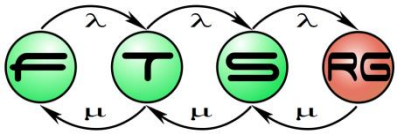
Model-based test generation

Zoltan Micskei, Istvan Majzik

Budapest University of Technology and Economics
Fault Tolerant Systems Research Group



Main topics of the course

- Overview (1.5)
  - Introduction, V&V techniques
- Static techniques (1.5)
  - Specification, Verifying source code
- Dynamic techniques: Testing (7)
  - Testing overview, Test design techniques
  - **Test generation**, Automation
- System-level verification (3)
  - Verifying architecture, Dependability analysis
  - Runtime verification

Learning outcomes

- Illustrate how models can be used in testing (K2)
- Explain the typical model-based test generation process (K2)
- Apply different selection criteria to finite state machines to select test cases (K3)
- Use an MBT tool to generate test cases (K3)

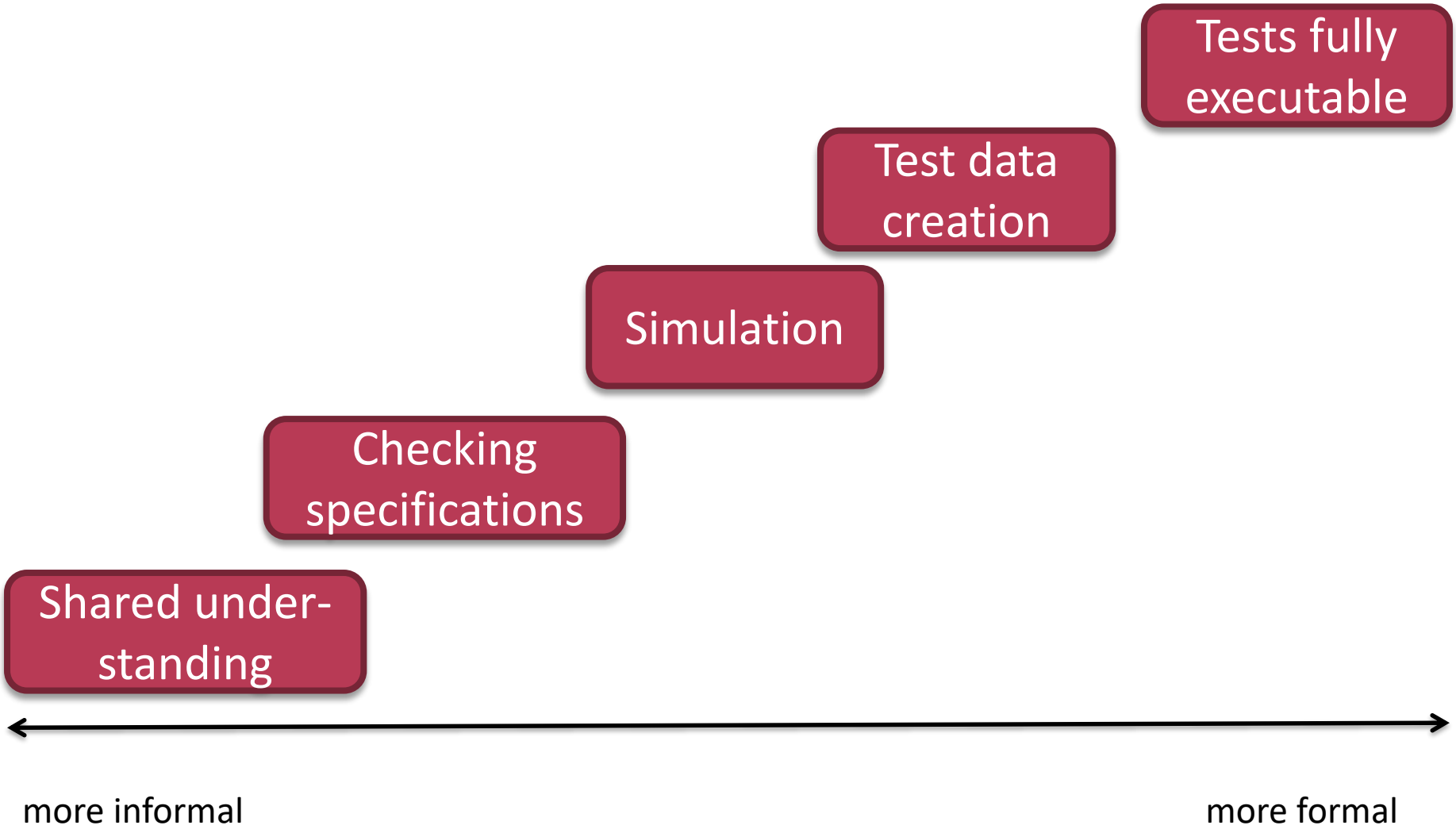
What is model-based testing?

“Testing based on or involving models” [ISTQB]

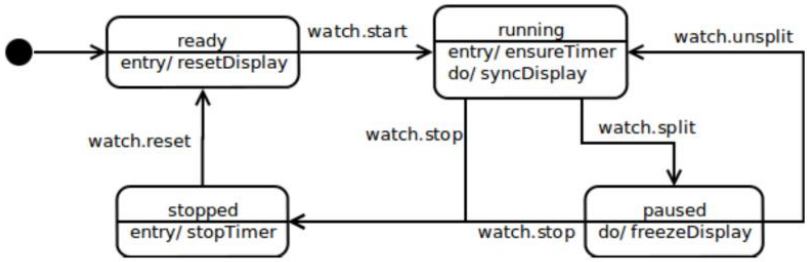
- Not just test generation
- Not just automatic execution
- Not just for model-driven engineering

Source of definition: ISTQB. “Foundation Level Certified Model-Based Tester Syllabus”, Version 2015

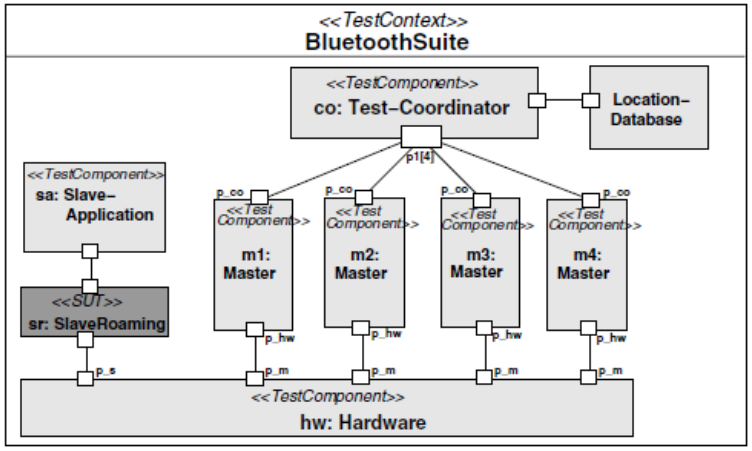
Landscape of MBT goals



Using models in testing (examples)



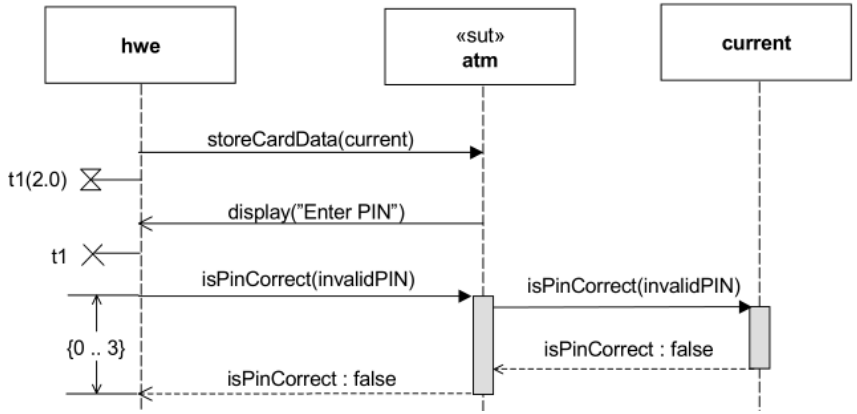
Behavior of SUT



Test configuration

```
timer t;
t.start(5.0);
alt {
 [] i.receive("coffee") {
  Count := Count+1; }
 [] t.timeout { }
}
```

