ETSI TECHNICAL REPORT

ETR 094

November 1993

Source: ETSI TC-MTS

Reference: DTR/MTS-02002

ICS: 33.080

Key words: SDL, ASN.1, conformance testing, OSI

Methods for Testing and Specification (MTS); Guide for the implementation of the ISO/IEC 9646 Conformance Assessment Process (CAP)

ETSI

European Telecommunications Standards Institute

ETSI Secretariat

Postal address: F-06921 Sophia Antipolis CEDEX - FRANCE
Office address: 650 Route des Lucioles - Sophia Antipolis - Valbonne - FRANCE
X.400: c=fr, a=atlas, p=etsi, s=secretariat - **Internet:** secretariat@etsi.fr

Tel.: +33 92 94 42 00 - Fax: +33 93 65 47 16

Contents

Foreword

This ETSI Technical Report (ETR) has been published by the Methods for Testing and Specification (MTS) Technical Committee of the European Telecommunications Standards Institute (ETSI) and its purpose is to provide a guide for the harmonised European implementation of conformance test procedures in line with the ISO/IEC 9646 [1] - [6] OSI conformance assessment processes. This ETR supports an open test environment and acts as a bridge between the ISO/IEC 9646 [1] - [6] methodology and test tool functionality and test laboratory procedures.

ETRs are informative documents resulting from ETSI studies which are not appropriate for European Telecommunication Standard (ETS) or Interim European Telecommunication Standard (I-ETS) status. An ETR may be used to publish material which is either of an informative nature, relating to the use or the application of ETSs or I-ETSs, or which is immature and not yet suitable for formal adoption as an ETS or an I-ETS.

Blank page

1 Scope

In order to have an effective and efficient conformance testing environment in Europe supporting the M-IT-03 [12], it is necessary to have harmonised test environments and testing procedures. To date, the ISO/IEC 9646 [1] - [6] OSI conformance assessment processes have been implemented in different ways using different test tools, making it difficult to evaluate and compare test tool products. Consequently, it is often difficult and costly for different tools supporting the ISO/IEC 9646-5 [4] Conformance Assessment Process (CAP) to be integrated within a test laboratory.

This ETR is a technical guide for the implementation of the ISO/IEC 9646-5 [4] Conformance Assessment Process (CAP). Systems, such as clients systems, test laboratories and test tools supporting the recommendations of this guide support an "Open Test Environment".

The main component of this ETR is an abstract model (see Clause 5) of the ISO/IEC 9646-5 [4] Conformance Assessment Process (CAP) described in SDL/GR. The model breaks down the CAP into a number of individual sub-tasks and identifies relevant information flows between processes and between processes and clients. Each information flow in the model is called a **conformance data object** and its data is defined using ASN.1 type notation [13]. Since these data objects can be implemented in different ways within real test systems an interchange format is specified (see Clause 6) for certain conformance data objects thought suitable for exchange between test environments for further processing. The ETR also provides a Test Tool Support Statement Proforma (see Clause 7) which allows test tool developers to indicate support for CAP processes and any interfaces that support the exchange of conformance data objects.

This ETR can be seen as a bridge between the ISO/IEC 9646 [1] - [6] methodology on the one hand and test tool functionality and test laboratory procedures on the other. It provides guidelines which are likely to contribute to harmonising the implementation of ISO/IEC 9646 [1] - [6], while providing a basis for easier evaluation/comparison of test tools, interchange of a number of conformance data objects in machine processable format, and ultimately re-use of generic functions from one technical area to another. It should be noted that the CAP model is an abstraction which allows a range of different realisations. It is not the intention of this ETR to impose any constraints on test tool functionality or identify any specific distribution of functions within test tools.

2 References

For the purpose of this ETR the following references apply.

[1] ISO/IEC 9646-1:1991: "Information technology -Open Systems Interconnection - Conformance testing methodology and framework - Part 1: General concepts".

[2] ISO/IEC 9646-2:1991: "Information technology -Open Systems Interconnection - Conformance testing methodology and framework - Part 2: Abstract test suite specification".

[3] ISO/IEC 9646-4:1991: "Information technology -Open Systems Interconnection - Conformance testing methodology and framework - Part 4: Test realisation".

[4] ISO/IEC 9646-5:1991: "Information technology -Open Systems Interconnection - Conformance testing methodology and framework - Part 5: Requirements on test laboratories and clients for the conformance assessment process".

[5] ISO Draft International Standard (DIS) 9646-6.

[6] ISO IEC/JTC1/SC21/N 7451: "ISO Committee Document (CD) for ISO/IEC 9646-7, 20 November 1992".

[7] CPS Forum Technical Framework Specification, Issue 2.1, October 91.

[8] ITU-TS Recommendation Z.100 (1988): "Functional Specification and Description Language - SDL" (Annexes A to E).

[9] ETR 022 (1992): (EWOS ETG009): "Advanced Testing Methods (ATM); Vocabulary of terms used in communications protocols conformance testing".

[10] ETR 040 (1992): (EWOS ETG016): "Advanced Testing Methods (ATM); Profile test specifications and conformance test reports".

[11] ISO/IEC 8613 (1989): "Information Technology - Text and Office Systems - Office Document Architecture (ODA) and Interchange Formats (ODIF)".

[12] M-IT-03, Edition 1, April 1989.

[13] ISO/IEC 8824 (1990): "Information technology - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1)".

NOTE: The vote for edition 2 of standards [1] - [4] closed on 18th May 1993 and the vote for standard [6] closed on 27th April 1993. At the time of publishing this ETR, the ISO central secretariat awaits the final text of these standards prior to publication.

3 Definitions

For the purpose of this ETR all definitions from ISO/IEC 9646 part 1,2,4,5,6,7 [1] - [6] from ETR 022 [9] and from ITU-TS Recommendation Z.100 [8] apply. In addition, the following definitions apply:

Conformance data object: the model of a flow of information between two processes or between the test laboratory and its client in the course of a Conformance Assessment.

Open Test Environment: an Open Test Environment is defined as a Test Environment which supports interchange of one or more of the conformance data objects for which this ETR provides a machine processable format.

4 Abbreviations

For the purpose of this ETR the following abbreviations from ISO/IEC 9646 parts 1,2,4,5,6,7 [1] - [6] apply:

ATM	Abstract Test Method/Advanced Testing Methods
ATS	Abstract Test Suite
BIT	Basic Interconnection Test
ETG	EWOS Technical Guide
ETR	ETSI Technical Report
ICS	Implementation Conformance Statement (Protocol or Profile)
IPRL	ISP Requirements List
ISP	International Standardised Profile
IUT	Implementation Under Test
IXIT	Implementation eXtra Information for Testing
MOT	Means of Testing
PCTR	Protocol Conformance Test Report
PETS	Parameterized Executable Test Suite
PICS	Protocol Implementation Conformance Statement
PIXIT	Protocol Implementation eXtra Information for Testing
SCS	System Conformance Statement
SUT	System Under Test
TMP	Test Management Protocol

In addition, the following abbreviations apply:

ATC	Abstract Test Case
CAP	Conformance Assessment Process
CFL	Conformance Log
CL_C	Client Checklist
CL_C_PF	Client Checklist proforma
PICS_PF	Protocol Implementation Conformance Statement Proforma
PIXIT_PF	Protocol Implementation eXtra Information for Testing Proforma
PTS	Parameterized Test Suite
SA	Selection Agreements
SCR_Report	Static Conformance Review Report
SCS_PF	System Conformance Statement Proforma
SCTR	System Conformance Test Report
STS	Selected Test Suite
TC	Test Case
TCP	Test Co-ordination Procedures
TL_C	Test Laboratory Checklist
TMPis	Test Management Protocol Implementation Statement
TMPis_PF	Test Management Protocol Implementation Statement Proforma

5 Abstract model of the ISO/IEC 9646 Conformance Assessment Process (CAP)

5.1 Overview

ISO/IEC 9646 [1] - [6] is a multi-part international standard which specifies a general methodology for testing the conformance of products to OSI specifications. Part 5 of ISO/IEC 9646 [4] defines the **Conformance Assessment Process (CAP),** which is the complete process of accomplishing all conformance testing tasks necessary to enable the assessment of the conformance of an implementation or system to one or more protocol or profile specifications. The CAP is standardised in order to achieve some degree of comparability of test results on similar products tested by different test laboratories.

This ETR describes an abstract model of the CAP defined in ISO/IEC 9646-5 [4] . The model presents a system consisting of the following parts:

a) a number of processes, each designed to meet the specifications for a sub-task of the CAP as defined in ISO/IEC 9646-5 [4], and a description of the processes and their interrelationships in SDL/GR (see subclause 5.3.1); and

b) a data model, defining the content of the data objects exchanged between these processes, using ASN.1 syntax notation as the description language (see subclause 5.3.4).

ISO/IEC 9646-5 [4] defines the CAP by describing the necessary actions to be undertaken by the test laboratory and the client. Although the interactions between the test laboratory and the client are covered, the model addresses the CAP mainly from the perspective of the test laboratory: the necessary processes on the client side are outside the scope of the model. By providing a more formal view of the CAP for the test laboratories, the model facilitates a **common understanding** of the flows of information between sub-tasks in the CAP and the interaction between test laboratories and clients.

The description of the CAP in a formal way requires interpretation of some of the informal definitions in ISO/IEC 9646-5 [4], this is especially the case in parts of the data model describing the data objects exchanged between the CAP sub-tasks inside the test laboratory. Depending on the level of detail that is used in ISO/IEC 9646 [1] - [6] to define the individual data objects, each data object in the model is classified according to the following four categories:

- the structure and content of the data object is defined in ISO/IEC 9646 [1] - [6];

- the content of the data object is defined but not structured in ISO/IEC 9646 [1] - [6];

- the data object is mentioned in ISO/IEC 9646 [1] - [6], but its content and structure are not defined;

- the data object is resulting from an interpretation of the ISO/IEC 9646 [1] - [6] methodology.

All data objects identified by the model are listed in subclause 5.3.3.

The model is an abstraction, it can be used as a guide to implement the CAP in a test laboratory and as a guide to the implementation of tools supporting the CAP. However, it is not intended to constraint test tool architectures or functionality: it does not imply any specific software/hardware architecture or any specific distribution of functionality within test tools.

The formal description of part a) of the model (see above) is limited to single protocol testing to increase the readability; how to apply the model to multi protocol testing is described in subclause 5.3.2, extensions required for profile testing are described in Clause 8.

5.2 Description technique

The description language used to specify the abstract model is SDL Graphical Representation (SDL/GR) [8]. The design of the model and the style in use of SDL is based on the following decisions:

- the SDL **system** is the test laboratory domain;

- the client is modelled as the **system environment**;

- the **processes** are based on the sub-tasks of the CAP as identified in ISO/IEC 9646-5 [4]. These processes are described in the same level of detail as the CAP sub-tasks in ISO/IEC 9646-5 [4].

  The use of SDL usually requires that the behaviour within a process is described in a fully formalised way. In order to increase the readability and the ease of use of the model for non-SDL experts the description of the processes is formal only down to a certain level of detail. For some commonly understood decisions and actions free text instead of detailed SDL specifications has been chosen to express the actions and the questions for the decisions. In addition some semantic requirements are expressed in comments;

- parameterized **signals** are used to provide for the flow of control and information between processes and between the system and its environment. The information is carried by the signal parameters which are instances of **Conformance Data Objects**. These are defined using ASN.1 type notation, rather than using the abstract data type paradigm of SDL, because the ASN.1 type notation is widely used to define complex data types and also commonly understood. Although ASN.1 is being used, the intent here is to define the contents of the conformance data objects, not the syntactic form they may take within a specific test environment.

  The signals together with the corresponding conformance data objects used as signal parameters are listed in subclause 5.3.2. In the SDL model the signal parameters identify instances of the conformance data object which are listed in the parameter list or in the declaration list of the relevant process or procedure.

  As a simplification for readability the term conformance data object is used throughout this ETR to refer to both, definition of the content of a conformance data object (provided by the ASN.1 type notation) and instances of a Conformance Data Object as used in the SDL model;

- it is assumed that some of the processes have access to their own "Document Store" where all data local to the process can be stored and retrieved (e.g. access to a previously defined proforma). This mechanism is not being used for inter-process communication.

5.3 Conformance assessment process for protocol testing

5.3.1 SDL model

In the SDL specification of the model only process instances necessary for single protocol testing are shown to reduce the complexity and thus increase the readability of the SDL specification. Profile testing was not included in the model due to its current status of standardisation. However, an outline of the extensions necessary for multi protocol in the SDL part of the model and profile testing throughout the model can be found in subclause 5.3.2 and Clause 8 respectively.

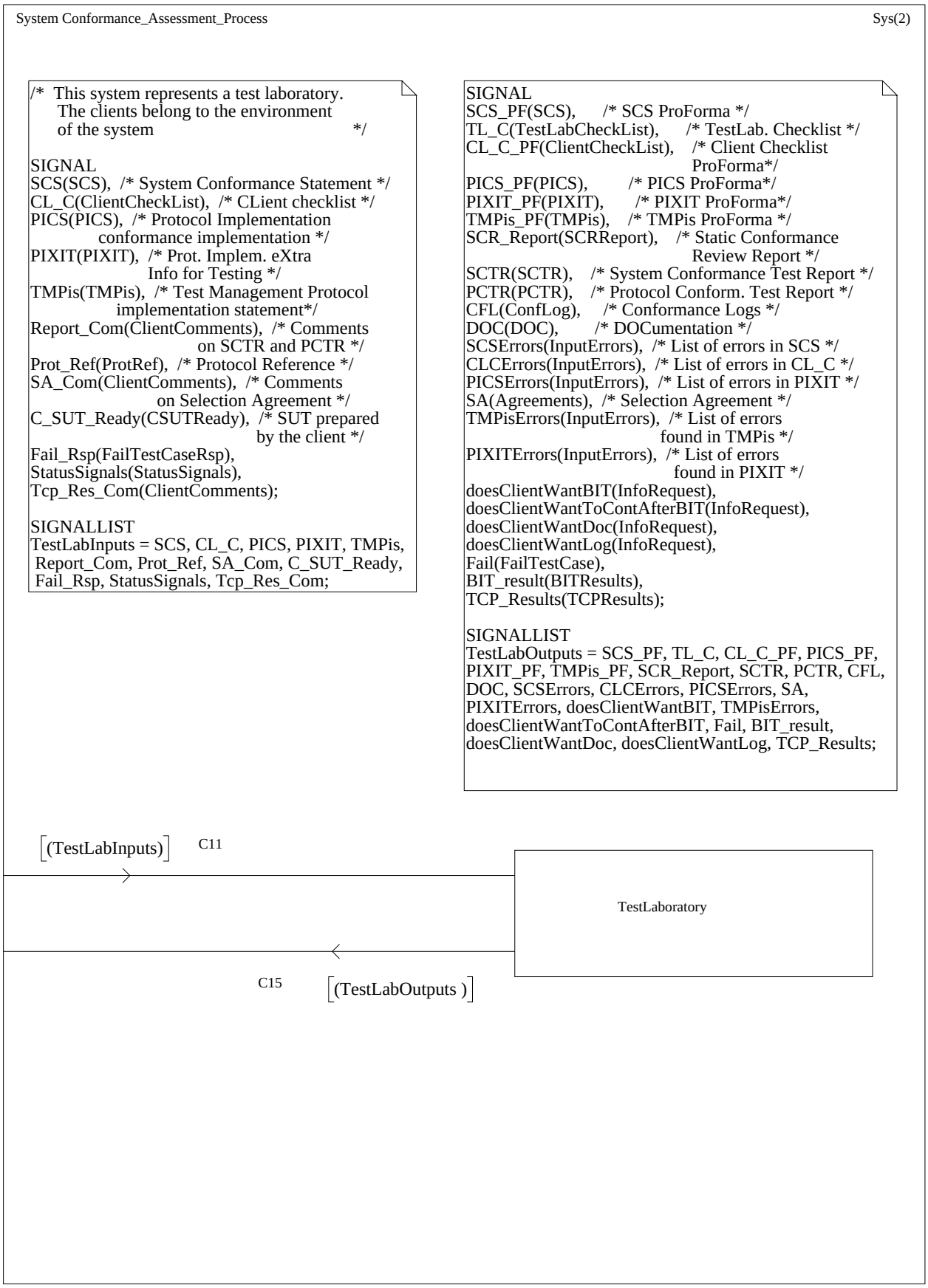


Figure 1: Conformance assessment process for protocol testing SDL: System conformance assessment process

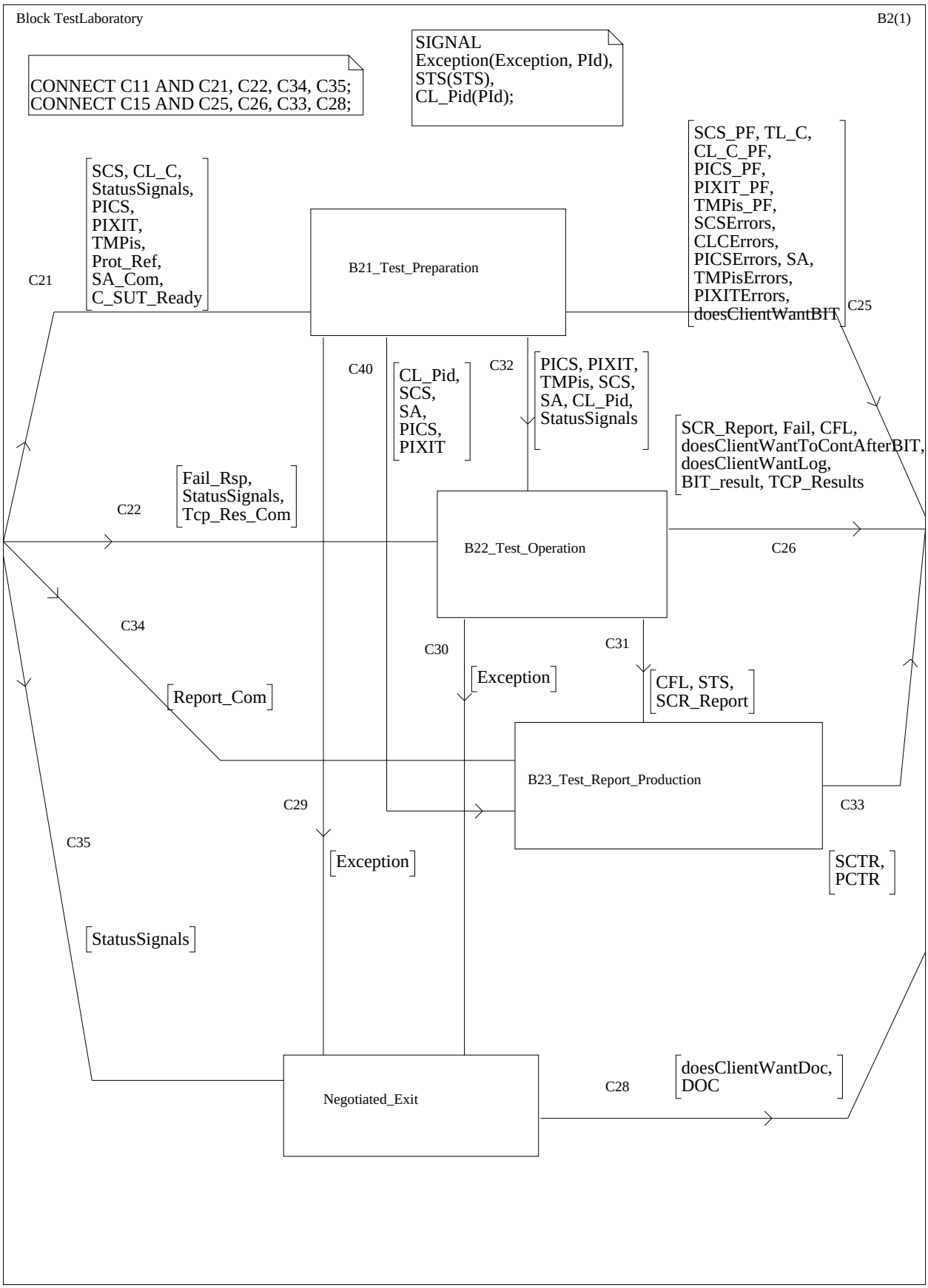


Figure 2: Conformance assessment process for protocol testing SDL: Block test laboratory

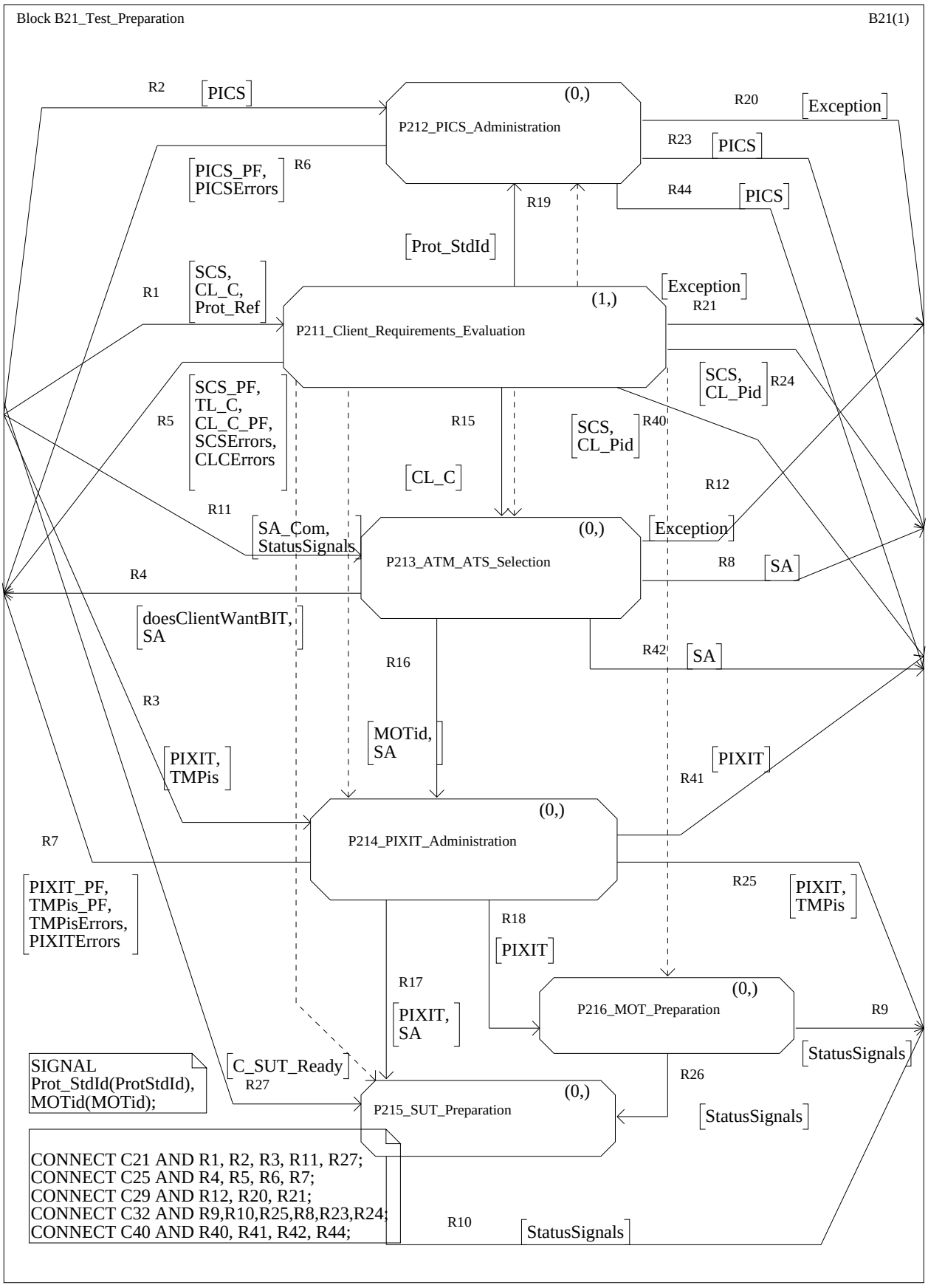


Figure 3: Conformance assessment process for protocol testing SDL: Block B21-test preparation

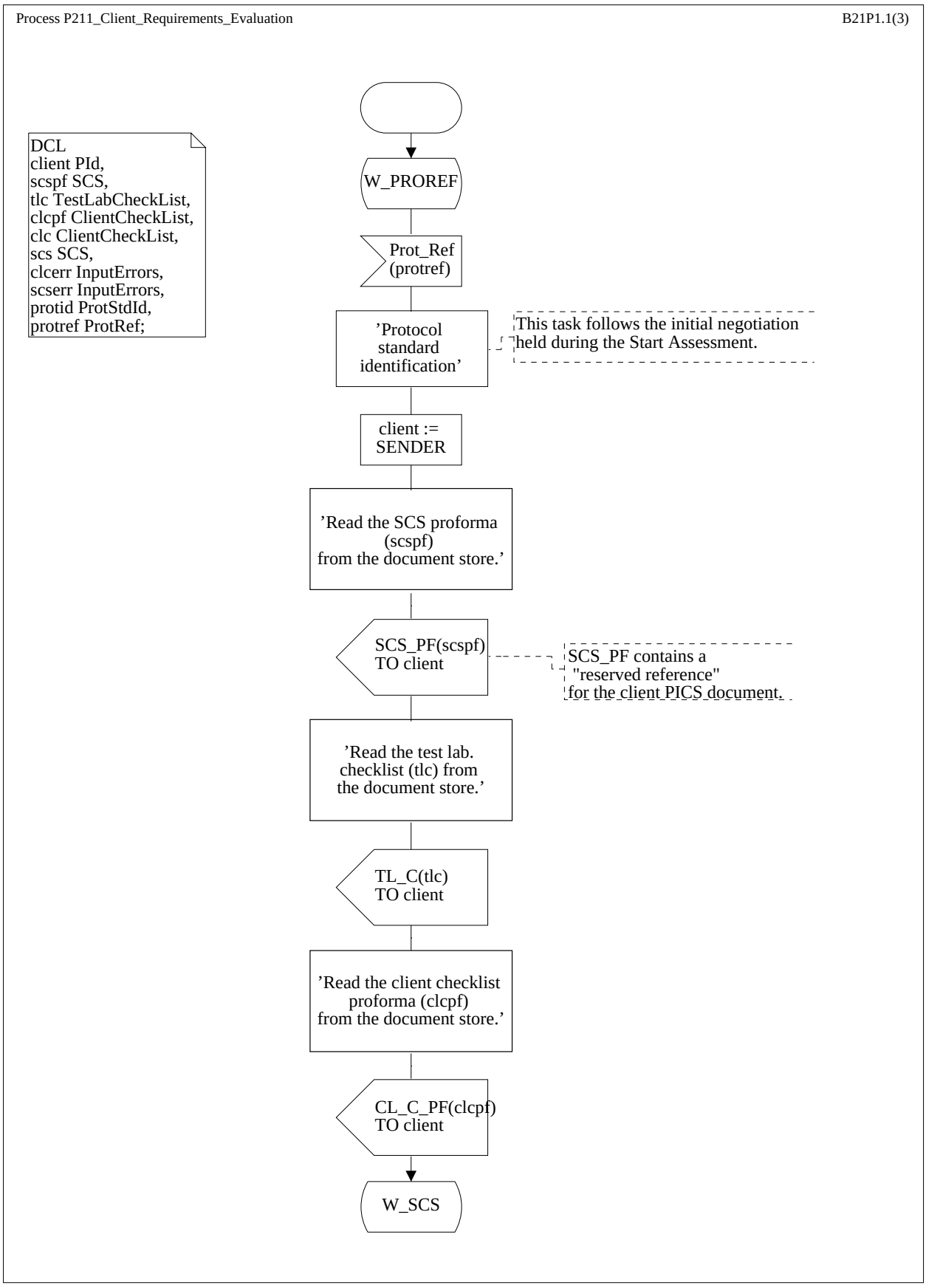


Figure 4 (sheet 1 of 3): Conformance assessment process for protocol testing SDL: Process P211-client requirements evaluation

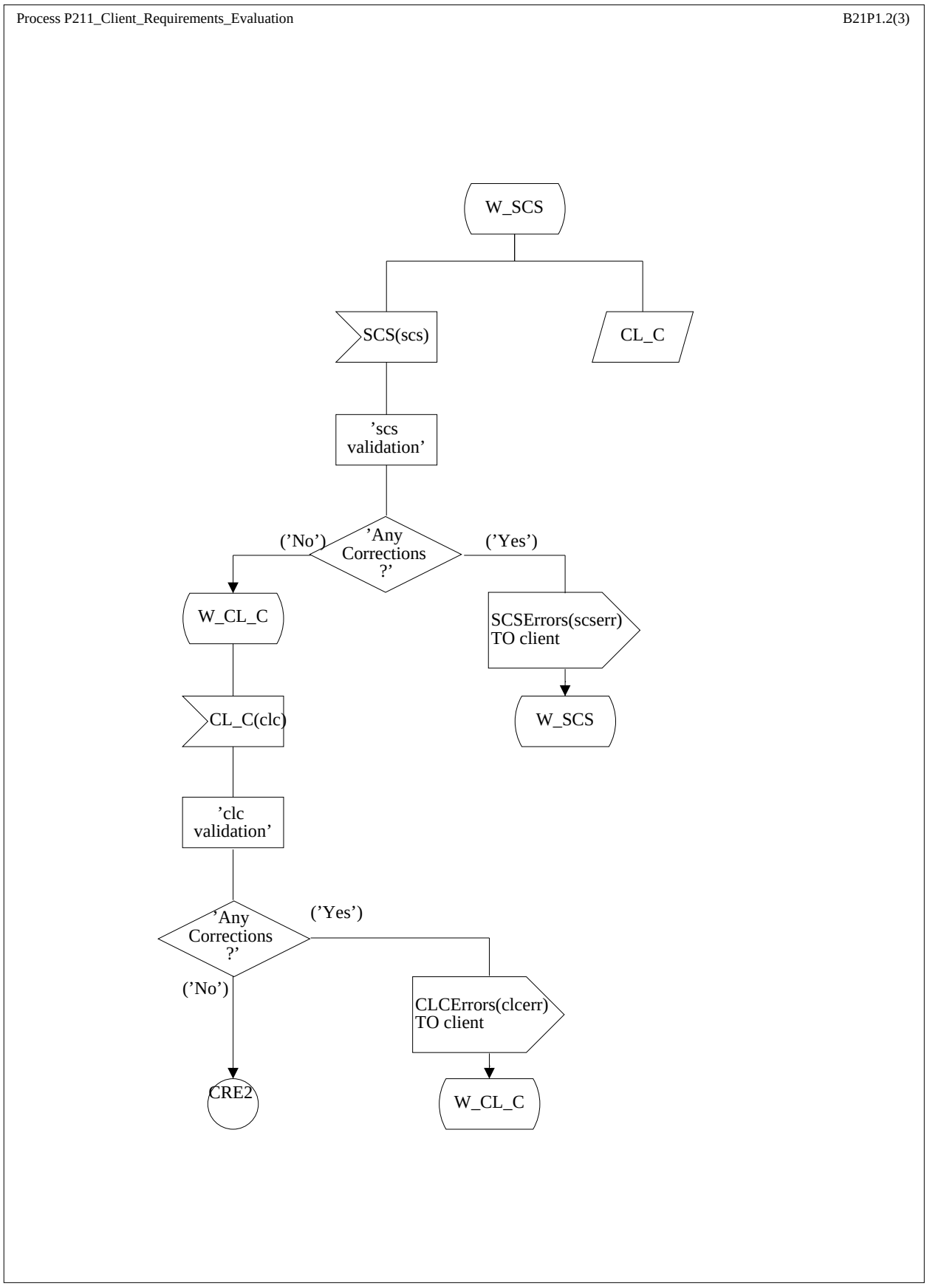


Figure 4 (sheet 2 of 3): Conformance assessment process for protocol testing SDL: Process P211-client requirements evaluation

