Usage
Model-Based Testing for Hardware In the Loop

THE SOLUTION TO BOOST
YOUR TEST EFFICIENCY
ALL4TEC

Since 1998

French: 45 experts

Turnover 2009: 4 M€

Expertise offer:

- QUALITY
- SAFETY ENGINEERING
- SYSTEM ENGINEERING
- TESTING

EMBEDDED SYSTEMS
QUESTION YOUR TEST DEPARTMENT

What is your cost dealing with various:

- Environment Models
- Test Tools
- Specification Approaches
- Test Platforms
- Measurement Devices
- Specification Approaches

How long do you spend trying to:

- Reuse thru different projects
- Reuse thru different engineers
- Reuse thru different car brands

How hard do you fight to:

- Estimate the quality
- Calculate the functional coverage
- Make documents, reports

Imagine you can use all this invest for:

INCREASING TRUST YOU HAVE IN YOUR PRODUCTS
MODEL BASED TESTING PROCESS

- Test Cases Generator
- Requirements Management
- Usage Model Editor
- Test-Cases Generator
- Test Campaign Analysis
- Platform API
- Test Runner
- Hardware In the Loop
- System Under Test
WHAT IS MaTeLo?

- Formal Test Specification
- Product point of view
- Enhanced Coverage
- Fully Measured process
- Test environment Compatibility
- Requirements Traceability
- Better Engineering Productivity

ALL4TEC MaTeLo Markov Test Logic
WHAT IS EXAM?

Focused on Reuse & Collaboration

Better Usage of Simulators & Devices

Centralized Test Repository

MicroNova EXAM
EXtended Automation Method

Method designed & consolidate

Focused on Reuse & Collaboration

Formal Test Cases Specification

Abstraction Layers

Automotive Ready Library

ASAM

vector

ETAS

dSPACE

garts GmbH

MicroNova

Audi

Volkswagen

NATIONAL INSTRUMENTS

EXAM database
MATELO USAGE MODEL

(1) ALL POSSIBLE USAGES SCENARIO

(2) DYNAMIC REPRESENTATION

(3) FORMAL REQUIREMENT TEST SPECIFICATION

(4) CONSISTENCY BETWEEN
1. TESTS
2. REQUIREMENTS
3. USE CASES
TESTING SCOPE

- Process
  - Black Box testing

- Field Application
  - Functional Testing
  - Integration Testing
  - Acceptance Testing

- System Under Test
  - Model
  - Software
  - System
  - Product
MaTeLo Editor: Model Based Test Specification
- Design the usage model
- Qualify the possible path with profiles
- Assign test operations and requirements

MaTeLo Testor: Model Based Test Cases Generation
- Define the test strategy
- Generate the test cases

EXAM: Test Automation
- Manage the test cases on the test repository
- Manage the test environment heterogeneity
- Execute an log automatically the test sequences

MaTeLo Test Campaign Analysis: Model Based Quality Insurance
- Analysis of the Test Campaign
- Quality & Reliability assessment
MaTeLo EDITOR
Usage Model Design
MATELO USAGE MODELS

→ Probability on every Choices
→ Association of Requirements
→ Configuration of Test Operations

MACRO CHAIN

BEGINNING
TRANSITION
STATE
END
MODEL TRANSITION = TEST STEP

Stimulations

Inputs Stimulation

Classes of Equivalence

Doors Req.

EXAM operations

Test Oracle

Outputs = f(\text{Inputs})

Expected Results

- Input ports
- Networks signals
- Variables
- GUI objects
- ...

Output ports

- Network signals
- Environ. Variables
- Wave Form
- ...

Checks Outputs
Profiles can be embedded to qualify the usage model

- Operational profiles
- Test profiles

Data distribution

Usage path probability
MaTeLo EDITOR
Requirements Management
REQUIREMENTS MANAGEMENT

IBM Doors

MaTeLo Req Library

MaTeLo Usage Model

CSV
(Reqtify, Excel ...)

