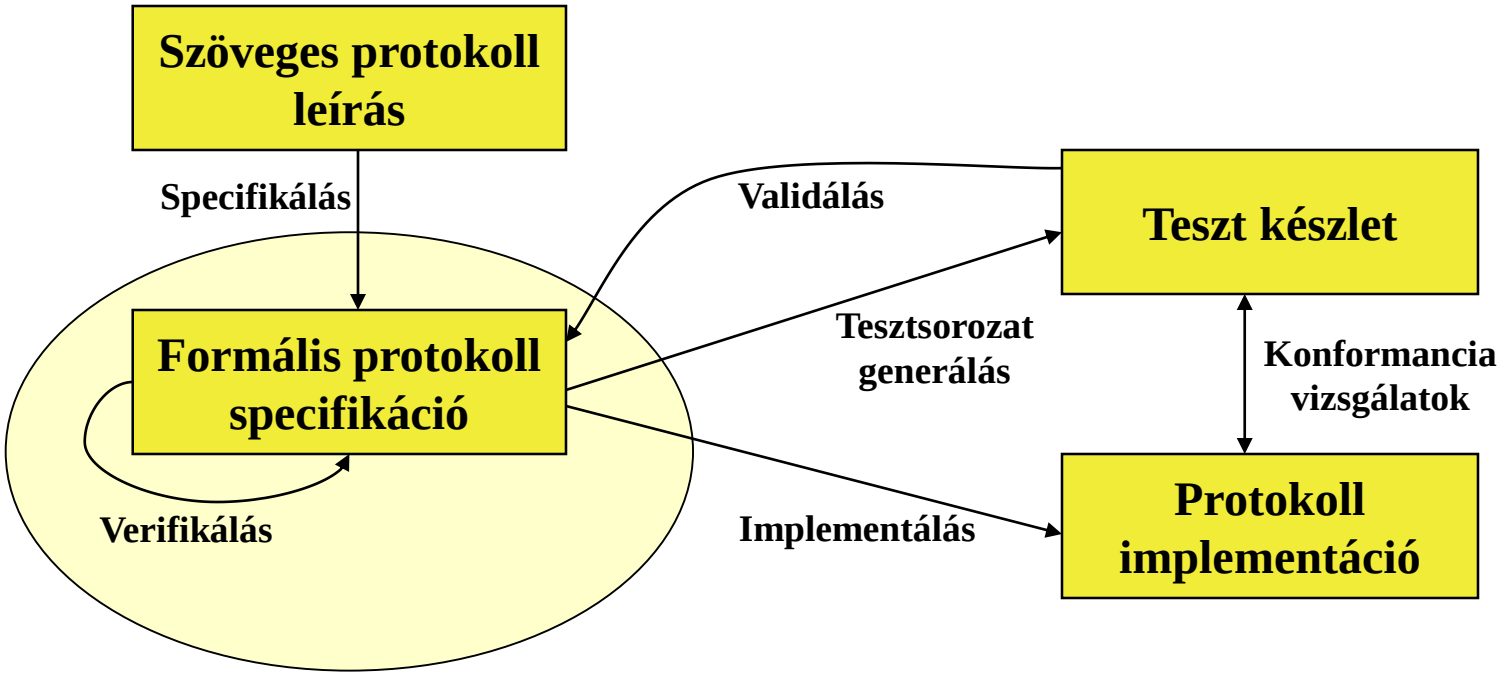
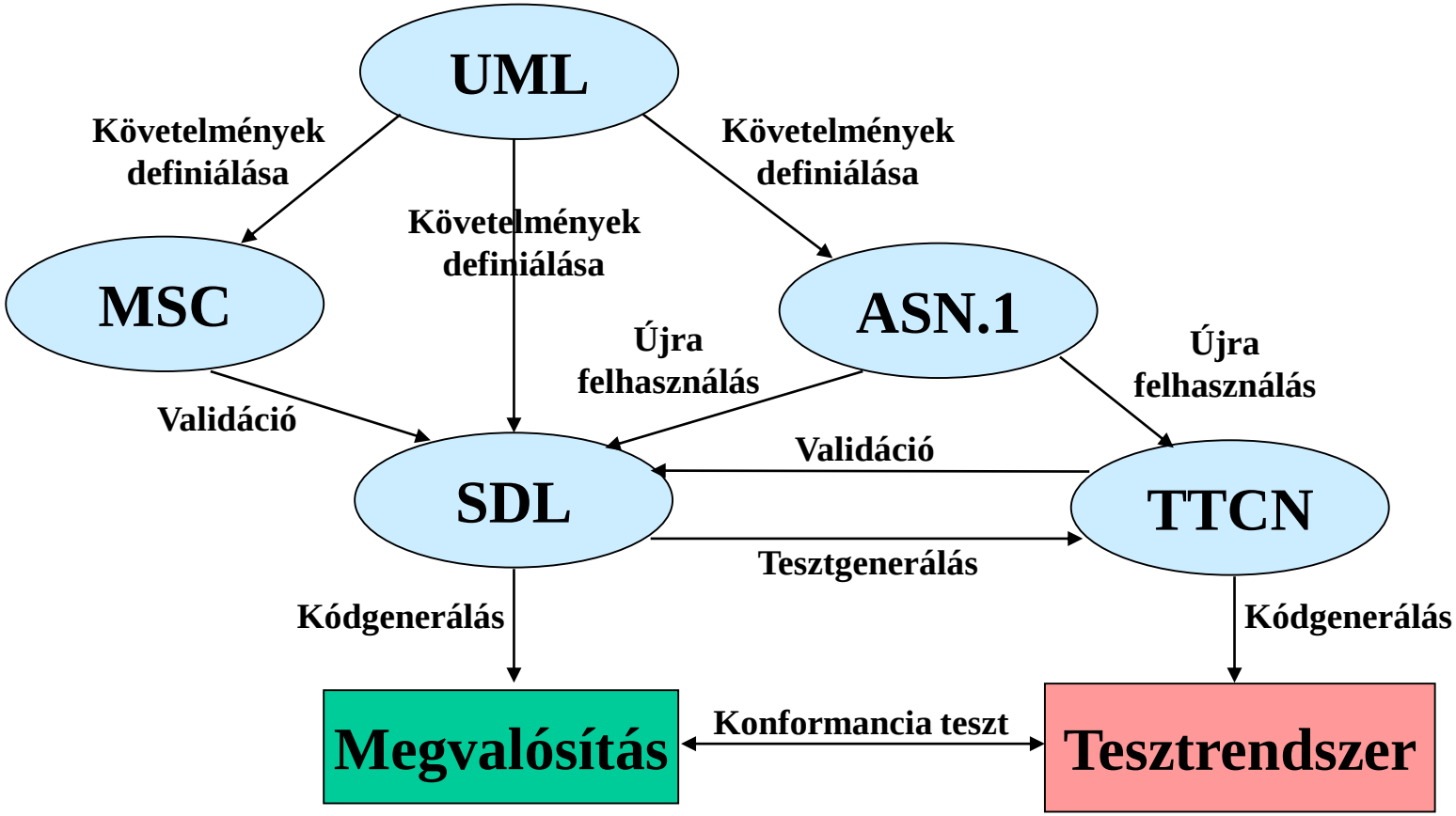
Konformancia tesztelés

Csöndes Tibor
Ericsson Kft., R&D,
BME-TMIT
Tibor.Csondes@ericsson.com,
csondes@tmit.bme.hu

Protokoll technológia (Protocol Engineering)



Kapcsolat a különböző leírónyelvek között



WHITE AND BLACK BOX TESTING

- White box testing – typically during development
  - Access to code
  - Access to development environment
- Black box testing
  - Internal structure of the code is not known/interested
  - Checks the communication between the tested entity and its environment
  - IUT/SUT – Implementation/System Under Test
  - Tester – may be decomposed
  - PCO – Point of Control and Observation

BLACK BOX TESTING

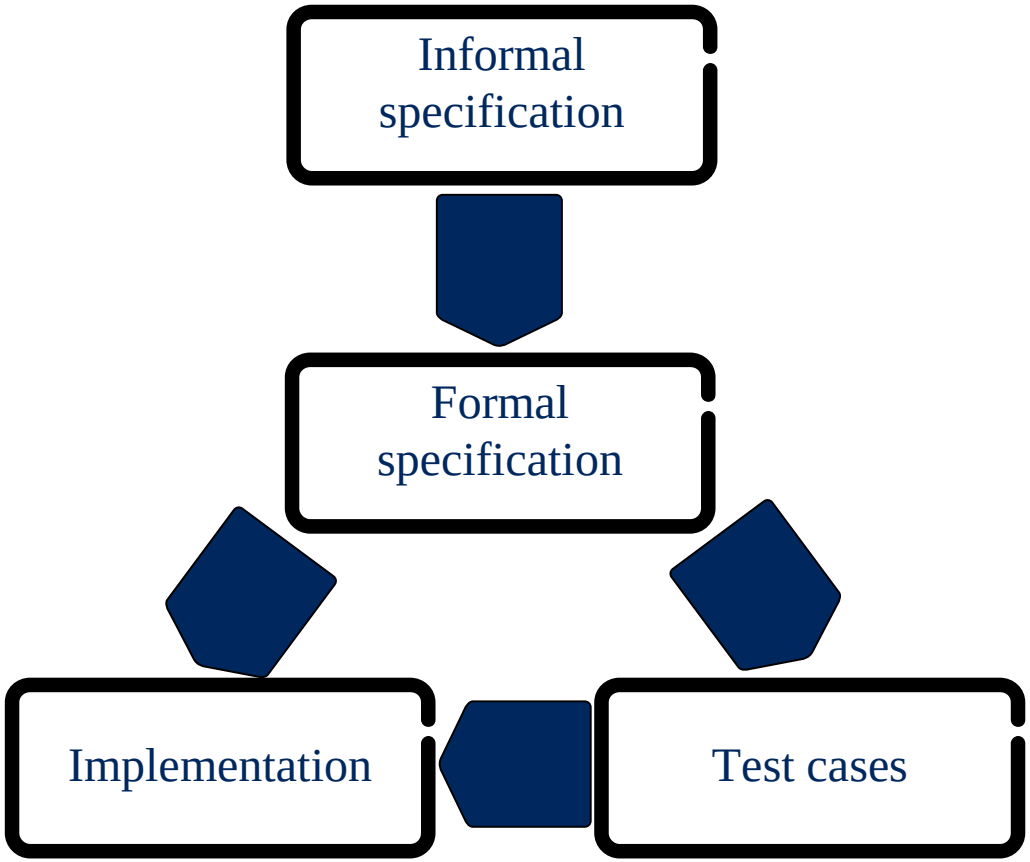
- Black box testing
  - Implementation/System Under Test
  - Point of Control and Observation

