Fraunhofer FOKUS

Oracles in TTCN-3 and UTP

Ina Schieferdecker

2012, May 22nd, CREST Workshop, London

Outline

- Oracles, Test Automation and (Test) Models
- Oracles in TTCN-3
- Test Automation (Oracle) Examples

Test oracle as part of a test case

- A test case is a tuple of
  - Pre-, post- and side conditions
  - Test inputs (stimuli)
  - Test outputs (expected responses)

  (see e.g. ISO/IEC CTMF, FMCT or ISTQB)

→ Do not separate test inputs from oracles

→ Both can/are to be realized by an automated test system

Test oracle as part of a test automation solution

- Two principle ways
  - “Liveness” checking

    → logical description
    → used in monitoring, passive testing, active testing

    TTCN-3 for passive and active testing, but also “logical complement/extension TTCN-3” for continuous invariant checking during test execution

  - “Safety” checking

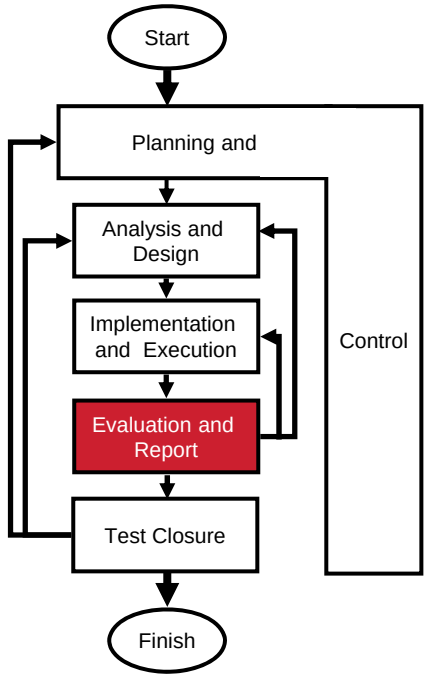
    → declarative description
    → used in active testing

    just TTCN-3

Test oracle as part of a generic automated test solution

- Oracles can be part of generic test automation solution
  - Specify expected responses
  - Evaluate received responses
  - Decide about end of test case or if and how to continue
  - If possible, help the tester to find the place of mismatch in expected and received system responses
- However, need to differentiate test case verdict determination (the oracle) and system under test evaluation (the overall test result)

Test Case Oracle and System Oracle



- Compare test results with the defined test objective

  → individual test results (given by oracle and arbitrated wrt. test objective) are consolidated into overall test result

  → system oracle needed

- Checking test logs against the exit criteria

  → system oracle (like a test oracle) needs to decide about test campaign termination or continuation

- Test case arbiter → test (case) verdict evaluation scheme
- Test case oracle → test (case) verdict production
- System arbiter → test result evaluation scheme
- System oracle → test result production

Expressiveness of test case oracles

- Oracles
  - Need to be “weak or strong” as required
  - Should produce the test case verdict automatically based on an arbitration relating to the test objectives
  - Are potentially not just saying “yes/no”, but rather provide also hints/guidance what to test in addition in order to say yes/no

Oracle hierarchy

- Basic oracle
  → predefined fixed set of response accepting
  → often used in software testing and embedded systems testing, applicable for offline and online test result analysis

- Dynamic oracle
  → set of responses accepting that are parameterized with SUT response elements
  → smarter way of determining acceptable system responses, basically used to improve test efficiency; required if responses depend on stimuli
  → often used in protocol and service testing and in testing of distributed systems, applicable for offline and online test result analysis

- Interactive oracle
  → in addition, set of stimuli providing that are parameterized with SUT response elements, which are needed to complete test evaluation
  → same as above, but applicable for online test result analysis only

- Composite oracle
  → several (different) oracles (with potentially different arbitration) contributing to test result

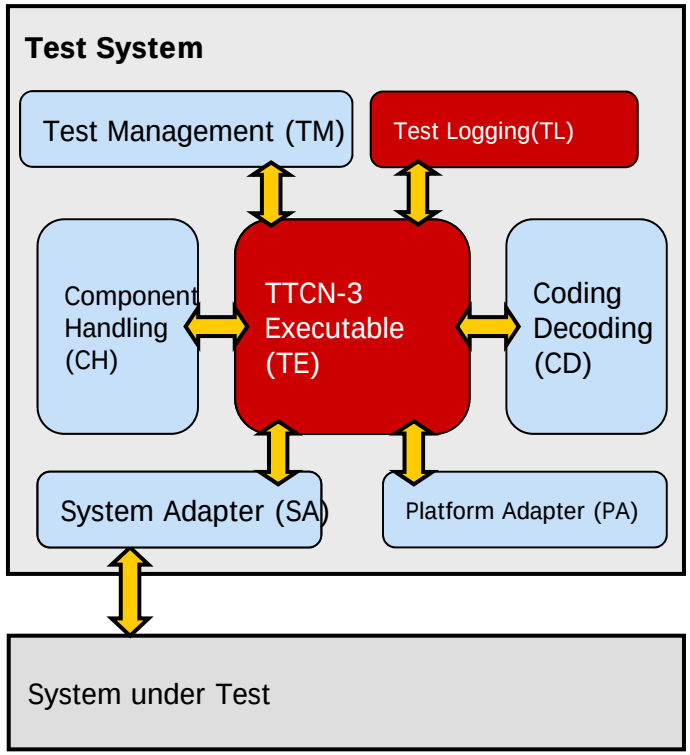
Oracle determination

- Oracles can be
  - Generated from models → model-based testing
  - Extracted from code → oracle mining
  - Specified → test modeling
- Oracles need to be
  - Validated → empirics
  - Verified → power, correctness, completeness

