VETESS : MDE, Testing approaches and SysML

May 2010
Project organization

- Vérification de systèmes embarqués
  VEHicules par génération automatique de
  TESTs à partir des Spécifications
  Checking automotive embedded systems
  with automatic test case generation from
  specifications

- Started in 07/2008
- En in 08/2010
Plan

- Test in system engineering
- Model Based Testing
- The VETESS tool chain
- The front wiper case study
- Outlook
Plan

- Test in system engineering
- Model Based Testing
- The VETESS tool chain
- The front wiper case study
- Outlook
Test in system engineering

Product specifications

- Requirements
- Specifications
- Conception
- Prototyping
- Implementation

Validation plan, integration

Product

- Fonctionnal testing
- Integration
- Unit testing
- Integration

Design

HIL
Problem: sharing specifications

Product specifications

- Requirements
- Specifications
- Conception
- Prototyping

Product

- Fonctionnal testing
- Integration
- Unit testing
- Implementation

Validation plan, integration

Design

HIL

Integration
Project objectives

- Checking automotive embedded systems with automatic test case generation from specifications
  - The purpose is to generate test cases directly from the models representing system specification
Project objectives

Checking automotive embedded systems with automatic test case generation from specifications

- Specification models:
  - UML or SysML
- Test cases:
  - Enter test beds
Plan

- Test in system engineering
- Model Based Testing
- The VETESS tool chain
- The front wiper case study
- Outlook
Testing

- Verification: checking coherence between
  - A model as engineered from requirements to represent the system
    - Functional behavior
    - Non-functional properties (performance…)
  - The system
    - At runtime
    - Alone or in situation
Testing

- Static verification
  - Code reviews
  - Model checking / proof

- Dynamic verification
  - System stimulation and behavior control
    - Can’t cover all possible cases
      - Most representative test cases have to be selected
    - Often more expensive than the system itself
      - Execute tests
      - Produce / maintain tests
      - Analyze results
Testing

Questions:

• Do I have enough tests?
• Are my tests covering all possibilities offered by the specification?
• Do my tests execute in a reasonable time?
Test approaches

- Manual approach
  - Scenarios describe system stimulation and responses
  - Human execute scenarios and check results
Test approaches

Capture/replay approach

• Stimuli and responses are recorded and can be replayed automatically
Test approaches

- **Scripted approach**
  - Scripts are written
    - JUnit, TestNG,…
Test approaches

- **Keyword-based approach**
  - Scripts are written in some pseudo-code
  - Concretization matches keywords with real-life messages

Diagram:
- Requirements
- Test plan
- Test cases
- Keyword environment
- Test Scripts
- Adapter
- Test Execution Tool
- SUT
Test approaches

- Model Based Testing
  - A large number of keyword-based tests are generated from an executable model
Advantages

- Automatically covers the specification model as much as possible
- Test case mutualization
  - 1 test case for covering different parts of the model
- Predicts system behavior (oracle)
Model Based Testing

1. Modeling
2. Generation
3. Concretization
4. Analysis

Spécification

Test case generation tool

Test cases

Test Runner Interface

14/01/11
Simulation vs. test beds

1. Modeling
2. Generation
3. Concretization
4. Analysis

Test case generation tool

Spécification

Test cases

Test Runner Interface

Simulation Mode

Test Beds

14/01/11
Plan

- Test in system engineering
- Model Based Testing
- The VETESS tool chain
- The front wiper case study
- Outlook
Test Designer

Model-based testing tool for discrete systems

- The model is a behavioral specification of the system
  - UML model or SysML models
- Test can be exported
  - Keyword
  - JUnit
  - HP Quality Center
Testing continuous systems
Input and outputs are sampled signals
- Expected signals are compared to actual signals
  - Specific comparators (ignoring threshold, accepting latency...)

Input → System → Actual → Comparator → Result
Test may be exercised on

- A system on a test bed
- A simulation model (Matlab, .NET…)

Input → Model → System → Comparator → Expected → Actual → Result
Tool chain

UML MODEL
(RSM, Topcased, …)

SysML MODEL
(Topcased)

UML4MBT MODEL

SysML4MBT MODEL

Test Designer™

Test Cases

Adaptation Layer
Tool chain
UML4MBT / SysML4MBT

- **UML4MBT:**
  - Accepted input data
    - Formalized as a metamodel
    - Subset of UML
    - API (required interface) for Test Designer

- **SysML4MBT**
  - Subset of SysML
  - Model transformation from SysML4MBT to UML4MBT
Other languages can be supported through model transformation
Tool chain

UML MODEL (RSM, Topcased, …)

SysML MODEL (Topcased)

UML4MBT MODEL

SysML4MBT MODEL

Test Designer ™

Test Cases

Adaptation Layer
Adaptation Layer

- Generated test cases
  - Have to be concretized
  - Messages: discrete nature

- Test bed inputs
  - Have to be mapped to reality
  - I/O Signals: continuous nature (sampled)

- Automatic matching
  - Generate variables
  - Generate signals
Adaptation Layer

Discrete messages generate continuous signals

- Example: `turnLeft(moderately)` message changes the signal to -90° to be sent on the `SteeringWheelAngle` variable

- Curves are smoothed
Testing systems in 4 steps

- Create specification model in SysML
  - States discrete system behavior
    (Could also state environment behavior)
- Generate tests from model
  - Take the shape of sampled signals
    sent to / read from variables
- Map variables to reality
  - CAN messages, input pins…
  - Select comparators
- Run the tests
Refining expected signals

Additional step:
provide a continuous mathematical model
- Generated test are run against the mathematical model
- Output signals are promoted to expected signals
Plan

- Test in system engineering
- Model Based Testing
- The VETESS tool chain
- The front wiper case study
- Outlook
Plan

- Test in system engineering
- Model Based Testing
- The VETESS tool chain
- The front wiper case study
- Outlook