10th Dutch Testing Day
Leiden, October 8 - 2004

Formal Testing of Smart Cards

Lars Frantzen
lf@cs.kun.nl
www.cs.kun.nl/~lf/

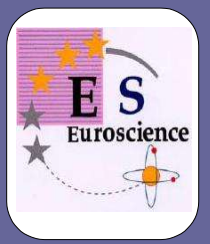
Radboud University Nijmegen

The Group

- Arjen van Weelden <arjenw>
- Martijn Oostdijk <martijno>
- Pieter Koopman <pieter>
- Jan Tretmans <tretmans>
- Lars Frantzen <lf>

Motivation

- Combine testing and verification expertise
- Proof of concept for the GAST tool
- Embed automatic testing in the development process



Smart Cards

Smart Cards are <u>omnipresent</u>:

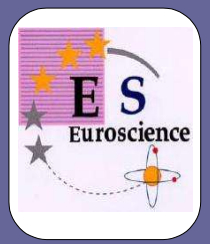
- Electronic Banking
- Telecommunication
- Identity determination
- Everyday life

Hence they are <u>critical</u> w.r.t.:

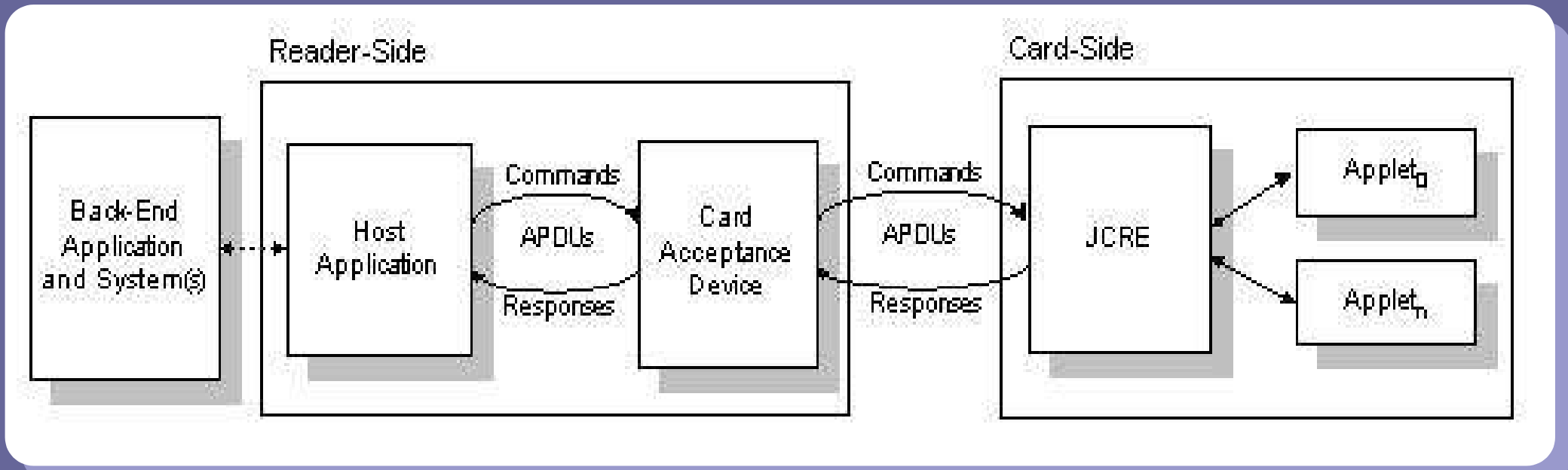
- Safety
- Security
- Interoperability

Java Cards

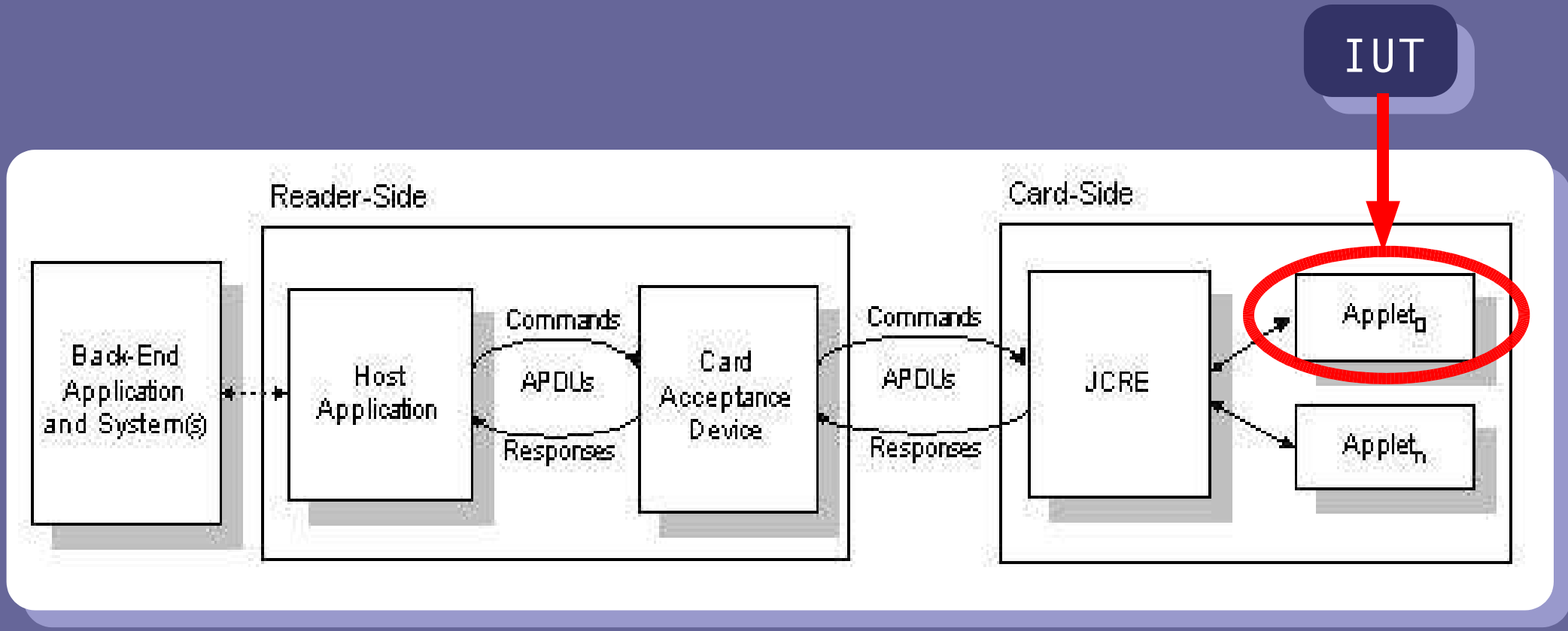
- Secure environment for applications that run on SC
- Very limited memory and processing capabilities
- Multiple applications can be deployed
- New ones can be added