Test Execution with TTCN-3

- TTCN-3: Testing and Test Control Notation

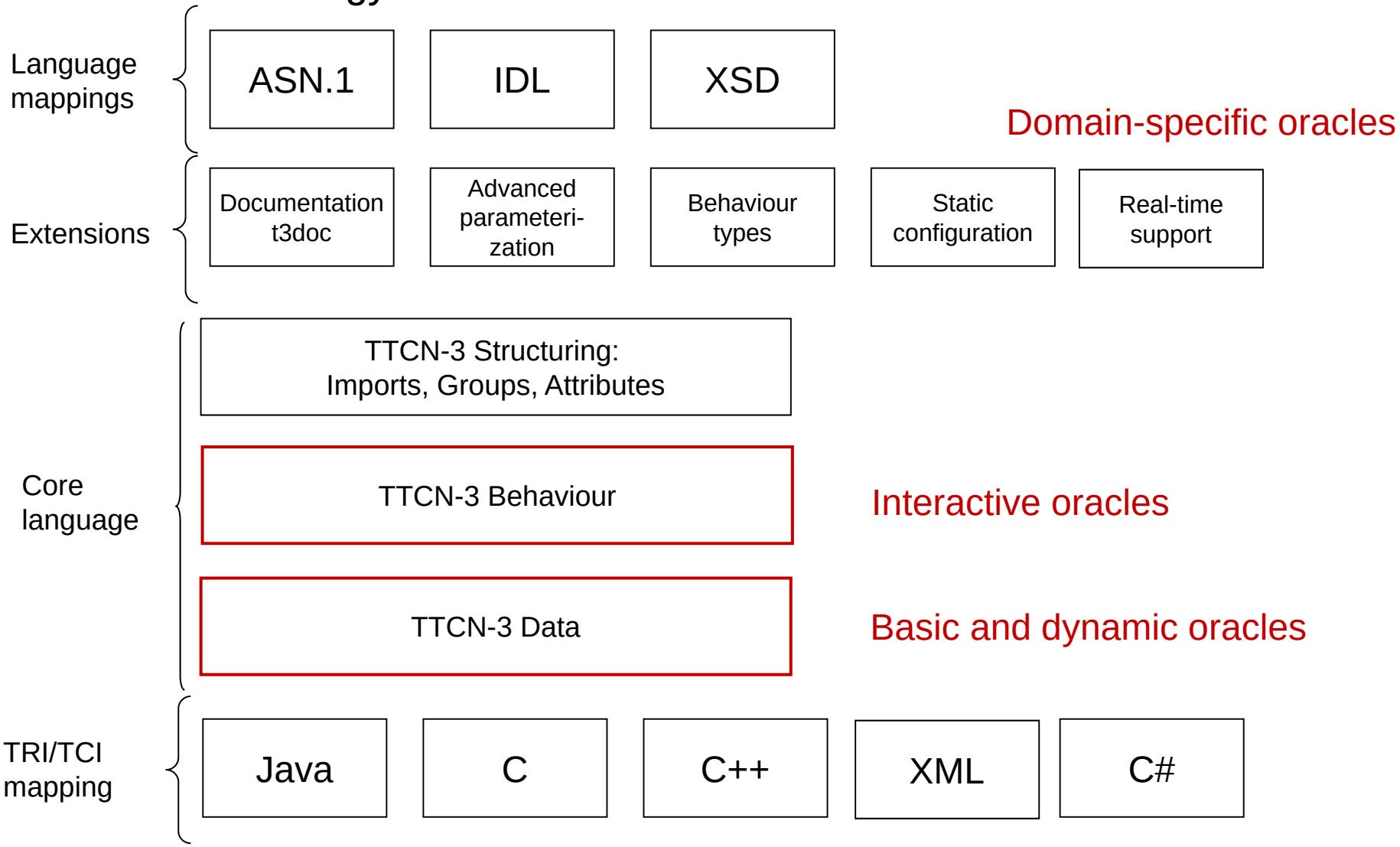
- Abstract test specification
  - Data templates allow structuring, parameterization, construction and reuse of test data
  - Matching mechanism are the main concept to define oracles
  - Interaction with SUT: at message-based and procedure-oriented ports
  - Test behavior: sequential, branching and recursive in a test component, parallel between test components
- Concrete test implementation
  - Adapter and codec
  - Logging interface



- Official web page: http://www.ttcn-3.org
- Standard: http://www.ttcn-3.org/StandardSuite.htm
- Tool: TTworkbench: http://www.testingtech.com/

Testing Technologies

TTCN-3 Technology Overview



TTCN-3 Templates

- Define test data representing stimuli to and responses from the SUT
  - Template specification
    - Type-based → messages
    - Operation-based → operation invocation, reply, exception
  - Explicit definition or inline definition
  - Global or local
  - Value sets
  - Parameterized value sets
  - Function-generating value sets

Basic TTCN-3 Matching Mechanisms

```
omit |
"(" { TemplateInstance [","] } ")" |
complement "(" { TemplateInstance [","] } ")" |
"?" |
"*" |
"(" ( ConstantExpression | -infinity ) ".." ( ConstantExpression | infinity ) ")" |
superset "(" { ConstantExpression [","] } ")" |
subset "(" { ConstantExpression [","] } ")" |
pattern Cstring
```


Oracle Specifications in TTCN-3

- Basic oracles

  Values:

```
template integer valueTempl:= 7;
```

  Value sets:

```
template integer valueSetTempl_1:= complement(7);
template integer valueSetTempl_2:= ?;
```

- Dynamic oracles

  Parameterization:

```
template integer valueTemplParam (integer p):= 2*p;
template integer valueSetTemplParam (integer p):= complement(2*p);
```

  Template reuse:

```
template integer TemplOfTemplates:=
                              (valueSetTempl_1, valueSetTemplParam(10))
```


Oracle Specifications in TTCN-3

- Interactive oracles

  Template computation together with control and alternative behaviors:

```
var integer x;
// any value, keeping value
p.receive(integer: ?) -> value x;
// send template depending on received value
if (x mod 2 == 0 ) { x:= 2*x } else { x:=0 }
p.send(x);
// different expectations depending on send value
alt {
    [] p.receive(4*x) {setverdict(pass)}
    [] p.receive(?) {setverdict(fail)}
}
```

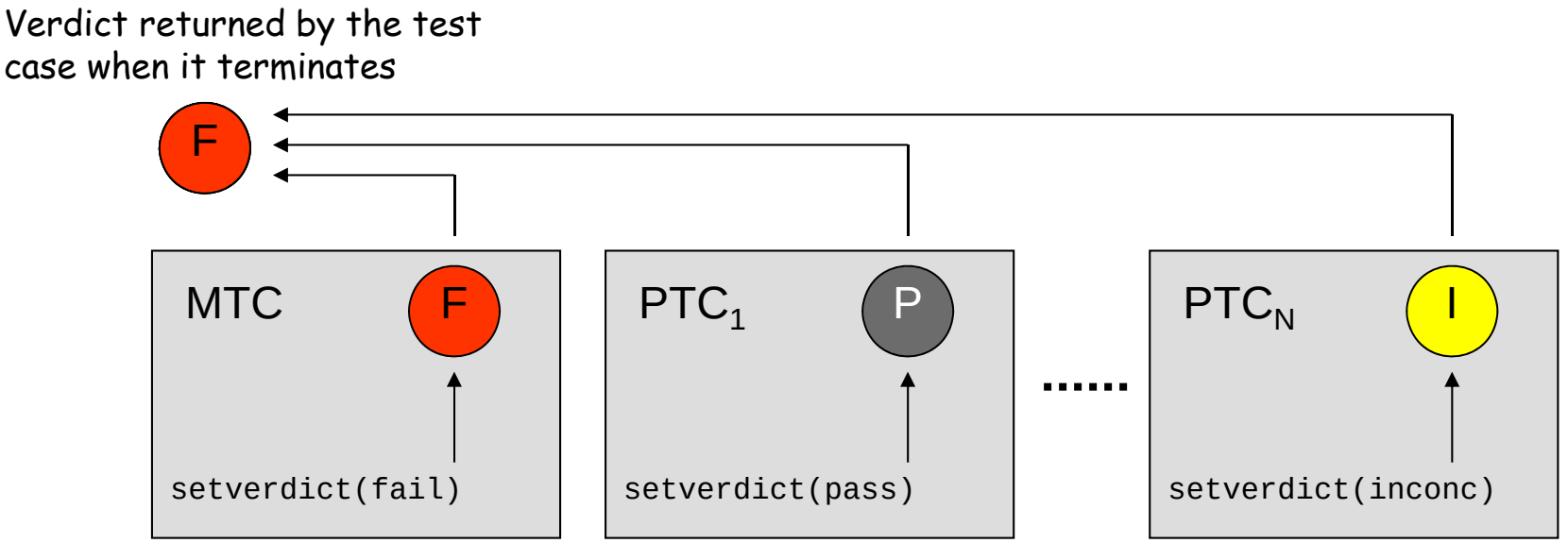

Since TTCN-3 v3.1.1: Template Computation

- First-order templates
  - Template variables → for computation
  - Template parameters → for reuse
  - Template returning functions → for computation structuring
- Template Characterization

```
template "(" ( omit | present | value ) ")" Type
```