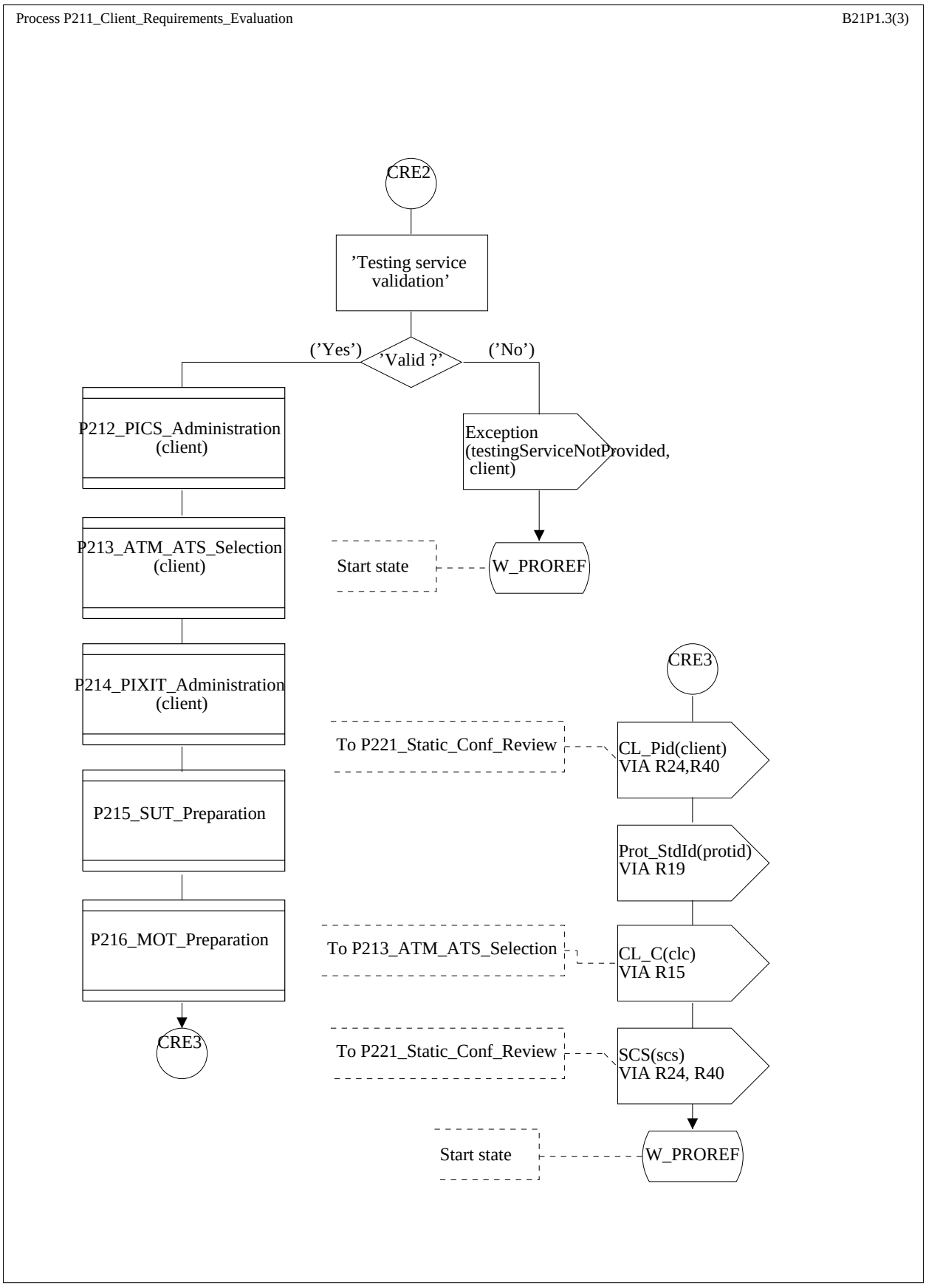


Figure 4 (sheet 3 of 3): Conformance assessment process for protocol testing SDL: Process P211-client requirements evaluation

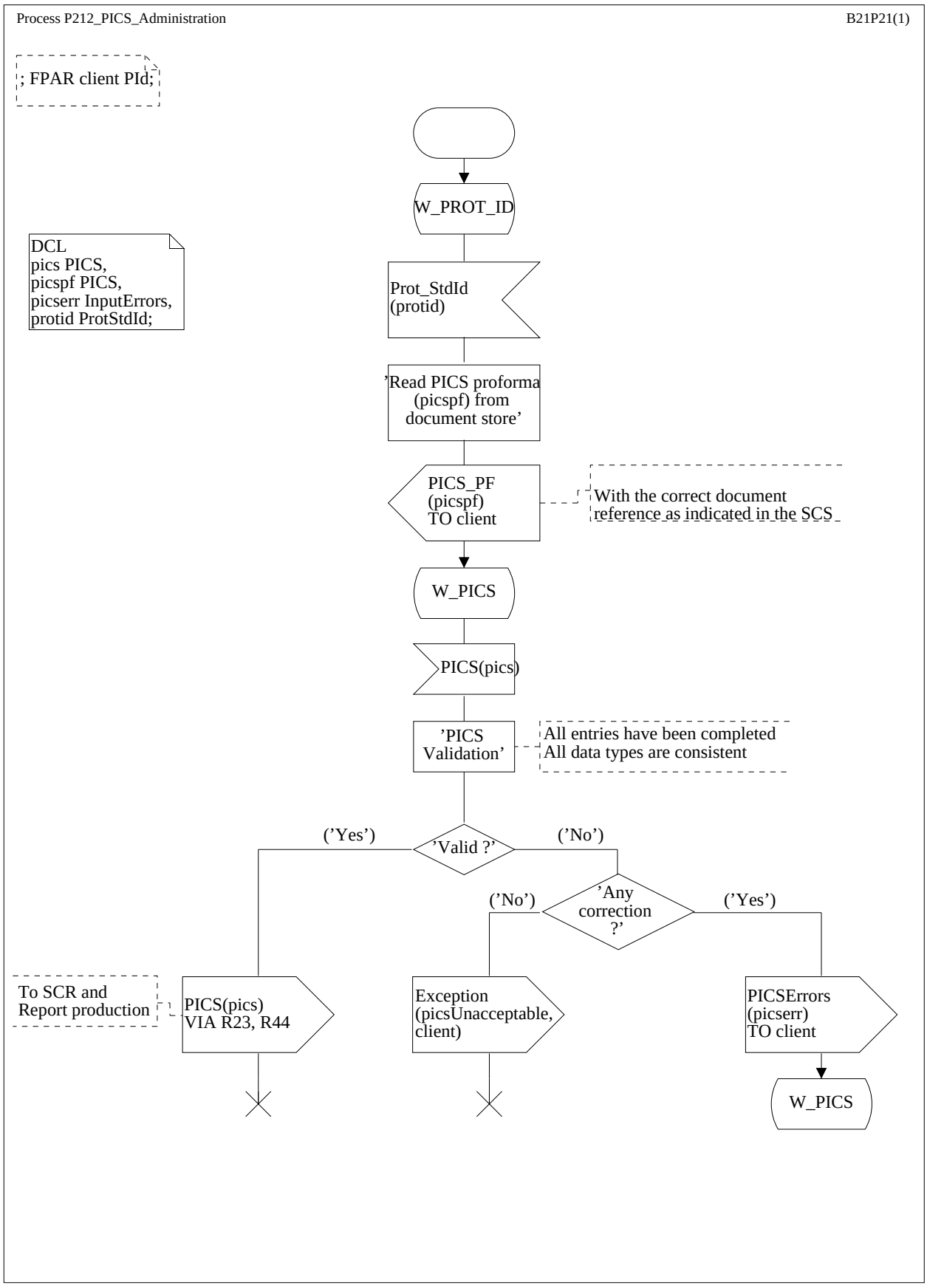


Figure 5: Conformance assessment process for protocol testing SDL: Process P212-PICS administration

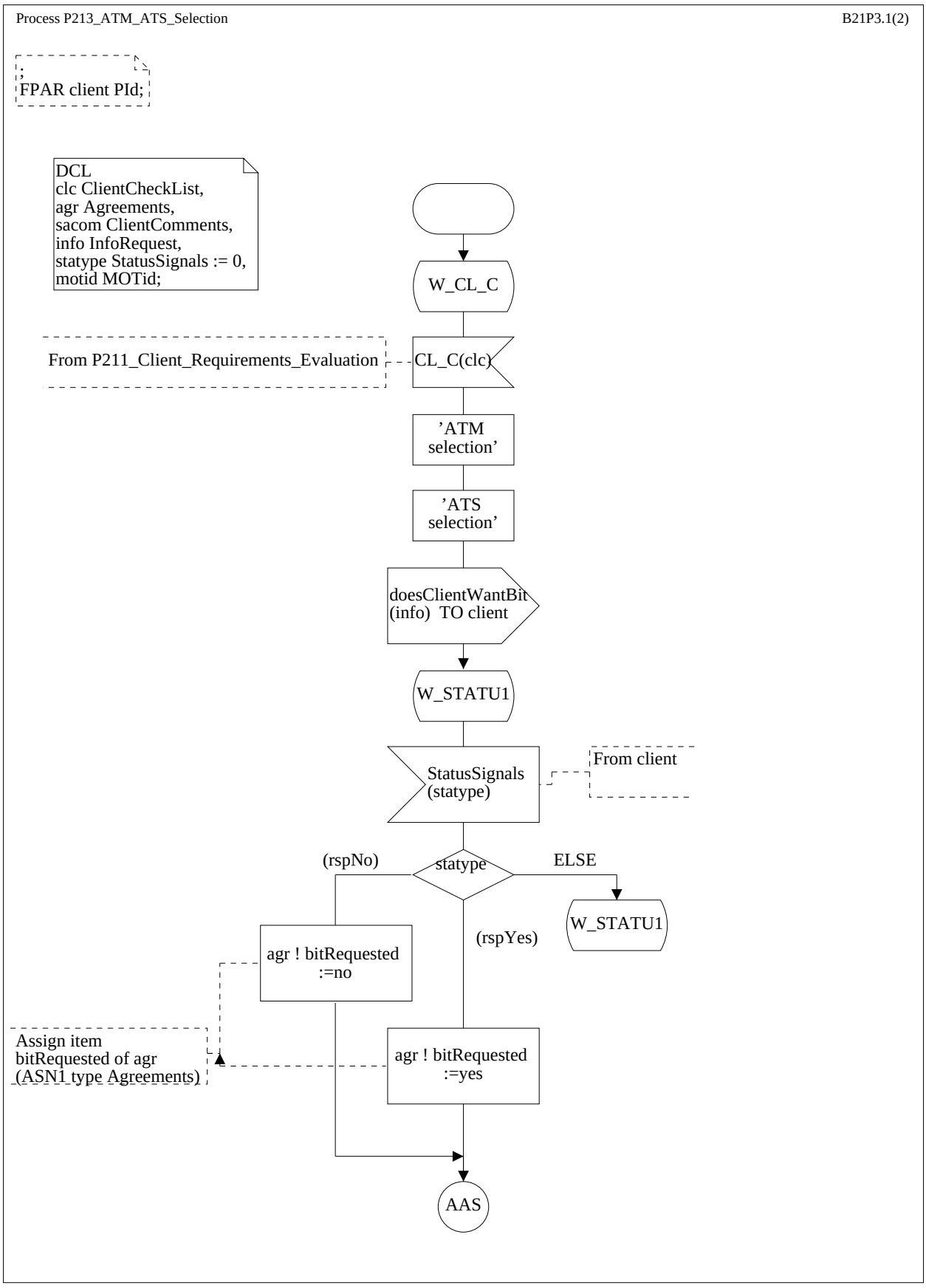


Figure 6 (sheet 1 of 2): Conformance assessment process for protocol testing SDL: Process P213-ATM ATS selection

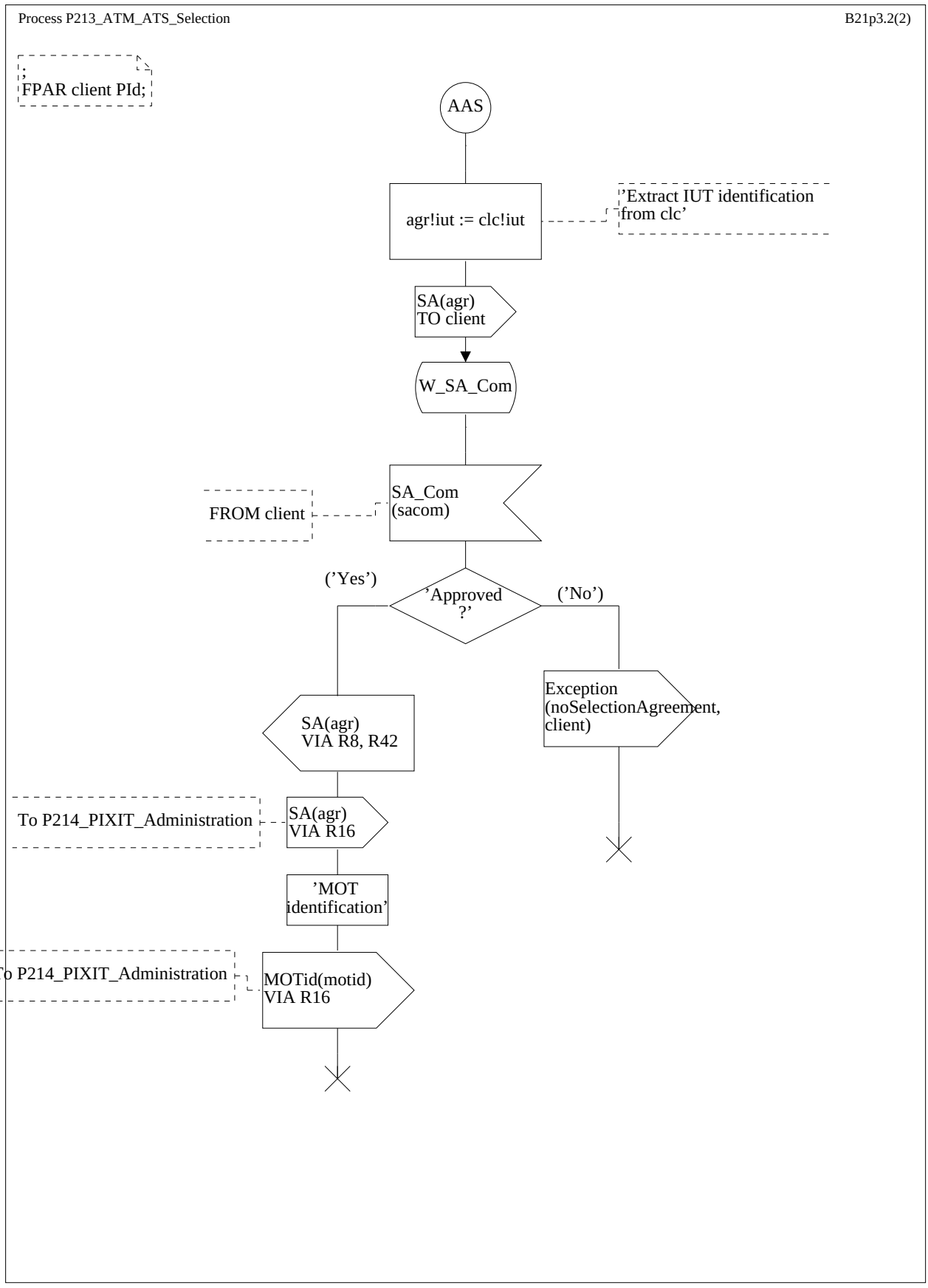


Figure 6 (sheet 2 of 2): Conformance assessment process for protocol testing SDL: Process P213-ATM ATS selection

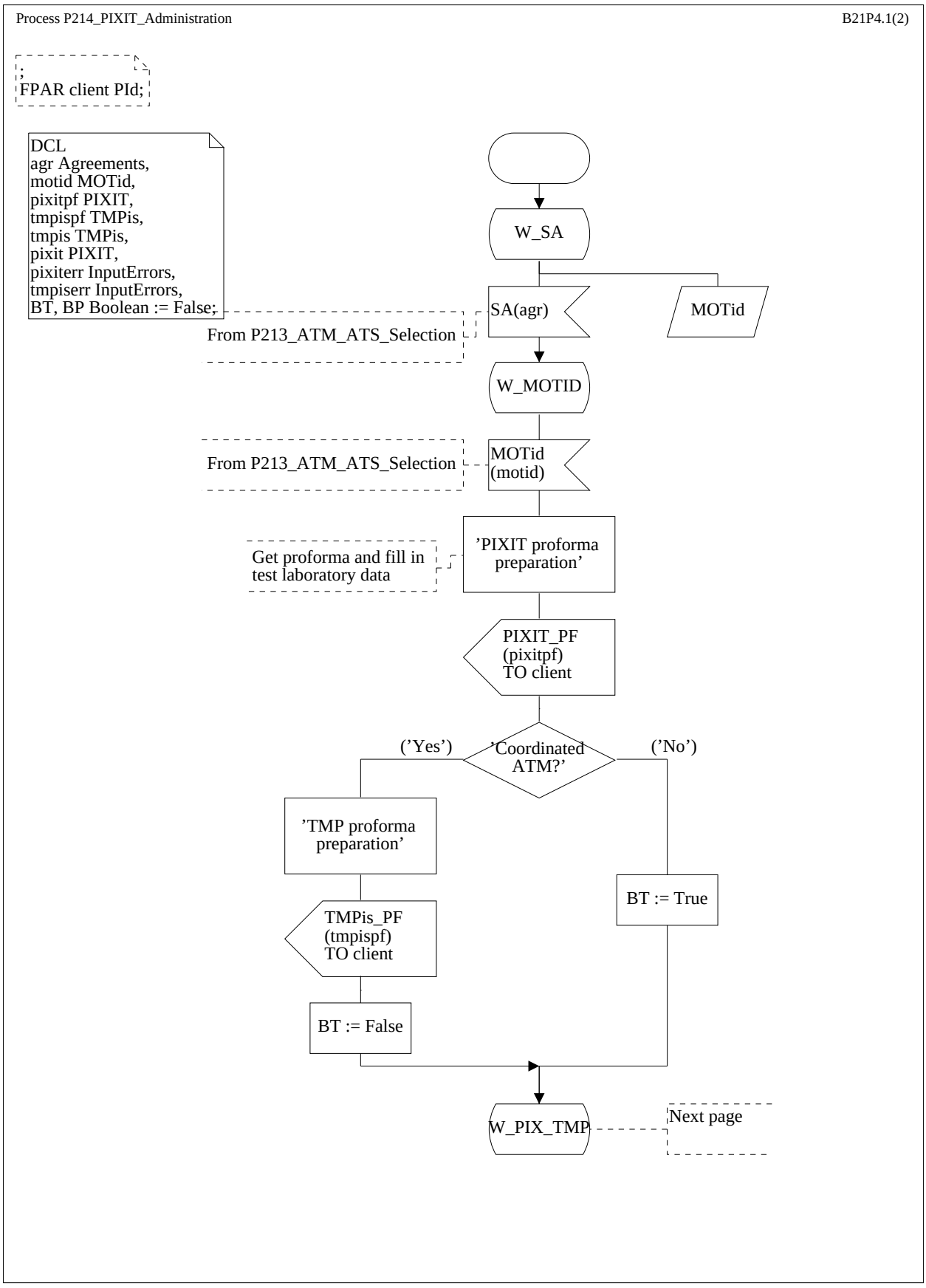


Figure 7 (sheet 1 of 2): Conformance assessment process for protocol testing SDL: Process P214-PIXIT administration

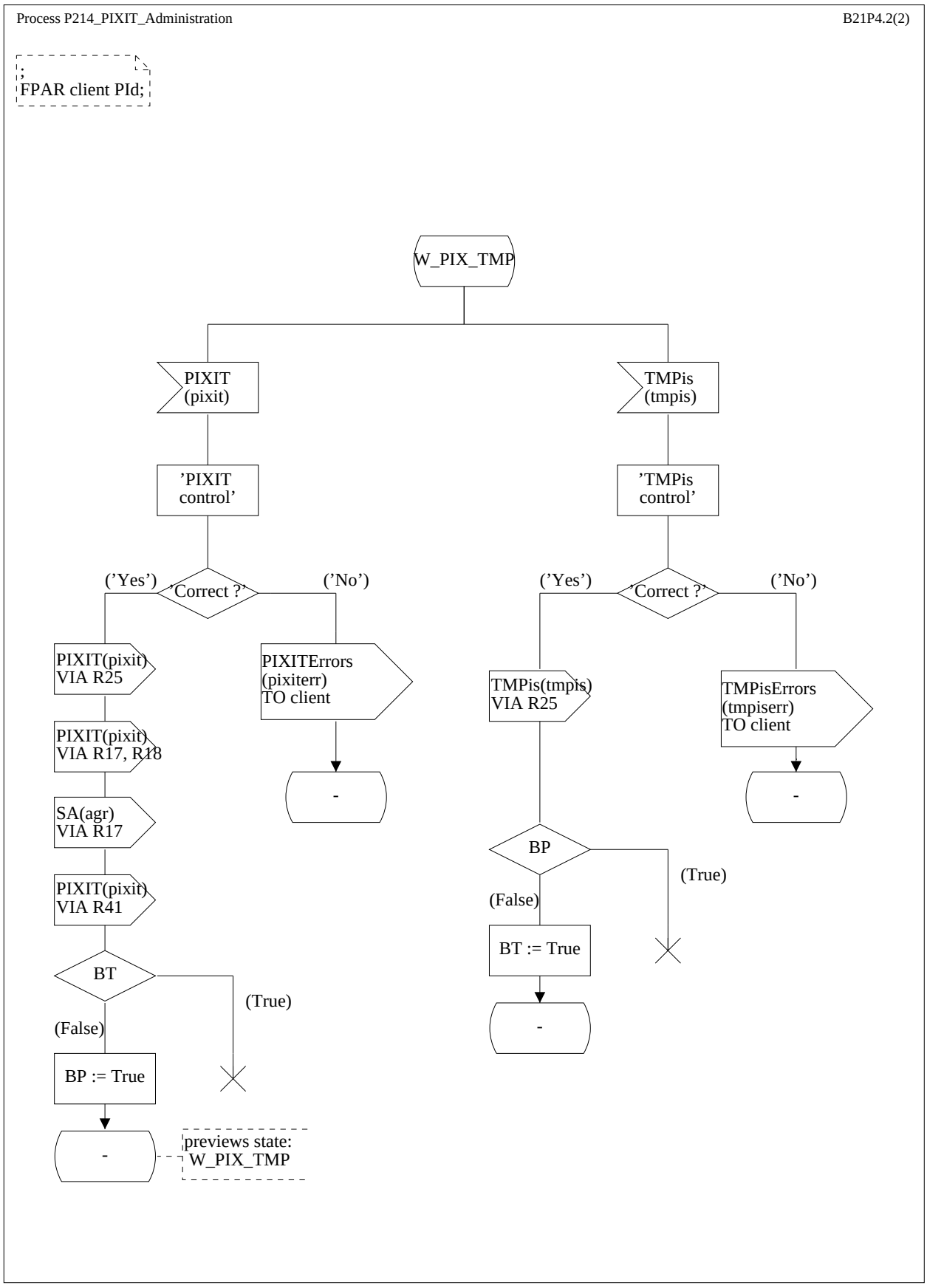


Figure 7 (sheet 2 of 2): Conformance assessment process for protocol testing SDL: Process P214-PIXIT administration

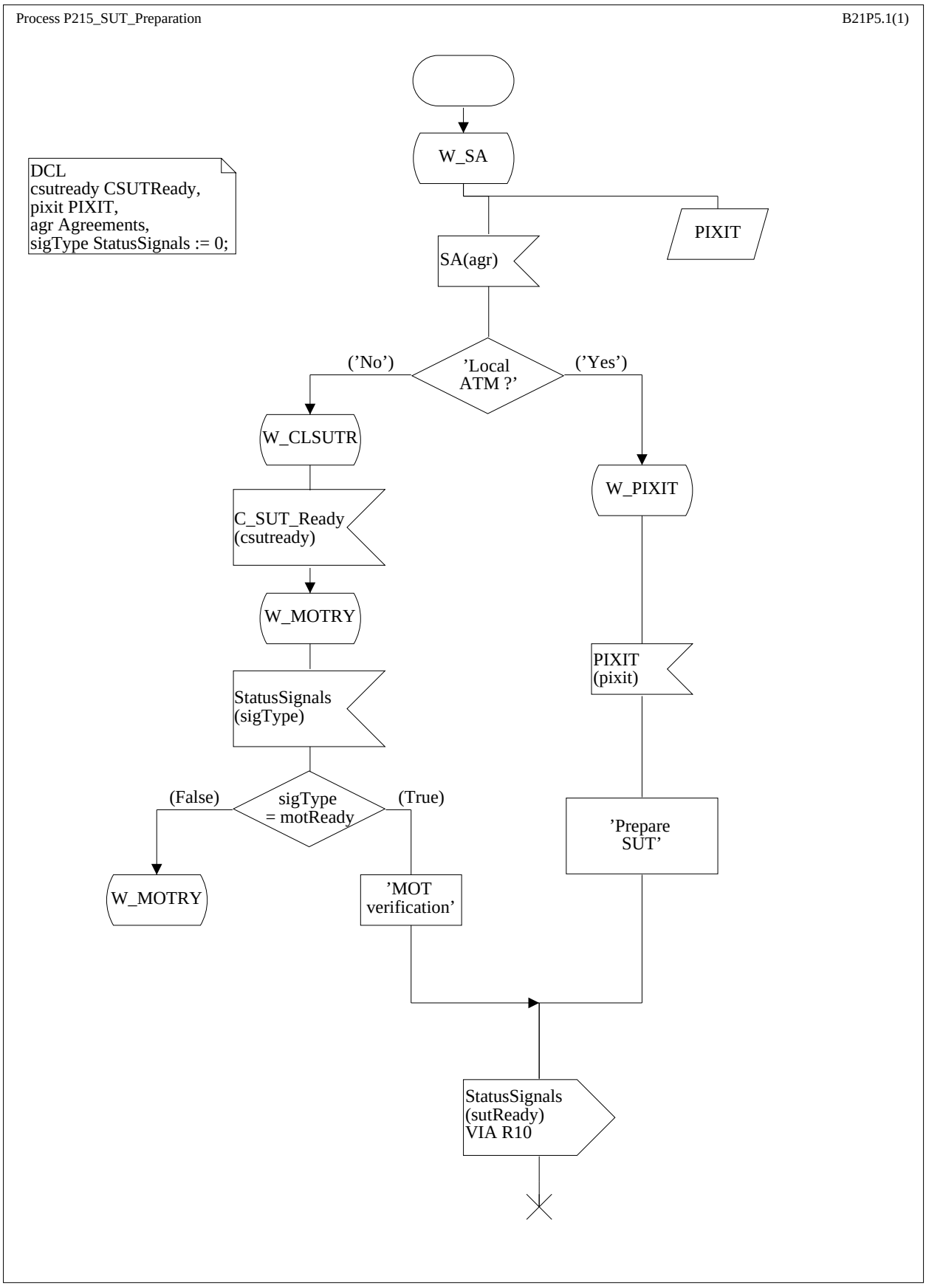


Figure 8: Conformance assessment process for protocol testing SDL: Process P215-SUT preparation

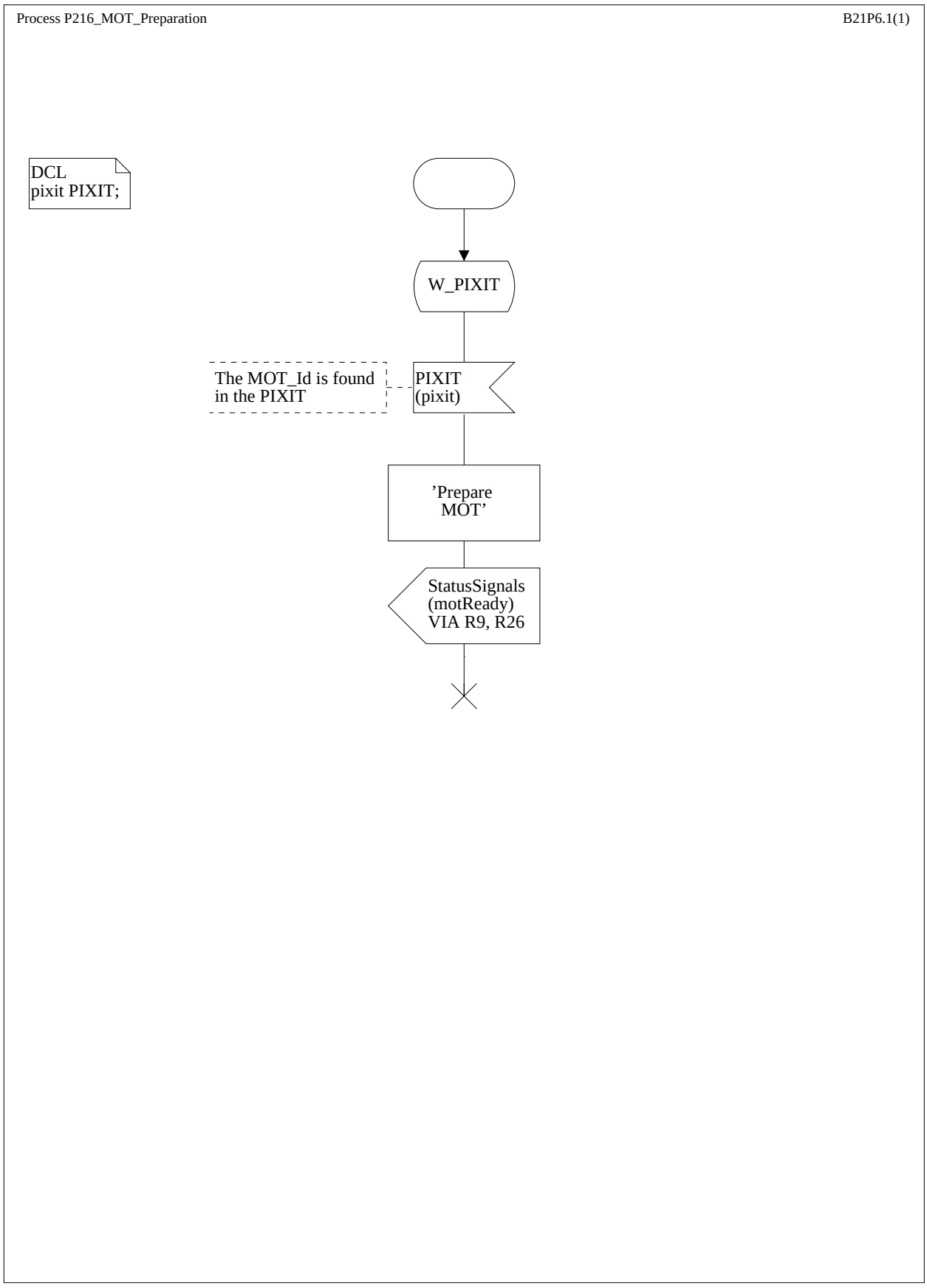


Figure 9: Conformance assessment process for protocol testing SDL: Process P216-MOT preparation