Test sequences



Test sequences

Source: OMG UTP

Benefits of using models

- Close communication with stakeholders
  - Understanding of domain and requirements
- Early testing: modeling/simulation/generation
- Higher abstraction level (manage complexity)
- Automation (different artefacts)

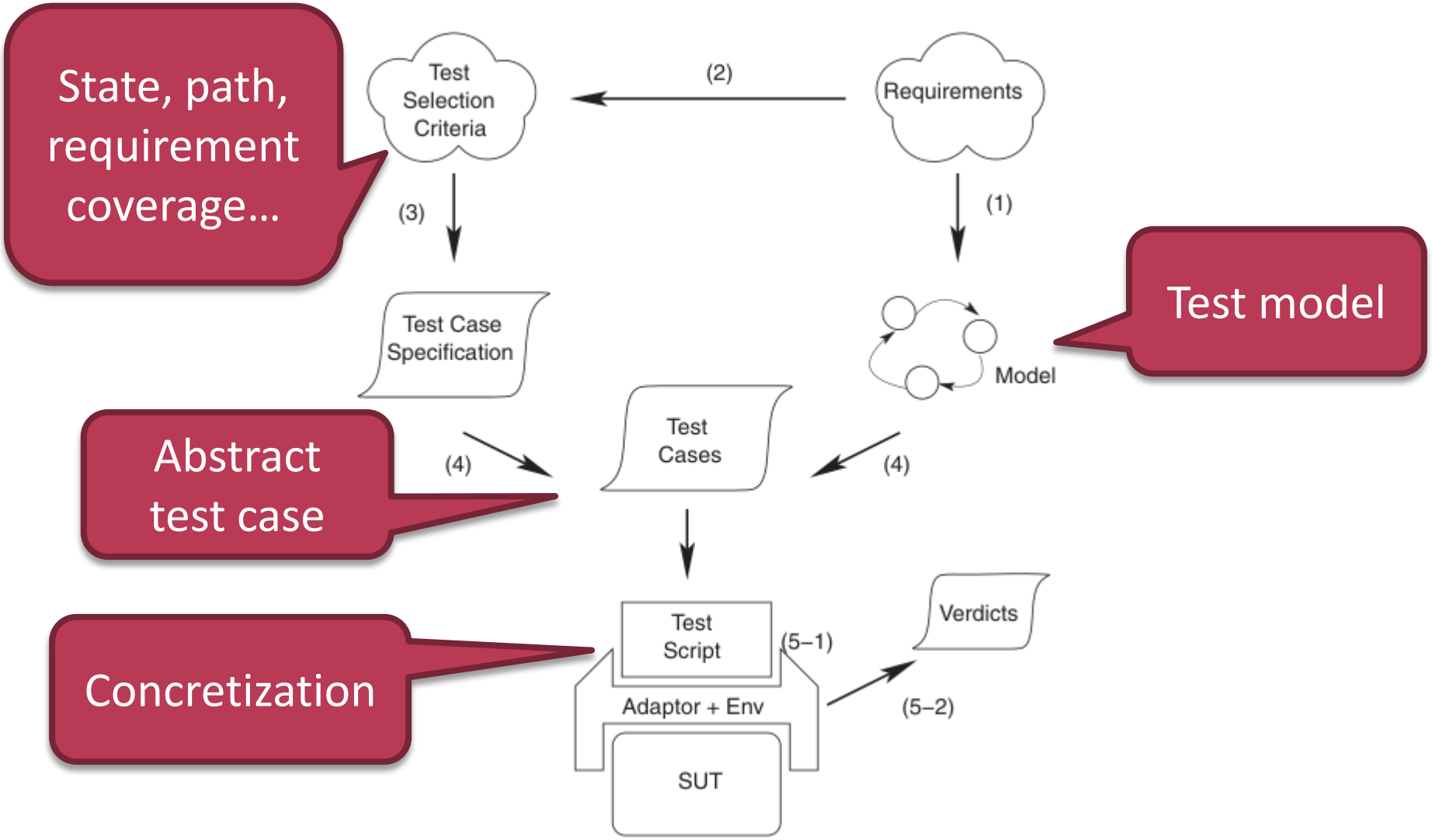
More specific meaning: Test generation

„MBT encompasses the processes and techniques for

- the automatic derivation of abstract test cases from abstract models,
- the generation of concrete tests from abstract tests,
- the manual or automated execution of the resulting concrete test cases”

Source: M. Utting, A. Pretschner, B. Legeard. „A taxonomy of model-based testing approaches”, STVR 2012; 22:297–312

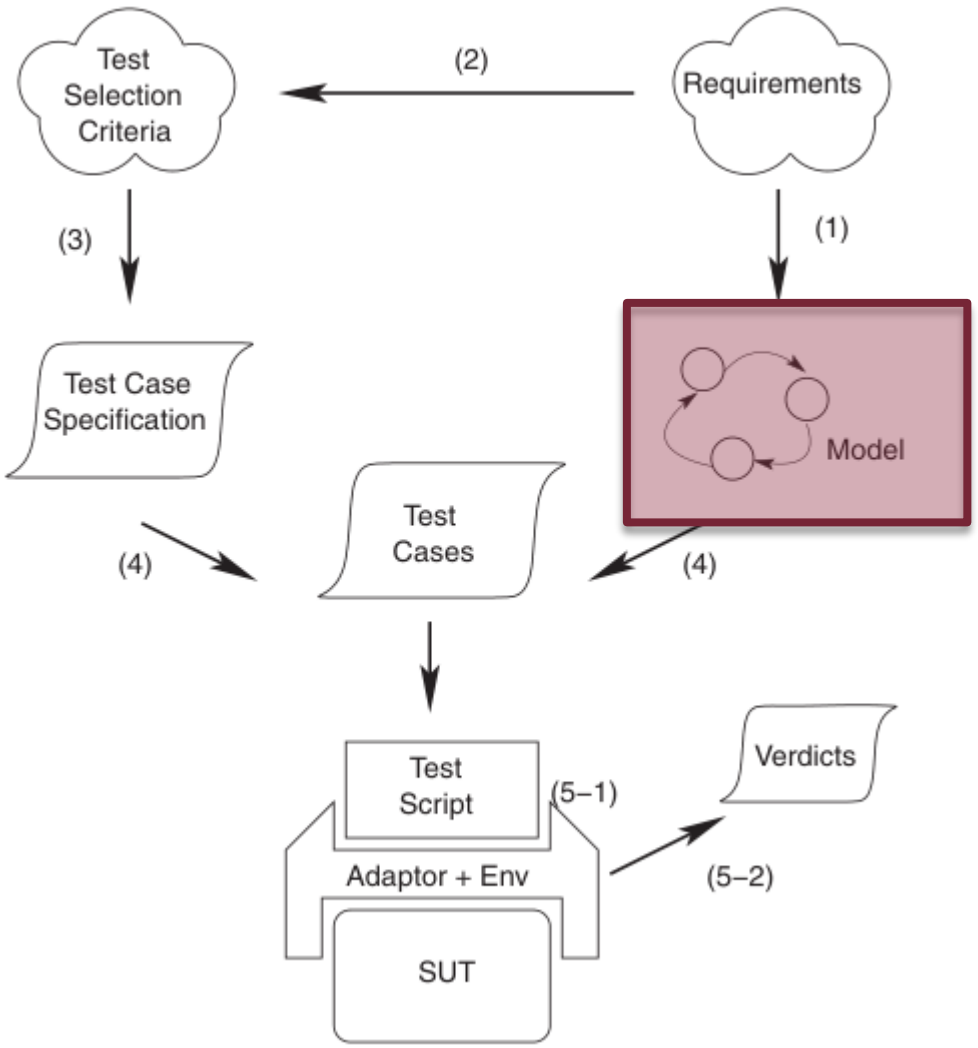
Typical MBT process



Source: M. Utting, A. Pretschner, B. Legeard. „A taxonomy of model-based testing approaches”, STVR 2012; 22:297–312

