L3 - L3LAPB207	FRMR L5 W=1,X=1,C/R=0 L4LAPB207	FRMR L5 - L5LAPB207	FRMR L5 W=1,X=1,C/R=0 L6LAPB207	FRMR L5 W=1,X=1,C/R=0 L7LAPB207
UA WITH INFO. FIELD	UA(F=1) (INF=00)	A S D T C	DISCARD/ACT1 L1 / - L1LAPB208	DISCARD L2 - L2LAPB208	DISCARD L3 - L3LAPB208	FRMR L5 W=1,X=1,C/R=1 L4LAPB208	DISCARD L5 - L5LAPB208	FRMR L5 W=1,X=1,C/R=1 L6LAPB208	FRMR L5 W=1,X=1,C/R=1 L7LAPB208
NOTE1	SABM(P=1) WITH ADDR FIELD=01	A S D T C	DISCARD/ACT1 L1 / - L1LAPB209	DISCARD L2 - L2LAPB209	FRMR L3 - L3LAPB209	DISCARD L5 W=1,C/R=1 L4LAPB209	FRMR L5 - L5LAPB209	DISCARD L5 W=1,C/R=1 L6LAPB209	DISCARD L7 - L7LAPB209
WRONG ADDRESS	ANY FRAME WITH ADDR FIELD<>01 <>03	A S D T C	DISCARD/ACT1 L1 / - L1LAPB210	DISCARD L2 - L2LAPB210	DISCARD L3 - L3LAPB210	DISCARD L4 - L4LAPB210	DISCARD L5 - L5LAPB210	DISCARD L6 - L6LAPB210	DISCARD L7 - L7LAPB210

PDU FROM TESTER	PDU AT DTE	TYPE	STATE L1 (DISCONNECT)	STATE L2 (SABM SENT)	STATE L3 (DISC SENT)	STATE L4 (INFO)	STATE L5 (FRMR SENT)	STATE L6 (REJ SENT)	STATE L7 (RNR SENT)
DM WITH INFO. FIELD	DM (F=0) (INF=11)	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB211	DISCARD L2 - L2LAPB211	DISCARD L3 - L3LAPB211	FRMR L5 W=1,X=1,C/R=1 L4LAPB211	DISCARD L5 - L5LAPB211	FRMR L5 W=1,X=1,C/R=1 L6LAPB211	FRMR L5 W=1,X=1,C/R=1 L7LAPB211
I FRAME WITHOUT INFO. FIELD	I (P=0) N(S)=V(R)	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB212	DISCARD L2 - L2LAPB212	DISCARD L3 - L3LAPB212	RR(P/F=0) L4 - L4LAPB108 P	DISCARD L5 - L5LAPB314 0	RR(P/F=0) L4 - L6LAPB106 P	RNR(P/F=0) L7 - L7LAPB112 P
I FRAME IN-CORRECT N(S)	I (P=0) N(S)<>V(R)	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB213	DISCARD L2 - L2LAPB213	DISCARD L3 - L3LAPB213	REJ(P/F=0) L6 - L4LAPB212	DISCARD L5 - L5LAPB212	DISCARD L6 - L6LAPB212	RNR(P/F=0) L7 L7LAPB212
SABM WITH INFO. FIELD	SABM(P=1) (INF=11)	A S D T C	DM(F=1)/ACT1 L1 / - - L1LAPB214	DISCARD L2 - L2LAPB214	DISCARD L3 - L3LAPB214	FRMR(F=1) L5 W=1,X=1,C/R=0 L4LAPB213	FRMR(F=1) L5 - L5LAPB213	FRMR(F=1) L5 W=1,X=1,C/R=0 L6LAPB213	FRMR(F=1) L5 W=1,X=1,C/R=0 L7LAPB213
I FRAME BAD N(R)	I (P=1) N(R)<>V(S)	A S D T C	DM(F=1)/ACT1 L1 / - - L1LAPB215	DISCARD L2 - L2LAPB215	DISCARD L3 - L3LAPB215	FRMR(F=1) L5 Z=1,C/R=0 L4LAPB214	FRMR(F=1) L5 - L5LAPB214	FRMR(F=1) L5 Z=1,C/R=0 L6LAPB214	FRMR(F=1) L5 Z=1,C/R=0 L7LAPB214
RR RESP. WITH INFO. FIELD	RR(P=0) (INF=11)	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB216	DISCARD L2 - L2LAPB216	DISCARD L3 - L3LAPB216	FRMR L5 W=1,X=1 L4LAPB215	DISCARD L5 - L5LAPB215	FRMR L5 W=1,X=1 L6LAPB215	FRMR L5 W=1,X=1 L7LAPB215
RNR RESP. WITH INFO. FIELD	RNR(P=1) (INF=00)	A S D T C	DM(F=1)/ACT1 L1 / - - L1LAPB217	DISCARD L2 - L2LAPB217	DISCARD L3 - L3LAPB217	FRMR(F=1) L5 W=1,X=1,C/R=0 L4LAPB216	FRMR(F=1) L5 - L5LAPB216	FRMR(F=1) L5 W=1,X=1,C/R=0 L6LAPB216	FRMR(F=1) L5 W=1,X=1,C/R=0 L7LAPB216

PDU FROM TESTER	PDU AT DTE	TYPE	STATE L1 (DISCONNECT)	STATE L2 (SABM SENT)	STATE L3 (DISC SENT)	STATE L4 (INFO.)	STATE L5 (FRMR SENT)	STATE L6 (REJ SENT)	STATE L7 (RNR SENT)
I	I (P=1) N(S)=V(R)	A	DM(F=1)/ACT1	DISCARD	DISCARD	RR(F=1)	FRMR(F=1)	RR(F=1)	RNR(F=1)
		S	L1 / -	L2	L3	L4	L5	L4	L7
		D	-	-	-	-	-	-	-
		T	L1LAPB301	L2LAPB301	L3LAPB301	L4LAPB106	L5LAPB301	L6LAPB107	L7LAPB304
		C	0	0	0	P	0	P	0
	I (P=0) N(S)=V(R)	A	DISCARD/ACT1	DISCARD	DISCARD	RR(F/P=0)	DISCARD	RR(F/P=0)	RNR(F/P=0)
		S	L1 / -	L2	L3	L4	L5	L4	L7
		D	-	-	-	-	-	-	-
		T	L1LAPB302	L2LAPB302	L3LAPB302	L4LAPB107	L5LAPB302	L6LAPB108	L7LAPB305
		C	0	0	0	P	0	P	0
RR RESPONSE	RR(F=1)	A	DISCARD/ACT1	DISCARD	DISCARD	ACTION2	DISCARD	ACTION2	ACTION2
		S	L1 / -	L2	L3	-	L5	-	-
		D	-	-	-	-	-	-	-
		T	L1LAPB303	L2LAPB303	L3LAPB303	L4LAPB304	L5LAPB303	L6LAPB304	L7LAPB306
		C	0	0	0	0	0	0	0
RNR RESPONSE	RNR(F=1)	A	DISCARD/ACT1	DISCARD	DISCARD	ACTION2	DISCARD	ACTION2	ACTION2
		S	L1 / -	L2	L3	-	L5	-	-
		D	-	-	-	-	-	-	-
		T	L1LAPB304	L2LAPB304	L3LAPB304	L4LAPB305	L5LAPB304	L6LAPB305	L7LAPB307
		C	0	0	0	0	0	0	0
REJ RESPONSE	REJ(F=1)	A	DISCARD/ACT1	DISCARD	DISCARD	ACTION2	DISCARD	ACTION2	ACTION2
		S	L1 / -	L2	L3	-	L5	-	-
		D	-	-	-	-	-	-	-
		T	L1LAPB305	L2LAPB305	L3LAPB305	L4LAPB306	L5LAPB305	L6LAPB306	L7LAPB308
		C	0	0	0	0	0	0	0
RR RESPONSE	RR(F=0)	A	DISCARD/ACT1	DISCARD	DISCARD	DISCARD	DISCARD	DISCARD	DISCARD
		S	L1 / -	L2	L3	L4	L5	L6	L7
		D	-	-	-	-	-	-	-
		T	L1LAPB306	L2LAPB306	L3LAPB306	L4LAPB109	L5LAPB106	L6LAPB307	L7LAPB309
		C	0	0	0	P	P	0	0
RNR RESPONSE	RNR(F=0)	A	DISCARD/ACT1	DISCARD	DISCARD	DISCARD	DISCARD	REJ(P/F=0)/ DISCARD	RNR(P/F=0)/ DISCARD
		S	L1 / -	L2	L3	L4	L5	L6	L7
		D	-	-	-	-	-	-	-
		T	L1LAPB307	L2LAPB307	L3LAPB307	L4LAPB110	L5LAPB107	L6LAPB308	L7LAPB310
		C	0	0	0	P	P	0	0

PDU FROM TESTER	PDU AT DTE	TYPE	STATE L1 (DISCONNECT)	STATE L2 (SABM SENT)	STATE L3 (DISC SENT)	STATE L4 (INFO.)	STATE L5 (FRMR SENT)	STATE L6 (REJ SENT)	STATE L7 (RNR SENT)
REJ RESPONSE	REJ(F=0)	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB308 0	DISCARD L2 - L2LAPB308 0	DISCARD L3 - L3LAPB308 0	DISCARD L4 - L4LAPB111 P	DISCARD L5 - L5LAPB108 P	DISCARD L6 - L6LAPB309 0	DISCARD L7 - L7LAPB311 0
RR COMMAND	RR(P=1)	A S D T C	DM(F=1)/ACT1 L1 / - - L1LAPB309 0	DISCARD L2 - L2LAPB309 0	DISCARD L3 - L3LAPB309 0	RR(F=1) L4 - L4LAPB112 P	FRMR(F=1) L5 - L5LAPB306 0	REJ(F=1)/ RR(F=1) L6 - L6LAPB109 P	RNR(F=1) L7 - L7LAPB106 P
RNR COMMAND	RNR(P=1)	A S D T C	DM(F=1)/ACT1 L1 / - - L1LAPB310 0	DISCARD L2 - L2LAPB310 0	DISCARD L3 - L3LAPB310 0	RR(F=1) L4 - L4LAPB113 P	FRMR(F=1) L5 - L5LAPB307 0	REJ(F=1)/ RR(F=1) L6 - L6LAPB110 P	RNR(F=1) L7 - L7LAPB107 P
REJ COMMAND	REJ(P=1)	A S D T C	DM(F=1)/ACT1 L1 / - - L1LAPB311 0	DISCARD L2 - L2LAPB311 0	DISCARD L3 - L3LAPB311 0	RR(F=1) L4 - L4LAPB114 P	FRMR(F=1) L5 - L5LAPB308 0	REJ(F=1)/ RR(F=1) L6 - L6LAPB111 P	RNR(F=1) L7 - L7LAPB108 P
RR COMMAND	RR(P=0)	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB312 0	DISCARD L2 - L2LAPB312 0	DISCARD L3 - L3LAPB312 0	DISCARD L4 - L4LAPB115 P	DISCARD L5 - L5LAPB309 0	DISCARD L6 - L6LAPB112 P	DISCARD - - L7LAPB109 P
RNR COMMAND	RNR(P=0)	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB313 0	DISCARD L2 - L2LAPB313 0	DISCARD L3 - L3LAPB313 0	DISCARD L4 - L4LAPB116 P	DISCARD L5 - L5LAPB310 0	REJ(P/F=0)/ DISCARD L6 - L6LAPB113 P	RNR(P/F=0)/ DISCARD L7 - L7LAPB110 P
REJ COMMAND	REJ(P=0)	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB314 0	DISCARD L2 - L2LAPB314 0	DISCARD L3 - L3LAPB314 0	DISCARD L4 - L4LAPB117 P	DISCARD L5 - L5LAPB311 0	DISCARD L6 - L6LAPB114 P	DISCARD L7 - L7LAPB111 P

PDU FROM TESTER	PDU AT DTE	TYPE	STATE L1 (DISCONNECT)	STATE L2 (SABM SENT)	STATE L3 (DISC SENT)	STATE L4 (INFO.)	STATE L5 (FRMR SENT)	STATE L6 (REJ SENT)	STATE L7 (RNR SENT)
FRMR	FRMR(F=0)	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB318 0	DISCARD L2 - L2LAPB317 0	DISCARD L3 - L3LAPB317 0	ACTION2 - - L4LAPB307 0	ACTION2 - - L5LAPB312 0	ACTION2 - - L6LAPB310 0	ACTION2 - - L7LAPB312 0
	FRMR(F=1)	A S D T C	DISCARD/ACT1 L1 / - - L1LAPB319 0	DISCARD L2 - L2LAPB318 0	DISCARD L3 - L3LAPB318 0	ACTION2 - - L4LAPB308 0	ACTION2 - - L5LAPB313 0	ACTION2 - - L6LAPB311 0	ACTION2 - - L7LAPB313 0

NOTE1 : U Frame with primary and secondary addresses inter-changed

ACTION1 (ACT1) : Some DTE's might initiate the link set up procedure as soon as they get into the disconnected state (Active DTE's). Prior to link set up some active DTE's may initiate link disconnection by transmitting a DISC(P=1) and go to state L3, or they might transmit an unsolicited DM(F=0) response to request the DCE to initiate link set up and stay in L1, or initiate link set up by transmitting a SABM(P=1) and going to state L2.

