Application of a LOTOS based Test Environment on AAL5

J. Burmeister
A. Rennoch

GMD FOKUS
Research Institute for Open Communikation Systems

email: burmeister@fokus.gmd.de

Contents

LOTOS based Test Environment: Scope

- to generate Executable Test Cases from ATCs
  - based on a formal test scenario specification (including ADT)
  - formalized interface descriptions,
  - coordination support of distributed interfaces
- to run protocol and service tests
- to visualize the test execution,
  - post-animation of the test steps,
  - verdict analysis

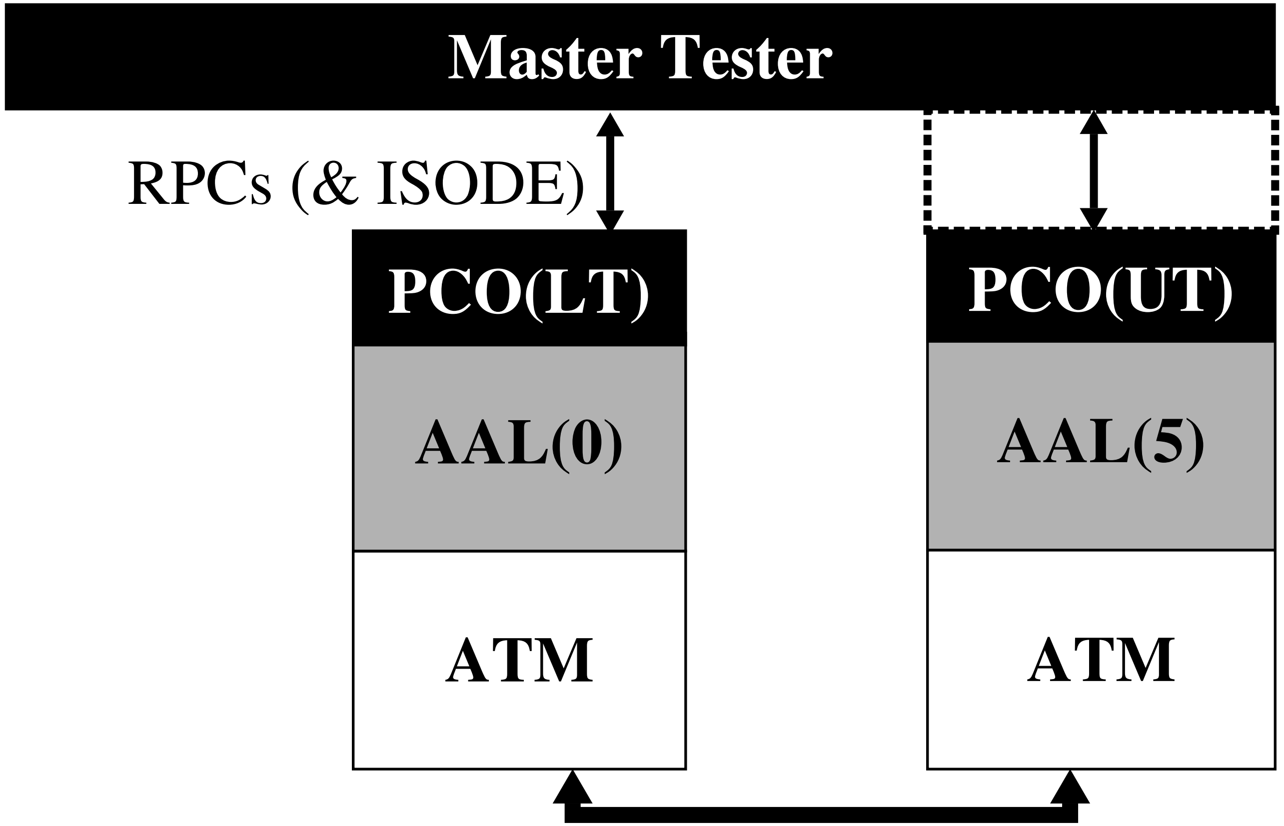
Requirements on Test Notations

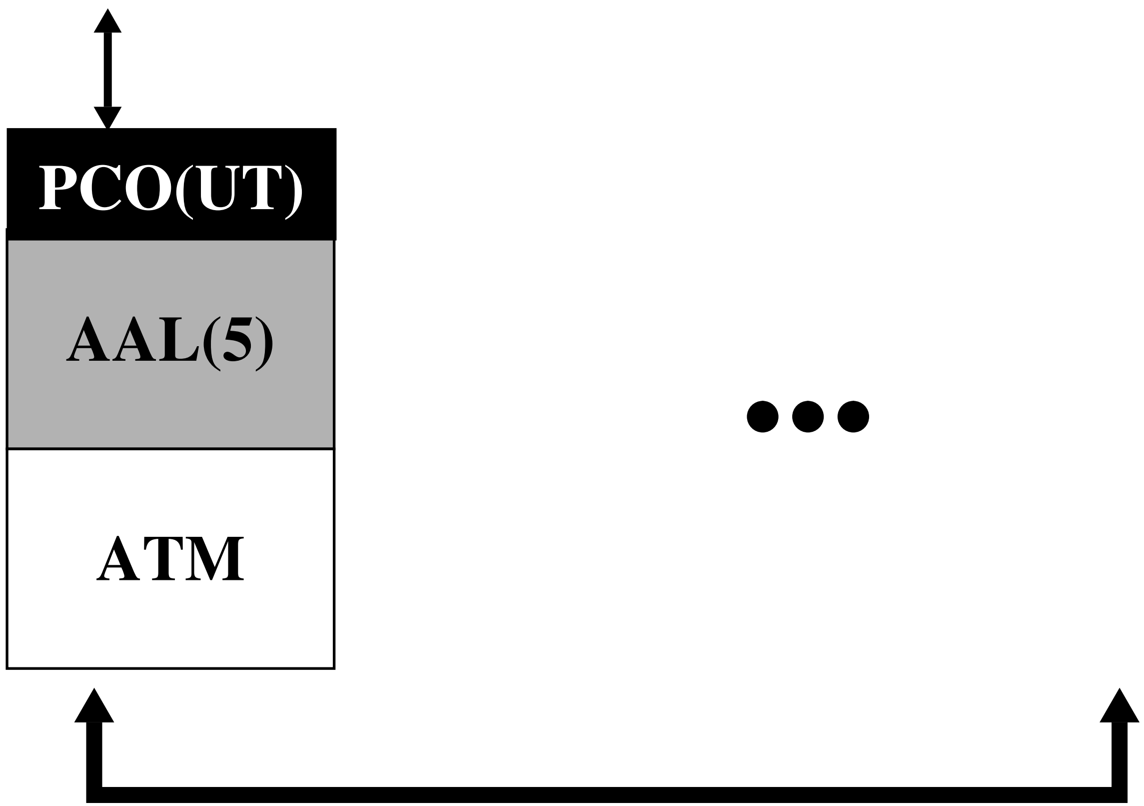
- Test Suite:
  - parameterization
  - structuring
- Test Event:
  - send/receive
  - data presentation
  - constraints
- Test Sequence:
  - alternative
  - otherwise
  - verdict assignment
- Timer
- Relation to the IUT Specification

