Conformance and Interoperability Testing Tutorial

ITU-T SG 17

Edited by O. Monkewich
Vice-Chairman SG 17
Chairman WP 3/17
Chairman JCA-CIT

Acknowledgements

In preparing this Tutorial, SG 17 recognizes the special contributions of the following experts and their organizations

- Ostap Monkewich - Industry Canada and OMCI Consultants
- Denis Andreev - Central Research Telecommunication Institute (ZNIIS), Russian Federation, Q.8/11 Q.3900 editor
- Ultan Mulligan - The European Telecommunications Standards Institute (ETSI)
- Stephen Randall - The European Telecommunications Standards Institute (ETSI) and PQM Consultants
- Ina Schieferdecker - Fraunhofer Institute for Open Systems (FOCUS)

CIT Objectives

- **Interoperability of products from different suppliers**
- **Test the product only once**
- **Acceptance of test results in different geographical regions**
- **Meet regulatory or market driven requirements**

How to achieve them

- Recommendations error-free and unambiguous
- All use the same terms and definitions
- One testing methodology
- Requirements and Options listed in point form
- Questionnaire on what was implemented, what was not
- One set of test purposes
- One test suite per standard or Recommendation
- Same test verdicts for same tests
- Conformance before Interoperability
- Accredited test facilities
- Certified products

Conformance Testing Methodology Recommendations

- X.290 - General Concepts
- X.291 - Abstract Test Suite Specification
- X.292 - (Superceded by Z.140 series Recommendations)
- X.293 - Test Realization
- X.294 - Requirements on Test Laboratories and Clients
- X.295 - Protocol Profile Test Specification
- X.296 - Implementation Conformance Statements
- Z.140 through Z.146 - Testing and Test Control Notation

Why Do We Need A Common Testing Methodology

- **All actors must understand each other in all geographical regions and global markets**
  - Equipment suppliers
  - Equipment buyers
  - Test laboratories
  - Accreditation organizations
  - Certification organizations
- **Test results must have the same meaning in all global regions**
- **Test results must be accepted in all global regions**
- **Time to market - equipment must be tested only once without the need to retest for different markets**

Why Conformance to Standards is important?

- Equipment from different vendors conforming to the same standards have a higher likelihood of interoperability
- Different vendors can independently implement standards with higher assurance of product interoperability
- Equipment buyers can buy products that will interoperate with previously purchased equipment form different supplier

Why Interoperability is important?

- The ultimate objective is that independent implementations of the same standard interoperate
- Conformance improves the chances of interoperability while interoperability testing checks at a user level if interoperability has been achieved

Conformance and Interoperability are Complementary

- Conformance to the standard is achieved first and should not be compromised during Interoperability testing
- Without conformance, two implementations can be made to interoperate by destroying interoperation with all other systems

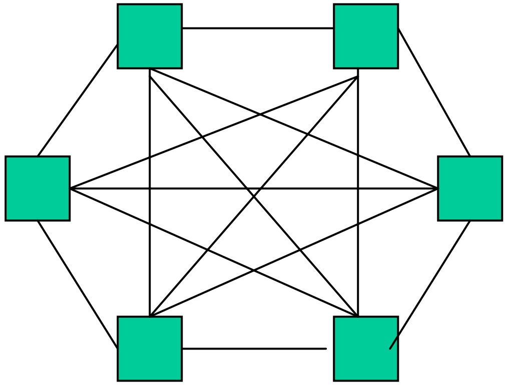
Causes of Interoperability Problems

- Standards
  - Errors and ambiguities in standards
  - Incompatible standards (standards with different QoS, traffic priorities)
- Implementations
  - Human errors, e.g. programmer errors
  - Different interpretations of the standard
  - Different choice of options allowed by the standard
- Technology
  - networks use different traffic queuing techniques
  - device compatibility
  - host system configuration

Nature of Interoperability Testing

- **Interoperability testing is only meaningful in single-pair combinations of products**
- **N interconnected products present $(N^2 - N)/2$ distinct product pair combinations**

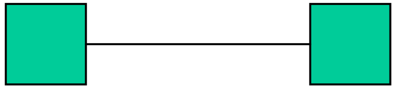
- Example:



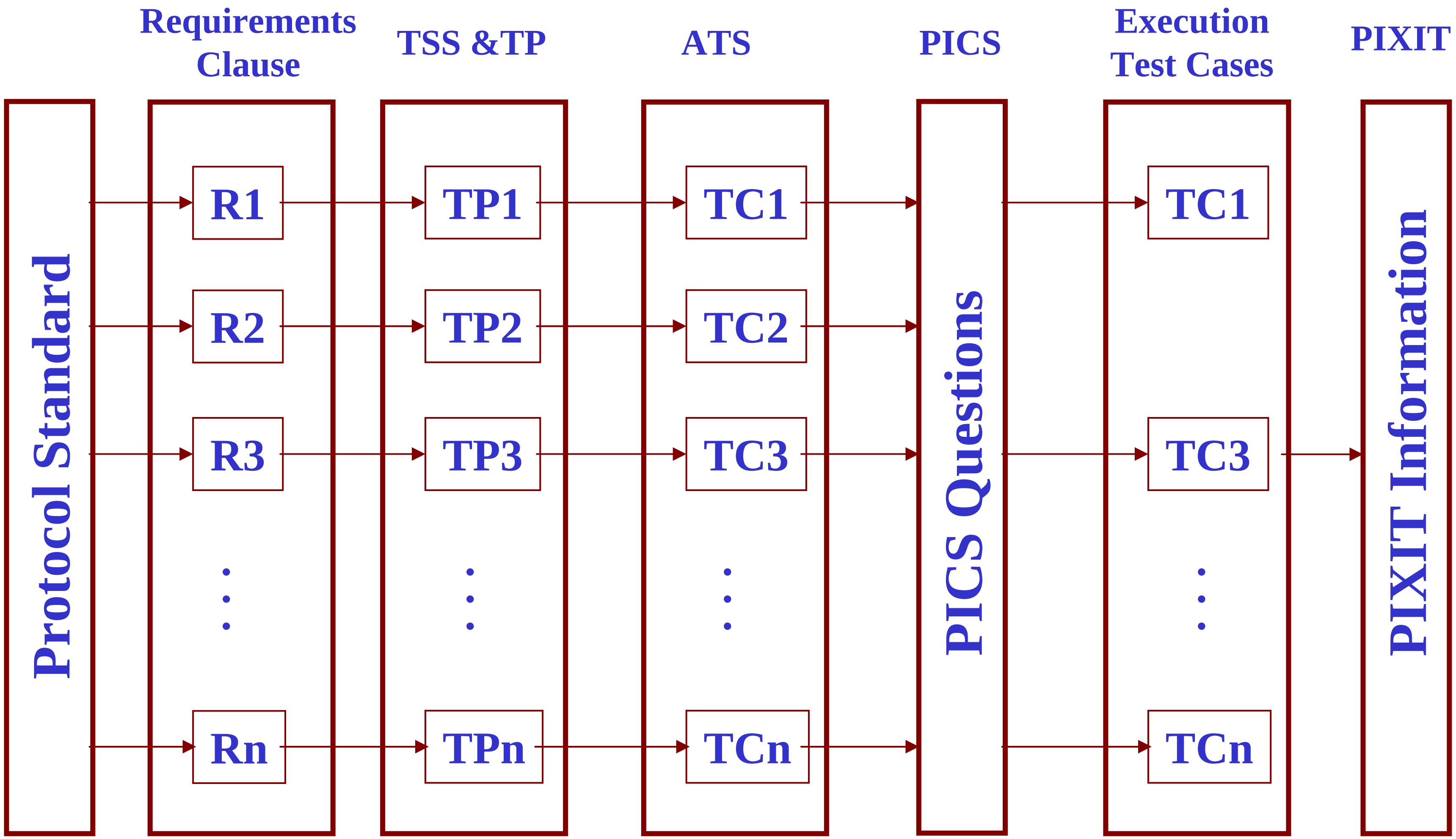
- **N = 6 products or 15 pairs**
- *Each product is tested 15 times*
- **N = 100 or ~5000 pairs**
- *Each product is tested 5000 times*

Nature of Conformance Testing

- **Testing to determine if the product does what the Recommendation says it is supposed to do**
- **Each product is tested only once, against the standard (represented by the test suite)**



Standards that Facilitate Testing

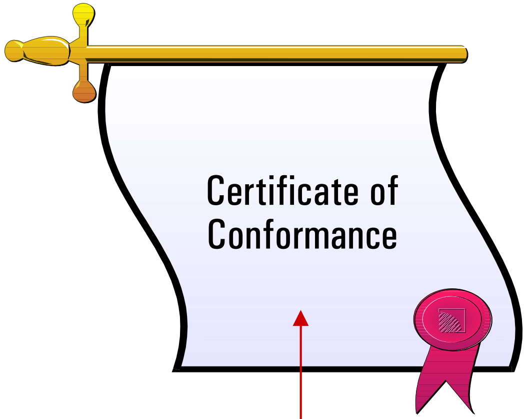


TSS & TP	-	Test Suite Structure and Test Purposes
ATS:	-	Abstract Test Suite
PICS:	-	Protocol Implementation Conformance Statement

TP	-	Test Purpose
TC	-	Test Case
R	-	Requirement

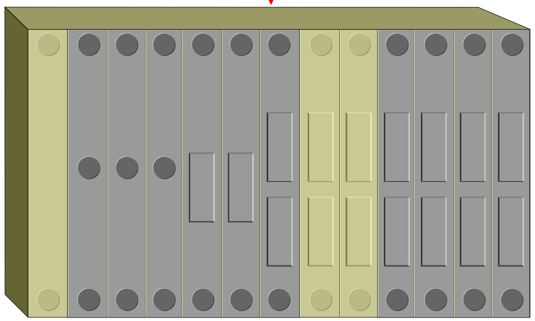
Overview of Conformance Testing

1. Static Review
2. Dynamic Tests
3. Test Report
4. *Certification*



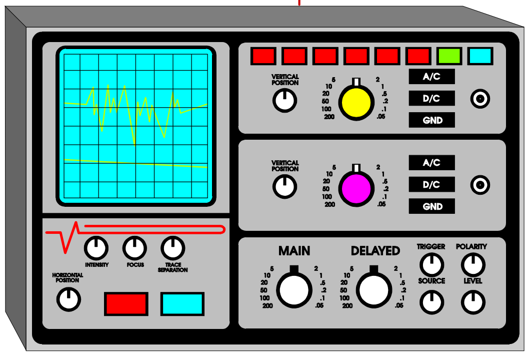
Test Report

Implementation with a formal declaration of which parts of the standard were implemented



STIMULUS

RESPONSE



Implementation Under Test (IUT)

Test Equipment and Test Suite

Static vs Dynamic Testing

- Static (PICS review)
  - what mandatory, optional or conditional features of the protocol were implemented - declared in the PICS proforma by the supplier
  - PICS becomes a shopping list for finding compatible products
  - a car PICS analogy:
    - does the car have an ignition system?
    - does the car have a steering wheel?
- Dynamic (execution of the ATS)
  - behaviour of mandatory, optional or conditional features
  - a car ATS analogy
    - turn the ignition key, does the engine turn over? Pass, Fail, Inconclusive
    - turn the steering wheel, do the front wheels turn? Pass, Fail, Inconclusive

A Requirements Clause from a Standard

Extract from the User-Network Interface (UNI) Specification 3.1

3.3 ATM Cell Structure and Encoding at the UNI

(R) CPE at the UNI shall encode the GFC value to all zeros (0000).

(R) Public network equipment at the public UNI shall encode the GFC value to all zeros (0000).

(O) CPE shall inform Layer Management if a count of the non-zero GFC fields measured for non-overlapping intervals of 30,000 +/- 10,000 cell times reached ten (10) or more.

(O) Public network equipment shall inform Layer Management if a count of non-zero GFC fields measured for non-overlapping intervals of 30,000 +/- 10,000 cell times reaches ten (10) or more.