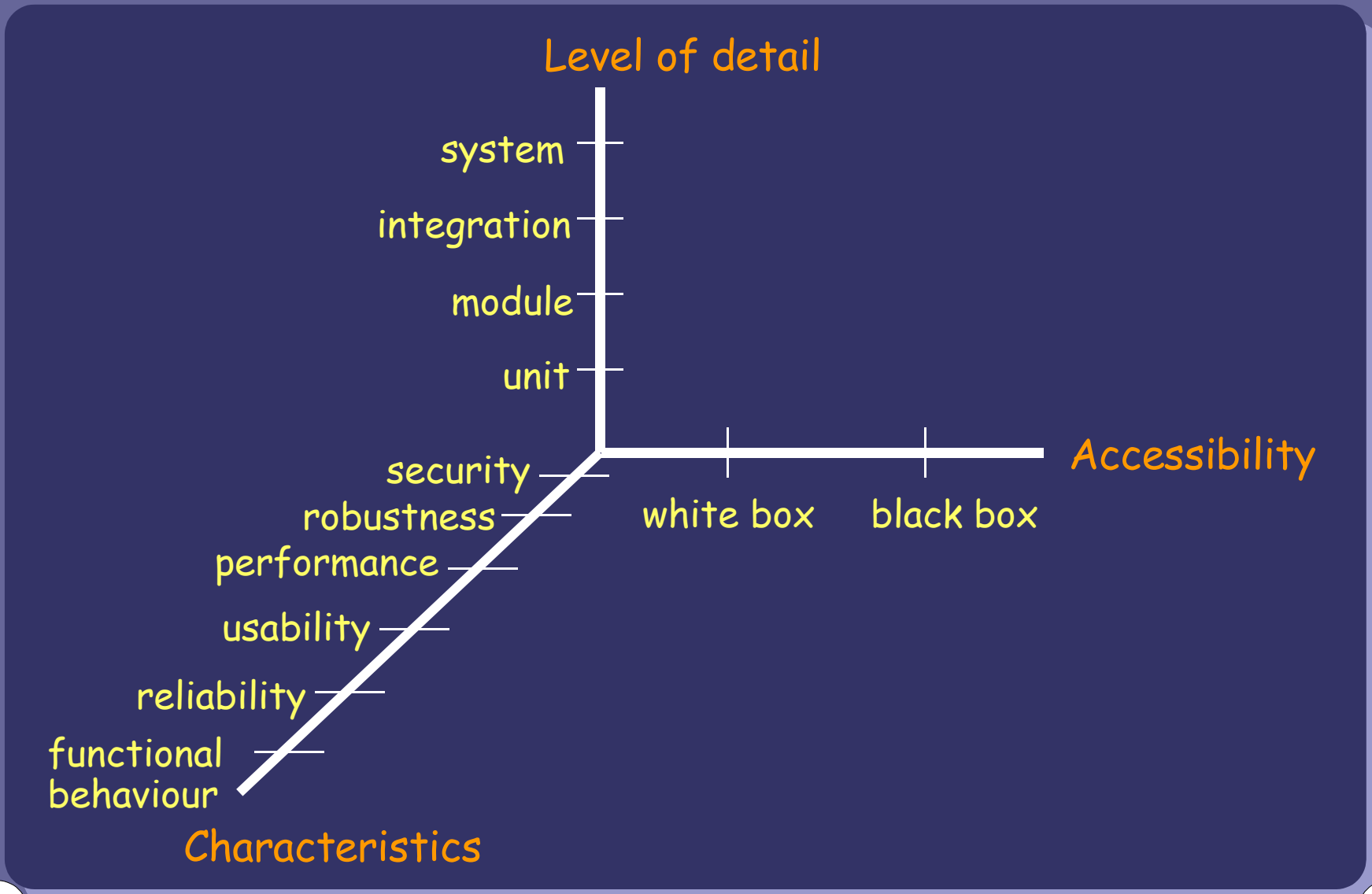
Java Cards


Reader-Side
Card-Side
Back-End Application and System(s)
Host Application
Commands
APDUs
Responses
Card Acceptance Device
Commands
APDUs
Responses
JCRE
Applet0
Appletn