Test Verdict Handling

- Verdict type with ordered verdict values:
  `none < pass < inconc < fail < error`
- Each test component has its own local verdict, which can be set (`setverdict`) and read (`getverdict`)
- Predefined functional/conformance verdict computation, verdict arbitration possible by own computations
- A test case returns the test case verdict



Test Logging

- Used to support the tracing of test runs and the evaluation of the test results
- Test execution interface for automated logging: TLI
  - XML based
  - Can log all test events
  - Can be filtered
- Log statements for test case specific logging
  - literal values
  - templates and variables
  - component, port and timer states
  - ...

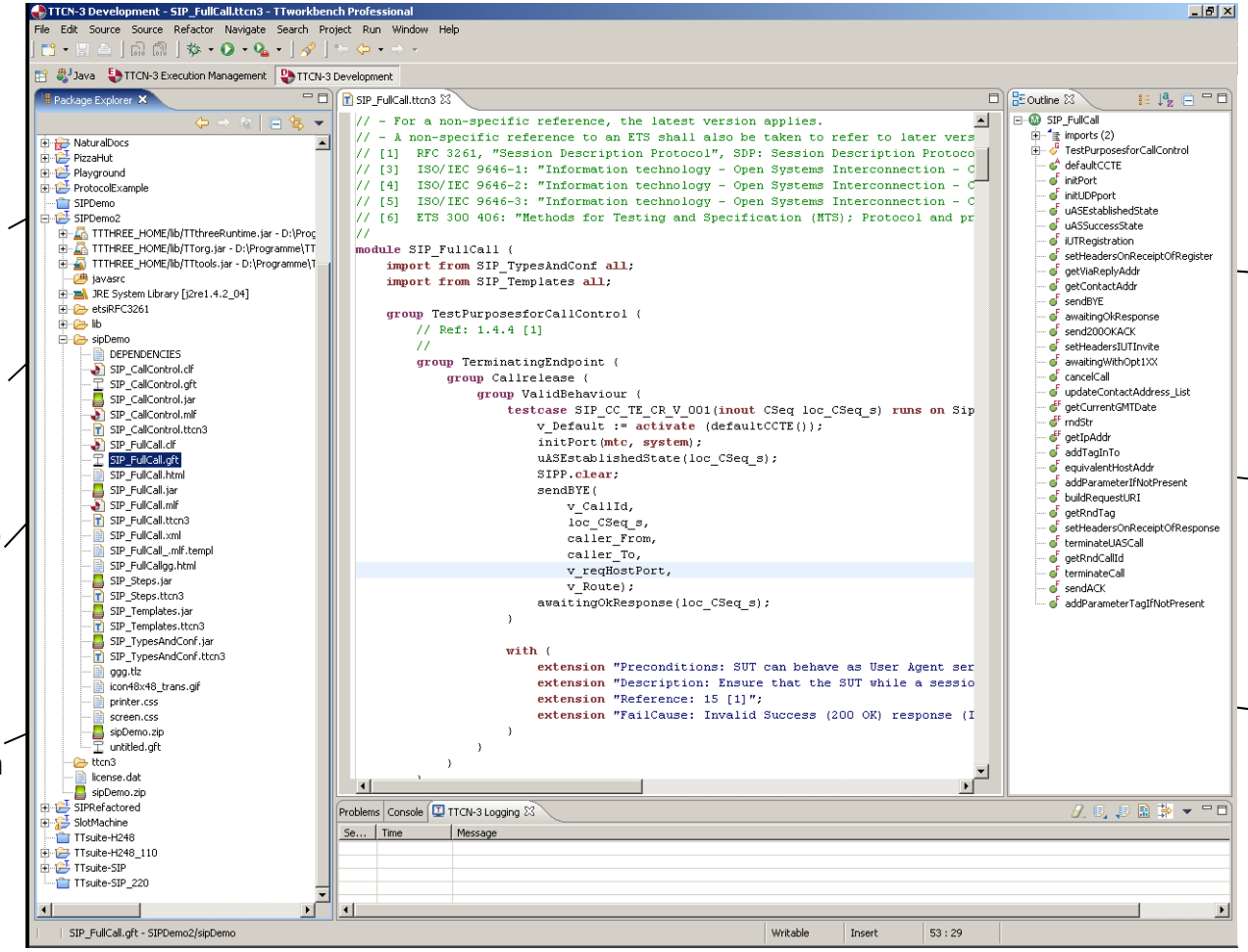
TTworkbench – An Impression of TTCN-3 Tooling

Developers Perspective for Modification

Test Execution

Test Campaign Designer (Test Automation)