- Not possible to test all the situations
  - Test Purposes

Verdict:

pass,
fail,
inconclusive

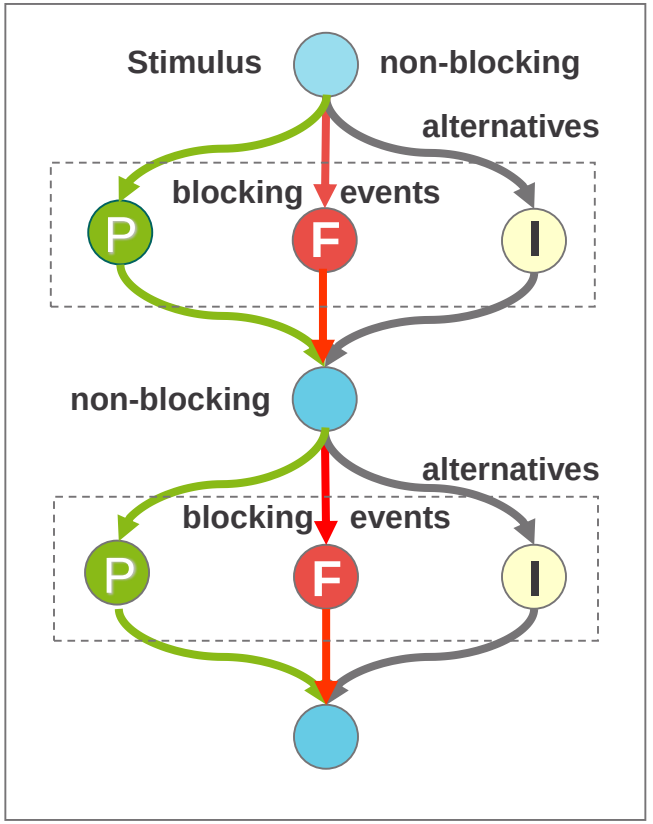
CONFORMANCE TESTING



- Checks if IUT conforms to its specification
- Experiments programmed into Test Cases

TEST CASES IN BLACK-BOX TEST

- Implementation of a Test Purpose
  - TP defines an experiment
- Focuses on a single requirement
- Returns verdict (pass, fail, inconclusive)
- Typically a sequence of action-observation-verdict update:
  - Action (stimulus): non-blocking (e.g. transmit PDU, start timer)
  - Observation (event): takes care of multiple alternative events (e.g. expected PDU, unexpected PDU, timeout)



Conformance Testing Methodology and Framework (CTMF)

- A szabványokat eredetileg OSI protokollok tesztelésére fejlesztették ki

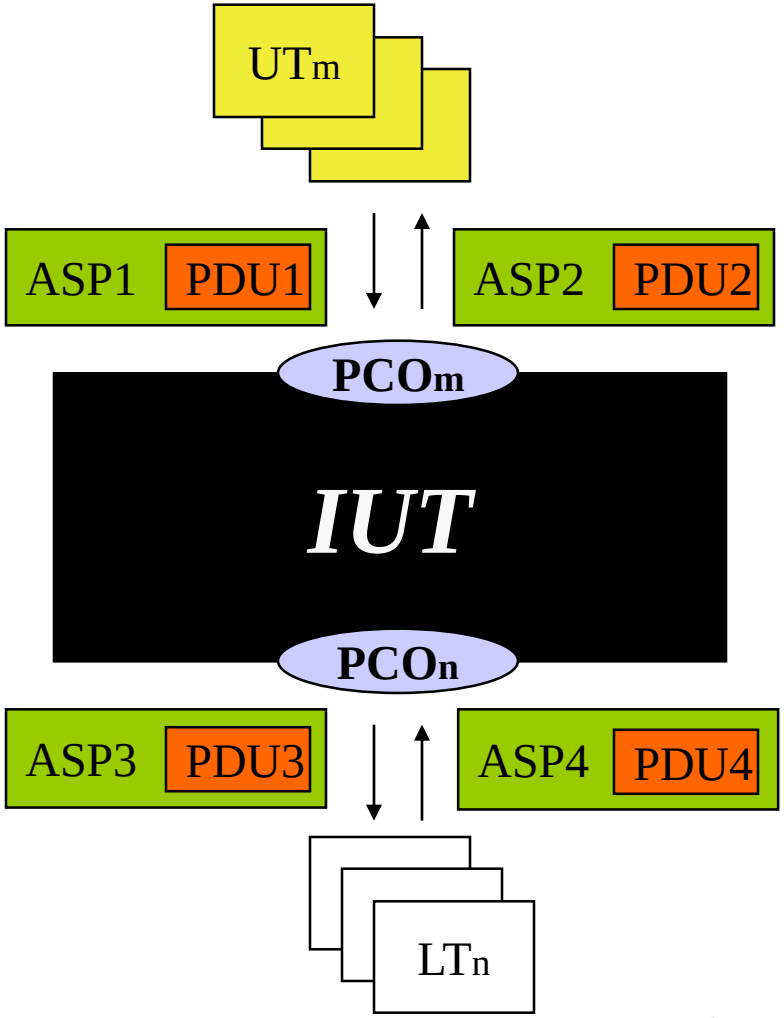
ISO/IEC	ITU-T	Title
9646-1	X.290	General Concepts
9646-2	X.291	Abstract Test Suite Specification
		Multi-protocol Testing
		Multi-party Testing
9646-3	X.292	TTCN Notation
		Concurrent TTCN
		Encoding and Modular TTCN
9646-4	X.293	Test Realization
9646-5	X.294	Requirements on Test Laboratories and Clients for the Conformance Assessment Process
9646-6	X.295	Protocol Profile Test Specification
9646-7	X.296	Implementation Conformance Statements

Terminológia

- **ASP** **Abstract Service Primitive**
- **ATM** **Abstract Test Method**
- **CP** **Coordination Point**
- **IUT** **Implementation Under Test**
- **LT** **Lower Tester**
- **PCO** **Point of Control & Observation**
- **PCTR** **Protocol Conformance Test Report**
- **PICS** **Protocol Implementation Conformance Statement**
- **PIXIT** **Protocol Implementation Extra Information for Testing**
- **PDU** **Protocol Data Unit**
- **SCS** **System Confomance Statement**
- **SCTR** **System Conformance Test Report**
- **SUT** **System Under Test**
- **TCP** **Test Coordination Procedure**
- **UT** **Upper Tester**

Conformance Testing Methodology and Framework - CTMF

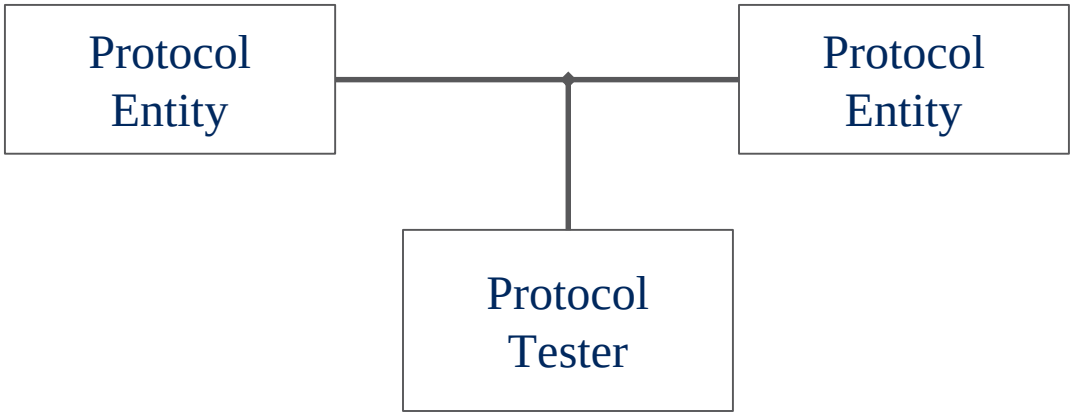
- Az IUT egy fekete doboz
- Látható viselkedés PCO-kon keresztül vezérelhető és megfigyelhető
- A PDU-k ASP-be beágyazva kerülnek küldésre és fogadásra a felső (UT) vagy az alsó (LT) teszter által
- A teszt koordináció többrésztvevős tesztelés esetén koordinációs pontokon (CP) keresztül történik