Protocol Implementation Conformance Statement (PICS) Proforma

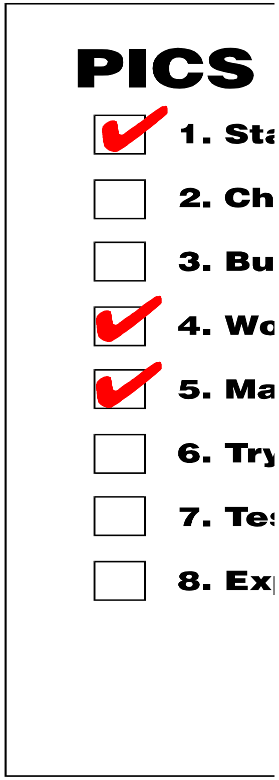
Extracted from af-test-0059.000: PICS Proforma for the UNI 3.1 ATM Layer

3.5 Generic Flow Control (GFC) Field

Index	Text	Status	Ref.	Values	Support
3.5.1	Does the IUT operate the GFC protocol in "uncontrolled access" mode, encoding the GFC field to be all zeros?	M	3.3		__Yes__No
3.5.2	If the IUT is an intermediate node, does the IUT overwrite any non-zero GFC field received before sending it into the network?	M	3.3		__Yes__No
3.5.3	Does the IUT, on receipt of 10 or more non-zero GFC fields measured for non-overlapping intervals over 30000+/-10000 cell times, generate an error to layer management?	O	3.3		__Yes__No

Static Review

- ICS Proforma - Implementation Conformance Statement Proforma
  - formatted questionnaire for declaring what optional features have been implemented
  - part of the specification or standard
- ICS
  - Filled-out ICS Proforma
  - A list of requirements and options claimed to have been implemented
- Used for
  - Shopping list for matching products for interoperability
  - Test case selection (from test suite) for execution



PICS

☑ 1. Sta

□ 2. Ch

□ 3. Bu

☑ 4. Wo

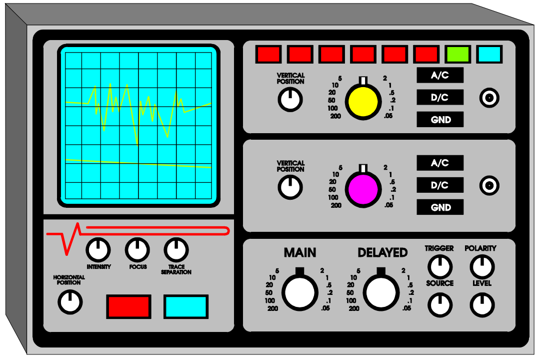
☑ 5. Ma

□ 6. Try

□ 7. Tes

□ 8. Exp

Dynamic Tests

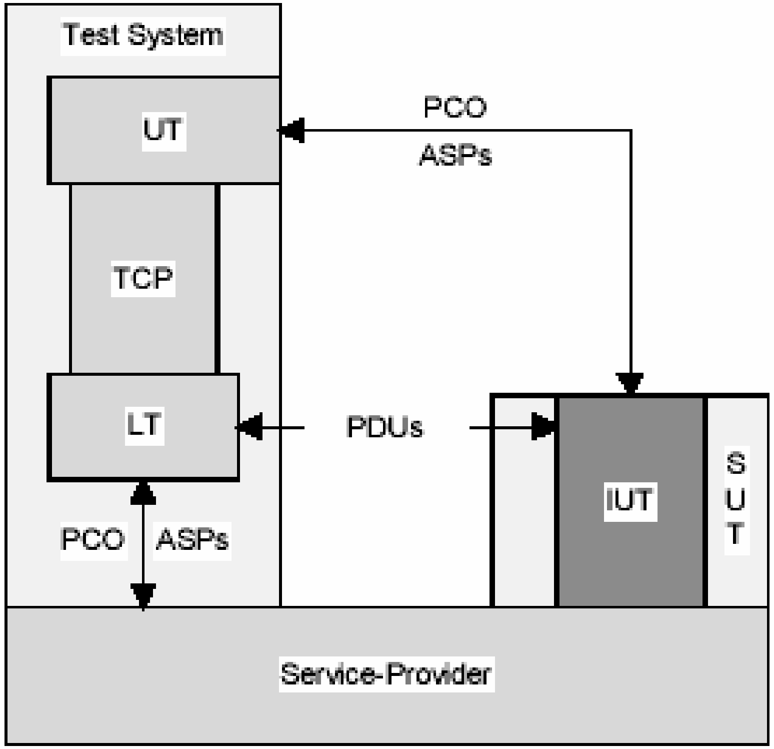


- Abstract Test Suite (ATS)
  - – Defined by a standards organization, written in an abstract language like
    - Testing and Test Control Notation (TTCN-3), ITU-T Rec. Z.140
- Executable Test Suite (ETS)
  - – AT .mp file “compiled” to run on specific test equipment
  - – creation of the ETS is proprietary to the test equipment vendor

The Local Test Method

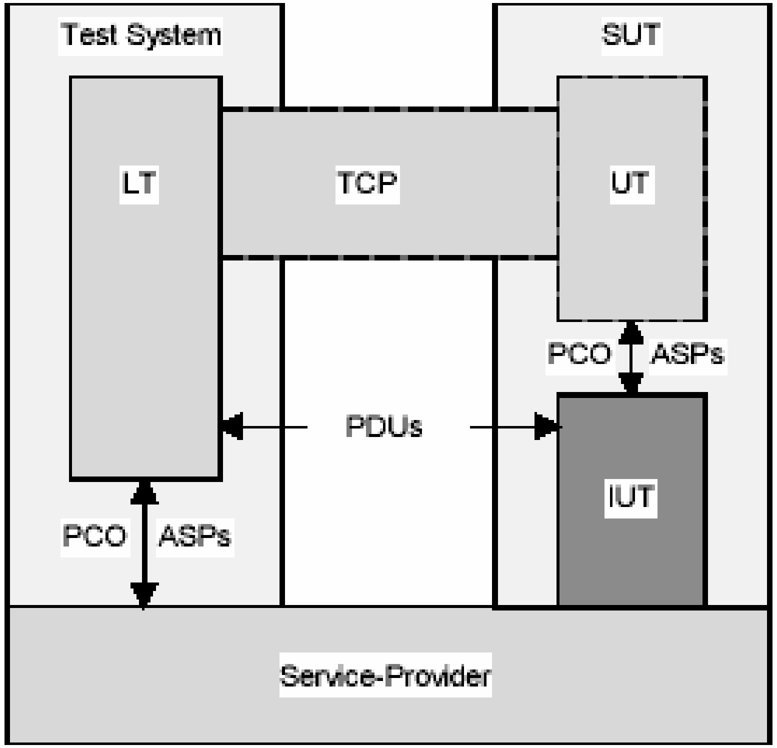
There are two PCOs. UT and LT both reside on the Test System. The upper boundary of the IUT is standardized hardware interface that plugs into the Test System.

UT Upper Tester
LT Lower Tester
PCO Point of Control and Observation
IUT Implementation Under Test
SUT System Under Test
ASP Abstract Service Primitive
PDU Protocol Data Unit
TCP Test Coordination Procedure



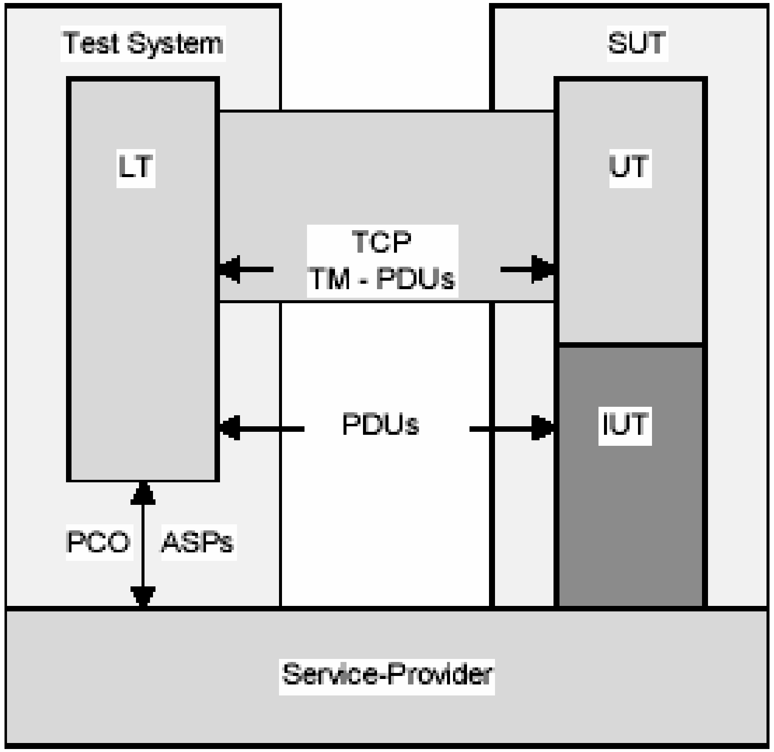
The Distributed Test Method

There are two PCOs. The UT is located in the SUT. The LT is located in the Test System. Access to the upper boundary of the IUT is required to carry out testing either by human action or a programming interface.



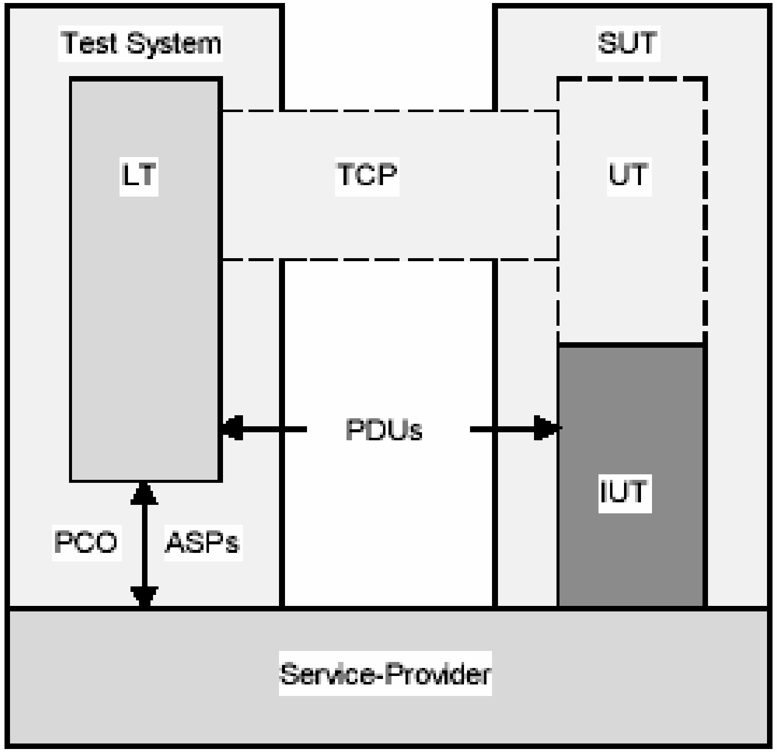
The Coordinated Test Method

There is only one PCO and no UT. UT is integrated with TCP. The desired effects at the upper boundary of the IUT are realized by a special TCP called the standardized Test Management protocol. The method facilitates the highest degree of automation and security.



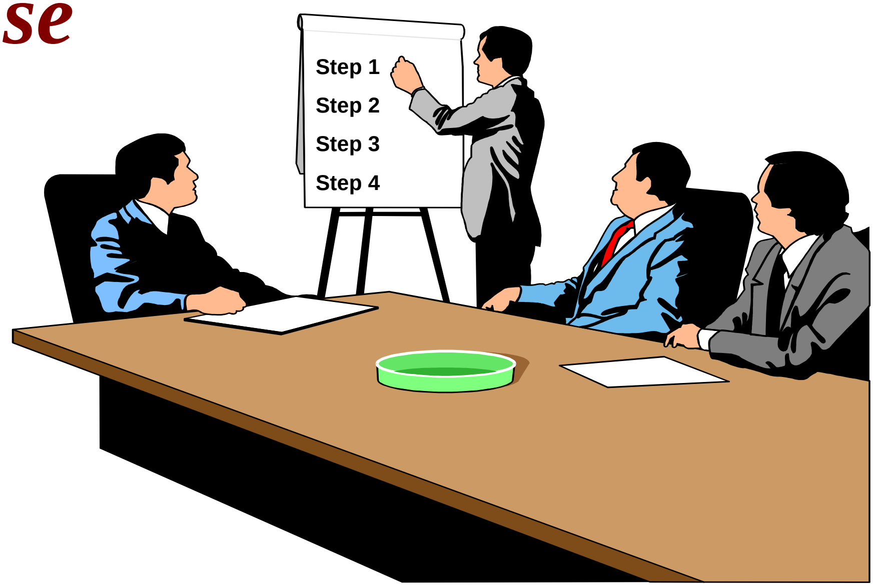
The Remote Test Method

There is only one PCO and no UT or TCP. The Tester has no access to the upper boundary of the IUT. The desired effects at the upper boundary are informally described in the test suite and are carried out at the SUT by the test operator



What is a Test Suite?