Import
Update

Model
Association

XML

Model Object
- Transition
- Input, Expected Result
- Value, Class of Equivalence

MaTeLo Project

Requirement Role
- Necessary
- Sufficient
MaTeLo TESTOR
Test Cases Generation
MaTeLo TEST STRATEGY

- DEFINE THE TEST STRATEGY, BY CHOOSING
  - Test Algorithm
  - Test Profile
  - Part of model to test

Most probable approach

Field application approach

Arcs coverage approach

Usage approach

- FREQUENCY
- RISK, CRITICISM
- UPDATE...
- COVERAGE
- USAGE
  (Determinist, Stochastic)

Generate
### MaTeLo TESTOR: HTML TEST PLAN

<table>
<thead>
<tr>
<th>State</th>
<th>Requirement</th>
<th>Input</th>
<th>Expected Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>S_Automatic Mode</td>
<td>T_Push Unrolled Awning Button</td>
<td>myCharString : Push Unrolled Awning Button</td>
<td></td>
</tr>
</tbody>
</table>

**Requirements:**
- Req_1006
MaTeLo EDITOR
Test Bench Management
MaTeLo FOR EXAM

**Test Repository**
- SUT Interface
- Test Operations
  - Stimulation
  - Measurement
  - Administration
- Sub Test Sequences

**Requirements**
- Name
- UUID
- Description

**Automatic Test Cases**
- Automatic call of Test Operation
- Automatic parameterisation
- Requirements association
- Test Case description generation
- Usage model respect

**Association**
- Generate
- Test Repository
- Requirements
- Automatic Test Cases
ABSTRACTION LAYERS

MaTeLo

<< derive >>
Usage Scenario Description
Test Case Specification
Test Flow Control Device Driver

<< generate >>
Markov-Chain Usage Model
UML Sequence Diagram
Python Precompiled Application

<< control >>
System Under Test
Hardware In the Loop Test Bench

EXAM

Third Party Equipment
COLLABORATIVE ARCHITECTURE

Multi Users
Test Environment

Test Cases Implementer
Python Code Generator

EXAM Modeller
- DoorsSync
- SystemConfig
- Composer
- ParameterManager
- Mapping
- Messaging

DB-Access
- Hibernate

Messaging Server
(JBoss)

EXAM database
(Oracle, MySQL, PostgreSQL)

Test Repository
Project, Libraries, Test Campaign
EXAM Modeller
Test Cases Description
**EXAM UML TEST SEQUENCE**

**BEGINNING**

- T Check
- T Check
- T Set BP
- T Exec
- sub2
- T Wait
- T Check
- sub4
- T Kill

**Test Operation**

- checkTolerance(0, 0, 0, 0)
- createNewSubTest('Step 3', any aActualValue[in] =, x
- setBreakPoint('nothing')
- executeCommand('cmd', '')
- sub2_1
- wait(50)
- checkBit(0, 10, 0)
- sub4_1
- killProcess('explorer.exe')

**Sub Sequence**

<table>
<thead>
<tr>
<th>T Exec</th>
<th>T Exec</th>
<th>T Check</th>
<th>T Check</th>
<th>createNewSubTest('Step 1', any aActualValue[in] =, x</th>
</tr>
</thead>
<tbody>
<tr>
<td>checkErrorMemory(2, 1, list IOItem[output] =, x)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>createNewSubTest('Step 2', any aActualValue[in] =, x)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sub3</td>
<td>sub3_1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T Wait</td>
<td>wait(60)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
EXAM INTERFACE CONCEPT

Test Case Description

Test Case Implementations

Interface Implementation
- Virtual Function
- Virtual Car
- Sub Sequence
- Test Environment Access
- ...