Test Parametrization



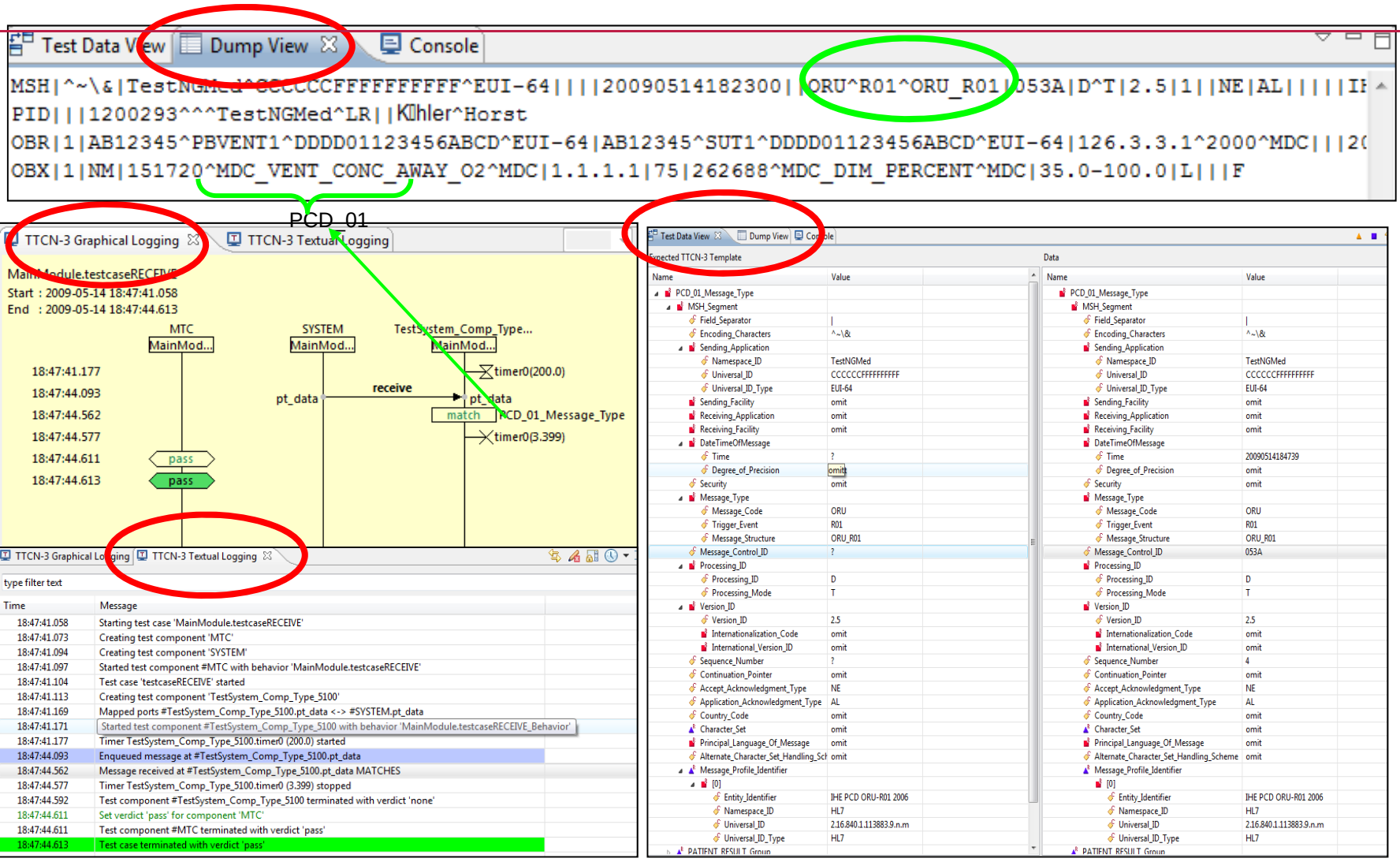
Result Analyzer

Test Report

Online Logging, Filter, Reporting

Fraunhofer FOKUS

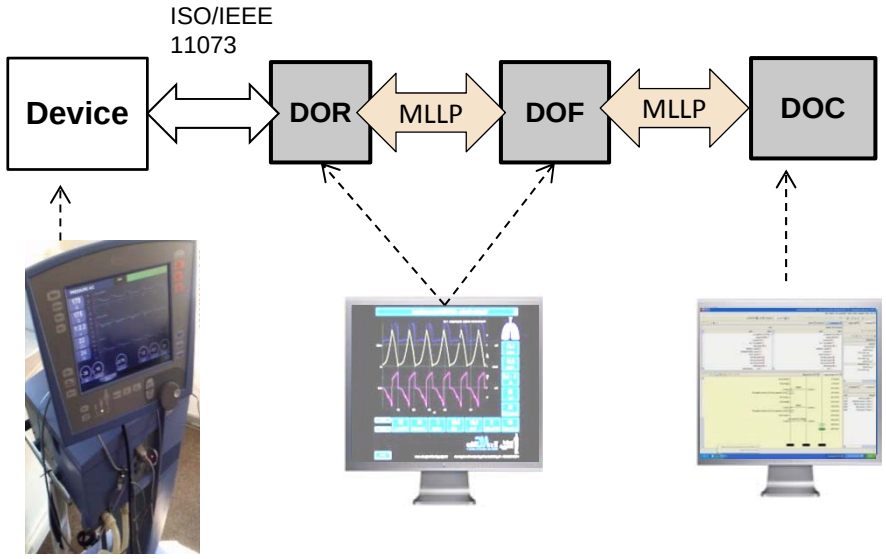
Automated Oracle and Evaluation Support



Example: IHE/HL7 Testing

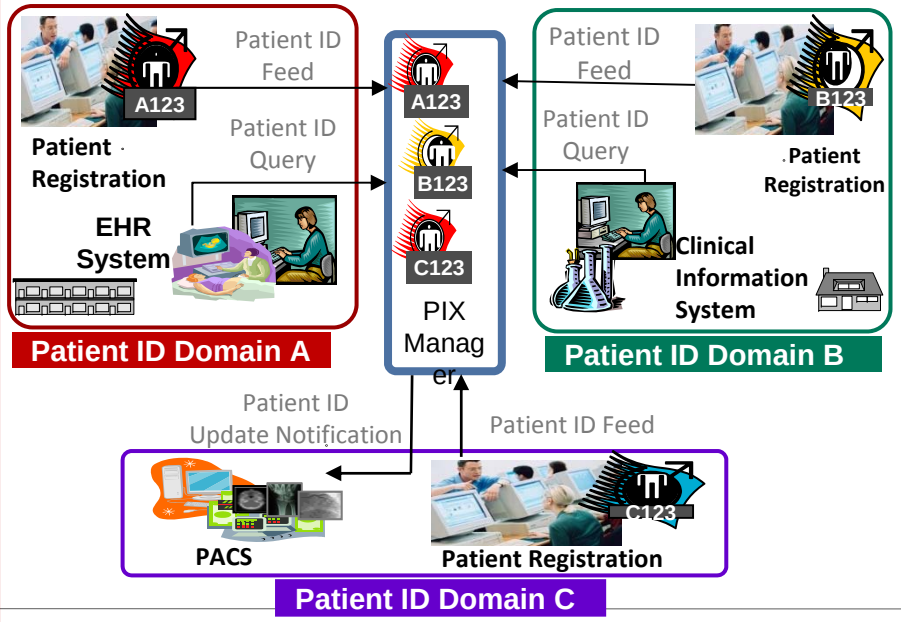
IHE Patient Care Device (PCD)
Device Enterprise Communication (DEC)

- **ReTeMes** (EU) / **TestNGMed** (DE) research project
  - Dec.2007- Sept. 2009
  - TUB, sepp.med, Applied Biosignals, UPB, InfoWorld
  - Goal: A test methodology (incl. IOT) based on TTCN-3 for automated testing of HL7 based medical systems