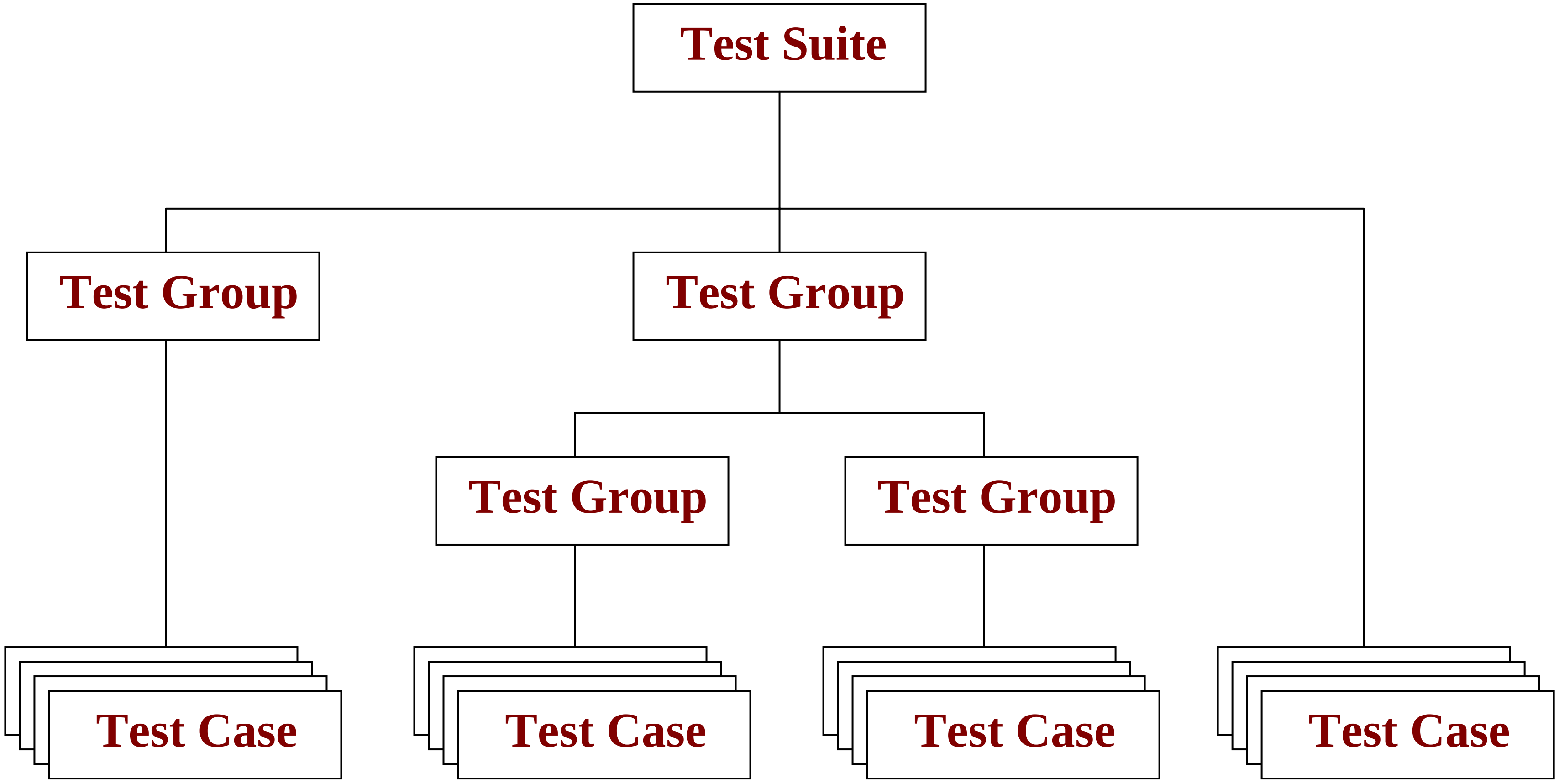
- A *test suite* is a collection of *test cases,* one for each test purpose, specified in accordance to the test method used
- A test case verifies *conformance/interoperability* for a particular Requirement or Option according to the *test purpose*



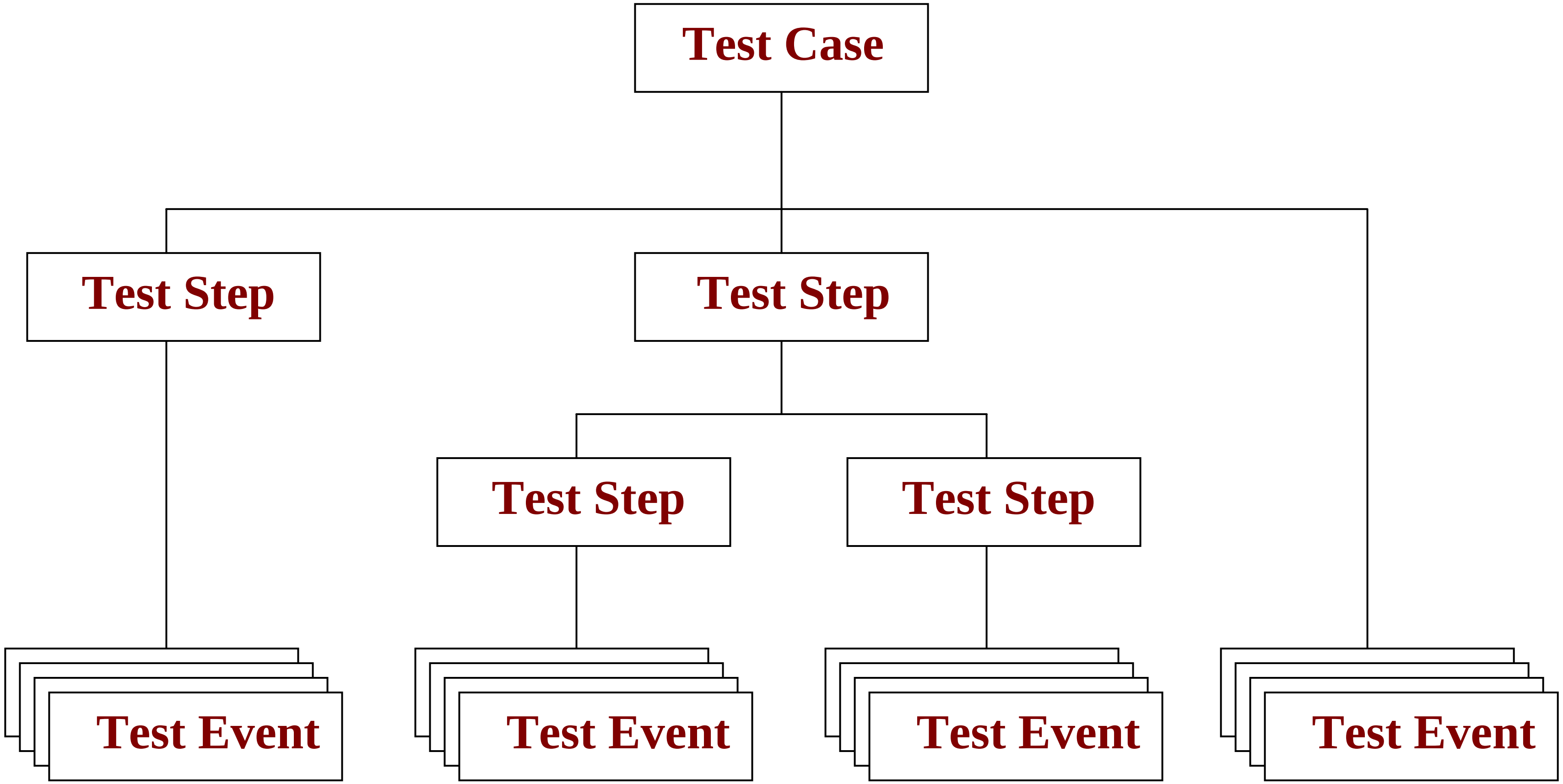
Test Suite Development

- **Start with a *PICS***
  - This ensures that complete coverage is obtained
- **Develop *Test Suite Structure***
  - This logically groups the test cases
- **Develop *Test Purposes***
  - This defines the objectives of the *test cases*
- **Write a *Test Case* for each Test Purpose**
  - The *test purpose* is then included with its test case in the test suite

Test Suite Structure



Test Case Structure



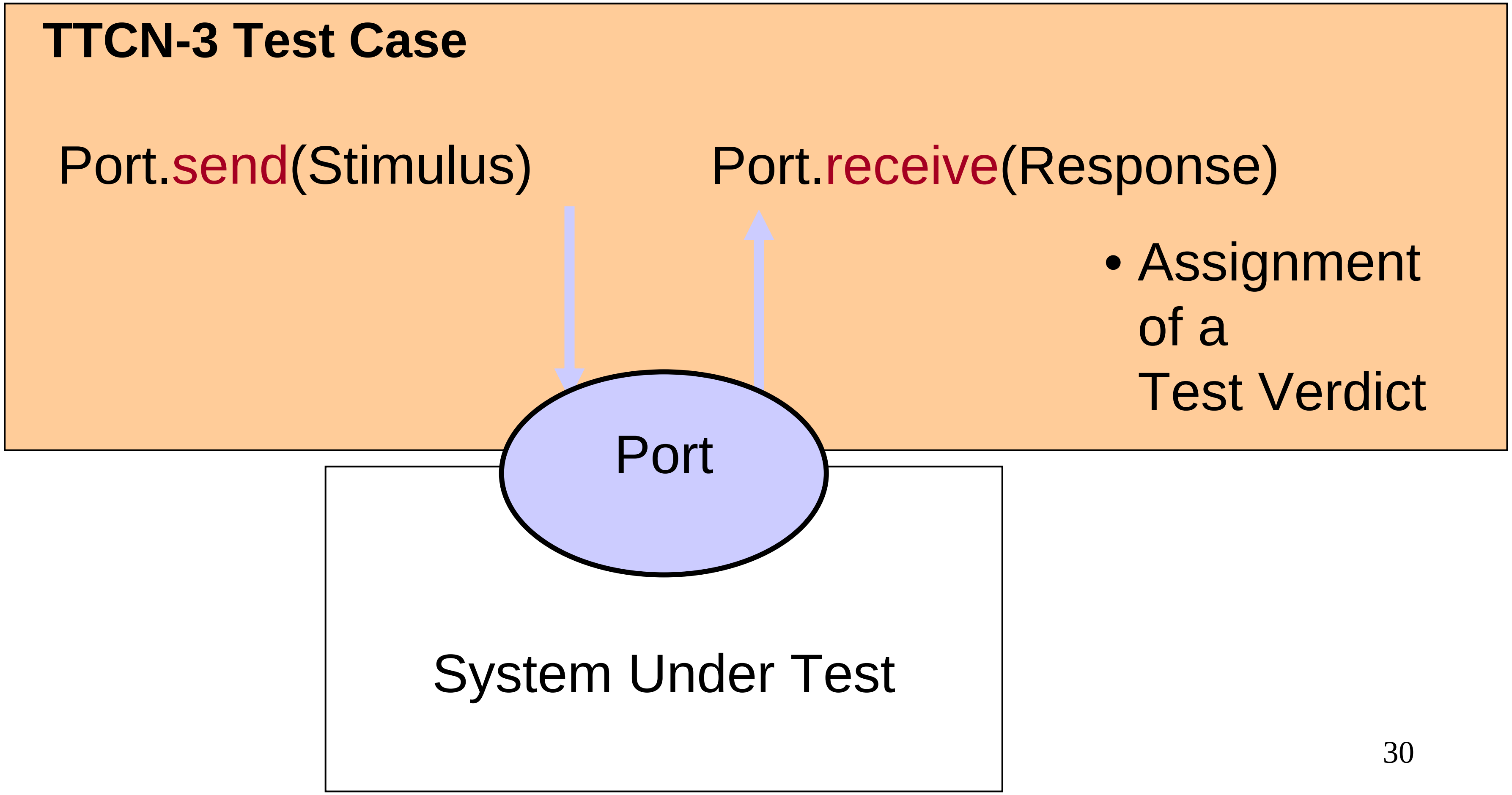
Extra Information for Testing

- IXIT - Implementation eXtra Information for Testing
  - Additional information required before testing can proceed
    - administrative: identification of client, laboratory staff, IUT, protocol, test suite
    - technical: address of the IUT, timer values, configuration, parameters, procedures, test cases that cannot be executed
- IXIT Proforma
  - Standardized template to be completed by the client and the test laboratory to produce the IXIT
- PIXIT - Protocol IXIT
  - A special case of IXIT, widely used