Követelmények csoportosítása

- Kötelező (mandatory)
  - Mindenképp teljesíteni kell
- Feltételes (conditional)
  - Adott feltételektől függően kell teljesíteni
- Opcionális (options)
  - Gyártó által szabadon választható
- Pozitív vagy negatív
- Statikus vagy dinamikus

PASSIVE TESTER



- Only observes
  - waits for error
    - no guarantee to happen
- Protocol Analyzer

ACTIVE TESTER

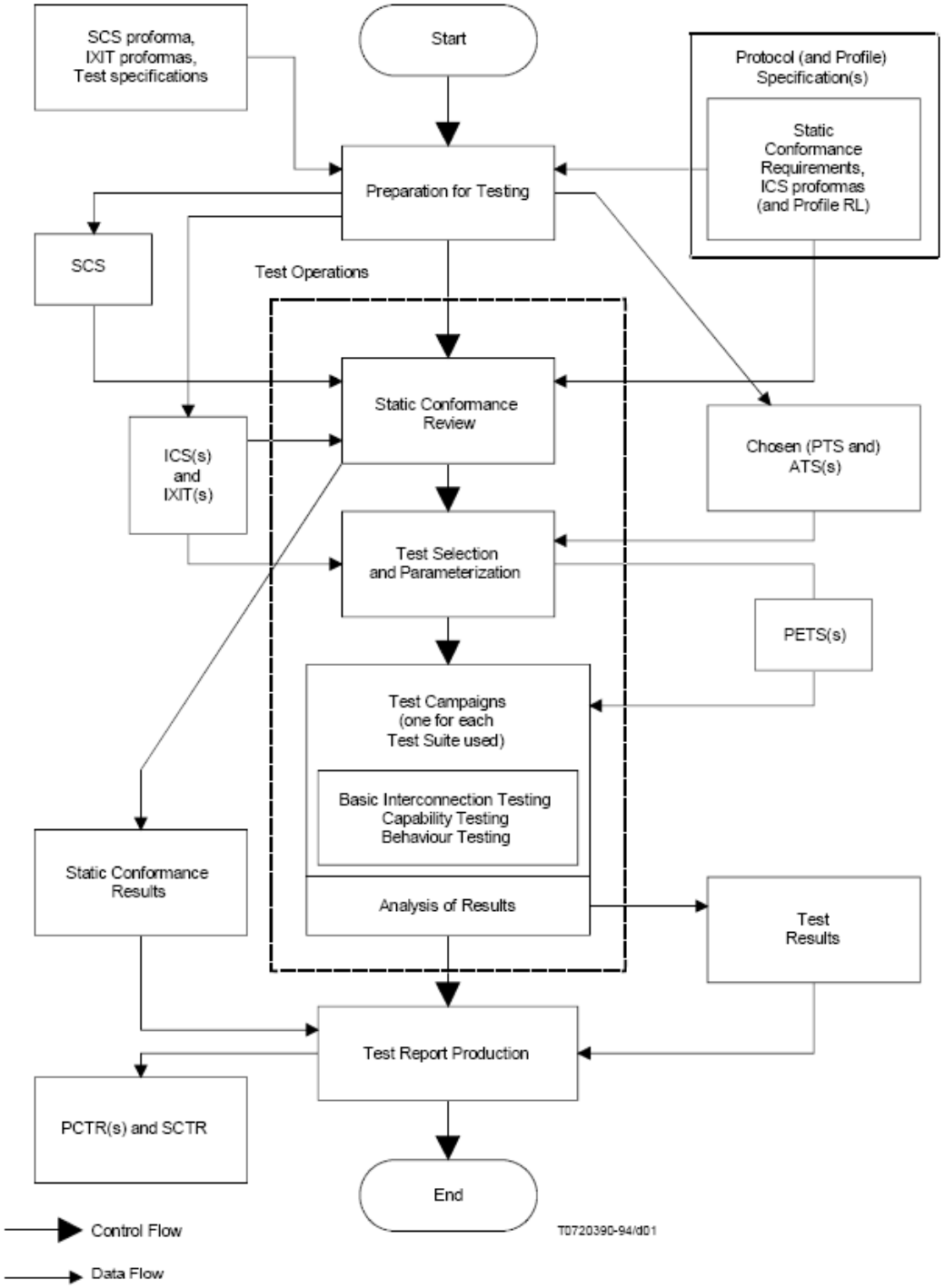
- Active
  - can send messages
- Valid testing
- Provocative testing
  - Invalid
    - Sends syntactically incorrect messages
  - Improper
    - Sends syntactically correct messages, but at wrong time/state
- Test cases are generated before testing starts

Konformancia tesztek típusai

- Basic Interconnection Tests
  - Alapvető konformancia követelményeket teljesíti-e a rendszer
- Capability Tests
  - Az alap képességek összhangban vannak-e a ICS-sel
- Behaviour Tests
  - Igazi tesztek melyekkel a megfigyelhető viselkedést teszteljük
- Conformance Resolution Tests
  - Nem szabványos tesztek, protokoll részleteibe menő tesztelés

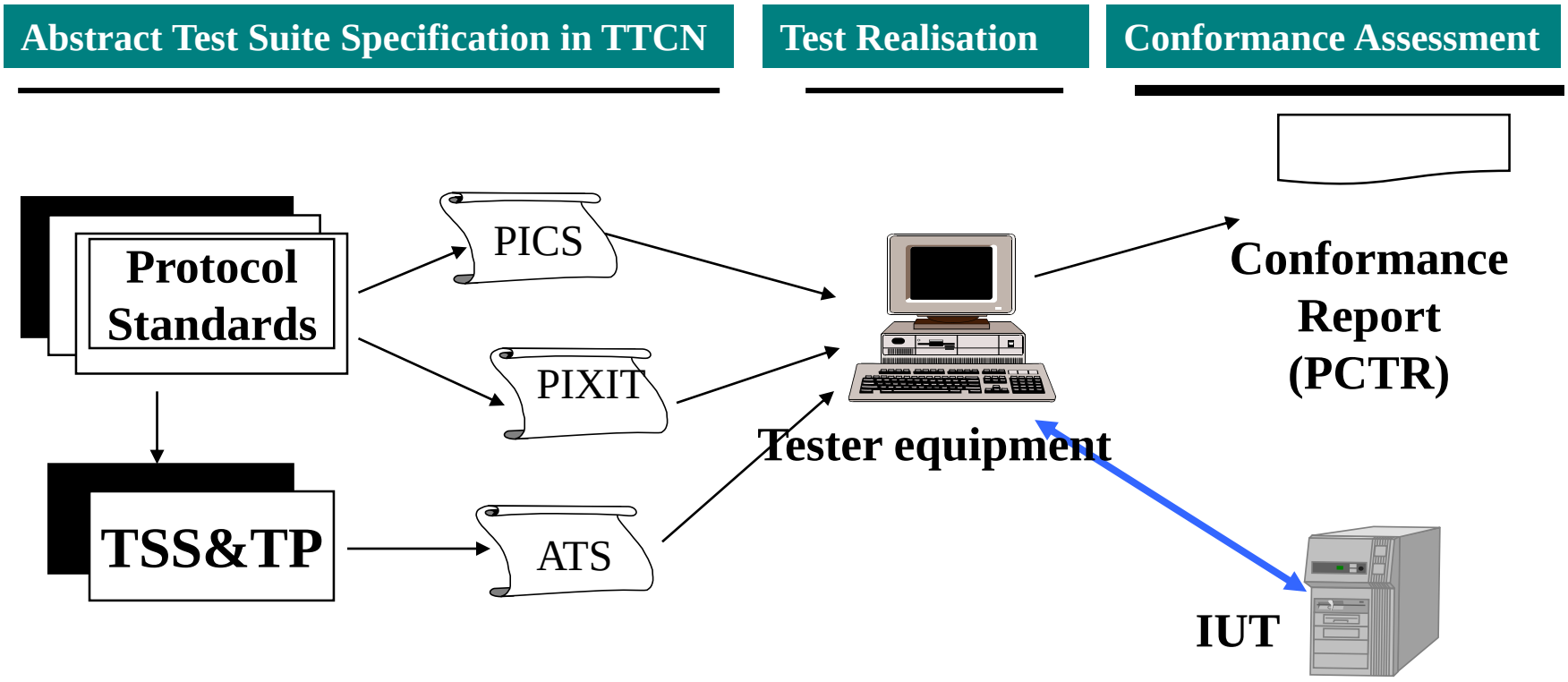
Konformancia tesztelés módszertana

- A tesztelés lépései:
  1. Tesztelés előkészítése (PICS, PIXIT, parametrizálás)
  2. Teszt végrehajtás - test campaign (teszt kiválasztás)
  3. Teszt jelentés elkészítése (PCTR, SCTR)
- Teszt cél (Test Purpose): egy vagy több követelmény megfogalmazása
- Teszt eset (Test Case): egy teszt cél megvalósítása (ETSI), ITU-nál nem minden esetben egyértelmű a megfeleltetés
- A teszt esetek teszt csoportokba gyűjthetők az absztrakt tesztsorozatban (ATS)



Conformance assessment process overview

Konformancia tesztelés dokumentumai



Static Testing

Dynamic Testing

PICS:Protocol Implementation Conformance Statement
PIXIT: Protocol Implementation eXtra InformaTion