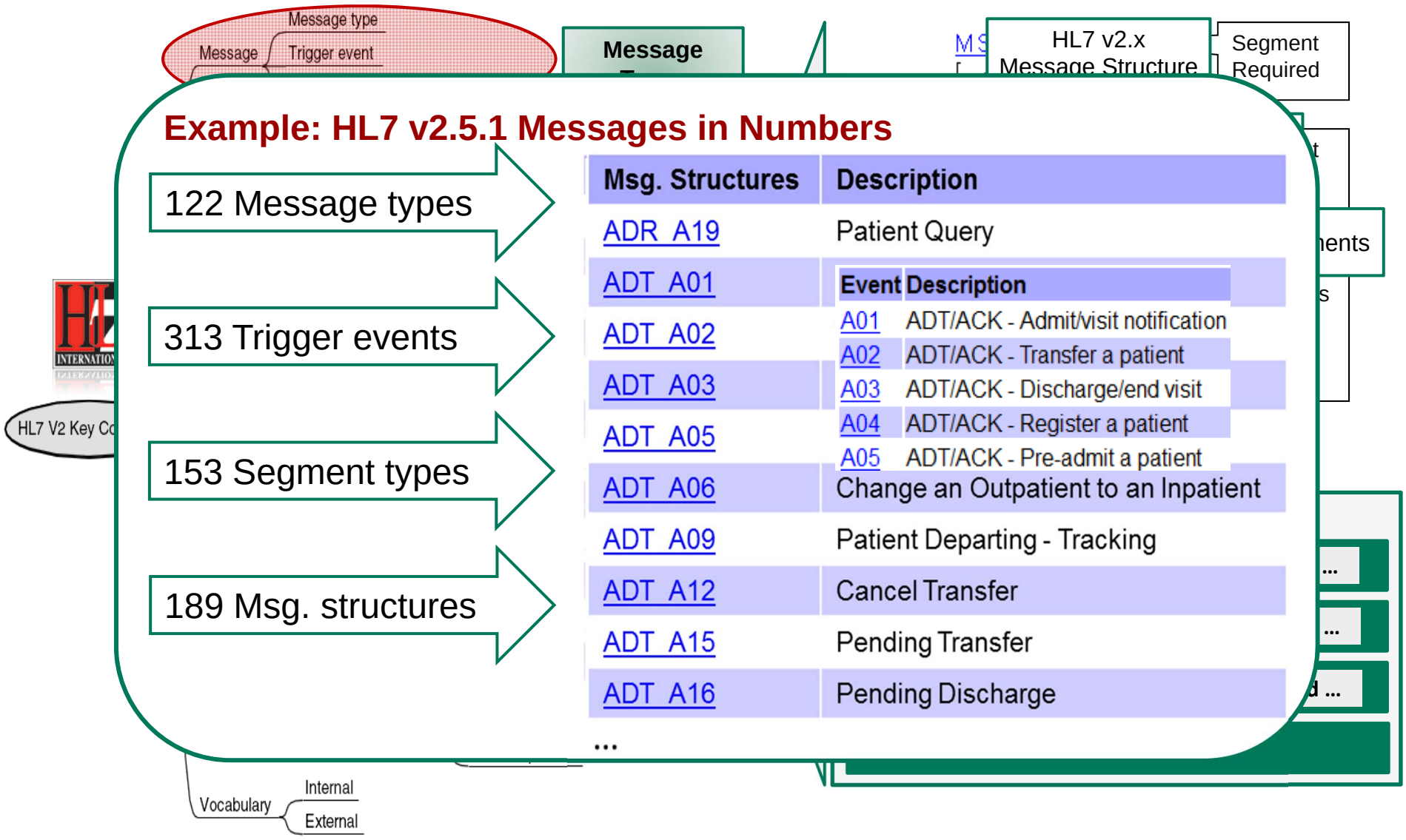


IHE IT Infrastructure (ITI)
Patient Identifier Cross-Referencing (PIX)

- **PIX IOT Connectathon Test Suite**
  - Connectathon 2010, 2011, 2012
  - Fraunhofer FOKUS, ETSI
  - Goal: demonstrate the use of TTCN-3 technology for interoperability of HISs compliance with IHE profiles
  - Results contributed by ETSI to the HITCH research project