$\Rightarrow$ **ASN.1 for the data types part**

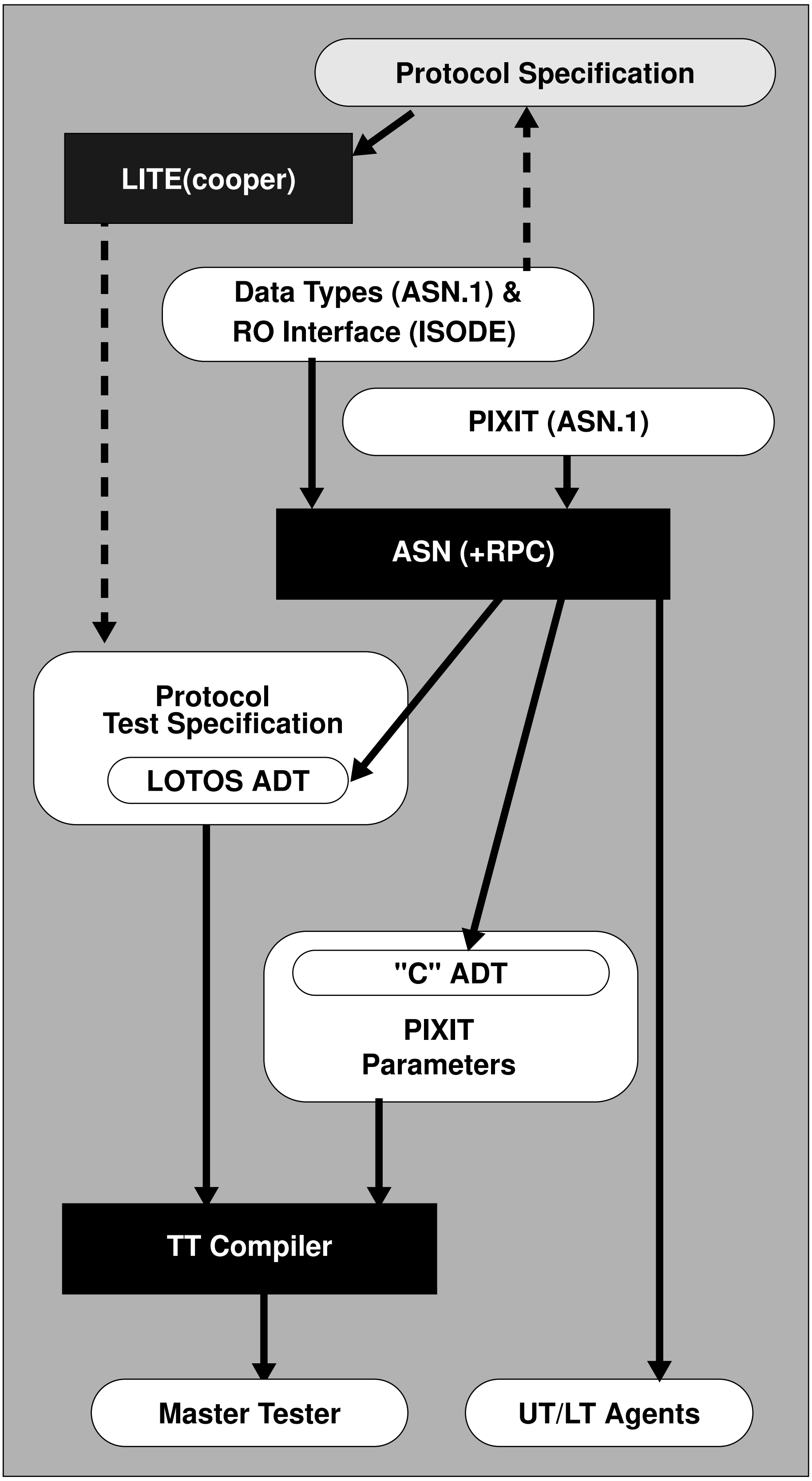
LOTOS for the behavioural Part

Technical Approach



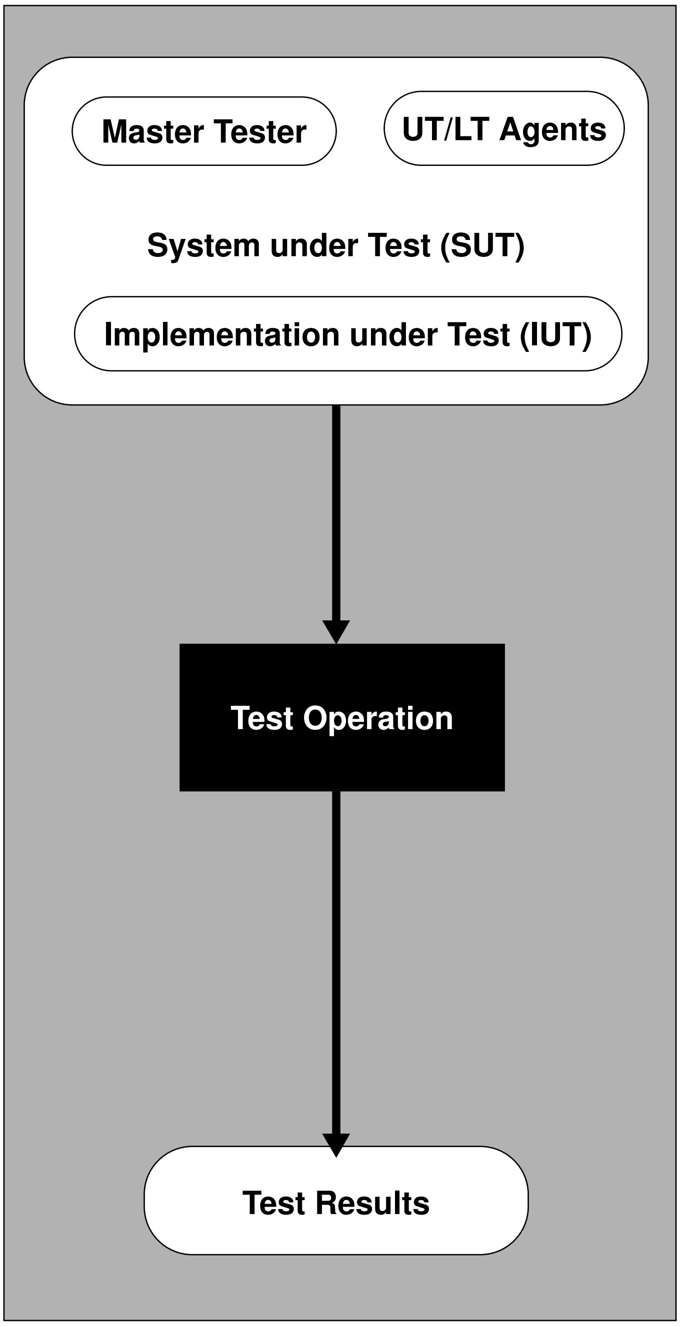

PCO(UT)
AAL(5)
ATM

Preparation Phase

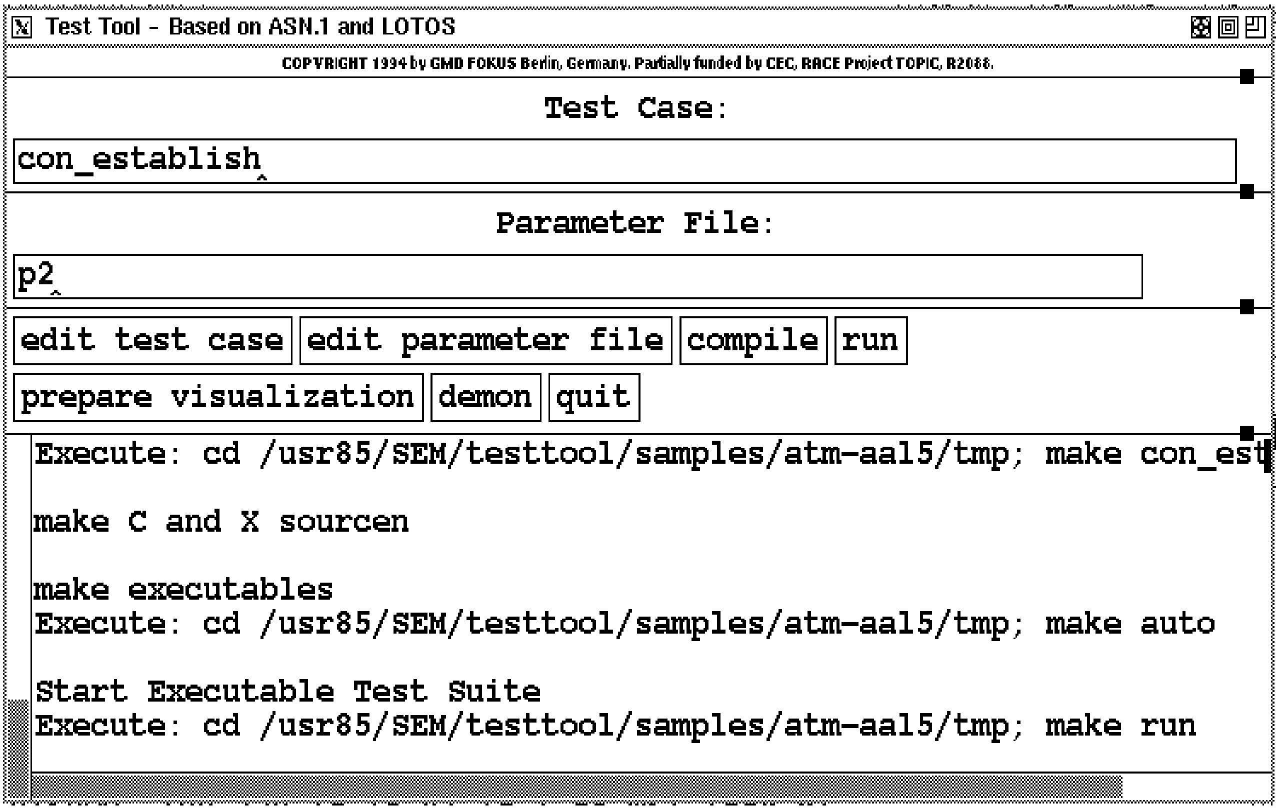


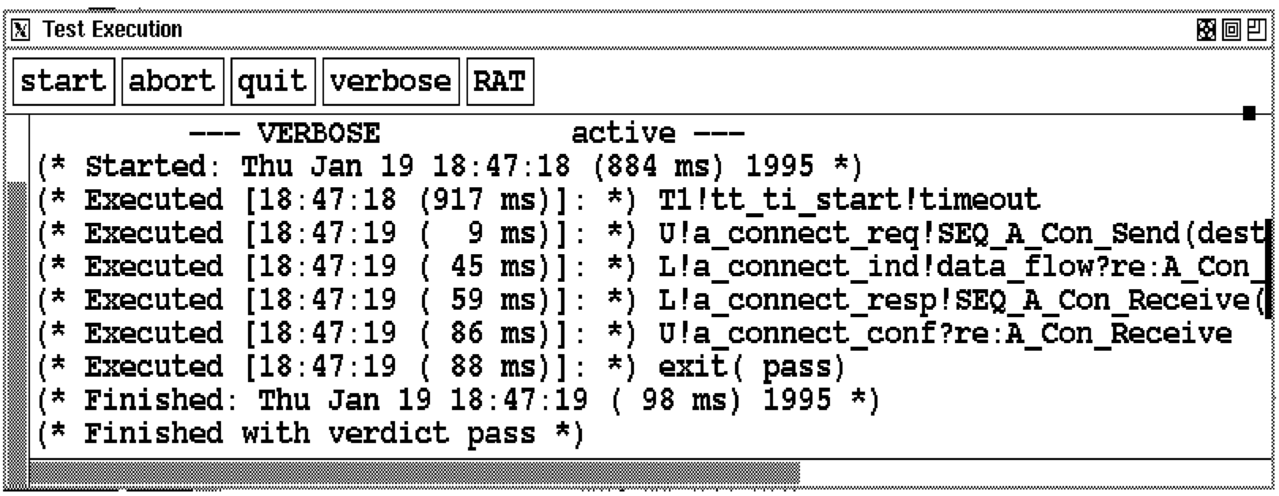
```
specification test_connect_establish [T,L,U]
                    (remote    :CHARACTERSTRING,
                     any_qos   ,
                     sel_qos   :DL_qos,
                     data_flow :DL_dataflow,
                     timeout   :INTEGER)
                    : exit(verdict)
library Verdict endlib
library Timer endlib
library tt__primitive endlib
#include "../pixit/datatypes.adt"
behaviour
  test_con_est [T,L,U]
   (remote,any_qos,sel_qos,data_flow,timeout)
  where
    process test_con_est [T,L,U]
                          (remote    :CHARACTERSTRING,
                           any_qos   :DL_qos,
                           sel_qos   :DL_qos,
                           data_flow :DL_dataflow,
                           timeout   :INTEGER)
                          : exit(verdict) :=
   T ! tt_ti_start ! timeout;
      ( U ! a_connect_req
          ! SEQ_A_Con_Send(  dest (remote),
                             qos (any_qos),
                             dataflow (data_flow));
          L ! a_connect_ind ? re: A_Con_Send;
           L  ! a_connect_resp
              ! SEQ_A_Con_Receive(qos (sel_qos),
                                  dataflow (data_flow));
              U ! a_connect_conf ? re: A_Con_Receive;
              exit(pass)
      [>
        T ! tt_ti_expired  ? x:BOOLEAN ;
        exit(fail)
      )
    endproc
endspec
```