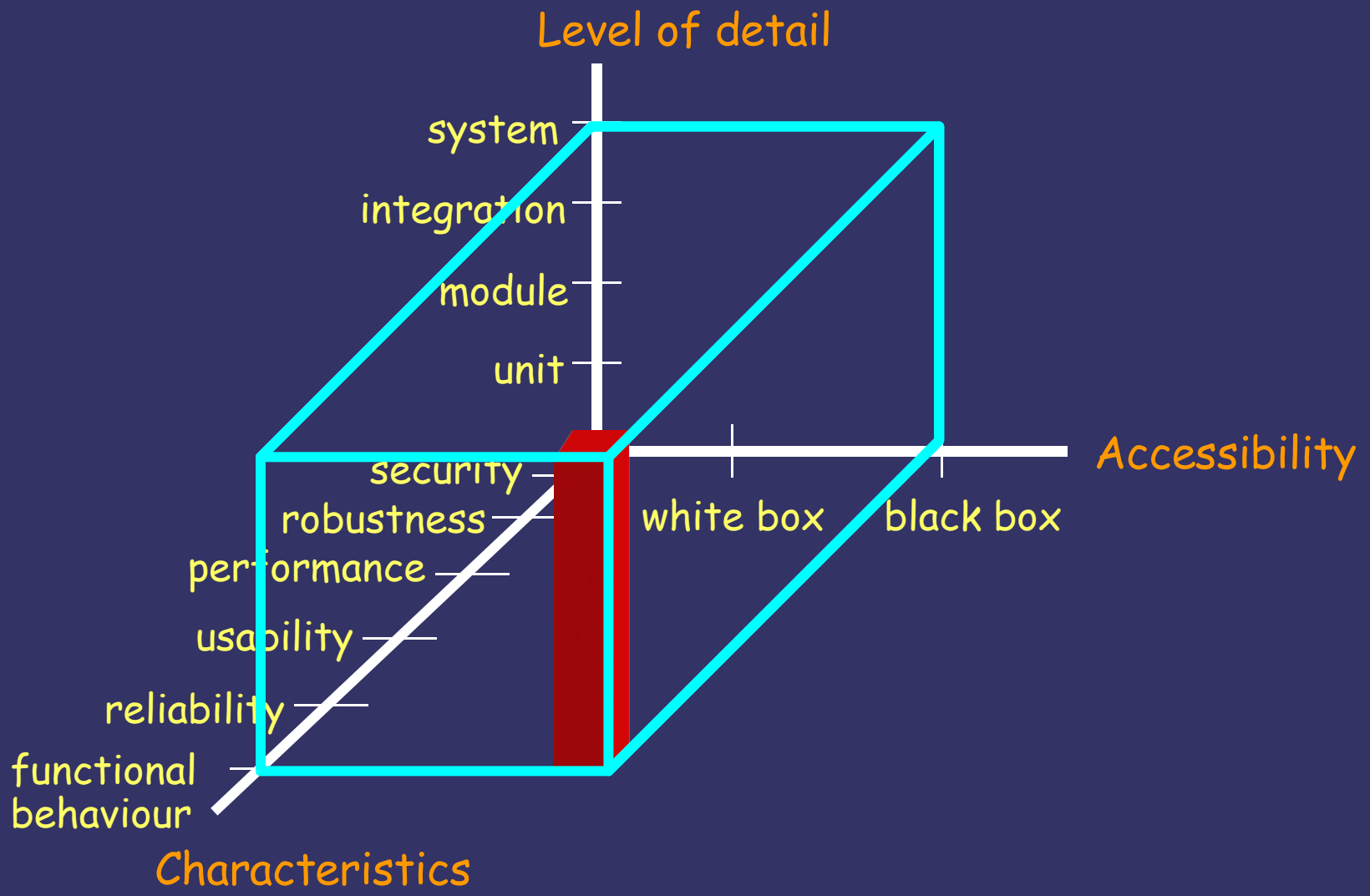
Java Cards


IUT
Reader-Side
Card-Side
Back-End Application and System(s)
Host Application
Commands
APDUs
Responses
Card Acceptance Device
Commands
APDUs
Responses
JCRE
Applet0
Appletn