ACTION2 : Once the link is up (STATE L4,L5,L6,L7) the DTE will initiate link resetting procedures by transmitting a DISC(P=1) and going to state L3 or by transmitting a SABM(P=1) and going to state L2 or ask the DCE to initiate link resetting by transmitting a DM(F=0) and going to state L1.

APPENDIX B

PICS FOR X.25 DATA LINK LAYER

PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT PROFORMA

The PICS proforma for the Data Link layer is illustrated below. The item descriptin column in the PICS proforma describes the details of the implementar's claims. As well, guide lines are provided for the standard test method to be used. The item numbers are serial, and indicate the level of functional groupings. For example, item 5 is for the sequence numbering and 5.1, 5.2 are the mandatory requirement of modulo 8 and the optional modulo 128, respectively. The mandatory/ option column clearly indicates this implementation choice. In cases where a value is supported by a particular protocol parameter, the Value column is used. For example, the value of the mandatory T1 timer is 3 minutes. A dash (-) is used to indicate not supported parameters in the value column.

IUT Identifier: xxx99.9

Test of Conformance to: CCITT X.25 (1984) Data Link Layer

IUT Manufacturer: Name of Manufacturer.

ITEM NO.	ITEM DESCRIPTION (PROTOCOL SECTION)	MANDATORY/ OPTIONAL	SUPPORTED (Y/N)	VALUE IF SUPPORTED
1.	Testing method allowed	-	-	RS or CS
2.	Upper interface accessible	-	N	-
3.	ASP or PDU to be observed	-	-	ASP
4.	Basic Interconnection Test session required	O	Y	-
5.1	Frame Sequence numbering is modulo 8	M	Y	-
5.2	Frame Sequence number of modulo 128 supported (Extended mode)	O	N	-
6.	Multilink Procedure suppurted	O	N	-
7.	LAP Procedure also supported	O	N	-
8.	Link Setup Procedure			
8.1	Passive DTE	-	N	-
8.2	Active DTE	-	N	-
8.3	Hyperactive DTE	-	Y	-

9.	Timers Implemented			
9.1	T1 Timer	M	Y	3 Min.
9.2	T2 Timer	M	Y	15 Sec.
9.3	T3 Timer	O	N	-
9.4	T4 Timer	O	N	-
10.	Maximum frame size Supported (N1 bits)	-	-	4096 bits
11.	Maximum retry count supported (N2)	O	-	10
12.	Window size of 7 supported	M	-	-
12.1	Optional window size supported	O	Y	2
13.	DTE states forced by upper tester			
13.1	Busy state	-	Y	-
13.2	DTE waiting DISC Acknowledgement	-	Y	-
13.3	DTE Sending I-frame	-	Y	-

Appendix C

TTCN TABLES FOR BASIC INTERCONNECTION TESTS

REFERENCE : BASIC INTERCONNECTION Test ID : DCESABMDISC	TEST PURPOSE : Verify proper Link set up and disconnect for basic interconnection operation			PAGE: 1 of : 2
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
DCESABMDISC ----------				
L! START TIMER (T1)				
L? DISC		DISC(S1)		
L? TIMEOUT (T1)] N2			Rec'd nothing for N2 retrys	Fail
L? UA (AD=03, F=1)				Pass
L? DM (AD=03, F=1)				Pass
L? DISC (AD=01, P=1)			Disc Collision	
L! UA		UA(S1)		
L? UA (AD=03, P=1)				Pass
L? TIMEOUT T1				Fail
L? SABM (AD=01, P=1)			SABM Collision	
L! DM		DM(S1)		
L? DM (AD=03, P=1)				Pass
L? TIMEOUT T1				Fail
L? Otherwise				Fail
EXTENDED COMMENTS :				

REFERENCE : Basic Interconnection Test ID : DCESABMDISC	TEST PURPOSE : Verify Link set up phase			PAGE: 2 OF : 2
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
DCESABMDISC				
(L1 START TIMER (T1) L1 SABM	1	SABM(S1)		
L? TIMEOUT (T2) ! N2			Rec'd nothing (1)	Fail
L? UA (AD=03, F=1) + HYPER			Hyper active DTE	Pass
L? DISC (AD01, P1) L1 DM		DM(S1)	(2)	
L? DM (AD=03, F=1) ==> 1				
L? SABM (AD=01, P=1) L1 UA		UA(S1)		
L? UA (AD=03, F=1) ==> 1				
L? Otherwise			(1)	Fail

EXTENDED COMMENTS :
(1) Test was failed after N2 retrys
(2) Test allows the reception of any syntactically correct frame but acks only DISC and SABM

REFERENCE : DCESABMDISC/HYPER SUBTREE : HYPER	TEST PURPOSE : Check if the DTE is hyperactive			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
HYPER				
L? I (AD=01, P=0, UD=1000FB00*)			(1)	
L! I		I(S2)		
L? RR (AD=03, NR=1, F=1) (ISHYPER := 1)			(2)	Pass
L? TIMER (T1) + INFO TRANSFER			No response Info Transfer phase	Fail
L? I (AD=01, P=1, UD=1000FB??*)			(1)	
L! I		I(S2)		
L? RR (AD=03, NR=1, F=1) (ISHYPER := 1)			(2)	Pass
+ INFO TRANSFER			Info Transfer phase	
L? TIMER (T1)				Fail
L? TIMER (T1) (ISHYPER := 0)			DTE is not hyper	Fail
L? Otherwaise (ISHYPER := 0)				Fail

EXTENDED COMMENTS :
(1) This is a restart request packet
(2) ISHYPER is a global variable indicating a hyperactive DTE

REFERENCE : DCESABMDISC/HYPER/ INFO TRANSFER SUBTREE : INFO TRANSFER	TEST PURPOSE : Check information transfer phase after link set up.			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
INFO_TRANSFER L! RR Command (ISHYPER = 1) L? RR (AD=03, NR=1, F=1) + LINK_DISCON L? TIMER (T1)		RRC(S1)	(1) (2)	Fail Fail
L! RR Command (ISHYPER = 0) L? RR (AD=03, NR=0, F=1) + LINK_DISCON L? TIMER (T1)		RRC(S1)	(2)	Pass Fail
EXTENDED COMMENTS : (1) DUT must respond to this in info transfer phase (2) The link is disconnected at the end of basic interconnection testing				

REFERENCE : DCESABMDISC/HYPER/ INFO TRANSFER/LINK DISCON SUBTREE : LINK_DISCON	TEST PURPOSE : Check proper Link disconnection after entering info transfer phase			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
LINK_DISCON L! DISC L? UA (AD=03, P=1) + L!CHK L? TIMER (T1)		DISC(S1)	Verify L! state No respond	Pass Fail
EXTENDED COMMENTS :				

Appendix D

TTCN TABLES FOR FUNCTIONAL RANGE TESTS

REFERENCE : FUNCTIONAL RANGE T1 TIMER Test ID : T1_TIMER	TEST PURPOSE : Test the functional range of the T1 timer			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
T1_TIMER				
+ L1SATTE			Put in L1 state	
L! DM		DM(S0)	(1)	
(T1_TOLERANCE := T1 * 0.8)			(2)	
L? TIMER (T1)			No response	Fail
L? SABM (AD=01, P=1)		SABM(S1)		
L! START TIMER (T1_TOLERANCE)				
L? TIMEOUT TIMER (T1_TOLERANCE)				Pass
(T1_TOLERANCE := T1 * 1.2)			(3)	
L? SABM(AD=01,P=1)				Pass
+ L2CHK			Verifv L2 State	
L? TIMEOUT TIMER (T1_TOLERANCE)				Fail
L? Otherwise				Fail
+ DM_COLLISION			Collision	Pass
+ DM_DISC_COLLISION			Sitiuations	Pass

EXTENDED COMMENTS :
(1) Soliciting a Link set up by DTE
(2) T1 checked 20% below limit
(3) T1 checked 20% above limit

REFERENCE : T1 TIMER/ DM COLLISION SUBTREE : DM COLLISION	TEST PURPOSE : To resolve a DM DM Collision and check T1 functional range	PAGE: 1 OF : 1

BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
DM COLLISION				
L? DM (AD=03, F=0)			(1)	
L?:SABM (AD=01, P=1)				
(T1 TOLERANCE := T1 * 0.8)			Set lower limit	
L! START TIMER (T1 TOLERANCE)				
L? TIMEOUT TIMER (T1 TOLERANCE)			Timer went off too soon	
L? SABM (AD=01, P=1)				
(T1 TOLERANCE := T1 * 1.2)			(2)	
L! START TIMER (T1 TOLERANCE)				
L? SABM(AD=01,P=1) + L?CHK				Pass
L? TIMEOUT TIMER (T1 TOLERANCE)			Too late	Fail

EXTENDED COMMENTS :
(1) DM DM Collisions can take place in case of an active DTE
(2) T1 tested for higher limit (120%)

REFERENCE : T1 TIMER/DM DISC COLLISION SUBTREE : DM_DM_COLLISION	TEST PURPOSE : To resolve a DM Collision and verify functional range of T1 timer			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
DM_DM_COLLISION --- L? DISC (AD=01, P=1) (T1_TOLERANCE := T1 * 0.8) L! START TIMER (T1_TOLERANCE) L? TIMEOUT TIMER (T1_TOLERANCE) L? DISC (AD=01, P=1) (T1_TOLERANCE := T1 * 1.2) L! START TIMER (T1_TOLERANCE) L? TIMEOUT TIMER (T1_TOLERANCE) L? DISC (AD=01, P=1) + L3CHK			Active DTE Too soon Too late Verify L3 state	 Fail Fail
EXTENDED COMMENTS :				

REFERENCE : Functional range T2 TIMER Test ID : T2TIMER	TEST PURPOSE : Functional range test for T2 timer			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
T2 TIMER (SUPPORT_T2 = 1)			(1)	
+ L1STATE			Put DTE in L1	
L! START timer1 (T1)				
L! DM (LIMIT := 0.2 * T2)		DM(S0)	Sollicite SABM	
L? TIMEOUT timer1 (T1)			No response	Fail
L? SABM (AD=01, P=1)				
L! START timer2 (T2 LIMIT)			(2)	
L? Otherwise			Too soon for T2	
			(3)	
L? TIMEOUT timer2 (T2-LIMIT)				
L! UA		UA(S1)		
+ L4CHK			Verify L4 state	Pass
+ DM COLLISION (T2-LIMIT)				Pass
+ DISC COLLISION (T2-LIMIT)				Pass
L? Otherwise				Fail

EXTENDED COMMENTS :

(1) SUPPORT_T2 = 1 is a boolean Expression qualifying the entire tree. It checks if T2 timer is supported by the IUT, otherwise test is not executed.