Execution Phase

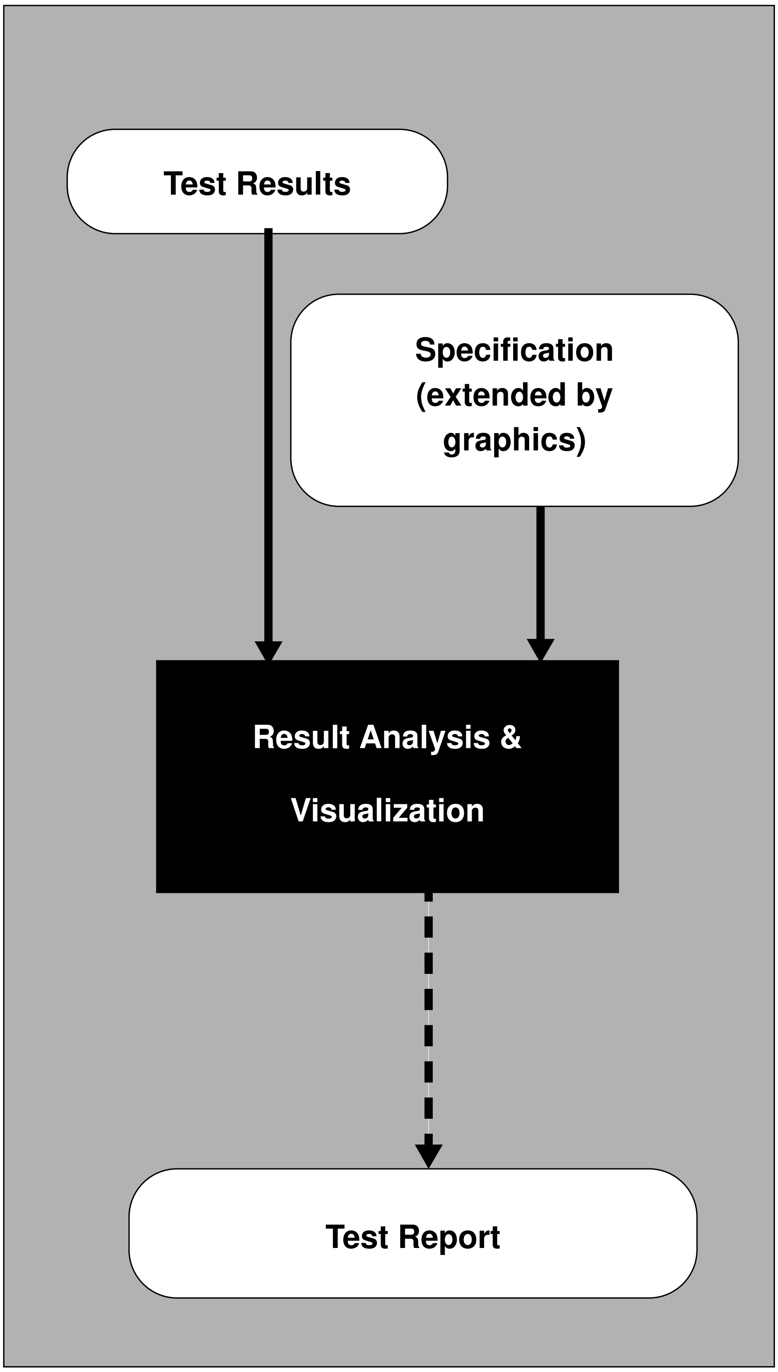


User Interface



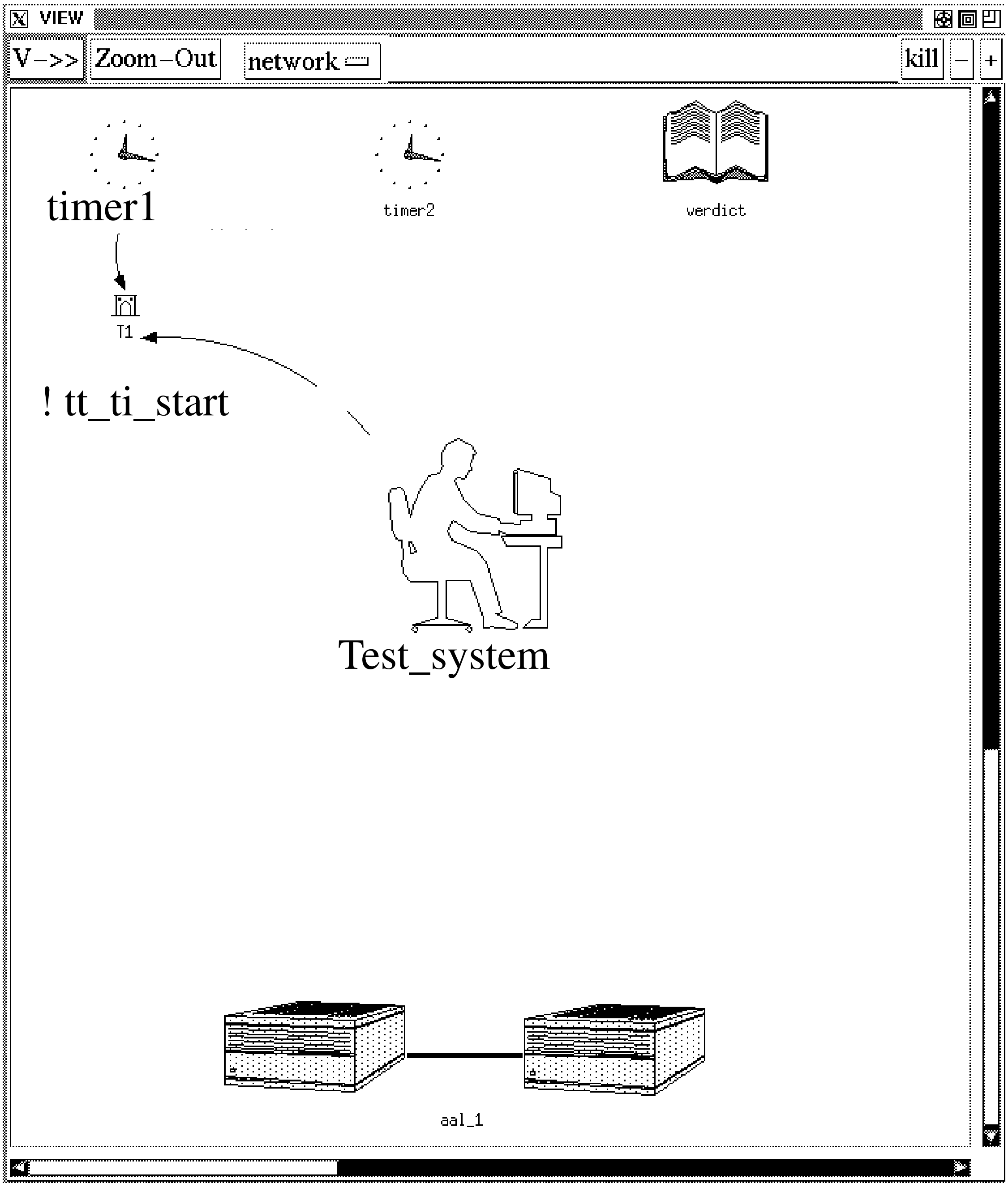


Analysis Phase

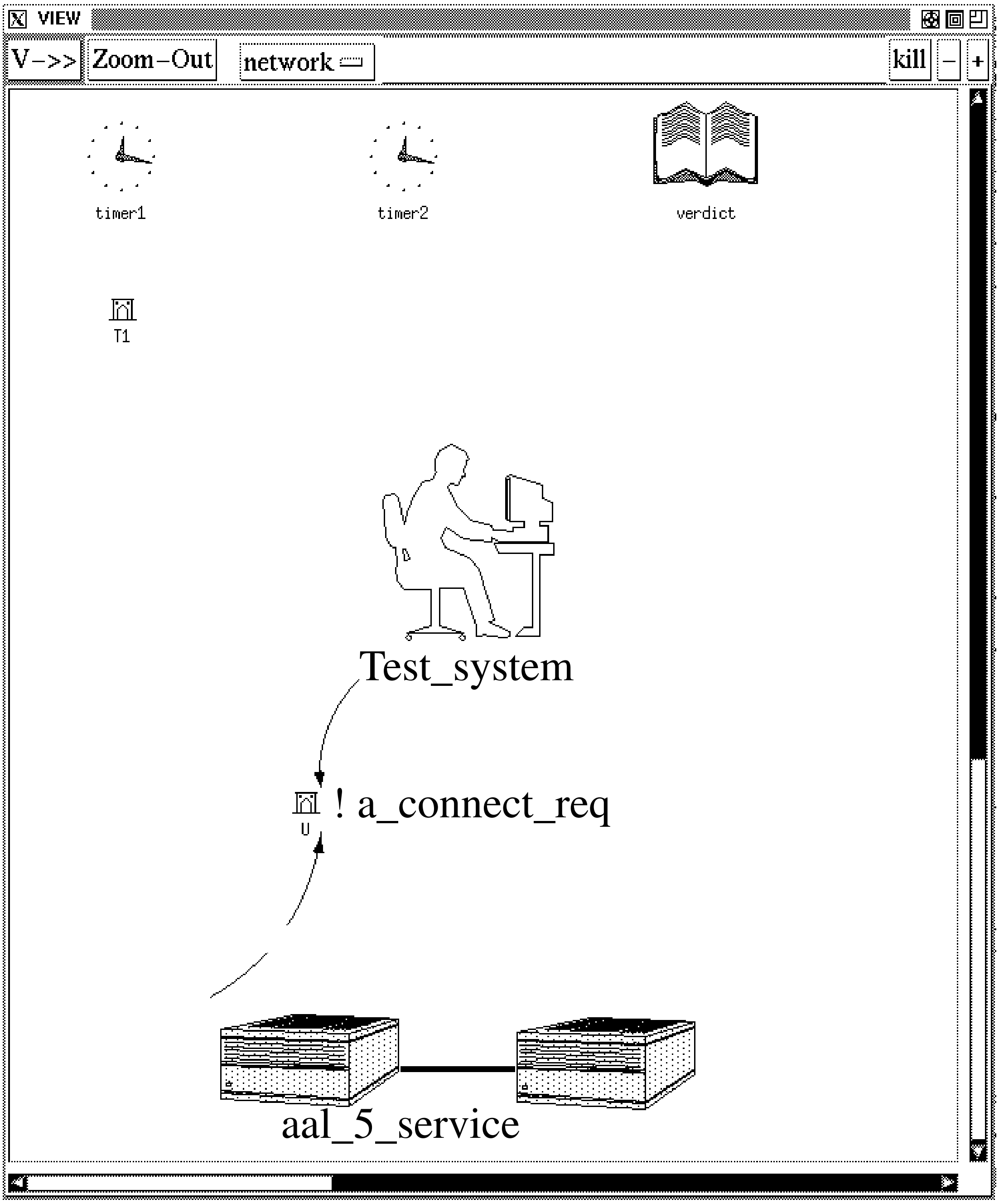


```
specification aal_env_spec[U(*# region 2 #*),.,
                           T2(*# region 5 #*),
                           info (*# region 5 #*) ]:noexit
   (*# user_defined_icons
   region_definitions
   GRID R3 50  665 8 8 1.0;  ... # timer & verdict
   GRID R7 140  30 8 8 2.0;      # aal
   #*)

   type  number is ...   endtype
   type primitives is
     sorts primitive
     opns
        a_connect_req, ... , a_close_req,
        pass, ..., tt_ti_start, tt_ti_expired :-> primitive
   endtype

behavior
 test_system[U,L,T1,T2,info]
  ||
 (aal[U,L](1) ||| timer1[T1] ||| timer2[T2] ||| verdict[info])
  where
     process aal[U,L](PN:number):noexit:=
       (*# region 7 #*)
      (  U ! 1 ! a_connect_req ; (* The AAL5 Specification *) ... )
    endproc

   process test_system[U,L,T1,T2,info]:noexit:=
       (*# region 1 #*)
#include "auto.lot"
       (* Integration of the observed trace expressed in LOTOS *)
     stop
   endproc

   (* Specification of the environment: *)
   process verdict [info]:noexit:=
          (*# region 3 #*)
        info ! 1 ! pass ; stop
     [] info ! 1 ! fail ; stop
     [] info ! 1 ! inconclusive ; stop
   endproc
   process timer1[T1]:noexit:=
          (*# region 3 #*)
        T1 ! 1 ! tt_ti_start ; timer1[T1]
     [] T1 ! 1 ! tt_ti_expired ; timer1[T1]
   endproc
   process timer2[T2]:noexit:= ... endproc
endspec
```