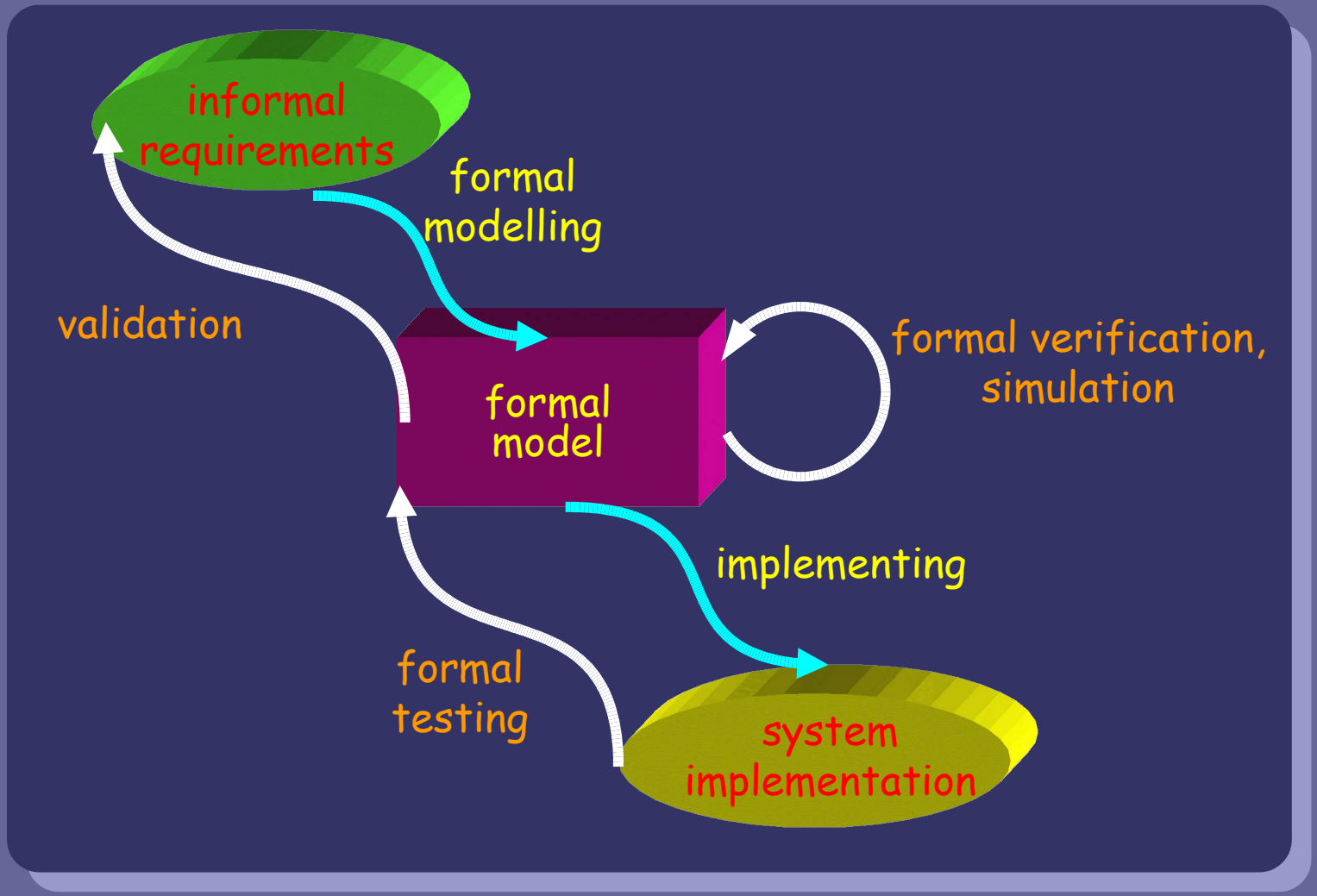
Testing - Kinds



Testing - Kinds



Model-Based Testing



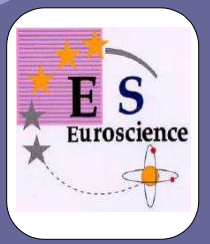
A Simple Purse Applet