(2) Lower Limit (20% less) of T2 is checked

(3) Anything ==> any PDU

REFERENCE : T2 TIMER/ DM COLLISION SUBTREE : DM_COLLISION .	TEST PURPOSE : Functional range check for T2 timer with DM collision.			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
DM_COLLISION (TIMER_LIMIT) L? DM (AD=03, F=0) L? SABM (AD=01, P=1) L! START timer (TIMER_LIMIT) L? Otherwise			(1) This is (0.8 * T2) Too soon	Fail
L? TIMEOUT timer (TIMER_LIMIT) L! UA + ISHYPER_DTE + L4CHK		UA(S1)	handle hyper-active DTE	
EXTENDED COMMENTS : (1) DTE resolves DM_DM_COLLISION				

REFERENCE : T2 TIMER/ DISC_COLLISION SUBTREE : DISC_COLLISION	TEST PURPOSE : Functional range test for T2 with DM_DISC collision			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
DISC_COLLISION (TIMER_LIMIT) L? DISC (AD=01, P=1) L! START timer (TIMER_LIMIT) L? Anything L? TIMEOUT timer (TIMER_LIMIT) L! DM + L1CHK		DM(S1)	Too soon Verify L1 state	Fail Pass
EXTENDED COMMENTS :				

REFERENCE : Fuctional range N1 bits Test ID : N1BITS	TEST PURPOSE : Test the maximum number of bits (N1 bits) allowed in an I-frame			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
N1BITS (TESTER_N1 > DTE_N1) ------ + L1STATE + LINK_UP			(1) Link is set up by the tester or DTE in case of hyper	
+ ISHYPER_DTE				
L! START timer (T1) L! I (UD := ('H41) (N1-32)) L? TIMEOUT timer(T1) L? I (AD=01, P=1, UD=1000FB) L? Otherwise + LONG_FRAME (DTE_N1)		I(S3)	Just right size (2) Packet Level clear	Fail Pass Fail
+ ISNOTHYPER_DTE (DTE_N1)			Test is repeated for passive DTEs.	Pass
EXTENDED COMMENTS : (1) Check whether the DTE's N1 value is less than the maximum output buffer size of the tester, otherwise do not run this test. (2) User data field is set to (N1 - 32) octets of ASCII 'A'.				

REFERENCE : N1BITS/LONG_FRAME SUBTREE : LONG_FRAME	TEST PURPOSE : Test maximum frame size N1 with a long frame.			PAGE: 1 OF: 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
LONG_FRAME --------- (TESTER_N1 > DTE_N1 + 8) L! I L! START timer (T1) L? FRMR (AD=03, F=1, DIAG=Y*) + L5CHK (DIAG) L? TIMEOUT timer (T1) L? Otherwise		I (S4) FRMR (S1 (Y))	check buffer size Exception Correct Diagnostics No response Wrong response	 Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : N1BITS/ISNOTHYPER SUBTREE : ISNOTHYPER	TEST PURPOSE : Test maximum frame size for passive DTEs.			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
ISNOTHYPER_DTE ----------			check buffer size	
(TESTER_N1 > DTE_N1 + 8)				
L! I		I(S5)	Exception	
L! START timer (T1)				
L? FRMR (AD=03, F=1, DIAG=Y*)		FRMR(S1(Y))		
+ L5CHK (DIAG)			Correct Diagnostics	Pass
L? TIMEOUT timer (T1)			No response	Fail
L? Otherwise				Fail
EXTENDED COMMENTS :				

REFERENCE : Functional range N2 COUNT Test ID : N2COUNT	TEST PURPOSE : To test the DTE retransmission parameter (N2) value.			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
N2_COUNT + L1STATE L! START timer (T1) L? DM L? TIMEOUT timer (T1) {L? SABM} N2 + DM_COLLISION_COUNT (N2) + DISC_COLLISION_COUNT (N2) L? Otherwise		 DM(S0) SABM(S2) 	Put DTE in L1 Ignored! (1) Collision Sitiuations are resolved Wrong N2 or wrong frames	 Fail Pass Pass Pass Fail
EXTENDED COMMENTS : (1) Observe SABM for N2 retransmissions				

REFERENCE : N2COUNT/ DM COLLISION COUNT SUBTREE : DM_COLLISION_COUNT	TEST PURPOSE : Resolve DM DM collision and check N2 parameter.			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
DM_COLLISION_COUNT (N2)				
L? DM (AD=03, F=0)			collision resolved	
L? SABM (AD=01, P=1) {L? SABM (AD01, P=1)} (N2 -1)			(1)	Pass
L? Otherwise			Wrong N2 value	Fail
L? Otherwise			Wrong frame	Fail
EXTENDED COMMENTS : (1) Observed N2 retransmission of SABM				

REFERENCE : N2COUNT/ DISC COLLISION COUNT SUBTREE : DISC_COLLISION_COUNT	TEST PURPOSE : Resolve DM DISC collision and check N2 parameter			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
DISC_COLLISION_COUNT (N2)				
L? DISC (AD=01, P=1) L! DM (AD=01, F=0)			was a collision resolved by tester	
L? SABM (AD=01, P=1) {L? SABM (AD=01, P=1)} (N2 - 1)				
L? Otherwise			Wrong N2 value	Fail
L? Otherwise			Wrong frame	Fail
EXTENDED COMMENTS :				

REFERENCE : LAPB/HYPERACTIVE_DTE SUBTREE : ISHYPER_DTE	TEST PURPOSE : For Hyperactive DTE, this subtree handles the packet level restart requests			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
ISHYPER_DTE				
L? I (NS=NR=0, UD='H1000FB*)		I(S7)	Restart Indication	
L! I (start(T1))		I(S3)	Ack, Restart Confirmation	
L? RR (AD=03, F=0); Cancel (T1)				Pass
L? Timeout (T1)				Fail
L? Otherwise; Cancel(T1)				Fail
L? I (NS=NR=0, UD='H1000FB*)		I(S6)	Restart packet with poll bit set	
L! I (start (T1))		I(S3)		
L? RR (AD=03, F=1); Cancel (T1)				Pass
L? Timeout (T1)				Fail
L? Otherwise; Cancel (T1)				Fail
EXTENDED COMMENTS :				

Appendix E

TION TABLES FOR INITIALIZATION STEP

REFERENCE : LAPB/L1STATE/Initial-step TEST-ID : L1STATE		TEST PURPOSE : Put DTE into L1 state		
BEHAVIOR DESCRIPTION	LABEL	PDU REF	COMMENTS	RESULT
L1STATE[L]				
{ L! DISC		PDU DISC(S1)		
L! START(T1)				
L? TIMEOUT(T1) ¦ N2	1		After N2 retrys	FAIL
L? UA (AD=03 & F≠1)				
L! CANCEL(T1)				PASS
L? DM (AD=03 & F=1)				
L! CANCEL(T1)				PASS
L? DISC (AD=01 & P=1)				
L! UA			Unnumbered frame	
==> 1		PDU UA(S1)	Collision	
L? Otherwise			Try again	
==> 1				

REFERENCE : LAPB/L2STATE/Initial-step TEST-ID : L2STATE	TEST PURPOSE : Put DTE into L2 state			
BEHAVIOR DESCRIPTION	LABEL	PDU REF	COMMENTS	RESULT
L2STATE [L] ————				
+L1STATE				
{ L! DM	2	PDU DM(S0)		
L! START(T1)				
L? TIMEOUT(T1) } N2	3		After N2 retrys	FAIL
L? SABM (AD=01 & P=1)				
L! CANCEL(T1)				PASS
L? DISC (AD=01 & P=1)				
L! CANCEL(T1)				
L? UA		PDU UA(S1)		
==> 2				
L? DM (AD=03 & F=0)				
==> 3				
L? Otherwise				FAIL

REFERENCE : LAPB/L3STATE/ Initialization Step SUBTREE : L3STATE	TEST PURPOSE : To put DTE in L3 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L3STATE [U, L] ————				
(UPTESTER = "PRESENT")			(1)	
U! DTE_DISC		DTE DISC(S1)	(2)	
L? DISC (AD=01, P=1)				
L! Start timer (T1)				
L? DISC (AD=01, P=1)			Retransmission Observed	Pass
L! Cancel (T1)				
L? Otherwise;				Fail
Cancel (T1)				Fail
L? Otherwise				

EXTENDED COMMENTS :
(1) Upper tester is required to run this step
(2) DTE initiated DISC PDU

REFERENCE : LAPB/L4STATE/ Initialization Step SUBTREE : L4STATE	TEST PURPOSE : To put DTE in L1 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4STATE (HYPER_DTE) ------- + L2STATE				
L! UA; Start (T1)		UA (S1)	Ack Link up	
L? I (UD=1000FB*, NS=NR=0)		I (S7)	Hyper DTE	
(HYPER_DTE := 1); Cancel (T1)				Pass
L? Timeout T1 (HYPER_DTE :=0)				Pass
L? Otherwise; Cancel (T1)			Wrong frame	Fail
EXTENDED COMMENTS :				

REFERENCE : LAPB/L5STATE/ Initialization Step SUBTREE : L5STATE	TEST PURPOSE : To put DTE in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5STATE ------- + L4STATE (HYPER DTE) L! I (HYPER DTE = 1); Start (T1) L? FRMR; Cancel (T1) L? Timeout (T1) L? Otherwise; Cancel (T1) L! I (HYPER DTE = 0); Start (T1) L? FRMR; Cancel (T1) L? Timeout (T1) L? Otherwise; Cancel (T1)		I(S9) FRMR(S0 (WXYZ)) I (S10) FRMR(S0 (WXYZ)) 	Exception Retain Diagnostics no FRMR sent Exception Check and Retain Diagnostic 	 Pass Fail Fail Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : LAPB/L6STATE/ Initialization Step SUBTREE : L6STATE	TEST PURPOSE : Put DTE in L6 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L6STATE ------- + L4STATE (HYPER DTE) L! I (HYPER DTE = 1 (UD := ¯´H0)) L! Start (T1) L? REJ; Cancel (T1) L? REJC; Cancel (T1) L? REJC; Cancel (T1) L? Timeout (T1) L? Otherwise; Cancel (T1) L! I (HYPER DTE = 0 (UD := ¯´H0)) L! Start (T1) L? REJ; Cancel (T1) L? REJC; Cancel (T1) L? Timeout (T1) L? Otherwise; Cancel (T1)		I(S8) REJ(S2) REJC(S2) REJC(S3) I(S11) REJ(S3) REJC(S4) 	Wrong NS Wrong NS 	 Pass Pass Pass Fail Fail Pass Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : LAPB/L7STATE/ Initialization Step SUBTREE : L7STATE	TEST PURPOSE : Put DTE in L7 STATE			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L7STATE [L, U] ------- (UPTESTER = "PRESENT") + L4STATE (HYPER DTE) U! DTE BUSY (HYPER DTE = 1) L? RNRC; Start (T1) L? RNRC; Cancel (T1) L! RR (NR := 0) L? Timeout (T1) L? Otherwise; Cancel (T1) U! DTE BUSY (HYPER DTE = 0) L? RNRC; Start (T1) L? RNRC; Cancel (T1) L! RR (NR := 0) L? Timeout (T1) L? Otherwise; Cancel (T1)		RNRC (S2) RNRC (S2) RNRC (S2) RR (S1) RNRC (S3) RNRC (S3) RNRC (S3) RR (S1)	(1) Put in L4 state Make DTE BUSY Observe Retransmission Put in L4 state Observe Retransmission	Pass Fail Fail Pass Fail Fail
EXTENDED COMMENTS : (1) This test step requires Uppertester for puting DTE in busy condition				--