MBT PROCESS

Typical MBT process



Source: M. Utting, A. Pretschner, B. Legeard. „A taxonomy of model-based testing approaches”, STVR 2012; 22:297–312

Questions for modeling

- What to model?
  - What is the test object?
  - Functionality / performance factors / …
- What abstraction level to use?
  - Too many or too few details
  - Separate models for different test objectives
- What modeling language to use?
  - Structural, behavioral

Focus of the model

System

- System as intended to be
- Conformance of model-SUT

Usage

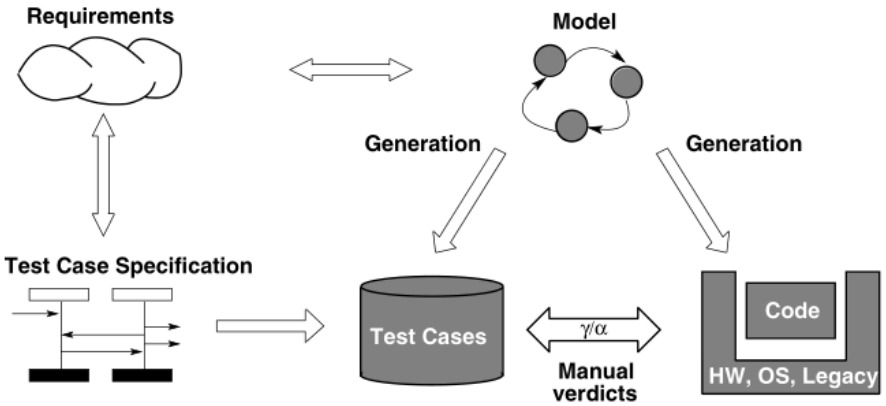
- Model environment/users
- Inputs to the system

Test

- Model one or more test case
- E.g. sequences + evaluation

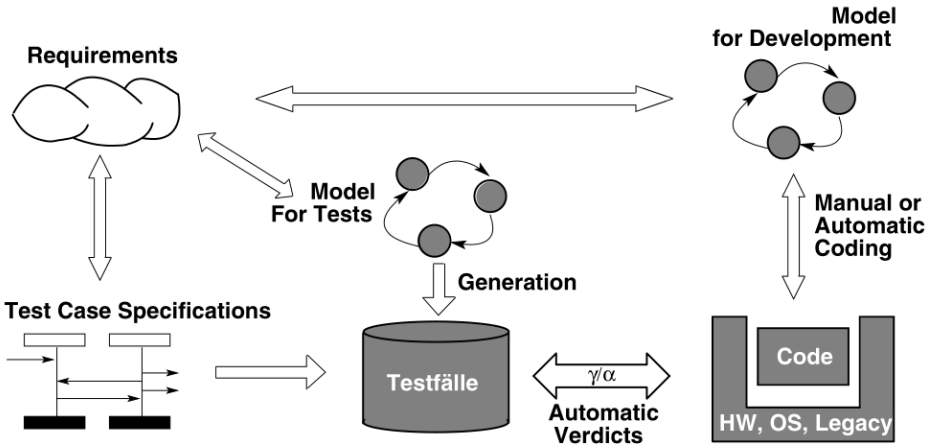
Reuse: Development and Test modeling

What if I have existing design models?



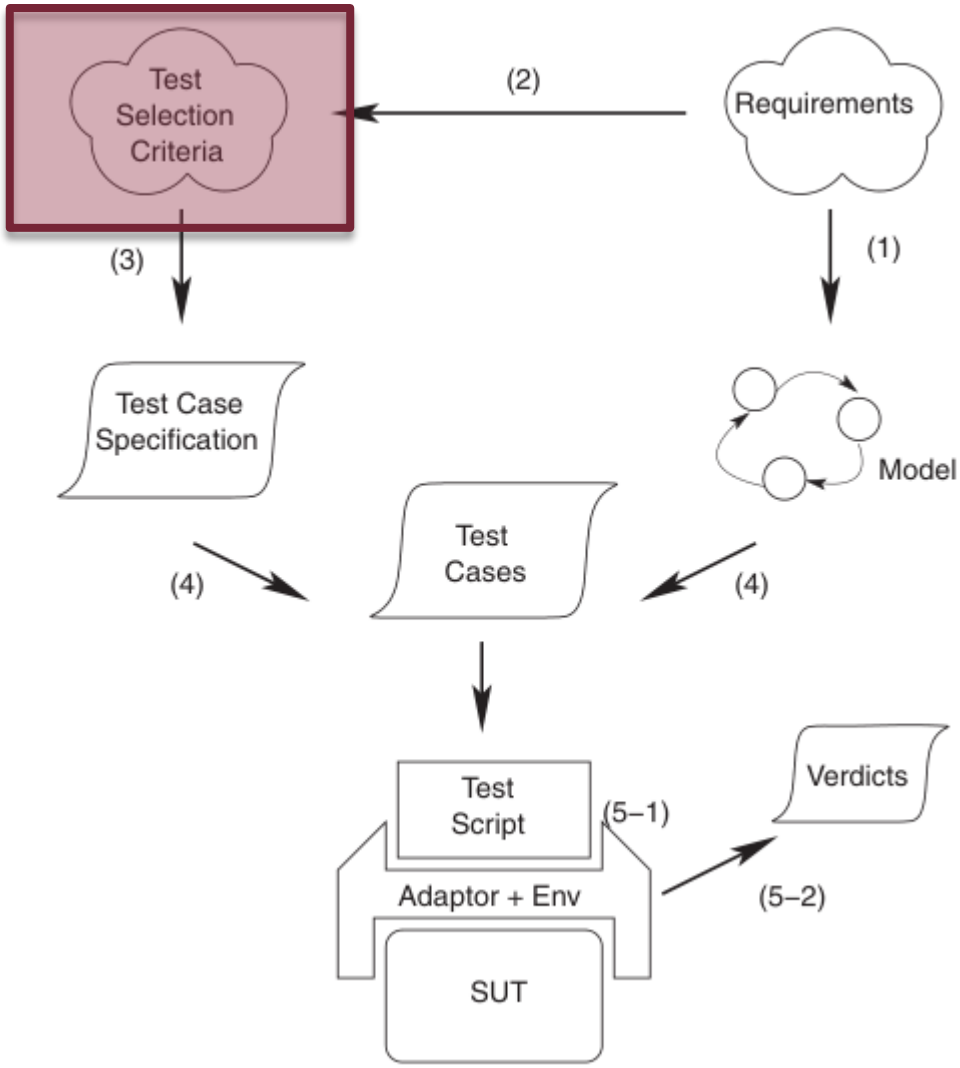
Problem: what do we test here?

Approach: separate dev. and test models



A. Pretschner, J. Philipps. „Methodological Issues in Model-Based Testing", Model-Based Testing of Reactive Systems, 2005.

Typical MBT process

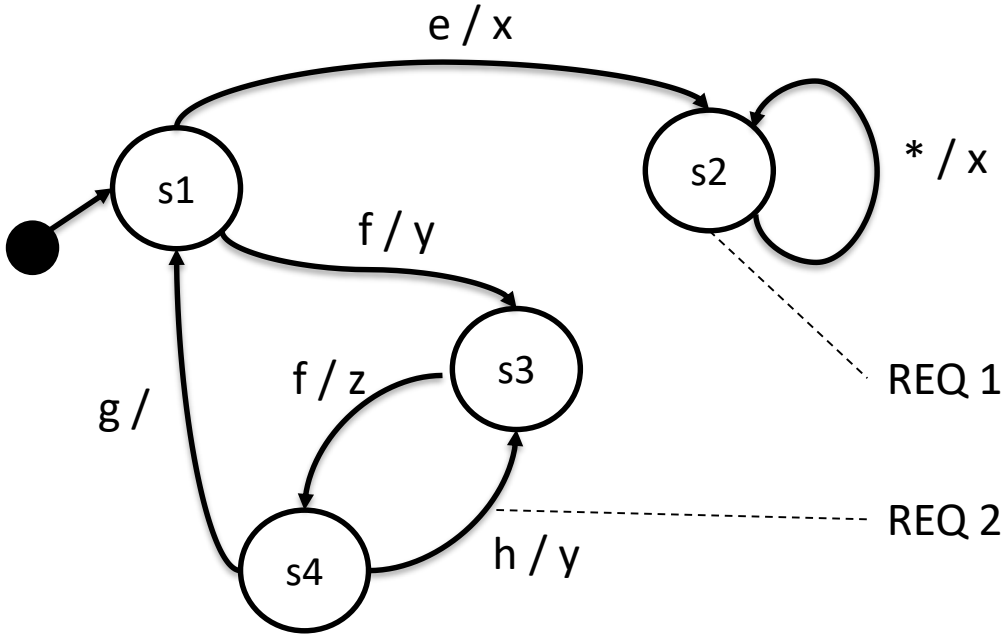


Source: M. Utting, A. Pretschner, B. Legeard. „A taxonomy of model-based testing approaches”, STVR 2012; 22:297–312