Test Cases, Test Purposes and Verdicts

- One Test Case for each Test Purpose representing one Requirement from the Requirements Clause
- To get a **Pass** verdict, the Implementation Under Test (IUT) must respond correctly when the Tester exhibits three different kinds of behaviour:
  - Valid
  - Invalid
  - Inopportune
- For each of the three Tester behaviours, the IUT may be assigned a **Pass**, **Fail** or **Inconclusive** verdict

Test Case Architecture in TTCN-3



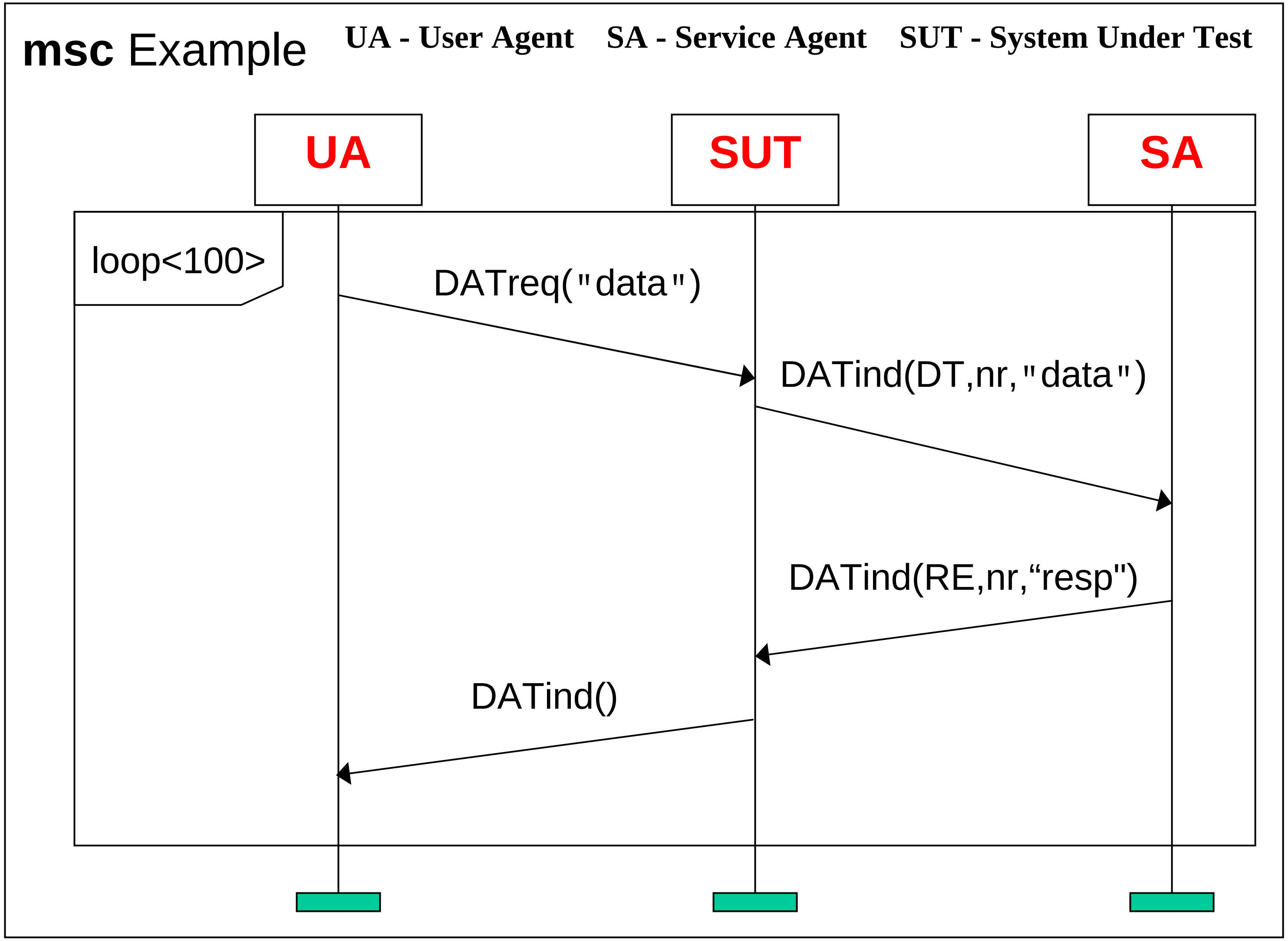
TTCN-3 Test Case

Port.send(Stimulus)

Port.receive(Response)

- Assignment of a Test Verdict

Sequence Diagram for a Simple Behaviour Example

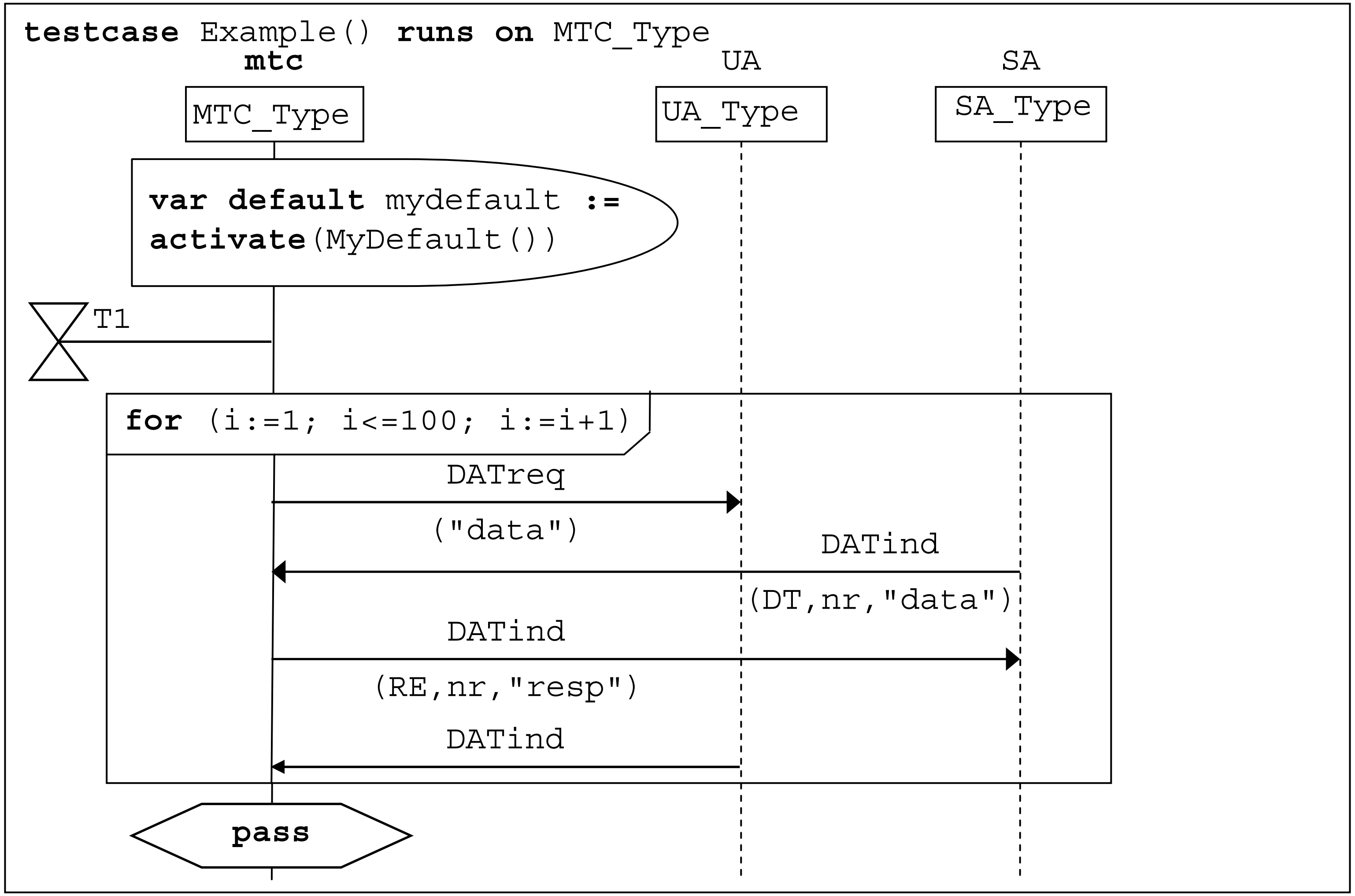


Same Example in TTCN-3 Core Language

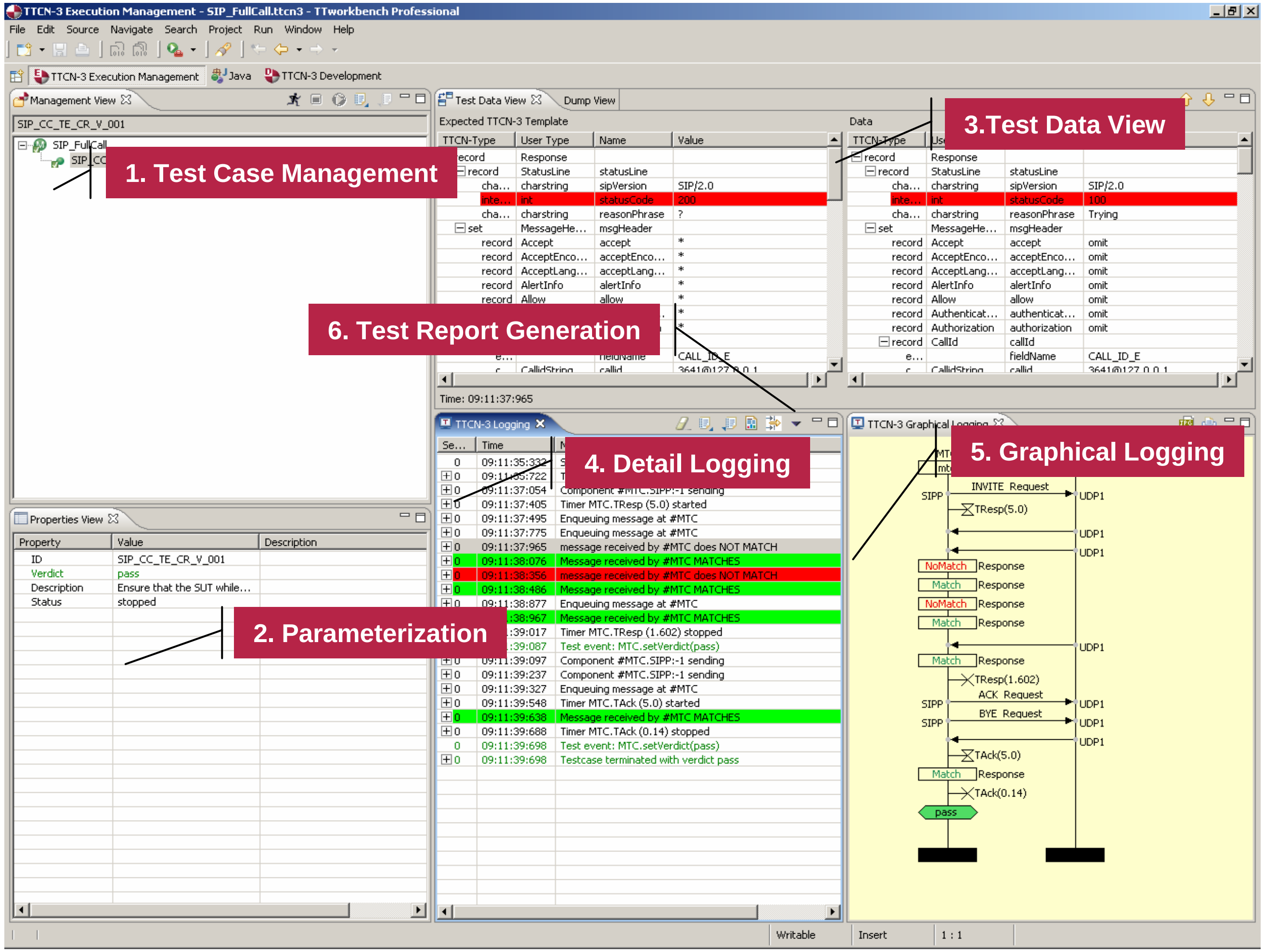
```
testcase Example( )              runs on MTC_Type {

        var default mydefault := activate (DefaultDef());
        T1.start;
        for (integer i:=1; i<=100; i:=i+1) {
                UA.send(DATreq:{"data"});
                SA.receive(DATind:{DT, nr, "data"});
                SA.send(DATind:{RE,nr, “resp”});
                UA.receive(DATind :{});
        }
        setverdict(pass);
        T1.stop;}
```