The input events which the electronic purse can receive from the terminal are:

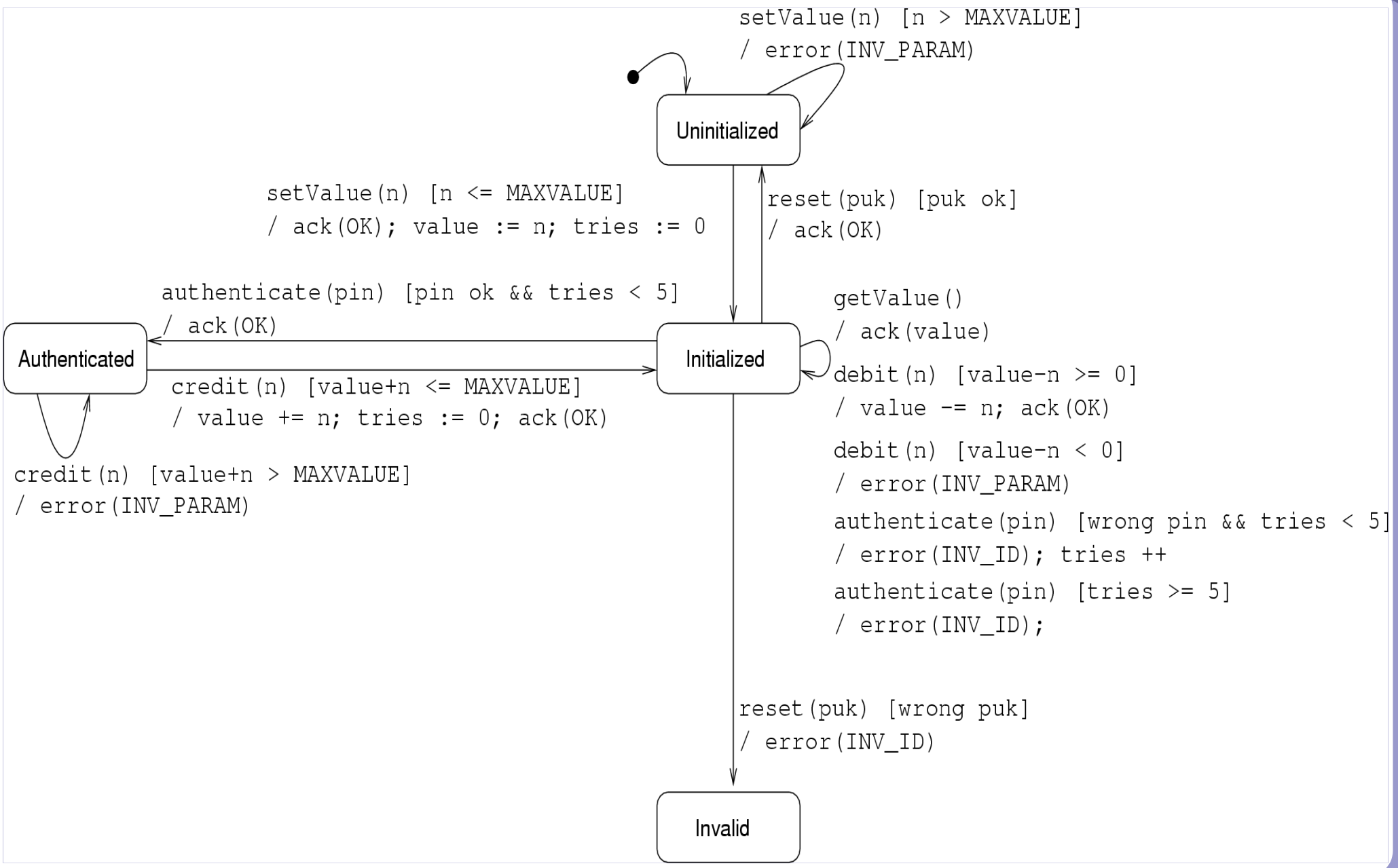
- setValue(n)
- getValue()
- debit(n)
- authenticate(pin)
- credit(n)
- reset(puk)