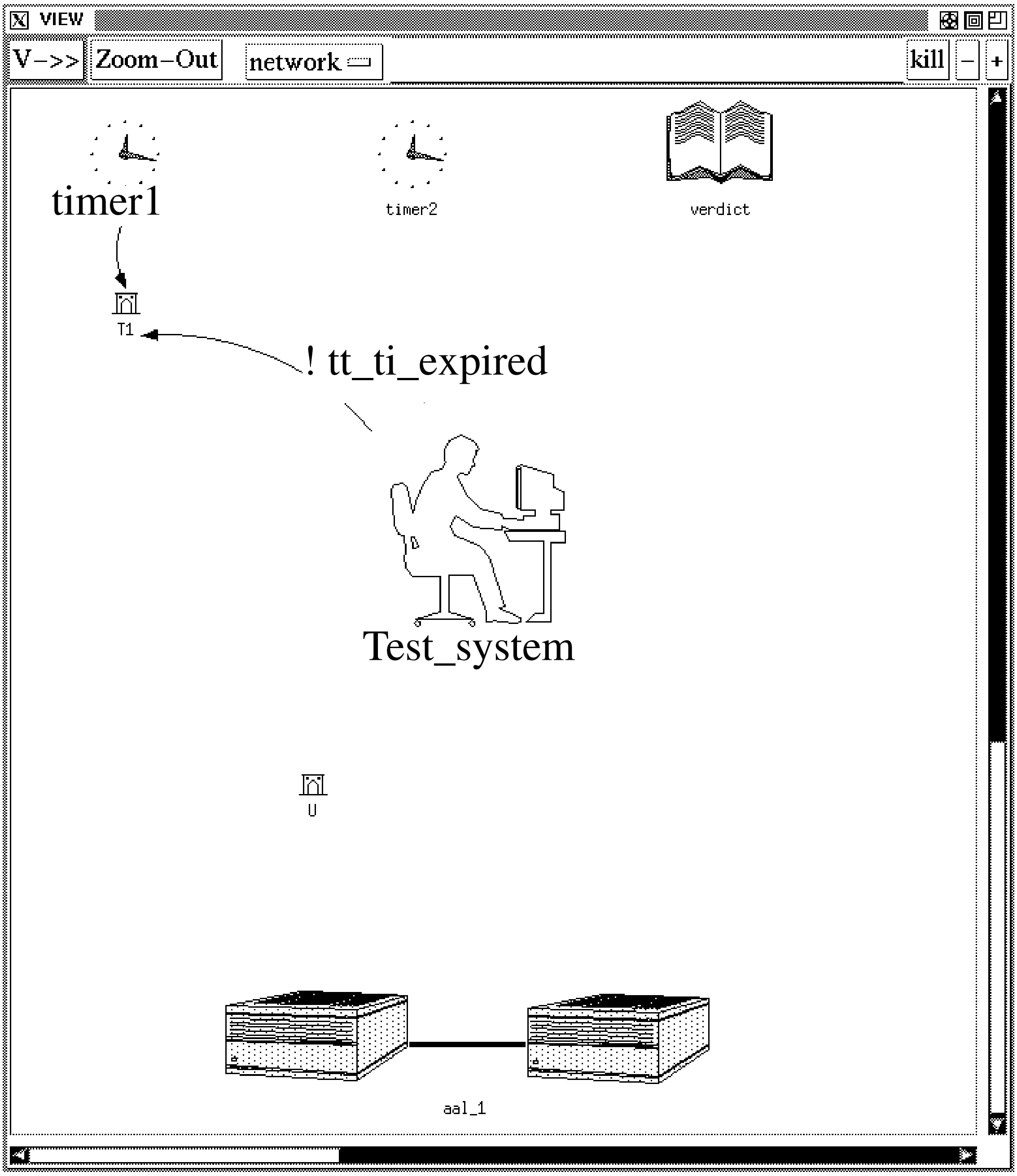
Visualization Demo (1)