Same Example in Graphical Format of TTCN-3


testcase Example() **runs on** MTC_Type
mtc
UA
SA
MTC_Type
UA_Type
SA_Type
var default mydefault **:=**
activate(MyDefault())
T1
for (i:=1; i<=100; i:=i+1)
DATreq
("data")
DATind
(DT,nr,"data")
DATind
(RE,nr,"resp")
DATind
pass

Commercial Tool Example for TTCN-3



1. Test Case Management

2. Parameterization

3.Test Data View

4. Detail Logging

5. Graphical Logging

6. Test Report Generation

Expected TTCN-3 Template

TTCN-Type	User Type	Name	Value
record	Response		
record	StatusLine	statusLine	
cha…	charstring	sipVersion	SIP/2.0
inte…	int	statusCode	200
cha…	charstring	reasonPhrase	?
set	MessageHe…	msgHeader	
record	Accept	accept	*
record	AcceptEnco…	acceptEnco…	*
record	AcceptLang…	acceptLang…	*
record	AlertInfo	alertInfo	*
record	Allow	allow	*
[illegible]	[illegible]	[illegible]	*
[illegible]	[illegible]	[illegible]	*
[illegible]	[illegible]	[illegible]	
e…		fieldName	CALL_ID_E
c…	CallidString	callid	3641@127.0.0.1

Time: 09:11:37:965

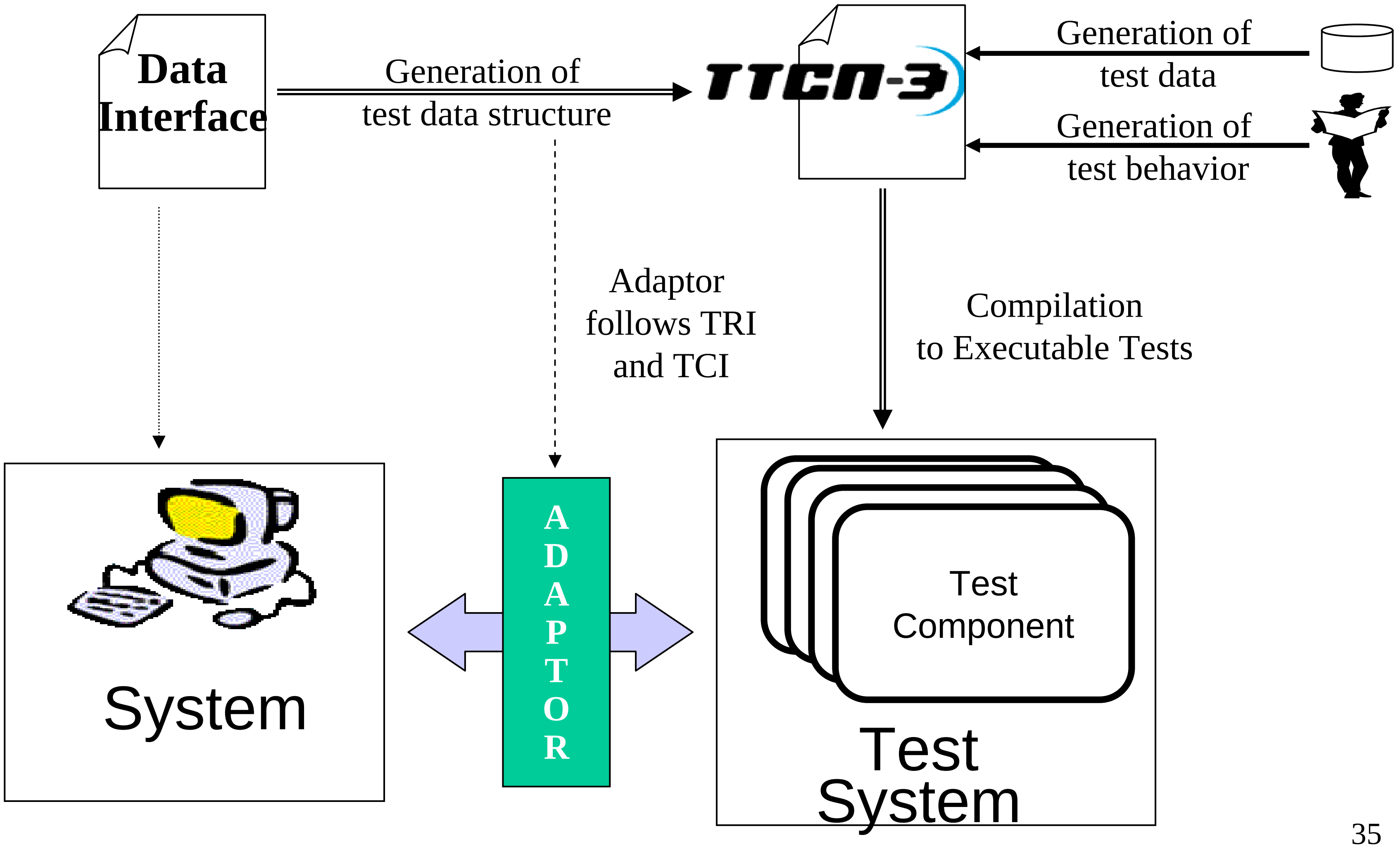
Data

TTCN-Type	User Type	Name	Value
record	Response		
record	StatusLine	statusLine	
cha…	charstring	sipVersion	SIP/2.0
inte…	int	statusCode	100
cha…	charstring	reasonPhrase	Trying
set	MessageHe…	msgHeader	
record	Accept	accept	omit
record	AcceptEnco…	acceptEnco…	omit
record	AcceptLang…	acceptLang…	omit
record	AlertInfo	alertInfo	omit
record	Allow	allow	omit
record	Authenticat…	authenticat…	omit
record	Authorization	authorization	omit
record	CallId	callId	
e…		fieldName	CALL_ID_E
c…	CallidString	callid	3641@127.0.0.1

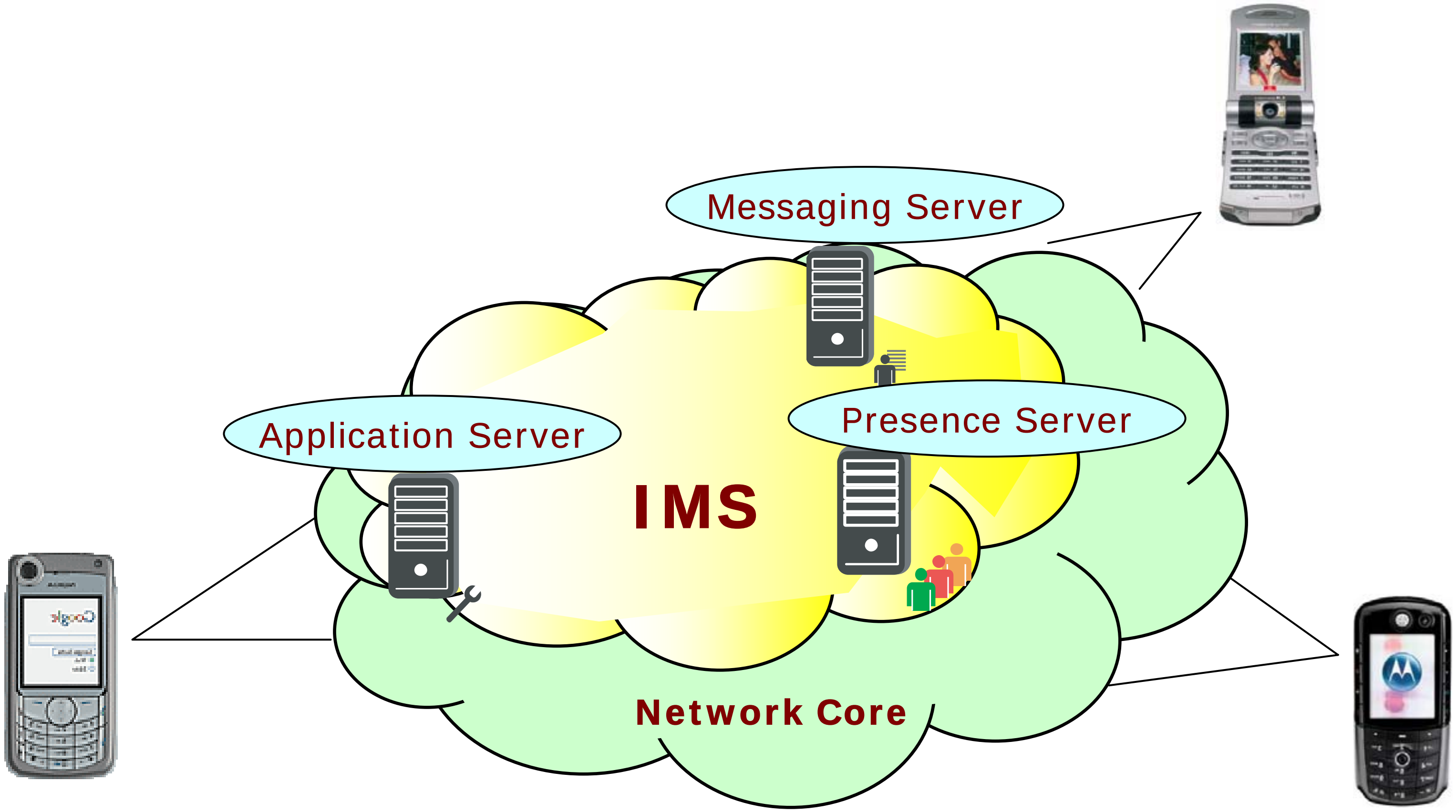
```
Se…  Time          M…
0    09:11:35:332  S…
0    09:11:35:722  T…
0    09:11:37:054  Component #MTC.SIPP:-1 sending
0    09:11:37:405  Timer MTC.TResp (5.0) started
0    09:11:37:495  Enqueuing message at #MTC
0    09:11:37:775  Enqueuing message at #MTC
0    09:11:37:965  message received by #MTC does NOT MATCH
0    09:11:38:076  Message received by #MTC MATCHES
0    09:11:38:356  message received by #MTC does NOT MATCH
0    09:11:38:486  Message received by #MTC MATCHES
0    09:11:38:877  Enqueuing message at #MTC
     :38:967       Message received by #MTC MATCHES
     :39:017       Timer MTC.TResp (1.602) stopped
     :39:087       Test event: MTC.setVerdict(pass)
0    09:11:39:097  Component #MTC.SIPP:-1 sending
0    09:11:39:237  Component #MTC.SIPP:-1 sending
0    09:11:39:327  Enqueuing message at #MTC
0    09:11:39:548  Timer MTC.TAck (5.0) started
0    09:11:39:638  Message received by #MTC MATCHES
0    09:11:39:688  Timer MTC.TAck (0.14) stopped
0    09:11:39:698  Test event: MTC.setVerdict(pass)
0    09:11:39:698  Testcase terminated with verdict pass
```