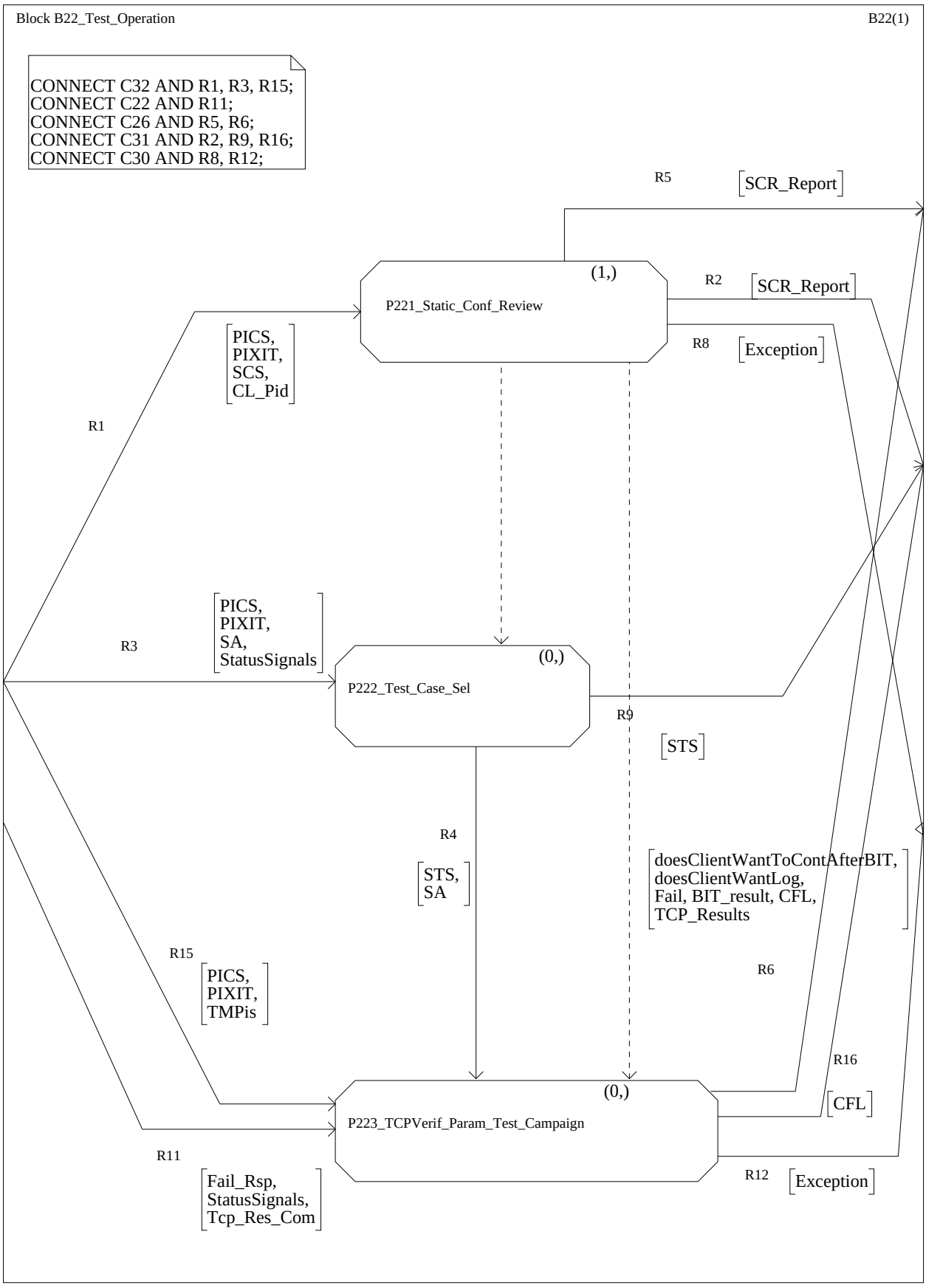


Figure 10: Conformance assessment process for protocol testing SDL: Block B22-test operation

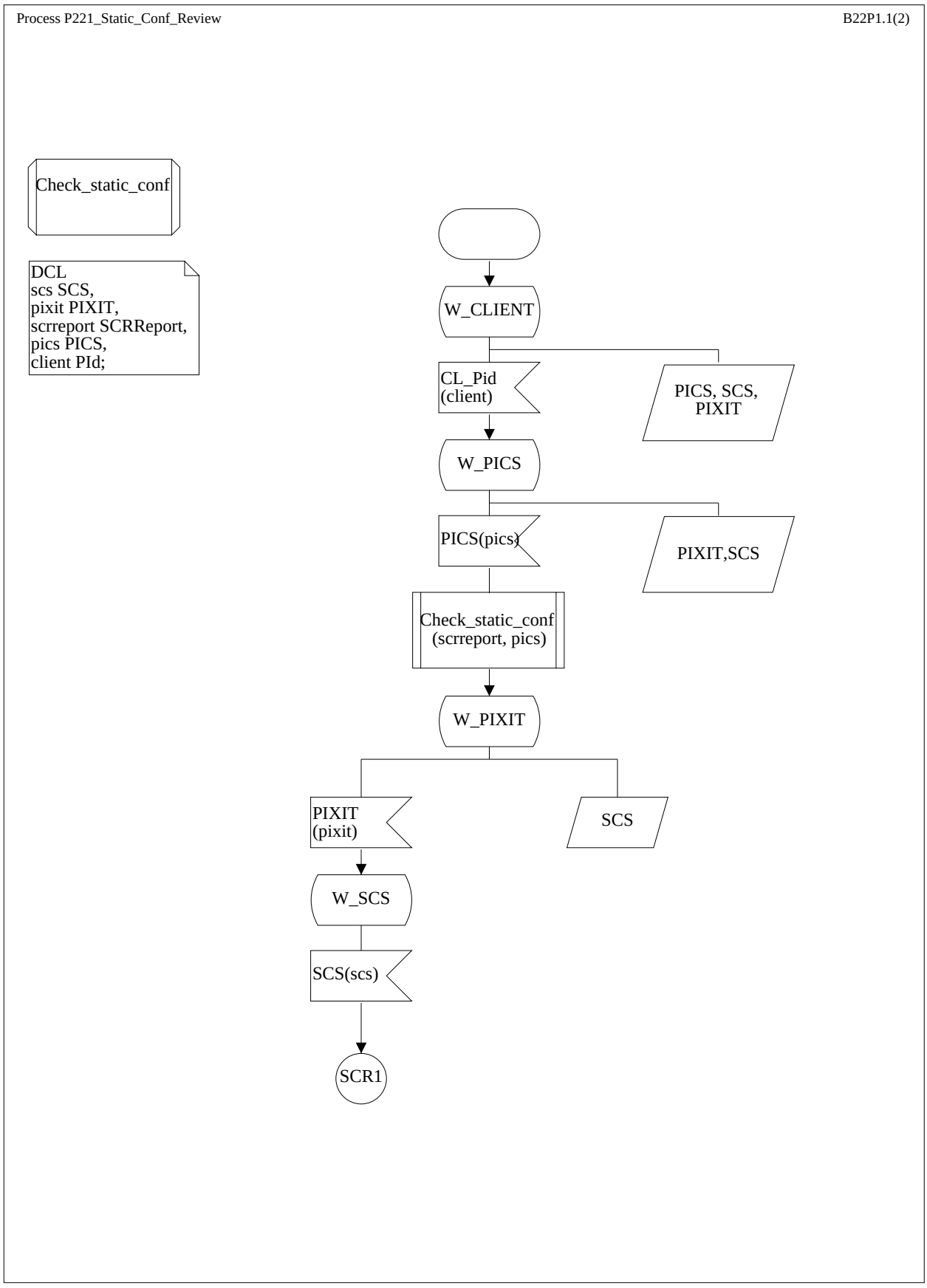


Figure 11 (sheet 1 of 2): Conformance assessment process for protocol testing SDL: Process P221-static conf review

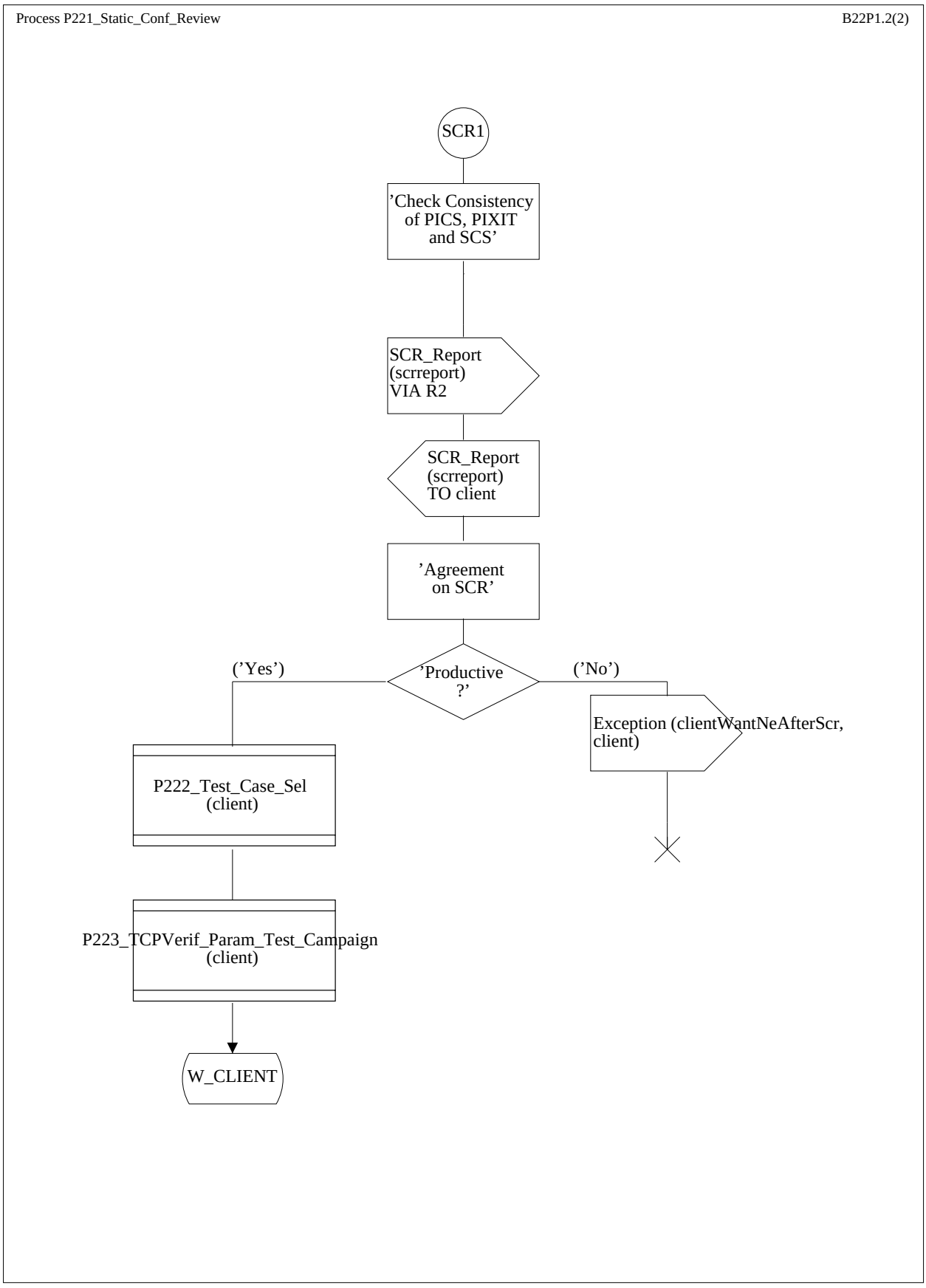


Figure 11 (sheet 2 of 2): Conformance assessment process for protocol testing SDL: Process P221-static conf review

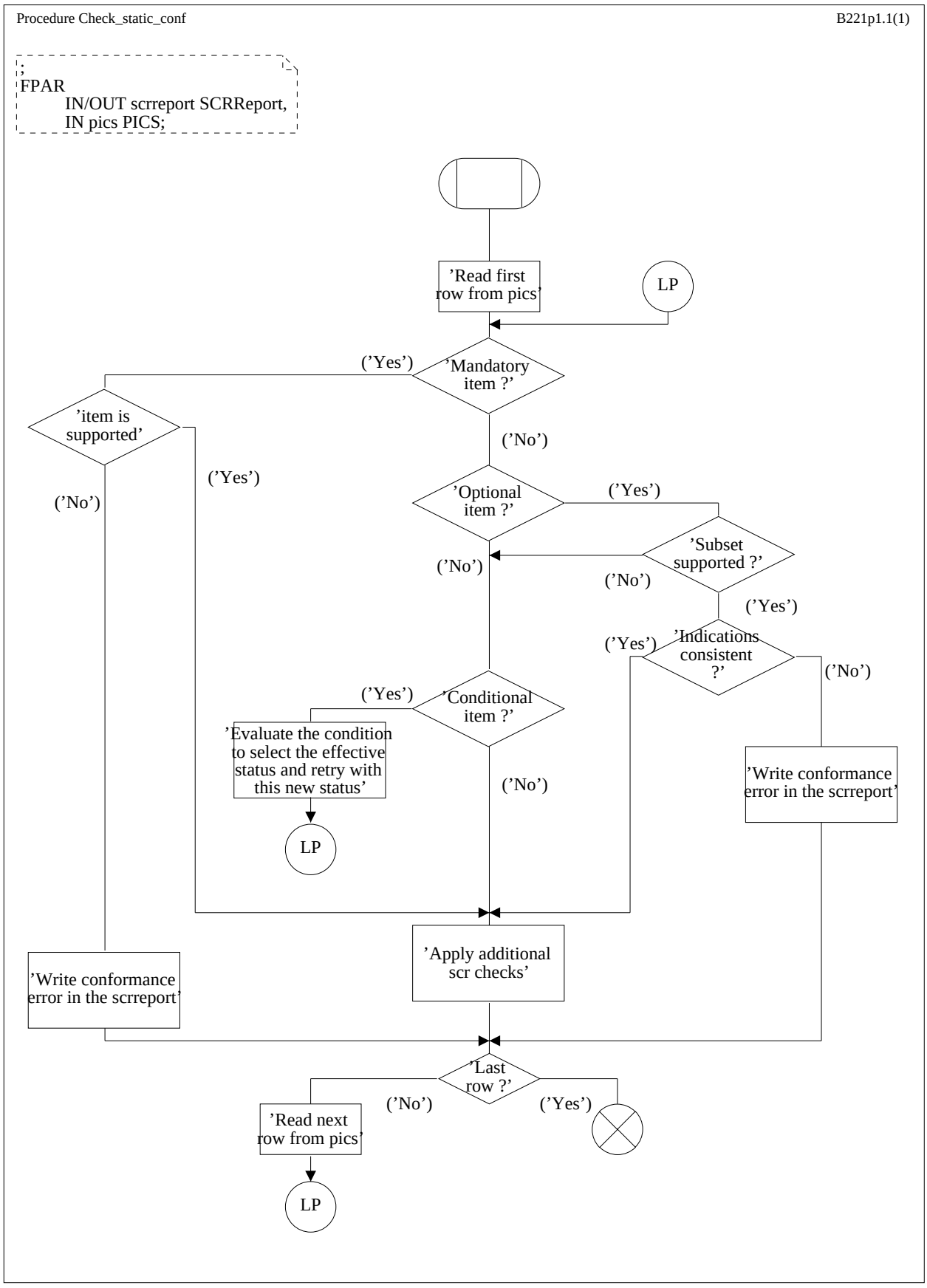


Figure 12: Conformance assessment process for protocol testing SDL: Procedure-check static conf

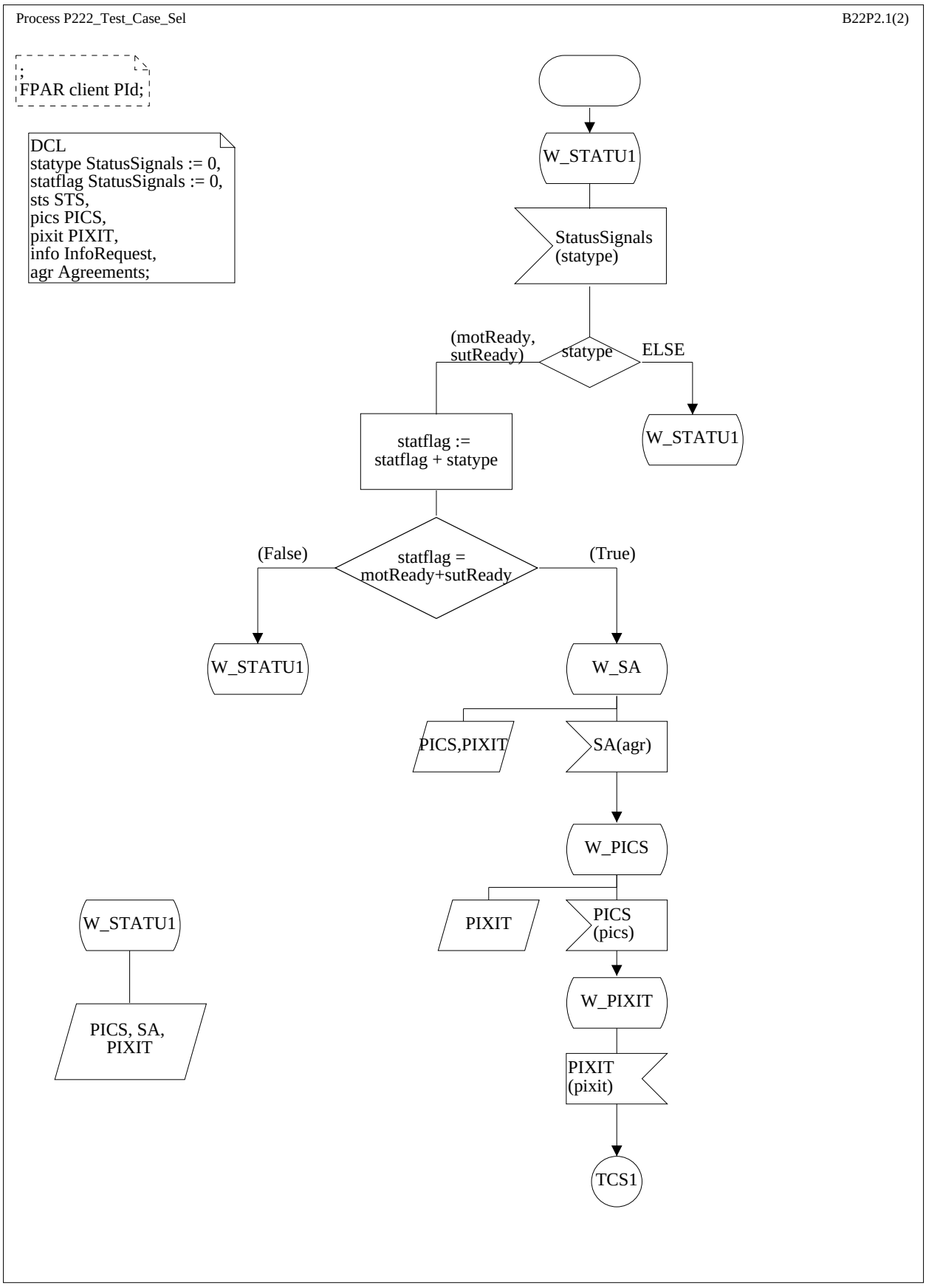


Figure 13 (sheet 1 of 2): Conformance assessment process for protocol testing SDL: Process P222-test case sel

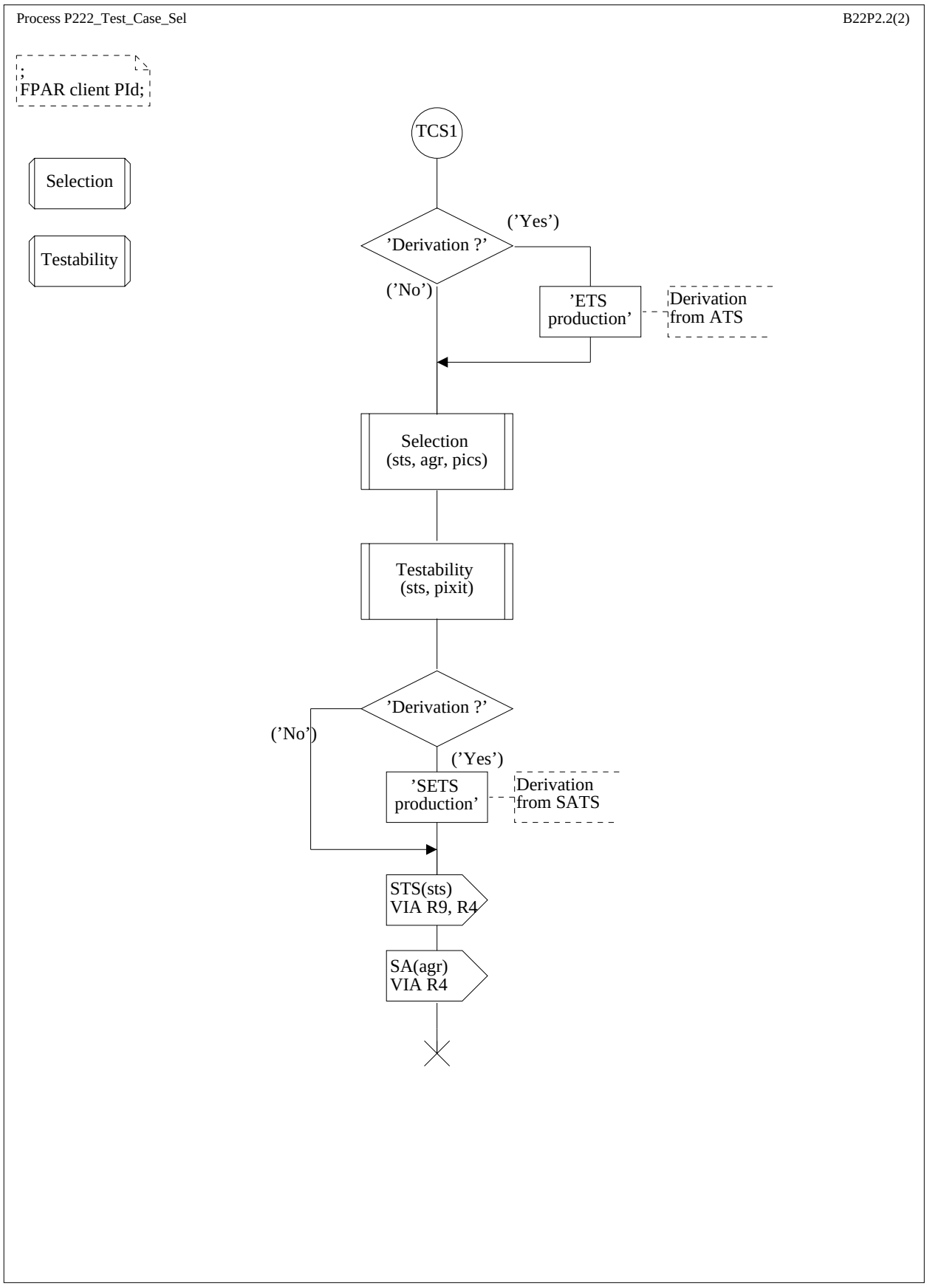
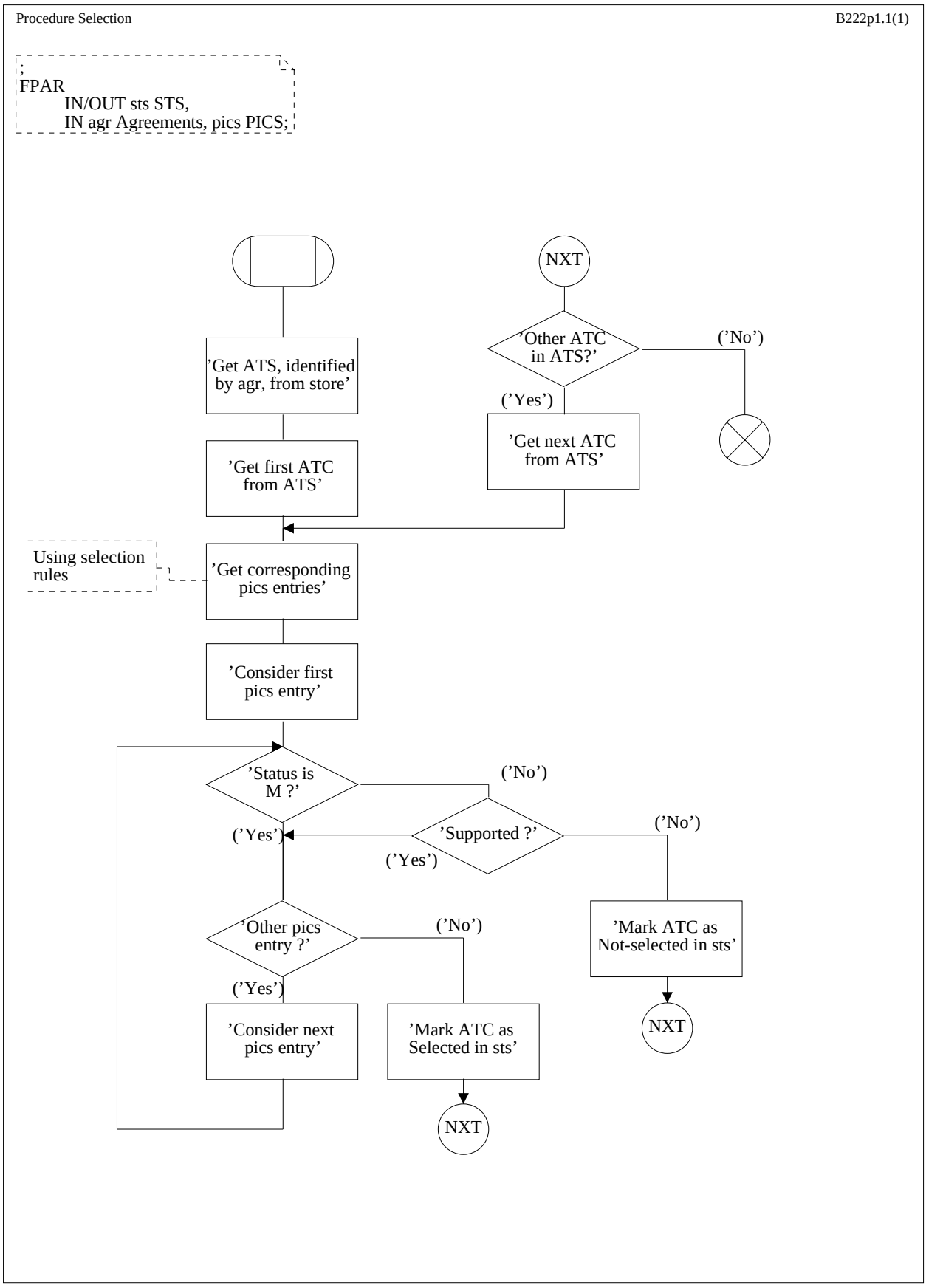


Figure 13 (sheet 2 of 2): Conformance assessment process for protocol testing SDL: Process P222-test case sel



NOTE: The procedure Select assumes that PICS items having conditional status are evaluated and resolved to either mandatory or optional prior to the evaluation in Select.

