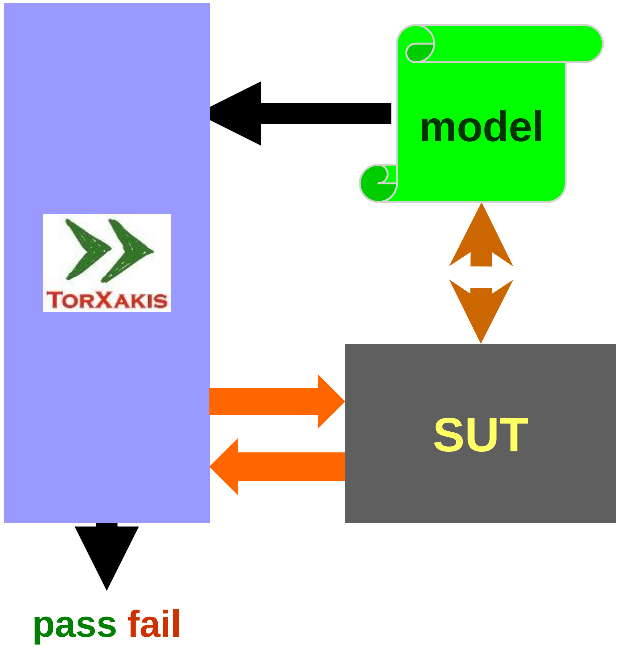
Model-Based Testing

Theory

Tools

Applications



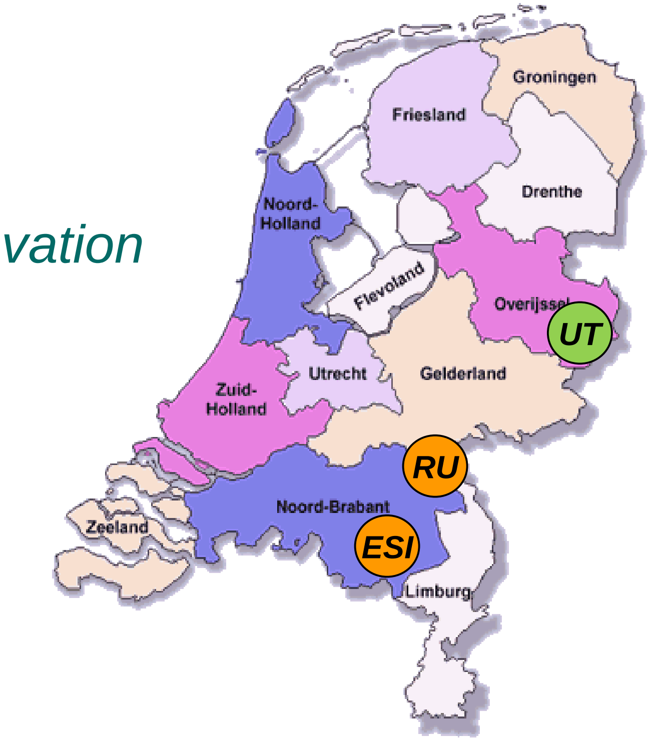
Jan Tretmans

ESI – Embedded Systems Innovation by TNO
Radboud University Nijmegen
Högskolan i Halmstad
jan.tretmans@tno.nl

Jan Tretmans

ESI

Embedded Systems Innovation
by TNO
Eindhoven
The Netherlands



IN DEI NOMINE FELICITER

Radboud University
Nijmegen
The Netherlands

HÖGSKOLAN
I HALMSTAD

ESI

Embedded Systems Innovation

by TNO

Vision:

“Create economic and societal impact & value by embedded systems technology”

Mission:

“To advance industrial innovation and academic excellence in embedded systems engineering”

ESI WITHIN TNO

TNO innovation for life

TNO THEMES AND INNOVATION AREAS

OUR THEMES

- HEALTHY LIVING
- INDUSTRIAL INNOVATION
- DEFENCE, SAFETY & SECURITY
- ENERGY
- TRANSPORT & MOBILITY
- BUILT ENVIRONMENT
- INFORMATION SOCIETY

OUR INNOVATION AREAS

HEALTHY FOR LIFE
FOOD AND NUTRITION
WORK AND EMPLOYMENT
BIOMEDICAL INNOVATIONS

HIGH TECH SYSTEMS AND MATERIALS
SUSTAINABLE CHEMICAL INDUSTRY
SPACE

DEFENCE RESEARCH
SAFETY AND SECURITY RESEARCH

OIL AND GAS
ENERGY EFFICIENCY
GEOLOGICAL SURVEY OF THE NETHERLANDS
MARITIME AND OFFSHORE

RELIABLE MOBILITY SYSTEMS
SAFE AND CLEAN TRANPORT

URBAN DEVELOPMENT
BUILDING AND INFASTRUCTURE

INFOSTRUCTURES
INFRASTRUCTURES

ESI : Embedded Systems Engineering

Embedding intelligence
into physical products

Typical characteristics:

□ Multi-disciplinary design

□ Software complexity

□ Physical environments

□ Distributed or networked

□ Constrained resources

□ Critical applications

□ Quality, reliability, testing

□ System evolution

Noldus
Information Technology

PHILIPS
Consumer Electronics
Medical Systems
Research
Applied Technologies

NXP founded by Philips

THALES

Technolution
AUTOMATION TECHNOLOGY

ASML

FEI COMPANY
TOOLS FOR NANOTECH

océ

VANDERLANDE INDUSTRIES

CHESS

Imtech
ICT

TASS software professionals

DEMCON advanced mechatronics

Universiteit Leiden

TU/e technische universiteit eindhoven

Radboud Universiteit Nijmegen

RuG

TUDelft
Delft University of Technology

imec

University of Twente
The Netherlands

KATHOLIEKE UNIVERSITEIT LEUVEN

vrije Universiteit amsterdam

Universiteit Antwerpen

Universiteit Maastricht

HÖGSKOLAN I HALMSTAD

CWI

UNIVERSITEIT VAN AMSTERDAM

DESIGN TECHNOLOGY INSTITUTE ti

Groningen
Friesland
Drenthe
Noord-Holland
Flevoland
Overijssel
Utrecht
Gelderland
Zuid-Holland
Zeeland
Noord-Brabant
Limburg
ESI

Industrial Network

Academic Network

Research cooperation with leading Dutch high-tech multinational industries & SME's

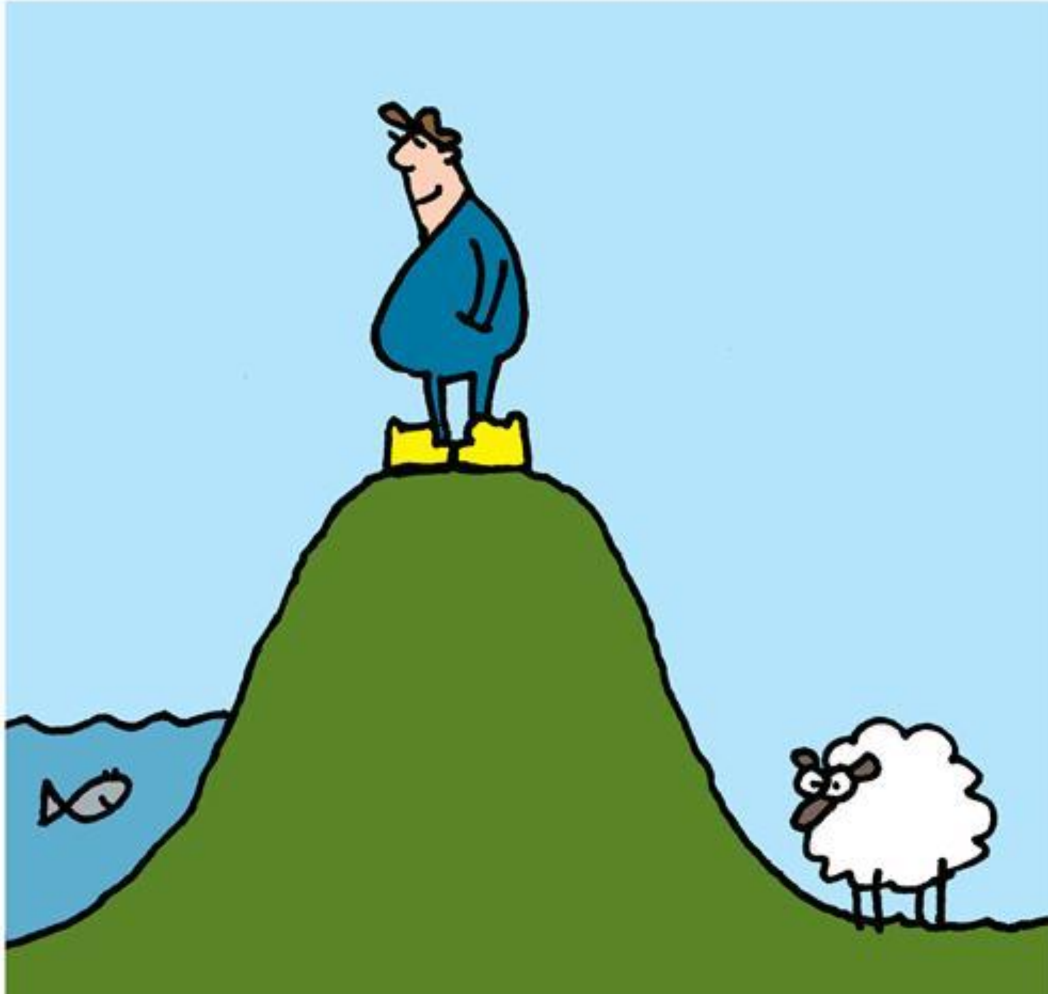
Research cooperation with all Dutch universities with embedded systems research

Research cooperation in EU projects

Model-Based Testing

Motivation

What do Dykes have to do with Quality of Embedded Systems ?



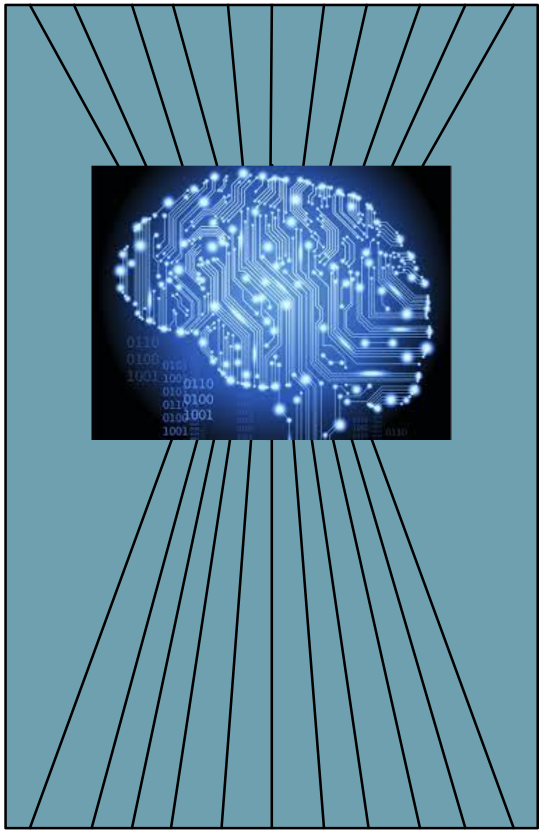
What do Dykes have to do with Quality of Embedded Systems ?

Embedded Systems

Quality of Embedded Systems

Software is brain of system

- software controls, connects, monitors almost any aspect of ES system behaviour
- majority of innovation is in software



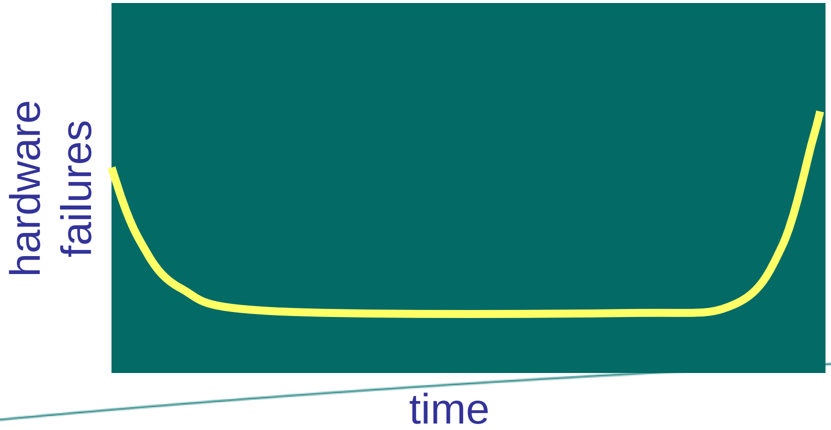
Software determines quality and reliability of Embedded System