REFERENCE : LAPB/LINK_UP/ Initialization Step SUBTREE : LINK_UP	TEST PURPOSE : DTE behavior to tester initiated link set up			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	**LABEL**	**PDU/ASP REFERENCE**	**COMMENTS**	**RESULTS**
LINK_UP				
+ L1STATE				
{L! SABM; Start (T1)	1	SABM(S1)		
L? Timeout (T1)				
==>1				
L? DISC (AD=01, P=1)				
L! DM		DM(S1)		
L? DM (AD=03, F=1)				
L! Cancel (T1)				
==> 1} N2			After retrys	Fail
L? Timeout (T1)				Fail
L? Otherwise;				
Cancel (T1)				Fail
L? SABM (AD=01, P=1)				
L! UA		UA(S1)		
L? UA (AD=03, F=1);			Ack to Link	
Cancel (T1)			Set up recd.	Pass
L? UA (AD=03, F=1),				
Cancel (T1)				Pass
L? Otherwise				Fail
EXTENDED COMMENTS :				

Appendix F

TTCN TABLES FOR VERIFICATION STEP

REFERENCE : LAPB/L1STATE/Verify-step TEST-ID : L2CHK		TEST PURPOSE : Verify DTE into L2 state		
BEHAVIOR DESCRIPTION	LABEL	PDU REF	COMMENTS	RESULT
L2CHK[L] ——— L! TIMER(T1) L? SABM (AD=0 & P=1) L? Otherwise			Wait T1 Second	 PASS FAIL

REFERENCE : LAPB/L3STATE/ Verification Step SUBTREE : L3CHK	TEST PURPOSE : Verify that DTE is in the L3 state.			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L3CHK ——— L! START timera (T1) L? Timeout (T1) L! START timerb (T2) L? DISC (AD=01, P=1); Cancel (T2) L? Timeout timerb (T2) L? DISC (AD=01, P=1) L? Otherwise; Cancel (T2) L? Otherwise; Cancel (T1)			DTE time out (1) DTE retransmits	 Pass Fail Pass Fail Fail
EXTENDED COMMENTS : (1) Test allows the DTE T2 time to react				

REFERENCE : LAPB/L4STATE/ Verification step SUBTREE *L4CHK	TEST PURPOSE : Verify that DTE is in the L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4CHK (VS, VR) ----- L! RRC		RRC(S1(VR))		
L! Start timer (T1) L? RR (AD=03, P=1)				Pass
L? RRC (AD=03, P=1, NR <= VS)			RRC collision	
L! RR; Cancel (T1)		RR(S1(VR))		Pass
L? I (AD=03, P=?, NR <= VS); Cancel (T1)			(1)	Pass
L? Timeout timer (T1)				Fail
L? Otherwise; Cancel (T1)				Fail
EXTENDED COMMENTS : (1) DTE chooses to send a piggyback response				

REFERENCE : LAPB/L5STATE/ Verification Step SUBTREE : L5CHK	TEST PURPOSE : To verify DTE is in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5CHK (WXYZ) ——— L! RRC L! START timer (T1) L? FRMR (AD=03, F=1); Cancel (T1) L? Timeout timer (T1) L? Otherwise; Cancel (T1)		RRC(S1) FRMR(S0 (WXYZ))	(1) Diagnostic check	 Pass Fail Fail
EXTENDED COMMENTS : (1) This supervisory command will be discarded by the DTE, but the P bit will be examined.				

REFERENCE : LAPB/L6STATE/ Verification Step SUBTREE : L6CHK	TEST PURPOSE : To verify DTE is in L6 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L6CHK (HYPER_DTE) ———			(1)	
L! I (UD := 'H0) (HYPER_DTE = 1)		I(S8)	Resend frame	
L! START timer (T1)				
L? RR (AD=03, F=1, NR=1);				
Cancel (T1)				Pass
L? TIMEOUT timer (T1)				Fail
L? Otherwise; Cancel (T1)				Fail
L! I (UD := 'H0) (HYPER_DTE = 0)				
L! START timer (T1)				
L? RR (AD=03, F=1, NR=0);				
Cancel (T1)				Pass
L? TIMEOUT timer (T1)				Fail
L? Otherwise; Cancel (T1)				Fail
EXTENDED COMMENTS : (1) The HYPER_DTE parameter tells us expected NS, NR values are 01, 01 respectively				

REFERENCE : LAPB/L7STATE/ Verification Step SUBTREE : L7CHK	TEST PURPOSE : To verify that DTE is in L7 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L7CHK ——— L! RRC (NR := 0); Start (T1) L? RNR (AD=03, F=1, NR=?) L! Cancel (T1) L? Timeout (T1) L? Otherwise; Cancel (T1)		RRC(S1)	Still busy	Pass Fail Fail
EXTENDED COMMENTS :				

Appendix G

TION TABLE FOR L1 TRANSITION TESTS

REFERENCE : L1_TRANSITION/IMPROPER/ L1LAPB317 TEST-ID : L1LAPB317		TEST PURPOSE : DTE response to UA/F=0 (an inopportune frame)		
BEHAVIOR DESCRIPTION	LABEL	PDU REF	COMMENTS	RESULT
L1-317[L] ——— +L1STATE L! UA		PDU UA(S0)		
L! TIMER(T1) +L1CHK			Frame discarded Verify L1 state	PASS
L? DM (AD=03 & F=0) +L1CHK			Verify L1 state	PASS
L? SABM (AD=01 & P=1) +L2CHK			Verify L2 state	PASS
L? DISC (AD=01 & F=1) +L3CHK			Verify L3 state	PASS
L? Otherwise				FAIL

REFERENCE : L1_TRANSITION/PROPER/L1LAPB103		TEST PURPOSE : DTE reponse		
TEST-ID : L1LAPB103		to SABM/P=1, Proper frame		
BEHAVIOR DESCRIPTION	**LABEL**	**PDU REF**	**COMMENTS**	**RESULT**
L1-103[L]				
+L1STATE				
{ L! SABM		PDU SABM(S1)		
L! START(T1)				
L? TIMEOUT(T1) } N2	1		After N2 retrys	FAIL
L? UA (AD=03 & F=1)				
L! CANCEL(T1)				
+L4CHK			Verify L4 state	PASS
L? DM (AD=03 & F=1)				
L! CANCEL(T1)				
+L1CHK			Verify L1 state	PASS
L? SABM (AD=01 & P=1)				
+L2CHK			Verify L2 state	PASS
L? DISC (AD=01 & P=1)				
!UA				
==> 1				
L? DM (AD=03 & F=0)				
==> 1				
L? Otherwise				FAIL

REFERENCE : L1_TRANSITION/IMPROPER/L1LAPB212		TEST PURPOSE : DTE response to		
TEST-ID : L1LAPB212		I frame/P=0 (an improper frame)		
BEHAVIOR DESCRIPTION	**LABEL**	**PDU REF**	**COMMENTS**	**RESULT**
L1-212[L]				
+L1STATE				
L! I		PDU I(S6)	Exception	
L! START(T1)				
L! TIMEOUT(T1)			Frame discarded	
+L1CHK			Verify L1 state	PASS
L? DM (AD=03 & F=0)				
L! CANCEL(T1)				
+L1CHK			Verify L1 state	PASS
L? SABM (AD=01 & F=1)				
L! CANCEL(T1)				
+L2CHK			Verify L2 state	PASS
L? DISC (AD=0 & P=1)				
L! CANCEL(T1)				
+L3CHK			Verify L3 state	PASS
L? Otherwise				FAIL

Appendix H

TTCN TABLE FOR L2 TRANSITION TESTS

REFERENCE : L2 TRANSITION/ PROPER/L2LAPB101 TEST ID : L2LAPB101, L2LAPB102	TEST PURPOSE : Verify DTE sends a DM response to DISC in L2 state			PAGE: 1 OE : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L2LAPB101 + L2STATE L! Start (T1) L! DISC L? DM (AD=03, F=1); Cancel (T1) + L1CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		DISC(S1)	(1)	Pass Fail Fail
L2LAPB102 + L2STATE L! Start (T1) L! DISC L? DM (AD=03, F=0); Cancel (T1) + L1CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		DISC(S0)	(1)	Pass Fail Fail
EXTENDED COMMENTS : (1) This is a SABM_DISC collision situation				

REFERENCE : L2 TRANSITION/ PROPER/L2LAPB103 TEST ID : L2LAPB103	TEST PURPOSE : Verify DTE sends a UA/F=1 response to SABM/P=1 in L2 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L2LAPB103 --------- + L2STATE L! Start (T1) L! SABM L? UA (AD=03, F=1); Cancel (T1) L! UA + ISHYPER DTE + L4CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		SABM(S1) UA(S1)	(1)	 Pass Fail Fail
EXTENDED COMMENTS : (1) This is a SABM_SABM collision situation				

REFERENCE : L2 TRANSITION/ PROPER/L2LAPB104 TEST ID : L2LAPB104	TEST PURPOSE : Verify DTE sends a UA/F=0 in response to SABM/P=0 in L2 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L2LAPB104 --- + L2STATE L! Start (T1) L! SABM L? UA (AD=03, F=0); Cancel (T1) L! UA + ISHYPER DTE + L4CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		SABM(S0) UA(S0)		Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : L2 TRANSITION/ PROPER/L2LAPB105 TEST ID : L2LAPB105	TEST PURPOSE : Verify DTE enters L1 state on Link denial in L2 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L2LAPB105 --- + L2STATE L! DM + L1CHK L? Otherwise		DM(S1)		Pass Fail
EXTENDED COMMENTS :				

REFERENCE : L2 TRANSITION/ PROPER/L2LAPB106 TEST ID : L2LAPB106	TEST PURPOSE : Verify DTE enters the info. transfer phase on receiving ack in L2 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L2LAPB106 : --------- + L2STATE L! UA + ISHYPER DTE + L1CHK L? Otherwise		DM(S1)		Pass Fail
EXTENDED COMMENTS :				

REFERENCE : L2_TRANSITION/INOPPORTUNE/ L2LAPB315 TEST-ID : L2LAPB315		TEST PURPOSE : DTE response to DM/F=0 (an inopportune frame)		
BEHAVIOR DESCRIPTION	LABEL	PDU REF	COMMENTS	RESULT
L2-315[L] ------ +L2STATE L! DM L! TIMER(T1) +L2CHK		PDU DM(S0)	Frame discarded Verify L2 state	PASS

REFERENCE: L2_TRANSITION/PROPER/ L2LAPB101			TEST PURPOSE : DISC and SABM collision a proper DISC frame sent	
TEST-ID : L2LAPB101				
BEHAVIOR DESCRIPTION	LABEL	PDU REF	COMMENTS	RESULT
L2-101[L]				
+L2STATE				
L! DISC		PDU DISC(C1)		
L! TIMER(T1)			No response	FAIL
L? DM (AD=03 & F=1)				
L! CANCEL(T1)				
+L1CHK			Verify L1 state	PASS
L? Otherwise				FAIL

REFERENCE :			TEST PURPOSE : DTE reacts to undefined command (an improper frame)	
TEST-ID : L2LAPB206				
BEHAVIOR DESCRIPTION	LABEL	PDU REF	COMMENTS	RESULTS
L2-206[L]				
+L2STATE				
L! SARM		PDU SARM(S0)	This frame is not defined in LAPB	
L! TIMER(T1)			Frame discarded	
+L2CHK			Verify L2 state	PASS
L? SABM (AD=01 & P=1)				PASS
L? Otherwise				FAIL

Appendix I

TION TABLE FOR L4 TRANSITION TESTS

REFERENCE : L4 TRANSITION/ PROPER/L4LAPB101 TEST ID : L4LAPBL01	TEST PURPOSE : Verify DTE send UA/F=1 response to a DISC/P=1 command in L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPBL01 + L4STATE L! Start (T1) L! DISC		DISC(S1)		
L? UA (AD=03, F=1); Cancel (T1) + L1CHK			Verify L1 state	Pass
L? Timeout (T1)			No response	Fail
L? Otherwise; Cancel (T1)			Wrong response	Fail
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSITION/PROPER/L4LAPB102 TEST ID : L4LAPB102	TEST PURPOSE : Verify DTE sends UA/F=0 to a DISC/P=0 in L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB102 + L4STATE L! Start (T1) L! DISC L? UA (AD=03, F=0); Cancel (T1) + L1CHK L? Timeout (T1) L? Otherise; Cancel (T1)		DISC(S0)	Verify L1 state No response Wrong response	Pass Fail Fail
EXTENDED COMMENTS:				

