

# Usage Model-Based Testing for Hardware In the Loop

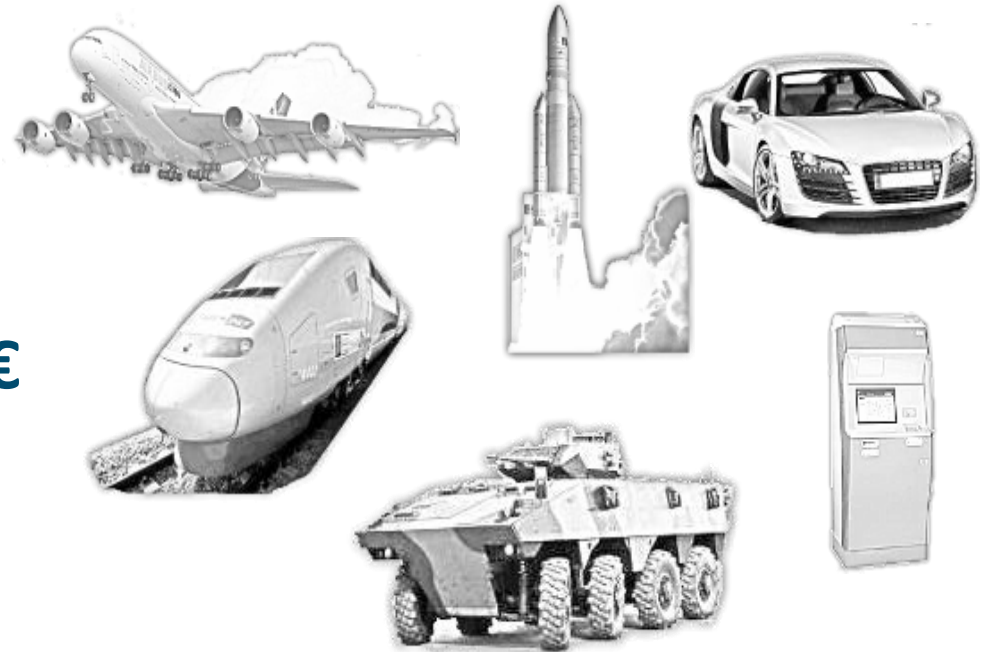


***THE SOLUTION TO BOOST  
YOUR TEST EFFICIENCY***

Since 1998

French: 45 experts

Turnover 2009: 4 M€



Expertise offer:

- ⇒ QUALITY
- ⇒ SAFETY ENGINEERING
- ⇒ SYSTEM ENGINEERING
- ⇒ TESTING



EMBEDDED  
SYSTEMS

What is your cost dealing with various:

Environment Models

Test Tools

Specification Approaches

Test Platforms

Measurement Devices

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

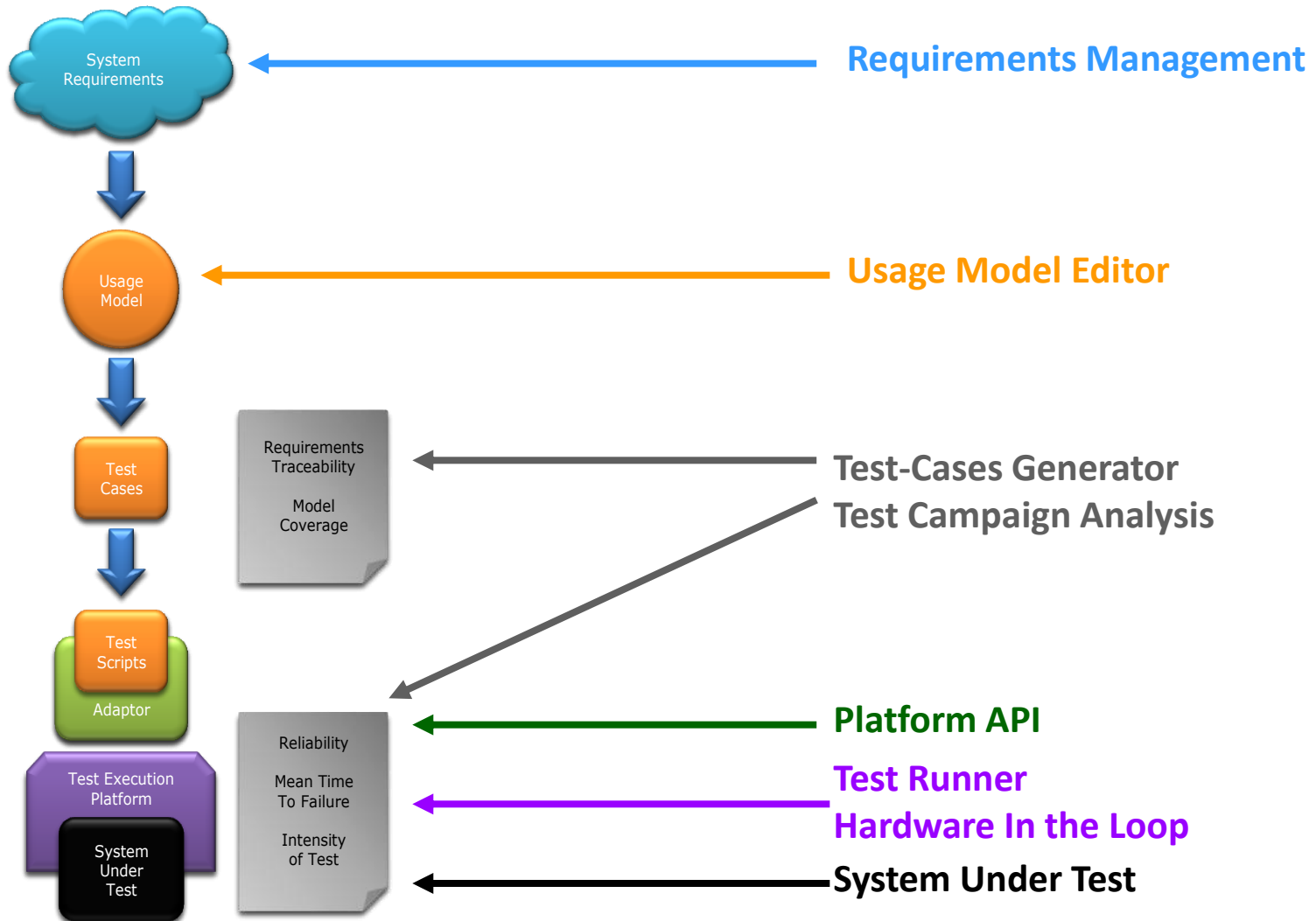
Calculate the functional coverage

Make documents, reports

Imagine you can use all this invest for:

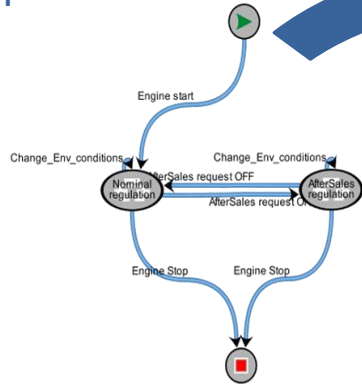
**INCREASING TRUST YOU HAVE IN YOUR PRODUCTS**

# MODEL BASED TESTING PROCESS

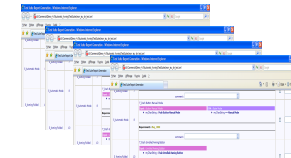


# WHAT IS MaTeLo?

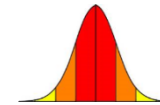
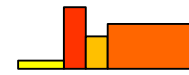
Formal Test Specification



Product point of view

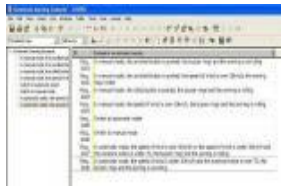
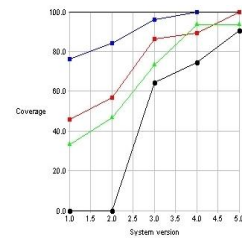