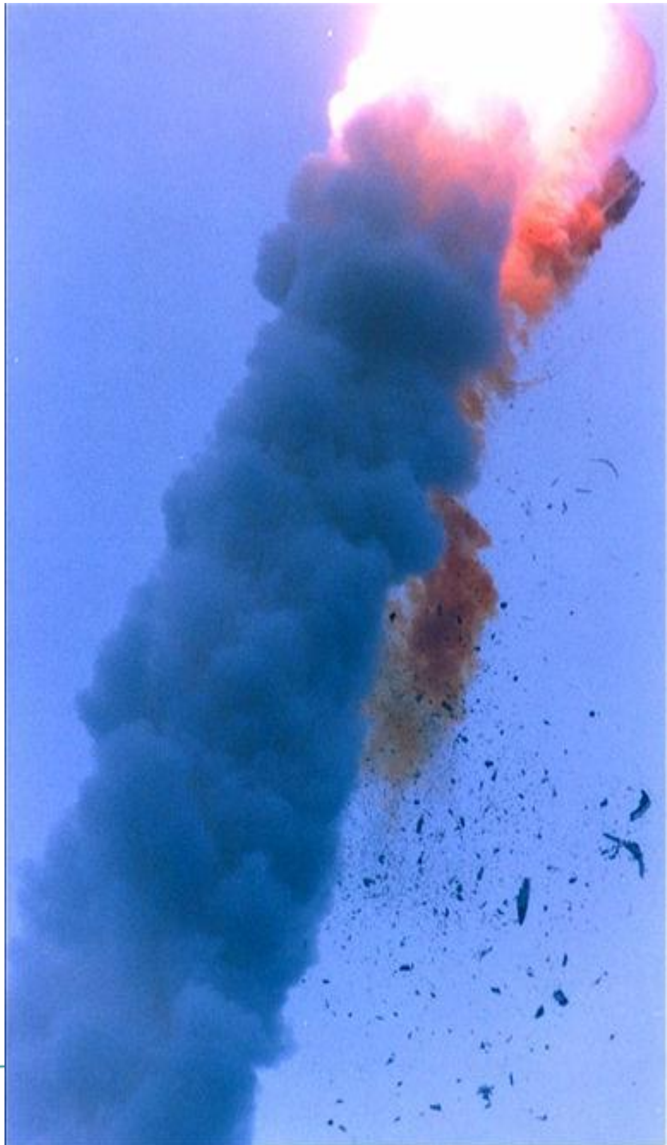
- often **>** 50 % of system defects are software bugs

Software is Different

Software is different from hardware :

- non-continuous
- any bug is a design error
- adopting redundancy is useless
- no wear and tear
- no MTBF; what is software reliability?





Trends & Challenges

complexity
size

connectivity
systems-of-systems

multi
disciplinarity

quality
challenges

change
variability
evolvability

heterogeneous
components

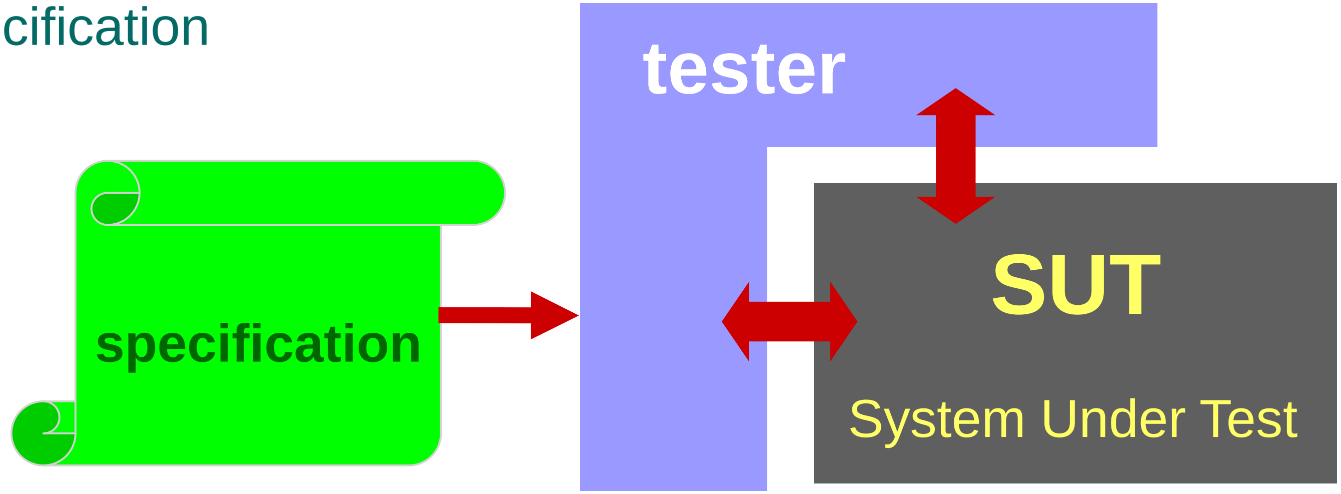
uncertainty

Model-Based Testing

Software Testing

specification-based,
active, black-box testing
of functionality

Checking or measuring
some quality characteristics
of an executing software object
by performing experiments
in a controlled way
w.r.t. a specification



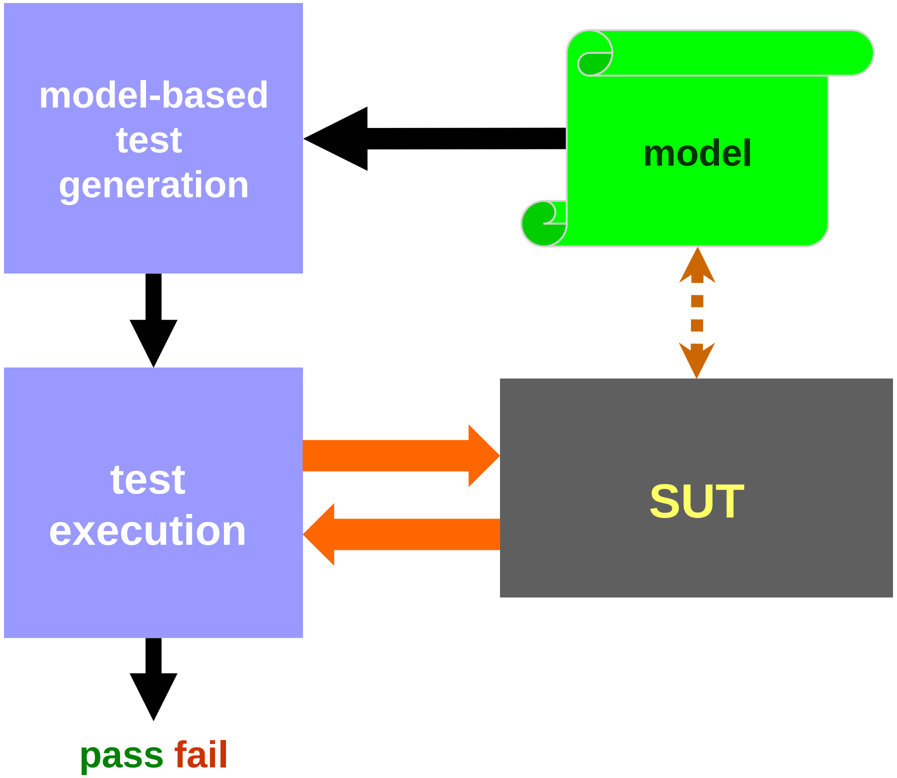
Model-Based Testing

MBT

next step in

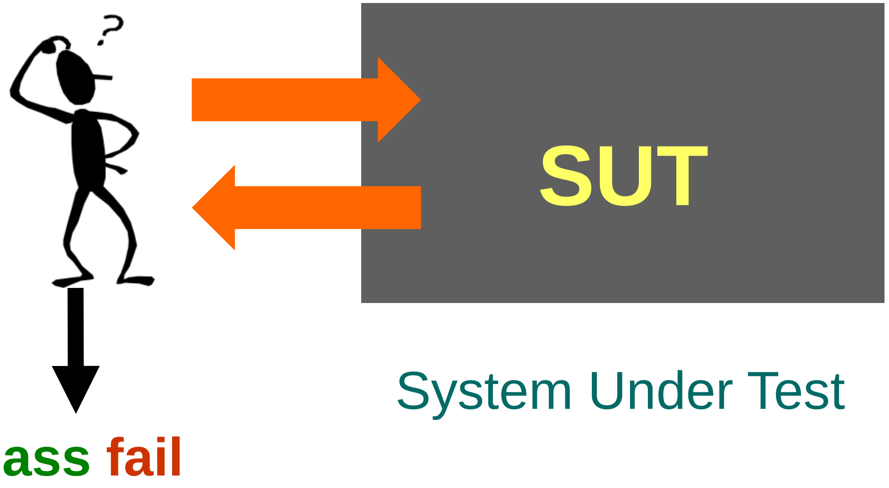
test automation:

- + test generation
- + result analysis



1 : Manual Testing

1. **Manual testing**

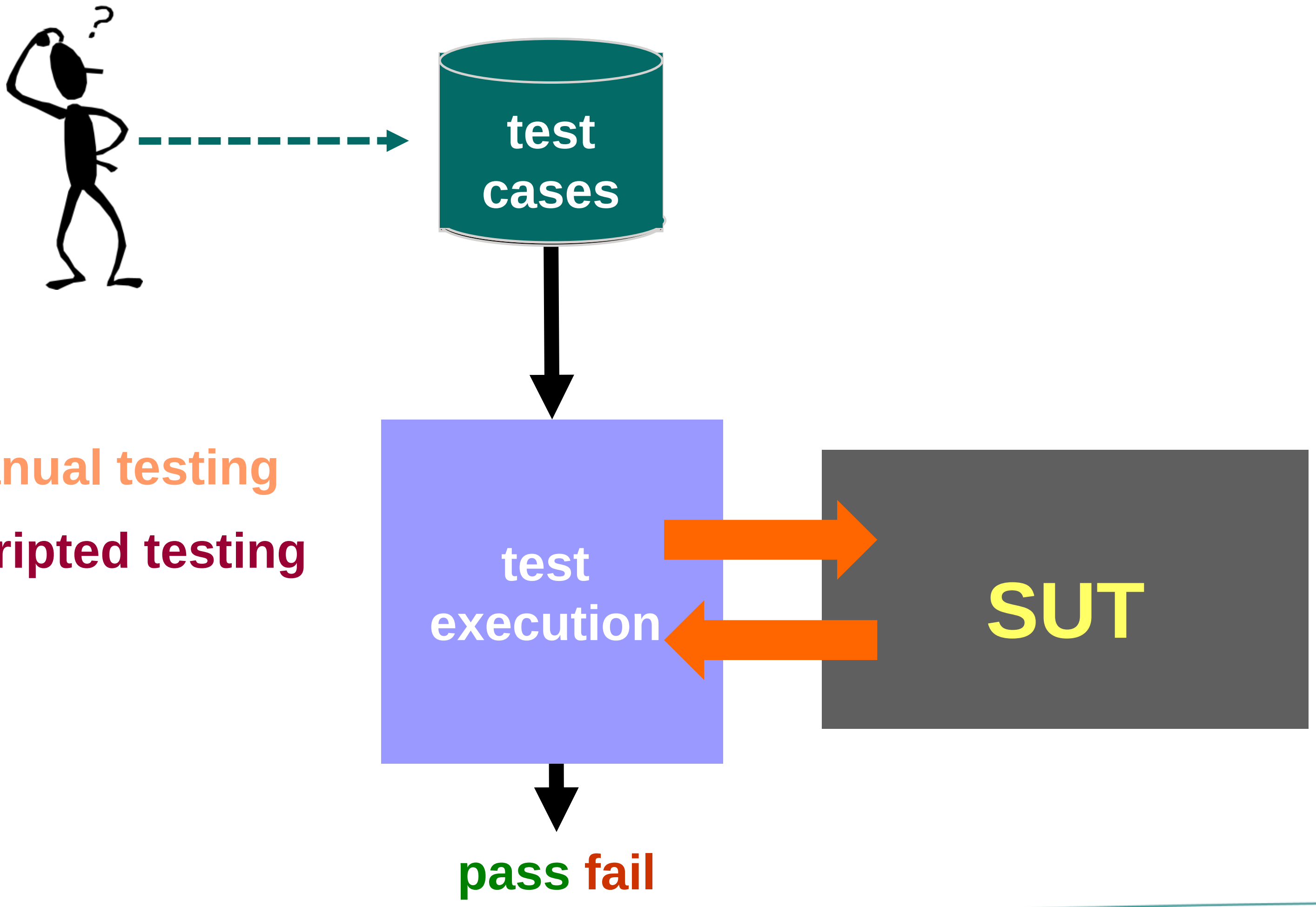


SUT

System Under Test

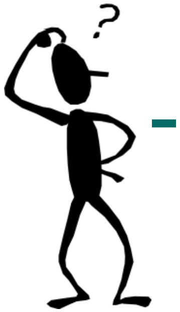
pass fail

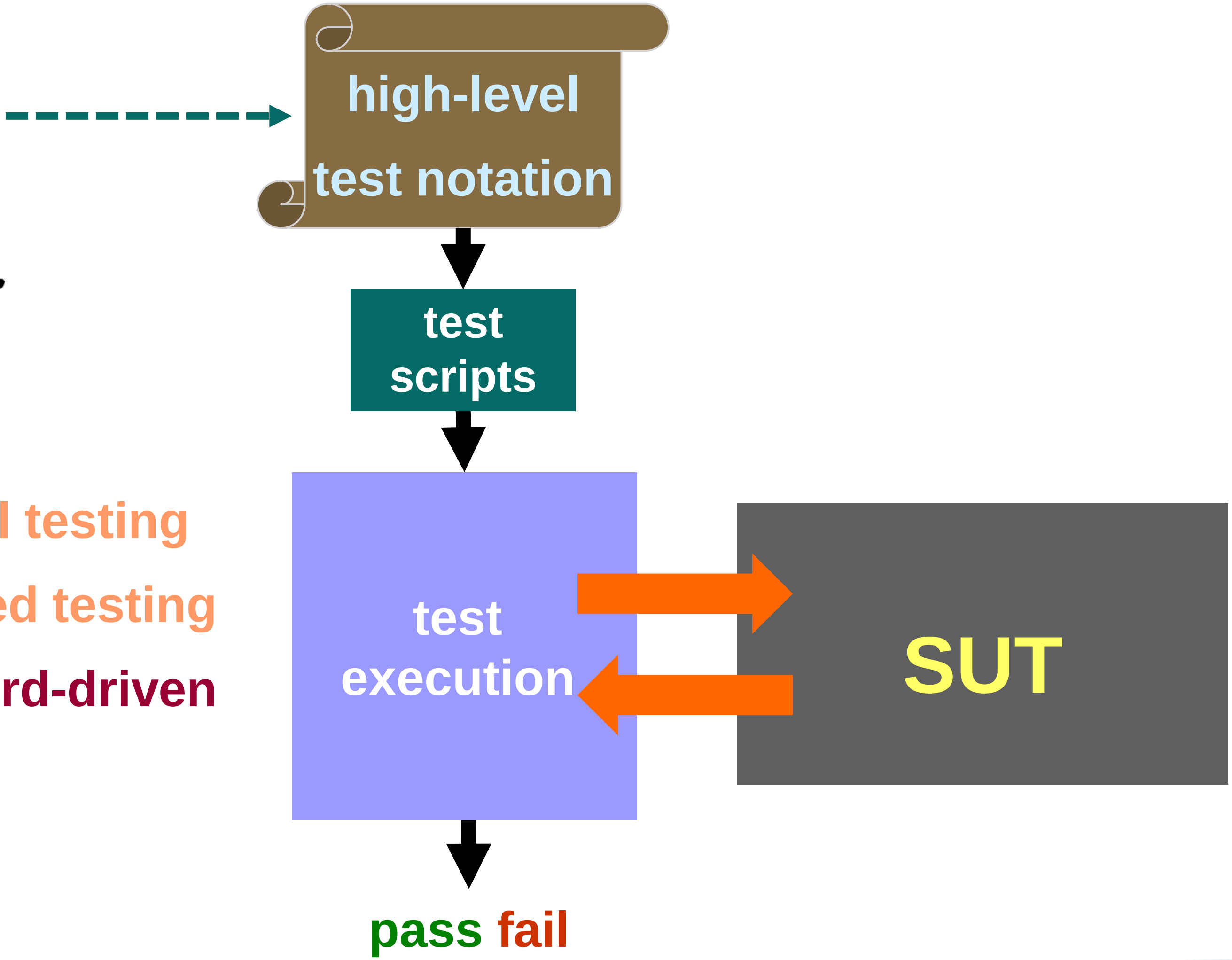
2 : Scripted Testing



1. Manual testing
2. Scripted testing

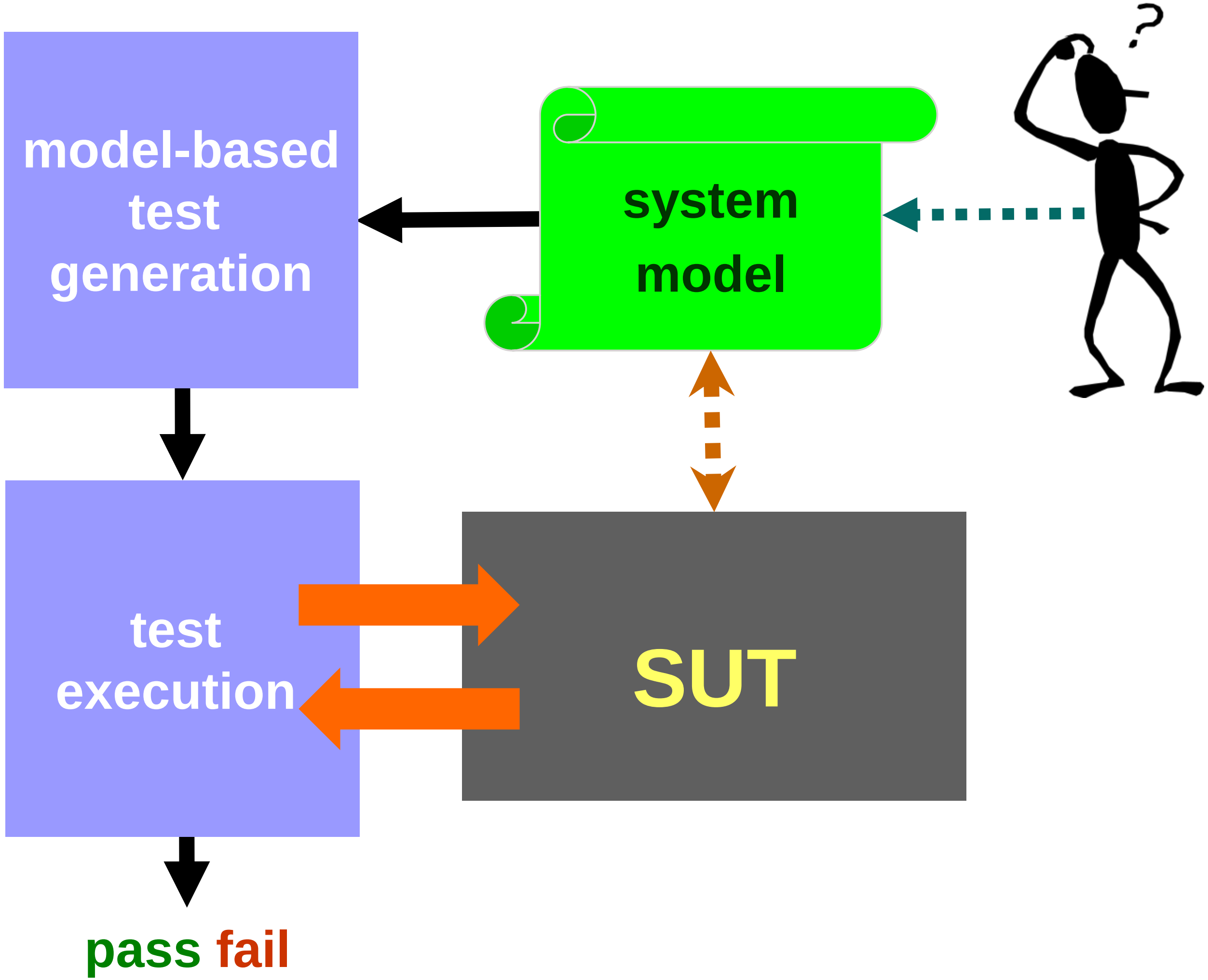
3 : Keyword-Driven Testing





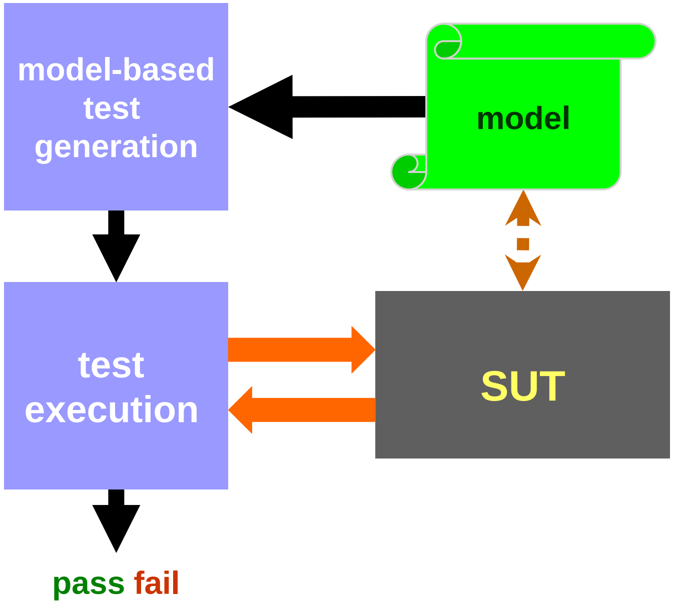
1. Manual testing
2. Scripted testing
3. Keyword-driven testing

4 : Model-Based Testing



1. Manual testing
2. Scripted testing
3. Keyword-driven testing
4. **Model-based testing**

MBT : Benefits



detecting more bugs

faster and cheaper

MBT: next step in test automation

- **Automatic test generation**

  + test execution + result analysis

- **More, longer, and diversified test cases**

  more variation in test flow and in test data

- **Model is precise and consistent test basis**

  unambiguous analysis of test results

- **Test maintenance by maintaining models**

  improved regression testing

- **Expressing test coverage**

  model coverage

  customer profile coverage

MBT : Benefits ?

But ....

If doing MBT is so smart,

why ain't you rich ?

MBT : State of the Art

- promising, emerging
- a number of successful applications
- many companies are experimenting

MBT : State for the Future

(*for High-tech Embedded Systems*)

- **?**

MBT : State of Practice

- lagging behind

Reasons

- technical
- tools
- organizational
- maturity of testing
- educational
- . . . . .

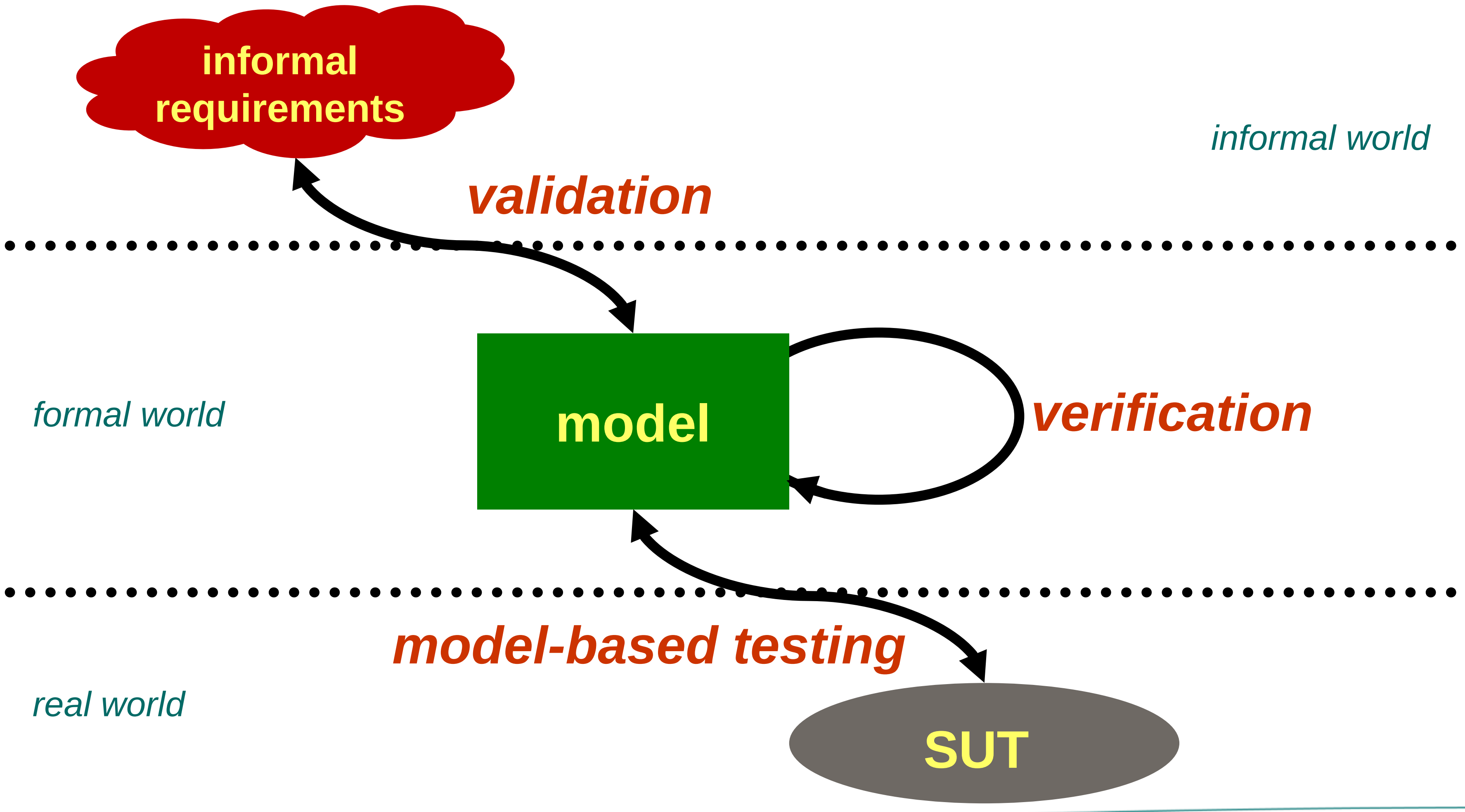
Model-Based

Verification, Validation, Testing, ...