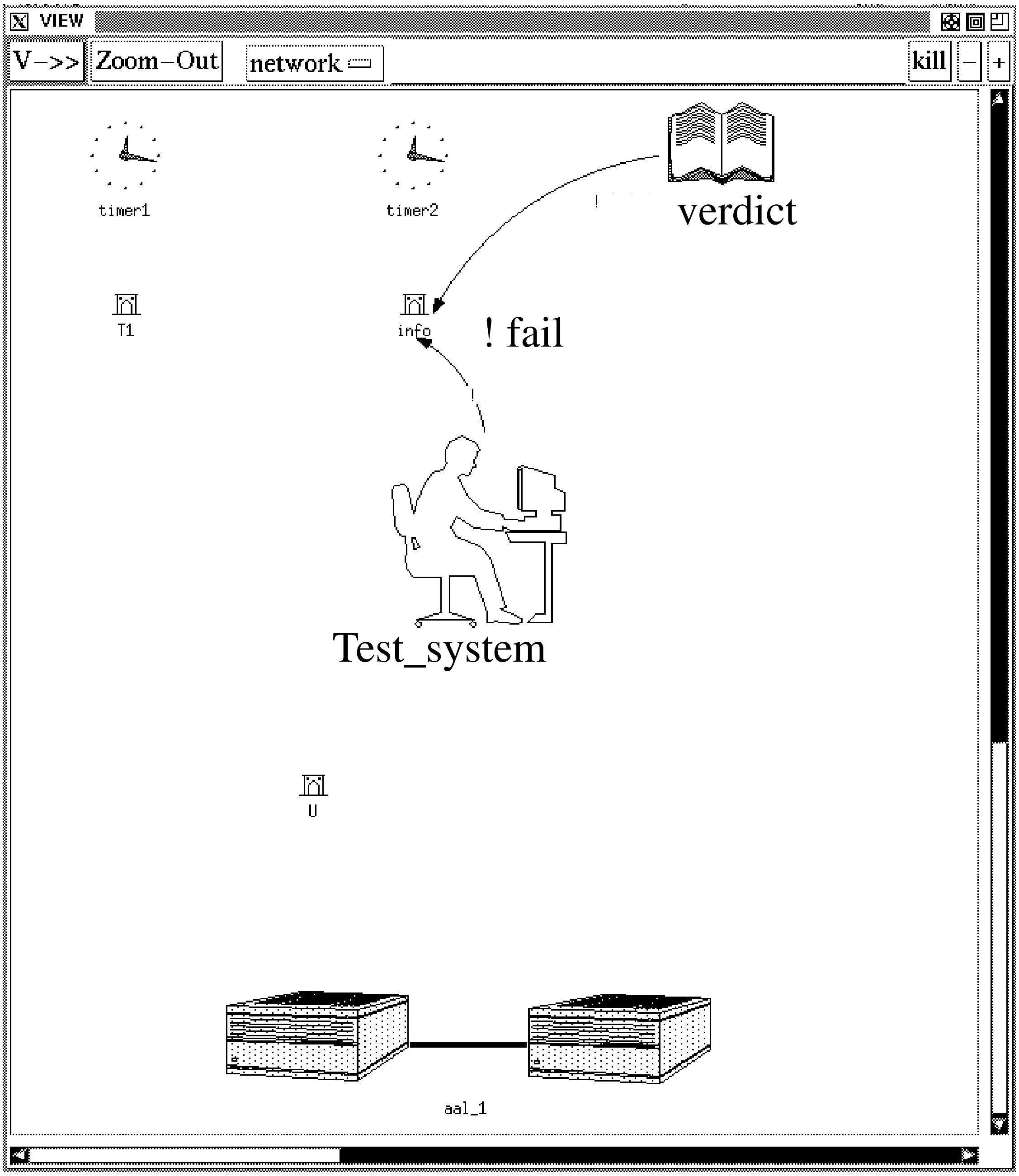
Visualization Demo (2)



Visualization Demo (3)



Visualization Demo (4)



Conclusions

- Close Relationship between IUT Specification and Test Cases
- Consistency of Data Types
- Debugging Facilities
  - Composition of Specification, Test Traces, and Test Environment
- Universal Testing Environment
  - Extension to Multi-Point Testing

Future Research

- Time extended Test Cases
- Protocol Performance
- Quality of Service