Typical test selection criteria

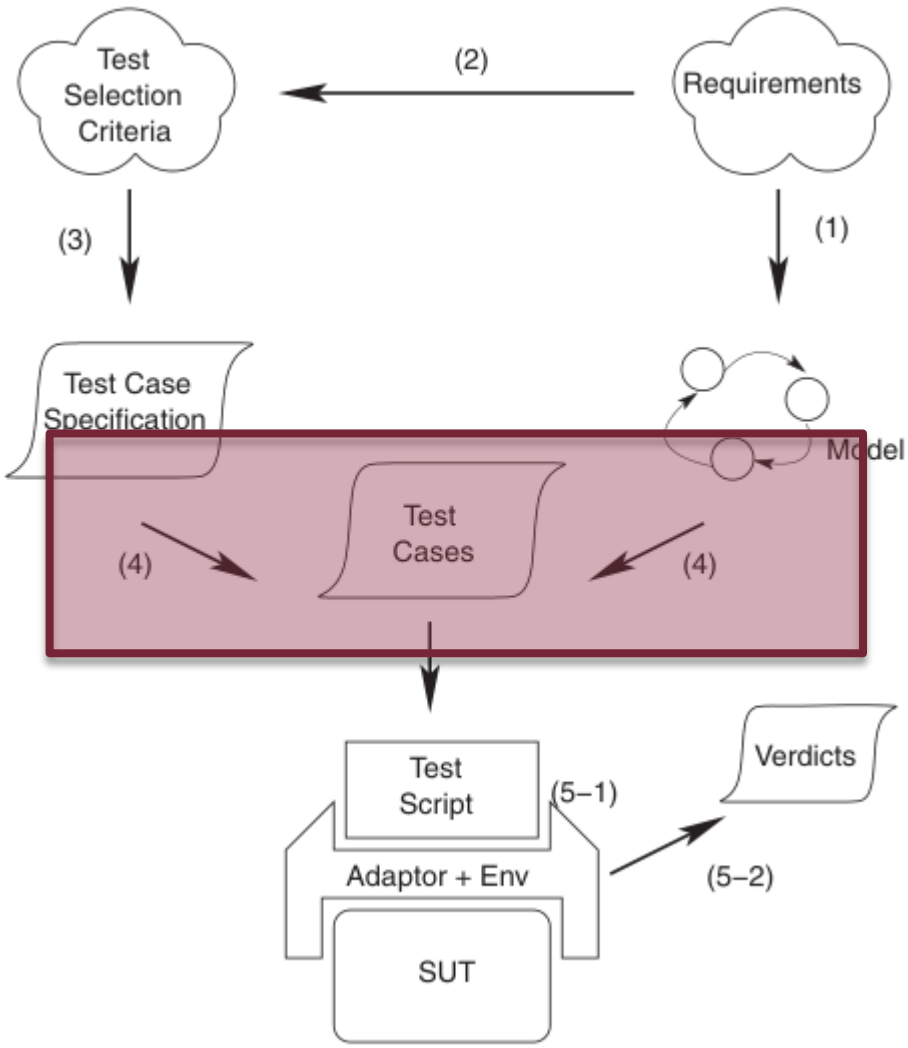
- Coverage-based
  - Requirements linked to the model
  - MBT model elements (state, transition, decision…)
  - Data-related (see spec. test design techniques)
- Random / stochastic
- Scenario- and pattern based (use case…)
- Project-driven (risk, effort, resources…)

EXERCISE Test selection for state models



- Select test cases for full
  - requirement coverage
  - state coverage
  - transition coverage

Typical MBT process

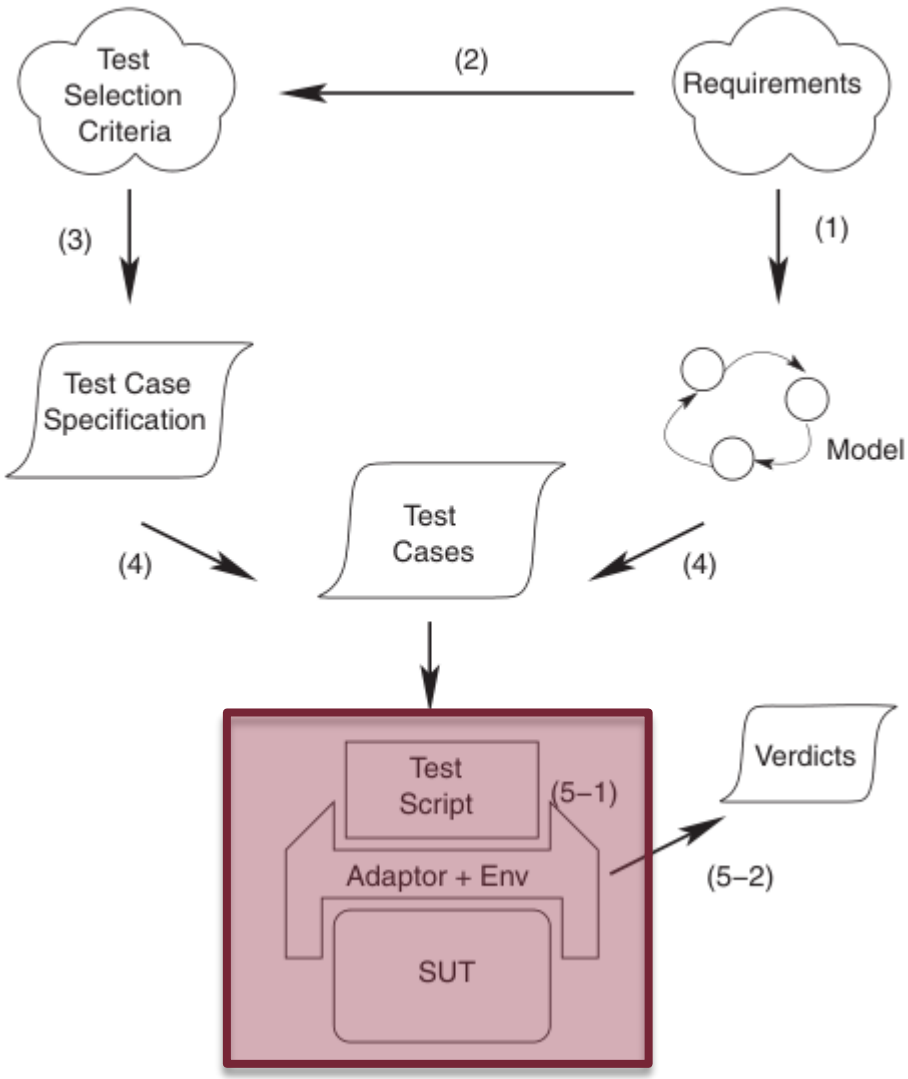


Source: M. Utting, A. Pretschner, B. Legeard. „A taxonomy of model-based testing approaches", STVR 2012; 22:297–312

Test generation methods (sample)

- Direct graph algorithms
  - Transition coverage → "New York Street Sweeper problem"
- Finite State Machine (FSM) testing
  - Homing and synchronizing sequences, state identification and verification, conformance…
- Labeled Transition System (LTS) testing
  - Equivalence and preorder relations, ioco
- Using model checkers
- Fault-based (mutation)

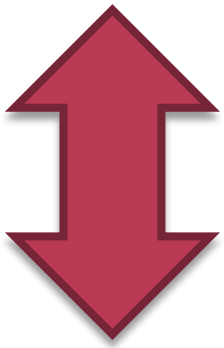
Typical MBT process



Source: M. Utting, A. Pretschner, B. Legeard. „A taxonomy of model-based testing approaches", STVR 2012; 22:297–312

Abstract and concrete test cases

- Abstract test case
  - Logical predicate instead of values (e.g. SLOW/FAST instead of 122.35)
  - High-level events and actions



Abstraction gap!