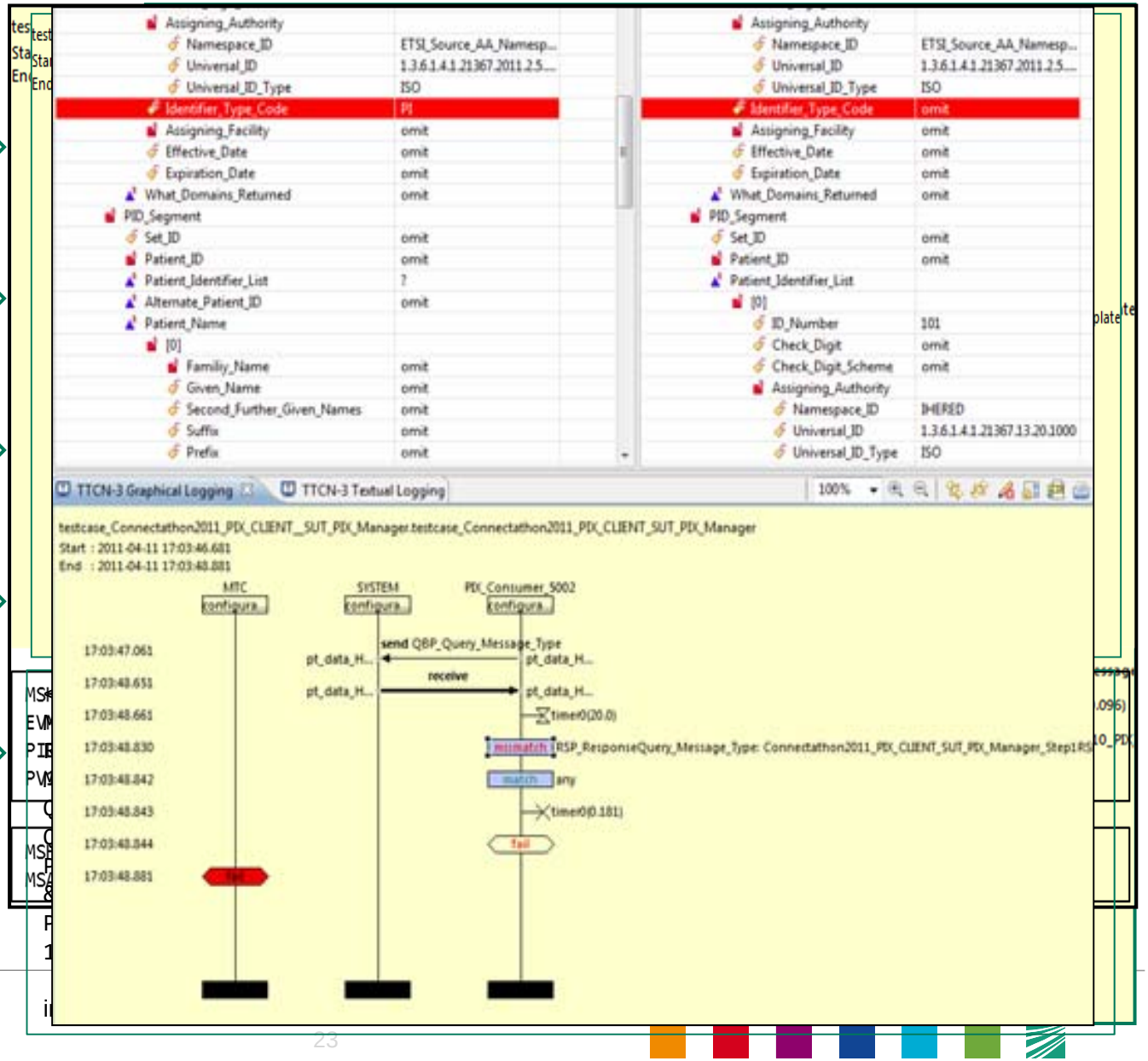


HL7 v2.x Messaging Standard

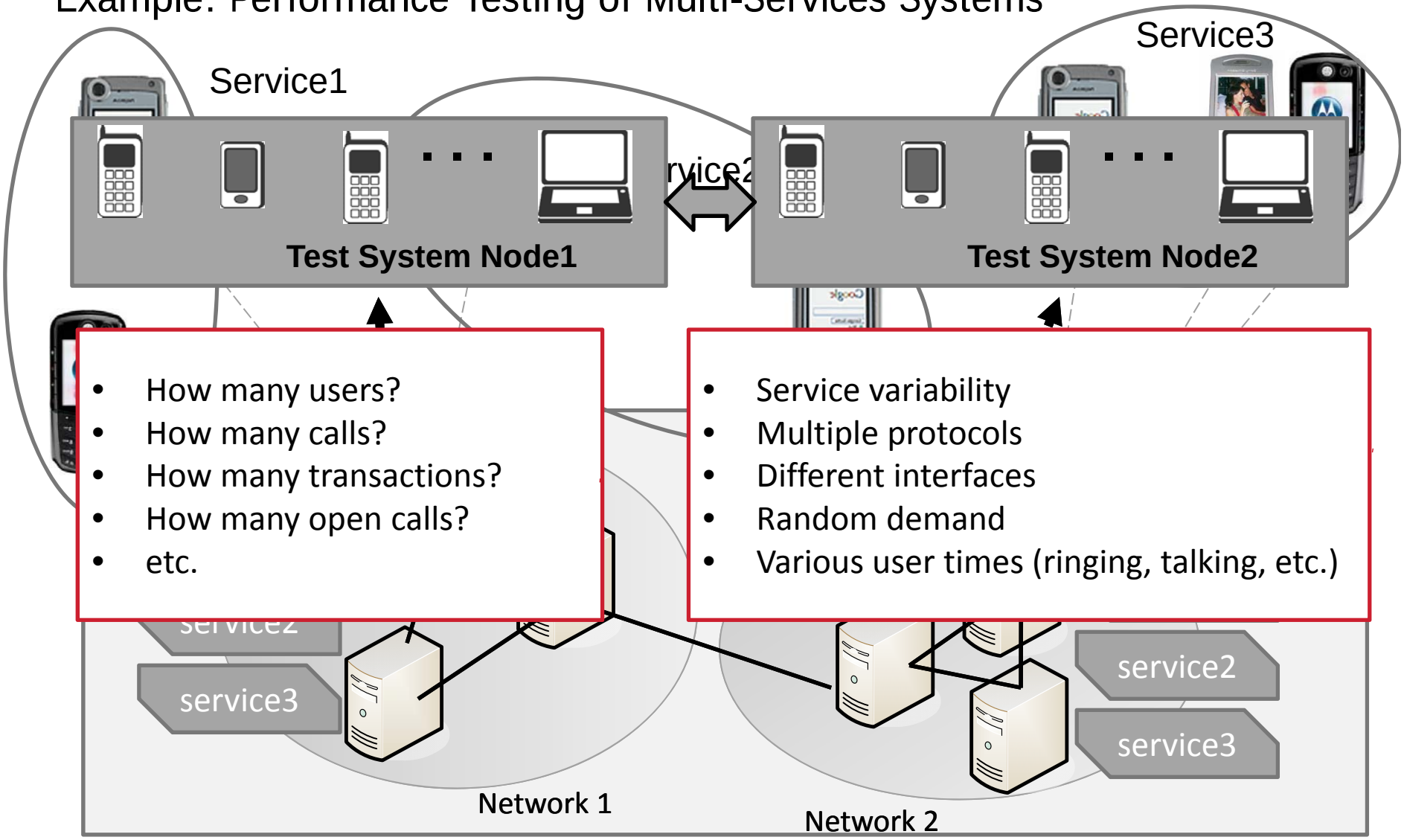


Possible IOP Problems

- interaction scenario
- message type checking
- fields conditionality
- message content
- semantic correlations



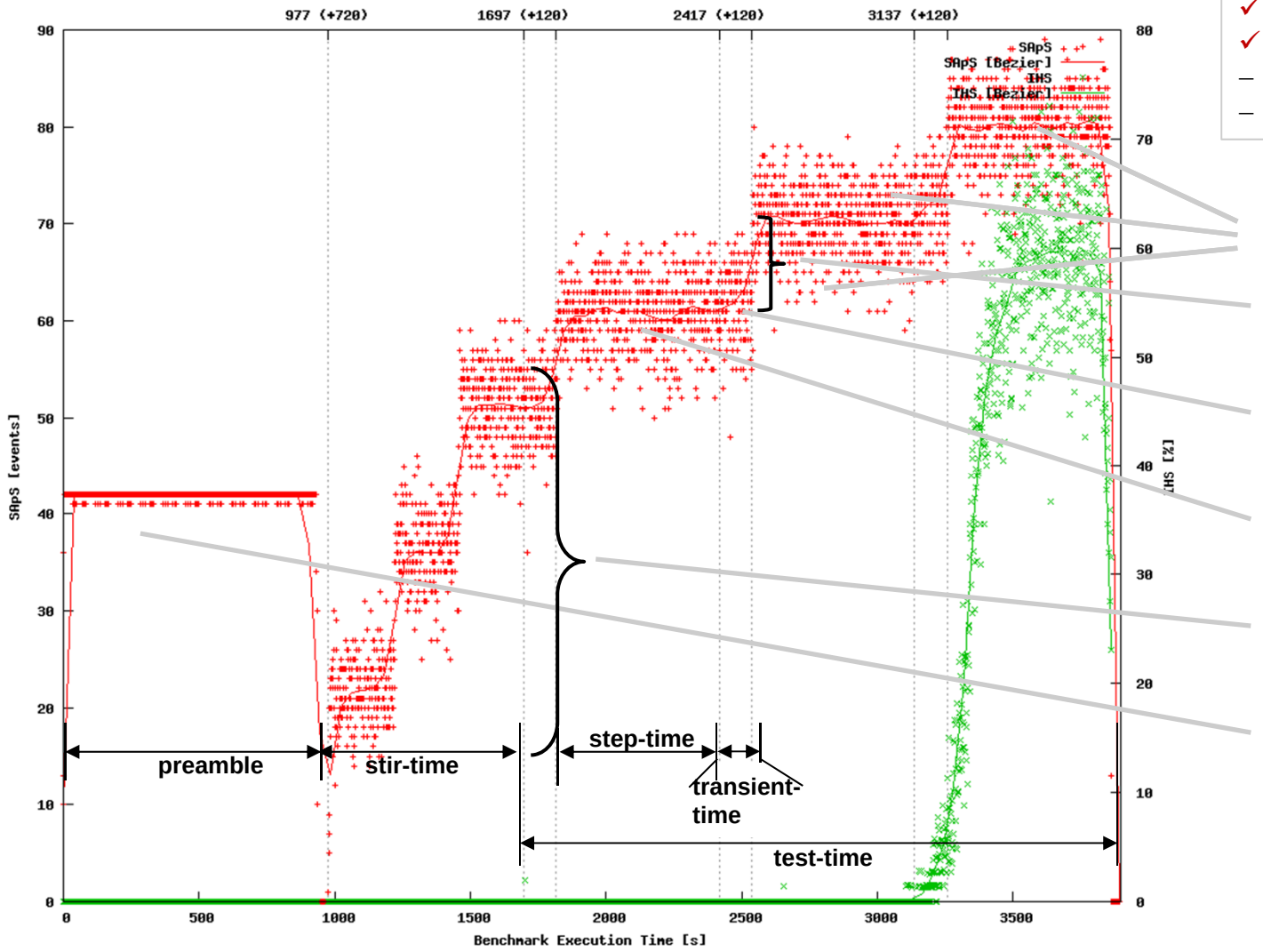
Example: Performance Testing of Multi-Services Systems



- How many users?
- How many calls?
- How many transactions?
- How many open calls?
- etc.