Enhanced Coverage



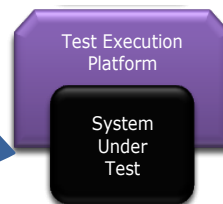
ALL4TEC MaTeLo  
Markov Test Logic

Fully Measured process



Requirements Traceability

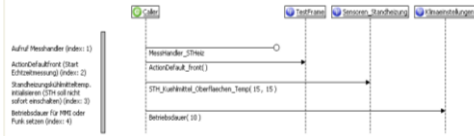
Test environment Compatibility



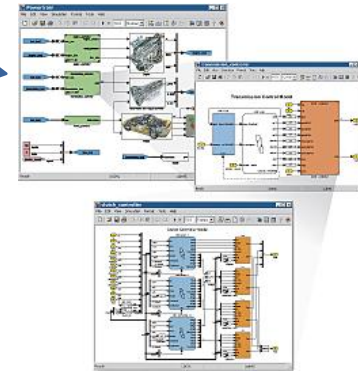
Better Engineering Productivity

# WHAT IS EXAM?

Formal  
Test Cases  
Specification



Abstraction  
Layers



Better  
Usage of  
Simulators  
& Devices

MicroNova EXAM  
EXtended Automation  
Method

Centralized  
Test Repository



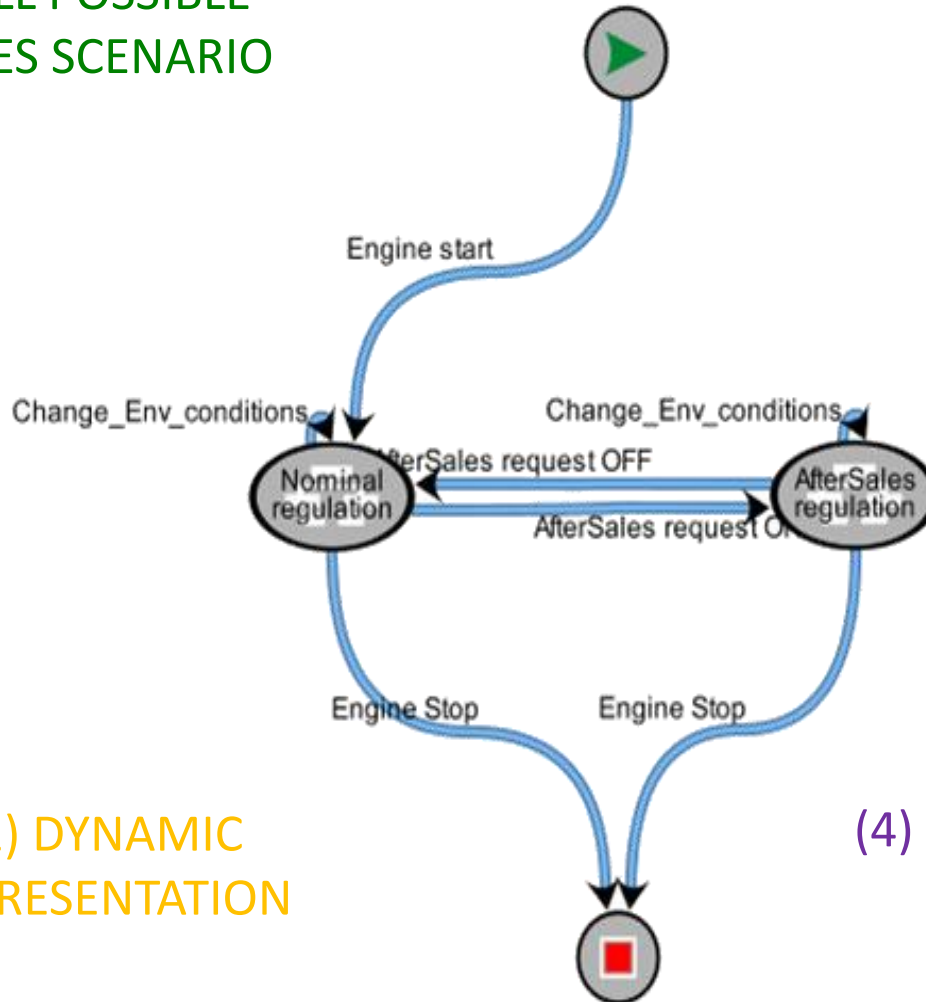
Automotive  
Ready Library

Method designed  
& consolidate



Focused on Reuse  
& Collaboration

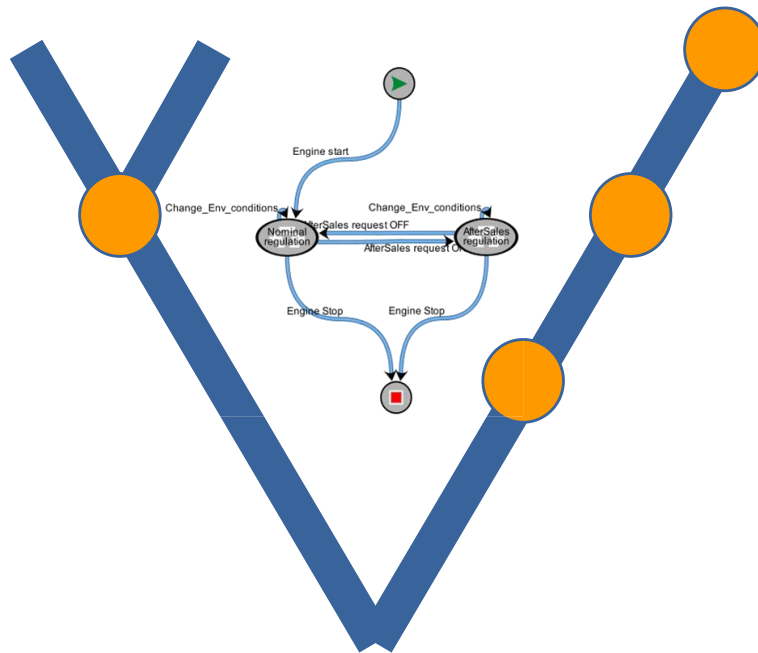