Doing Something with Models

• **Modelling**	making a model reveals errors
• **Simulation**	go step-by-step through the model
• **Model checking**	go through all states of the model
• **Theorem proving**	prove theorems about the model
• **Code generation**	executable code from the model
• **Testing**	test an implementation for compliance
• **Model learning**	generate a model from observation

Validation, Verification, Testing



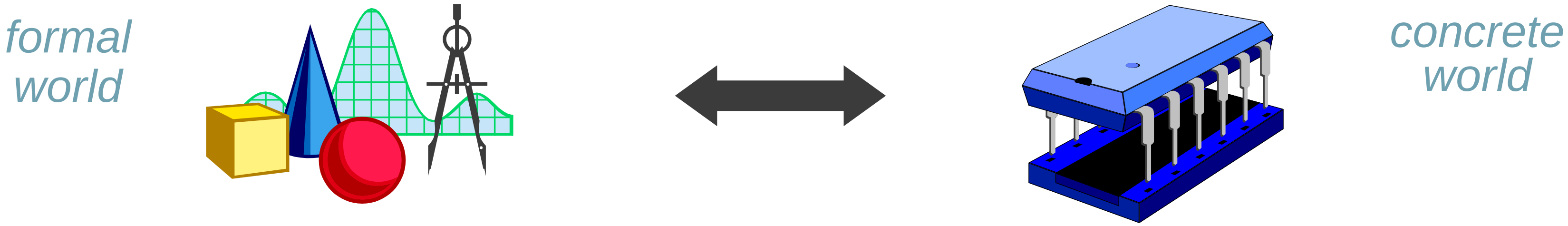
Verification and Testing

Model-based verification :

- formal manipulation
- prove properties
- performed on model

Model-based testing :

- experimentation
- show error
- concrete system



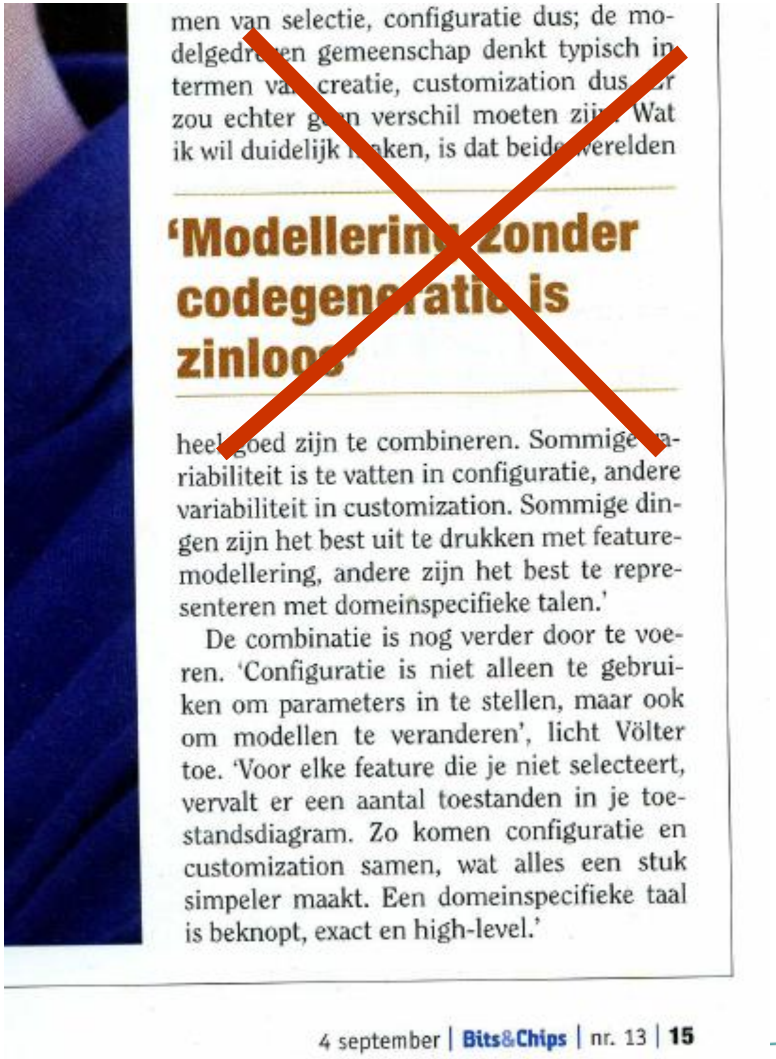
formal world

concrete world

Verification is only as good as the validity of the model on which it is based

Testing can only show the presence of errors, not their absence

Code Generation from a Model



men van selectie, configuratie dus; de modelgedreven gemeenschap denkt typisch in termen van creatie, customization dus. Er zou echter geen verschil moeten zijn. Wat ik wil duidelijk maken, is dat beide werelden

~~‘Modellering zonder codegeneratie is zinloos’~~

heel goed zijn te combineren. Sommige variabiliteit is te vatten in configuratie, andere variabiliteit in customization. Sommige dingen zijn het best uit te drukken met featuremodellering, andere zijn het best te representeren met domeinspecifieke talen.’

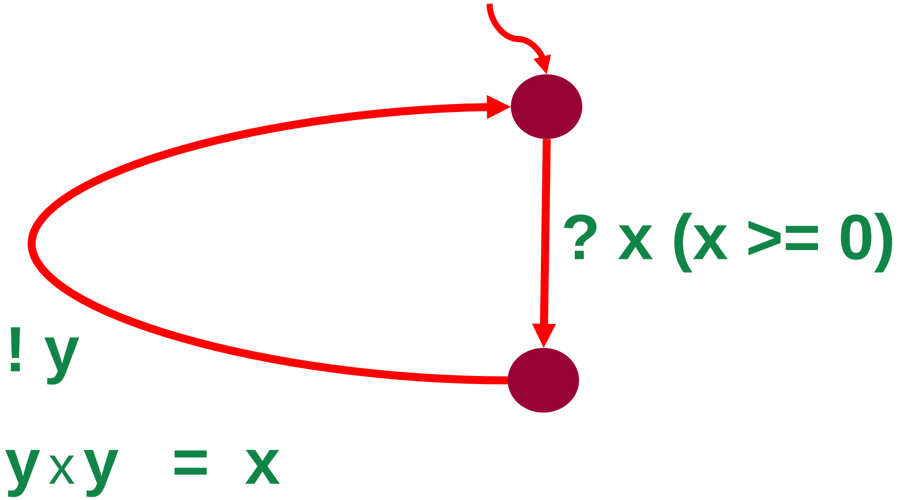
De combinatie is nog verder door te voeren. ‘Configuratie is niet alleen te gebruiken om parameters in te stellen, maar ook om modellen te veranderen’, licht Völter toe. ‘Voor elke feature die je niet selecteert, vervalt er een aantal toestanden in je toestandsdiagram. Zo komen configuratie en customization samen, wat alles een stuk simpeler maakt. Een domeinspecifieke taal is beknopt, exact en high-level.’

4 september | Bits&Chips | nr. 13 | 15

A model is more (*less*)
than code generation:

- views
- abstraction
- testing of aspects
- verification and validation of aspects

Code Generation from a Model



? x (x >= 0)

! y

$y \times y = x$

model of $\sqrt{x}$

- *specification of **properties** rather than construction*
- ***under-specification***
- ***non-determinism***

Model-Based Testing

*The **ioco** Theory*

for Labelled Transition Systems

Model-Based Testing Tools

MBT Tools

- AETG
- Agatha
- Agedis
- Autolink
- Axini Test Manager
- Conformiq
- Cooper
- Cover
- DTM
- fMBT
- G∀st
- Gotcha
- Graphwalker
- JTorX
- MaTeLo
- MBTsuite
- M-Frame
- MISTA
- NModel
- OSMO
- ParTeG
- Phact/The Kit
- PyModel
- QuickCheck
- Reactis
- Recover
- RT-Tester
- SaMsTaG
- Smartesting CertifyIt
- Spec Explorer
- StateMate
- STG
- tedeso
- Temppo
- TestGen (Stirling)
- TestGen (INT)
- TestComposer
- TestOptimal
- TGV
- Tigris
- TorX
- TorXakis
- T-Vec
- Tveda
- Uppaal-Cover
- Uppaal-Tron
- . . . .

MBT Tools *ioco*

- AETG
- Agatha
- [✓ Agedis]
- Autolink
- [✓ Axini Test Manager]
- Conformiq
- Cooper
- Cover
- DTM
- fMBT
- G∀st
- Gotcha
- Graphwalker
- [✓ JTorX]
- MaTeLo
- MBTsuite
- M-Frame
- MISTA
- NModel
- OSMO
- ParTeG
- Phact/The Kit
- PyModel
- QuickCheck
- Reactis
- Recover
- RT-Tester
- SaMsTaG
- Smartesting CertifyIt
- [✓ Spec Explorer]
- StateMate
- [✓ STG]
- tedeso
- Temppo
- [✓ TestGen (Stirling)]
- TestGen (INT)
- TestComposer
- TestOptimal
- [✓ TGV]
- Tigris
- [✓ TorX]
- [✓ TorXakis]
- T-Vec
- [✓ Tveda]
- Uppaal-Cover
- [✓ Uppaal-Tron]
- . . . .

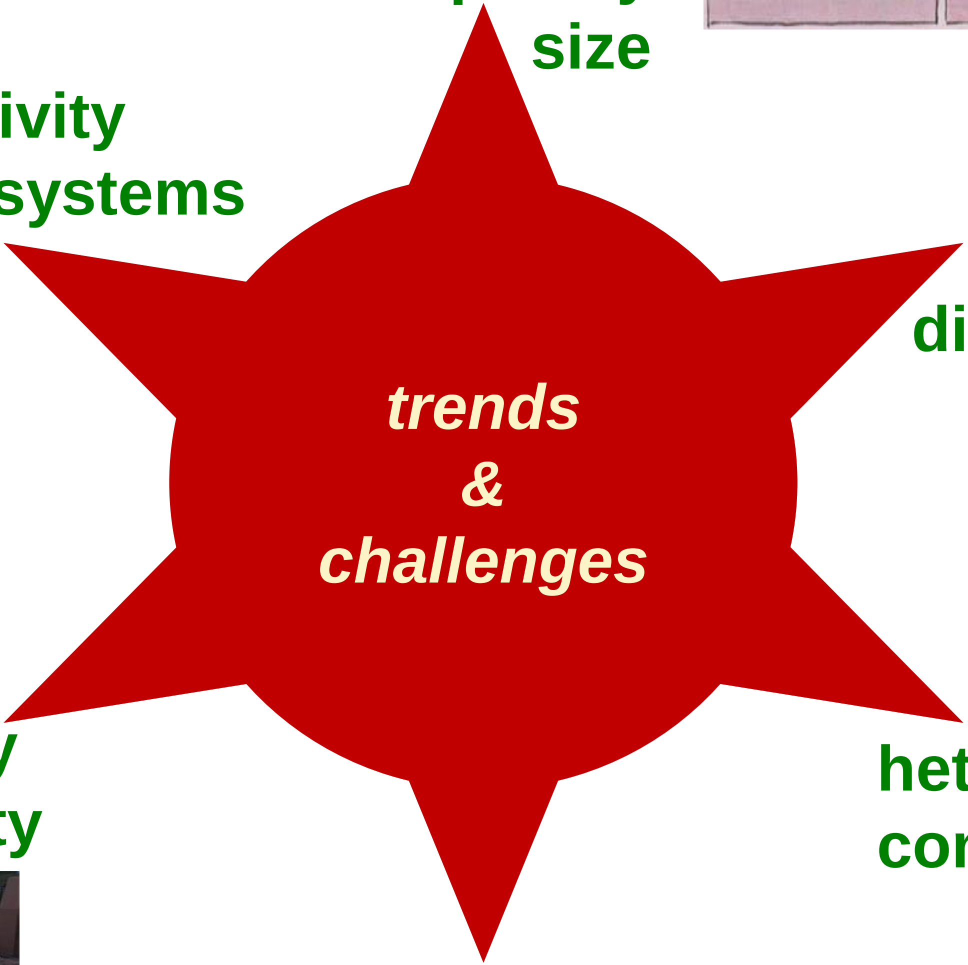
Yet Another MBT Tool : TorXakis

- AETG
- Agatha
- Agedis
- Autolink
- Axini Test Manager
- Conformiq
- Cooper
- Cover
- DTM
- fMBT
- G∀st
- Gotcha
- Graphwalker
- JTorX
- MaTeLo
- MBTsuite
- M-Frame
- MISTA
- NModel
- OSMO
- ParTeG
- Phact/The Kit
- PyModel
- QuickCheck
- Reactis
- Recover
- RT-Tester
- SaMsTaG
- Smartesting CertifyIt
- Spec Explorer
- StateMate
- STG
- tedeso
- Temppo
- TestGen (Stirling)
- TestGen (INT)
- TestComposer
- TestOptimal
- TGV
- Tigris
- TorX
- TorXakis
- T-Vec

TorXakis

- Uppaal-Cover
- Uppaal-Tron
- . . . .