ATS:Abstract Test suite
PCTR:Protocol Conformance Test Report

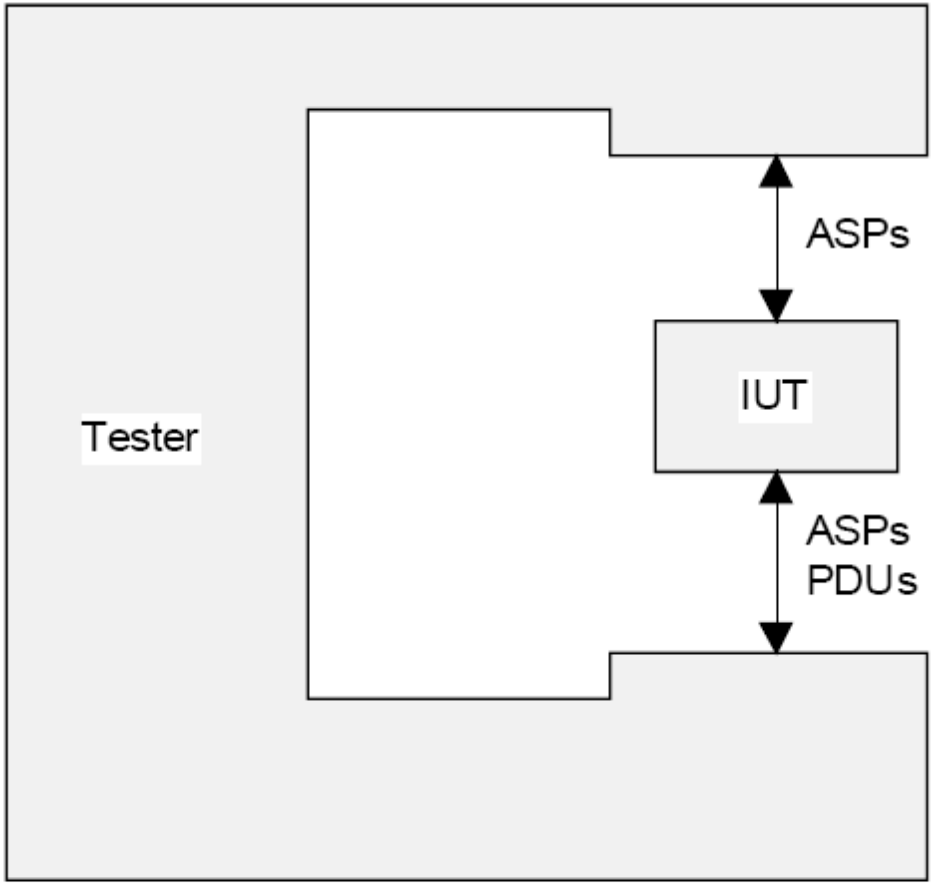
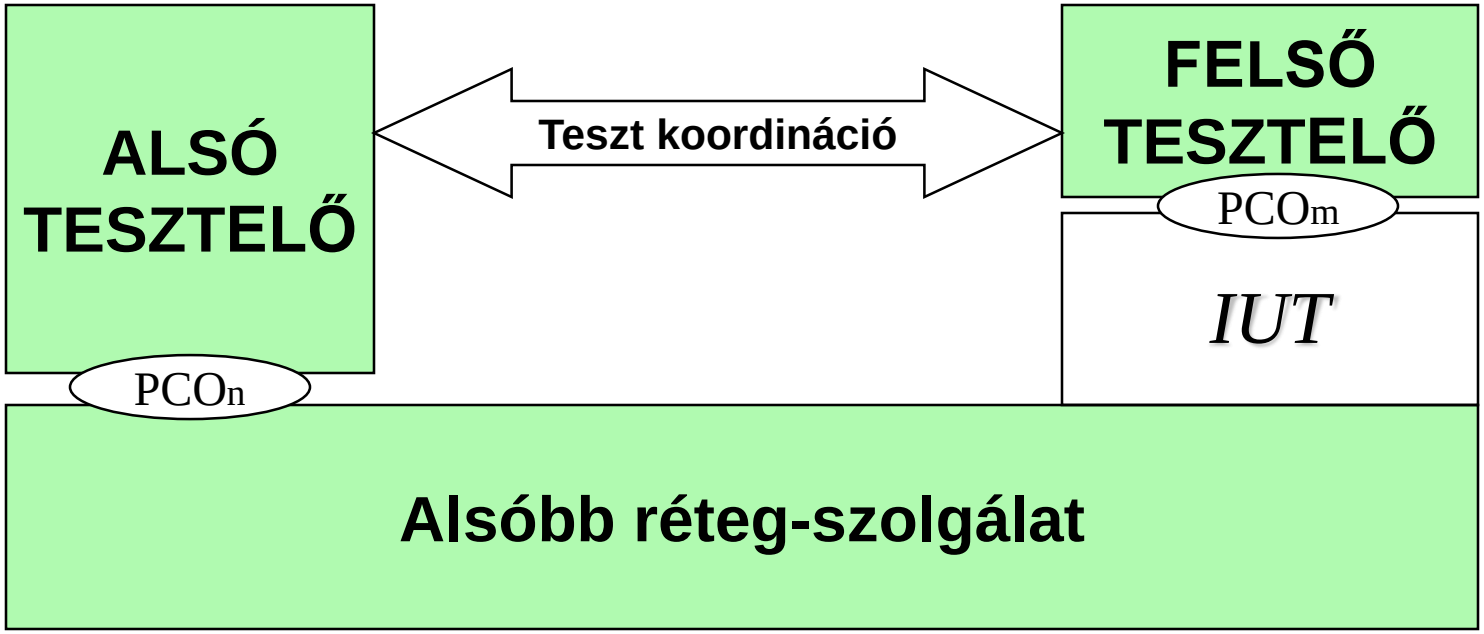


FIGURE 6/X.290

Conceptual testing architecture

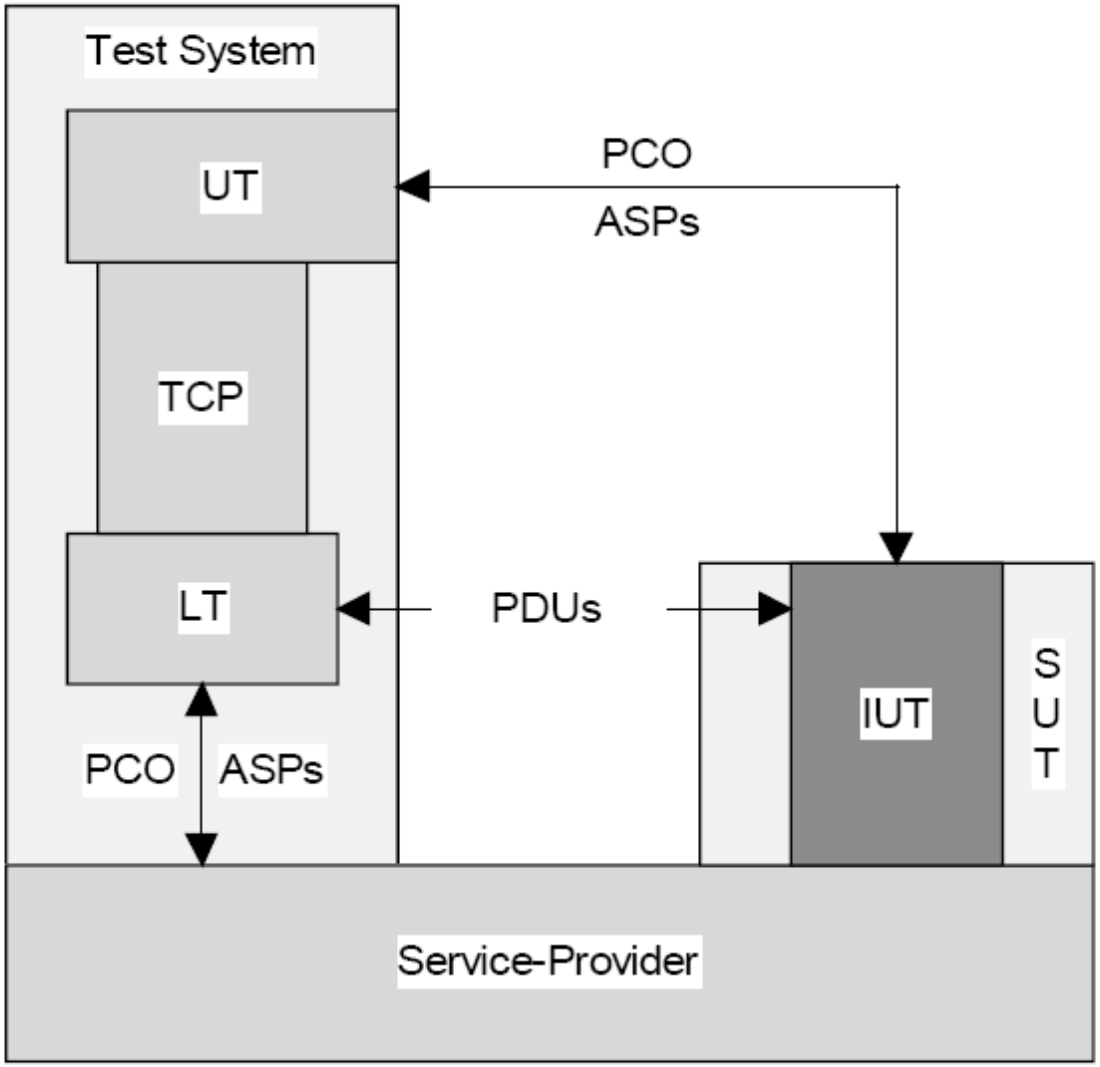
Absztrakt teszt módszerek egy résztvevős tesztelés esetén



