Model-Based Testing

The Next Step
in Test Automation ! ?

Jan Tretmans

**SUMBAT:** **SUPersizing**
**Model**
**BAased**
**Testing**

TorXakis

SUT

pass fail

*TNO – Embedded Systems Innovation*
*Radboud University, Nijmegen*
Overview

Model-Based Testing:

• Motivation
• What
• Model-Based Development
• Status
• Research
• A Tool
• Example
Embedded Systems
or: What do Dutch Dykes have to do with Model-Based Testing?
Embedded Systems

or: What do Dutch Dykes have to do with Model-Based Testing?
Testing High-Tech 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
Quality Software

Exclusive: Pentagon report faults F-35 on software, reliability

Software glitch hindered Vettel’s qualifying

2014 Australian Grand Prix

March 15, 2014 at 10:00 am by Keith Collantine

Sebastian Vettel’s qualifying performance was impaired by a software problem on his car, Red Bull have confirmed.

Vettel was knocked out in the second phase of qualifying having been 2.4 seconds off new team mate Daniel Ricciardo’s pace.

A faulty sensor was later found to have contributed to Vettel’s problem. Team principal Christian Horner said: “It was unlucky for Seb.”

“His engine software meant he was down on power with extremely poor drivability and we need to understand that, as it compromised his qualifying.”

Despite his difficulties Vettel said the team have “made a big
Testing Challenges: Complexity

Testing effort grows exponentially with system size

Testing cannot keep pace with development

- $x: [0..9]$ → 10 ways that it can go wrong
  - 10 combinations of inputs to check
- $x: [0..9]$ → $y: [0..9]$ → 100 ways that it can go wrong
  - 100 combinations of inputs to check
- $y: [0..9]$ → 1000 ways that it can go wrong
  - 1000 combinations of inputs to check
- $x: [0..9]$ → $y: [0..9]$ → $z: [0..9]$ → combinatorial explosion of required testing effort

Testing Challenges: Complexity
Testing Challenges: Complexity

Testing effort grows exponentially with system size

Testing cannot keep pace with development

→ combinatorial explosion of required testing effort
Testing Challenges: Components

Components come from anywhere
Testing Challenges: Uncertainty

- Sometimes you don’t know …..
  - testing a search engine,
    weather forecast, …
  - systems-of-systems,
    big data, …

- Sometimes you don’t want to know ….
  - no details
  - abstraction
  - particular view

Uncertainty of test outcomes & oracles

- non-determinism,
  probabilities, constraints
Trends & Challenges

- complexity
- connectivity
- multi-disciplinarity
- heterogeneous components
- model based testing
- change
- variability
- evolvability
- uncertainty
Model-Based Testing
Model-Based Testing

**MBT**

next step in test automation:

+ test generation
+ result analysis
1. Manual testing

![Diagram showing System Under Test (SUT) and pass/fail outcomes]
2: Scripted Testing

1. Manual testing
2. Scripted testing

Test cases

Test execution

SUT

Pass fail
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 : Example Models
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 : Many 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
- Temppo
- TestGen (Stirling)
- TestGen (INT)
- TestComposer
- TestOptimal
- TGV
- Tigris
- TorX
- TorXakis
- T-Vec
- Uppaal-Cover
- Uppaal-Tron
- Tveda

...
Model-Based
Verification, Validation, Testing, . . . . .
Validation, Verification, Testing

Informal requirements → Validation → Formal world → Model → Verification → Model-based testing → Real world → SUT
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
Code Generation from a Model

A model is more \textit{(less)} than code generation:

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

- abstract (test) models
- design models
- realization
- virtualization
Code Generation from a Model

\[ y \times y = x \]

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

model of \( \sqrt{x} \)
Model-Based Testing
Status
MBT : Benefits?

MBT : State of the Art
• promising, emerging
• a number of successful applications
• many companies are experimenting

MBT : State of Practice
Reasons
• technical
• tools
• organizational
• maturity of testing
• educational
• ...

MBT : State for the Future
(for High-tech Embedded Systems)
• ?