Trends & Challenges



trends

&

challenges

complexity
size

connectivity
systems-of-systems

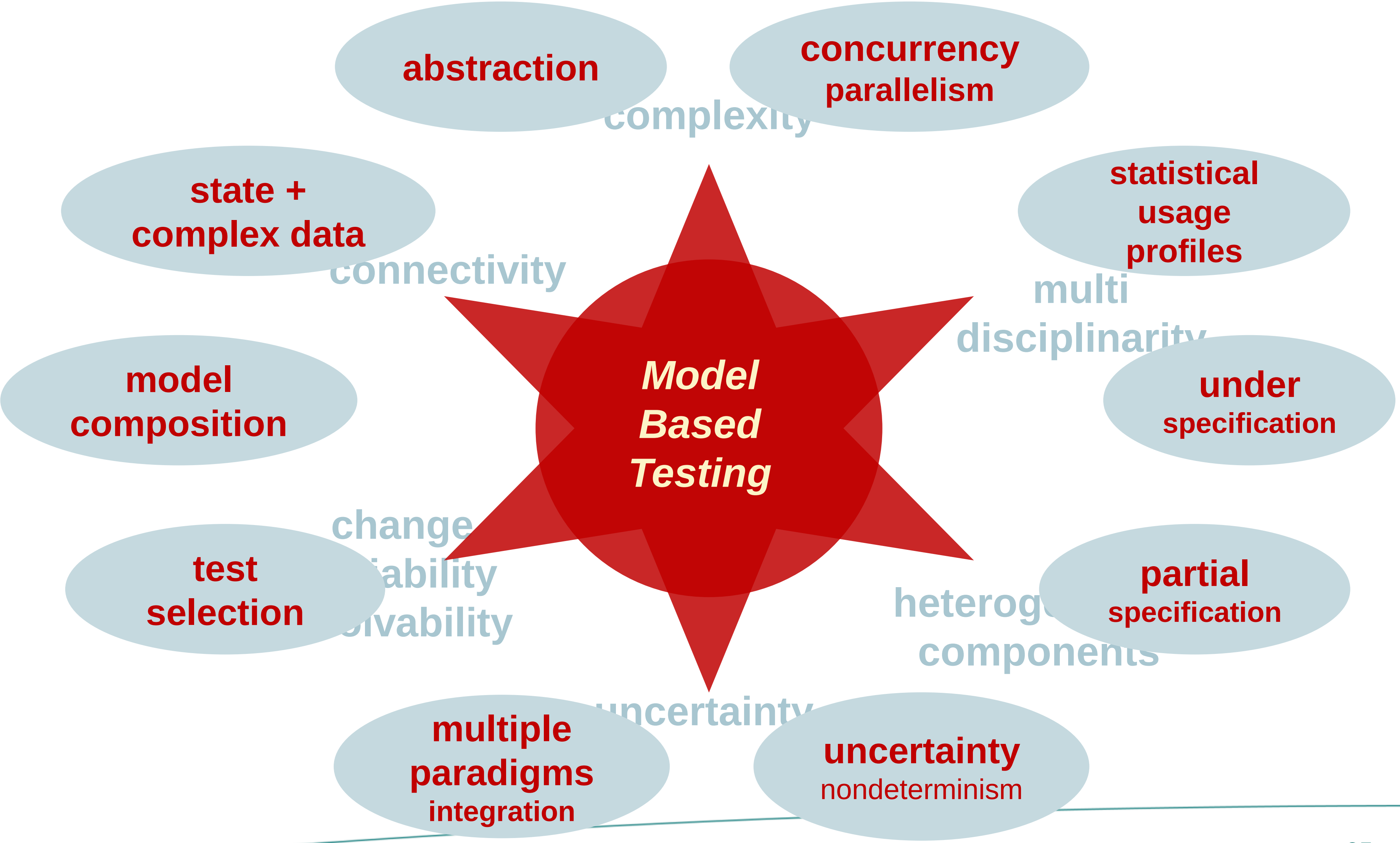
multi
disciplinarity

change
variability
evolvability

heterogeneous
components

uncertainty

MBT : Next Step Challenges



Model Based Testing

abstraction

concurrency
parallelism

state +
complex data

statistical
usage
profiles

model
composition

under
specification

test
selection

partial
specification

multiple
paradigms
integration

uncertainty
nondeterminism

complexity

connectivity

multi
disciplinarity

change
variability
evolvability

heterogeneous
components

uncertainty

Model-Based Testing

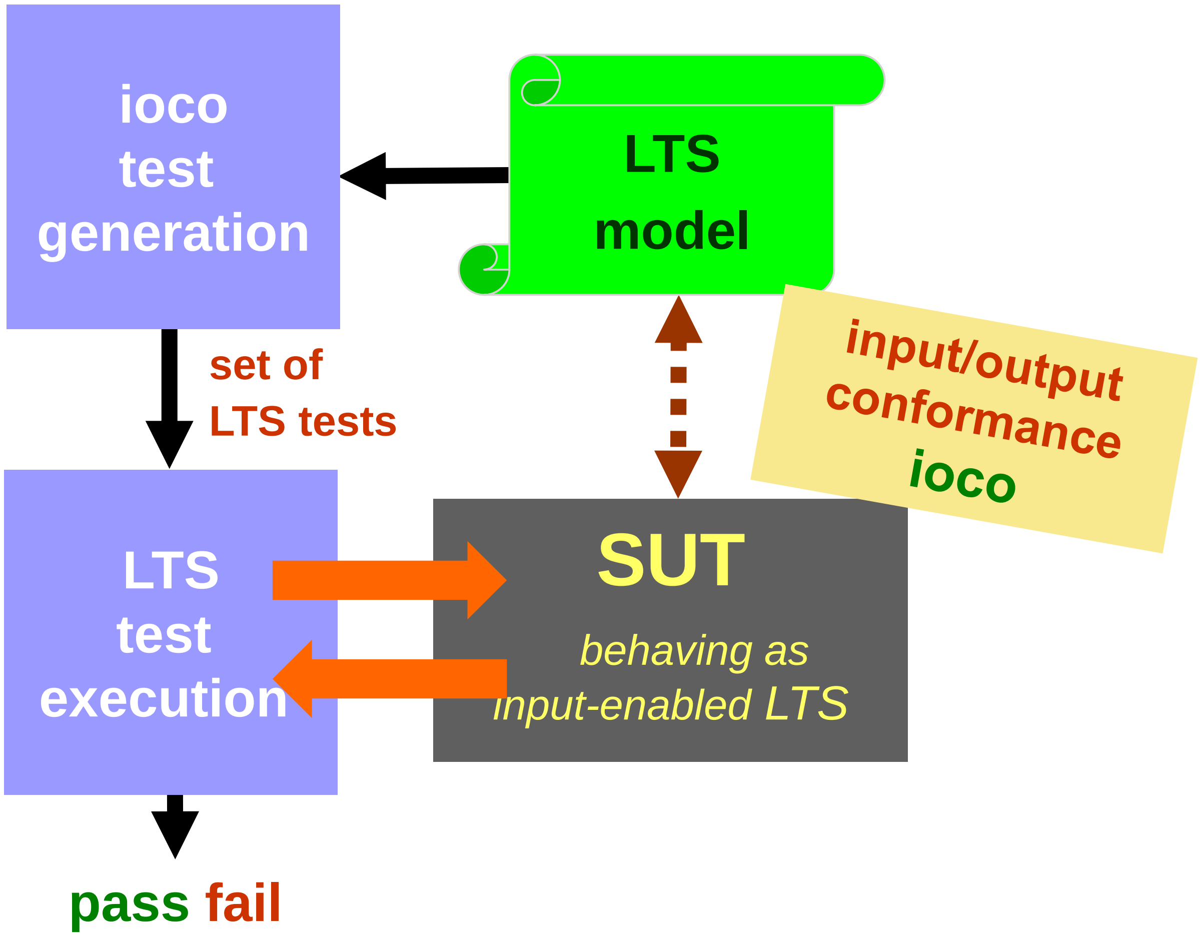
TorXakis

TorXakis : LTS & ioco

SUT ioco model

sound ⇓ ⇑ *exhaustive*

SUT passes tests



ioco test generation

LTS model

set of LTS tests

input/output conformance ioco

LTS test execution

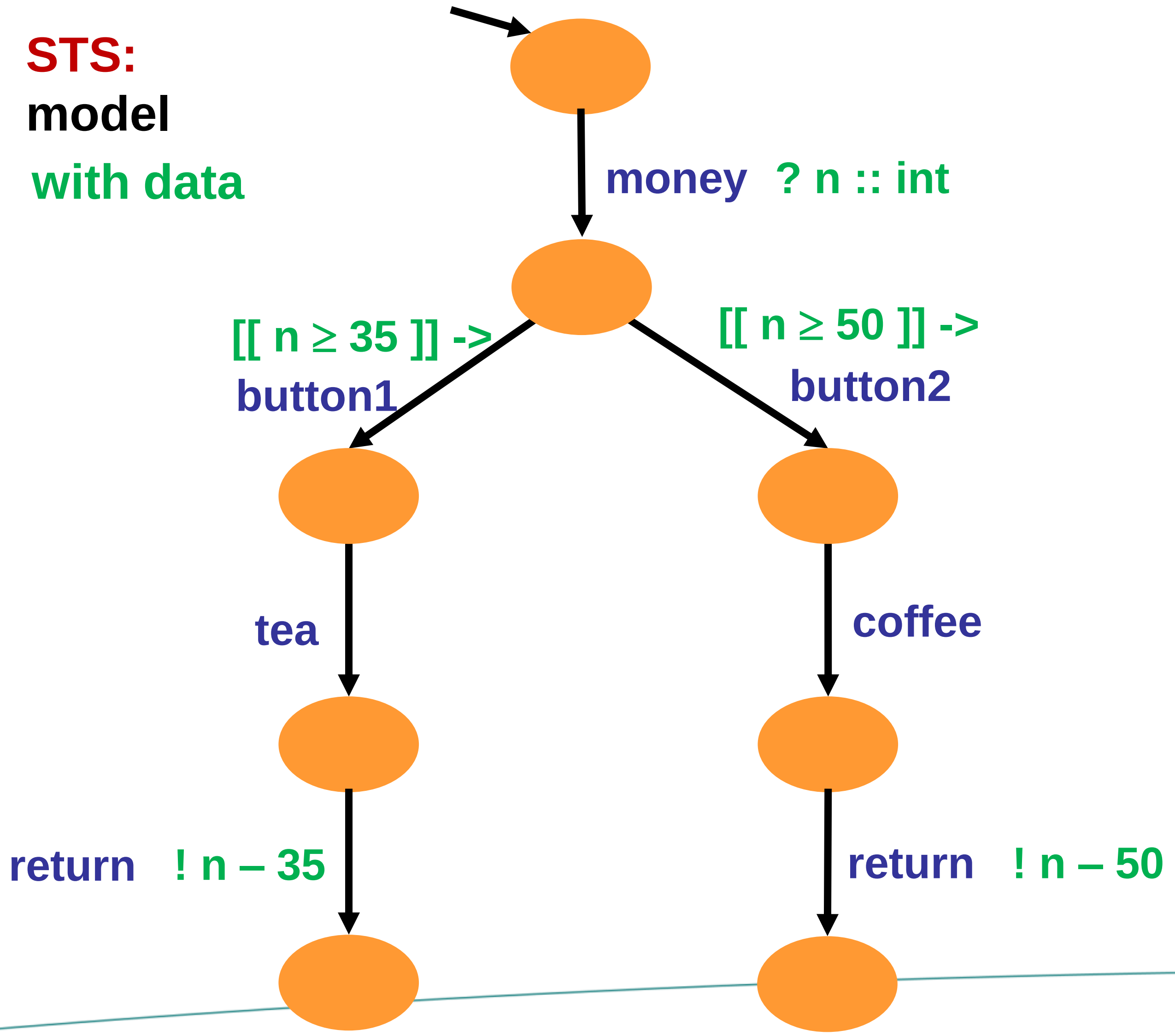
SUT

behaving as input-enabled LTS

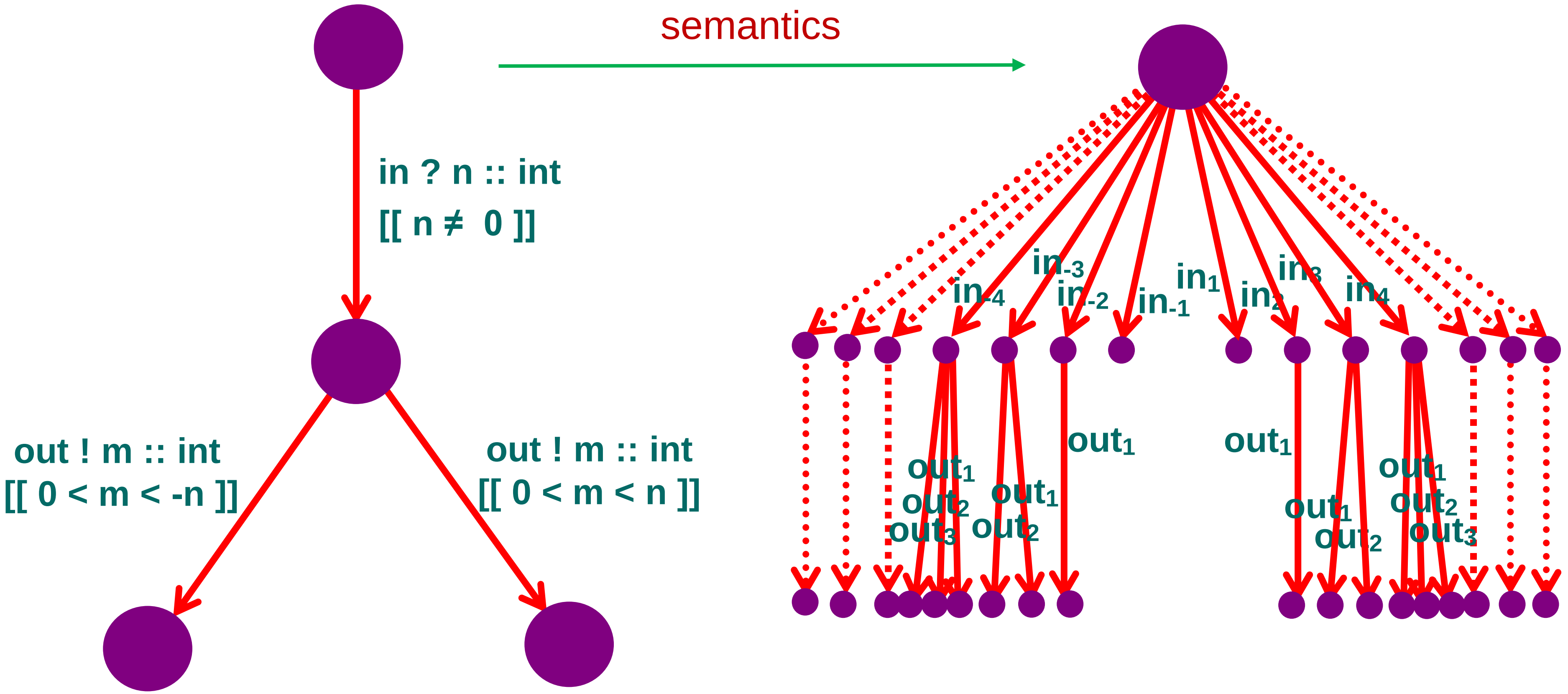
pass fail

STS : Symbolic Transition Systems

STS:
model
with data



STS : Symbolic Transition Systems



in ? n :: int
[[n ≠ 0]]