Property	Value	Description
ID	SIP_CC_TE_CR_V_001	
Verdict	pass	
Description	Ensure that the SUT while…	
Status	stopped	

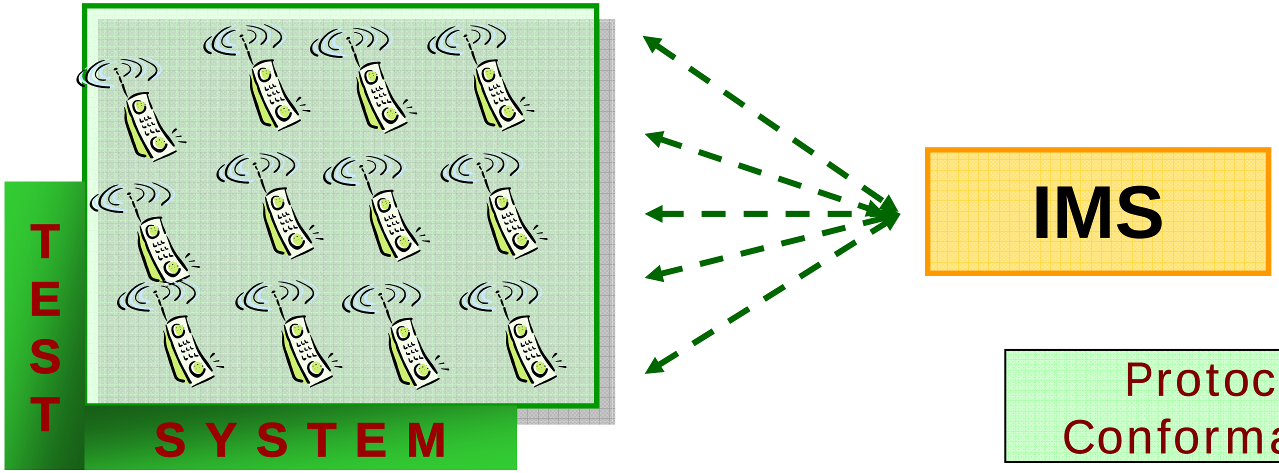
Overall Picture



IP Multimedia Subsystem (IMS)



IMS Testing with TTCN-3



Protocol Conformance

↓

Protocol Interoperability

↓

Service Conformance and Performance

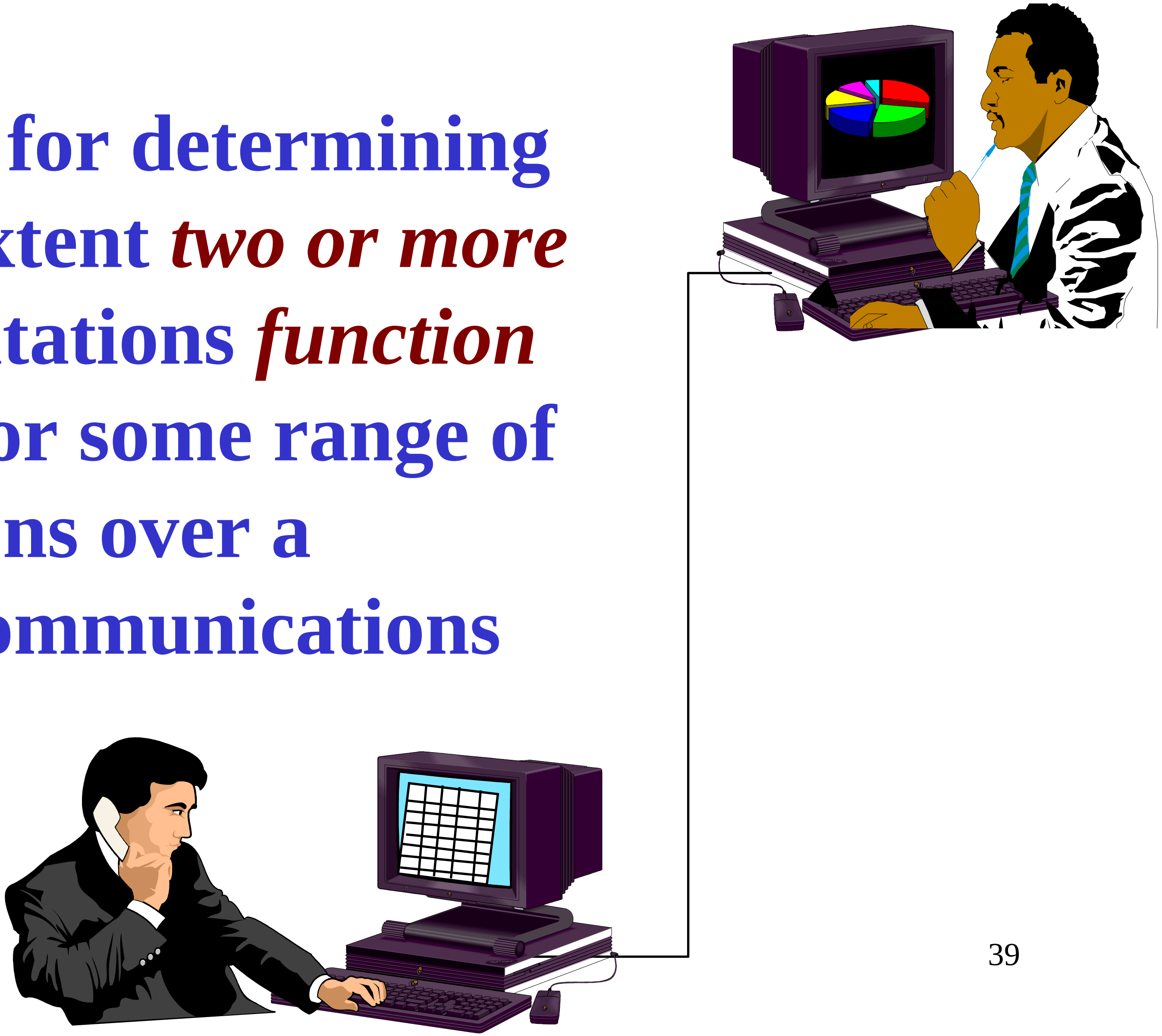
- **Benchmarking**
  - for comparison
- **Load/stress**
  - how system performs under load conditions
- **Capacity testing**
  - max load the system can handle before failing
- **Scalability testing**
  - to plan capacity improvements

What Standards are Missing Today

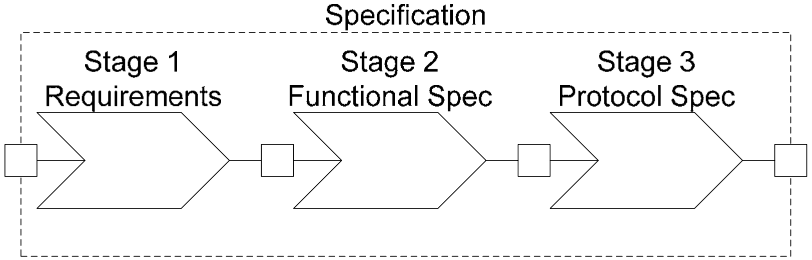
- For each Protocol
  - Requirements Clauses
  - TSS&TP
  - PICS and ICS Proformas
  - PIXIT and IXIT Proformas
  - Abstract Test Suites

Interoperability Testing

- A *method* for determining to what extent *two or more* implementations *function together* for some range of applications over a specific communications medium

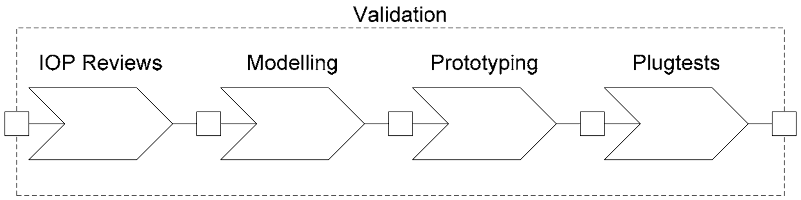


Specify for Interoperability



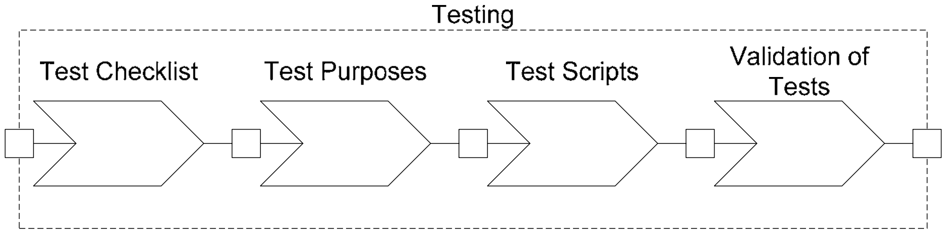
- **ITU-T I.130 3-stage model for protocol specification**
- **Requirements**
- **Functional Architecture and Information Flows**
  - – standardize interoperable interfaces, not internal behaviour
- **Detailed protocol specification**
  - – use most relevant techniques: text, UML, SDL, ASN.1, XML etc.

Validate for Interoperability



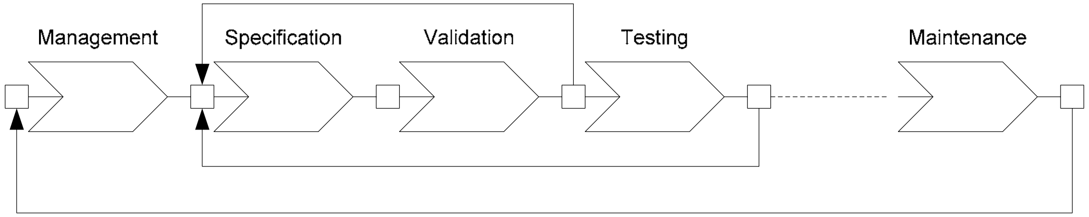
- Validation through technical reviews and simulation
- Validation through interoperability events
- Validation through test specification development