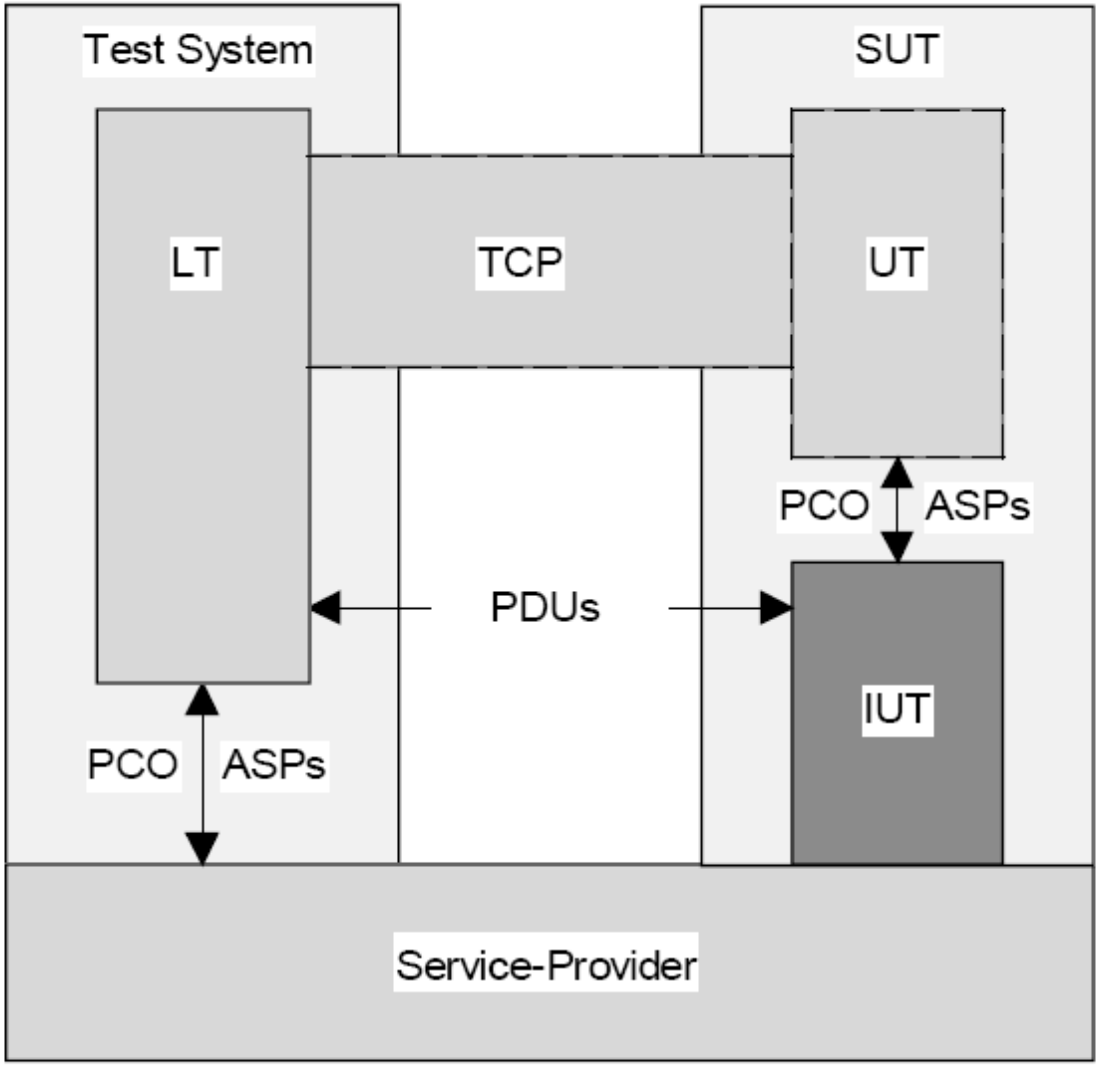
ATMs: Helyi, Távoli, Elosztott és Koordinált (X.290 vagy ISO9646-1) teszt architektúra

Teszt módszerek csoportosítása

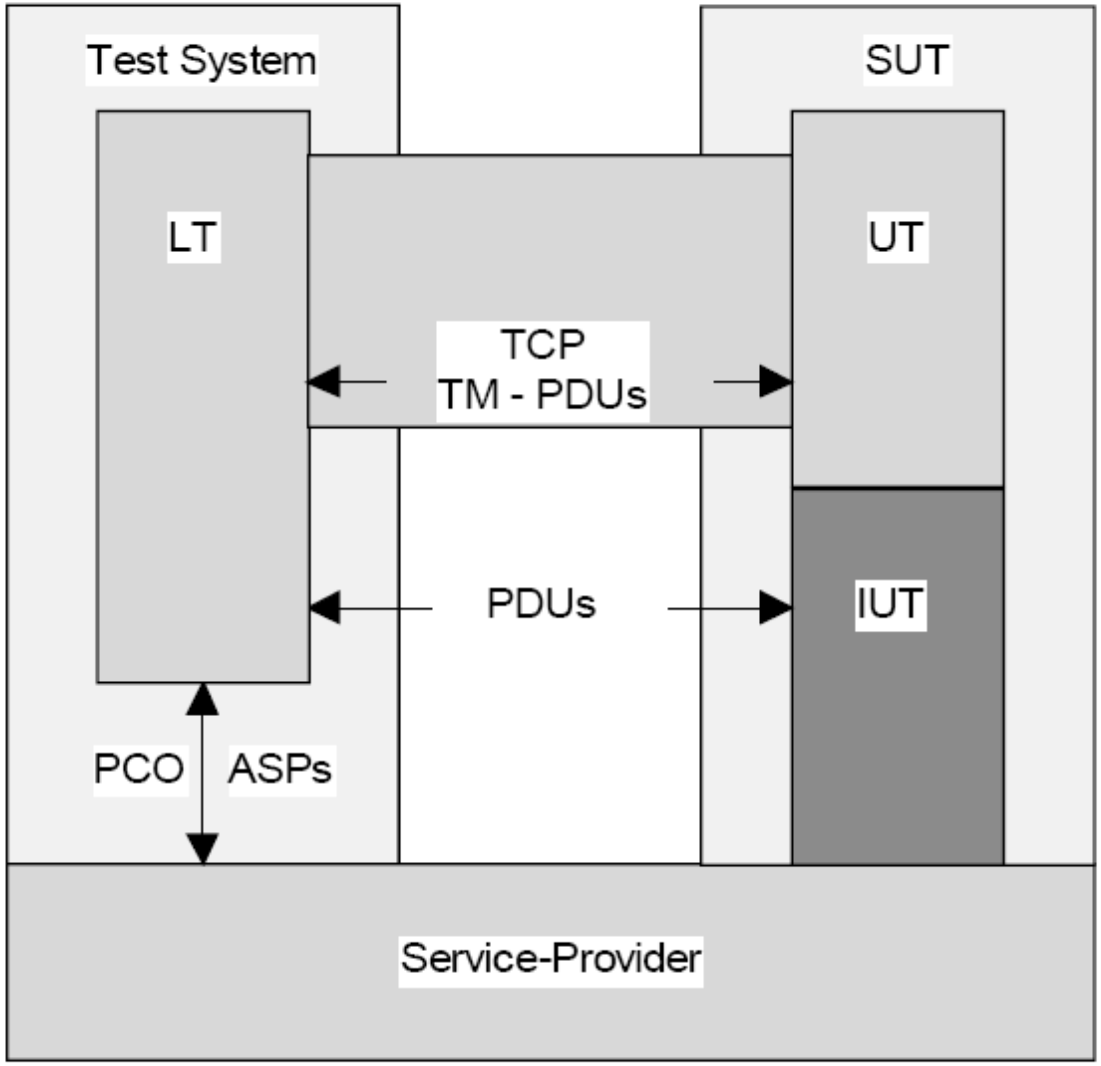
- Mind az alsó és mind a felső tesztelő PCO-ja elérhető:
  - Helyi (Local)
    - Felső PCO egy szabványos hardver interfész
  - Elosztott (Distributed)
    - Felső tesztelő az SUT része (akár szabványos TTCN nyelven)
- Csak az alsó tesztelő PCO-ja érhető el:
  - Koordinált (Coordinated)
    - Felső tesztelő szabványos teszt menedzsment protokollal vezérelhető
  - Távoli (Remote)
    - Nincs igazi felső tesztelő vagy nem érhető el, teszt koordináció is esetleges (helyette pl. implicit send lehet)