REFERENCE : L4 TRANSITION/PROPER/L4LAPB103 TEST ID : L4LAPB103	TEST PURPOSE : Verify DTE sends UA/F=1 response to a SABM/P=1 command in L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB103 +L4STATE L! Start (T1) L! SABM L? UA (AD=03, F=1); Cancel (T1) + L2CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		SABM(S1)	Verify L2 state	Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSITION/ PROPER/L4LAPB104 TEST ID : L4LAPB104	TEST PURPOSE : Verify DTE ends UA/F=0 response to a SABM/P=0 command in L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB104 ———————— + L4STATE *L1 Start (T1) L1 SABM L2 UA (AD=03, F=0); Cancel (T1) L2CHK L2 Timeout (T1) L2 Otherwise; Cancel (T1)		SABM(S0)	Verify L2 State	Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : L4 transition/ PROPER/L4LAPB105 TEST ID : L4LAPB105	TEST PURPOSE : Verify DTE initiates Link resetting procedure in response to DM/F=0			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB105 + L4STATE L! Start (T1) L! DM		DM(S0)		
L? DM (AD=03, F=0) L! SABM		SABM(S1)	DM DM collision	
L? DA(AD=03, F= ancel (T1) + L4CHK				Pass
L? Otherwise				Fail
L? SABM (AD=01, P=1) + L2CHK				Pass
L? DISC (AD=01, P=1) + L3CHK				Pass
L? Timeout (T1)				Fail
L? Otherwise				Fail
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSITION/ PROPER/L4LAB107 TEST ID : L4LAPB107	TEST PURPOSE : Verify DTE sends RR/F=0 or nothing to I/P=0 frame from tester			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB107				
+ L4STATE; Start (T1)				
L! I (HYPER DTE = 1)		I(S12)	(1)	
L? RR (AD=03, F=0, NR=2); Cancel (T1)				
+ L4CHK				Pass
L? Timeout (T1)				
+ L4CHK				Pass
L? Otherwise; Cancel (T1)				Fail
L! I(HYPER DTE = 0)			(1)	
L? RR (AD=03, F=0, NR=1); Cancel (T1)				
+ L4CHK				Pass
L? Timeout (T1)				
+ L4CHK				Pass
L? Otherwise; Cancel (T1)				Fail
EXTENDED COMMENTS : (1) N				

REFERENCE: L4 TRANSITION/ PROPER/4LAPB108				
TEST ID : L4LAPB108	response to I frame without information field			OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB108				
+ L4STATE				
L! Start (T1)				
L! I (HYPER DTE = 1)		I(S14)		
L? RR (AD=03, F=0, NR=2);				
Cancel (T1)				
+ L4CHK				Pass
L? Timeout (T1)				
+ L4CHK				Pass
L? Otherwise				Fail
L! I (HYPER DTE = 0)				
L? RR (AD=03, F=0, NR=1);				
Cancel (T1)				
+ L4CHK				Pass
L? Timeout (T1)				
+ L4CHK				Pass
L? Otherwise; Cancel (T1)				Fail
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSITION/ PROPER/L4LAPB109 TEST ID : L4LAPB109	TEST PURPOSE : Check DTE response to RR/F=0 frame in the L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LABP109 + L4STATE L! Start (T1) L! RR (HYPER DTE = 1 (NR := 1)) L? Timeout (T1) + L4CHK L? Otherwise; Cancel (T1) L! RR (HYPER DTE = 0 (NR := 0)) L? Timeout (T1) + L4CHK L? Otherwise; Cancel (T1)		RR(S0) RR(S0)	Discarded Discarded	Pass Fail Pass Fail
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSITION/ IMPROPER/L4LAPB201 TEST ID : L4LAPB201	TEST PURPOSE : Verify if DTE checks for non octet frames at Data Link Level			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L1LAPB201 (OCTET_ALIGNMENT_CHECK = "Yes") + L4STATE L! Start (T1) L! I (HYPER DTE = 1) L? Timeout (T1) + L4CHK L? Otherwise; Cancel (T1) L! I (HYPER DTE = 0) L? Timeout (T1) + L4CHK L? Otherwise; Cancel (T1)		I (S16) I (S17)	(1) non octet frame Discarded non octet frame Discarded	 Pass Fail Pass Fail
EXTENDED COMMENTS : (1) This test is executed only if DTE claims octet alignment check at Data Link Level.				

REFERENCE : L4 TRANSITION/ IMPROPER/L4LAPB202 TEST ID : L4LAPB202	TEST PURPOSE : Verify the DTE discards a short frame (less than 32 bits)			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB202 --------- + L4STATE T1 Start (T1) L! 'H03 L? Timeout (T1) L4CHK L? Otherwise; Cancel (T1)			Send Hex string Discarded	Pass Fail
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSTION/ IMPROPER/L4LAPB203 TEST ID : L4LAPB203	TEST PURPOSE : Verify DTE sends FRMR/Y=1 in response to a long frame in L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB203 (TESTER_N1 > DTE_N1 + 8) + L4STATE L! Start (T1) L! I (HYPER DTE = 1 (UD := {'H41} (N1 + 8))) L? FRMR (Y = 'B1); Cancel (T1) + L5CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		I(S4) FRMR(S3(Y))	Exception	Pass Fail Fail
L! I (HYPER DTE = 0 (UD := {'H41} (N1 + 8))) L? FRMR (Y = 'B1); Cancel (T1) L5CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		FRMR(S4(Y))	Exception	Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSITION/ IMPROPER/L4LAPB204 TEST ID : L4LAPB204	TEST PURPOSE : Verifv that DTE discards a frame with FCS error			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB204 --------- + L4STATE L! Start (T1) L! FCS ERROR L? Timeout (T1) + L4CHK L? Otherwise; Cancel (T1)			Exception Discarded	Pass Fail
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSITION/ IMPROPER/L4LAPB205 TEST ID : L4LAPB205	TEST PURPOSE : Verify DTE sends FRMR/W=1, CR=0 in response to undefined command in L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB205 --------- + L4STATE L! Start (T1) L! SARM		SARM(S0)	Exception	
L? FRMR (HYPER DTE = 1); Cancel (T1) + L5CHK		FRMR(S7)		Pass
L? FRMR (HYPER DTE = 0); Cancel (T1) + L5CHK		FRMR(S8)		Pass
L? Timeout (T1)				Fail
L? Otherwise; Cancel (T1)				Fail
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSITION/ IMPROPER/L4LAPB206 TEST ID : L4LAPB206	TEST PURPOSE : Verify DTE responds FRMR/W=1, C/R=1 to undefined response in L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB206 --------- + L4STATE L! Start (T1) L! 'H01FF L? FRMR (HYPER_DTE = 1); Cancel (T1) + L5CHK L? FRMR (HYPER_DTE = 0); + L5CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		 FRMR(S5) FRMR(S6) 	 Exception (1) 	 Pass Pass Fail Fail
EXTENDED COMMENTS : (1) A control field of 'HFF is undefined				

REFERENCE : L4 TRANSITION/ INOPPORTUNE/L4LAPB301 TEST ID : L4LAPB301	TEST PURPOSE : Verify DTE initiates Link resetting in response to DM/F=1 in L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB301 --------- + L4STATE L! Start (T1) L! DM + LINK_RESET		 DM(S1) 	 	 Pass
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSITION/ INOPPORTUNE/L4LAPB302 TEST ID : L4LAPB302, L4LAPB303	TEST PURPOSE : Verify DTE initiates Link resetting procedure in response to UA in L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB302 --------- + L4STATE L! Start (T1) L! UA + LINK_RESET		UA(S1)		Pass
L4LAPB303 --------- + L4STATE L! Start (T1) L! UA + LINK_RESET		UA(S0)		Pass
EXTENDED COMMENTS :				

REFERENCE : L4 TRANSITION/ INOPPORTUNE/L4LAPB304 TEST ID : L4LAPB304	TEST PURPOSE : Verify DTE's response to RR/F=1 in L4 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L4LAPB304 + L4STATE L! Start (T1) L! RR (NR := HYPER DTE) L? DM (AD=03, F=1); Cancel (T1) + L1CHK L? SABM (AD=01, P=1); Cancel (T1) + L2CHK L? DISC (AD=01, P=1); Cancel (T1) + L3CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		RR(S1)	NR is 0 or 1	Pass Pass Pass Fail Fail
EXTENDED COMMENTS :				

Appendix J

TTCN TABLE FOR L5 TRANSITION TEST

REFERENCE : L5 TRANSITION/PROPER/L5LAPB101 TEST ID : L5LAPB101, L5LAPB102	TEST PURPOSE : Verify that DTE responds UA to a DISC in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB101 --------- + L5STATE L! DISC; Start (T1) L? UA (AD=03, F=1); Cancel (T1) + L1CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		DISC(S1)		 Pass Fail Fail
L5LAPB102 --------- + L5STATE L! DISC; Start (T1) L? UA (AD=03, F=0); Cancel (T1) + L1CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		DISC(S0)		 Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : L5 TRANSITION/ PROPER/L5LAPB103 TEST ID : L5LAPB103, L5LAPB104	TEST PURPOSE : Verify DTE responds UA to a SABM command in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB103 ————				
+ L5STATE				
L! SABM; Start (T1)		SABM(S1)		
L? UA (AD=03, F=1);				
Cancel (T1)				
+ L4CHK				Pass
L? Timeout (T1)				Fail
L? Otherwise; Cancel (T1)				Fail
L5LAPB104				
+ L5STATE				
L! SABM; Start (T1)		SABM(S0)		
L? UA (AD=03, F=0);				
Cancel (T1)				
+ L4CHK				Pass
L? Timeout (T1)				Fail
L? Otherwise; Cancel (T1)				Fail
EXTENDED COMMENTS :				

REFERENCE : L5 TRANSITION/ PROPER/L5LAPB105 TEST ID : L5LAPB105	TEST PURPOSE : Verify that DTE initiates Link resetting in responds to DM/F=0 in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB105 --------- L5STATE L! Start (T1) L! DM + LINK_RESET		DM(S0)		Pass
EXTENDED COMMENTS :				

REFERENCE : X TRANSITION/ LINK_RESET TEST ID : LINK_RESET	TEST PURPOSE : Check Link Reset procedure			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
LINK_REET --------- L? DM (AD=03, F=0); Cancel (T1) + L1CHK L? SABM (AD=01, P=1); Cancel (T1) + L2CHK L? DISC (AD=01, P=1); Cancel (T1) + L3CHK L? Timeout (T1) L? Otherwise; Cancel (T1)				Pass Pass Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : L5 TRANSITION/ PROPER/L5LAPB106 TEST ID : L5LAPB106	TEST PURPOSE : Verify DTE discards a RNR/F=0 response in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB106 + L5STATE (HYPER_DTE) L! Start (T1) L! RR (NR := HYPER_DTE) L? Timeout (T1) L? Otherwise; Cancel (T1)		RR(S0)	NR is 0 or 1	Pass Fail
EXTENDED COMMENTS :				

REFERENCE : L5 TRANSITION/ PROPER/L5LAPB107 TEST ID : L5LAPB107	TEST PURPOSE : Verify DTE discards a RNR/F=0 response in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB107 + L5STATE (HYPER_DTE) L! Start (T1) L! RNR (NR := HYPER_DTE) L? Timeout (T1) + L5CHK L? Otherwise; Cancel (T1)		RNR(S0)	NR is 0 or 1 Discarded	 Pass Fail
EXTENDED COMMENTS :				

EFERENCE : L5 TRANSITION/ PROPER/L5LAPB108 TEST ID : L5LAPB108	TEST PURPOSE : Verify DTE discards a REJ/F=0 response in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB108 --------- + L5STATE (HYPER_DTE) L! Start (T1) L! REJ (NR := HYPER_DTE) L? Timeout (T1) + L5CHK L? Otherwise; Cancel (T1)		REJ(S0)	Discarded	Pass Fail
EXTENDED COMMENTS :				