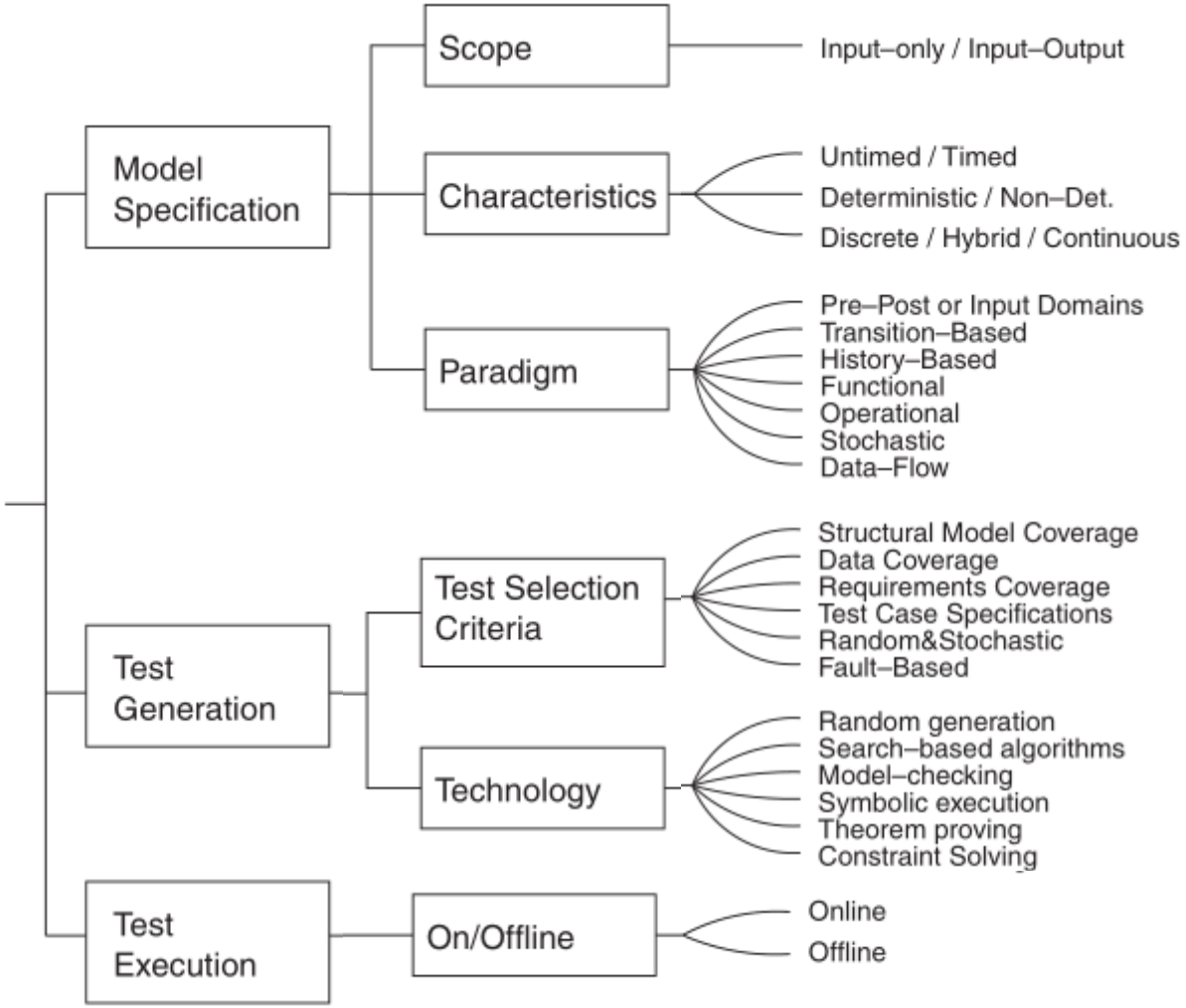
- Concrete test case
  - Concrete input data
  - Detailed test procedure (manual or automatic)

Adaptation (automatic execution)

- Adaptation layer
  - Code blocks for each model-level event and action
  - Wrapper around the SUT

- See: Keyword-driven testing

Summary: Taxonomy of MBT approaches



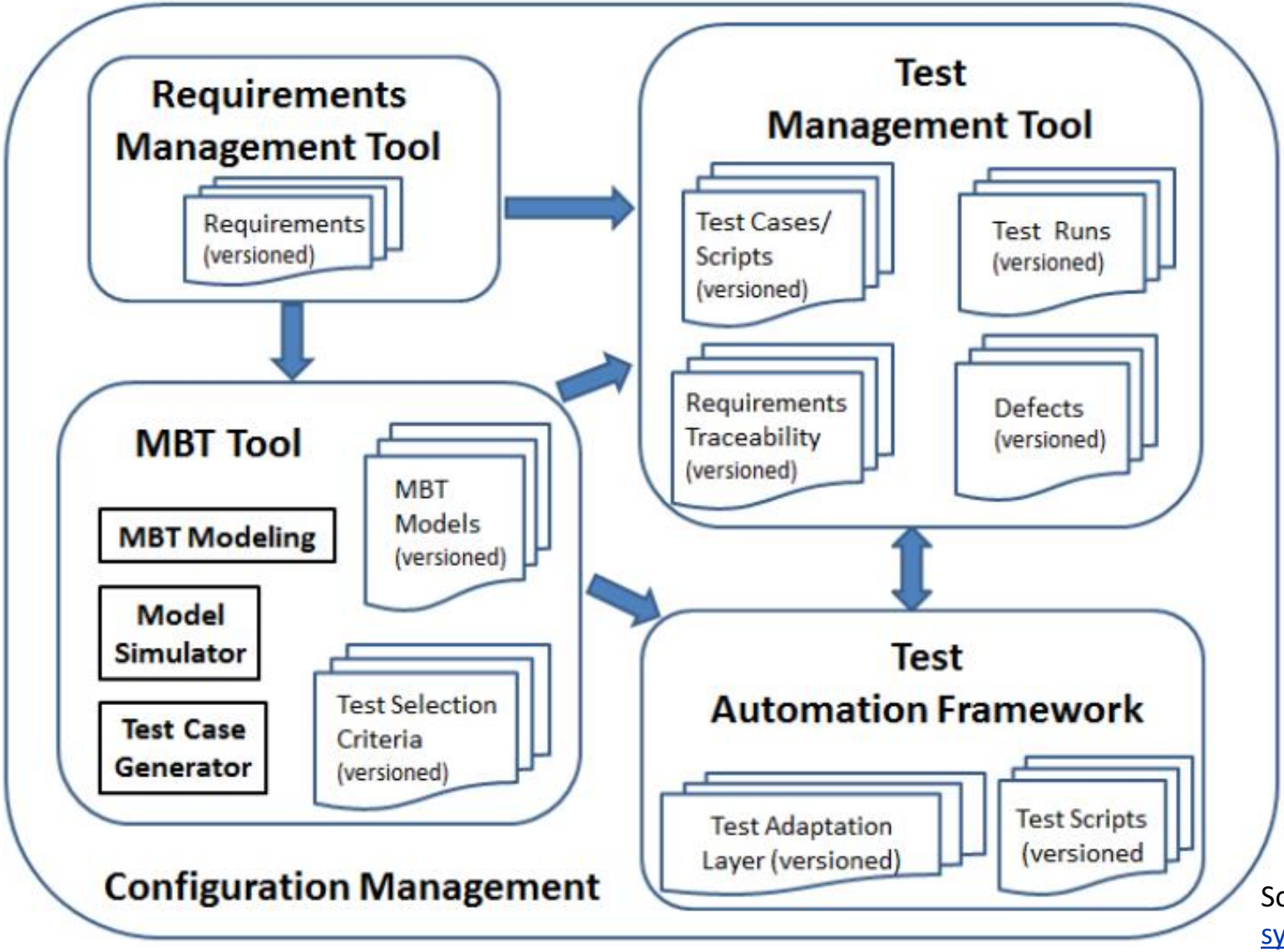
Source: M. Utting, A. Pretschner, B. Legeard. „A taxonomy of model-based testing approaches", STVR 2012; 22:297–312