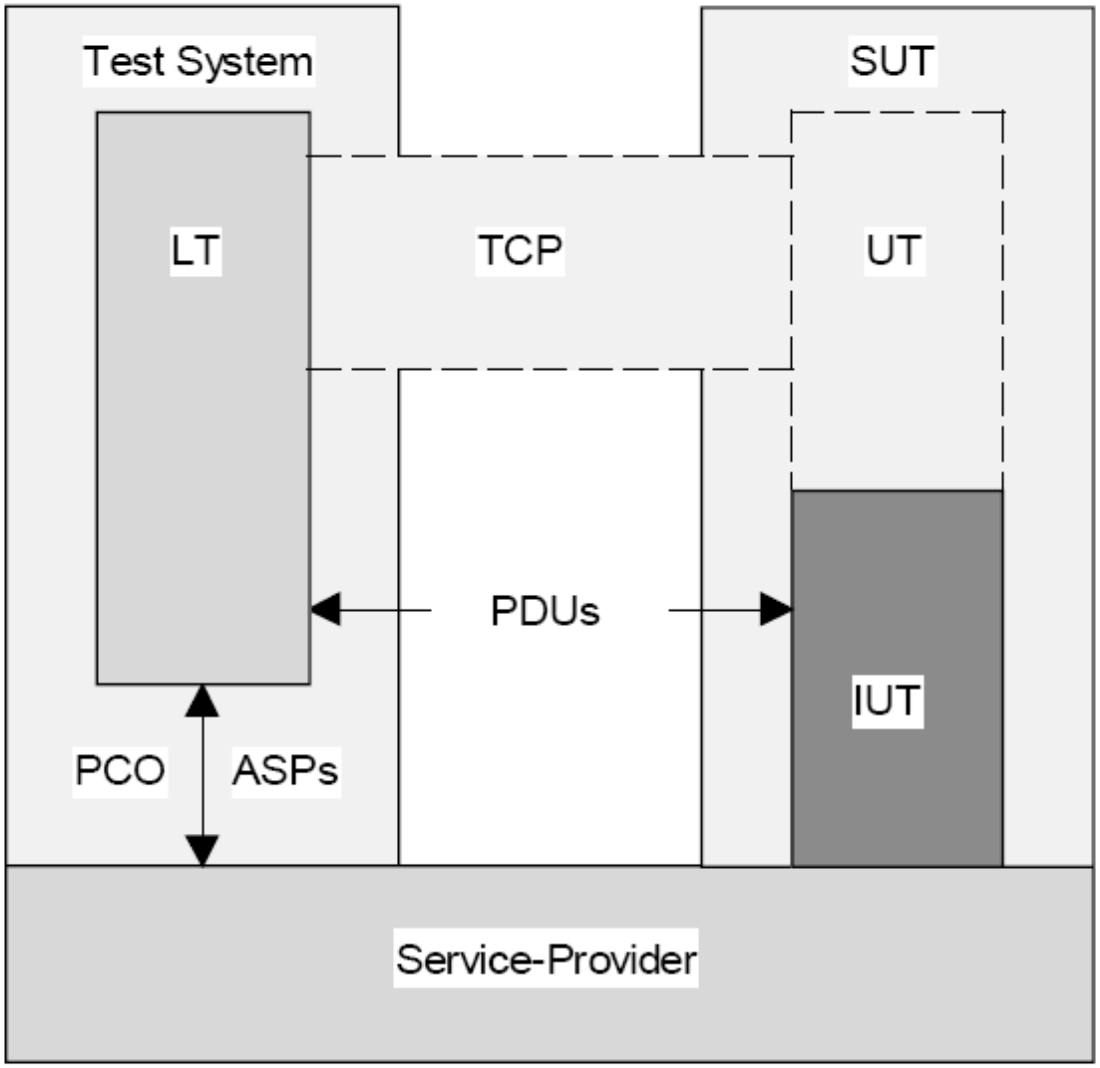
a) The Local test methods



b) The Distributed test methods

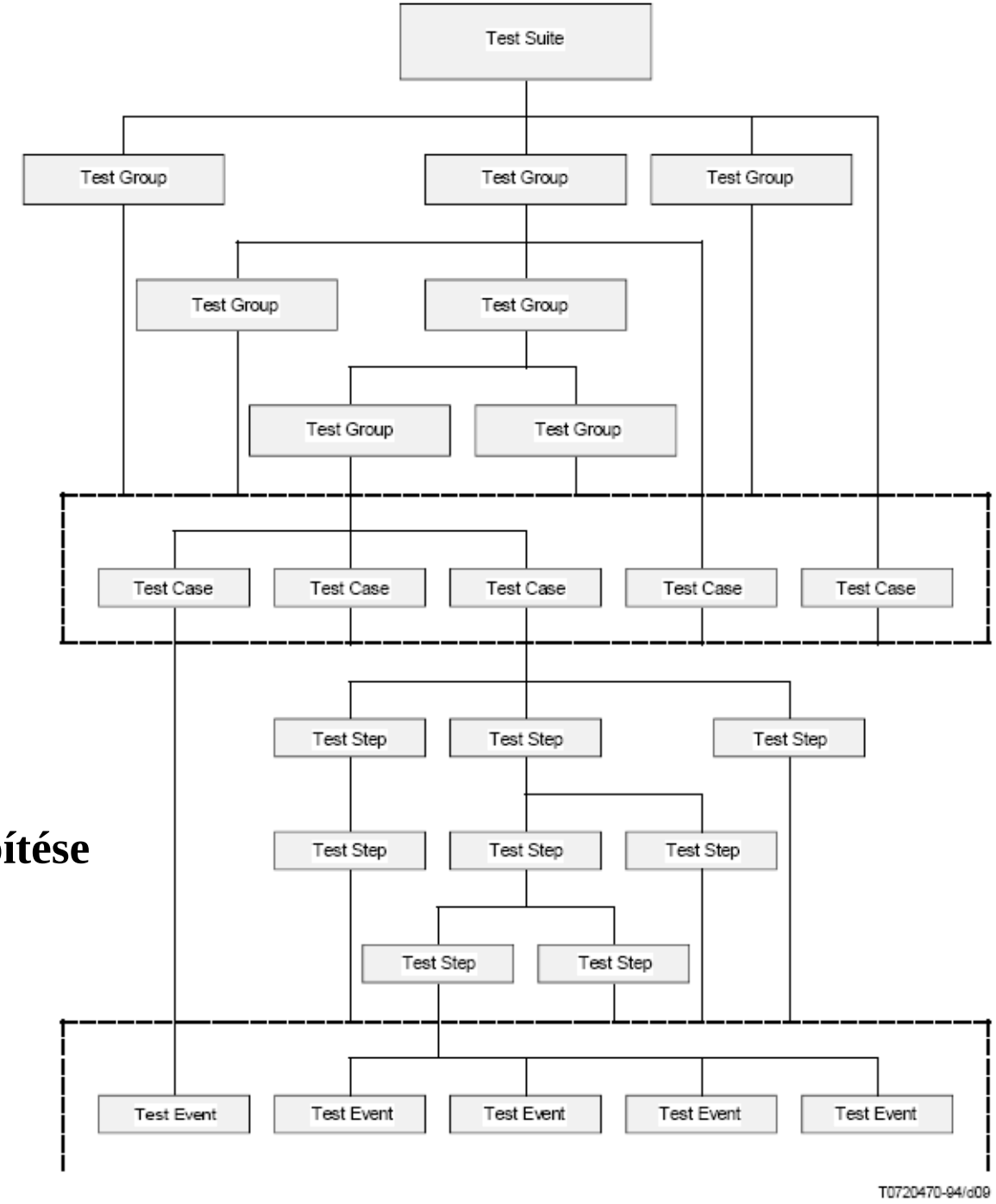


c) The Coordinated test methods



d) The Remote test methods

Teszt készlet felépítése



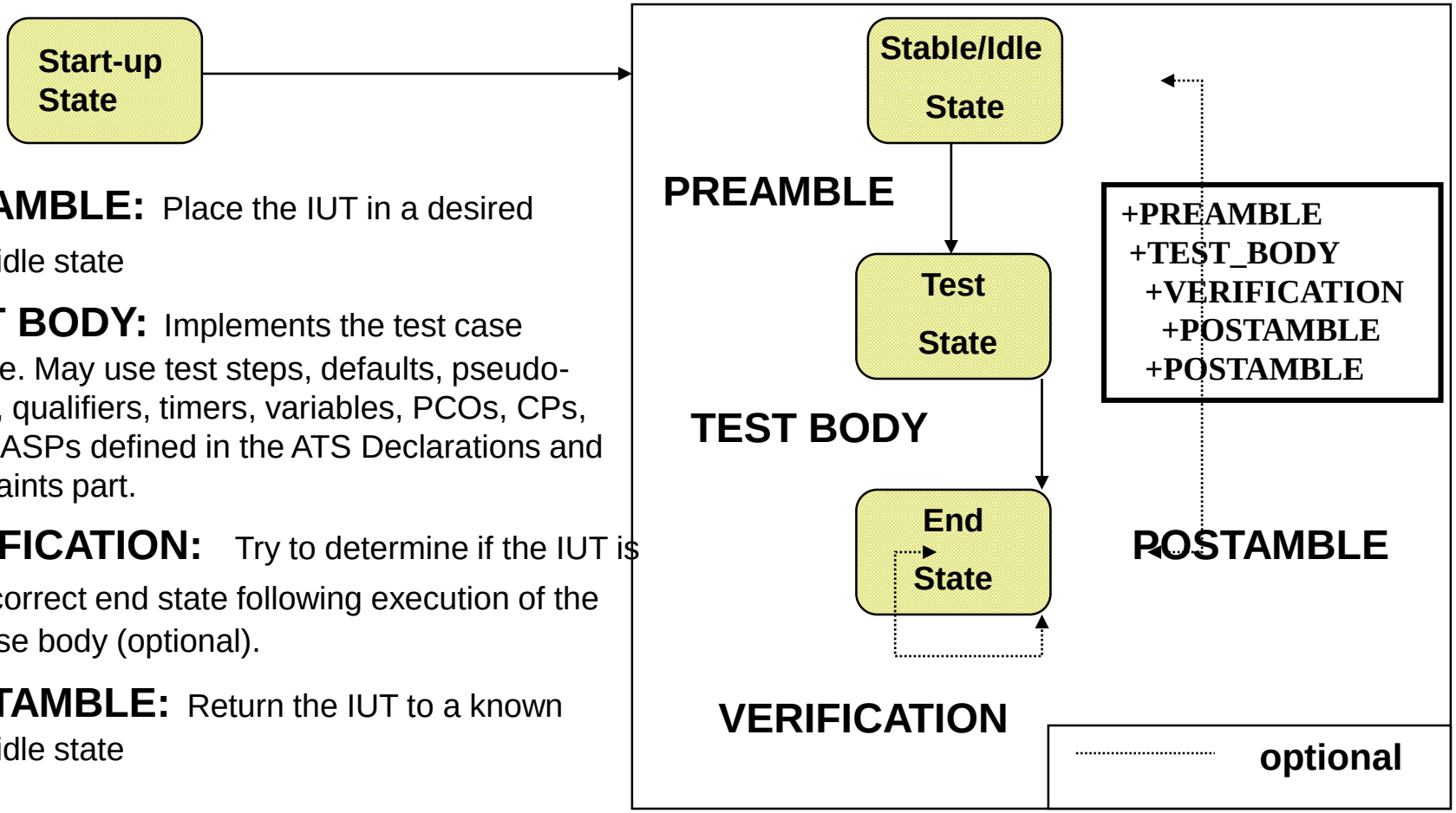
A Test Case consists of...