- Service variability
- Multiple protocols
- Different interfaces
- Random demand
- Various user times (ringing, talking, etc.)

Traffic Time Profile



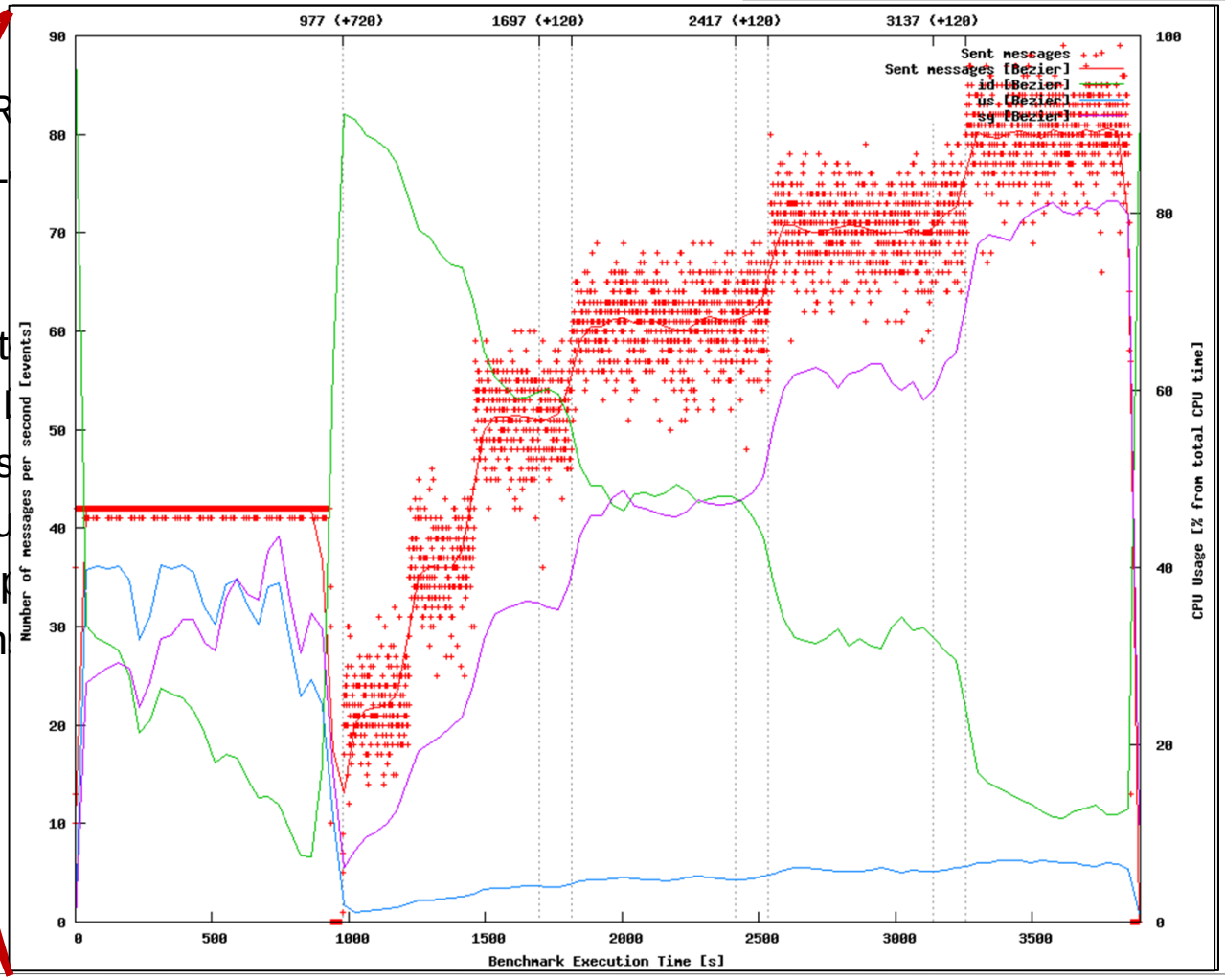
- ✓Use cases
- ✓ User State Machine
- ✓ Design Objectives
- ✓ Traffic Set
- ✓ Traffic Time Profile
  - Design Objective Capacity
  - Metrics

- Steps Number
- Increase Rate
- Transient Time
- Initial Load
- Stir Steps
- Preamble Rate

Performance Metrics

- ✓Use cases
- ✓ User State Machine
- ✓ Design Objectives
- ✓ Traffic Set
- ✓ Traffic Time Profile
- ✓ Design Objective Capacity
- ✓ Metrics

- Per scenario
  - TRT – Transaction R
  - TL – Transmission L
- Global
  - SAPS – Scenario Att
  - IHS% – Percent of I
  - RETR – Retransmiss
  - SIMS – Simultaneou
  - CPU – CPU consump
  - MEM – Memory con