Test for Interoperability



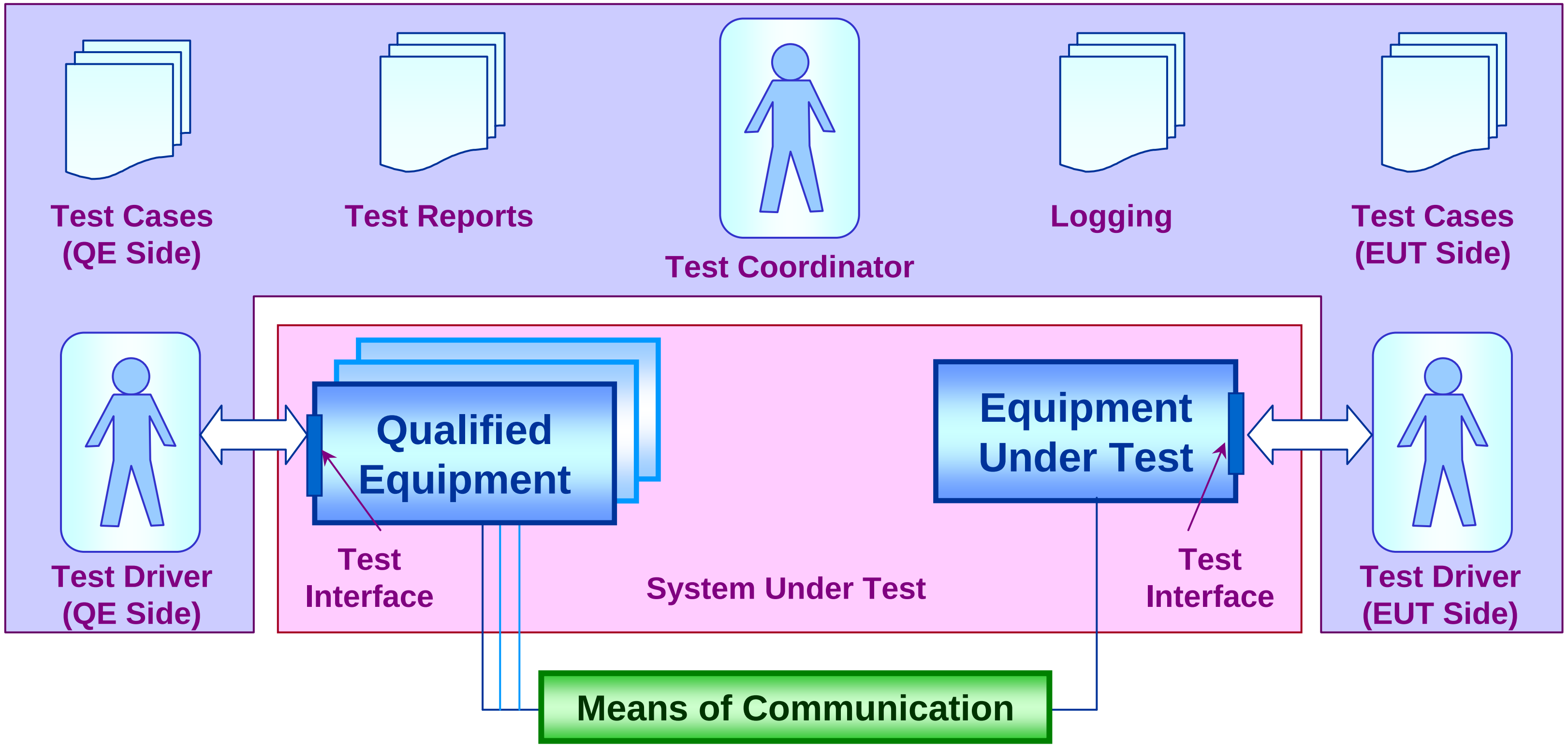
- **Plan for validation and Plan for testing**
- **Conformance Testing and Interoperability Testing**
- **Use existing methodologies**
  - X.290 Recommendations, TTCN-3, ISO/IEC 9646, ETSI Interoperability Testing Methodology
- **Validate test specifications**

Maintain for Interoperability



- Good Recommendations can be broken by poor maintenance or no maintenance
- Corrections should be made with care
- Extensions require same process as original development
- Feedback should be sought and captured

Basic Interoperability Concepts



Interoperability Test Specification

- Specification process steps
  - Specify abstract architecture
  - Write draft Interoperable Features Statement (IFS)
  - Specify Test Suite Structure (TSS)
  - Write Test Purposes (TP)
  - Write Test Cases (TC)
  - Validate Test Cases

Specify Abstract Architecture

- General framework(s) within which concrete test arrangements must fit
- Can be graphical, tabular or textual
- Should identify:
  - EUT
  - QE(s)
  - Paths between EUT and QE(s) (MoC)
  - Valid equipment types for EUT and QE(s)
  - Expected protocols to be used

Write Draft Interoperable Functions Statement

- An IFS Identifies:
  - Functions that an EUT must support
  - Functions that are optional
  - Functions which are conditional
- The IFS provides structure to the test specification
- It can also be used like a PICS as a proforma for a manufacturer to declare which functions are supported in an EUT

Specify Test Suite Structure

Identify test groups based upon, e.g.:

- Abstract Architecture
- Functionality
- Behaviour:
  - Normal
  - Exceptional

- **Define test coverage for each group**
  - What range of tests is to be included in each test group

Write Test Purposes

- For each possible test case, describe WHAT is to be tested
- Use the most appropriate means of expressing Test Purposes:
  - Plain language
  - Tables
  - MSCs
  - A specialist notation such as TPLan which offers:
    - Consistency in TP descriptions
    - Clear identification of preconditions, test actions and verdict criteria
    - Checkable syntax

Write Test Cases

- Test cases should include:
  - Preconditions
    - Configuration
    - Initial status
  - Test steps
    - Detailed instructions to Test Driver
      - Clear
      - Precise
      - No unnecessary restrictions
  - Verdicts
    - “Pass” means “EUT Pass”!
    - “Fail” may not mean” EUT Failure”
      - QE failure
      - MoC failure
      - Requires investigation

Test Case Specification

- **Tabulated free text**
  - Ideal for implementation by human Test Drivers
  - Individual test steps and their relation to each other is easy to understand
  - Only supports simple, serial test path, .i.e, very difficult to describe alternate paths following an unsuccessful intermediate verdict
- **Test language (TTCN-3)**
  - Ideal for machine implementation of Test Drivers
  - Highly repeatable
  - Allows comprehensive handling of unexpected behaviour
  - Difficult for the human user to read and follow
  - Establishing a testing environment is complex

Sample Tabular Specification

Identifier	TC_SS_0001_01		
Summary:	Supervised call transfer from User B to User A		

Test Purpose:	ensure that {
	when { A call is established between User_C and User_B }
	then { User_B can transfer the call from User_B to User_A
	after User_B and User_A communicate }
	}		

TP Identifier	TP_SS_0001	**Configuration:**	Test Architecture 2
Pre-test conditions:	• User A, User B and User C configured with Bearer Capability set to "Speech, 64 kbit/s" • User A configured to support the Call Transfer service		

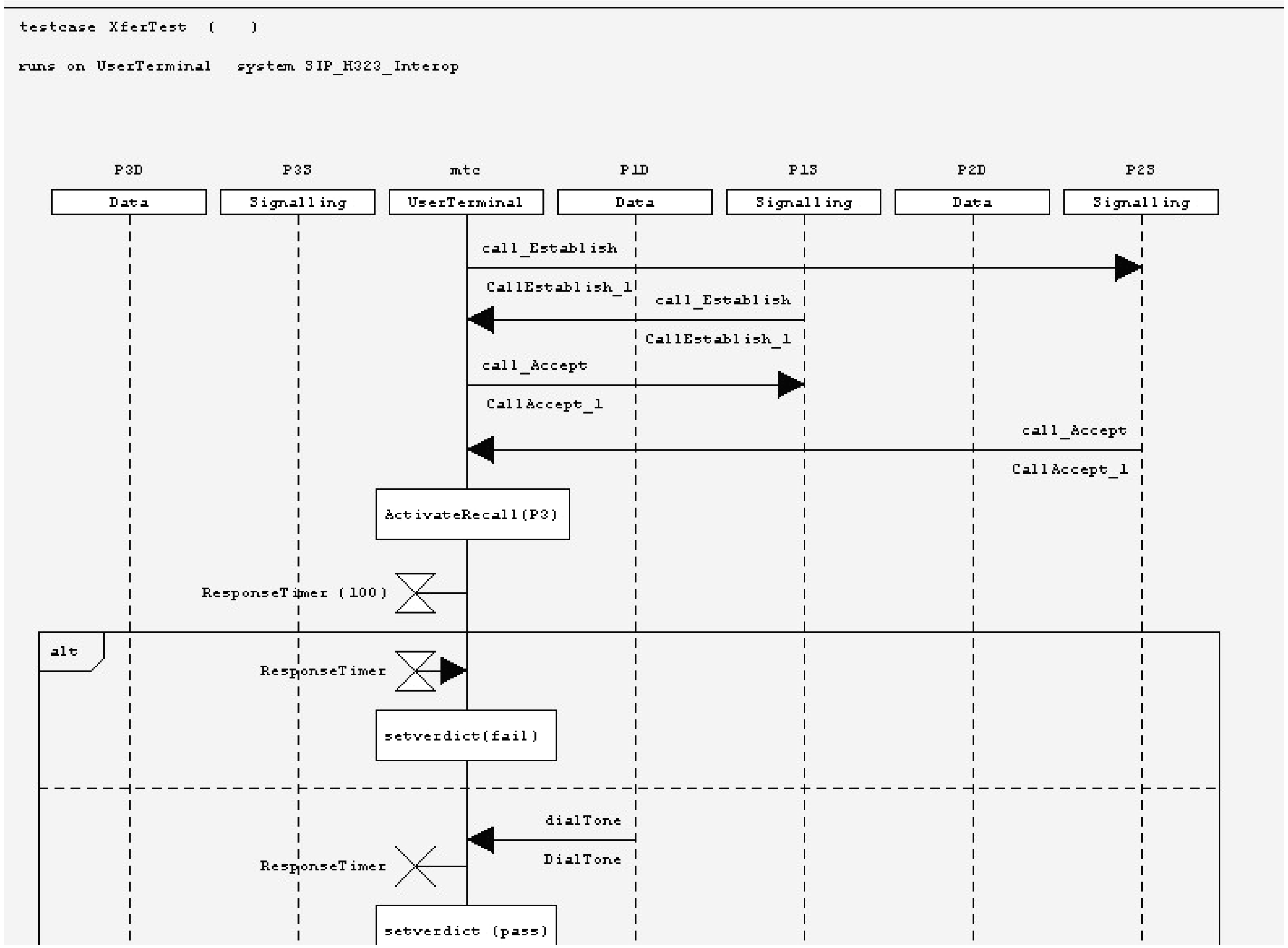
Step	**Test sequence**	**Verdict**	
		Pass	**Fail**
1	Initiate new call at User C to the address of User B		
2	Accept call at User B		
3	Activate the "recall" button (or equivalent) at User B's terminal		
4	*Is dial tone (or an equivalent indication) present at User B's terminal?*	Yes	No
5	Initiate a new call from User B to the address of User A		
6	*Is User A's terminal alerting (visual or audible indication)?*	Yes	No
7	Accept call at User A		
8	Apply speech at User A		
9	*Can speech from User A be heard and understood at User B?*	Yes	No
10	*Can speech from User A be heard and understood at User C?*	No	Yes

Sample Specification in TTCN-3 Core Language

```
// Define Supervised Transfer test case
testcase SupervisedTransfer() runs on userTerminalType
{  timer ResponseTimer := 100E-3;

 // Preamble: Establish call between Users B & C
 m3s.send (CallEstablish_1);
 m2s.receive (CallEstablish_1);
 m2s.send (CallAccept_1);
 m3s.receive (CallAccept_1);
 // Register recall test
 m2s.send (Recall);
 ResponseTimer.start;
 alt
 { [] ResponseTimer.timeout
     { setverdict(fail);
      stop
     }
   [] m2d.receive (DialTone)
     { setverdict(pass);
      ResponseTimer.stop
      // Hold call test
. . . . . . .
```

Sample Specification in TTCN-3 GFT



testcase XferTest ( )