TOOLS AND CASE STUDIES

Typical use cases

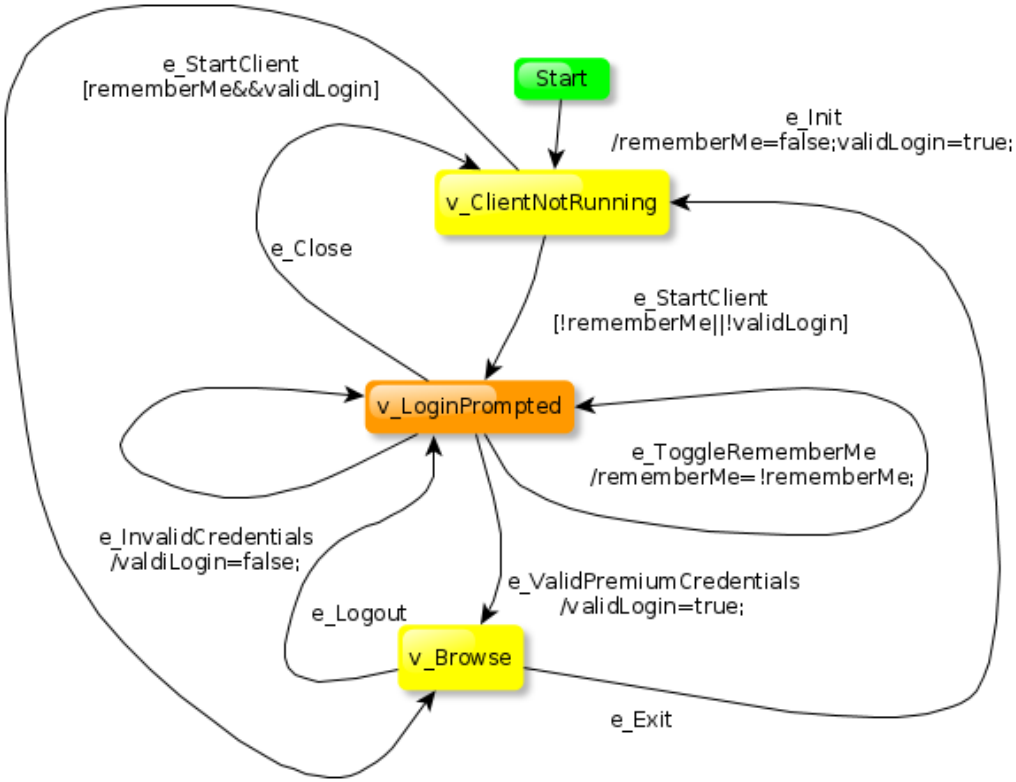
- **Fast & easy**
  - Simple modeling
  - Using open tools
- **Full fledged**
  - Complex, commercial tool
  - Full lifecycle support
- **Advanced**
  - Custom modeling languages/tools

MBT tool chain



Source: ISTQB syllabus

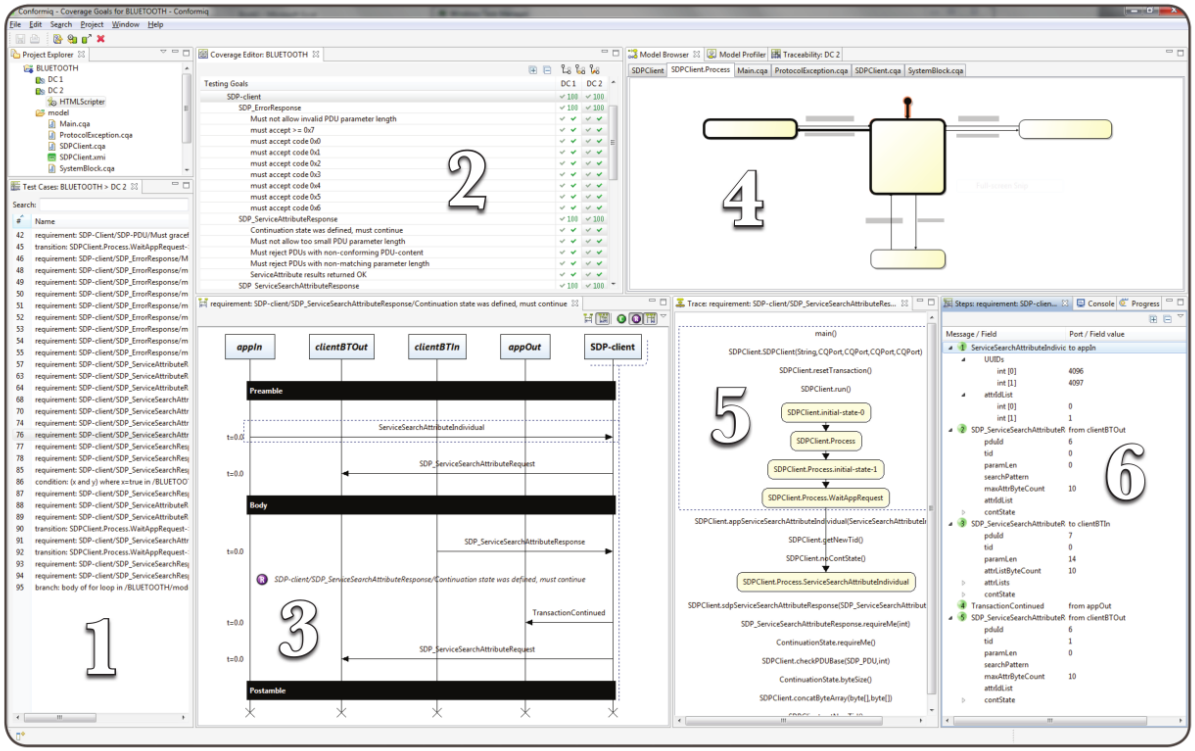
Open source tool: GraphWalker



Source: GraphWalker

- FSM model + simple guards
- Coverage: state, transition, time limit (random walk)
- Traversing the graph: random, A*, shortest path
- Generating JUnit test stubs (adapter)