Python Code

Sub Sequence
EXAM Modeller
Test Cases Management
EXAM ABSTRACTION MANAGEMENT

Variable Mapping

Simulation Model Abstraction

Short-name Mapping

Embedded Network Abstraction

Parameter Set Composing

Concrete Test Case Abstraction

System Configuration

Test Platform Abstraction
EXAM ABSTRACTION MANAGEMENT

Possible Usage Scenarios

Strategic Test Cases

Available Test Configurations

**HIL Plateform**
- dSpace
- NovaSim
- Carts
- ASAM HIL API
- Proprietary

**Network**
- CAN_1
- CAN_2
- LIN_X
- FlexRay_1
- FlexRay_2

**Functions**
- Manual Gear
- Auto Gear
- Hand Free
- ACC
- StartStop

**Test Tools**
- CANoe
- CANape
- INCA
- MS Excel
- Diag Tool

**Env. Model**
- Gasoline
- Diesel
- Turbo
- Hybrid
- Electric
SYSTEM CONFIGURATION 1

Possible Usage Scenarios

Strategic Test Cases

Available Test Configurations

<table>
<thead>
<tr>
<th>HIL Platform</th>
<th>Network</th>
<th>Functions</th>
<th>Test Tools</th>
<th>Env. Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>dSpace</td>
<td>CAN_1</td>
<td>Manual Gear</td>
<td>CANoe</td>
<td>Gasoline</td>
</tr>
<tr>
<td>NovaSim</td>
<td>CAN_2</td>
<td>Auto Gear</td>
<td>CANape</td>
<td>Diesel</td>
</tr>
<tr>
<td>Carts</td>
<td>LIN_X</td>
<td>Hand Free</td>
<td>INCA</td>
<td>Turbo</td>
</tr>
<tr>
<td>ASAM HIL API</td>
<td>FlexRay_1</td>
<td>ACC</td>
<td>MS Excel</td>
<td>Hybrid</td>
</tr>
<tr>
<td>Proprietary</td>
<td>FlexRay_2</td>
<td>StartStop</td>
<td>Diag Tool</td>
<td>Electric</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Test Case 1

Test Case 2

Test Case n

Strategic Test Cases

Possible Usage Scenarios
SYSTEM CONFIGURATION 2

Possible Usage Scenarios

Strategic Test Cases

Available Test Configurations

<table>
<thead>
<tr>
<th>HIL Plateform</th>
<th>Network</th>
<th>Functions</th>
<th>Test Tools</th>
<th>Env. Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>dSpace</td>
<td>CAN_1</td>
<td>Manual Gear</td>
<td>CANoe</td>
<td>Gasoline</td>
</tr>
<tr>
<td>NovaSim</td>
<td>CAN_2</td>
<td>Auto Gear</td>
<td>CANape</td>
<td>Diesel</td>
</tr>
<tr>
<td>Carts</td>
<td>LIN_X</td>
<td>Hand Free</td>
<td>INCA</td>
<td>Turbo</td>
</tr>
<tr>
<td>ASAM HIL API</td>
<td>FlexRay_1</td>
<td>ACC</td>
<td>MS Excel</td>
<td>Hybrid</td>
</tr>
<tr>
<td>Proprietary</td>
<td>FlexRay_2</td>
<td>StartStop</td>
<td>Diag Tool</td>
<td>Electric</td>
</tr>
</tbody>
</table>

Test Case 1

Test Case 2

Test Case n

Proprietary Network

CAN_1

CAN_2

LIN_X

FlexRay_1

FlexRay_2

FlexRay_2
EXAM Test Runner
Test Cases Automatic Execution
EXAM TEST RUNNER

AUTOMATIC TEST SCHEDULER
EXAM Report Manager
Verdict and Analyze Test Runs
EXAM TEST RECORD MANAGEMENT

REAL TIME RECORD ANALYSIS AND VERDICTS
MaTeLo TCA
Test Campaign Analysis
TEST CAMPAIGN PROCESS

Target Quality Criteria
1. Coverage
2. Reliability

Trust Indicators

New version

System Requirements

Usage Model

Test Cases

Test Scripts

Adaptor

Test Execution Platform

System Under Test

New Test Cases Generation

Metrics
Quality/Version

New version

Bugs

Criteria

<table>
<thead>
<tr>
<th>Software</th>
<th>New test cases</th>
<th>New version</th>
<th>Reuse</th>
<th>New functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Beta</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Gamma</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Delta</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Omega</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Phi</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

New version
**COVERAGE CAPABILITIES**

States coverage

Transitions coverage

Requirements coverage

**States**
- S0
- S1
- S2
- S3
- S4
- S5

**Transitions**
- T 0-1
- T 0-2
- T 0-3
- T 2-3
- T 3-4
- T 3-5

**Inputs**
- Req 341
- Signal_A: [0-10;11-50;51-100]
- Req_3411; Req_3412; Req_3413

**Expected Result**
- Signal_B: > 47 & <= 62
- Req_342

**Classes of Equivalence coverage**

- Indicators by SUT versions
- **Unitary** by version
- **Cumulated** over all SUT versions
TEST CAMPAIGN REPORT

Model Coverage

Requirements Coverage

Trust Indicators
1) **Provide all needed test layers**
   1) Usage Model
   2) Test Case Description
   3) Test Case Implementation

2) **Optimize all your test investments**
   1) Hardware In the Loop & Test tools
   2) Test Engineers

3) **Provide an open platform focused on**
   1) Formalization and abstraction
   2) Reuse and collaboration
   3) Automatic generation

4) **Optimize the test campaign for**
   1) Functional Coverage
   2) Product Availability
QUESTION?

www.all4tec.net
Booth 1720

Technical contact
MaTeLo@all4tec.net

Sales contact
anthony.faucogney@all4tec.net
+33 6 80 88 40 59

Wiki & Forum & Documentation
www.all4tec.net
www.exam-ta.de