REFERENCE : L5 TRANSITION/ IMPROPER/L5LAPB201 TEST ID : L5LAPB201	TEST PURPOSE : Verify DTE reaction to non octet fram in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB201 (OCTET_ALIGNMENT CHECK = "Yes") + L5STATE (HYPER_DTE) L! Start (T1) L! I (HYPER_DTE = 1) L? FRMR; Cancel (T1) + L5CHK L? Timeout (T1) L? Otherwise; Cancel (T1) L! I (HYPER_DTE = 0) L? FRMR; Cancel (T1) + L5CHK L? Timeout (T1) L? Otherwsie; Cancel (T1)		I (S16) FRMR (S3) FRMR (S4) 	(1)	 Pass Pass Fail Pass Pass Fail
EXTENDED COMMENTS : (1) Test is executed only if octet alignment check is claimed by IUT at Data Link Layer				

REFERENCE : L5 TRANSITION/ IMPROPER/L5LAPB202 TEST ID : L5LAPB202	TEST PURPOSE : Verify DTE reaction to a short frame (32 bits) in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB202 --------- + L5STATE (WXYZ) L! Start (T1) L! 'H03 L? Timeout (T1) + L5CHK (WXYZ) L? Otherwise; Cancel (T1)			Exception (1) Discarded Same diagnostic as before	Pass Fail
EXTENDED COMMENTS : (1) This is a short frame, bounded by flags				

REFERENCE : L5TRANSITION/ IMPROPER/L5LAPB203 TEST ID : L5LAPB203	TEST PURPOSE : Verify DTE's response to a long frame in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB203 (TESTER_N1 > DTE_N1 + 8)			(1)	
+ L5STATE (HYPER_DTE) L! Start (T1) L! I (HYPER_DTE = 1 (UD := {'H41} (N1 + 8))) L? Timeout (T1) L5CHK L? FRMR (AD=03, F=1) L5CHK L? Otherwise; Cancel (T1)		I(S4) FRMR(S0) 	Exception	 Pass Pass Fail
L! I (HYPER_DTE = 0 (UD := {'H41} (N1 + 8))) L? Timeout (T1) + L5CHK L? FRMR (AD=03, F=1) + L5CHK L? Otherwise; Cancel (T1)		I(S5) FRMR(S0) 	Exception	 Pass Pass Fail
EXTENDED COMMENTS : (1) Check sufficient tester buffer.				

REFERENCE : L5 TRANSITION/ INOPPORTUNE/L5LAPB301 TEST ID : L5LAPB301	TEST PURPOSE : Verify DTE retransmits FRMR in response to I frame in L5 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB301 + L5STATE (HYPER_DTE) L! Start (T1) L! (NS := NR := HYPER_DTE (UD := 'H00')) L? Timeout (T1) L! Start (T2) L? FRMR (AD=03, F=1) L! Cancel (T2) L? Otherwise; Cancel (T2) L? Otherwise; Cancel (T2)		I(S1,R)		
		FRMR('0)	retransmission	Pass
				Fail
				Fail
EXTENDED COMMENTS :				

REFERENCE : L5 TRANSITION/ IMPROPER/L5LAPB204 TEST ID : L5LAPB204	TEST PURPOSE : Verifv DTE discards a frame with FCS error in L5 satate			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L5LAPB204 --------- + L5STATE (WXYZ) L! Start (T1) L! FCS ERROR L? Timeout (T1) + L5CHK L? FRMR L? Otherwise; Cancel (T1)		FRMR(S0 (WXYZ))	Exception (1) FRMR retransmits	 Pass Pass Fail
EXTENDED COMMENTS : (1) A Infor frame with FCS error is sent to DTE				

Appendix K

TTCN TABLE FOR L6 TRANSITION TEST

REFERENCE : L6 TRANSITION/ PROPER/L6LAPB101 TEST ID : L6LAPB101	TEST PURPOSE : Verify DTE response to DISC/P=1 in L6 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L6LAPB101 --------- + L6STATE L! Start (T1) L! DISC L? UA (AD=03, F=1); Cancel (T1) + L1CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		DISC(S1)		Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : L6 TRANSITION/ PROPER/L6LAPB102 TEST ID : L6LAPB102	TEST PURPOSE : Verify DTE response to DISC/P=0 in L6 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L6LAPB102 + L6STATE L! Start (T1) L! DISC L? UA (AD=03, F=0); Cancel (T1) + L1CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		DISC(S0)		Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : L6 TRANSITION/ PROPER/L6LAPB103 TEST ID : L6LAPB103	TEST PURPOSE : Verify DTE sends a UA/F=1 in response to SABM/P=1 in L6 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L6LAPB103 + L6STATE L! Start (T1) L! SABM L? UA (AD=03, F=1); Cancel (T1) + L4CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		SABM(S1)		Pass Fail Fail
EXTENDED COMMENTS :				

REFERENCE : L6 TRANSITION/ PROPER/L6LAPB104 TEST ID : L6LAPB104	TEST PURPOSE : Verify DTE sends a UA/F=0 in response to SABM/P=0 in L6 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L6LAPB104 —————				
+ L6STATE				
L! Start (T1)				
L! SARM		SABM(S0)		
L? UA (AD=03, F=0);				
Cancel (T1)				
+ L4CHK				Pass
L? Timeout (T1)				Fail
L? Otherwise; Cancel (T1)				Fail
EXTENDED COMMENTS :				

REFERENCE : L6 TRANSITION/ IMPROPER/L6LAPB215 TEST ID : L6LAPB215	TEST PURPOSE : Verify DTE sends FRMR to RR with info field in L6 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L6LAPB215 --------- + L6STATE (HYPER DTE) L! Start (T1) L! 'H032111 (HYPER DTE = 1) L? FRMR (RCF='H21, W=1, X=1) + L5CHK (WXYZ) L? Timeout (T1) L? Otherwsie; Cancel (T1) L! 'H031500 (HYPER DTE = 0) L? FRMR (CF='H15, W=1, X=1) + L5CHK (WXYZ) L? Timeout (T1) L? Otherwsie; Cancel (T1)		S(9)	Exception (1) Exception (1)	 Pass Fail Fail Pass Fail Fail
EXTENDED COMMENTS : (1) This hex string corresponds to a RR frame with one octet info field				

REFERENCE : L6-TRANSITION/IMPROPER/L6LAPB216 TEST ID : L6LAPB216	TEST PURPOSE : Verify DTE sends FRMR with diagnostics for RNR with info field in L6 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L6LAPB216 --------- + L6STATE (HYPER_DTE) Start (T1) L! 'H033500 (HYPER_DTE = 1) L? FRMR (RCF='H35, W=1, X=1) + L5CHK (WXYZ) L? Timeout (T1) L? Otherwsie; Cancel (T1) L! 'H031500 (HYPER_DTE = 0) L? FRMR (RCF='H15, W=1, X=1) + L5CHK (WXYZ) L? Timeout (T1) L? Otherwsie; Cancel (T1)		S(9)	Exception (1) Exception (1)	Pass Fail Fail Pass Fail Fail
EXTENDED COMMENTS : (1) This is an RNR PDU with one octet info field				

Appendix L

TTCN TABLE FOR L7 TRANSITION TEST

REFERENCE : L7 TRANSITION/ PROPER/L7LAPB112 TEST ID : L7LAPB112	TEST PURPOSE : Verify DTE sends RNR in response to I frame without info field in L7 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L7LAPB112				
+ L7STATE (HYPER DTE)				
L! Start (T1)				
L! I (HYPER DTE = 1)		I(S14)	Exception (1)	
L? RNR ($\overline{A}$D=03, F=0, NR=1)				
L! Cancel (T1)				
+ L7CHK				Pass
L? Timeout (T1)				Fail
L? Otherwsie; Cancel (T1)				Fail
L! I (HYPER DTE = 0)		I(S15)	Exception (1)	
L? RNR ($\overline{A}$D=03, F=0, NR=0)				
L! Cancel (T1)				
+ L7CHK				Pass
L? Timeout (T1)				Fail
L? Otherwsie; Cancel (T1)				Fail
EXTENDED COMMENTS :				

REFERENCE : L7 TRANSITION/ PROPER/L7LAPB111 TEST ID : L7LAPB111	TEST PURPOSE : Verify DTE response to REJ response frame in L7 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L7LAPB111 + L7STATE (HYPER_DTE) L! Start (T1) L! REJ (NR := HYPER_DTE) L? Timeout (T1) + L7CHK L? Otherwise; Cancel (T1)		REJ(S1)	Discarded	Pass Fail
EXTENDED COMMENTS :				

REFERENCE : L7 TRANSITION/ IMPROPER/L7LAPB201 TEST ID : L7LAPB201	TEST PURPOSE : Verify DTE response to non octet aligned I frame in L7 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L7LAPB201 (OCTET ALIGNMENT CHECK = "Yes") + L7STATE L! Start (T1) L! I (HYPER DTE = 1) L? Timeout (T1) + L7CHK ? Otherwise; Cancel (T1) L! I (HYPER DTE = 0) L? Timeout (T1) + L7CHK L? Otherwise; Cancel (T1)		S(6) I (S17)	(1)	P ss Fail P Fail
EXTENDED COMMENTS : (1) Test is executed only if octet alignment is checked at the Data Link Level				

REFERENCE : L7 TRANSITION/ IMPROPER/L7LAPB202 TEST ID : L7LAPB202	TEST PURPOSE : Verify DTE discards a short frame while in L7 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L7LAPB202 ———— + L7STATE L! Start (T1) L! H03 L? Timeout (T1) + L7CHK L? Otherwise; Cancel (T1)			Exception (1)	Pass Fail
EXTENDED COMMENTS : (1) This is a short frame, less than 32 bits bounded by proper flags				

REFERENCE : L7 TRANSITION/ INOPPORTUNE/L7LAPB301 TEST ID : L7LAPB301	TEST PURPOSE : Verify DTE initiates Link reet in response to DM frame in L7 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L7LAPB301 ———— + L7STATE L! Start (T1) L! DM + LINK REER		DM(S1)		Pass
EXTENDED COMMENTS :				

REFERENCE : L7 TRANSITION INOPPORTUNE/L7LAPB302 TEST ID : L7LAPB302, L7LAPB303	TEST PURPOSE : Verify DTE initiates Link reset in response to UA frame in L7 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L7LAPB302 + L7STATE L! Start (T1) L! UA + LINK_RESET		UA(S1)		Pass
L7LAPB303 + L7STATE L! Start (T1) L! UA + LINK_RESET		UA(S0)		Pass
EXTENDED COMMENTS :				

REFERENCE : L7 TRANSITION/ INOPPORTUNE/L7LAPB304 TEST ID : L7LAPB304	TEST PURPOSE : Verify DTE sends a RNR response to an I/P=1 frame while in L7 state			PAGE: 1 OF : 1
BEHAVIOR DESCRIPTION	LABEL	PDU/ASP REFERENCE	COMMENTS	RESULTS
L7LAPB304 --------- +"L7STATE (HYPER_DTE) "L! Start (T1) L! I (NS := NR := HYPER_DTE UD := 'H00)) L? RNR (AD=03, F=1, NR=HYPER_DTE) L! Cancel (T1) + L7CHK L? Timeout (T1) L? Otherwise; Cancel (T1)		I(S1⁹)		Pass Fail Fail
EXTENDED COMMENTS :				

Appendix M

TTCN TABLE FOR DECLARATIONS

Test Suite Parameters		
Name	Range	Comments
T1	As specified in IS7776	Time value, SCR Ref.
T2	As specified in IS7776	Time value, SCR Ref.
N2	As specified in IS7776	Retry count, SCR Ref.
N1	As specified in IS7776	Maximum frame size, SCR Ref. to be used

SAP Declaration	
SAP	Role
L	SAP at the lower tester controlling and observing (N) protocol data units, user to physical layer service.