Figure 14: Conformance assessment process for protocol testing SDL: Procedure-select

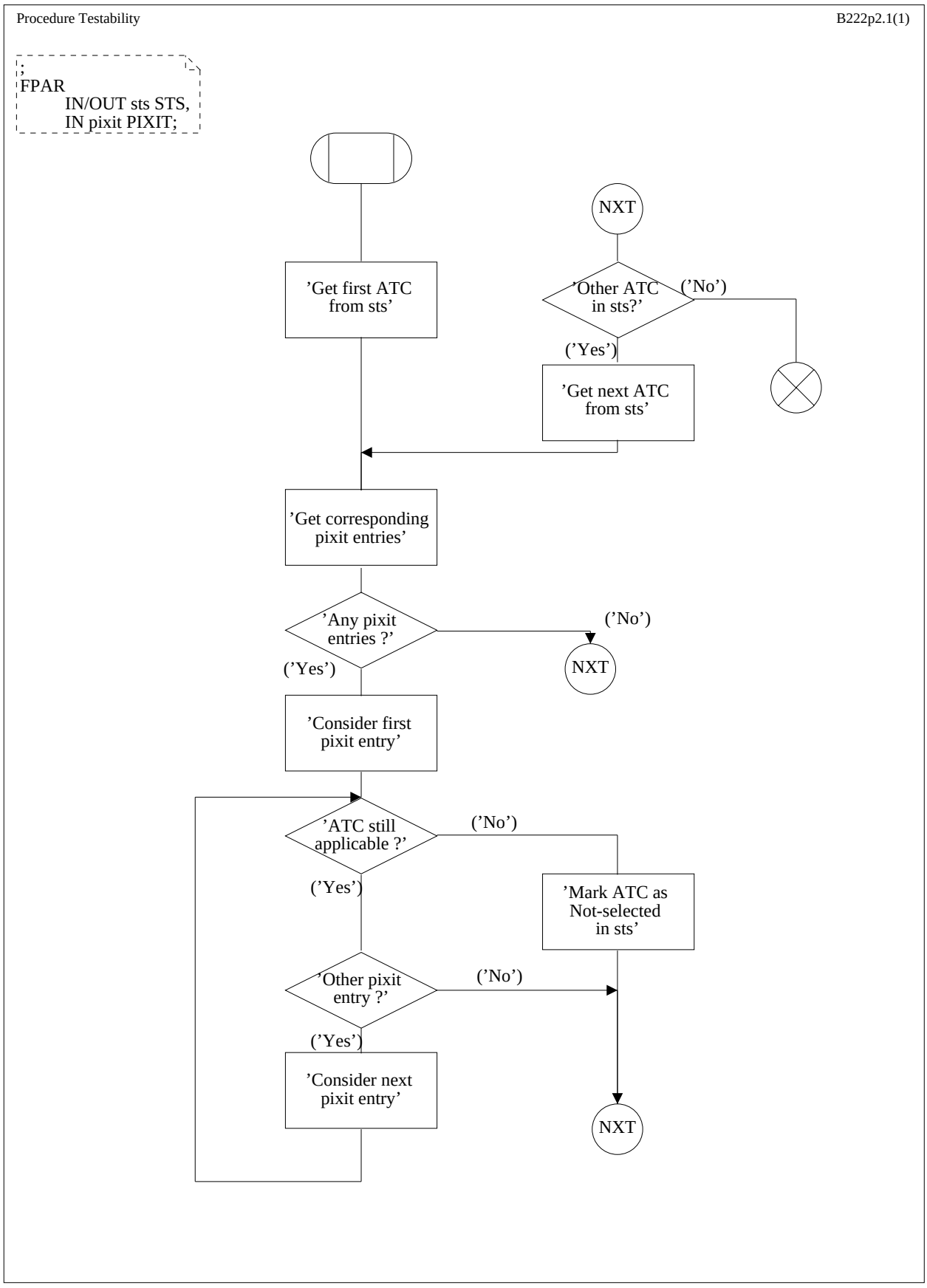


Figure 15: Conformance assessment process for protocol testing SDL: Procedure-testability

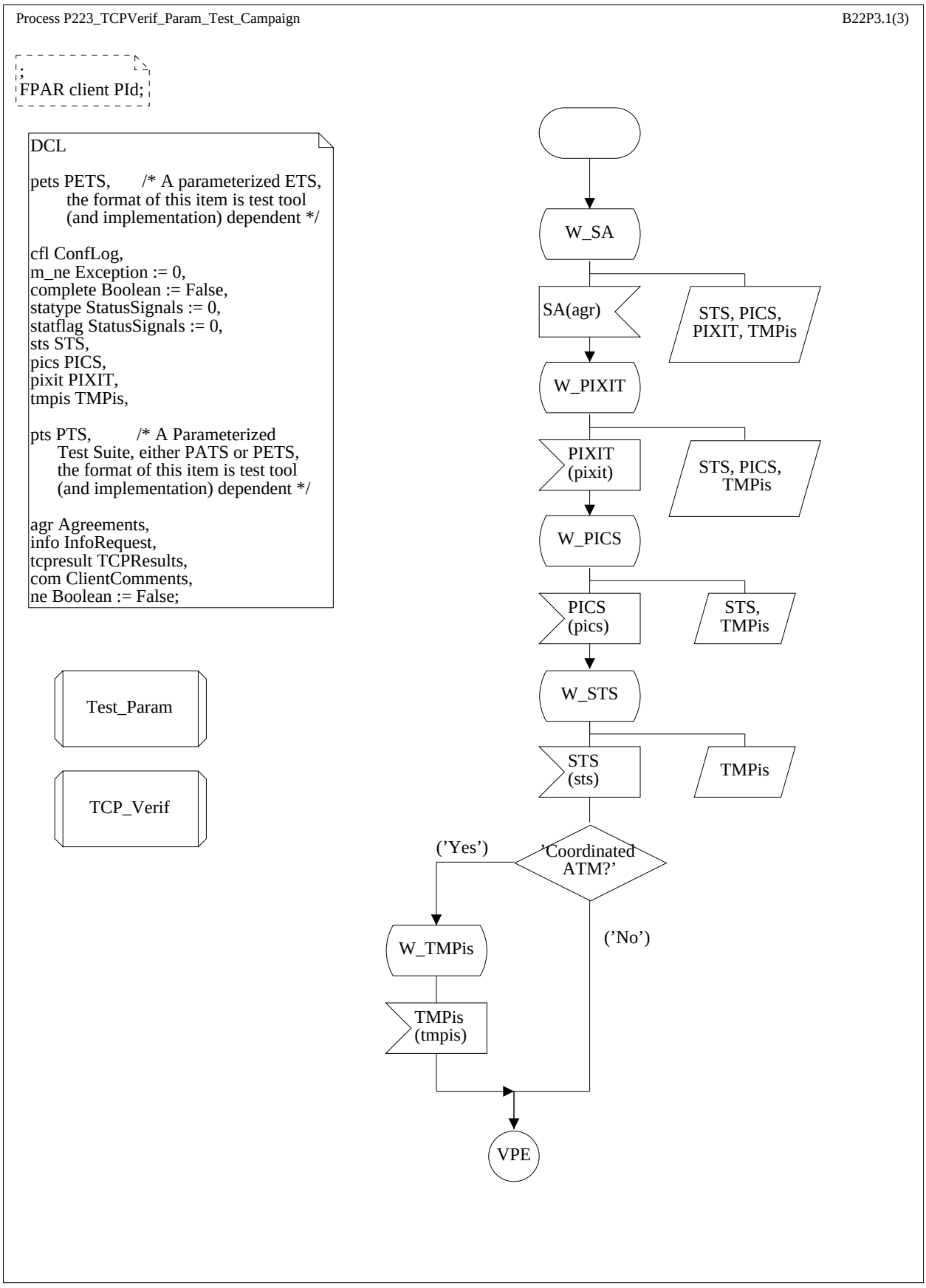


Figure 16 (sheet 1 of 3): Conformance assessment process for protocol testing SDL: Process P223- TCPVerif param test campaign

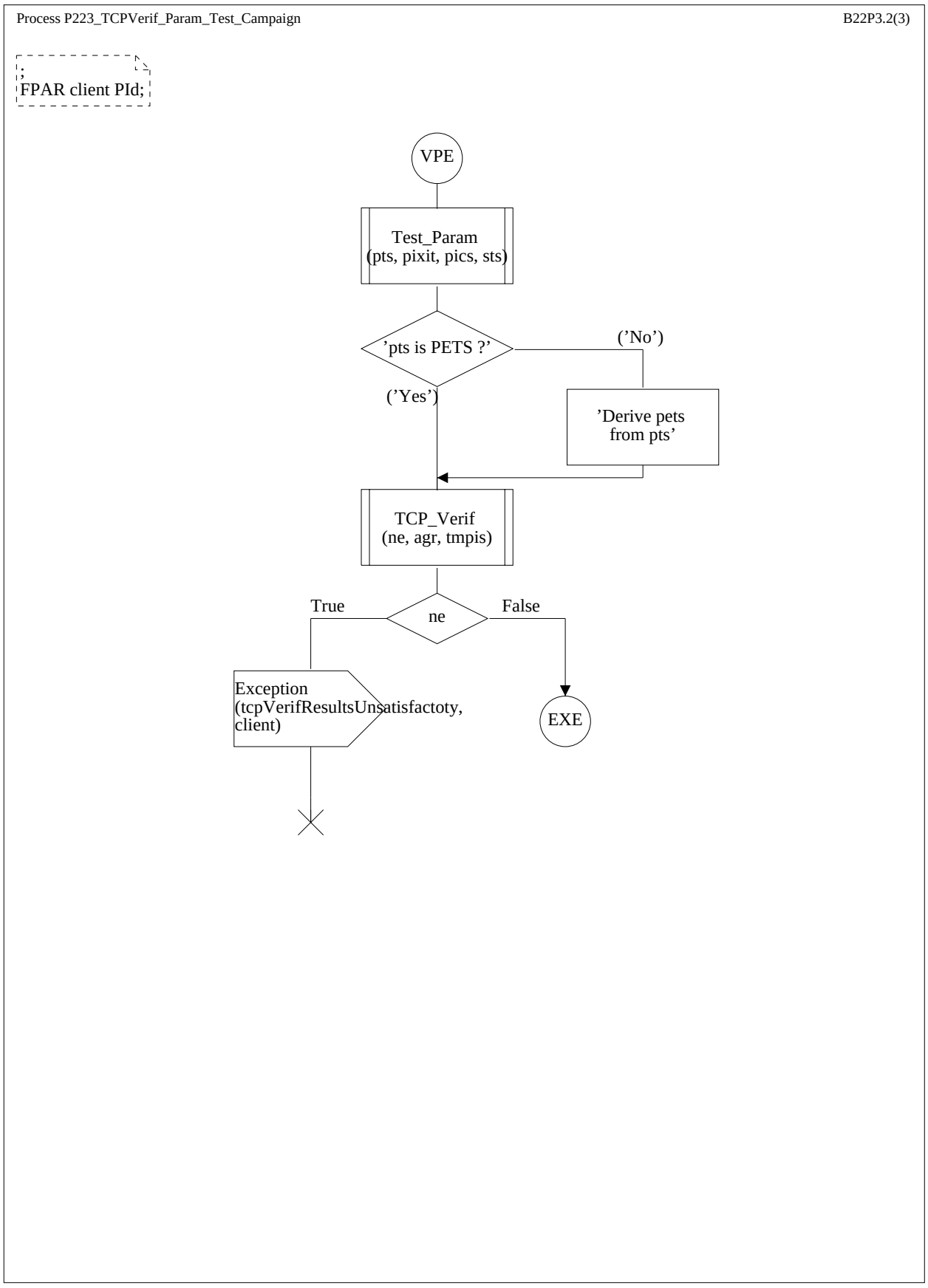


Figure 16 (sheet 2 of 3): Conformance assessment process for protocol testing SDL: Process P223- TCPVerif param test campaign

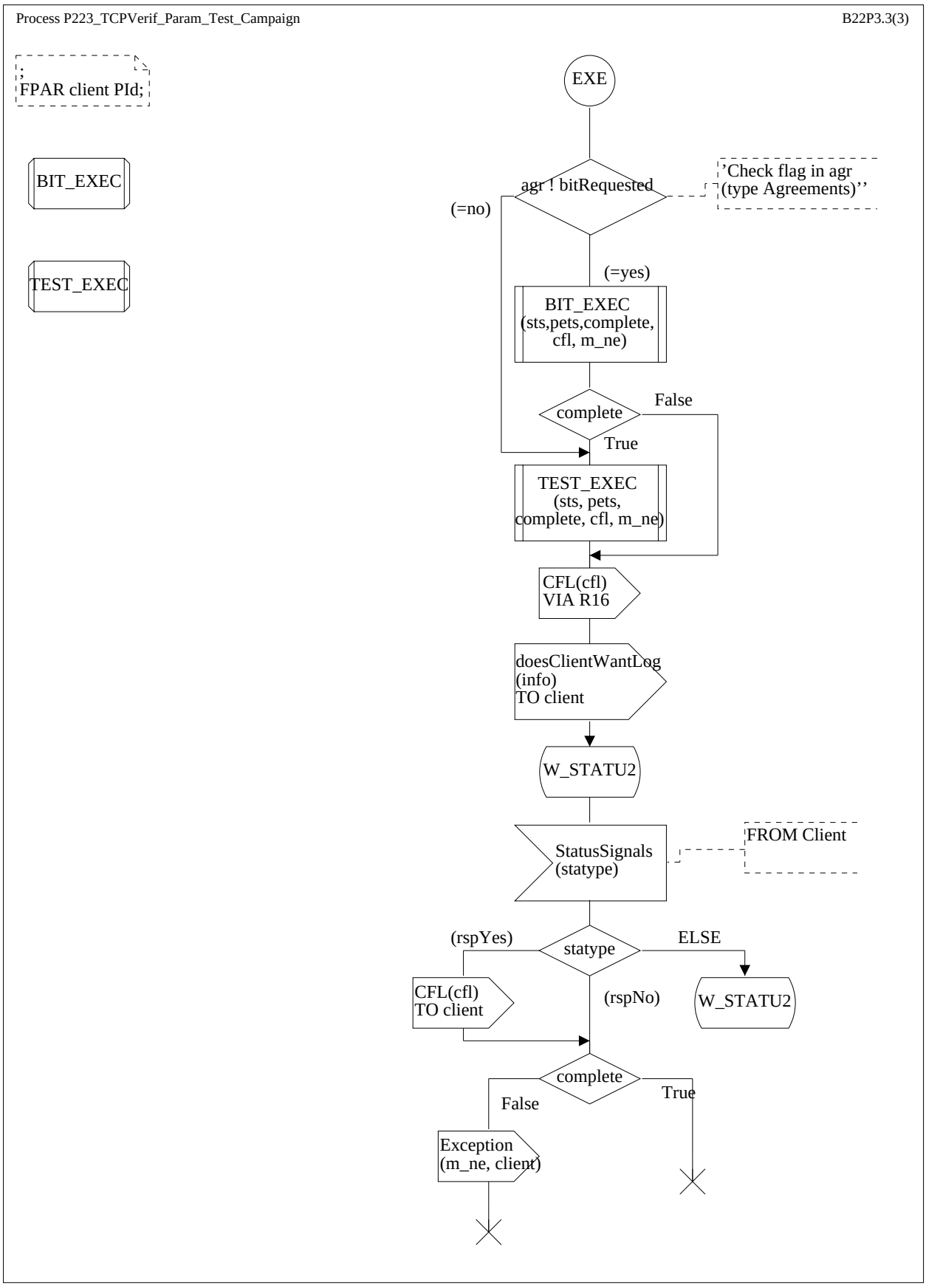


Figure 16: Conformance assessment process for protocol testing SDL: Process P223- TCPVerif param test campaign (sheet 3 of 3)

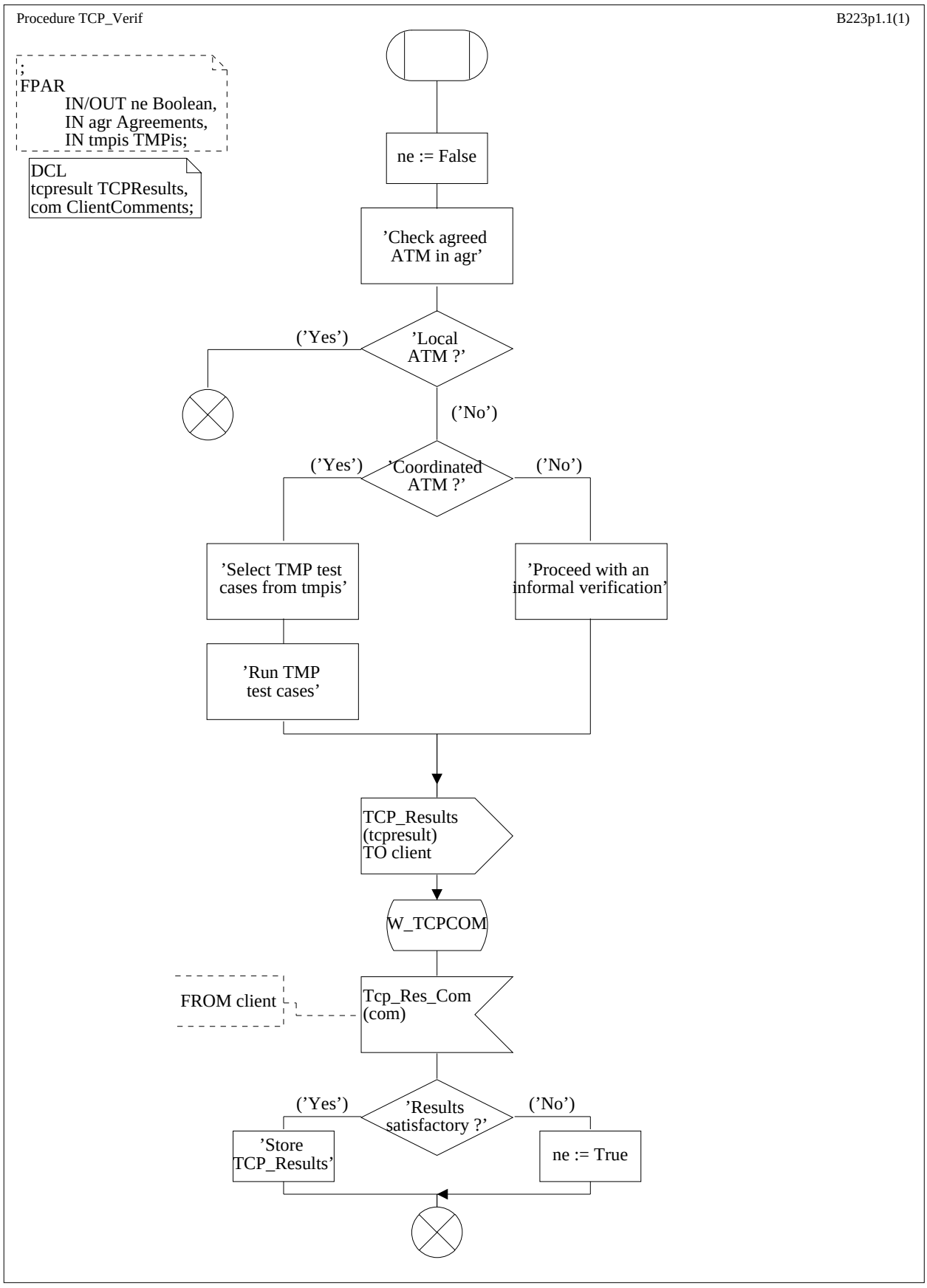


Figure 17: Conformance assessment process for protocol testing SDL: Procedure- TCP_Verif

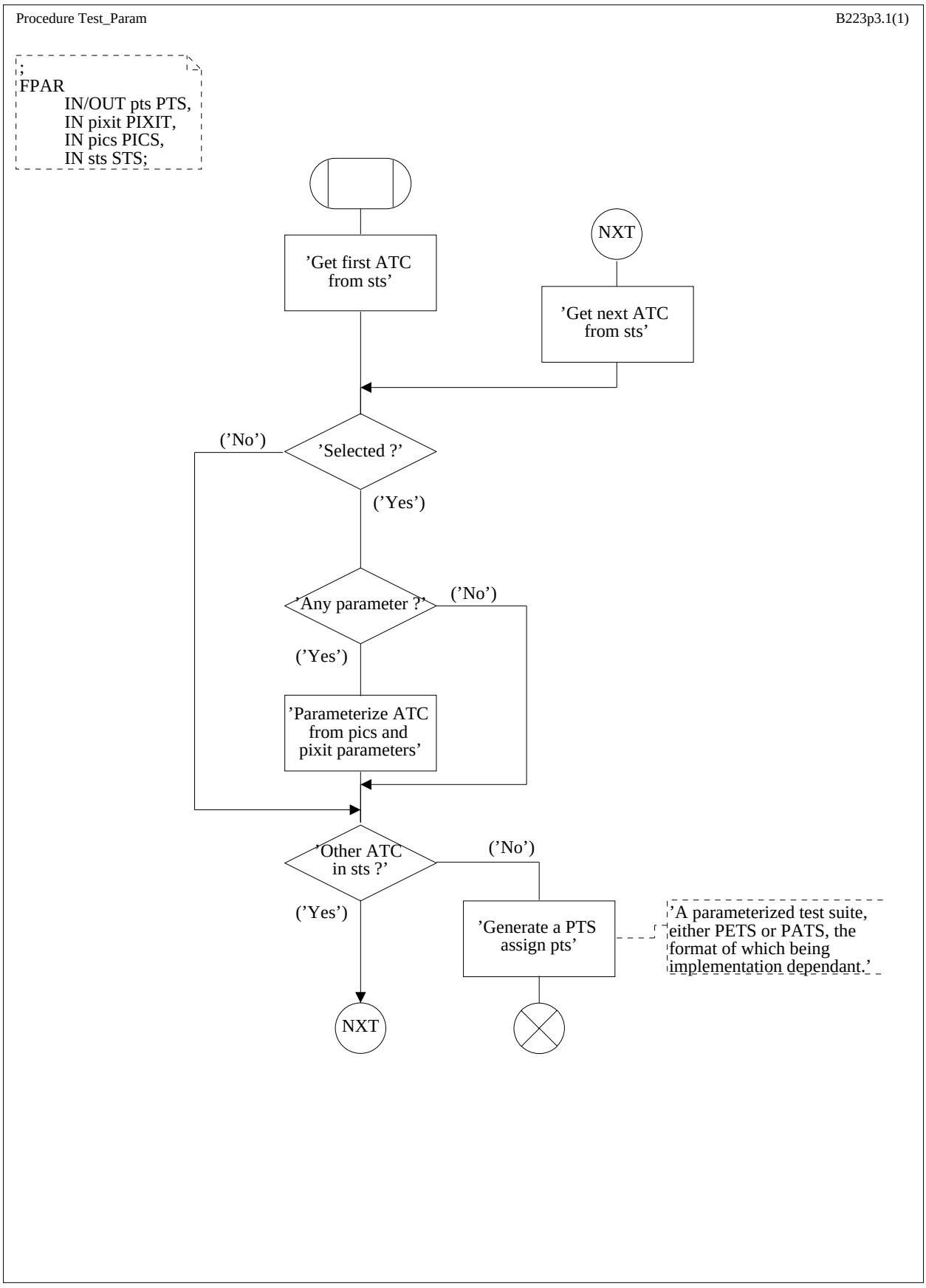


Figure 18: Conformance assessment process for protocol testing SDL: Procedure- TCP_Param

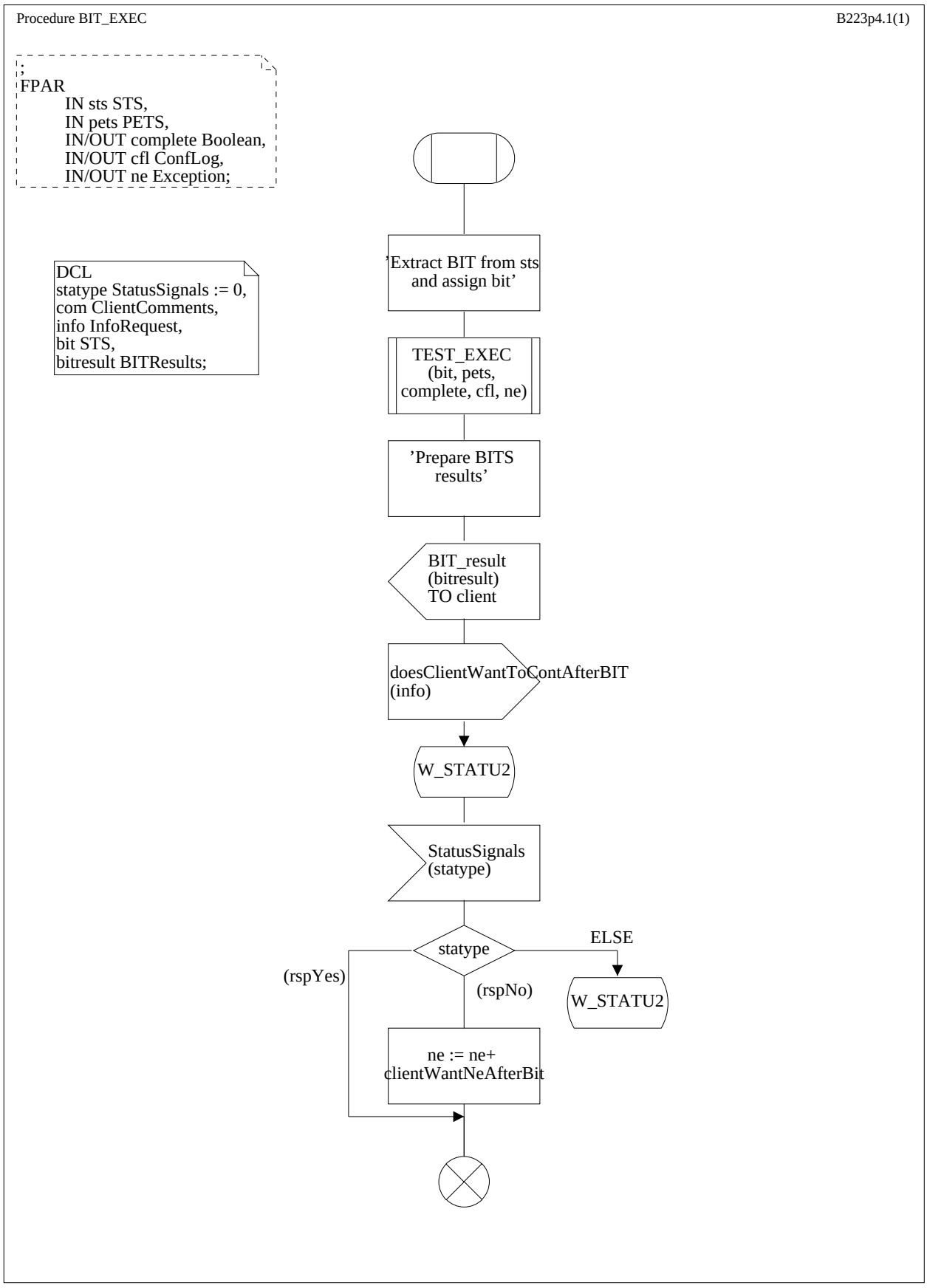


Figure 19: Conformance assessment process for protocol testing SDL: Procedure-BIT_EXEC

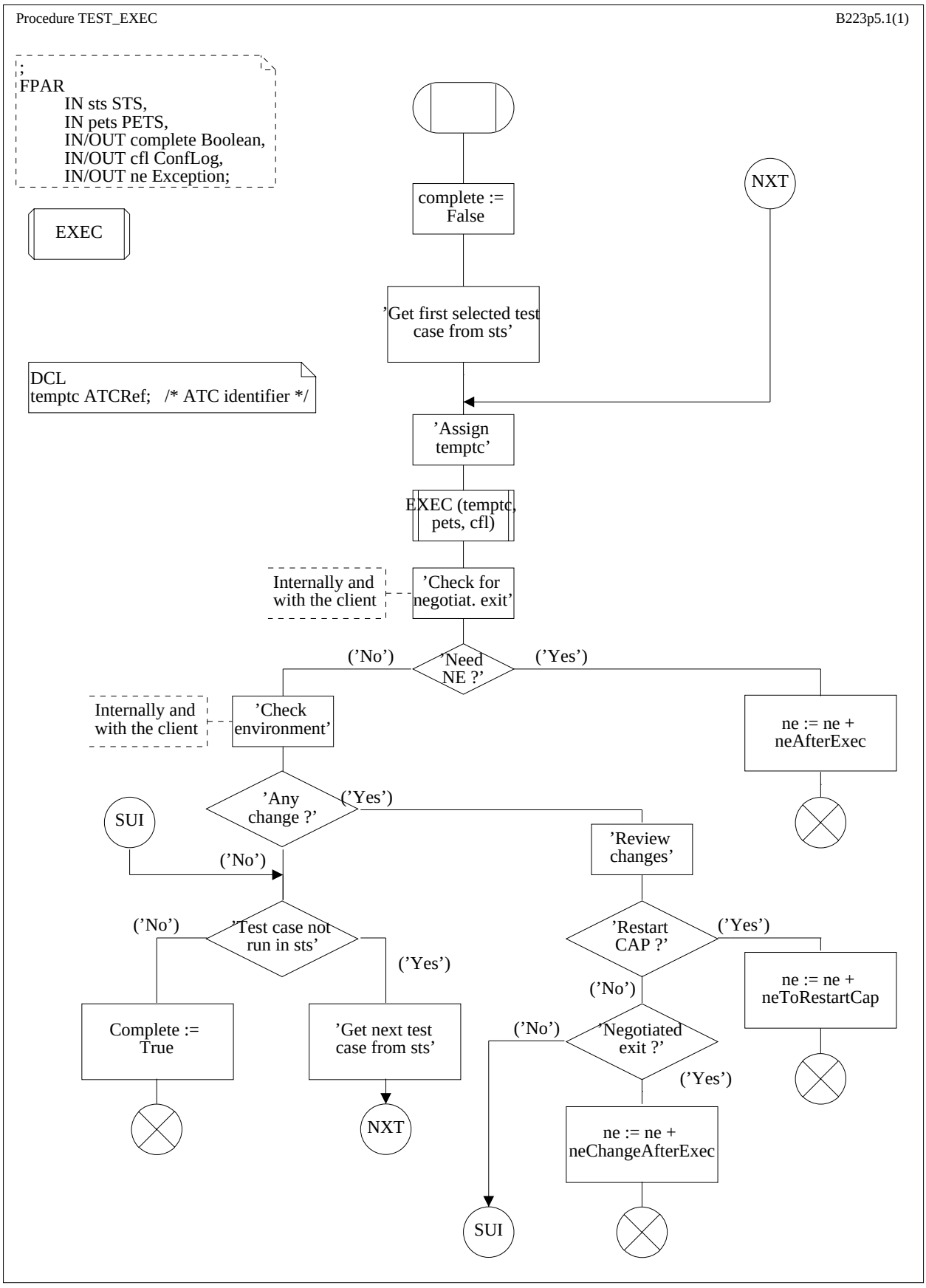


Figure 20: Conformance assessment process for protocol testing SDL: Procedure-TEST_EXEC

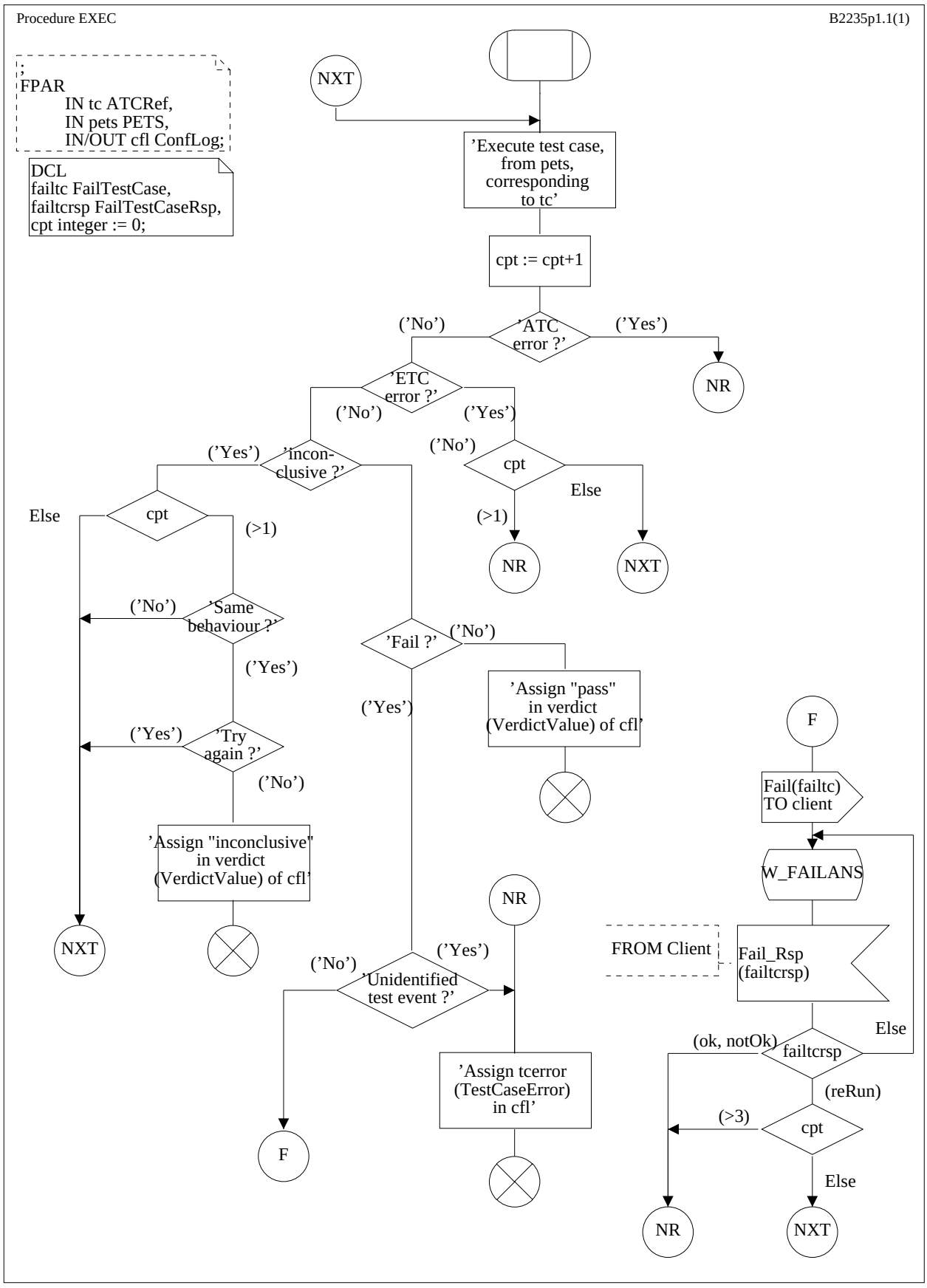


Figure 21: Conformance assessment process for protocol testing SDL: Procedure-EXEC