out ! m :: int
[[0 < m < -n]]

out ! m :: int
[[0 < m < n]]

semantics

in$_{-4}$ in$_{-3}$ in$_{-2}$ in$_{-1}$ in$_{1}$ in$_{2}$ in$_{3}$ in$_{4}$

out$_{1}$ out$_{2}$ out$_{3}$

out$_{1}$ out$_{2}$

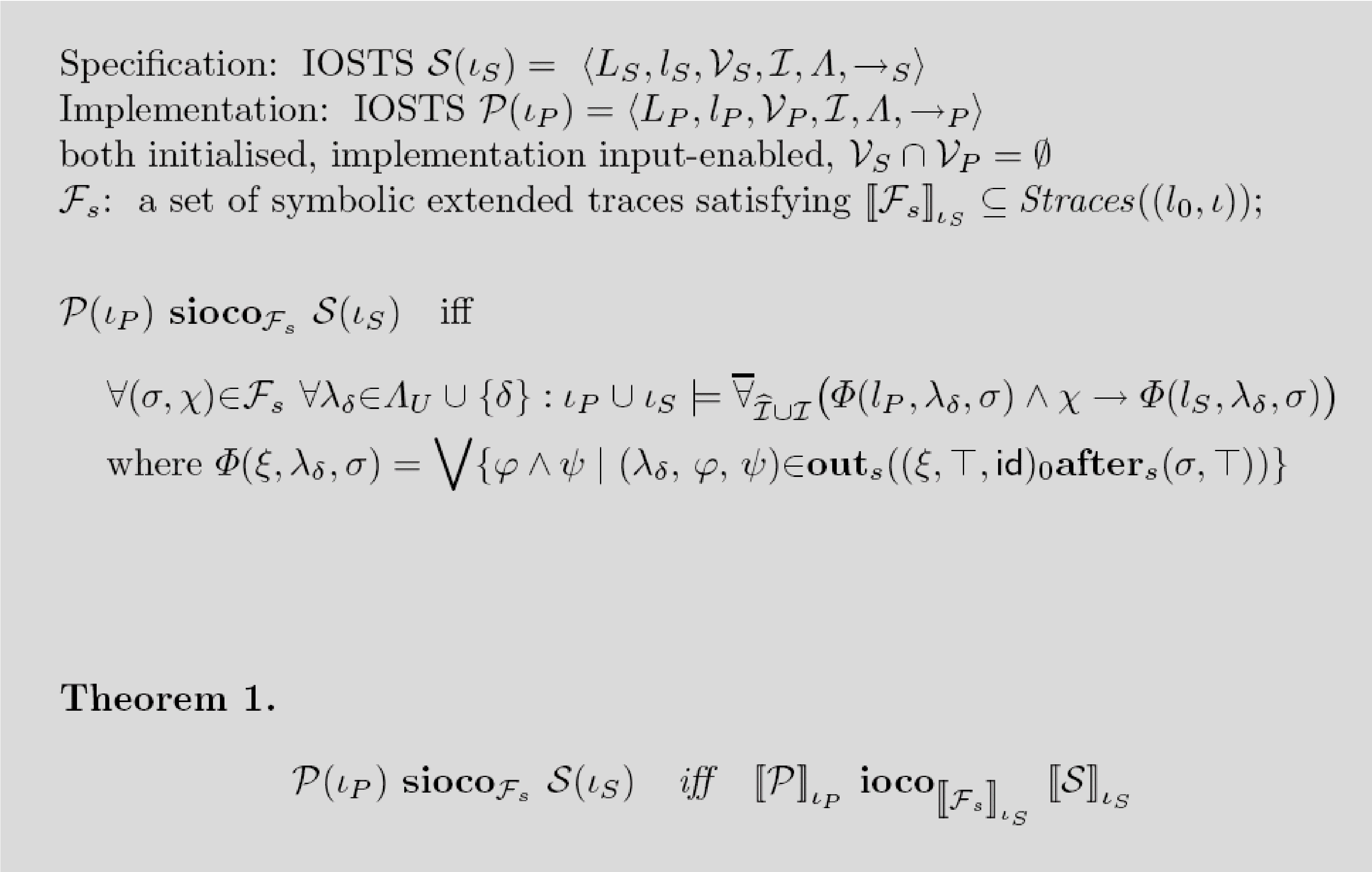
out$_{1}$

out$_{1}$

out$_{1}$ out$_{2}$

out$_{1}$ out$_{2}$ out$_{3}$

sioco : Symbolic ioco



Specification: IOSTS $\mathcal{S}(\iota_S) = \langle L_S, l_S, \mathcal{V}_S, \mathcal{I}, \Lambda, \rightarrow_S\rangle$
Implementation: IOSTS $\mathcal{P}(\iota_P) = \langle L_P, l_P, \mathcal{V}_P, \mathcal{I}, \Lambda, \rightarrow_P\rangle$
both initialised, implementation input-enabled, $\mathcal{V}_S \cap \mathcal{V}_P = \emptyset$
$\mathcal{F}_s$: a set of symbolic extended traces satisfying $[\![\mathcal{F}_s]\!]_{\iota_S} \subseteq \mathit{Straces}((l_0, \iota))$;

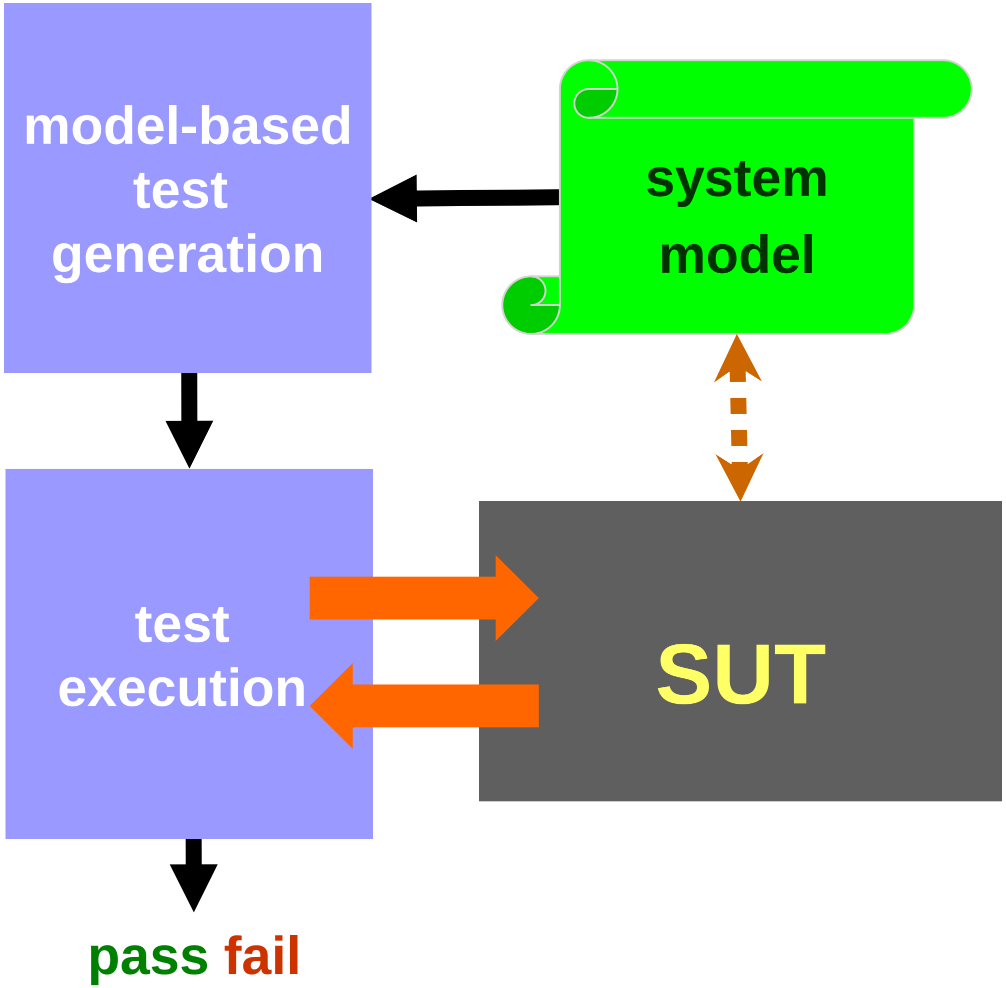
$\mathcal{P}(\iota_P)\ \mathbf{sioco}_{\mathcal{F}_s}\ \mathcal{S}(\iota_S)$ iff

$$\forall(\sigma,\chi)\in\mathcal{F}_s\ \forall\lambda_\delta\in\Lambda_U \cup \{\delta\} : \iota_P \cup \iota_S \models \overline{\forall}_{\hat{\mathcal{I}}\cup\mathcal{I}}(\Phi(l_P,\lambda_\delta,\sigma) \wedge \chi \rightarrow \Phi(l_S,\lambda_\delta,\sigma))$$
$$\text{where } \Phi(\xi,\lambda_\delta,\sigma) = \bigvee\{\varphi \wedge \psi \mid (\lambda_\delta, \varphi, \psi)\in\mathbf{out}_s((\xi,\top,\mathsf{id})_0\mathbf{after}_s(\sigma,\top))\}$$

Theorem 1.

$$\mathcal{P}(\iota_P)\ \mathbf{sioco}_{\mathcal{F}_s}\ \mathcal{S}(\iota_S) \quad \textit{iff} \quad [\![\mathcal{P}]\!]_{\iota_P}\ \mathbf{ioco}_{[\![\mathcal{F}_s]\!]_{\iota_S}}\ [\![\mathcal{S}]\!]_{\iota_S}$$