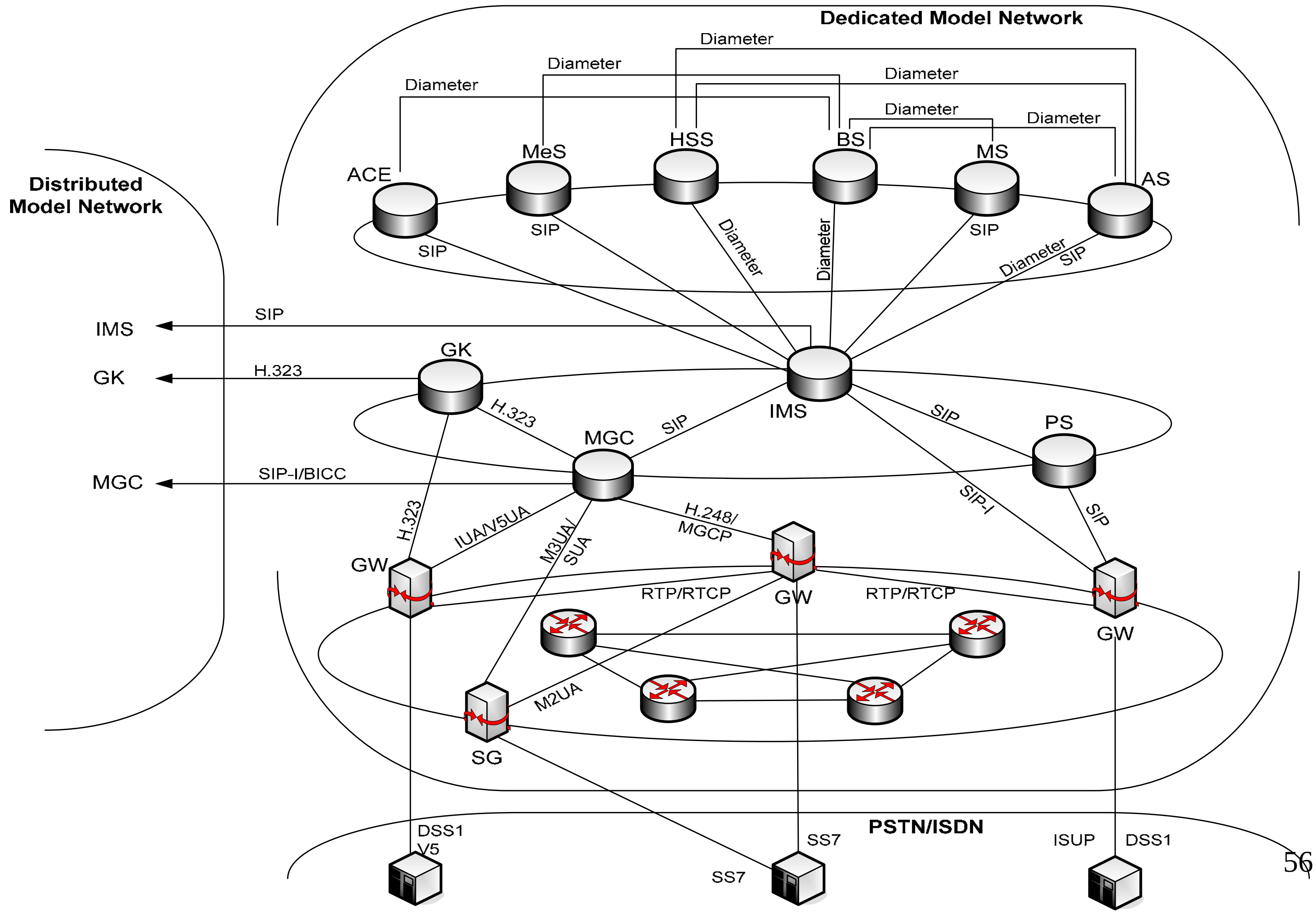
runs on UserTerminal system SIP_H323_Interop

Interoperability Testing Standards

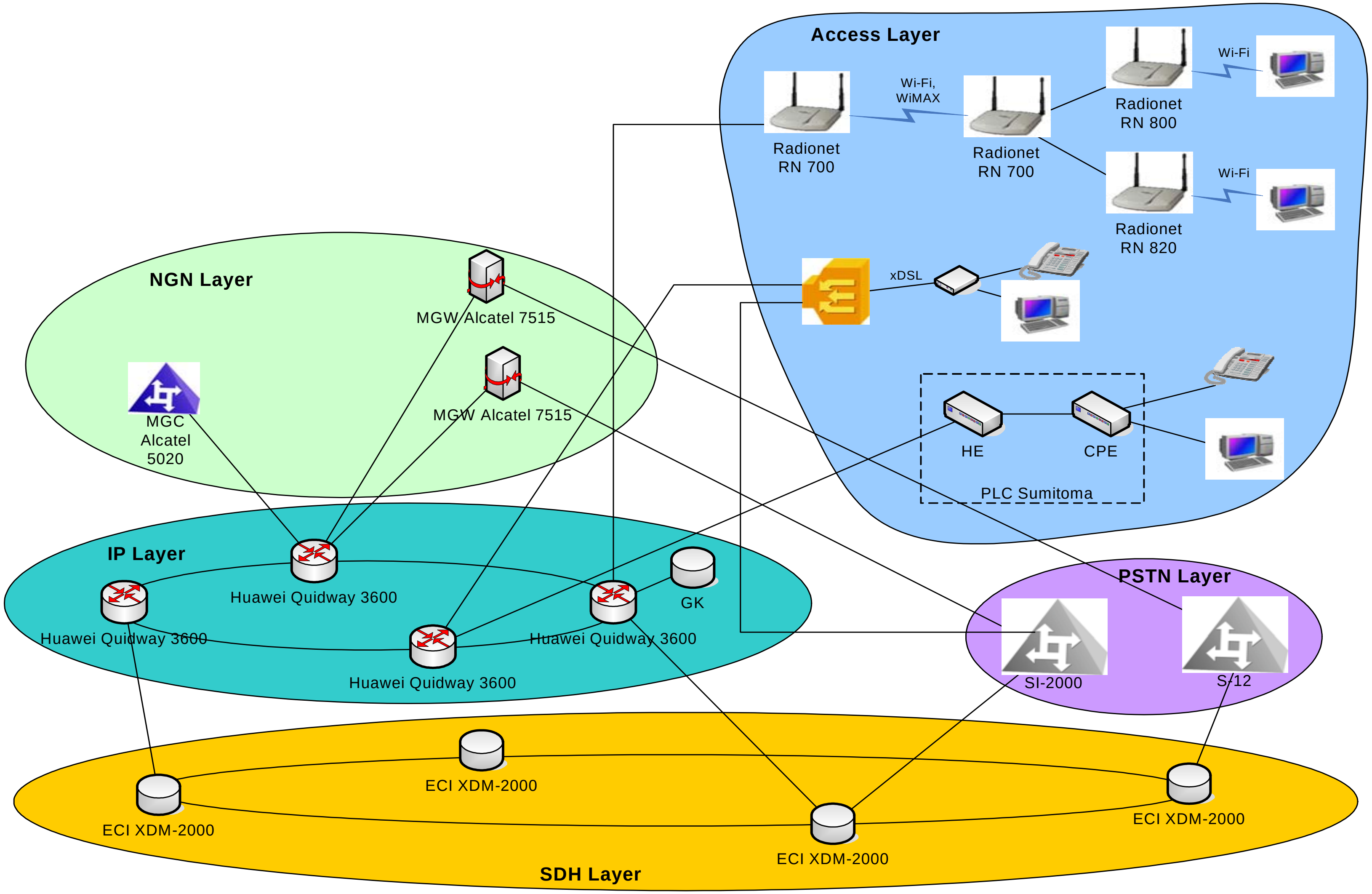
- Z.itfm series Recommendations on Methodology
  - work in progress
  - two major contributions - ETSI, Korea
  - similar methodology to X.290 Recommendation
  - some new concepts are being discussed
- Standards to be produced as required by Z.itfm
  - Requirements Clause
  - Test Suite Structure and Test Purposes (TSS&TP)
  - Implementation Conformance Statement (ICS)
  - The Abstract Test Suite
  - Implementation eXtra Information for Testing (IXIT)

Configuration of a Model Network

Q.3900 Recommendation identifies model network configuration for NGN testing

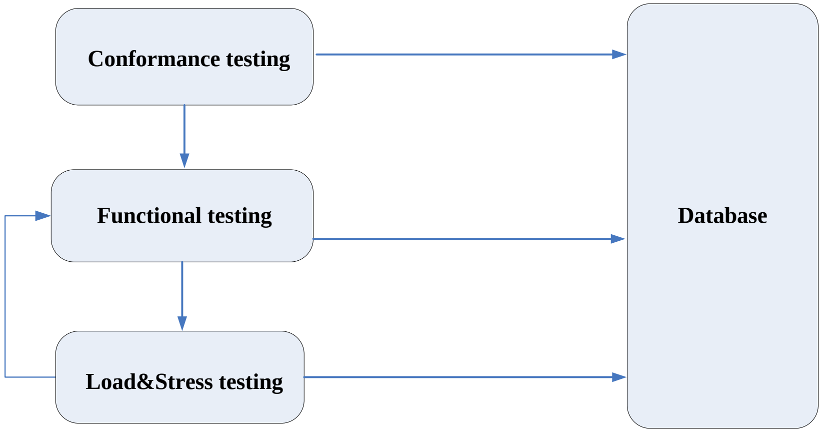


Model Networks can help Development of CIT Recommendations

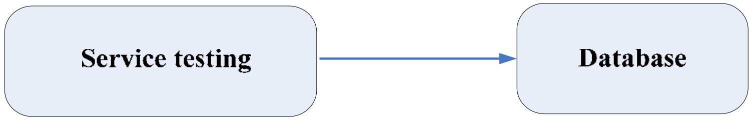


Testing Methods for Model Networks

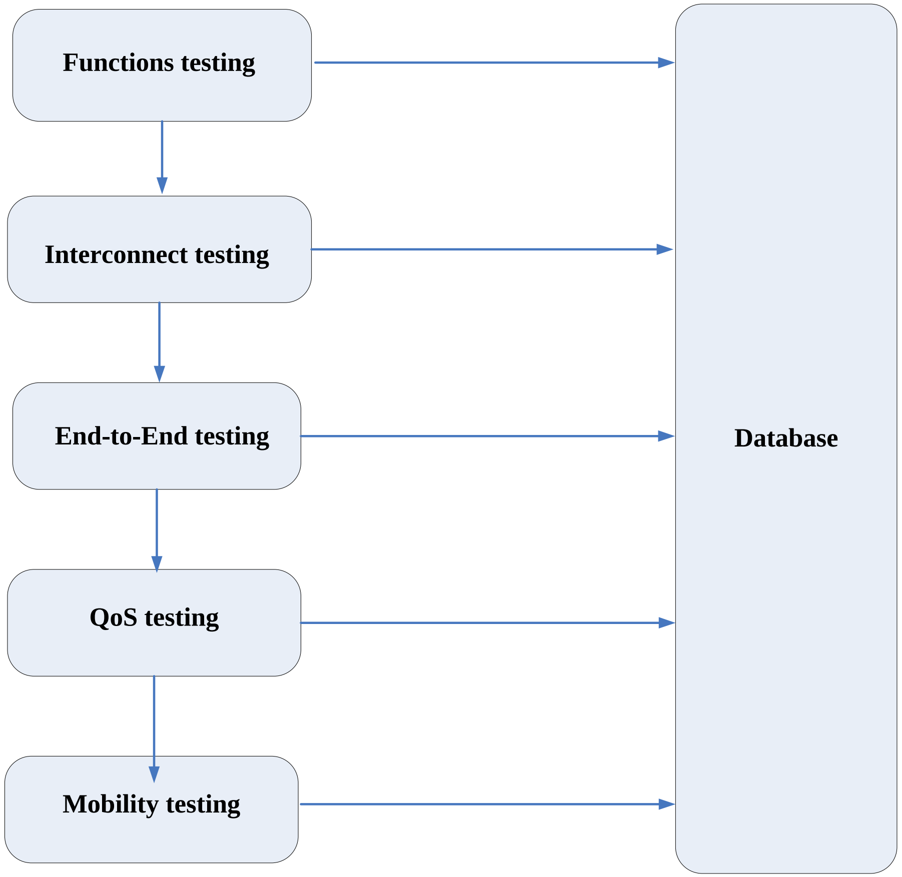
Possible NGN Test Methods when using Model Networks



NGN TM testing



Services testing method



NUT testing method

Trusting the Test Results

- Who does the testing
  - Third-party testing (independent test laboratory)
  - Second-party testing (product procurer or user)
  - First-party testing (product supplier)
- Testing can be made formal enough for certification of tested products

Adding Confidence

- Accreditation
  - checks for competence to carry out testing
  - checks for competence to issue certifications
- Certification
  - checks for conformity to a quality system standard
  - checks for conformance to the protocol standard
- Test Laboratories
  - Carry out testing
  - Prepare the Test Report

Conclusions

- **To the writers of Recommendations:**
  - Specify for Interoperability (Requirements, Functional Architecture, Protocol Details)
  - Validate for Interoperability (Technical Reviews, Interoperability Events, Test Specifications)
  - Test for Interoperability (Plan, Conformance and Interop., Use X.290 and Z.140 series Recommendations, Validate Test Specifications)
  - Maintain for Interoperability (Good standards are broken by poor maintenance, Make changes with care, Use well defined process)
- **If the above is not done, it is too late for interoperability**
- **Supporting standards for conformance and interoperability testing do not exist but must be developed**
- **Standards should be tested for errors and ambiguities prior to approval (as is done in IETF)**