PREAMBLE: Place the IUT in a desired stable/idle state

TEST BODY: Implements the test case purpose. May use test steps, defaults, pseudo-events, qualifiers, timers, variables, PCOs, CPs, PDUs, ASPs defined in the ATS Declarations and Constraints part.

VERIFICATION: Try to determine if the IUT is in the correct end state following execution of the test case body (optional).

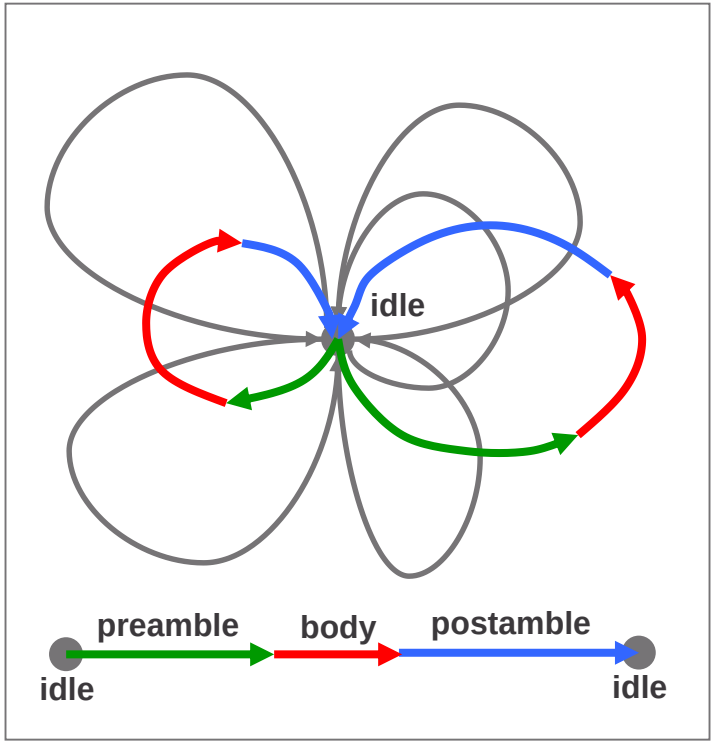
POSTAMBLE: Return the IUT to a known stable/idle state



Typical Test Case Structure

INDEPENDENCE AND STRUCTURE OF ABSTRACT TEST CASES

- *Abstract test cases* should contain
  - *preamble:* sequence of test events to drive IUT into *initial testing state* from the *starting stable testing state*
  - *test body:* sequence of test events to achieve the *test purpose*
  - *postamble*: sequence of test events which drive IUT into a *finishing stable testing state*
- Preamble/postamble may be absent



What is Model Based Testing

Conventional Testing vs. Model Based Testing

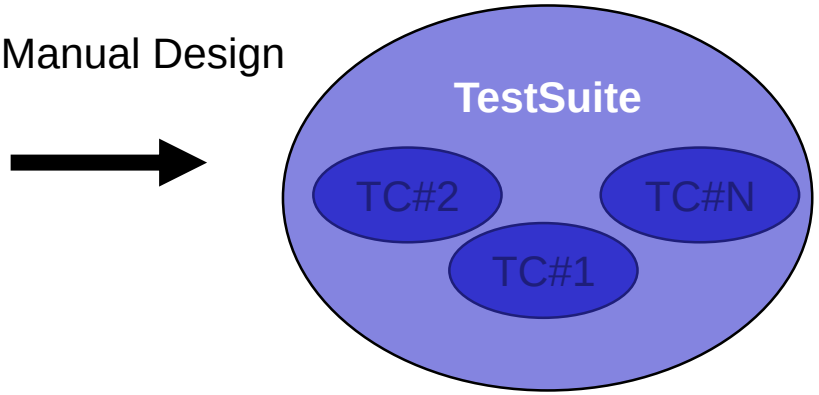
TCP Connection establishment
"The active open is performed by the client sending a SYN to the server.
It sets the segment's sequence number to a random value A.

In response, the server replies with a SYN-ACK. The acknowledgment number is set to one more than the received sequence number (A + 1), and the sequence number that the server chooses for the packet is another random number, B.

Finally, the client sends an ACK back to the server. The sequence number is set to the received acknowledgement value, and the acknowledgement number is set to one more than the received sequence number i.e. B + 1. "

Specification

Manual Design



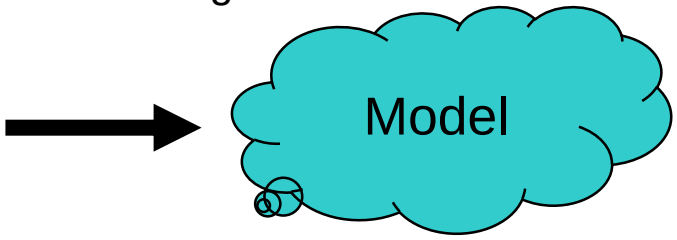
Model Based Testing

TCP Connection establishment
"The active open is performed by the client sending a SYN to the server.
It sets the segment's sequence number to a random value A.

In response, the server replies with a SYN-ACK. The acknowledgment number is set to one more than the received sequence number (A + 1), and the sequence number that the server chooses for the packet is another random number, B.

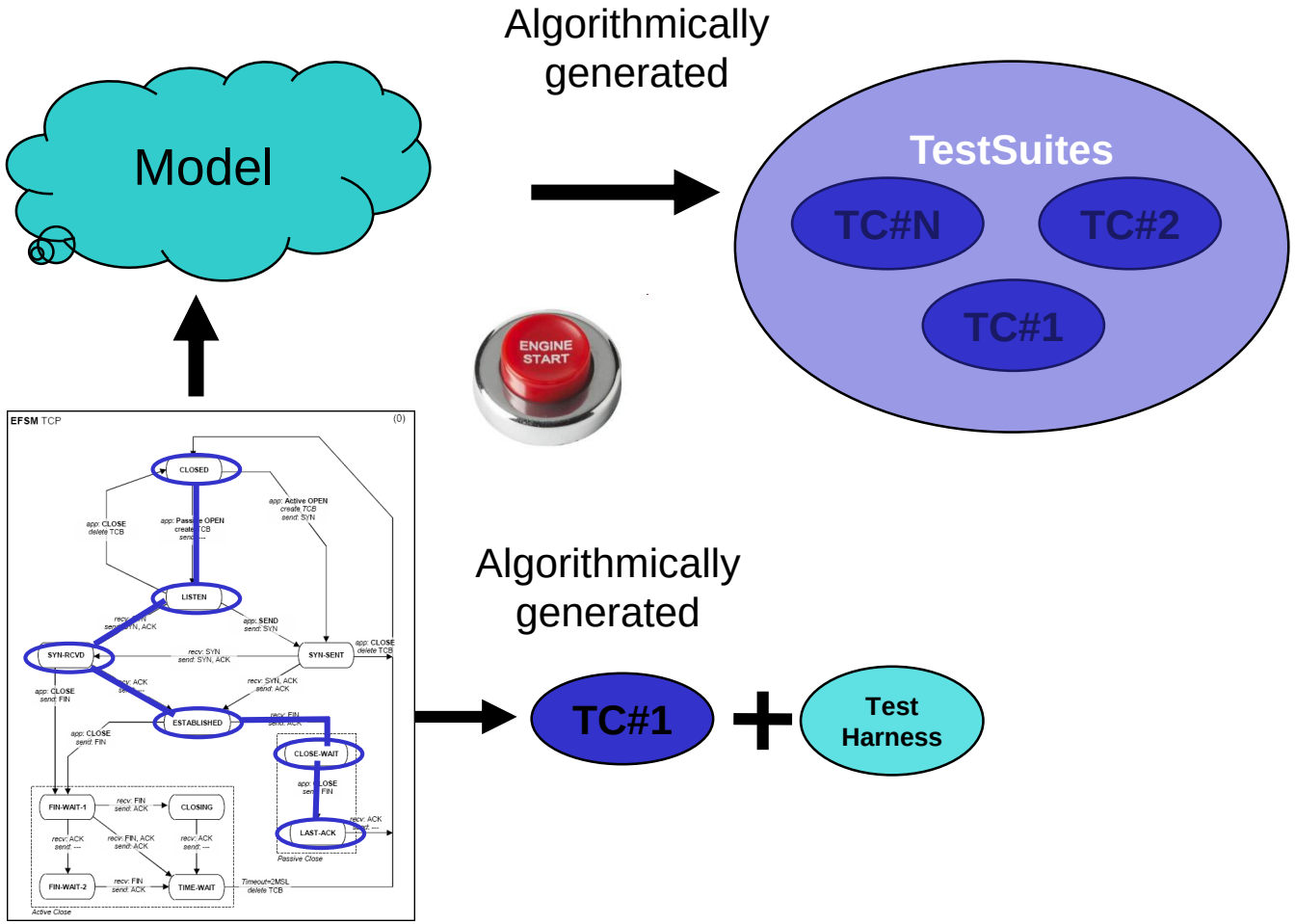
Finally, the client sends an ACK back to the server. The sequence number is set to the received acknowledgement value, and the acknowledgement number is set to one more than the received sequence number i.e. B + 1. "