MBT Tools



model-based test generation

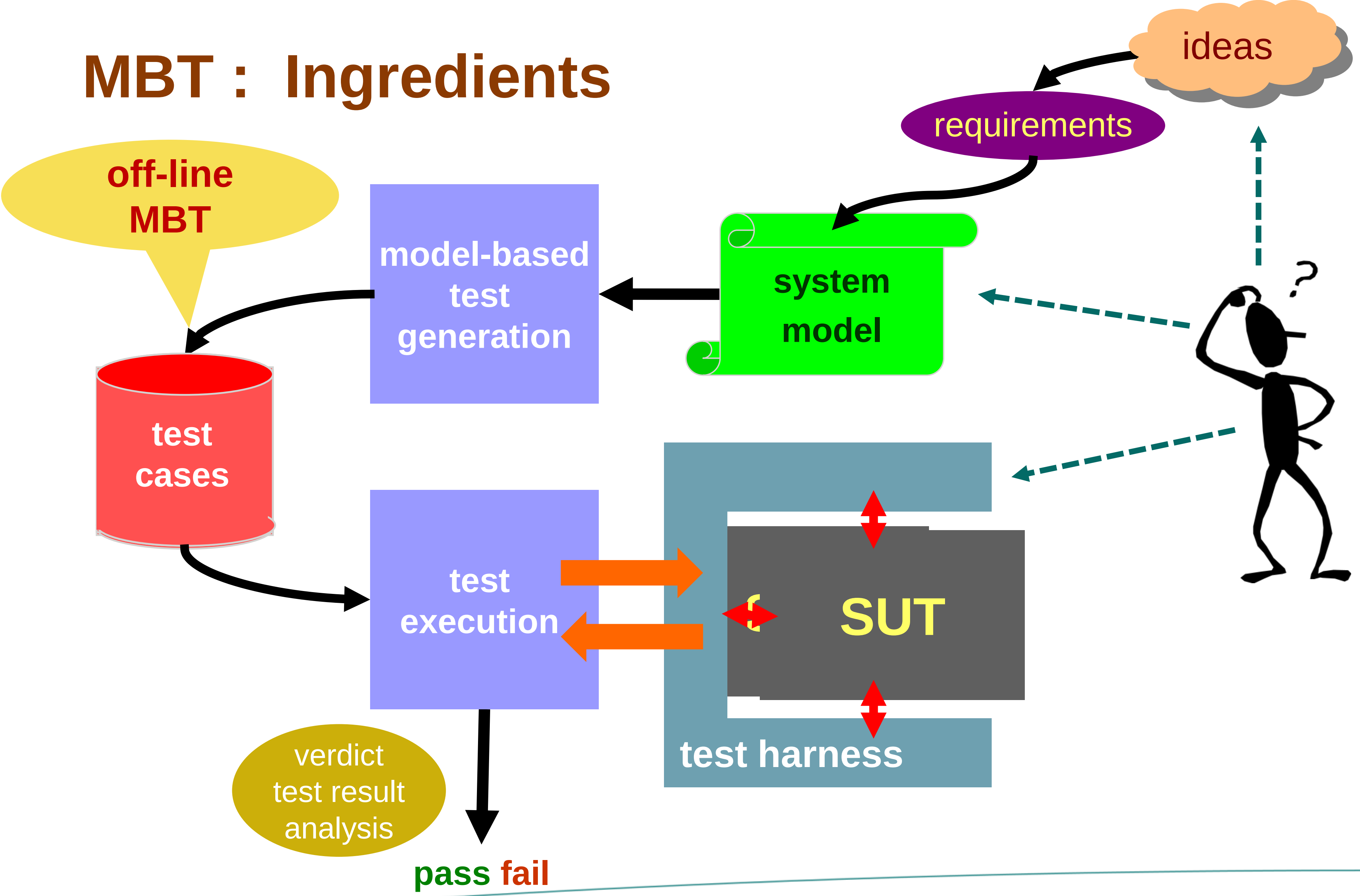
system model

test execution

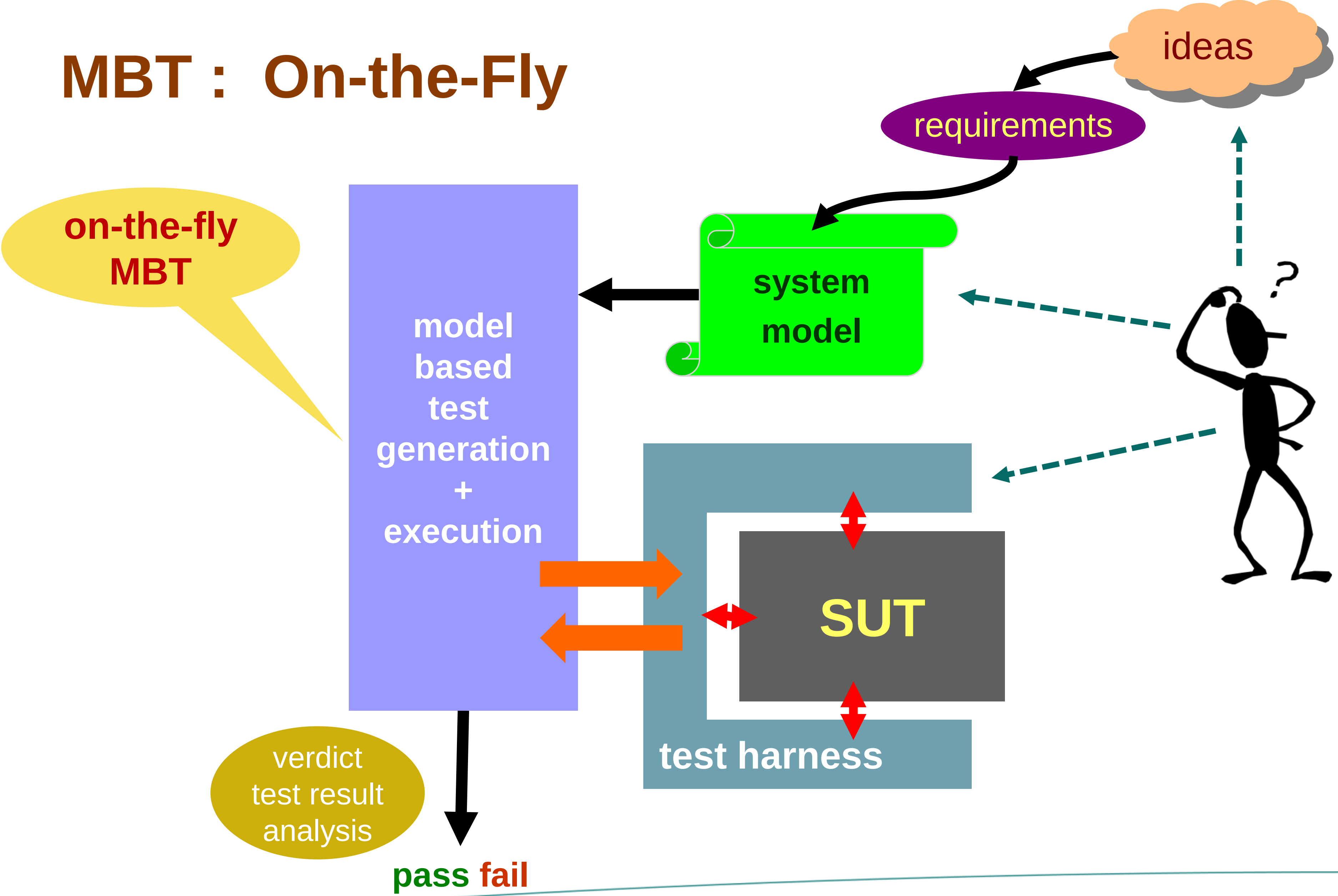
SUT

pass fail

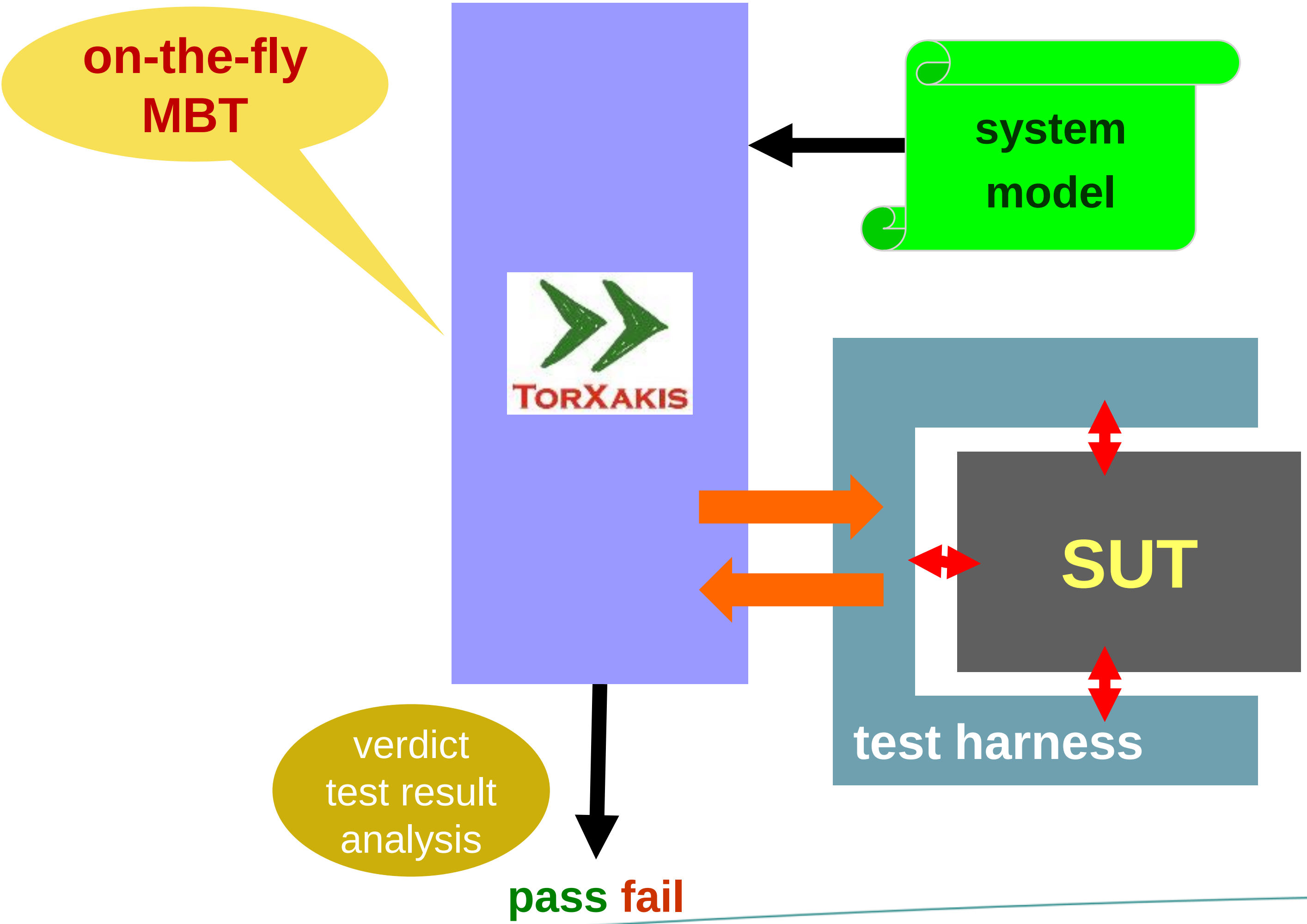
MBT : Ingredients



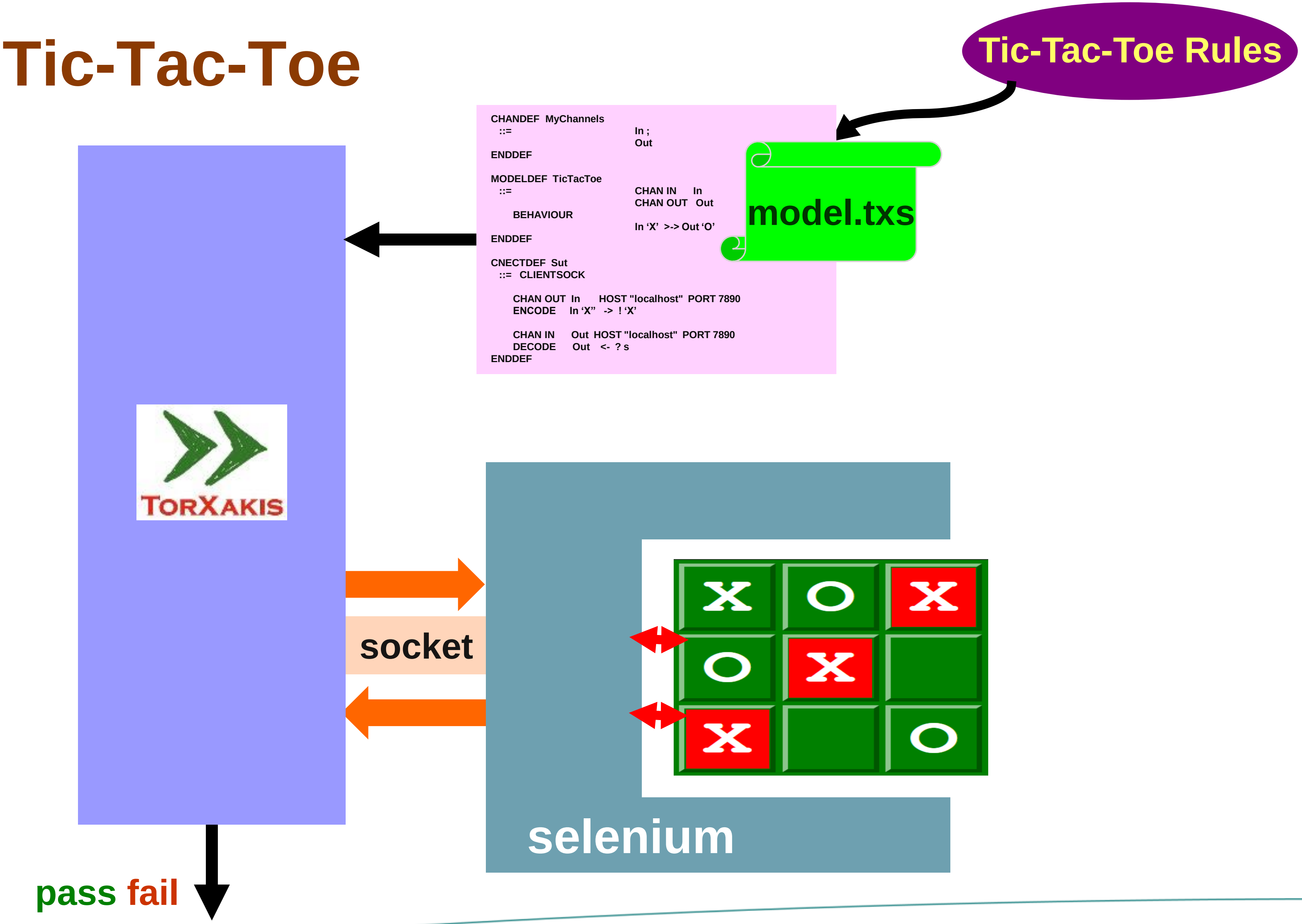
MBT : On-the-Fly


ideas
requirements
on-the-fly MBT
model based test generation + execution
system model
SUT
test harness
verdict test result analysis
pass fail

TorXakis : An On-the Fly MBT Tool



Tic-Tac-Toe



```
CHANDEF  MyChannels
  ::=                                In ;
                                     Out
ENDDEF

MODELDEF  TicTacToe
  ::=                                CHAN IN     In
                                     CHAN OUT   Out
      BEHAVIOUR
                                     In ‘X’  >-> Out ‘O’
ENDDEF

CNECTDEF  Sut
  ::=   CLIENTSOCK

      CHAN OUT  In       HOST "localhost"  PORT 7890
      ENCODE     In ‘X”   ->  ! ‘X’

      CHAN IN      Out  HOST "localhost"  PORT 7890
      DECODE      Out    <-  ? s
ENDDEF
```

X	O	X
O	X	
X		O

TorXakis : Overview

Models

- process-algebraic modelling language
- state-based control flow and complex data
- support for parallel, concurrent systems
- composing complex models from simple models
- non-determinism, uncertainty
- abstraction, under-specification

Applications

- several high-tech systems companies
- experimental level

But ....