Industrial MBT tool – Conformiq

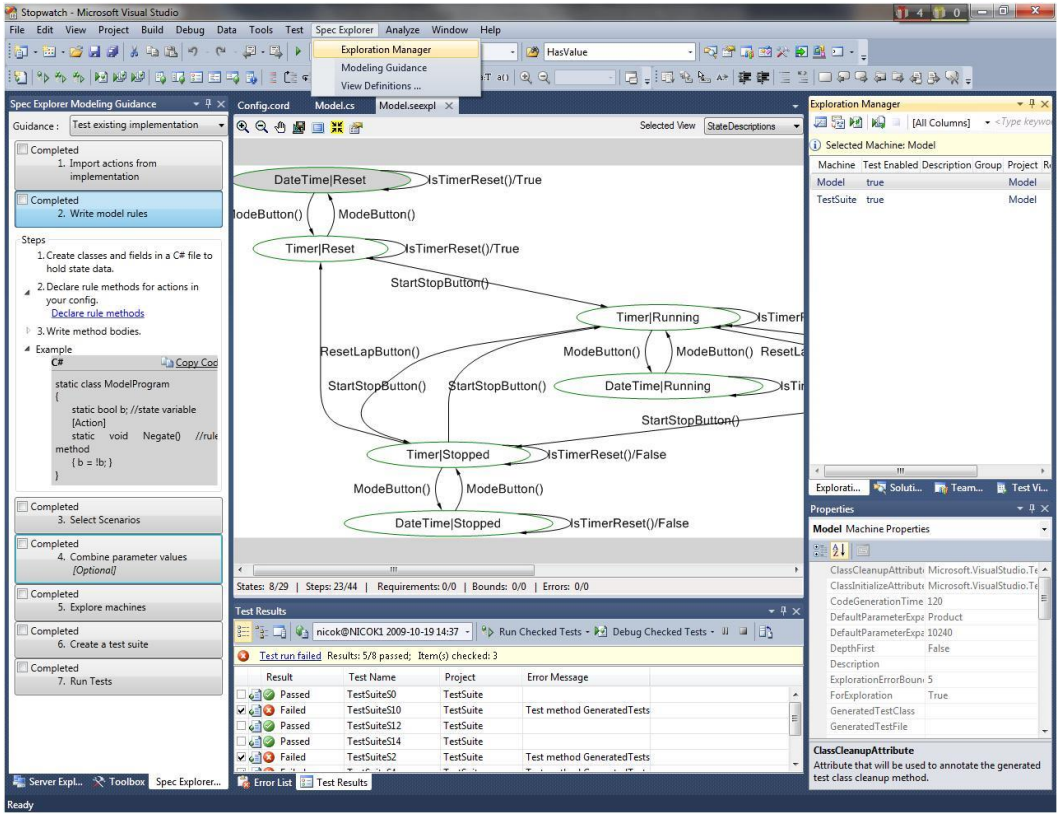


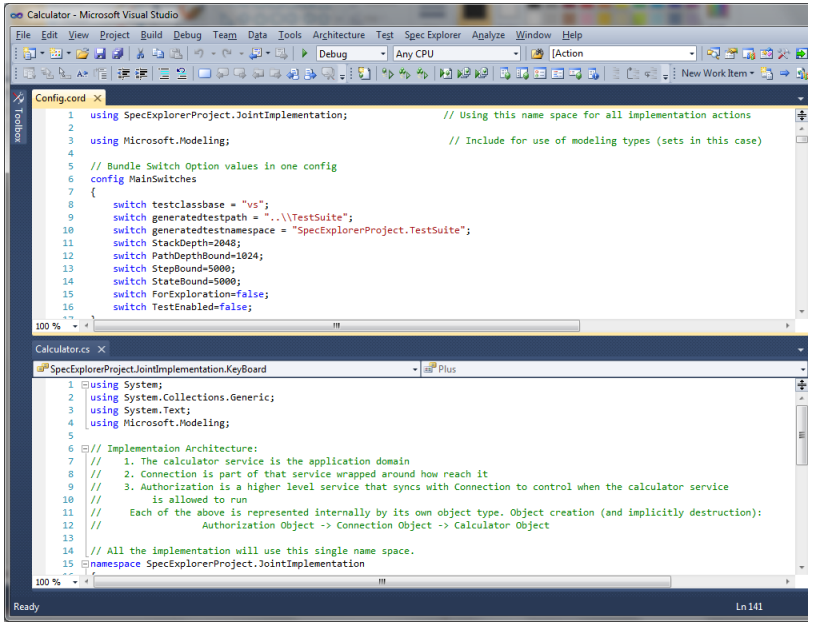
Conformiq Designer IDE for automatic test case generation

Source: Conformiq. „Testing Bluetooth Protocol Stacks with Computer-Generated Tests". Technology brief. 2010

- State machine models + Java action code
- Coverage: requirement, state, transition…
- Integration with numerous other tools

Industrial MBT tool – SpecExplorer





Source: https://visualstudiogallery.msdn.microsoft.com/271d0904-f178-4ce9-956b-d9bfa4902745

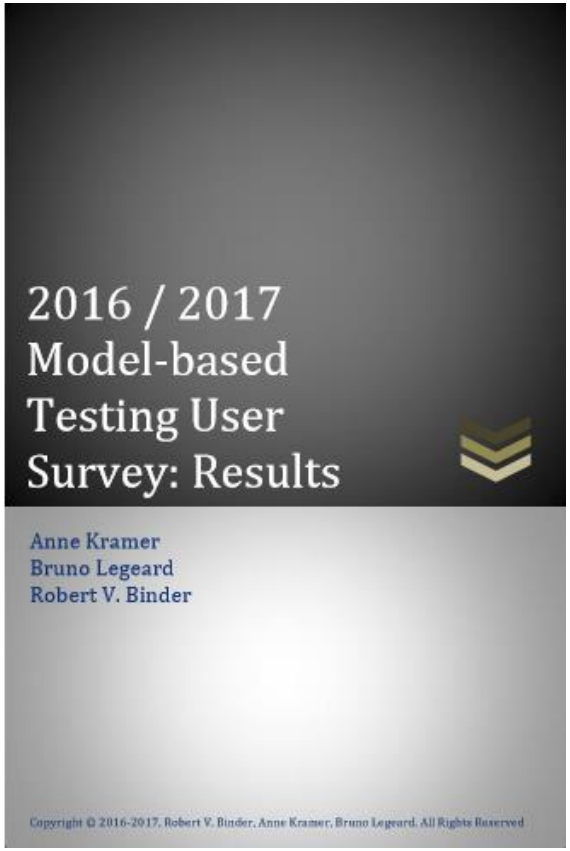
- C# model program + adapter code
- Slicing: scenarios, action patterns

Tools (cont'd)

- **CertifyIt (Smartesting)**
  - UML + OCL models
- **MoMuT::UML (academic)**
  - UML state machines, mutation testing
- **4Test-Plus (4test.io)**
  - Gherkin-like syntax for partitions/constraints

List of tools: http://mit.bme.hu/~micskeiz/pages/modelbased_testing.html

MBT User Survey



~100 participants
32 questions

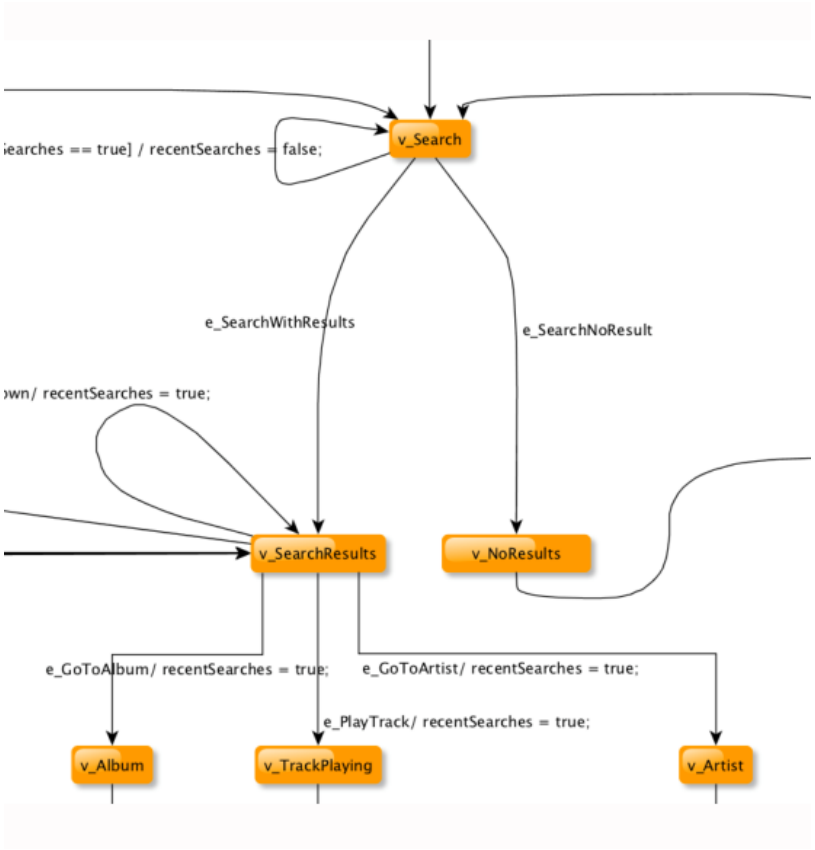
Testing levels	
System testing	77,4%
Integration testing	49,5%
Acceptance testing	40,9%
Component testing	31,2%
Generated artifacts	
Automated test scripts	84,2%
Manual test cases	56,6%
Test data	39,5%
Others (docs, test suites…)	28,9%

- "approx. 80h needed to become proficient"
- MBT is effective
- Lots of other details!