Output event (sent to the terminal) are:

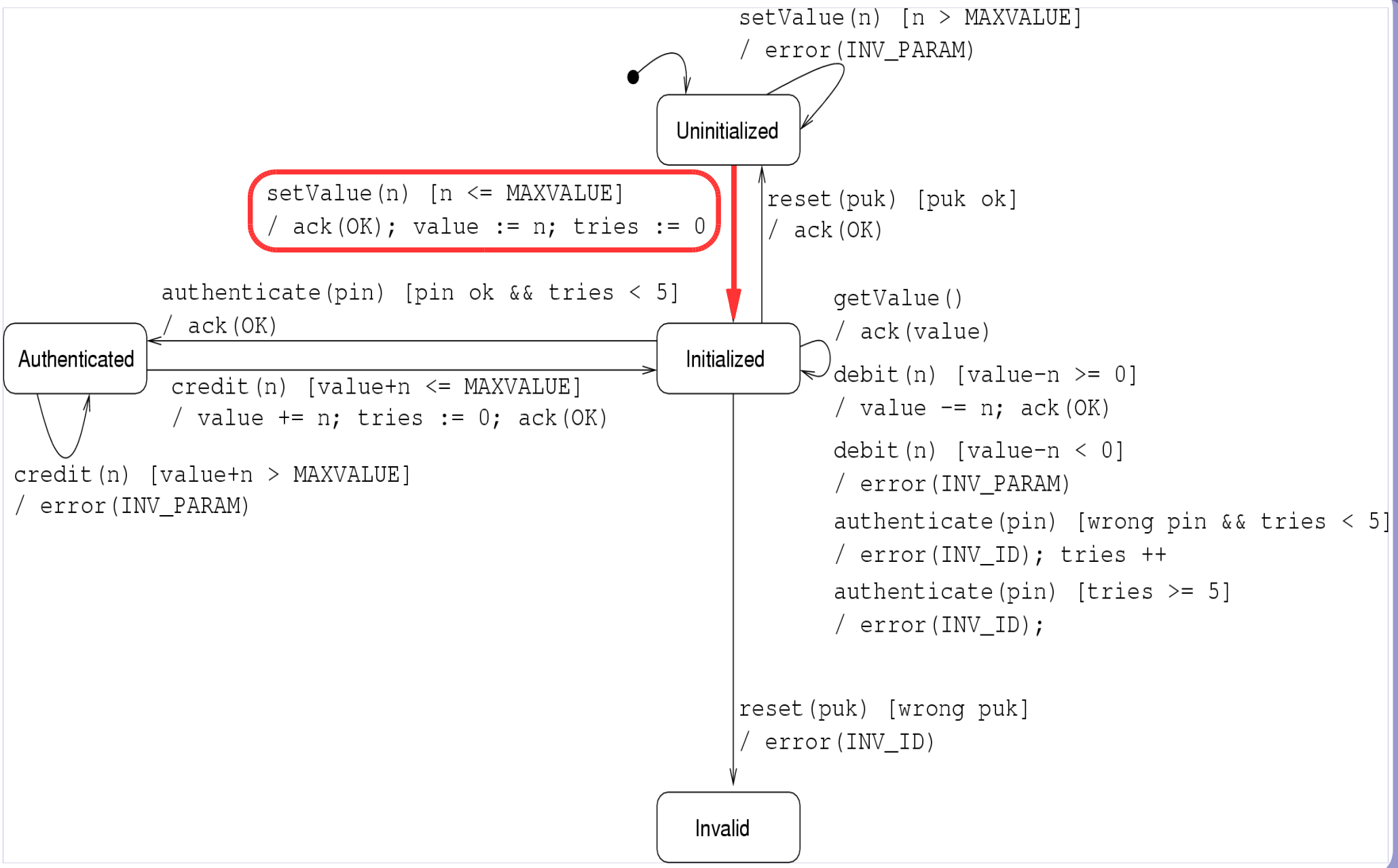
- ack(n)
- error(n)