PDU Declaration	
PDU : DISC (DISConnect)	Restrictions on use : Always used with poll bit set
Protocol Control Information	
Parameter Name	Range of values / Comments
ADdress (AD)	'H01, 'H03
Poll bit (P)	'B0, 'B1

PDU Declaration	
PDU : SABM (Set Asynchronous Balanced Mode)	Restrictions on use : Always used with poll bit set

Protocol Control Information		
Parameter Name	Range of values	Comments
ADdress (AD)	'H01, 'H03	
Poll bit (P)	'B0, 'B1	

PDU Declaration	
PDU : UA (Unnumbered Acknowledgement)	

Protocol Control Information		
Parameter Name	Range of values	Comments
ADdress (AD)	'H01, 'H03	
Final bit (F)	'B0, 'B1	

PDU Declaration	
PDU : DM (Disconnected Mode)	

Protocol Control Information		
Parameter Name	Range of value	Comments
ADdress (AD)	'H01, 'H03	
Final bit (F)	'B0, 'B1	

PDU Declaration	
PDU : RR (Received Ready response)	Restrictions on use : Used only as response PDU

Protocol Control Information		
Parameter Name	Range of values	Comments
ADdress (AD) Final bit (F) Receive Sequence Number (NR)	'H01, 'H03 'B0, 'B1 Modulo 8	

PDU Declaration	
PDU : RR Command (Receive Ready Command)	Restrictions on use : Used only as command PDU

Protocol Control Information		
Parameter Name	Range of values	Comments
ADdress (AD) Poll bit (P) Receive Sequence Number (NR)	'H01, 'H03 'B0, 'B1 Modulo 8	

PDU Declaration	
PDU : I (Information)	Restrictions on use : User Data field may encode Packet Level information

Protocol Control Information		
Parameter Name	Range of values	Comments
ADdress (AD)	'H01, 'H03	
Poll bit (P)	'B0, 'B1	
User Data (UD)	'H1000??*, 'H1001*, or ASCII string	User data may encode packet level restart or restart confirmation on LCN 0 or any other data on LCN 1.
Send Sequence Number (NS)	Modulo 8	
Receive Sequence Number (NR)	Modulo 8	

PDU Declaration	
PDU : RNR (Received Not Ready response)	Restrictions on use : Used only as response PDU

Protocol Control Information		
Parameter Name	Range of values	Comments
ADdress (AD)	'H01, 'H03	
Final bit (F)	'B0, 'B1	
Receive Sequence Number (NR)	Modulo 8	

PDU Declaration	
PDU : RNR Command (Receive Not Ready Command)	Restrictions on use : Used only as command PDU

Protocol Control Information		
Parameter Name	Range of values	Comments
ADdress (AD) Poll bit (P) Receive Sequence Number (NR)	'H01, 'H03 'B0, 'B1 Modulo 8	

PDU Declaration	
PDU : REJ (REJect response)	Restrictions on use : Used only as response PDU

Protocol Control Information		
Parameter Name	Range of values	Comments
ADdress (AD) Final bit (F) Receive Sequence Number (NR)	'H01, 'H03 'B0, 'B1 Modulo 8	

PDU Declaration		
PDU : FCS ERROR (Frame Check Sequence ERROR)	Restrictions on use : This is an improper frame	
Protocol Control Information		
Parameter Name	Range of value	Comments
no parameter	not applicable	This is an I frame ith FCS error sent bv tester ·

PDU Declaration		
PDU : SARM (Set Asynchronous Response Mode)	Restrictions on use : Improper frame in LAPB	
Protocol Control Information		
Parameter Name	Range of value	Comments
ADdress (AD)	H03	
Poll bit (P)	'B0, 'B1	

Appendix N

TTCN TABLE FOR CONSTRAINTS

Constraints Declarations			
PDU DISC (DISConnect)			
List Name	Field Name		Comments
	ADdress	P	
S0	'H03	'B0	AD is address corresponding to 'A' in standard IS7776 P is poll bit
S1	'H03	'B1	

Constraints Declarations			
PDU SABM (Set Asynchronous Balanced Mode)			
List Name	Field Name		Comments
	ADdress	P	
S0	'H03	'B0	Address is 'A' as specified in IS7776
S1	'H03	'B1	

Constraints Declarations			
PDU UA (Unnumbered Acknowledgement)			
List Name	Field Name		Comments
	ADdress	F	
S0	'H01	'B0	AD is 'B' as specified in IS 7776
S1	'H01	'B1	

Constraints Declarations			
PDU DM (Disconneted Mode)			
List Name	Field Name		Comments
	ADdress	F	
S0	'H01	0	This is a link set up request from tester (solicit link up)
S1	'H01	1	This is a link set up denial from tester

Constraints Declarations										
PDU FRMR (FRaMe Reject) (with control field)										
List Name	AD	F	Rejected frame Control	NS	NR	C/R	W	X	Y	Z
S5	'H03	'B1	'HFF	'B001	'B001	'B1	'B1	'B0	'B0	'B0
S6	'H03	'B1	'HFF	'B000	'B000	'B1	'B1	'B0	'B0	'B0
S7	'H03	'B0	'H0F	'B001	'B001	'B0	'B1	'B0	'B0	'B0
S8	'H03	'B0	'H0F	'B000	'B000	'B0	'B1	'B0	'B0	'B0
S9	'H03	'B1	RCF	?	?	?	W	X	Y	Z
S10	'H03	'B0	RCF	?	?	?	W	X	Y	Z

Constraints Declarations

PDU I (Information)

List Name	FIELD NAME					LENGTH	Comments
	AD	P	UD	NS	NR	PARAMETER	
S0	'H03	'B0	UD	?	?	-	proper frame UD is valid
S1	'H03	'B1	UD	'B001	'B010	(N1 - 32)	User Data proper, N1 is free variable
S2	'H03	'B1	'H1000 -FF00	'B000	'B001	-	proper frame
S3	'H03	'B0	'H1000 -FF00	'B000	'B001	-	proper frame
S4	'H03	'B1	UD	'B001	'B001	(N1 + 8)	Exception
S5	'H03	'B1	UD	'B000	'B000	(N1 + 8)	Exception
S6	'H01	'B1	UD	?	?	-	proper frame
S7	'H01	'B0	UD	?	?	-	proper frame
S8	'H03	'B1	UD	'B010	'B001	-	proper frame
S9	'H03	'B1	'H0	'B001	'B010	-	improper NR
S10	'H03	'B1	'H0	'B001	'B001	-	improper NR
S11	'H03	'B0	UD	'B001	'B000		
S12	'H03	'B0	'H4142	'B001	'B001	-	
S13	'H03	'B0	'H4142	'B000	'B000	-	
S14	'H03	'B0	-	'B001	'B001	-	no user data
S15	'H03	'B0	-	'B000	'B000	-	no user data
S16	'H03	'B1	'B1	'B001	'B001	-	non octet aligned
S17	'H03	'B1	'B1	'B000	'B000	-	non octet aligned
S18	'H03	'B1	UD	?	?	-	
S19	'H03	'B1	UD	?	?	-	proper frame
S20	'H03	'B0	UD	?	?	-	proper frame

Constraints Declarations

PDU FRMR (FRaMe Reject)

List Name	FIELD NAME								
	AD	F	NS	NR	C/R	W	X	Y	Z
S0	AD	F	?	?	B0	W	X	Y	Z
S1	H03	B1	B001	B010	B0	B0	B0	Y	B0
S2	H03	B1	B000	B010	B0	B0	B0	Y	B0
S3	H03	B1	B001	B001	B0	B0	B0	Y	B0
S4	H03	B1	B000	B000	B0	B0	B0	Y	B0

Constraints Declarations

PDU RR (Receive Ready response)

List Name	FIELD NAME			Comments
	ADdress	F	NR	
S0	H01	B0	?	NR is free variable for the
S1	H01	B1	?	receive sequence number.

Constraints Declarations

PDU RNR (Receive Not Ready response)

List Name	FIELD NAME			Comments
	ADdress	F	NR	
S0	H01	B0	?	NR is free variable.
S1	H01	B1	?	

Constraints Declarations				
PDU REJ (REJect response)				
List Name	FIELD NAME			Comments
	ADdress	F	NR	
S0	'H01	'B0	?	NR is free variable.
S1	'H01	'B1	?	
S2	'H03	'B1	'B001	

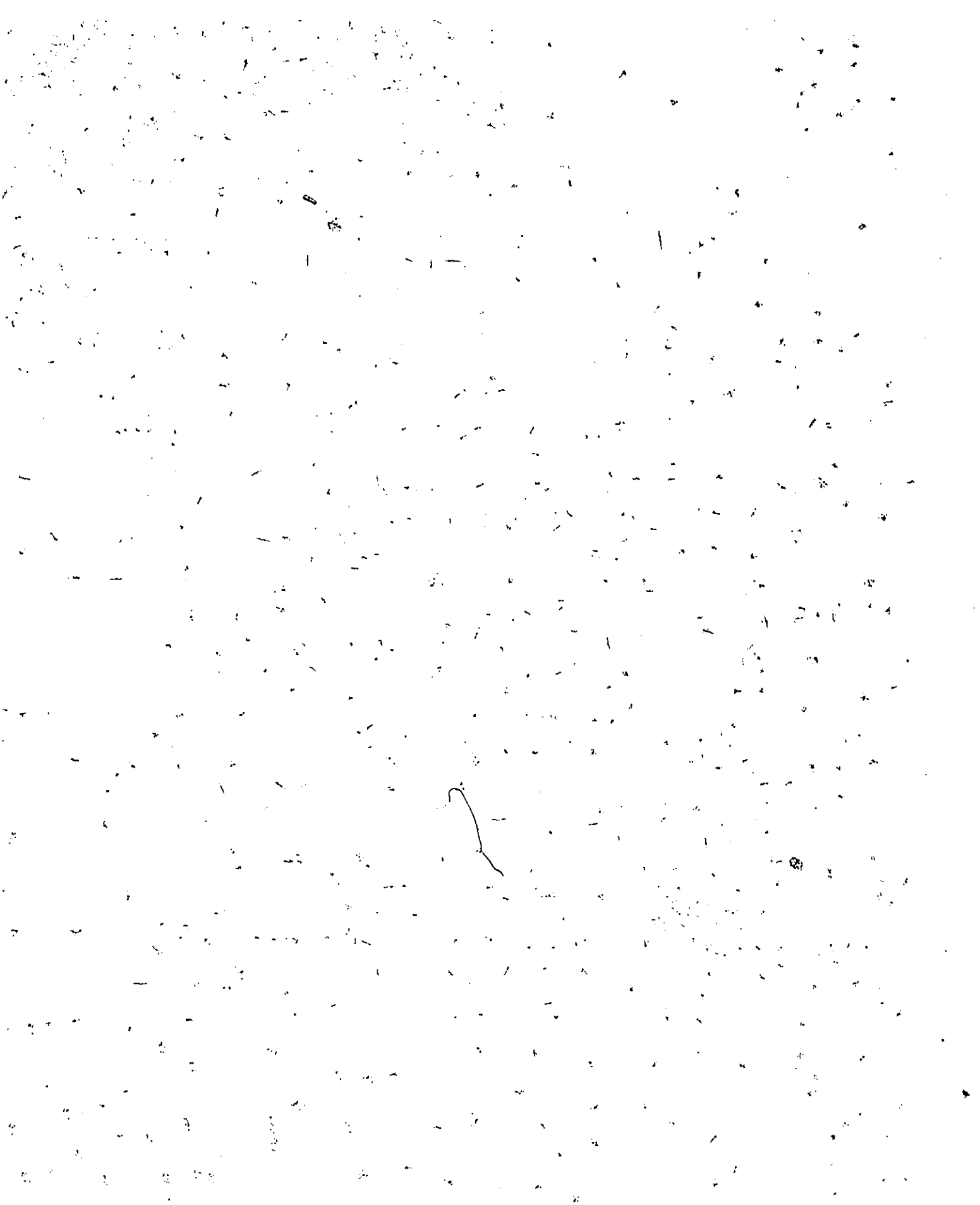
Constraints Declarations				
PDU RRC (Receive Ready Command)				
List Name	FIELD NAME			Comments
	ADdress	P	NR	
S0	'H03	'B0	?	NR is free variable.
S1	'H03	'B1	?	