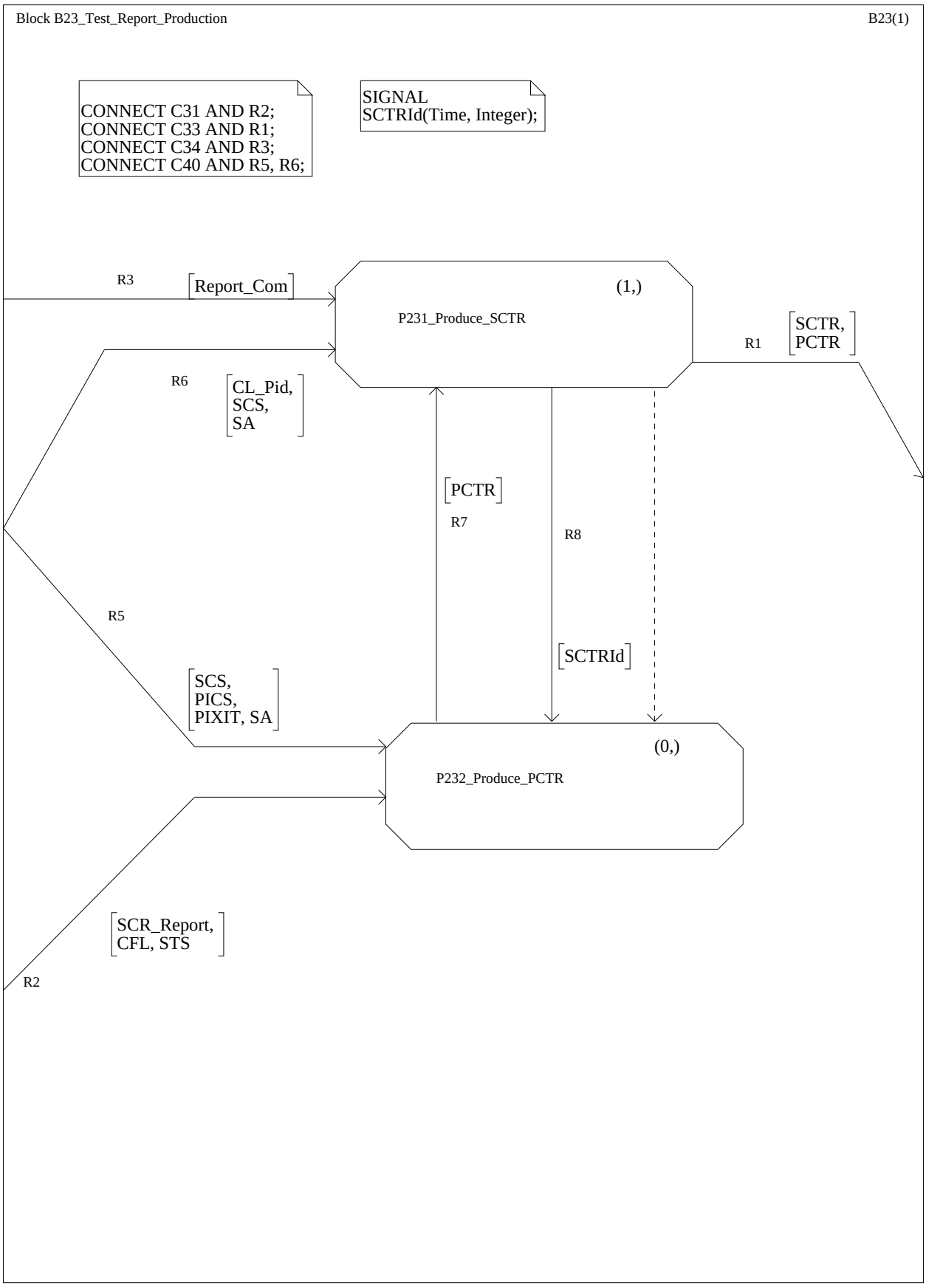


Figure 22: Conformance assessment process for protocol testing SDL: Block B23-test report production

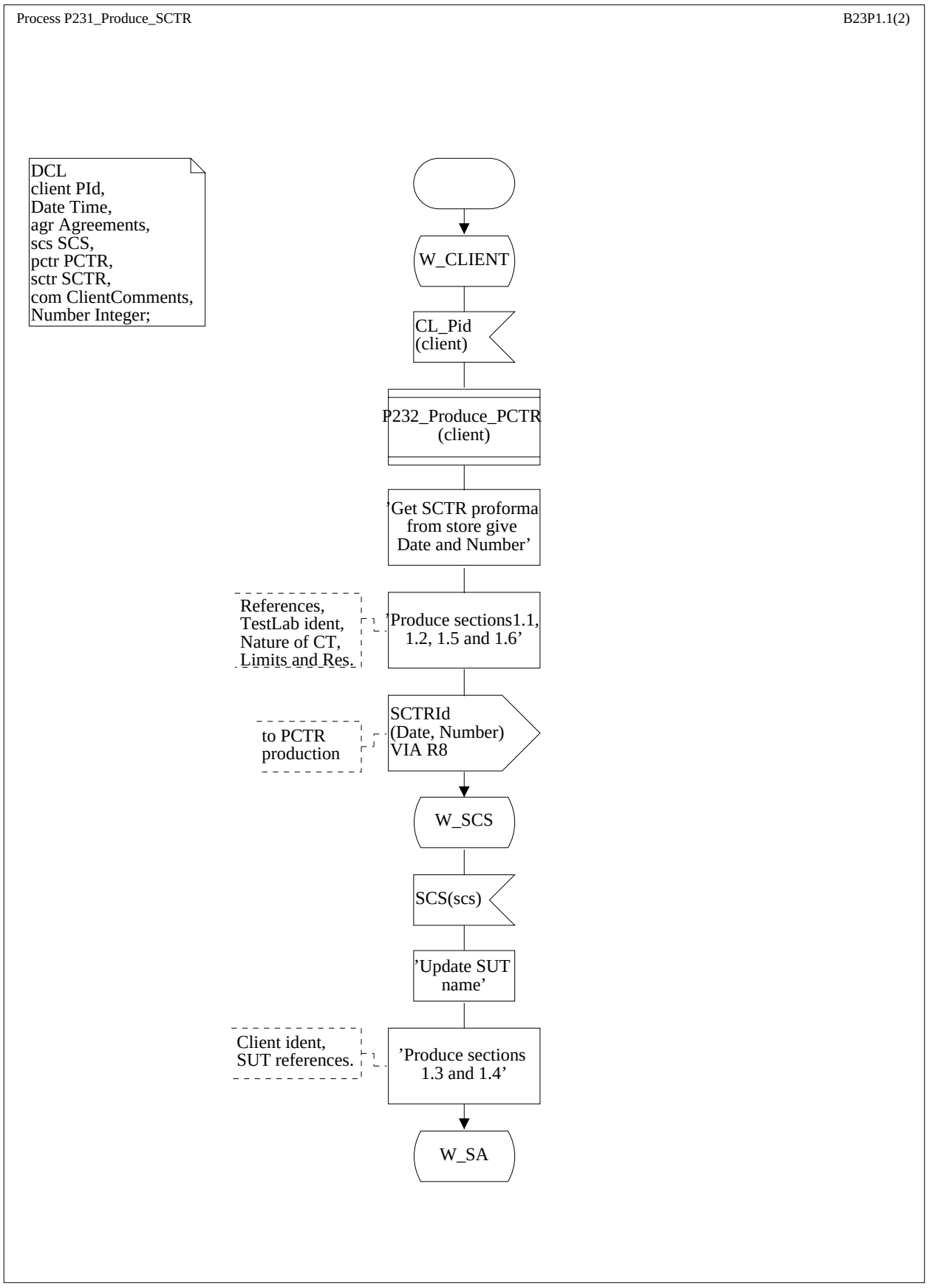


Figure 23 (sheet 1 of 2): Conformance assessment process for protocol testing SDL: Process P231-produce System Conformance Test Report (SCTR)

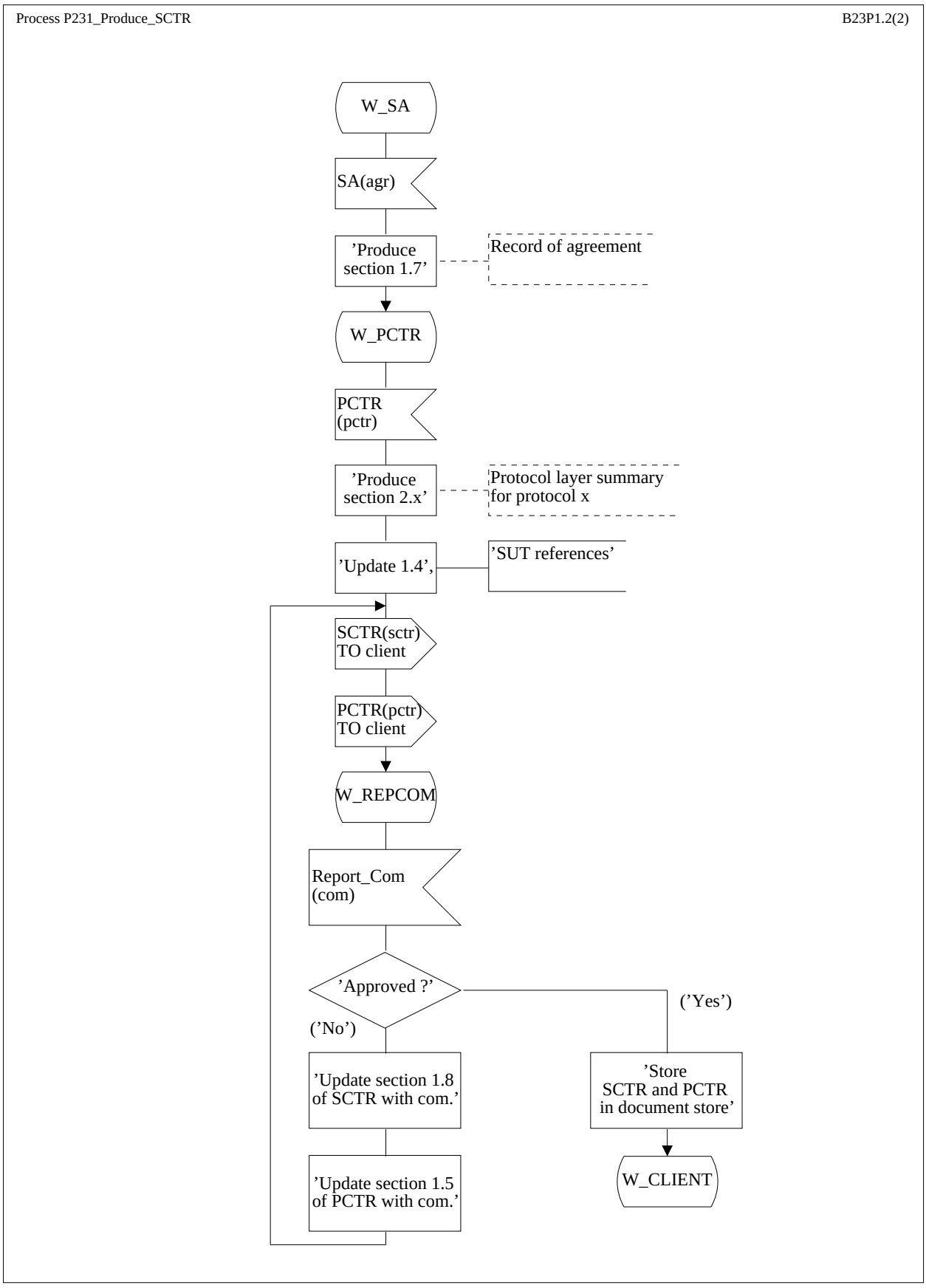


Figure 23 (sheet 2 of 2): Conformance assessment process for protocol testing SDL: Process P231-produce SCTR

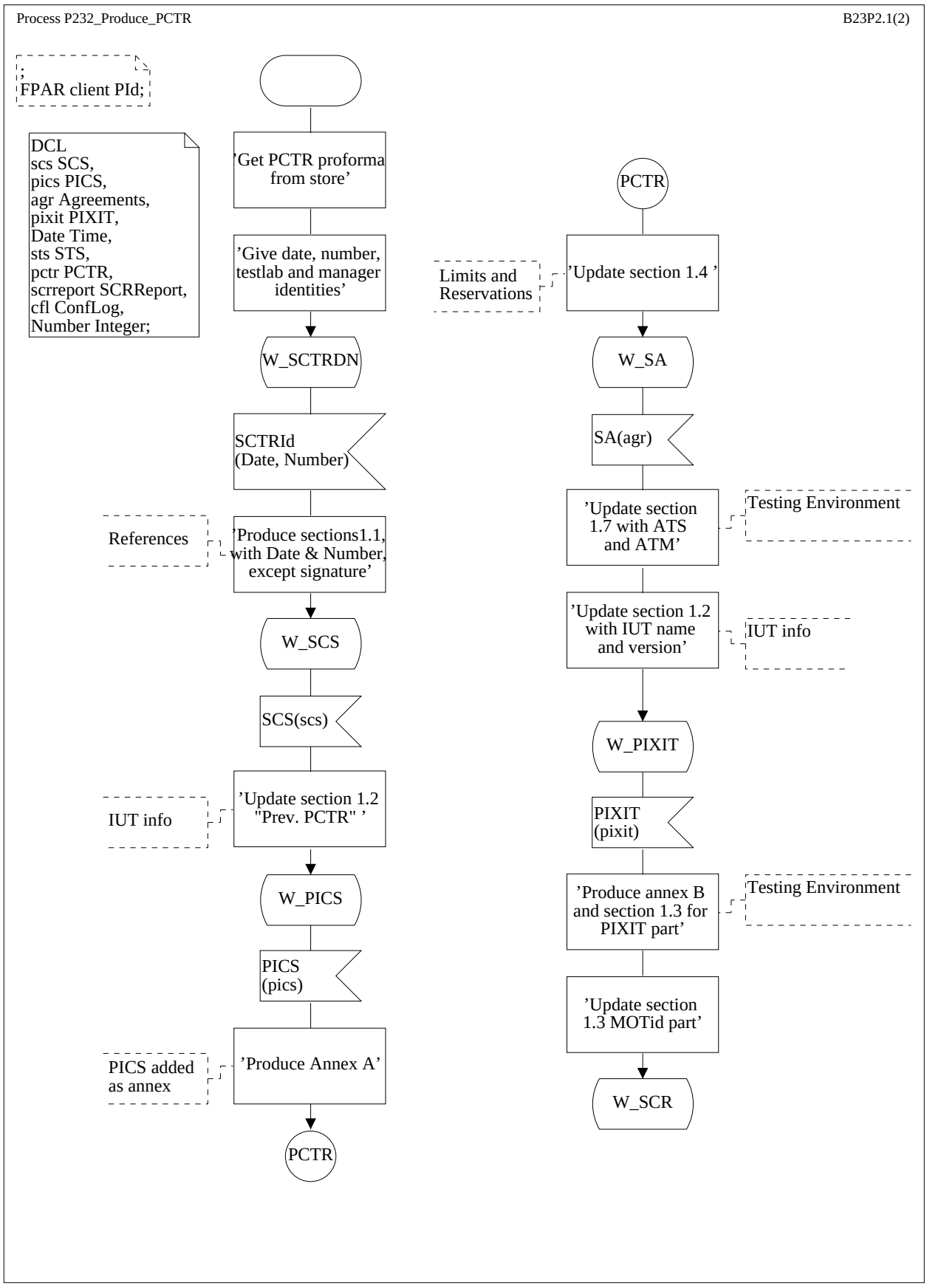


Figure 24 (sheet 1 of 2): Conformance assessment process for protocol testing SDL: Process P232-produce PCTR

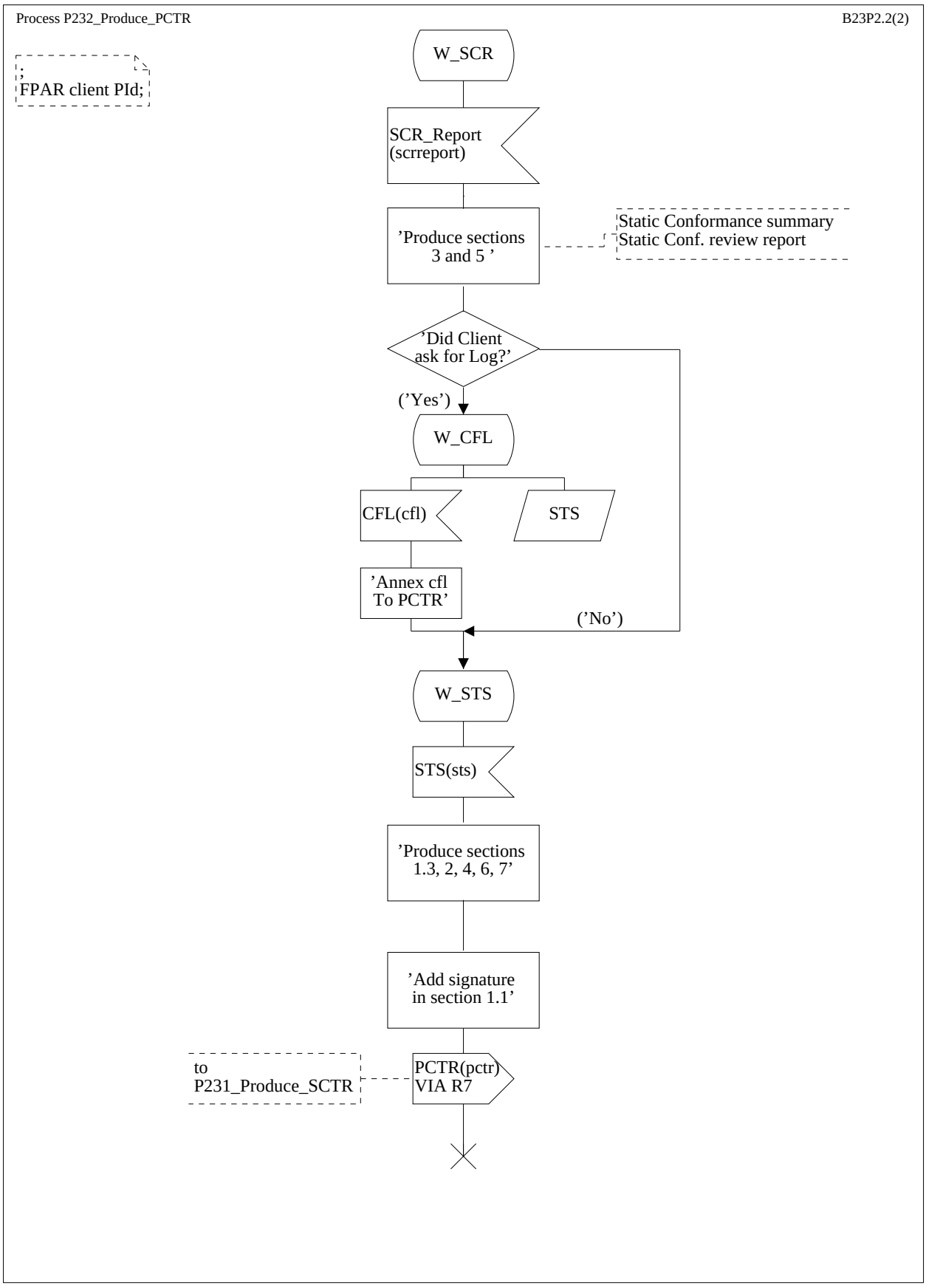


Figure 24 (sheet 2 of 2): Conformance assessment process for protocol testing SDL: Process P232-produce PCTR

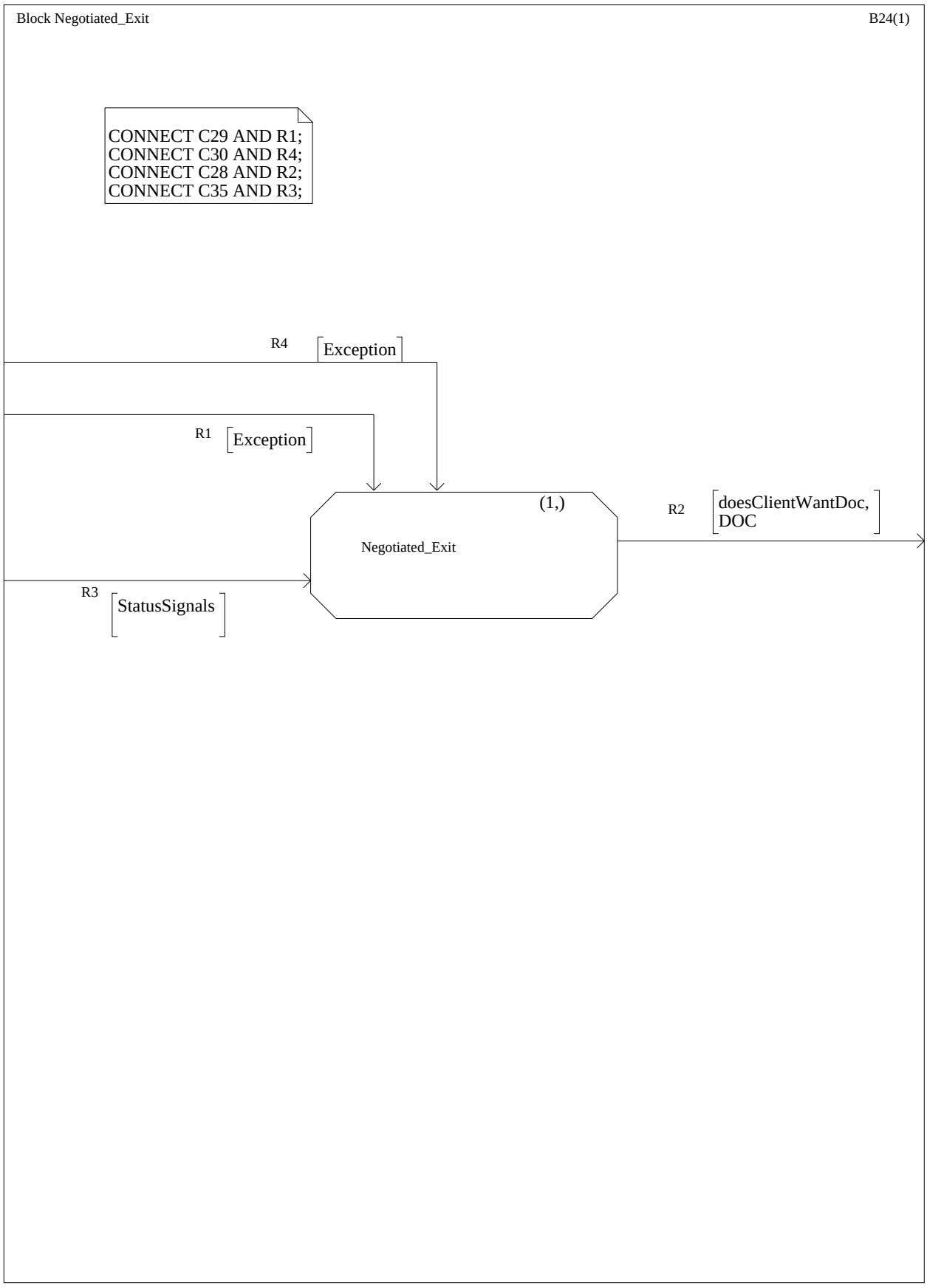


Figure 25: Conformance assessment process for protocol testing SDL: Block-negotiated exit

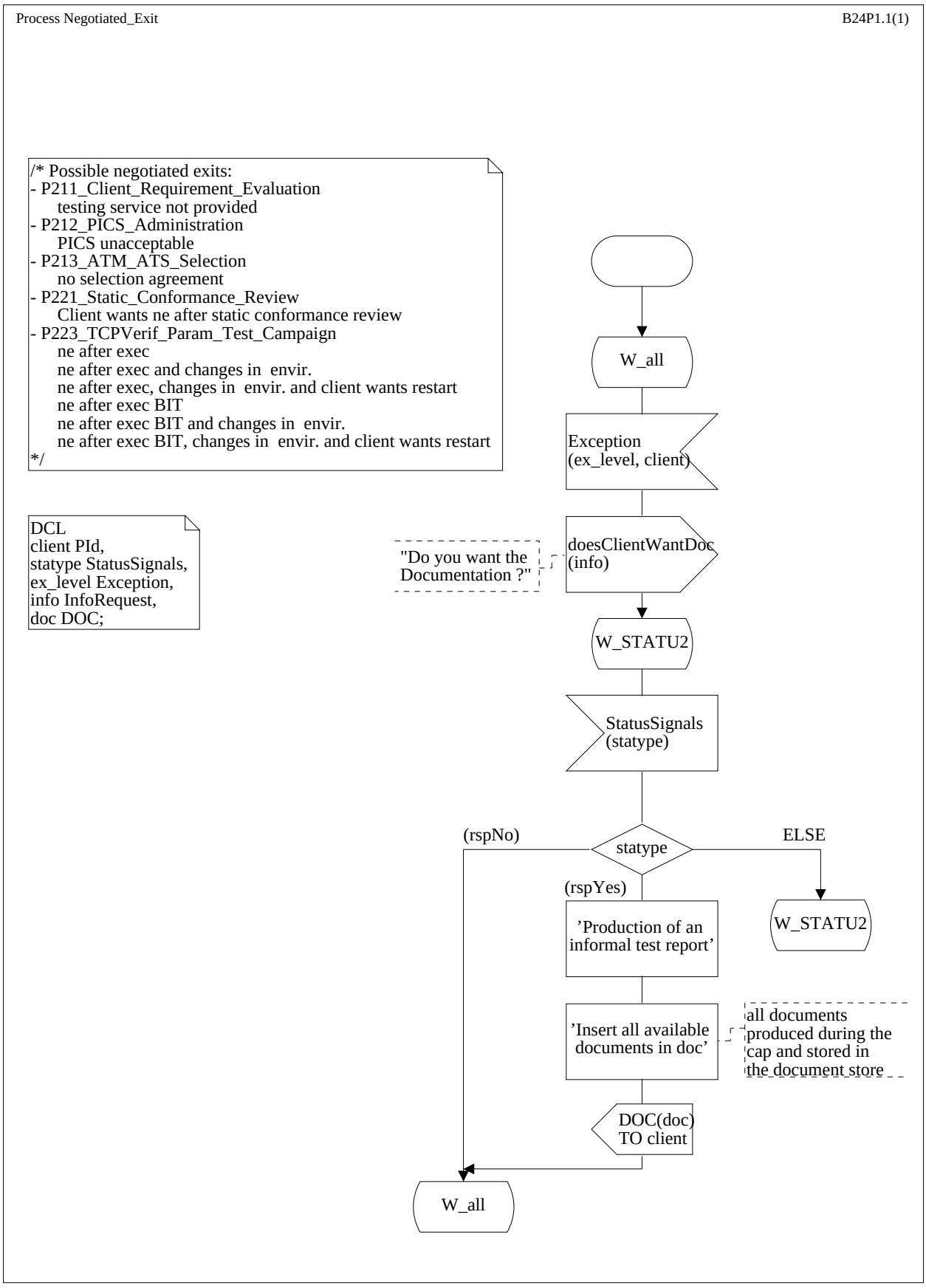


Figure 26: Conformance assessment process for protocol testing SDL: Process-negotiated exit

5.3.2 Extensions of the SDL model for multi-protocol testing

The SDL model presented in the previous subclause covers the testing for conformance of an Implementation Under Test (IUT) to a single protocol standard. This model can easily be extended to the more complex case of the testing for conformance of multi-protocol IUTs by applying the following principles:

- the processes which cover activities which are not intended to be limited to a specific protocol (e.g. client requirements evaluation, SCTR production) need to be extended to cover the case of multi-protocol IUTs;
- the processes which cover activities which by definition are to be undertaken on a protocol basis (e.g. ATM and ATS selection) need to be replicated for each protocol to which conformance is claimed;
- each of these processes communicate only with those other processes which cover activities related to the same protocol, according to the communication model expressed in the previous subclause.

Preparation for testing

In the case of a single protocol IUT, the protocol standard to which conformance is claimed is determined within the P211_Client_Requirements_Evaluation Process. This process is created at the start of the CAP. Based on the protocol identified by the client and referenced in the filled-in SCS, the activities related to the preparation for testing the conformance to this single protocol are carried out : exchange of PICS and PIXIT information, selection of the ATM, the ATS and the MOT, and preparation of the SUT and the MOT. This is done by creating a unique instance of the P212_PICS_Administration, the P213_ATM_ATS_Selection, the P214_PIXIT_Administration, the P216_MOT_Preparation and the P215_SUT_Preparation processes which in turn will handle the corresponding activities.

In case of a multi-protocol IUT, these activities need to be carried out for all protocols to which conformance is claimed. The following principle therefore applies:

- as for the single protocol, a single instance of the P211_Client_Requirements_Evaluation process is created at the beginning of the Conformance Assessment Process;
- the protocol references are given by the client and marked in his SCS. After validating the SCS, the P211_Client_Requirements_Evaluation process then creates for each protocol a specific instance of the following processes:

    - the P212_PICS_Administration process;
    - the P213_ATM_ATS_Selection process;
    - the P214_PIXIT_Administration process;
    - the P216_MOT_Preparation process;
    - the P215_SUT_Preparation processes.

Each of these processes communicate only with those other processes which cover activities related to the same protocol, according to the communication model expressed in the previous subclause.

Test operations

In the case of a single protocol IUT, the test operations consist of carrying out the static conformance review activity, to select the appropriate test cases from the chosen ATS and to undertake the actual test campaign. To do that, a unique instance of the P221_Static_Conf_Review process is created to check, when the related documents are available within the system, the IUT capabilities against the static requirements of the protocol standard. If all requirements are met, the process creates a unique instance of the P222_Test_Case_Sel and P223_TCPVerif_Param_Test_Campaign processes to cover the test campaign.

In the case of multi-protocol IUT, all these activities need to be carried out for each protocol to which conformance is claimed. This is done in the following way:

- one instance of the P221_Static_Conf_Review process is created for each protocol standard to which conformance is claimed;

- each of these process instances receives the PICS, PIXIT and SCS data objects from the corresponding processes from the preparation for testing phase which cover the same protocol standard. They each generate a protocol specific SCR_Report;
- if all static requirements are met by the IUT, each of these P221_Static_Conf_Review processes creates its own instance of the P222_Test_Case_Sel and P223_TCPVerif_Param_Test_Campaign processes to cover the test campaign against the reference protocol standard;
- these processes in turn generate their own selected test suite and conformance log data objects for the protocol they cover.

Test report production

In the case of a single protocol IUT, a SCTR needs to be produced which only reference a single PCTR. An instance of the P231_Produce_SCTR process produce this SCTR based on the results of all previous testing activities whose results become available within the system. To do that, it creates a unique instance of the P232_Produce_PCTR process to address the generation of the PCTR for the protocol standard to which conformance was tested.

In the case of a multi protocol IUT, a unique SCTR needs to be created which will reference as many PCTR that there are protocols to which conformance was tested. This will take place as follows:

- the P231_Produce_SCTR process will create as many instances of the P232_Produce_PCTR process as there are protocols implemented in the IUT;
- each of these processes will generate the PCTR for the protocol standard it covers, based on all information available within the system;
- the P231_Produce_SCTR process then collects the resulting PCTRs and generate the SCTR for the SUT.

A possible implementation of the model for a multi-protocol IUT is illustrated in figure 27 below.