A Statechart Model



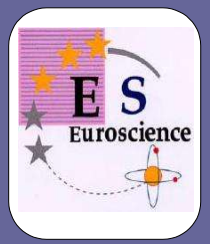
A Statechart Model



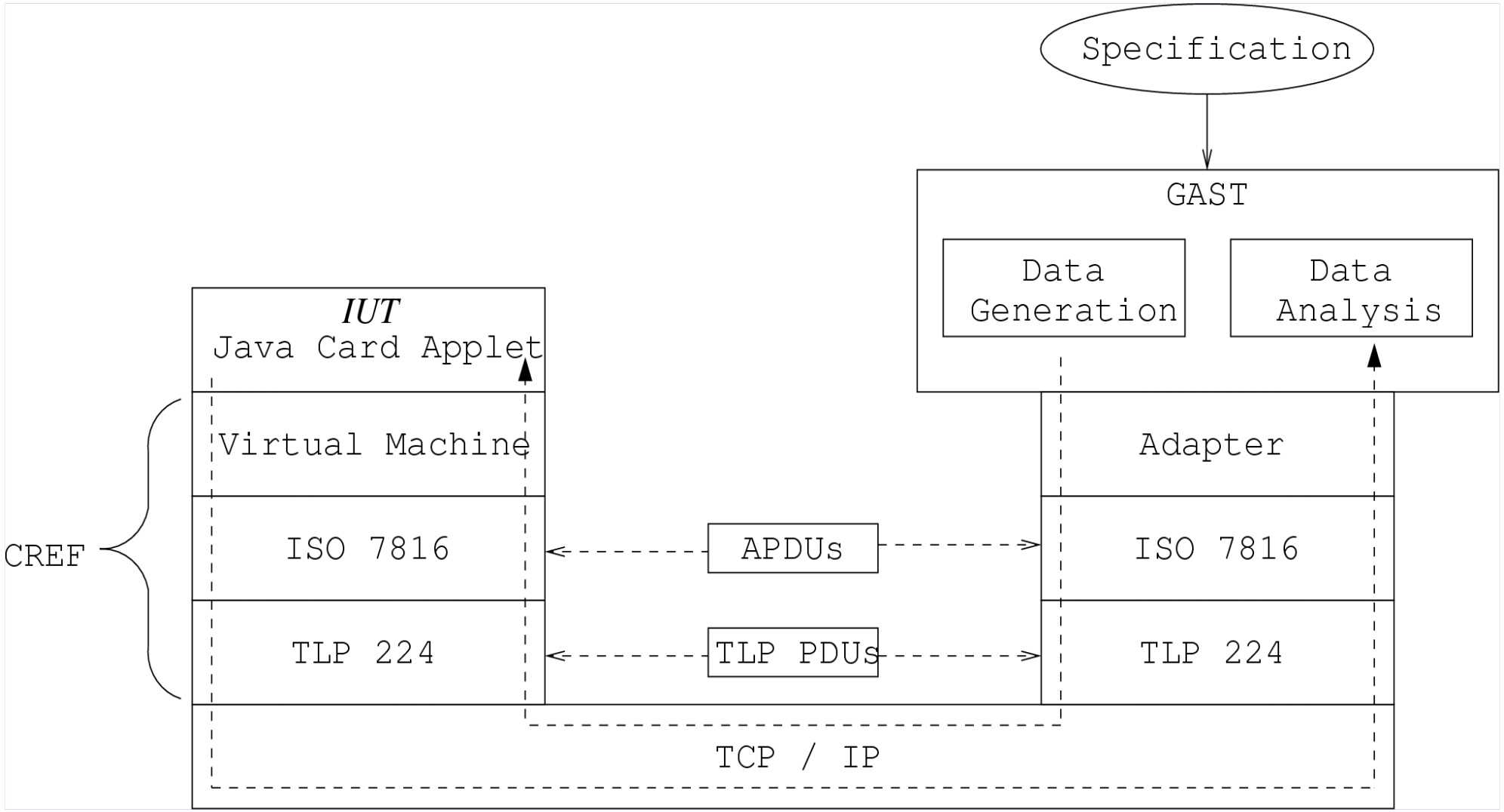
GAST

GAST (Generic Automatic Software Test)

- automatic test generation, execution, analysis
- implemented in the FL CLEAN
- EFSM-like specifications
- lazy evaluation
- on-the-fly execution

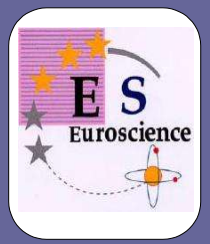


The Test Architecture



Experiences: Development

- The model and the applet were developed simultaneously
  → both evolved iterative
- Some issues here were:
- Gap between specifications (ISO-7816) and implementations (Java Cards)
- Implicit model assumptions, e.g. non-negativity of numbers
- Model may leave out crucial implementation issues



Experiences: Development

- Lessons learned:
- Implementing a simple applet is far from trivial
- Iterative co-development of model and implementation is very useful
- Both evolve simultaneously, leading to a complete and reliable specification and implementation
- Model-based, automatic testing is vital

Experiences: Mutants

- 20 Mutants were created, e.g.:

1) Omit check for MAXVALUE when doing setValue()

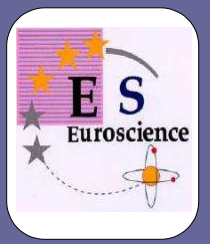
2) Check (value + n) <= MAXVALUE instead of