Constraints Declarations				
PDU RNRC (Receive Not Ready Command)				
List Name	FIELD NAME			Comments
	ADdress	P	NR	
S0	'H03	'B0	?	NR is free variable.
S1	'H03	'B0	?	
S2	'H01	'B1	'B001	
S3	'H01	'B1	'B000	

Constraints Declarations				
PDU REJC (REJect Command)				
List Name	FIELD NAME			Comments
	ADdress	P	NR	
S0	'H03	'B0	?	NR is free variable.
S1	'H03	'B1	?	
S2	'H03	'B0	'B001	
S3	'H03	'B1	'B001	
S4	'H03	'B0	'B000	

Constraints Declarations			
PDU DTE_DISC (DTE initiated DISConnect)			
List Name	Field Name		Comments
	ADdress	P	
S1	'H01	'B1	Upper tester implementation is required to generate this PDU

Constraints Declarations				
PDU SARM (Set Asynchronous Response Mode)				
List name	Field name		Control field	Comments
	ADdress	P		
S0	'H03	'B0	'H0F	Undefined frame in LAPB



BIBLIOGRAPHY

Barto84 Bartoli, P. D. et al, The X-Series recommendations for public data networks (1981 - 1984), Jour. of the Telecommunication Networks, 3(3), pp. 159-193, fall 1984.

Berth82 Berthelot, G., Terrat, R., Petri Net Theoru for the Correctness of Protocols, Int. Workshop on Protocol Specification, Testing and Verification, North Holland Pub. Co. 1982.

BocCh77 Bochmann, G. V., Chung, R. J. A formalized specification of HDLC classes of procedures, Proceedings of the National Telecommunications Conference, Los Angeles, Calif., Dec. 1977, pp. 03A:2.1-2.11.

Bochm78 Bochmann, G. V. Finite state description of communicating protocols, Computer Networks, Vol. 2, Oct. 1978, pp. 361-372.

Bochm80 Bochmann, G. V. A hybrid model and the representation of communication services Computer Network Architecture and Protocols, (Ed.) Green, P. J., Plenum Press, New York, 1980, chapter 23.

Bochm82 Bochmann, G. V., et. al., Some experience with the use of formal specifications, 2nd Workshop on Protocol Specification Testing and Verification, 2nd IFIP, North Holland Publ., 1982.

BraJo78 Brand, D., Joyer Jr., W. H. Verification of protocols using symbolic execution, Computer Networks, Vol. 2, Oct. 1978, pp. 351-360.

Brink85 Brinksma, E. A tutorial on LOTOS, International Workshop on Protocol Specification Testing and Verification, North Holland, IFIP 1985.

Brink86 Brinksma, E., et. al. LOTOS specification, their implementations and their tests, International Workshop on Protocol Spec., Testing and Verification, 6th IFIP, North Holland Publ., 1986.

CacSC686 Expert's Contribution to Canadian Ad-hoc Group on Conformance, University of Ottawa. X.25 DTE Data Link Layer Test Suite in TTCN, Document number TIES44, SC6 (national), WG1, Montreal September 1986. To be presented at TC97/ SC6/ WG1 Tokyo, October 1986.

CasSi86 Castanet, R., Sijelmassi, R. Methods and semi-automatic tools for preparing distributed testing, Int. Workshop on Protocol Specification, Testing and Verification, 6th IFIP, Montreal, pp. 5-53, 5-68, June 1986.

CCITX25a CCITT Recommendation X.25, Orange Book Tome VIII.2, October 1976.

CCITX25b CCITT Recommendation X.25, Interface Between Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment (DCE) for Terminals Operating in the Packet Mode on Public Data Networks, Geneva, 1980.

CCITX25c CCITT Recommendation X.25, Interface Between Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment (DCE) for Terminals Operating in the Packet Mode on Public Data Networks (Revised), 1984.

CheHo81 Chen, Z. C., Hoare, C. A. R. Correctness of communication protocols, Proc. of 1st IFIP Int. Workshop on Protocol Specefication, Testing and Verification, Teddington, U.K., May 1981.

Chow78 Chow, T. S. Testing software designed by finite state machines IEEE Trans. on SE, Vol. SE 4, No. 3, May 1978, pp. 178-187.

DP88821 ISO Draft Proposal X.25-DTE Conformance Testing - Part 1, Principles. DP 8882, ISO/TC 97/SC 6 WG 1, March 1986.

DP88823 ISO Draft Proposal Draft Proposal X.25-DTE Conformance Testing - Part 3, Packet Level Conformance Suite. DP 8882/3, ISO/TC 97/SC 6 WG 2.

Gill62 Gill, A. Introduction to the Theory of Finite State Machines, McGraw-Hill, New York, 1962.

Gouda85 Gouda, M. G. Modeling physical layer protocol using communicating finite state machines, Proceedings of Ninth Data Communication Symposium, British Columbia, Sept. 1985, pp. 54-62.

Hoare85 Hoare, C. A. R. Communicating Sequential Processes, Prentice - Hall International, U.K., 1985, Chapter 3.

Hornb86 Hornbeek, M. W. A., Probert, R. L., Ural, H. Design and development of a standardized conformance test suite for X.25 DTE testing, Conference Proceedings of GLOBECOM'86, 1986.

Hugas84 Hugashino, T., et. al., An algebric specification of HDLC procedures and its verification, IEEE Trans. on SE, Vol. SE 10, No. 6, Nov. 1984.

ISO7776 ISO International Standard IS-7776, Description of the 1984 X.25 LAPB-Compatible DTE Data Link Procedures, ISO TC 97/ SC6, 1984.

ISO8208 ISO International Standard IS-8208, X.25 Packet Level Protocol for Data Terminal Equipment, ISO TC 97/ SC6, 1984.

ISO909 ISO N909 Working Draft for OSI Conformance Testing Methodology and Framework. ISO/TC 97/SC 21 N909 Project No. 97.21.23, November 1985.

Kanun85 Kanungo, B. X.25 DTE Conformance Test Requirements, University of Ottawa, Protocols Research Group technical report, Nov. 1985

Kanun86 Kanungo, B., Lamont, L., Probert, R., Ural, H. A useful FSM representation for test suite design and development, International Workshop on Protocol Spec., Testing and Verification, 6th IFIP, North Holland Publ., 1986.

Kohav78 Kohavi, Z. Switching and Finite Automata Theory, McGraw-Hill, New York, 1978.

Lamon86a Lamont L., et al. X.25 DTE Conformance User's Guide, Source: BNR, University of Ottawa, February 1986.

Lamon86b Lamont L. X.25 Link Level Conformance Test Plan. Source: U of Ottawa, June 1986.

Meyer78 Meyers, G. J. Composite /Structure Design, New York: Van Nostrand, 1978.

Milne80 Milner, R. Lecture Notes in Computer Science, Springer - Verlag, Berlin, 1980.

Myers79 Myers, G. J. The Art of Software Testing, John Wiley & Sons, 1979.

N048 ISO Working Paper, Draft DTE X.25 Data Link Layer Test Suites. ISO/TC 97/SC6 /WG1 N048, October 1985.

N639 USA Contribution to SC6, Packet Level Static Conformance Requirements Proposal. ISO TC 97/SC6 /WG2 N640, USA Expert's working paper, January 1986.

N640 USA Contribution to SC6, Protocol Implementation Conformence Statement. ISO TC 97/SC6 /WG2 N640, January 1986.

N3566 ISO Draft Document, X.25 DTE Conformance Testing - Part 3, Packet Level Test Suite. ISO TC 97/SC6 /WG2 N3566[5], March 1985.

N3891 ISO Revised Document, X.25 DTE Conformance Testing - Part 3, Packet Level Test Suite. ISO TC 97/SC6 /WG2 N3891, project 97.6.38.3, February 1986.

Paval85 Paval, J. R. A new approach to the design and construction of protocol testers, NPL Report, DITC 54/85, March 1985.

PavDw84 Paval, J. R., Dwyer, D. J. Some experience of testing protocol implementations, NPL Report, DITC 48/84, Sept. 1984.

RafAn83 Rafiq, O., Ansart, J. P. VADILOC - A protocol validator and its applications, International Workshop on Protocol Specification Testing and Verification, 3rd IFIP, North Holland Publ., 1983.

RaMor85 Razouk, R. R., Morgan, E. T. The P-NUT system: An environment for modeling and analyzing concurrent systems, Technical Report #85-16, Department of Information and Computer Science, University of California, Irvine, July 1985.

RapWe85 Rapps, S., Weynker, E. J. Selecting software test data using data flow information, IEEE Trans. on SE, Vol. SE 11, No. 4, Apr. 1985, pp. 367-375.

Rayn82 Rayner, D. A system for testing protocol implementations, International Workshop on Protocol Specification Testing and Verification, 2nd IFIP, North Holland, 1982, pp. 539-553.

---

[5] Superseded by ISO TC 97/SC6 /WG2 N29 and then by N3891.

Rayne85 Rayner, D. Towards standardized OSI conformance tests, International Workshop on Protocol Specification Testing and Verification, North Holland, IFIP 1985.

SarBo82a Sarikaya, B., Bochmann, G. V. Some experience with test sequence generation for protocols, International Workshop on Protocol Specefication Testing and Verification, North Holland, IFIP 1982, pp. 389-395.

SarBo82b Sarikaya, B., Bochmann, G. V. Synchronization and specification issues in protocol testing, Research Report, Department d'informatique et de recherche operationnelle, Universite de Montreal, Publication #463, December 1982.

Sheri86 Sherif, M. H., Hoover, G. L., Wiederhold, R. P. X.25 conformance testing -- a tutorial, IEEE Communications, Vol. 24, No. 1, Jan. 1986, pp. 16-27.

UraPr83 Ural, H., Probert, R. User guided test sequence generation, International Workshop on Protocal Spec., Testing and Verification, 3rd IFIP, North Holland Publ., 1983.

Vuong86 Vuong, S. T., Cowan, D. D., Hui, D. D. VALIRA - A tool for protocol validation via reachability analysis, IFIP 6th International Workshop on Protocol Specification, Testing and Verification, Montreal, June 1986, pp 2-26 to 2-39.

Weir78 Weir, D. F., Proter, W. E., Dam, X. N. X.25 Test Facilities on DATAPAC Proceedings of the Fourth International Conference on Computer Communications, Koyoto, Japan, 1978, pp. 273-279.

X3S3 Proposed USA Contribution to, SC6/WG1 on X.25 DTE Data Link Layer Test Suites. USA X3S3.4/86-56, March 13, 1986.

YauTs86 Yau, S. S., Tsai, J. P. A survay of software design techniques, IEEE Trans. on SE, Vol. SE12, No. 6, June 1986.

YouCo79 Yourdon, E., Constantine, L. L. Structured Design, Englewood Cliffs, N.J.: Prentice-Hall, 1979.

ZafBr83 Zafiropulo, P., Brand, D. On communicating finite state machines Journal of the ACM, Vol. 30, No. 2, Apr. 1983, pp. 323-324.

Zafir80a Zafiropulo, P., et. al. Protocol analysis and synthesis using a state transition model, Computer Network Architecture and Protocols, (Ed.) Green, P. E. Jr., Plenum Press, New York, 1980.

Zafir80b Zafiropulo, P., West, C. H., Cowan, D. D. and Brand, D. Towards analyzing and synthesizing protocols, IEEE Trans. on Communications, COM-28(4), April 1980, pp. 651-660.

Zimer80 Zimmerman, OSI Reference Model - The ISO model of architecture for Open Systems Connection, IEEE Trans. on Communication, Vol COM-28, No. 4, April 1980.