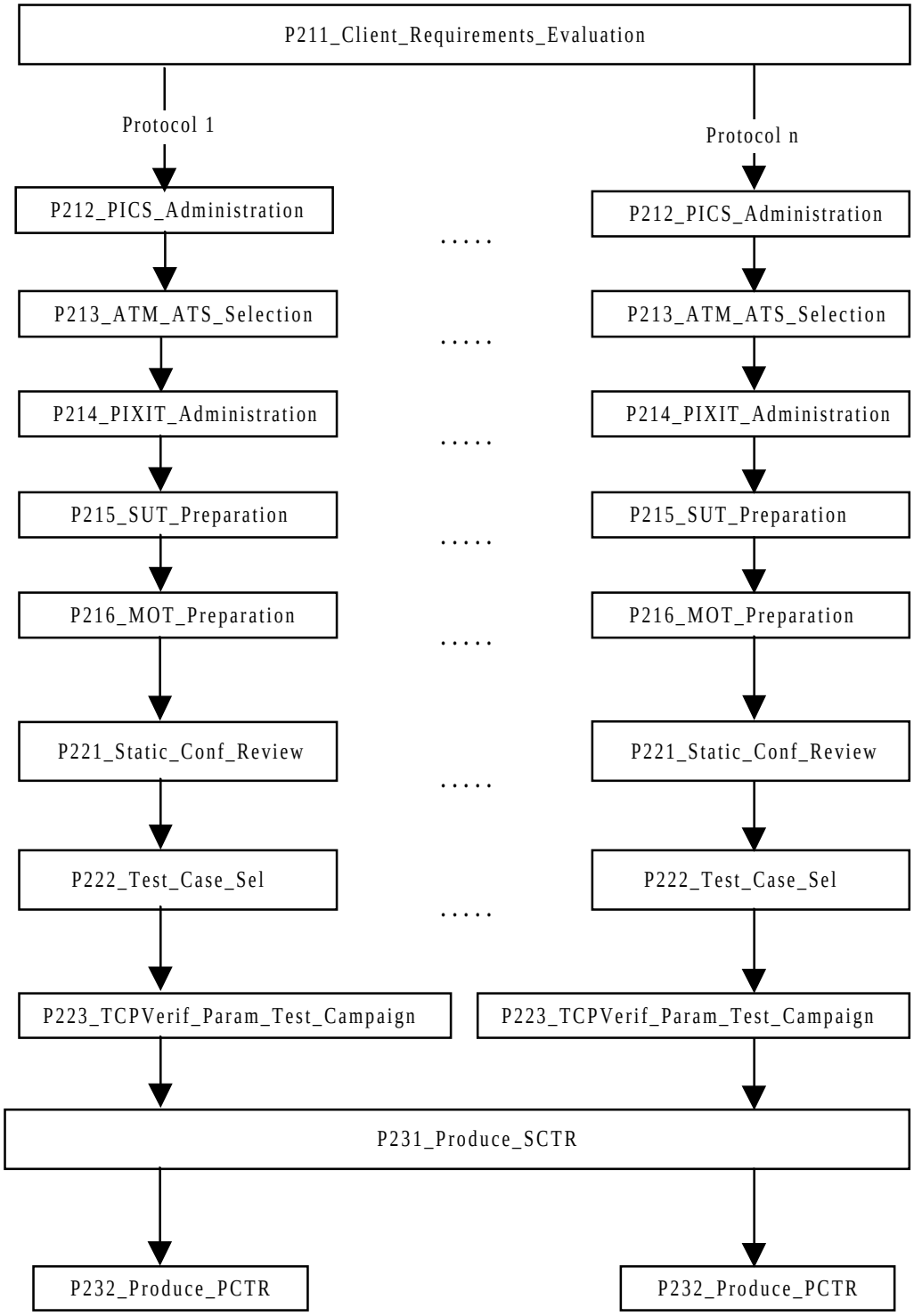


Figure 27: Possible implementation of the model for a multi-protocol IUT

5.3.3 Description of information flows between the CAP processes

Table 1 below describes each signal in the CAP together with a classification of the parameter of the signal according to the scheme described in subclause 5.1. That is, each information flow (data object) is classified as:

SC: the structure and content of the data object is defined in ISO/IEC 9646 [1] - [6];
C: the content of the data object is defined but not structured in ISO/IEC 9646 [1] - [6];
R: the data object is mentioned in ISO/IEC 9646, but its content and structure are not defined;
A: the data object is resulting from an interpretation of the ISO/IEC 9646 [1] - [6] methodology.

In addition, the ASN.1 data type name of the parameter of the signal is identified together with a clause reference.

Table 1: Description of signals between CAP processes

Signal	Description	Classifi-cation	ASN.1 Type of Signal Parameter	Defined in subclause
BITResults	Test lab informing the client about the result of the execution of the BIT	A	BITResults	5.3.4.13
CFL	Conformance Log	C	ConfLog	5.3.4.5
CL_C	Client checklist	C	ClientChecklist	5.3.4.9
CL_C_PF	Client checklist proforma	A	ClientChecklist	5.3.4.9
CLCErrors	List of errors detected in the CL_C	A	InputErrors	5.3.4.13
CL_Pid	Internal identification of the client	A	CLPid	5.3.4.13
C_SUT_Ready	Client informing the lab that the SUT is ready for testing	A	CSUTReady	5.3.4.13
doesClientWantBit	Test lab asking the client if he wants execution of the BIT	A	Inforequest	5.3.4.13
doesClientWantToContAfterBit	Test Lab asking the clientafter the execution of the BIT if he wants to continue the CAP	A	InfoRequest	5.3.4.13
doesClientWantLog	Test Lab asking the client if he wants the conformance log	A	InfoRequest	5.3.4.13
Exception	Specification of an event which is causing a negotiated exit	A	Exception	5.3.4.13
FAIL	The test lab indicating to the client that the specified test case yielded a fail verdict	A	FailTestCase	5.3.4.13

(continued)

Table 1 (continued): Description of signals between CAP processes

Signal	Description	Classifi-cation	ASN.1 Type of Signal Parameter	Defined in subclause
FAILANS	Client response to the notification of a FAIL verdict : accepted by the client, not accepted by the client, the client requests a re-run of the test case	A	FailTestCaseRsp	5.3.4.13
MOTId	Internal identifier of the selected MOT for the current test campaign	A	MOTId	5.3.4.13
PCTR	Protocol Conformance Test Report	SC	PCTR	5.3.4.4
PCTR_PF	PCTR proforma	SC	PCTR	5.3.4.4
PICS	PICS (formatted by protocol standard)	SC	PICS	5.3.4.6
PICS_PF	PICS proforma (formatted by protocol standard)	SC	PICS	5.3.4.6
PICSErrors	List of (syntactic) errors made by the client when filling in the PICS proforma	A	InputErrors	5.3.4.13
PIXIT	PIXIT	C	PIXIT	5.3.4.7
PIXIT_PF	PIXIT proforma	C	PIXIT	5.3.4.7
PIXITErrors	List of (syntactic) errors made by the client when filling in the PIXIT proforma	A	InputErrors	5.3.4.13
Prot_Ref	Client indicating the protocol to which conformance is to be assessed	A	ProtRef	5.3.4.13
Prot_STDId	Internal identification of the protocol standard to which conformance is to be assessed	R	ProtStdId	5.3.4.13
Report_Com	Comments from the client on the proposed test reports (SCTR/PCTR)	R	ClientComments	5.3.4.13
SA	Selection Agreements i.e. what has been agreed between the test lab and the client about ATM, ATS and the IUT	C	Agreements	5.3.4.12
SA_Com	Comments from the client on the Selection Agreements	A	ClientComments	5.3.4.13
SCS	System Conformance Statement	C	SCS	5.3.4.2
SCS_PF	System Conformance Statement proforma	R	SCS	5.3.4.2
SCSErrors	List of (syntactic) errors made by the client when filling in the SCS proforma	A	InputErrors	5.3.4.13
SCR_Report	Static Conformance Review Report	R	SCRReport	5.3.4.11

(continued)

Table 1 (concluded): Description of signals between CAP processes

Signal	Description	Classifi-cation	ASN.1 Type of Signal Parameter	Defined in subclause
SCTR	System Conformance Test Report	SC	SCTR	5.3.4.3
SCTR_PF	System Conformance Test Report Proforma	SC	SCTR	5.3.4.3
STS	Selected Test Suite, i.e. the list of all test identifiers from the ATS, qualified as selected or deselected (due to PICS or PIXIT)	A	STS	5.3.4.13
TCP_Result	The results of the TCP verification activity	A	TCPResults	5.3.4.11
TCP_Res_Com	Comments from the client on the results of the TCP verification activity	A	ClientComments	5.3.4.13
TL_C	Test Lab Checklist	C	TestLabChecklist	5.3.4.8
TMPis	Test Management Protocol Implementation Statement	C	TMPis	5.3.4.10
TMPis_PF	Test Management Protocol Implementation Statement proforma	C	TMPis	5.3.4.10
TMPisErrors	List of (syntactic) errors made by the client when filling in the SCS proforma	A	InputErrors	5.3.4.13
StatusSignals	Synchronisation signals. Possible values are : MOT_Ready, SUT_Ready, response (No), response (Yes)	A	StatusSignals	5.3.4.13

5.3.4 Definition of conformance data objects

5.3.4.1 Common type definitions

The following ASN.1 type definitions, appearing in alphabetical order, are shared by a number of Conformance data objet definitions.

```
AbsTestMethod ::= INTEGER {
                    ls(0), rs(1), cs(2), ds(3), lm(4), rm(5), cm(6), dm(7),
                    lse(8), dse(9), cse(10), rse(11), lme(12), dme(13), cme(14), rme(15),
                    yl(16), yt(17)
                    }       -- Test methods as identified in 9646-2 [2] , 12.3.6, 12.6.2, 12.6.3

AdditObservation ::= CharString

AnyRange ::= SEQUENCE {
                    lowerBound      ANY,
                    upperBound      ANY
                    }

CharString ::= SEQUENCE {
                    nonSpecific     IA5String,                  -- The non-specific part must
                    langSpecific    GeneralString OPTIONAL      -- always be present
                    }

ClientIdentification ::= SEQUENCE {
                        orgName             OrgName,
                        orgManager          CharString,
                        orgLiaisonOfficer   CharString OPTIONAL
                        }

Comments ::= SEQUENCE OF
                     SEQUENCE {
                        origin          INTEGER { testlab(0), client(1) },
                        identifier      INTEGER,
                        comment         CharString
                        }

DeselectionStatus ::= CHOICE {
                    deselectedPics  [1]     INTEGER{yes(0),no(1)},
                    deselectedPixit [2]     CharString
                                    -- PIXIT clause causing the deselection
                    }

DocReference ::= SEQUENCE {
                    reference               CharString,
                    version         [1]     CharString OPTIONAL,
                    creationdate    [2]     UTCTime OPTIONAL
                    }

ErrorStatus ::= INTEGER { noErrors(0), errors(1) }

IntegerRange ::= SEQUENCE {
                    lowerBound      INTEGER,
                    upperBound      INTEGER
                    }

IUTIdentification ::= NameAndVersion
```

```
MOTIdentification ::= SEQUENCE {
                        identification    NameAndVersion,
                        dateOfOrigin      UTCTime
                        }

NameAndSignature ::= SEQUENCE {
                        name              CharString,
                        signature         ANY OPTIONAL    -- For further study
                        }

NameAndVersion ::= SEQUENCE {
                        name              CharString,
                        version           CharString
                        }

OrgName ::= CHOICE {
                    name            [1]     CharString,
                    objectName      [2]     DirectoryName
                    }

PersonalIdentification ::= CHOICE {
                        name              [1]     CharString,
                        objectName        [2]     DirectoryName
                        }

SCTRId ::= DocReference

StandardReference ::= SEQUENCE {
                        origin                    CharString,
                        reference                 CharString,
                        title             [1]     CharString OPTIONAL,
                        version           [2]     CharString OPTIONAL,
                        date              [3]     UTCTime OPTIONAL
                        }

SUTIdentification ::= SEQUENCE {
                    name            NameAndVersion,
                    supplierName    CharString OPTIONAL
                    }

TestCaseError ::=  SEQUENCE {
                    type            INTEGER     {noError(0), atcError(1), etcError(2),
                                                 abnormalTermination(3), motError(4),
                                                 unqualifiedError(5),
                                                 tcNotImplemented(6)},
                    additInfo       AdditObservation OPTIONAL
                                        -- to be present at least if value in
                                        -- 'type' is'unqualified-error'
                    }

TestLabIdentification ::= SEQUENCE {
                    orgName                     OrgName,
                    orgManager                  CharString,
                    orgLiaisonOfficer   [1]     CharString OPTIONAL,
                    testEngineers       [2]     SEQUENCE OF CharString OPTIONAL,
                    accreditationStatus [3]     CharString OPTIONAL
                                                -- OPTIONAL because not inISO/IEC 9646
                                                -- [1] - [6] but mandatory in European
                                                -- environments
                    }
```

```
TimePeriod ::= SEQUENCE {
                    start    UTCTime,
                    end      UTCTime
                    }
VerdictValue ::= INTEGER { pass(0), fail(1), inconclusive(2) }

ATCRef ::= CharString
```

5.3.4.2 System Conformance Statement (SCS) and Proforma (SCS_PF)

This subclause contains the definition of the System Conformance Statement (SCS) conformance data object. The corresponding proforma (SCS_PF) shares the same definition.

```
SCS ::= SEQUENCE {
                identification          DocReference,
                protocolAndPics         SEQUENCE OF ProtocolAndPics,
                client                  ClientIdentification,
                sut                     SUTDescription,
                picsRef                 DocReference,
                previousSCTR            SEQUENCE OF DocReference OPTIONAL
                }

ProtocolAndPics ::= SEQUENCE {
                    protocolStandardReference,
                    pics            StandardReference OPTIONAL
                                        -- this is optional because multiple protocols may
                                        -- be referenced in the SCS but not all of them may
                                        -- be tested in the conformance assessment
                                        -- process
                    }

SUTDescription ::= SEQUENCE {
                    sutId               SUTIdentification,
                    sutComponents SUTComponents OPTIONAL
                    }

SUTComponents ::= SEQUENCE {
                    hardware        [1]     HWIdentification OPTIONAL,
                    operatingSys    [2]     OPSYSIdentification OPTIONAL,
                    commPlatform    [3]     SWIdentification OPTIONAL
                    }

HWIdentification ::= SEQUENCE {
                    name                NameAndVersion,
                    serialNumber        CharString OPTIONAL
                    }

OPSYSIdentification ::= NameAndVersion

SWIdentification ::= SEQUENCE {
                    product         [1]     CharString OPTIONAL,
                    supplier[2]     CharString OPTIONAL,
                    sctrRef         [3]     DocReference OPTIONAL,
                    standardRef     [4]     SEQUENCE OF StandardReference
                    }
```

5.3.4.3 System Conformance Test Report (SCTR) and Proforma (SCTR_PF)

This subclause contains the definition of the System Conformance Test Report (SCTR) conformance data object. The corresponding proforma (SCTR_PF) shares the same definition.

```
SCTR::= SEQUENCE {
        identification      SCTRIdentSummary,
        systemReport        SysRepSummary
        }

SCTRIdentSummary ::= SEQUENCE {
        sctrId                      SCTRIdentification,
        testlab                     TestLabIdentification,
        client                      ClientIdentification,
        sut                         SUTIdentification,
        datesAndLocs                DatesAndLocs,
        scsRef                      DocReference,
        natureOfCT          [1]     CharString OPTIONAL,
        limitsAndReserv     [2]     AdditObservation OPTIONAL,
        recordOfAgreements          SEQUENCE OF Agreement,
        comments                    Comments
        }

SCTRIdentification ::= SEQUENCE {
        reference           DocReference,
        tlManager           NameAndSignature
        }

SysRepSummary ::= SEQUENCE OF
        SEQUENCE {
        iutDefRef           CharString,
                            -- link to agreements, the Implementation
                            --identifier requested in 9646-5 [4] /A 2.n can
                            -- be obtained from there
        protocols           SEQUENCE OF StandardReference,
        picsRefs            SEQUENCE OF DocReference,
        pixitRef            DocReference,
        pctrRef             DocReference,
        ats                 SEQUENCE OF StandardReference,
        atm                 SEQUENCE OF AbsTestMethod,
        motRef              MOTIdentification,
        confStatus          CharString,     -- 2 possible values
        staticConform       ErrorStatus,
        dynamicConform      ErrorStatus,
        testCasesRun        INTEGER,
        runAndPassed        INTEGER,
        runAndFailed        INTEGER,
        runAndInconc        INTEGER,
        observations        AdditObservation OPTIONAL
                                    -- in case of errors and/or problems
        }


DatesAndLocs ::= SEQUENCE {
        datesForTesting             SEQUENCE OF TimePeriod,
        dateOfReceipt       [1]     UTCTime OPTIONAL,
        sutLocation         [2]     CharString OPTIONAL
        }
```

5.3.4.4 Protocol Conformance Test Report (PCTR) and Proforma (PCTR_PF)

This subclause contains the definition of the Protocol Conformance Test Report (PCTR) conformance data object. The corresponding proforma (PCTR_PF) shares the same definition.

```
PCTR::= SEQUENCE {
        stdRef              StandardReference,
                                    -- The Standard for which this PCTR is for
        identification      PCTRIdentSummary,
        iutConfStatus       CharString,
                                    --2 possible values as in 9646-5 [4] - Annex B
        iutStaticConf       ErrorStatus,
        iutDynamicConf ErrorStatus,
        staticReport        CharString,
        dynamicReport  SEQUENCE OF DrItem,
                                    -- Items forming the test  campaign report
        tcOrder [1]    TcOrder OPTIONAL,
                                    -- Mandatory in ETR 040/ETG 016 [10]
        pctrAnnexes    [2]    SEQUENCE OF PctrAnnex OPTIONAL
        }

PCTRIdentSummary ::= SEQUENCE {
                    pctrId              PCTRIdentification,
                    iutSpec        IUTSpecification,
                    envId               ENVIdentification,
                                        -- describes the testing environment
                    limitsAndRes   [1]  AdditObservation OPTIONAL,
                    comments            Comments
                    }

PCTRIdentification ::= SEQUENCE {
                    reference      DocReference,
                    sctrRef        DocReference,
                    tcSupervisor   NameAndSignature,
                    tlManager      NameAndSignature,
                    testlab        TestLabIdentification
                    }

IUTSpecification ::= SEQUENCE {
                    iutId          IUTIdentification,
                    stdRefs        SEQUENCE OF StandardReference,
                    picsRef        SEQUENCE OF DocrefOrAnnex,
                    prevPCTR       SEQUENCE OF DocReference OPTIONAL
                    }

ENVIdentification ::= SEQUENCE {
                pixitRef                DocrefOrAnnex,
                atsRef                  StandardReference,
                atm                     AbsTestMethod,
                motIdent                MOTIdentification,
                protInfo        [1]     CharString OPTIONAL,
                datesForTesting         SEQUENCE OF TimePeriod,
                conflogRef              DocReference,
                retDate                 UTCTime
                }
```

```
DocrefOrAnnex ::= CHOICE {
                    docref  [1]     DocReference,
                                        -- meaning the required information is contained in
                                        -- the referenced document
                    annex   [2]     CharString
                                        -- meaning the required info is annexed to the test
                                        -- report as Annex X.
                    }

TcOrder ::= CHOICE {
                tcList                  [1]     SEQUENCE OF CharString,
                                                        -- the actual order of execution
                orderStatement [2]      CharString
                                                        -- if executed as appearing in ATS
                }

PctrAnnex ::= CHOICE {
                pics    [1]     PICS,
                pixit   [2]     PIXIT
                }

DrItem ::= SEQUENCE {
                atcref                  CharString,     -- ATC identifier
                selected                DeselectionStatus,
                run                     TestCaseError,
                verdict         [1]     VerdictValue OPTIONAL,
                                        --for the case where the test case is considered as
                                        --not run and no verdict is assigned
                observations    [2]     AdditObservation OPTIONAL
                                        -- as for instance :
                                        -- PICS/PIXIT item reference resulting
                                        --- in the deselection of the test case ,
                                        -- ATS or ETS defect report reference
                                        -- for test case not run due to ATS or ETS error
                }

VerdictAssigned ::= CHOICE {
                    automatic       [1]     VerdictValue,
                                                -- Verdict was assigned by MoT and not
                                                -- changed by test operator
                    manual          [2]     ManualVerdict
                    }

ManualVerdict ::= SEQUENCE {
                    value           VerdictValue,
                    observations    AdditObservation
                    }
```

5.3.4.5 Conformance Log (CFL)

This subclause contains the definition of the Conformance Log (CFL) conformance data object.

```
ConfLog::= SEQUENCE {
        identification   LogIdentification,              -- 9646-4 [3]  / 6.4 a)
        motId            MOTIdentification,              -- 9646-4 [3]  / 6.4 b)
        etsId            MOTIdentification OPTIONAL,     -- 9646-4 [3]  / 6.4 b)
        etcLogs SEQUENCE OF EtcLog                       -- 9646-4 [3]  / 6.4 c)
        }

LogIdentification ::= SEQUENCE {
                logId          CharString,     -- 9646-4 [3]  / 6.4 a)
                timePeriod     TimePeriod      -- 9646-4 [3]  / 6.4 a)
                }

EtcLog ::= SEQUENCE {
        atcRef                  CharString,             -- ATC reference, 9646-4 [3]  / 6.4 c)
        duration                TimePeriod,
        finalResult             FinalResult,
        etcId           [1]     CharString OPTIONAL,    -- ETC identification
        ltlist          [2]     LTStartStop OPTIONAL,   -- for multiparty,9646-4 [3] , 6.4 c)
        absEvents               SEQUENCE OF AbsEvent,   -- Abstract events
        realEvents              SEQUENCE {
                                        readingRules    CharString,
                                        Contents        SEQUENCE OF OCTET
                                        } OPTIONAL
                                                        -- Real 'executed' events
        }

LTStartStop ::= SEQUENCE OF
                 SEQUENCE {
                        id              CharString,
                                                -- Lower Tester Id. - ISO/IEC 9646-4 [3] /6.4c
                        startstop       INTEGER {start(0), stop(1)}
                        }

AbsEvent ::= SEQUENCE {
                order           OrderInfo,      -- 9646-4 [3]  / 6.4 i)
                info            CharString
                                        -- This should fulfill 9646-4 [3] , 6.4 d), e), f), g), h)
                                        -- Type CharString was chosen to impose as less
                                        -- restrictions as possible
                }

OrderInfo ::= CHOICE {          -- 9646-4 [3]  allows for these types of ordering information
        time    [1]     UTCTime,        -- only one of these shall be used
        seq     [2]     INTEGER         -- throughout one conformance log
        }

FinalResult ::= CHOICE {
                verdict [1]     VerdictValue,
                tcerror [2]     TestCaseError
                }
```

5.3.4.6 Protocol Implementation Conformance Statement (PICS) and Proforma (PICS_PF)

This subclause contains the definition of the Protocol Implementation Conformance Statement (PICS) conformance data object. The corresponding proforma (PICS_PF) shares the same definition.

```
PICS ::= SEQUENCE  {
        identification                  PICSIdentification,
        compnstructions         [1]     CompletionInstructions OPTIONAL,
        standardVersionId               StandardVersionIdentification,
        pICSProformaId          PICSProformaIdentification
        confStatement                   ConformanceStatement,
        iutCapabilities                 IUTCapabilities,
        relationsDef            [2]     SEQUENCE OF Relation OPTIONAL,
        predicateDef            [3]     SEQUENCE OF Predicate OPTIONAL,
        condStatusExprDef       [4]     SEQUENCE OF CondStatusExpression OPTIONAL
        }
        -- ISO/IEC 9646-7 [6]/9.3

PICSIdentification ::= SEQUENCE {
                pICSReference DocReference,
                                -- unique reference of the PICS, a priori
                                -- identical to PICS paper document ref.
                                -- not explicitly specified by ISO/IEC 9646-7 [6],
                iUTReference            IUTIdentification,      -- 9646-7 [6]/9.3.5.a
                sUTReference            SUTIdentification,      -- 9646-7 [6]/9.3.5.a
                bodyIssuingPICSInfo     ClientIdentification,   --9646-7 [6]/9.3.5.b
                iUTContactId            PersonalIdentification, -- 9646-7 [6]/9.3.5.c
                scsRef                  DocReference,           -- 9646-7 [6]/9.3.5.d
                                                -- reference of associated SCS
                }

CompletionInstructions ::= CharString
                -- instructions for PICS Proforma completion, ISO/IEC 9646-7 [6]/9.3.4,
                -- needed only for PICS Proforma definition.

StandardVersionIdentification ::=
        SEQUENCE {
                SEQUENCE OF
                        SEQUENCE {
                                standardReference               StandardReference,
                                support         BOOLEAN  {
                                                versionImlemented(TRUE),
                                                versionNotImlemented(FALSE)
                                                }
                        },
                versionParamEntryRef  PICSRowIdentifier OPTIONAL
                                -- reference to version paramater entry
                                -- in the PICS if such a parameter is
                                -- specified by the standard,
                                -- needed only for PICS Proforma definition.
                }
                -- ISO 9646-7/9.3.6,
                -- with consistency choices with others definition of standard references

PICSProformaIdentification ::=
                SEQUENCE {
                        pICSProfStandardRef             StandardReference,
                        pICSProfCorrigenda      [1]     PICSProformaCorrigenda OPTIONAL
                        }
                        -- ISO/IEC 9646-7 [6]/9.3.7
```

```
PICSProformaCorrigenda ::= SEQUENCE OF StandardReference
                                    -- list of the PICS Proforma corrigenda actually filled by the
                                    -- supplier
                                    -- ISO/IEC 9646-7 [6]/9.3.7

ConformanceStatement ::= BOOLEAN {
                            allMandatoryCapabilitiesImplemented (TRUE),
                            notAllMandatoryCapabilitiesImplemented (FALSE)
                            }
                                    -- ISO 9646-7 [6]/9.3.8.2.

IUTCapabilities ::= SEQUENCE OF PICSTable

PICSTable ::= SEQUENCE {
                    subclauseIdentifier     CharString,
                    tableHeader             CharString,
                    picsRows                SEQUENCE OF PICSRow
                    }

PICSRow ::= SEQUENCE  {
                rowIdentifier                   PICSRowId,
                itemName            [1]         CharString OPTIONAL,
                statusAndSupport                StatusAndSupport
                }

PICSRowId ::= CHOICE {
                    mnemonic            [1]     Charstring,
                    rowLocalRefNumber   [2]     INTEGER
                    }

PICSRowIdentifier ::= SEQUENCE {
                    subclauseIdentifier         Charstring,
                    rowlocalRefNumber           INTEGER
                    }
-- ISO/IEC 9646-7 [6]/9.3.8.3.a.3 ,
--Note that the CharString type authorizes the use of mnemonics as in ISO/IEC 9646-7 [6]/10.5
-- Note that ISO/IEC 9646-3/A.3.3.3.2.79 consider PICS & PIXIT references as free text

PICSRowElementIdentifier ::=
                     SEQUENCE {
                        pICSRowId               PICSRowIdentifier,
                        label           [1]     CharString OPTIONAL,
                                                -- provided if more than one response
                                                -- occurs in the identified row (e.g. 'a','b','c',
                                                -- etc..),
                                                -- ISO/IEC 9646-7 [6]/9.3.8.3.c
                        }
-- reference to one element of the set of status/support items in the row
```

```
StatusAndSupport ::=
        SEQUENCE OF
                SEQUENCE {
                        label           [1]     CharString OPTIONAL,
                                                -- provided if more than one response occurs
                                                -- in the identified row (e.g. 'a','b','c' etc)
                                                -- ISO/IEC 9646-7 [6]/9.3.8.3.c
                        confReq         [2]     CharString OPTIONAL,
                                                -- reference to static conformance
                                                -- requirement clause in
                                                -- ISO/IEC 9646-7 [6] /9.3.8.3.c.3
                        status          [3]     ItemStatus OPTIONAL,
                                                        -- status of the item as defined in the
                                                        -- standard
                        support         ItemSupport,
                                                        -- implementation answer to the item
                        comment                 CharString
                        }

ItemStatus ::= SEQUENCE {
                        status                          StatusType,
                        relation        [3]     INTEGER OPTIONAL,
                                                        -- reference to a definition which can be
                                                        -- found in the adhoc definition section
                                                        -- and expressing exclusive or selectable
                                                        -- option among a set of items
                                                        --  ISO/IEC 9646-7 [6]/10.2.1
                        allowedValues           [4]     SEQUENCE OF ValueConstraint OPTIONAL
                                                                -- restrictions or prescriptions on
                                                                -- supported values ISO
                                                                -- 9646-7 [6]/9.3.8.3.6
                        }

StatusType ::= CHOICE {
                        unconditional   [1]     UnConditionalStatus,
                        conditional     [2]     ConditionalStatus
                        }

UnConditionalStatus ::= INTEGER {
                                mandatory(0),
                                optional(1),
                                conditional(2),
                                prohibited(3),
                                outOfScope(4),
                                notApplicable(5)
                                }
                                        -- ISO 9646-7 [6]/8.3, ISO 9646-7 [6]/10.2.1,
```

```
ConditionalStatus ::=
        CHOICE  {
                explicitEntryRefPred    [1]     SEQUENCE OF
                                                        SEQUENCE  {
                                                                row     PICSRowElementIdentifier,
                                                                status  UnConditionalStatus
                                                                },
                                                -- ISO/IEC 9646-7 [6]/10.2.2.a the value is TRUE
                                                -- if the referenced entry answer is YES,
                                                -- FALSE otherwise.
                predicateRef            [2]     SEQUENCE OF
                                                        SEQUENCE  {
                                                                name    CharString,
                                                                status  UnConditionalStatus
                                                                },
                                                -- ISO/IEC 9646-7 [6]/10.2.2.b - predicate name
                                                -- to reference a predicate defined in the
                                                -- ad hoc definition section
                condStatusExpRef        [3]     INTEGER
                                                -- ISO/IEC 9646-7 [6]/10.2.2.b- identifier to reference
                                                        -- a conditional status expression defined in
                                                        -- the ad hoc definition section
                }
ItemSupport ::= SEQUENCE {
                        supportStatement        [1]     INTEGER {
                                                                implemented (0),
                                                                notImplemented (1),
                                                                notApplicable (2),
                                                                }
                                                                        -- ISO 9646-7 [6]/9.3.8.3.c.4,
                                                                        -- ISO 9646-7 [6]/10.3.1
                        supportValues           [2]     SEQUENCE OF ValueConstraint OPTIONAL,
                                                                        -- ISO 9646-7 [6]/9.3.8.3.c.6,
                        nonSupportSpec          [3]     INTEGER {
                                                                ignored (0),
                                                                error (1),
                                                                } OPTIONAL,
                                                                        -- ISO 9646-7 [6]/10.3.1
                        additComments           [4]     CharString OPTIONAL
                                                                        -- ISO 9646-7 [6]/9.3.8.3.c.7,
                                                                        -- ISO 9646-7 [6]/10.3.1
                        }

ValueConstraint ::= CHOICE  {
                        typeConstraints [1]     SEQUENCE OF ANY,
                        lengthConstraints       [2]     SEQUENCE OF INTEGER,
                        valueConstraints        [3]     SEQUENCE OF
                        }                               -- ISO/IEC 9646-7 [6]/9.3.8.3.c.6

SpecificConstraint ::= CHOICE  {
                                set     [4]     SEQUENCE OF ANY,
                                range   [5]     AnyRange
                                }
```