But ....
*If doing MBT is so smart, why ain’t you rich?*
Testing High-Tech Systems
Trends & Challenges

- Complexity
- Connectivity
- Change
- Variability
- Evolvability
- Uncertainty
- Multi-disciplinarity
- Heterogeneous components

Model-based testing
MBT: Next Generation Challenges

- abstraction
- concurrency parallelism
- state + complex data
- usage profiles for testing
- model composition
- scalability
- test selection criteria
- link to MBSD
- multiple paradigms integration
- uncertainty nondeterminism
Model-Based Testing
Research
SUMBAT
SUMBAT

SUpersizing
Model
BAsed
Testing
SUMBAT goal

- **SU**persizing **Model-BA**sed **T**esting:

  making MBT applicable
to large, complex, high-tech embedded software systems, with

  - millions of lines of code,
  - distribution, concurrency,
  - complex data- and state-based behaviour,
  - uncertainty, non-determinism, and
  - complex and heterogeneous interfaces
SUMBAT method

- **theory**
  - LTS
  - ioco
  - automata learning
  - test selection
  - ...

- **tools**
  - TorXakis
    - LTSmin
    - LearnLib
    - ...

- **applications**
- Cases:
  - ASML
  - Océ
  - PANalytical
  - ...

- **Canada**
Next Generation MBT: TorXakis

- Abstraction
- Concurrency parallelism
- State + complex data
- Model composition
- Test selection criteria
- Multiple paradigms integration
- Uncertainty nondeterminism
- Usage profiles for testing
- Scalability
- Link to MBSD
TorXakis

A bit more detail
TorXakis: A Black-Box View on Systems

- Modelled as state-transition system
- Black-box system view

Diagram:
- Input: a
- Output: x
- a?n -> x!n+1
- y!`yes`
- b?m
- a?n
- x!42
- y!`no`
- a
- b
- modelled as state-transition system
TorXakis: Testing Theory

SUT conforms to model

sound

SUT passes tests

model-based test generation

system model

input/output conformance ioco

test execution

pass fail
TorXakis: On-the-Fly MBT
### TorXakis: Testing Theory \textit{ioco}

\[
i \text{ ioco } \ s \ =_{\text{def}} \ \forall \ \sigma \in \text{Straces} (s) : \ \text{out} (i \ \text{after} \ \sigma) \subseteq \text{out} (s \ \text{after} \ \sigma)
\]

\text{Straces} (s) = \{ \ \sigma \in (L \cup \{\delta\})^* \ | \ s \xrightarrow{\sigma} \} \]

\text{p after} \ \sigma = \{ \ p' \ | \ p \xrightarrow{\sigma} p' \} \]

\text{out} (P) = \{ \ !x \in L_U | p \xrightarrow{\delta} !x, p \in P \} \cup \{ \ \delta | p \xrightarrow{\delta} p, p \in P \} \]
TorXakis: Overview

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

Tool
• on-line MBT tool

Current Research
• scalability
• test selection

But ....
• research prototype
• poor usability

Under the hood
• powerful constraint/SMT solvers (Z3, CVC4)
• well-defined semantics and algorithms
• ioco testing theory for symbolic transition systems
• algebraic data-type definitions

https://github.com/TorXakis
Model-Based Testing

Research

Test Selection
Test Selection

• Exhaustiveness never achieved in practice

• Test selection = select subset of exhaustive test suite, to achieve confidence in quality of tested product
  – select best test cases capable of detecting failures
  – measure to what extent testing was exhaustive: coverage

• Optimization problem

best possible testing ↔ within cost/time constraints
Testing and Quality

- Quality-assurance costs
- Remaining-defects costs

\[ \text{Cost} \]

\[ \# \text{Test cases} \]
Test Selection

Extra information / domain information required:

- which test cases have high value?
- which errors are likely?
- which errors have high impact?
- what is the user / customer doing?

\[ \text{risk} \]

\[ \text{usage profiling} \]
\[ \text{statistical testing} \]
Model-Based Testing

Research

How to Get these D… Models
Testing: Model-Based Testing
Test-Based Modeling

Model Learner

model

system

test execution

Embedded Systems Innovation
BY TNO
TorXakis
Demo
Tic-Tac-Toe

Tic-Tac-Toe Rules

TorXakis

socket

pass fail

selenium
Model-Based Testing

The Next Step in Test Automation! ?

**MBT:**

- *Is it the promising future of software testing?*
- *Can we do without MBT?*
- *If not MBT, what then?*

https://github.com/TorXakis