- research prototype
- poor usability

Tool

- on-line MBT tool

Current Research

- test selection
- variability, features
- modelling
- integration in process

Under the hood

- SMT solvers for constraints and data generation (via SMT-LIB: Z3, CVC4)
- testing theory: **sioco** on STS
- algebraic data-type definitions with rewriting
- Haskell
- LPE: Linear process equations
- Other MBT tools for testing (QuickCheck)

Model-Based Testing

Applications

Electronic Passport

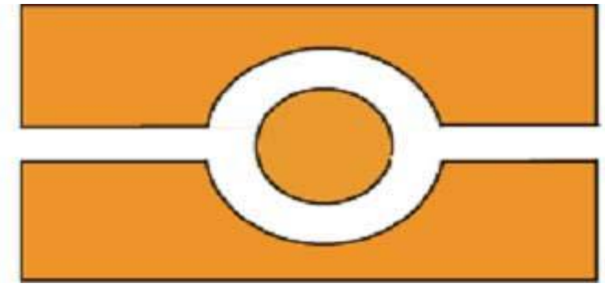
New Passport

- Machine Readable Passport (MRP, E-passport)
- with chip (JavaCard), contact-less
- storage of picture, fingerprints, iris scan, ...
- access to this data protected by encryption and a new protocol
- few years ago released in EU

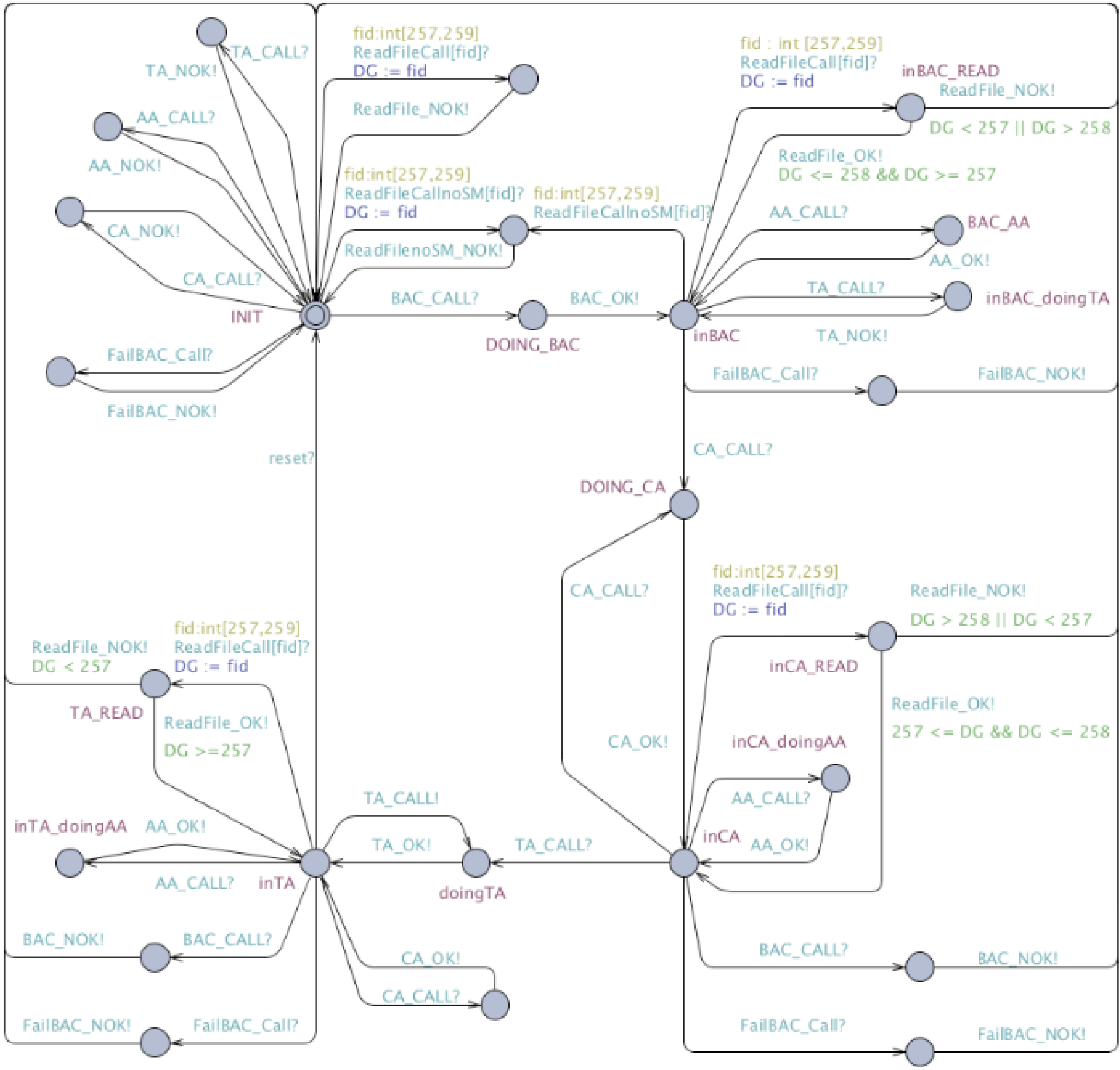
Our job: testing of e-passports

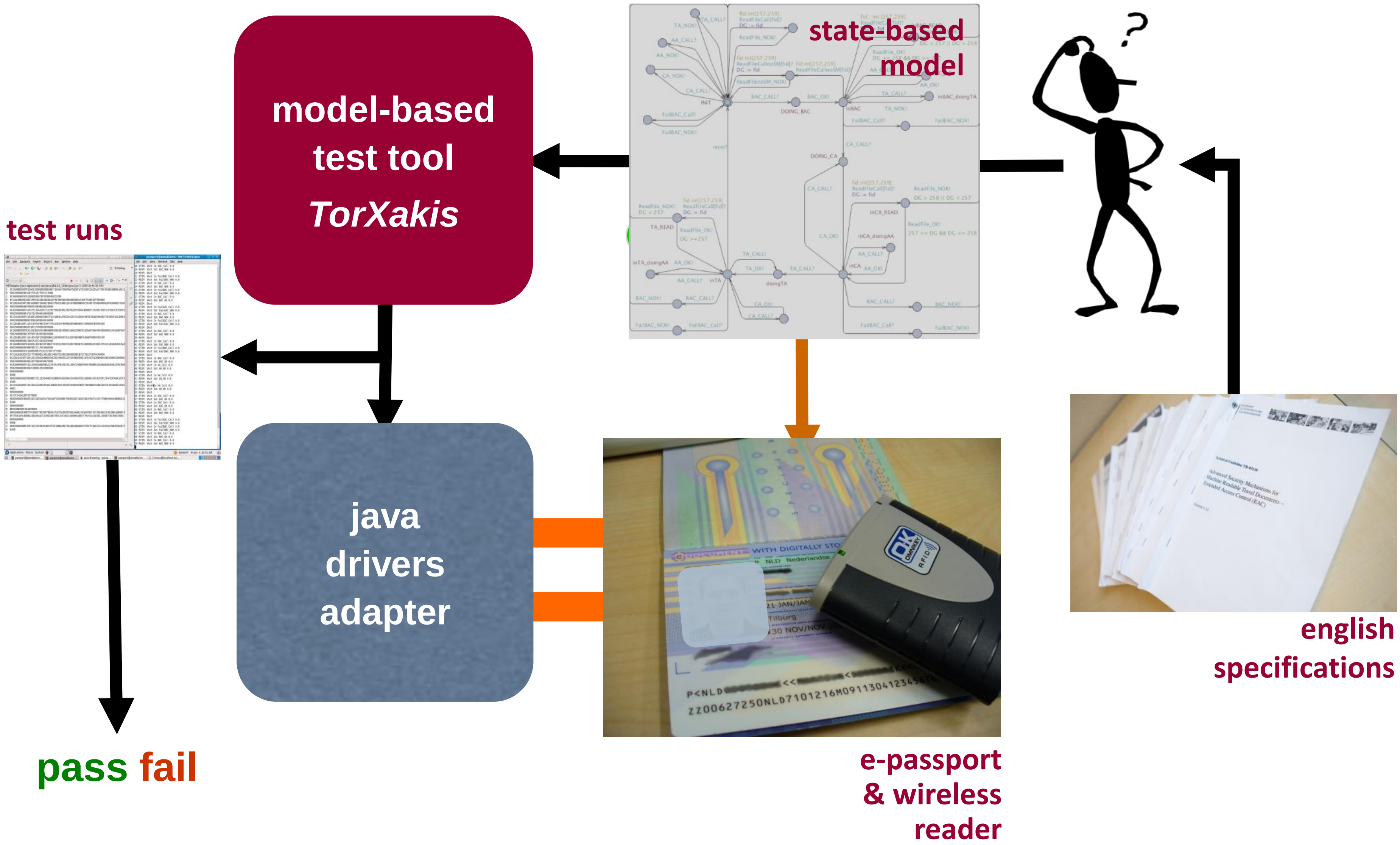
- emphasis on access protocol

  == exchange of request-respons messages
  between passport and reader (terminal)



MBT for E-Passports : Model



test runs
model-based
test tool
TorXakis
state-based
model
java
drivers
adapter
english
specifications
e-passport
& wireless
reader
pass fail

MBT for E-Passports : Results

- Tested:
  - Basic Access Control (BAC)
  - Extended Access Control (EAC)
  - Active Authentication (AA)
  - Data Reading
- Tests up to about 2,000,000 test events
  - complemented with manual tests
- No error found ......

MBT in High-Tech Embedded Systems

océ

A CANON COMPANY

PHILIPS

ASML

MBT in High-Tech Embedded Systems

Systems

- large, complex, system-of-systems
- complex state + complex data
- variability, product line
- not always up-to-date specifications
- compositional
- parallelism, under-specification
- uncertainty, non-determinism,

SUT

- testing on simulated SUT:

  virtual system, *digital twin*

Models

- how to make models ?
- who makes models? : *Testers*
- DSL (Domain Specific Languages)
- construct model from tests

Testing

- state of practice:

  keyword-driven test automation
- instrumentation: existing

  keyword-driven test automation
- test selection via usage-profiles

Model-Based Testing

Using TorXakis

Model-Based Testing

Theory, Tools, Applications

- *MBT: the next step*

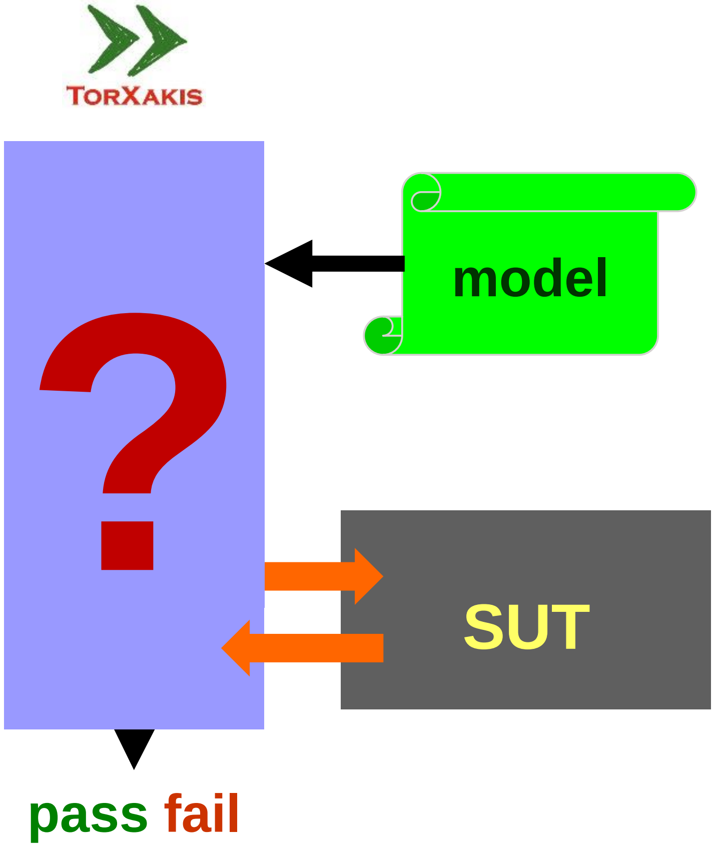
  in test automation ! ?

- *The future of testing*

  is model-based ! ?

- *If not, what is*

  the alternative ?


TorXakis
?
model
SUT
pass fail