-- Definition of relations between items of the PICS

```
Relation ::= SEQUENCE  {
        reference           INTEGER,
        itemList            SEQUENCE OF PICSRowElementIdentifier,
                                    -- list of concerned items
        type                INTEGER {
                                atLeastOne (0),
                                oneAndOnlyOne (1)
                                }
                                    -- ISO 9646-7 [6]/10.2.1
                                    -- other type of relations may be added
        }

-- Definition of Predicates
Predicate ::= SEQUENCE  {
                name    CharString,
                body    PredicateBody
                }
                        -- ISO/IEC 9646-7 [6]/10.2.2

PredicateBody ::= CHOICE {
                explicitRowRef  [1]     PICSRowElementIdentifier,
                                        -- ISO/IEC 9646-7 [6]/10.2.2.a the value is TRUE
                                        -- if the referenced entry answer is YES,
                                        -- FALSE otherwise.
                relationalExp           [2]     RelationalExpression,
                predicateExp            [3]     PredicateExpression
                }

RelationalExpression :: = SEQUENCE {
                        operator            INTEGER {
                                                equal (O)
                                                greater (1),
                                                greaterAndEqual (2),
                                                notEqual (3),
                                                smaller (4),
                                                smallerAndEqual (5)
                                                },
                        firstOperand        PICSRowElementIdentier,
                                                -- must identify an "value" entry
                        secondOperand       RelationalOperand
                        }
                                -- ISO 9646-7 [6]/10.2.2,b

RelationalOperand ::= CHOICE {
                        edValue         [1]     ANY,
                        try             [2]     PICSRowElementIdentifier
                                                -- must identify an "value" entry
                        }


PredicateExpression ::= SEQUENCE  {
                        operator            INTEGER {
                                                and (0)
                                                or (1),
                                                not (2)
                                                },
                        firstOperand        PredicateOperand,
                        secondOperand       PredicateOperand OPTIONAL
                        }
                                -- ISO/IEC 9646-7 [6]/10.2.2
```

```
PredicateOperand ::= CHOICE {
                        predicate       [1]     CharString,
                                                    -- reference to a predicate definition
                        relation [2]    RelationalExpression,
                        expression      [3]     PredicateExpression
                        }
                                -- ISO/IEC 9646-7 [6]/10.2.2

-- Definition of conditional status expressions

CondStatusExpression ::= SEQUENCE  {
                                reference       INTEGER,
                                body            ConditonalExpression
                                }

ConditionalExpression ::= SEQUENCE {
                                if              Expression,
                                then            StatusExpression,
                                else            StatusExpression OPTIONAL
                                }

StatusExpression ::= CHOICE {
                        simple          [1]     UnConditionalStatus,
                        conditional     [2]     ConditionalExpression
                        }

Expression ::= CHOICE {
                        relational      [1]     RelationalExpression,
                        predicate       [2]     PredicateExpression
                        }
```

5.3.4.7 Protocol Implementation eXtra Information for Testing (PIXIT) and Proforma (PIXIT_PF)

This subclause contains the definition of the Protocol Implementation eXtra Information for Testing (PIXIT) conformance data object. The corresponding proforma (PIXIT_PF) shares the same definition.

```
-- This description is based on:
-- ISO/IEC 9646-1 [1] subclause 6.2.1 (note that there is some edition mismatch in this clause in
-- the mock-up reference version),
-- ISO/IEC 9646-5 [4] subclause 6.4.3, skeleton PIXIT Proforma in ISO/IEC 9646-5 [4] Annex C
-- (normative);
-- Guidance for a PIXIT in ISO/IEC 9646-5 [4] Annex D (informative).

PIXIT ::= SEQUENCE {
        pIXITIdentification      PIXITIdentification,
        aTSSummary               ATSSummary,
        testLabPIXITInfo         TestLabPIXITInfo,
        clientPIXITInfo          ClientPIXITInfo,
        sutPIXITInfo             SutPIXITInfo,
        ancillaryProtocols       SEQUENCE OF AncillaryProtocol,
                                          -- one element per protocol, even if
                                          -- there is more than one protocol
                                          -- for a single layer of the Reference
                                          -- Model.
        protocolLayerinfo        ProtocolLayerInfo
        specificContent SpecificInfo OPTIONAL
                                          -- Additional specific information, if
                                          -- required, corresponding to SUT
                                          -- limitations and environmental
                                          -- conditions as specified in ISO/IEC 9646-5
                                          -- [4] /C.5, and to protocol layer
                                          -- (IUT) procedural information as
                                          -- specified  in ISO/IEC 9646-5 [4] /C.6
        }

PIXITIdentification ::= SEQUENCE {
                proformaReference        DocReference,
                                          -- unique reference to the proforma, a priori
                                          -- identical to reference of corresponding
                                          -- paper document filled by the test lab
                                          -- when issuing the proforma
                pIXITReference      DocReference,
                                          -- unique reference to the PIXIT, a priori
                                          --identical to reference of  corresponding
                                          -- paper document  filled-in by the client.
                                          -- This duplicate reference is not explicit
                                          -- in ISO/IEC 9646. No dupliction of test lab
                                          -- and client names as in ISO/IEC
                                          -- 9646-5 [4] /C.1, ISO/IEC 9646-5 [4] /C.3
                                          -- and ISO/IEC 9646-5 [4] /C.4
                }
                        -- ISO/IEC 9646-5 [4] /C.1
```

```
ATSSummary ::= SEQUENCE {
                    protocolStandardReference,
                    aTS             StandardReference
                    aTM             AbsTestMethod
                    }
                            -- ISO/IEC 9646-5 [4] /C.2

TestLabPIXITInfo ::= SEQUENCE {
                    testLabIdentification           TestLabIdentification,
                                                        -- idem common definition.Contains
                                                        -- more information than specified in
                                                        -- ISO9646-5 [4] /C.3.
                                                        -- Consistency issue to discuss ...
                    mOTName                         MOTIdentification,
                                                        -- as done by CPS, addresses are
                                                        -- gathered with other protocol info
                    compInstructions        [1]     CharString OPTIONAL
                    }
                    -- ISO/IEC 9646-5 [4] /C.3, authorizes the addition of other information

ClientPIXITInfo ::= SEQUENCE {
                    clientIdentification            ClientIdentification,
                                                        -- idem common definition.Contains
                                                        -- more information than specified in
                                                        -- ISO9646-5 [4] /C.3.
                                                        -- Consistency issue to discuss ...
                    testFacilitiesReqSet    [1]     SEQUENCE OF TestFacilities OPTIONAL
                    }
                    -- ISO/IEC 9646-5 [4] /C.3, authorizes the addition of other information

TestFacilities ::=  SEQUENCE OF CharString  -- ISO9646-5 [4] /C.4

SutPIXITInfo ::= SEQUENCE {
                    SUTIdentification               SUTIdentification,
                    sCSReference                    DocReference,
                    machineInformation      [1]     CharString OPTIONAL,
                    oSInformation           [2]     CharString OPTIONAL,
                                                        -- IUT information are gathered with
                                                        -- "protocol layer information",
                    sUTLimitation           [3]     SEQUENCE OF ImplementationInfo,
                    environmentalcond       [4]     CharString OPTIONAL
                    }
                            -- ISO/IEC 9646-5 [4] /C.5


UTInformation ::= CharString   -- UT identification, and validation date,if any
```

```
AncillaryProtocol ::= SEQUENCE  {
                        name                    StandardReference,
                        version         [1]     CharString OPTIONAL,
                                                        -- must be provided if the protocol has a
                                                        -- "version" notion, even if the information is
                                                        -- provided in the relevant PICS
                        picsRef         [2]     DocReference OPTIONAL,
                                                        -- SUT ancillary protocol implem. PICS.
                        pixitRef        [3]     DocReference OPTIONAL,
                                                        -- SUT ancillary protocol implemen. PIXIT.
                        pctrRef [4]     DocReference OPTIONAL,
                                                        -- SUT ancillary protocol implem. PCTR.
                        protocolInfo    [5]     ProtocolInfo OPTIONAL
                                                        -- SUT ancillary protocol implementation
                                                        -- information.
                                                        -- Addressing information for both SUT
                                                        -- and LT
                        }
                        -- ISO/IEC 9646-5 [4] /C.6

ProtocolLayerInfo ::= SEQUENCE {
                        iUTReference            IUTIdentification,
                                                        -- ISO/IEC 9646-5 [4] /C.5
                        protocolName    StandardReference,
                                                        -- if the protocol has a "version"
                                                        -- notion, this information is provided
                                                        -- in the relevant PICS
                        picsRef                 DocReference ,
                                                        -- IUT PICS
                        protocolInfo            ProtocolInfo
                                                        -- IUT information
                                                        -- addressing information for both
                                                        -- SUT and LT
                        }
                        -- ISO/IEC 9646-5 [4] /C.7

ProtocolInfo ::= SEQUENCE {
                        addresses               AddressesForTesting,
                                                        -- addressing info for both SUT and LT
                        parameters      [1]     SEQUENCE OF TestSuiteParameter OPTIONAL,
                        timers          [2]     SEQUENCE OF Timer OPTIONAL,
                        procInfo        [3]     SEQUENCE OF ImplementationInfo OPTIONAL
                        }

AddressesForTesting ::= SEQUENCE {
                                iutAddr         SEQUENCE OF AddressElem,
                                lowerTesterAddr         SEQUENCE OF AddressElem,
                                }

AddressElem ::= SEQUENCE {
                        elemId          CharString,
                        values          CharString
                        }
```

```
TestSuiteParameter ::= SEQUENCE {
                        name                CharString,
                        type        [1]     ANY OPTIONAL,       -- test suite specific
                        picsClause  [2]     PICSRowElementIdentifier OPTIONAL,
                        range       [3]     AnyRange OPTIONAL,
                        value       [4]     ANY OPTIONAL
                                            -- parameter range and value is function of
                                            -- the type of the parameter
                        }
                        -- ISO/IEC 9646-5 [4] /C.7.2.2

Timer ::= SEQUENCE {
            name                CharString,
            type                ANY,
            picsClause  [1]     PICSRowElementIdentifier OPTIONAL,
            range               IntegerRange,
            value               INTEGER
            } -- ISO/IEC 9646-5 [4] /C.7.2.3

SpecificInfo ::= SEQUENCE OF ImplementationInfo

ImplementationInfo ::= SEQUENCE {
                        referenceNb         CharString
                                                -- Reference to the question or the
                                                -- relevant clause in the PIXIT
                        implOption          ImplementationOption
                        }
                        -- more precise than ISO/IEC 9646

ImplementationOption ::= CHOICE {
                        additInfo           [1]     AdditInfo,
                        selectedAnswer      [2]     SelectedAnswer
                        }

-- Specific information corresponding to :
-- * SUT limitations and environmental conditions as specified in ISO9646-5 [4] /C.5,
-- * Ancillary protocols specific information,
-- * protocol layer (IUT) procedural information as specified in ISO9646-5 [4] /C.6,
-- * and generally all PIXIT information which is neither addressing information,
-- nor identified test suite parameters .

-- "SelectedAnswer" covers additional information provided by the client in order to understand
-- if test cases can be executed, in the case where PIXIT proforma lists and identifies (with a
-- number),for these entries, all possible answers (the client ticking then the answer which
-- corresponds to the situation in his implementation).
-- Example of such an entry : "is this ASP invokable ? -Y/N-", "is this element observable ? -Y/N-".

-- Note that this kind of information is bound to the test case selection , and corresponds to an
-- implicit relationship between the ATS and the PIXIT proforma, (i.e. the ATC and the PIXIT
-- entries).

-- "AdditInfo" covers information used by the test operator for preparation or execution of the
-- test campaign  (e.g. "how to perform some test related activities ? ").

-- "AdditInfo" covers also all complementary information related to limitations and environmental
conditions,
-- and not structured as indicated above.

-- In all cases, this information must be referenced by the relevant PIXITproforma clause.
```

```
AdditInfo ::= SEQUENCE OF CharString

SelectedAnswer ::= SEQUENCE {
                        answerIdentifier INTEGER
                                                -- identifier of the answer chosen by
                                                -- the client, among all pre-defined
                                                -- answers in the PIXIT proforma
                        additComments CharString OPTIONAL
                                                -- to be provided if additional info is
                                                -- required/requested
                        }
```

5.3.4.8 Test Laboratory Checklist (TL_C)

This subclause contains the definition of the Test Laboratory Checklist (TL-C) conformance data object.

```
TestLabCheckList ::=
        SEQUENCE {
                requiredItems                   SEQUENCE OF CharString,
                complianceStmt          CharString,
                globalInfo              [1]     SEQUENCE OF CharString OPTIONAL,
                                                -- This may be used for global
                                                -- statements concerning the test
                                                -- lab applicable to all test services
                                                -- offered by the lab, eg.
                                                --.accreditation status, test lab
                                                --  contact, ut assistance, ...
                capRelatedInfo                  SEQUENCE OF TestService,
                docInfo                         SEQUENCE OF DocReference,
                addInfo                 [2]     SEQUENCE OF TLCAddInfo OPTIONAL
                }

TestService ::= SEQUENCE {
                        protocolId              StandardReference,
                        globalInfo      [1]     SEQUENCE OF CharString OPTIONAL,
                                                -- This may be used for global statements
                                                -- concerning the test service applicable
                                                -- to all ATMs offered in this service, eg.
                                                -- accreditation status, test lab contact,
                                                -- ut assistance, ...
                        atmOffered              SEQUENCE OF AtmOffered
                        }

AtmOffered ::= SEQUENCE {
                        atm                     AbsTestMethod,
                        globalInfo      [1]     SEQUENCE OF CharString OPTIONAL,
                                                -- This may be used for global statements
                                                -- concerning the ATM applicable to all test
                                                -- specs supported for this ATM, eg;
                                                -- accreditation status, test lab contact,
                                                -- ut assistance, ...
                        testSpecSupported       SEQUENCE OF TestSpecSupported
                        }
```

```
TestSpecSupported ::= SEQUENCE
                        testspec                StandardReference,
                        globalInfo      [1]     SEQUENCE OF CharString OPTIONAL,
                                                -- This may be used for global statements
                                                -- concerning the test spec applicable to all
                                                -- lower testers available for the test spec,
                                                -- e.g. accreditation status, test lab contact,
                                                -- ut assistance,...
                        ltInfo                  SEQUENCE OF LtInfo
                        }

LtInfo ::= SEQUENCE {
                ltid                            MOTIdentification,
                conformancestmnt        [1]     CharString OPTIONAL, --9646-5 [4] /6.3.1.2 d),
                                                -- may be given in global info
                comprehensivestmnt      [2]     CharString OPTIONAL, --9646-5 [4] /6.3.1.2 e),
                                                -- may be given in global info
                limitations             [3]     SEQUENCE OF CharString OPTIONAL,
                                                --9646-5 [4] /6.3.1.2 f)
                utspecs         [4]     SEQUENCE OF CharString OPTIONAL,
                                                --9646-5 [4] /6.3.1.2 g)
                tcps                    [5]     SEQUENCE OF CharString OPTIONAL,
                                                --9646-5 [4] /6.3.1.2 g)
                tlprocs                 [6]     SEQUENCE OF CharString OPTIONAL,
                                                --9646-5 [4] /6.3.1.2 h)
                utassistance            [7]     CharString OPTIONAL,
                                                --9646-5 [4] /6.3.1.2 note b)
                contactinfo             [8]     CharString OPTIONAL,
                                                --9646-5 [4] /6.3.1.2 note c)
                timeinfo        [9]     CharString OPTIONAL,
                                                -- 9646-5 [4] /6.3.1.2note d)
                accredStatus            [10]    CharString OPTIONAL
                                                -- 9646-5 [4] /6.3.1.2note e)
                                        -- This may reflect accreditation by different accreditation
                                        -- autorities and different states of accreditation, eg
                                        -- temporary,limited, ...
                }

TLCAddInfo ::= CHOICE {
                        structInfo      [1]     TestService,
                        freeInfo        [2]     SEQUENCE OF CharString
                        }
```

5.3.4.9 Client Checklist (CL_C) and Proforma (CL_C_PF)

This subclause contains the definition of the Client Checklist (CL_C) conformance data object. The corresponding proforma (CL_C_PF) shares the same definition.

```
ClientChecklist::=
        SEQUENCE {
                compliance              CharString,
                iut                     IUTIdentification,
                protsforTest            SEQUENCE OF StandardReference,
                tcps                    SEQUENCE OF
                                                SEQUENCE {
                                                        atm                     AbsTestMethod,
                                                        testabilityClaim  CharString,
                                                        tcp                     CharString
                                                        },
                physicalReqs    [1]     SEQUENCE OF
                                                SEQUENCE {
                                                        atm             [1]     AbsTestMethod
                                                                                OPTIONAL,
                                                        physicalReqID           INTEGER,
                                                        physicalReqDesc         CharString
                                                        } OPTIONAL,
                clientContact   [2]     CharString OPTIONAL
                }
```

5.3.4.10 Test Management Protocol Implementation Statement (TMPis) and Proforma (TMPis_PF)

This subclause contains the definition of the Test Management Protocol Implementation Statement (TMPis) conformance data object. The corresponding proforma (TMPis_PF) shares the same definition.

```
TMPis ::= CHOICE {
                formalTMPIS             [1]     PICS,           -- to be used if it exists
                informalTMPIS           [2]     CharString      -- Statement that the used UT
                                -- implements the TMP defined in the referenced specification
                }
```

5.3.4.11 Static Conformance Review Report (SCR_Report)

This subclause contains the definition of the Static Conformance Review Report (SCR_Report) conformance data object.

```
SCRReport ::= SEQUENCE {
                        identification          SCRReportIdentification,
                        statConfstatus          ErrorStatus,
                        confErrors              SEQUENCE OF
                                                        CHOICE {
                                                        statReq         [1]     StatReqNotMet,
                                                        inconsistency   [2]     Inconsistency
                                                        } OPTIONAL
                        }
```

```
SCRReportIdentification ::= SEQUENCE {
                        reference           DocReference,
                        date                UTCTime,
                        scsRef              DocReference,
                        protocolRef         StandardReference
                        }

StaReqNotMet ::= SEQUENCE {
                row                 PICSRowIdentifier,
                expected        [1] ItemSupport OPTIONAL,
                encountered     [2] ItemSupport OPTIONAL,
                comments        [3] CharString OPTIONAL
                }               -- all these definitions come from the PICS definition

Inconsistency ::= SEQUENCE {
                expected        [1] CharString OPTIONAL,
                encountered     [2] CharString OPTIONAL,
                comments        [3] CharString OPTIONAL
                }
```

5.3.4.12 Selection Agreement (SA)

This subclause contains the definition of the Selection Agreement (SA) conformance data object.

```
Agreements ::= SEQUENCE {
                iutDefRef       CharString,
                iut             IUTIdentifier,
                atm             AbsTestMethod,
                                    -- the type and location of the PCO(s) are implicitly
                                    -- determined by the abstract test method
                ats             StandardReference,
                bitRequested    INTEGER{yes(0),no(1)}
                }
```

5.3.4.13 Other ASN.1 type definitions

```
BITResults::= CharString        -- informing the client of the results of the execution of the BIT

ClientComments::= CharString    -- Client comments on documents sent out by the test lab

CLPid ::= CharString            -- Identifier of the client within the test lab environment

CSUTReady ::= Charstring        -- Client notifying the test lab that the SUT is ready for testing

Exception ::= BITSTRING {
                testingServiceNotProvided (0),
                scsUnacceptable(1),
                clientChecklistUnacceptable(2),
                picsUnacceptable(3),
                testCampaignUnproductive(4),
                tcpVerifResultsUnsatisfactory(5),
                clientWantNeAfterBit(6),
                neDuringTestCampaign(7),
                neToRestartCap(8),
                neChangedAfterExec(9),
                noSelectionAgrement(10),
                clientWantNeAfterScr(11)
                }
```

```
InfoRequest::= CharString       -- a request for information from the test lab to the client

InputErrors ::= SEQUENCE OF
                    SEQUENCE {
                        errorType               CharString,
                        location [1]        CharString OPTIONAL,
                        details             [2]     AdditObservation OPTIONAL,
                        recomActions        [3]     Charstring OPTIONAL
                        }

FailTestCase ::= CharString             -- ATC reference of the test which yielded a FAIL verdict

FailTestCaseRsp ::= INTEGER { reRun (0), ok (1), notOk (2) }

MOTId::= CharString     -- Identifier of the MOT selected for the current test campaign

ProtStdId ::= StandardReference
                -- Identification in the test lab  of the protocol standard to which conformance
                -- is claimed

ProtRef ::= CharString
                -- client specifying the protocol to which conformance is claimed

StatusSignals ::= INTEGER {
                    motReady (0),
                    sutReady (1),
                    scrComplete (2),
                    rspYes(3),
                    rspNo(4)
                    }

STS ::= SEQUENCE OF                                     -- List of ATC identifiers selected
                SEQUENCE {                              -- for a test campaign
                    atcRef          ATCRef,             -- ATC identifier
                    status          DeselectionStatus,  -- Whether it was selected
                    reason  [1]     AdditObservation    OPTIONAL
                                                        -- PICS/PIXIT item reference resulting
                                                        -- in the deselection of the test case
                    }

TCPResults::= CharString        -- informing the client of the results of the TCP verification activity
```

6 Interchanging conformance data objects

6.1 Introduction

The model in Clause 5 of this ETR identifies a number of information flows between the CAP processes of a test system which in real test environments can be implemented in a number of ways. There are advantages in specifying an interchange format for these information flows in order that the data can be exchanged between the processes in a form that allows further processing.

This ETR identifies a number of conformance data objects from the model for which an ASN.1 syntax specification is specified to allow interchange of the data object representing the information flow. The conformance data objects selected are those initially thought to be best candidates for interchange between processes in the model.

This ETR does not identify applications that may process these interchange objects. It is recognised that different recipients of a given interchanged data object may process them in different ways. Some of the conformance data objects specify information flows (e.g. client check list) that have traditionally been exchanged as paper documents while other data objects may only require exchange for data processing purposes. This has led to the specification of three different possible interchange formats for each of selected conformance data objects. These forms are the ASN.1 values, the ODA representation and the ASN.1 type notation of the syntax definition of the conformance data objects.

6.2 Conformance data objects for which an interchange format is specified

Table 2

An interchange format is specified for:	An Interchange format is not specified for:
PICS	ProtStdId
PICS_PF	Exception
PIXIT	PICSErrors
PIXIT_PF	PIXITErrors
TMPis	TMPisErrors
TMPis_PF	MOTId
SCR_Report	C_SUT_Ready
PCTR	STS
PCTR_PF	StatusSignals
SCTR	SCTRId
SCTR_PF	CL_Pid
SCS	BIT_Result
SCS_PF	CLCErrors
SA	doesClientWantBit
CFL	doesClientWantToContAfterBit
CL_C	doesClientWantLog
CL_C_PF	FAIL
TL_C	FAILANS
	SCSErrors
	TCP_Result
	TCP_Res_Com
	SA_Com
	Report_Com

6.3 Interchange format definition

6.3.1 Interchange units

Those conformance data objects which have an interchange format are specified in the **InterchangeUnit** data type. The interchange unit specification contains a **header** which allows the identification of the entity or entities associated with the conformance assessment process to which the interchange unit applies. The header also includes an identification and start time of the conformance assessment process. The body of the interchange unit allows one or more of the conformance data objects to be specified in one interchange unit.

An interchange unit is defined as follows:

```
InterchangeUnit ::= SEQUENCE {
        header  HeaderInfo,
        body    SEQUENCE OF
                CHOICE {
                    pics        [1]   PicsObject,
                    picsPf      [2]   PicsPFObject,
                    pixit       [3]   PixitObject,
                    pixitPf     [4]   PixitPFObject,
                    tmpIs       [5]   TMPisObject,
                    tmpIsPf     [6]   TMPisPFObject,
                    scrReport   [7]   SCRReportObject,
                    pctr        [8]   PctrObject,
                    pctrPf      [9]   PctrPFObject,
                    sctr        [10]  SctrObject,
                    sctrPf      [11]  SctrPFObject,
                    scs         [12]  ScsObject,
                    scsPf       [13]  ScsPFObject,
                    sa          [14]  SaObject,
                    cfl         [15]  CflObject,
                    clc         [16]  ClcObject,
                    clcPF       [17]  ClcPFObject,
                    tlc         [18]  TlcObject
                }

HeaderInfo::= SEQUENCE {
        testLab [1]     OrgName     OPTIONAL,   -- at least one of these
        client  [2]     OrgName     OPTIONAL,   -- should be specified
        cap             SEQUENCE {
                            id          CharString,
                                        -- a unique identification assigned
                                        -- by the test lab for a given cap
                            starttime   UTCTime
                            }
        }
```

Using this interchange unit syntax has the following benefits:

- more than one conformance data object can be included in one interchange unit;
- including the header information allows a unique global identification for the interchanged conformance data objects.

Each conformance data object syntax within the interchange unit may be present in up to three machine processable forms:

dataForm: a set of ASN.1 values representing a data processable form of the data objects;
renditionForm: an ODIF stream representing an ODA [11] document version of the human readable form of the object;
syntaxForm: a CharString value representing the ASN.1 type notation of the object syntax.

It may not always be necessary or sensible to interchange all three for a given object. The three forms are provided for consistency and because the model is intended to be independent of any particular application.

ODA was selected as the document architecture for the human readable version of the data objects since it specifies an independent model for documents that allows the representation of document content, structure and **layout**. However, since there are currently no ODA document versions of the conformance data objects available there are no ODIF streams specified even though placeholders are provided in each data object's interchange format specification.

6.3.2 Interchange format for the PICS

The interchange format for the PICS conformance data object is defined as follows:

```
PicsObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     PICS,       -- see subclause 5.3.4.6
                        renditionForm   [2]     PicsODIF,   -- not specified
                        syntaxForm      [3]     CharString  -- see subclause 5.3.4.1
                        }
```

6.3.3 Interchange Format for the PICS Proforma

The interchange format for the PICS Proforma conformance data object is defined as follows:

```
PicsPFObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     PICS,       -- see subclause 5.3.4.6
                        renditionForm   [2]     PicsODIF,   -- not specified
                        syntaxForm      [3]     CharString  -- see subclause 5.3.4.1
                        }
```

6.3.4 Interchange format for the PIXIT

The interchange format for the PIXIT conformance data object is defined as follows:

```
PixitObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     PIXIT,      -- see subclause 5.3.4.7
                        renditionForm   [2]     PixitODIF,  -- not specified
                        syntaxForm      [3]     CharString  -- see subclause 5.3.4.1
                        }
```

6.3.5 Interchange format for the PIXIT Proforma

The interchange format for the PIXIT Proforma conformance data object is defined as follows:

```
PixitPFObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     PIXIT,      -- see subclause 5.3.4.7
                        renditionForm   [2]     PixitODIF,  -- not specified
                        syntaxForm      [3]     CharString  -- see subclause 5.3.4.1
                        }
```

6.3.6 Interchange format for the TMPis

The interchange format for the TMPis conformance data object is defined as follows:

```
TMPisObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     TMPis,          -- see subclause 5.3.4.10
                        renditionForm   [2]     TMPisODIF,      -- not specified
                        syntaxForm      [3]     CharString      -- see subclause 5.3.4.1
                        }
```

6.3.7 Interchange format for the TMPis Proforma

The interchange format for the TMPis Proforma conformance data object is defined as follows:

```
TMPisPFObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     TMPis,          -- see subclause 5.3.4.10
                        renditionForm   [2]     TMPisODIF,      -- not specified
                        syntaxForm      [3]     CharString      -- see subclause 5.3.4.1
                        }
```

6.3.8 Interchange format for the SCR Report

The interchange format for the SCR Report conformance data object is defined as follows:

```
SCRReportObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     SCRReport,      -- see subclause 5.3.4.11
                        renditionForm   [2]     SCRReportODIF,-- not specified
                        syntaxForm      [3]     CharString      -- see subclause 5.3.4.1
                        }
```

6.3.9 Interchange format for the PCTR

The interchange format for the PCTR conformance data object is defined as follows:

```
PctrObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     PCTR,           -- see subclause 5.3.4.4
                        renditionForm   [2]     PCTRODIF,       -- not specified
                        syntaxForm      [3]     CharString      -- see subclause 5.3.4.1
                        }
```

6.3.10 Interchange format for the PCTR Proforma

The interchange format for the PCTR Proforma conformance data object is defined as follows:

```
PctrPFObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     PCTR,           -- see subclause 5.3.4.4
                        renditionForm   [2]     PCTRODIF,       -- not specified
                        syntaxForm      [3]     CharString      -- see subclause 5.3.4.1
                        }
```

6.3.11 Interchange format for the SCTR

The interchange format for the SCTR conformance data object is defined as follows:

```
SctrObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     SCTR,           -- see subclause 5.3.4.3
                        renditionForm   [2]     SCTRODIF,       -- not specified
                        syntaxForm      [3]     CharString      -- seesubclause 5.3.4.1
                        }
```

6.3.12 Interchange format for the SCTR Proforma

The interchange format for the SCTR Proforma conformance data object is defined as follows:

```
SctrPFObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     SCTR,           -- see subclause 5.3.4.3
                        renditionForm   [2]     SCTRODIF,       -- not specified
                        syntaxForm      [3]     CharString      -- see subclause 5.3.4.1
                        }
```

6.3.13 Interchange format for the SCS

The interchange format for the SCS conformance data object is defined as follows:

```
ScsObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     SCS,            -- see subclause 5.3.4.2
                        renditionForm   [2]     SCSODIF,        -- not specified
                        syntaxForm      [3]     CharString      -- see subclause 5.3.4.1
                        }
```

6.3.14 Interchange format for the SCS Proforma

The interchange format for the SCS Proforma conformance data object is defined as follows:

```
ScsPFObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     SCS,            -- see subclause 5.3.4.2
                        renditionForm   [2]     SCSODIF,        -- not specified
                        syntaxForm      [3]     CharString      -- see subclause 5.3.4.1
                        }
```

6.3.15 Interchange format for the SA

The interchange format for the SA conformance data object is defined as follows:

```
SaObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     Agreements,     -- see subclause 5.3.4.12
                        renditionForm   [2]     SAODIF,         -- not specified
                        syntaxForm      [3]     CharString      -- see subclause 5.3.4.1
                        }
```

6.3.16 Interchange format for the CFL

The interchange format for the CFL conformance data object is defined as follows:

```
CflObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     ConfLog,        -- see subclause 5.3.4.5
                        renditionForm   [2]     CFLODIF,        -- not specified
                        syntaxForm      [3]     CharString      -- seesubclause 5.3.4.1
                        }
```

6.3.17 Interchange format for the CL_C

The interchange format for the CL_C conformance data object is defined as follows:

```
ClcObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     ClientChecklist, -- see subclause 5.3.4.9
                        renditionForm   [2]     CLCODIF,        -- not specified
                        syntaxForm      [3]     CharString      -- see subclause 5.3.4.1
                        }
```

6.3.18 Interchange format for the CL_C Proforma

The interchange format for the CL_C Proforma conformance data object is defined as follows:

```
ClcPFObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     ClientCheckList,  -- see subclause 5.3.4.9
                        renditionForm   [2]     CLCODIF,        -- not specified
                        syntaxForm      [3]     CharString      -- seesubclause 5.3.4.1
                        }
```

6.3.19 Interchange format for the TL_C

The interchange format for the TL_C conformance data object is defined as follows:

```
TlcObject ::= SEQUENCE OF
                    CHOICE {
                        dataForm        [1]     TestLabCheckList,-- see subclause 5.3.4.8
                        renditionForm   [2]     TLCODIF,        -- not specified
                        syntaxForm      [3]     CharString      -- seesubclause 5.3.4.1
                        }
```

7 Test tool support for CAP processes and exchange of objects

7.1 Introduction

The exchange of conformance data objects might typically take place in one of the following example scenarios:

- conformance data objects exchanged between a test laboratory and its clients:
  - an example of this could be when a test laboratory provides its client with the **renditionForm** (i.e. human readable form) of a conformance log for a test campaign;

- conformance data objects can be exchanged between processes within a given test system:
  - an example of this could be the **dataForm** (i.e. actual value) of a conformance log produced by a "Test Parameterization and Execution" process which is subsequently used by a "PCTR Production" process;

- conformance data objects can be exchanged between separate test environments:
  - an example of this could be the exchange of the **syntaxForm** (i.e. the ASN.1 type notation) of the conformance log definition in order for one test environment to indicate to another, the format (but not value) required for any conformance logs it receives.