   n <= (MAXVALUE – value), may lead to an overflow

3) Do not check, if one debits more than the actual
   credit, leading to a negative value

4) Do not reset tries-counter after authentication

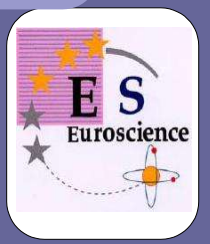
5) Do not reset tries-counter after reset()

...



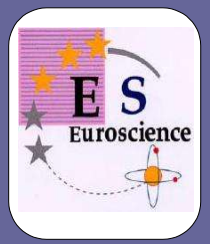
Experiences: Mutants

Mutant	Test events	Paths	Time	Type
1	166	33	0.4s	ADT
2	40676	7629	71.0s	SC
3	78	22	0.2s	ADT
4	1086	60	1.4s	SC
5	41704	6918	66.0s	ADT



Summary / Outlook

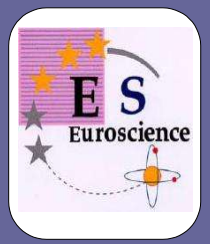
- Promising framework to support an iterative, model-based development of Smart Card applets.
- Integration of automatic testing allows for a quick and vast improvement of the quality of both specification and implementation.
- Testing a real-world application is planned.
- Also other test tools will be embedded.



Literature

- Weelden, Frantzen, Oostdijk, Koopman, Tretmans:
  On-the-fly Formal Testing of a Smart Card Applet
  NIII Report NIII-R0428, June 2004
  www.cs.kun.nl/research/reports/full/NIII-R0428.pdf

- Broy, Jonsson, Katoen, Leucker, Pretschner (Eds.):
  Model-based Testing of Reactive Systems -
  A seminar volume
  LNCS, to appear in 2004



- Thank You!