Manual Design



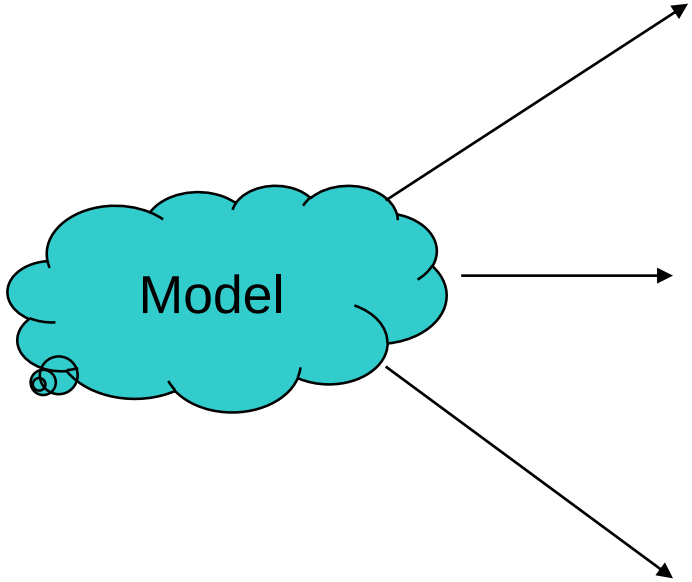
What is Model Based Testing

Conventional Testing vs. Model Based Testing

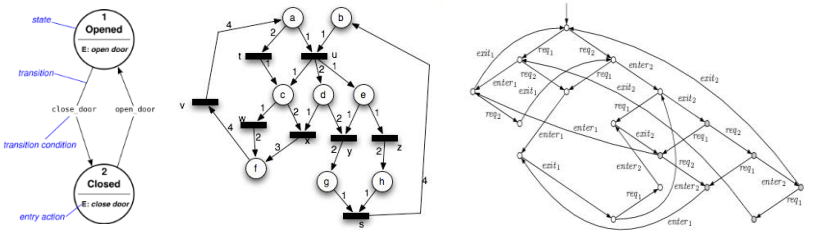


The Model

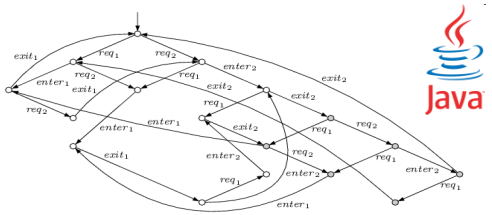
Model description



Graphical based: FSM;
Petri Net; Label Transition System



Mixed: Graphical Solution
extended by a language
(c++; java; C#...etc.)

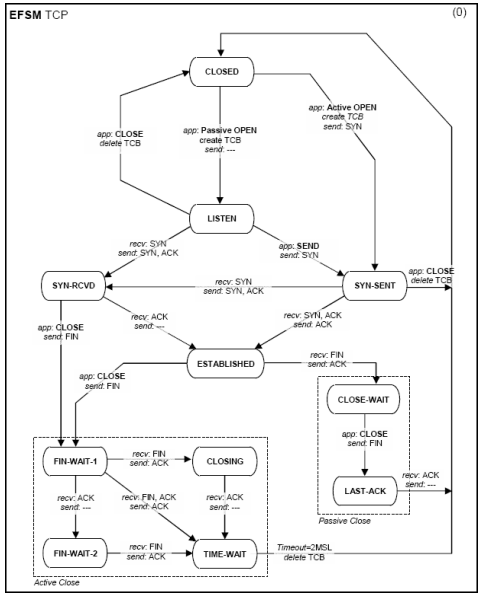


Pure language based: C++;
Java; C#; TTCN-3

Programming competence

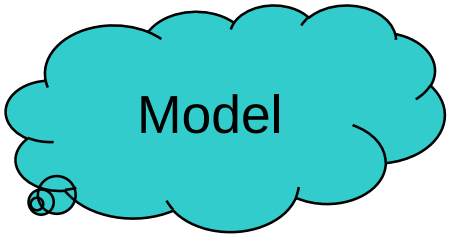
The Algorithm

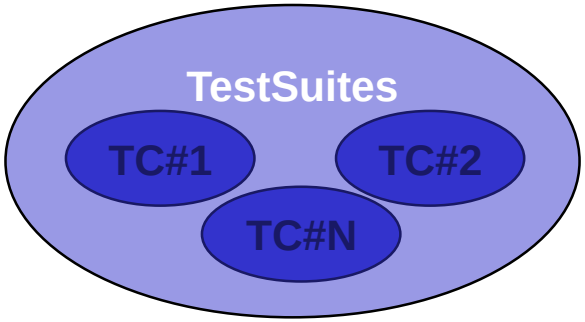
Magic Algorithm: describes how to generate the Test cases



Coverage:

- Traverse every state
- Traverse some state
- Traverse every link
- …etc



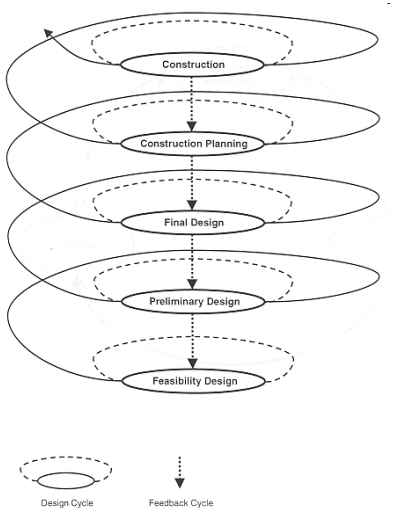


Model Based Testing on Field

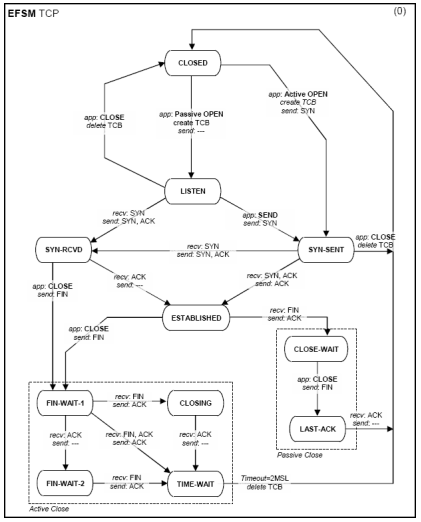
Pros and Cons of Model Based Testing

- Reduces fault slip through

Design phase



Testing phase



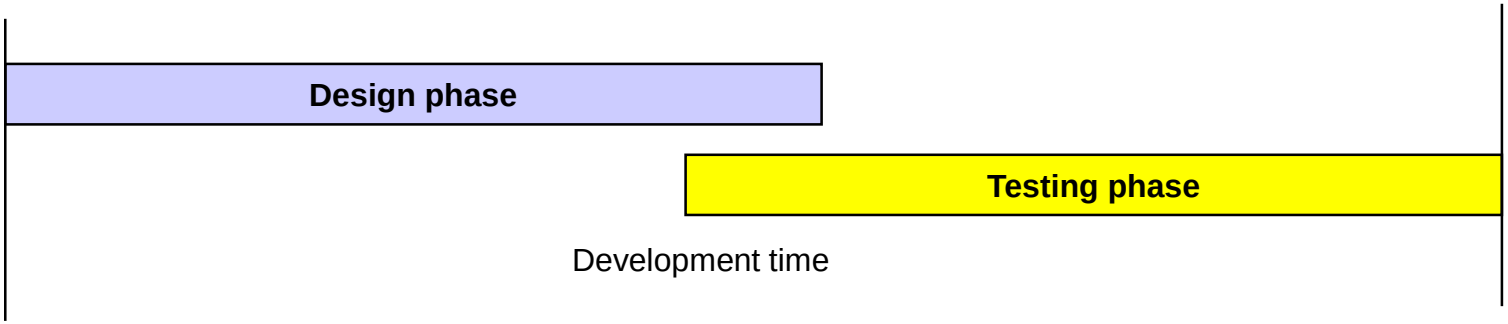
Model development of the Design and model development of the Testing could take place parallel

➔ model development for testing verifies the model of the design

➔some faults could be found in the “development phase”

➔Reduces development time

➔Model Driven Engineering



Model Based Testing Why?

Pros and Cons of Model Based Testing

- Reduces fault slip through
- Maintenance
  - ›If the specification changes
    - ›Change all the affected TestCases + TestHarenss
      - ›Complicated !!
    - ›Easier maintenance!!