To facilitate the exchange of conformance data objects in the interchange format within such scenarios, a mechanism is introduced to allow test tool developers to identify which CAP processes their implementations support and which of the conformance data objects the implementation can interchange.

The purpose of a Test Tools Support Statement Proforma (TTSS_PF) is to allow for easier evaluation/comparison of test tools, harmonised interchange of a number of conformance data objects and possibly re-use of generic functions from one technical area to another.

7.2 Test tools support statement proforma

7.2.1 Introduction

The Test Tools Support Statement Proforma (TTSS_PF) consists of three sub-proformas: the test tools description proforma, the CAP processes proforma and the access points proforma. When completed by a test tool provider, the TTSS_PF becomes a Test Tools Support Statement (TTSS). In subclause 7.2.2, an ASN.1 definition of the required data types is provided. A tabular form of the proforma is provided in Annex B.

The Test Tools Description Proforma (TTD_PF) lists the type of information a test tool provider can specify in order that they may be described and identified as a set of test tools supporting the ISO/IEC 9646 [1] - [6] methodology. When completed by a test tool provider, the TTD_PF becomes a Test Tools Description (TTD).

The CAP Processes Statement Proforma (CAPPS_PF) provides a list of the processes in ISO/IEC 9646 [1] - [6] which a given test tool can identify as being **supported, partially supported or not supported.** When completed by a test tool provider, the CAPPS_PF becomes a CAP Processes Statement (CAPPS).

The Access Points Statement proforma (APS_PF) provides for each interchange data object that can be imported or exported, for one or more processes, some specific access point in a given tool. When completed by a test tool provider, the APS_PF becomes an Access Points Statement (APS).

7.2.2 An ASN.1 definition of the test tools support statement proforma

This subclause contains the ASN.1 definition of the Test Tools Support Statement (TTSS) data object. The corresponding proforma (TTSS_PF) shares the same definition.

```
TTSS::= SEQUENCE {
        description     TestToolsDescription,
        processes       CAPProcessesStatement,
        accespoints     AccessPointsStatement
        }

TestToolsDescription::= SEQUENCE {
                tTSSNumber      INTEGER,
                tTSSSupplier    OrgName,
                tTSSDate        UTCTime,
                authority       CharString,
                tools           SEQUENCE OF
                                SEQUENCE {
                                toolId          ToolIdentification,
                                capProcesses    CAPProcessesStatement,
                                accessPoints    AccessPointsStatement
                                }
                }

ToolIdentification::= SEQUENCE {
                name            OrgName,
                release         CharString,
                version         CharString,
                status          CharString,
                date            UTCTime
                }

CAPProcessesStatement::=SEQUENCE OF
                SEQUENCE {
                        process         ProcessId
                        support         INTEGER {
                                        supported(0),
                                        partialSupport(1),
                                        notSupported(2),
                                        notApplicable(3)
                                        },
                        comments        CharString OPTIONAL
                        }

ProcessId::= INTEGER {
        clientRequirementsEvaluation(0), picsAdministration(1), atmAtsselection(2),
        pixitAdminstration(3), motPreparation(4), staticConformanceReview(5),
        testcaseSelection(6), tcpVerificationTestCampaign(7), produceSCTR(8),
        producePCTR(9), negotiatedExit(10)
        }

AccessPointsStatement::= SEQUENCE OF
                SEQUENCE {
                        dataObject              DataObjectId,
                        forProcess              SEQUENCE OF AccessType
                        }
```

```
AccessType::= SEQUENCE {
                processId       ProcessId,
                support         INTEGER {
                                        supported(0),
                                        notSupported(1),
                                        notApplicable(2)
                                        },
                interchange     INTEGER {
                                        forExportandImport(0),
                                        forExportOnly(1),
                                        forImportOnly(2)
                                        },
                accessPoint     AccessPointDescription
                }

DataObjectId::= INTEGER {
                        pics(1), picsPF(2), pixit(3), pixitPF(4), tmpis(5), tmpisPF(6),
                        scrReport(7), pctr(8), pctrPF(9), sctr(10), sctr(11), scs(12),
                        scsPF(13), sa(14), cfl(15), clc(16), clcPF(17), tlc(18)
                }

AccessPointDescription::= SEQUENCE {
                communication  CharString,          -- description of communication medium
                encoding               CharString   -- description of encoding
                }
```

8 Extension of the model for protocol profile testing

8.1 Introduction

The ISO/IEC 9646 [1] - [6] standard provides a general methodology for testing the conformance of products to OSI specifications which the products claim to implement. The OSI specifications can be:

- the specification of an OSI protocol;

- the specification of a transfer syntax used in combination with a specific OSI protocol;

- the specification of a combination of OSI protocols, possibly used in combination with a specified syntax;

- the specification of an OSI protocol profile.

The testing methodology for base standards or recommendations has gained international standard status and is currently specified in the main text of ISO/IEC 9646 parts 1 to 5 [1] - [4]. Protocol profile conformance testing is currently being addressed, and is specified in ISO/IEC draft amendment 1 (Protocol Profile Testing Methodology) of parts 1 through 5, in part 6 (Protocol Profile Test Specification) and specific aspects of part 7 (Implementation Conformance Statements).

In ISO/IEC 9646 [1] - [6], protocol profile conformance testing is based on the methodology and test specifications existing for the base standards and recommendations being referenced by the profile specification. Where the profile specification goes further in its specification than the related base standards or recommendations, additional conformance requirements need to be expressed and these need to be addressed as part of the conformance assessment process. The profile conformance requirements are expressed by:

- the ICS proforma of each base standard/recommendation being referenced by the profile;

- the **Profile Requirements List** (profile RL), expressing the profile constraints on the status and/or allowed answers in the ICS proforma of each protocol;

- the **Profile specific ICS proforma** specifying additional questions on the profile but which are not directly associated with any of the referenced protocols.

In order to test an implementation for conformance to a profile, a **Profile Test Specification** (PTS) is necessary. The PTS is the set of all conformance testing documents which are needed to assess the conformance to a profile. Its table of contents is listed in a standardised document called the **PTS-Summary**. The PTS-Summary references all the documents necessary to completely specify conformance to a profile, that is, the base protocol conformance testing specifications and the specific material created for the profile (**Profile Specific Test Specification** (PSTS)).

8.2 The conformance assessment process to profiles

ISO/IEC 9646 [1] - [6] currently addresses the testing methodology for conformance to profile specifications as an extension of protocol testing. The testing to be carried out has to be in accordance with the profile test specifications.

Accordingly, a model of the conformance assessment process to profiles can be derived from the model of protocol testing presented in Clause 5 of this ETR, by extending it in the following way:

- the general sub-tasks of the process need to be extended to address the profile specific aspects. This is the case for the client requirements evaluation and the SCTR Production processes;

- the protocol specific sub-tasks need to be extended or repeated to handle specific profile requirements. For example, this is the case for the static requirements evaluation process, which needs to be repeated for each base protocol being referenced by the profile, and adapted to address the profile specific requirements as expressed by the profile requirements list. The specific case of the profile specific ICS also needs to be addressed;

- a number of conformance data objects need to be adapted to be valid in the case of profile testing (e.g. the SCS, the SCTR, etc.), while some new information flows need to be identified and defined (e.g. the profile requirements list).

Because the extensions to ISO/IEC 9646 [1] - [6] are not yet fully standardised and stable, this ETR does not identify all the necessary extensions to the model for profile testing. These extensions could be developed when the extensions to ISO/IEC 9646 [1] - [6] have reached International Standard status and when further experience on the revised methodology has been gained by the test laboratories.

However, to illustrate how the general approach presented by this ETR can be extended for protocol profile testing, the definition of the profile requirements list is provided in subclause 8.3 the corresponding interchange format can be found in subclause 8.4.

8.3 Definition of the profile requirements list

This subclause contains the definition of the profile requirements list. This definition is based on that part of ISO/IEC 9646-7 [6], and makes use of type definitions previously introduced for the protocol ICS (PICS). It is foreseen that this definition will be applicable to a wide range of profiles.

```
RL ::= SEQUENCE  {
        identification                  RLIdentification,
        completionInstructions  [1]     CompletionInstructions OPTIONAL,
        profileId                       ProfileVersionIdentification,
        profileRequirements             ProfileRequirements,
        }

RLIdentification ::= SEQUENCE {
                ptsSummary              DocReference,
                rLStandardRef           StandardReference,
                rLCorrigenda    [1]     RLCorrigenda OPTIONAL
                }
                --"Standard" reference to the Profile Requirements List and
                -- associated technical corrigenda, if any

CompletionInstructions ::= CharString
                -- although the RL is not a proforma, instructions for the
                -- reading of the RL and the production of the Profile ICS may be
                -- added
                -- in line with  ISO/IEC 9646-7 [6]/9.3.4, but applied to the RL

ProfileVersionIdentification ::= SEQUENCE OF StandardReference      -- the version(s) of the
                                                                    -- profile

RLCorrigenda ::= SEQUENCE OF StandardReference
                                -- list of the RL corrigenda

ProfileRequirements ::= SEQUENCE OF ProtSpecRequirements
                                -- the profile requirements as a set of constraints
                                -- expressed on a protocol basis

ProtSpecRequirements ::=
        SEQUENCE {
                protocold               StandardVersionIdentification,
                                        -- protcol reference(s)
                picsPFId                PICSProformaIdentification,
                                        -- corresponding PICS  proforma
identification
                constraints             SEQUENCE OF ProtProfConstraints,
-- the following are provided if  needed to support the definition of some of the specific profile
-- requirements (as defined for the PICS)
                relationsDef    [1]     SEQUENCE OF Relation OPTIONAL,
                predicateDef    [2]     SEQUENCE OF PredicateOPTIONAL,
                condStatusExprDef [3]   SEQUENCE OF
                                                CondStatusExpression OPTIONAL
        }
```

```
ProtProfConstraints ::= SEQUENCE {
                        tableIdentifier   [1]   Charstring OPTIONAL,
                        tableHeader       [2]   Charstring OPTIONAL,
                        picsRows                SEQUENCE OF RLPICSRow
                                                    -- a PICS row as modified by this RL
                        }
                        -- this definition provides for both cases envisaged
                        -- in ISO/IEC 9646-7 [6]/9.7.3 : simple list of constraints or a copy of
some
                        --tables from the PICS

RLPICSRow ::= SEQUENCE {
                rowIdentifier           PICSRowIdentifier,      -- as defined for the PICS
                itemName        [1]     CharString OPTIONAL,
                itemStatus              RLStatus
                }

RLStatus ::=
        SEQUENCE OF
                SEQUENCE {
                label                   [1]     CharString OPTIONAL,
                                                -- provided if more than one response may
                                                -- occur in the identified row(e.g. 'a','b','c',
                                                -- etc..),
                                                -- ISO/IEC 9646-7 [6]/9.3.8.3.c
                confReq                 [2]     CharString OPTIONAL,
                                                -- reference to static conformance
                                                -- requirement clause - ISO/IEC 9646-7 [6]
                                                -- 7/9.3.8.3.c.3
                baseStdReq              [3]     ItemStatus OPTIONAL,
                                                -- status of the item as defined in the
                                                -- base standard
                                                -- (as defined for the PICS)
                profileConstraint       [4]     ItemStatus OPTIONAL,
                                                -- status of the item as constrained by the
                                                -- protocol profile
                }
-- the profile constraints are defined in a similar manner than for the PICS, but profile specific
-- relation, predicate and conditional status expressions definitions can be found in the Protocol
-- Specific Requirements expressed by the profile

ItemStatus ::= SEQUENCE {
                status                          StatusType,
                relation        [3]     INTEGER OPTIONAL,
                                                -- reference to a definition which can be
                                                -- found in the adhoc definition section
                                                -- and expressing exclusive or selectable
                                                -- option among a set of items
                                                --  ISO/IEC 9646-7 [6]/10.2.1
                allowedValues   [4]     SEQUENCE OF ValueConstraintOPTIONAL
                                                    -- restrictions or prescriptions on
                                                    -- supported values ISO
                                                    -- 9646-7 [6]/9.3.8.3.6
                }
```

-- the following definitions can be found in the section defining the PICS object:
-- Relation - the definition of a relation between PICS entries
-- Predicate - the definition of a predicate
-- CondStatusExpression - the definition of a conditional status expression
-- PICSRowIdentifier - unambiguous identification of a PICSrow
-- StatusType - the type of status for the current entry(eg 'M', 'O', etc.)
-- ValueConstraint - restrictions on supported values

8.4 Interchange format for the profile requirements list

The interchange format for the Profile Requirements List is defined as follows:

```
InterchangeUnit ::= SEQUENCE {
                        header  HeaderInfo,                     --see subclause 6.3.1
                        body            SEQUENCE OF
                                                CHOICE {
                                                        .
                                                        .       -- see subclause 6.3.1
                                                        .
                                                        profileRL       [31]    ProfileRL
                        }

ProfileRL ::= SEQUENCE OF
                CHOICE {
                        dataForm        [1]     RL,
                        renditionForm   [2]     RLODIF,         -- Not specified
                        syntaxForm      [3]     CharString
                        }
```

Annex A (informative): Message sequence charts

Message Sequence Charts (MSC), are specified in ITU-TS Recommendation Z.120 and are used to describe:

- signal exchanges between processes;
- specific executions of the system;
- exceptional behaviours.

The following MSCs are provided as examples to illustrate the **normal behaviour** (i.e. without exceptions or errors) of the system, for each block defined in the model. They are in accordance with the SDL model specified in subclause 5.3.1:

- Block B21_Test_Preparation;

- Block B22_Test_Operation;

- Block B23_Test_Report_Production;

- Block Negotiated_Exit.

In each MSC, the client (environment of the SDL model) is represented on the left hand side of each figure. Likewise, other SDL blocks are represented (as processes) on the right hand side of the figure. These blocks exchange signals that are going to (or coming from) one of their internal processes.

Each signal is indicated with its parameter(s) type(s) corresponding to the ASN.1 type previously described in subclause 5.3.4.

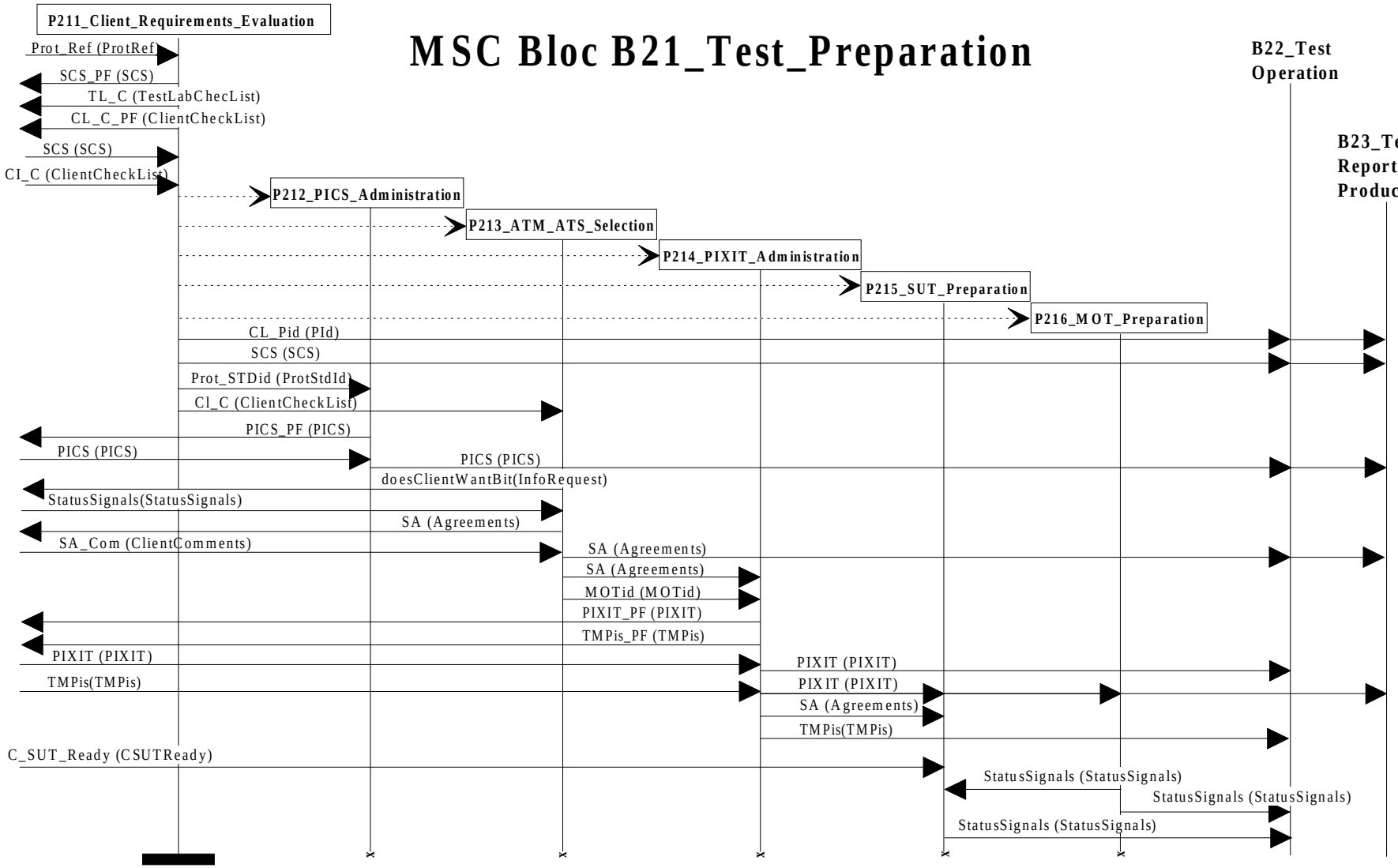


Figure A.1: MSC for Block B21 - test preparation

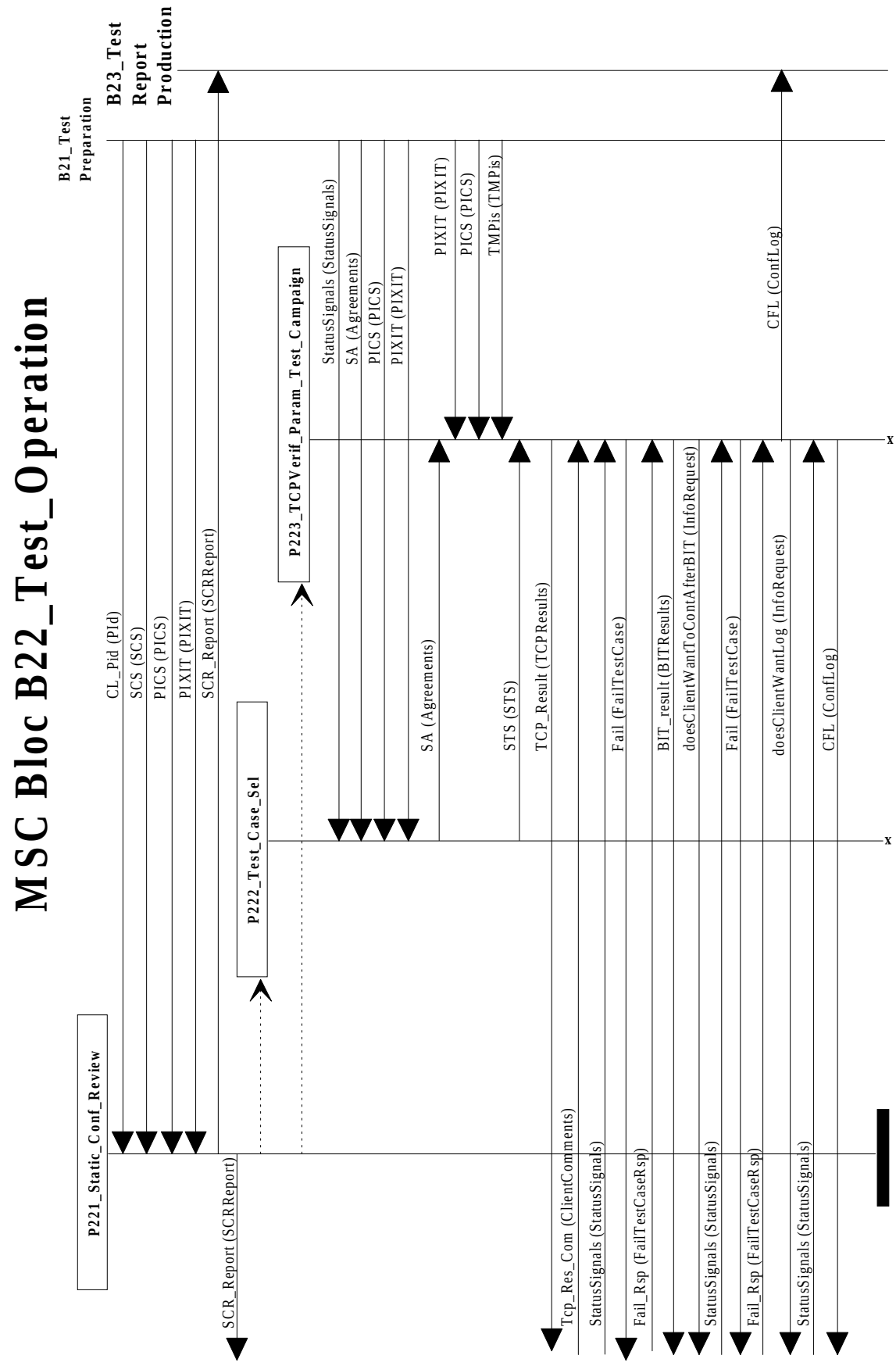


Figure A.2: MSC for Block B22 - test operation

MSC Bloc B23_Test_Report_Production

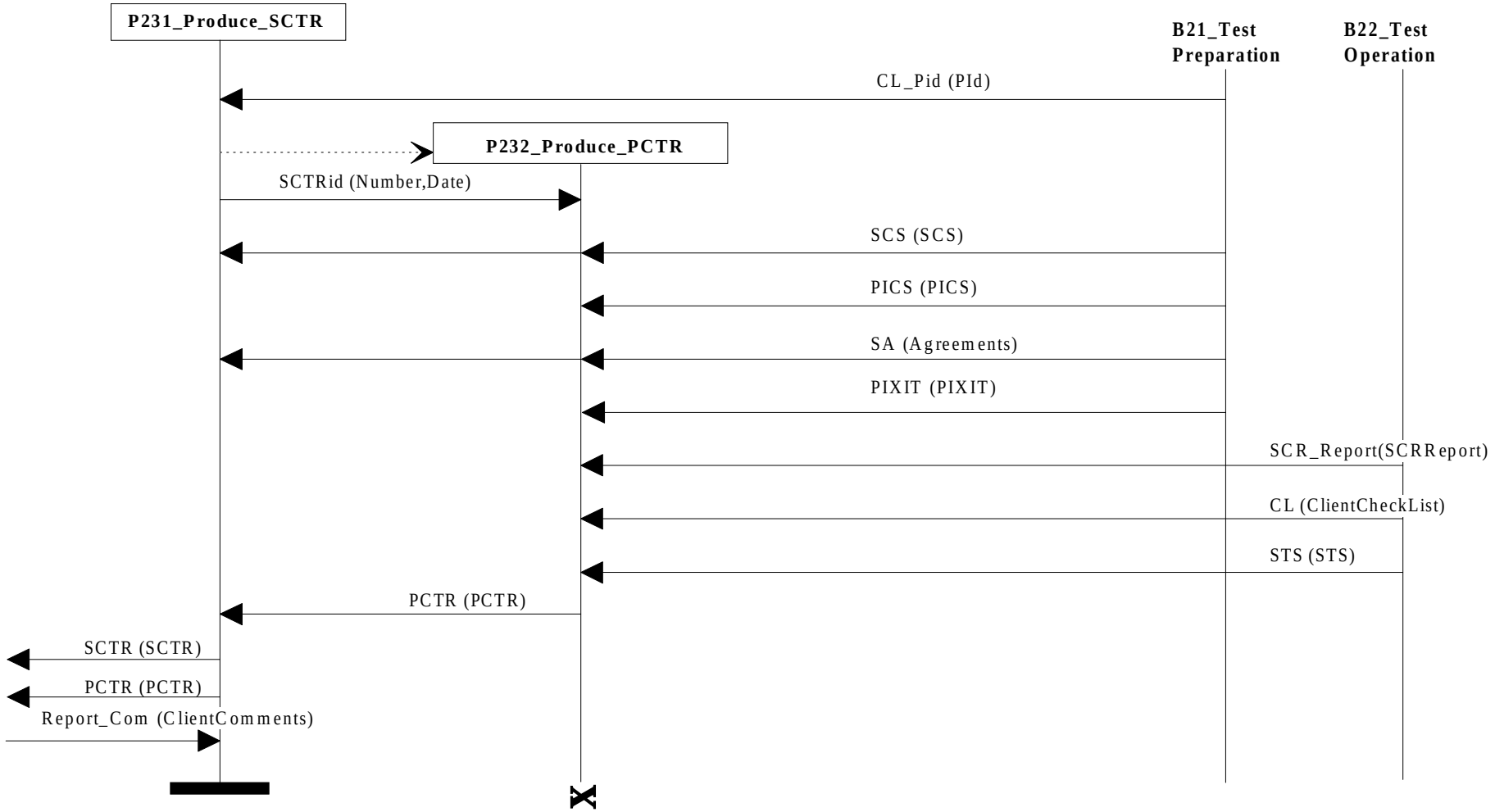


Figure A.3: MSC for Block B23 - test report production

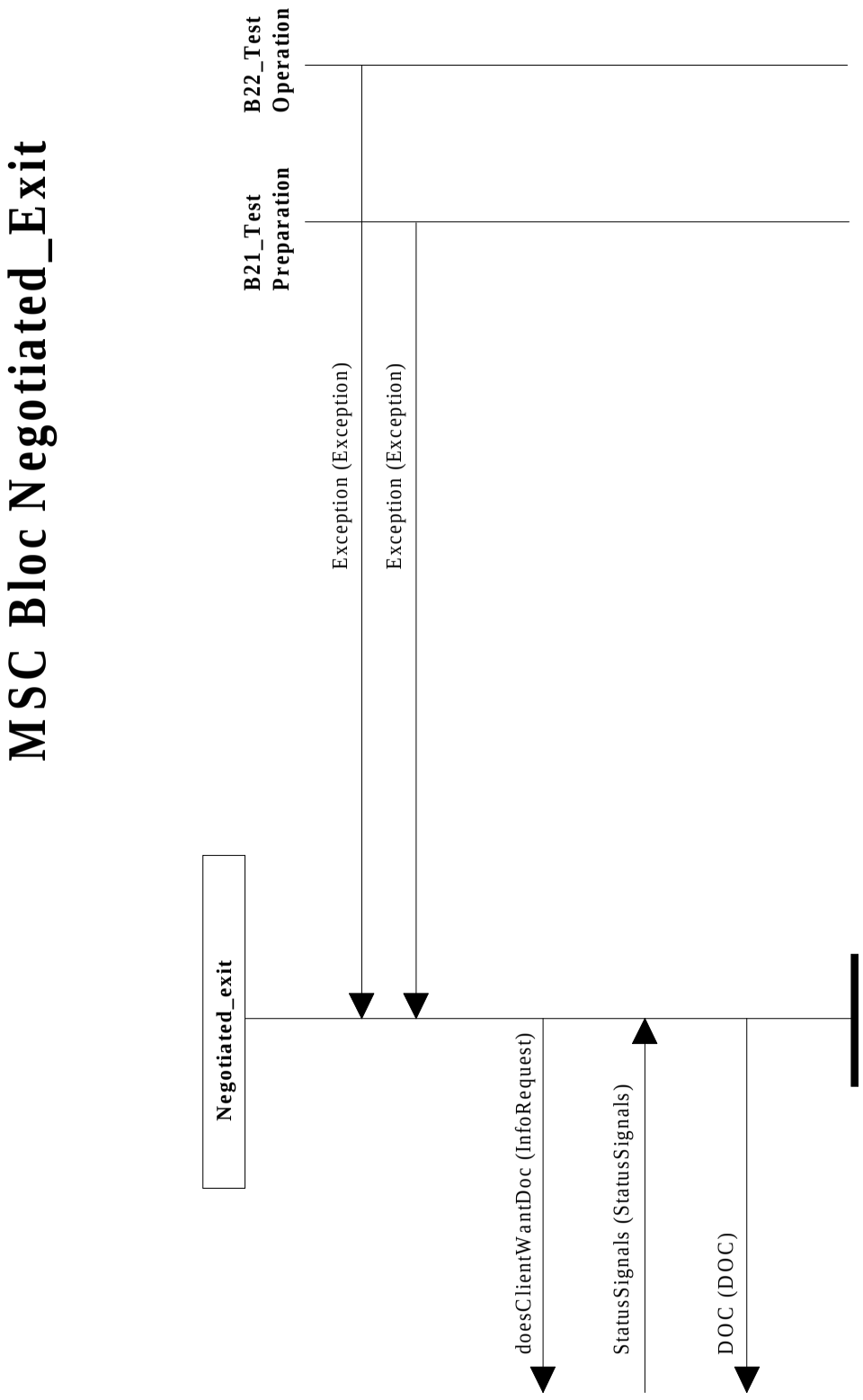


Figure A.4: MSC for Block - negotiated exit

Annex B (informative): Tabular form of Test Tool Support Statement Proforma (TTSS_PF)

B.1 Introduction

This annex provides a TTSS proforma which can be used by test tool developer organisations to document their interface support for the ISO/IEC 9646 [1] - [6] Conformance Assessment Process.

Comments for guidance purposes only are shown in **bold underlined text**, and should not be included in any actual TTSS.

The name of the organisation completing the proforma, the TTSS reference number, the page number and total number of pages should appear on every page of the TTSS.

B.2 TTSS Proforma

Table B.1 shows the tabular form of the TTSS Proforma.

Table B.1: Tabular form of the TTSS Proforma

reference specification (NOTE) **unique for tool supplier** **page number** **page count**	Ref Specification : TTSS Ref no : Page : No. of pages :
TEST TOOL SUPPORT STATEMENT FOR:	**Test Tool Supplier**
1. IDENTIFICATION SUMMARY	
TTSS Number:	**unique for test tool supplier**
TTSS Supplier:	**organisation providing TTSS**
TTSS Date:	**date proforma filled in**
Authority:	**person responsible**
n. TEST TOOL DESCRIPTION	**for each test tool**
n.1 TEST TOOL:	**tool name**
Release:	**release of test tool**
Version:	**version of test tool**
Status:	**status of test tool**
Date:	**date of release of test tool**
n.2 TOOL SUPPORT for CAP	**for each CAP process**
n.2.n SUPPORT FOR:	**CAP process identifier**
CAP process support:	**Yes/Partial/No/NA**
Comments (optional):	**any comments concerning support**
n.3 TOOL ACCESS POINTS	**for each conformance data object**
n.3.n SUPPORT FOR:	**conformance data object name**
Applicable Process:	**CAP process name**
Support:	**Yes/No/NA**
Interchange capability:	**Export/Import/Both**
Communication Description:	**description of exchanging medium**
Encoding Description:	**description of encoding**

NOTE: Identification of the specification which defines this proforma and its contents. This is the reference number of the ETR/ETG, possibly complemented with any of its future extensions.

History

Document history	
November 1993	First Edition
February 1996	Converted into Adobe Acrobat Portable Document Format (PDF)