Fraunhofer FOKUS

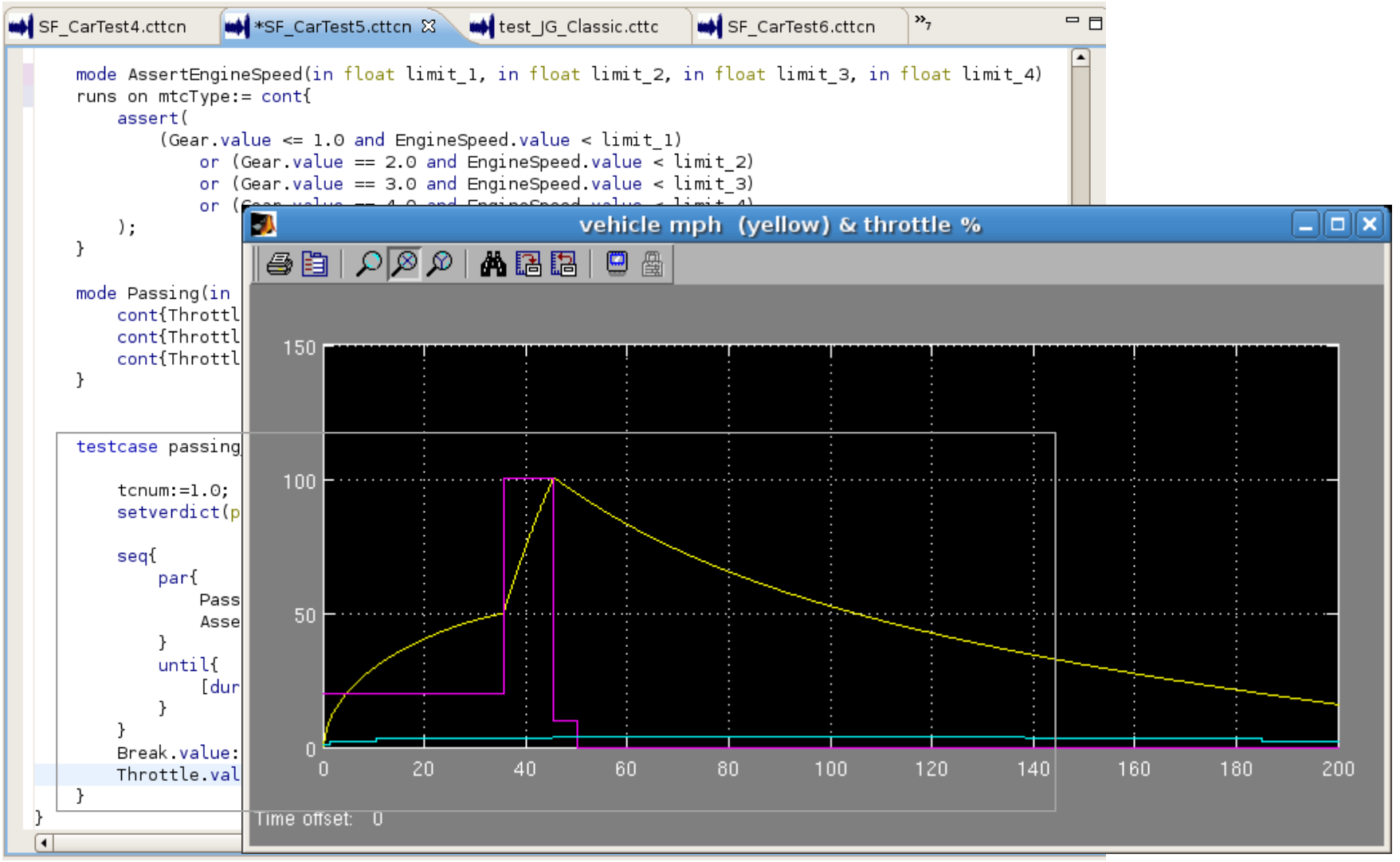
Example: ECU Testing

- Extensions for hybrid systems
  - real time systems (RT-TTCN-3)
  - continuous systems (Continuous TTCN-3)

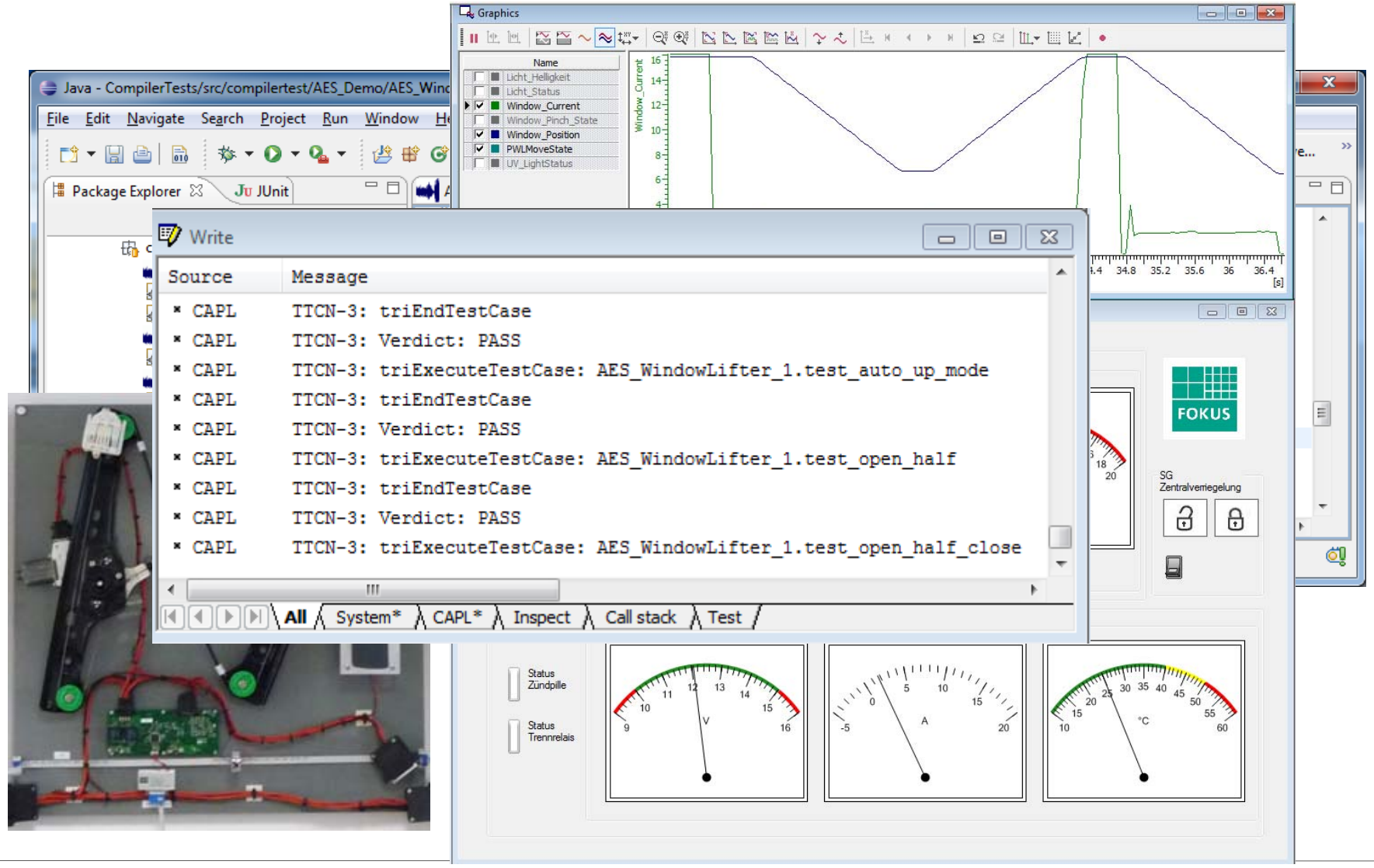
- RT-TTCN-3 Concepts
  - **clock** as a common basis for time measurement.
  - **timestamp** redirection for exact time measurement of message interaction.

- Continuous TTCN-3 Concepts
  - **sampled clock** as a common basis for discretization and stream definitions.
  - sampled **streams** that provide a data structure to define, access and manipulate discretized signal values and their history in time.
  - **hybrid automata** that provides a control flow structure to enable and control the simultaneous stimulation and evaluation of **stream ports**.

TTCN-3 Embedded Sample: Vehicle Passing



TTCN-3 and Vector CanOE



Summary

- Need for
  - Dynamic, interactive and composite oracles
  - Arbitration
  - Test case and system-level considerations
- Presented TTCN-3
  - Supports different forms of templates and arbitration
  - Automates the oracle
  - Supports the test result evaluation
- An aside: UTP
  - Addresses issues differently