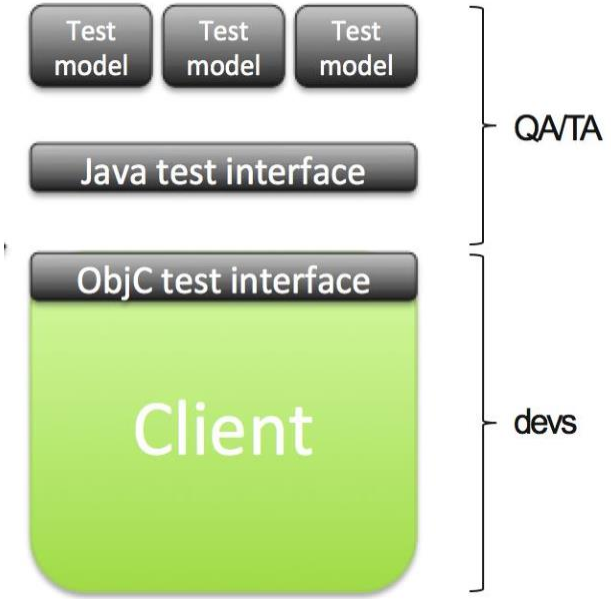
Source: http://www.cftl.fr/wp-content/uploads/2017/02/2016-MBT-User-Survey-Results.pdf

Case study: Spotify

Modell + GraphWalker



MBT + test automation

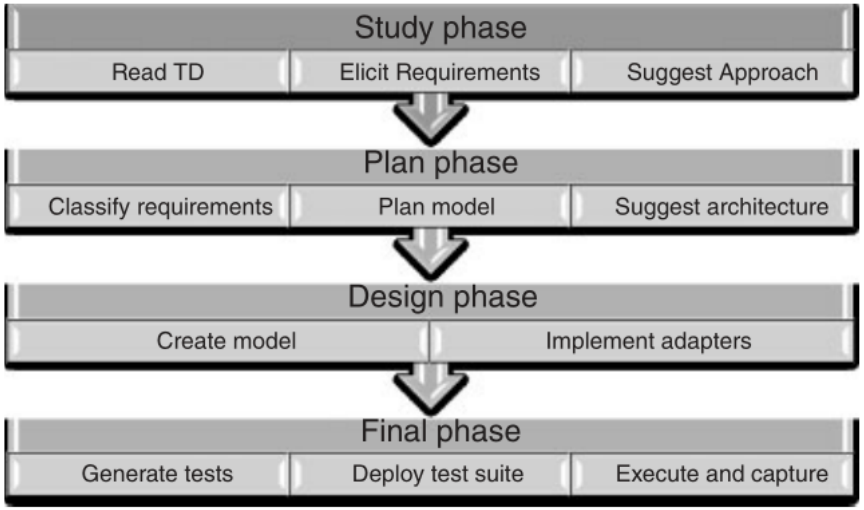


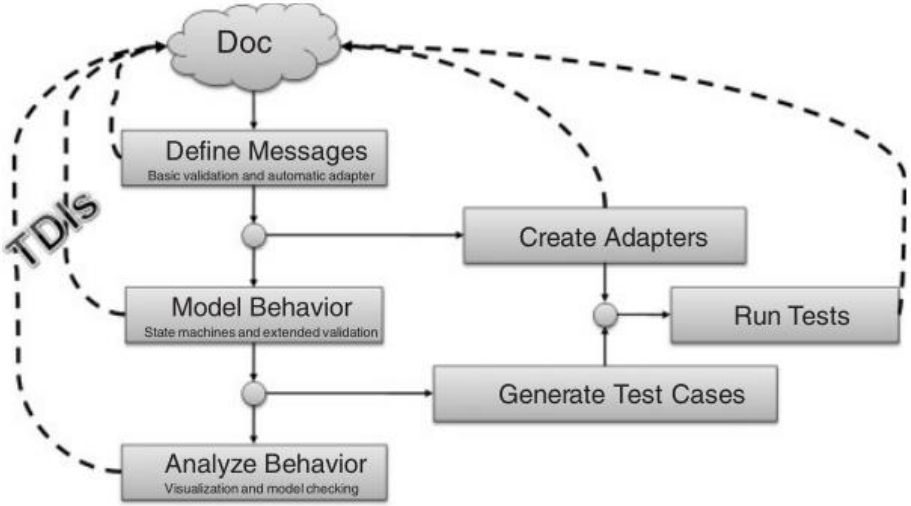
Spotify

Test automation and Model-Based Testing in agile dev cycle @ Spotify, UCAAT 2013

Case study: MS protocol documentation

- 250+ protocol, 25.000+ pages documentation
- 250+ man year, 350+ engineer
- Tool: SpecExplorer





Details: http://queue.acm.org/detail.cfm?id=1996412

Source: W. Grieskamp et al. „Model-based quality assurance of protocol documentation: tools and methodology," STVR, 21:55-71, 2011

“Cheat sheet” for introducing MBT

From Robert V. Binder (http://robertvbinder.com/)

Recommended	Not recommended
Complex SUT behavior	Simple functionality
Abstractable requirements	Subjective evaluation
Testable interfaces	Monolithic GUI
Must to regression testing	Low-value, deprecated GUI
Sophisticated test engineers	Little or no established testing
	Non-technical QA team

See also: „Model-Based Testing: Why, What, How,” http://www.slideshare.net/robertvbinder/model-basedtestingignite

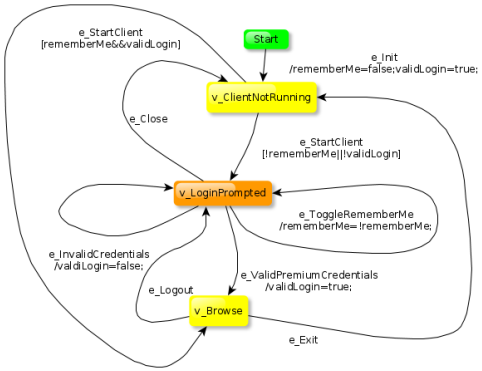
ISTQB CTFL-MBT training + exam

ISTQB® FOUNDATION LEVEL MODEL-BASED TESTER

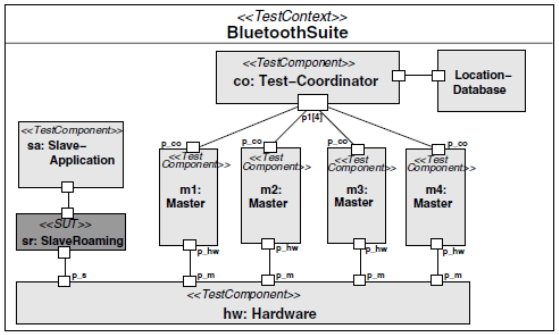
Introduction to Model-Based Testing	MBT Modeling	Selection Criteria for Test Case Generation	MBT Test Implementation and Execution	Evaluating and Deploying an MBT Approach
Objectives and Motivations for MBT	MBT Modeling activities	Classification of MBT Test Selection Criteria	Specifics of MBT Test Implementation and Execution	Evaluate an MBT Deployment
MBT Activities and Artifacts	Languages for MBT Models	Applying Test Selection Criteria	Activities of Test Adaptation in MBT	Manage and Monitor the Deployment of an MBT Approach
Integrating MBT into the Software Development Lifecycles	Good Practices for MBT Modeling Activities			

Source: ISTQB

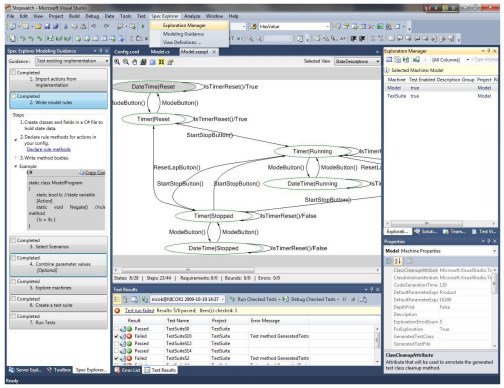
Summary



Many models,
test goals and tools



MBT = using models in testing



Scaling from
brainstorming to
fully automatic
test case generation