(1) ALL POSSIBLE  
USAGES SCENARIO



(2) DYNAMIC  
REPRESENTATION

(3) FORMAL  
REQUIREMENT TEST  
SPECIFICATION

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



## □ Process

⇒ Black Box testing

## □ Field Application

⇒ Functional Testing

⇒ Integration Testing

⇒ Acceptance Testing

## □ System Under Test

⇒ Model

⇒ Software

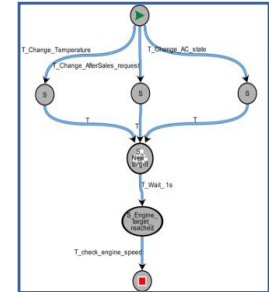
⇒ System

⇒ Product



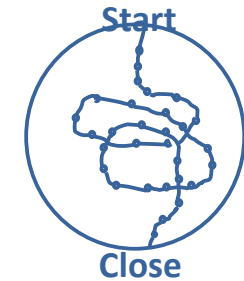
## MaTeLo Editor: Model Based Test Specification

- ⇒ 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



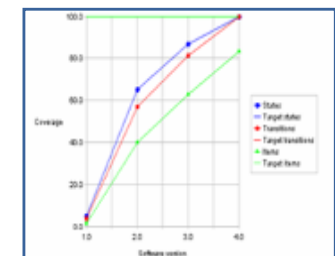
## EXAM: Test Automation

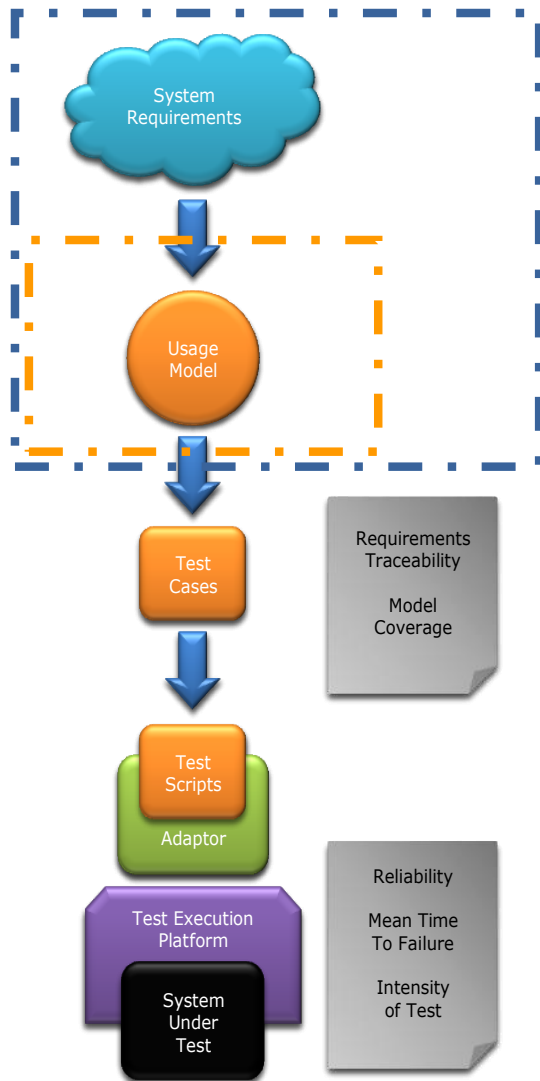
- ⇒ Manage the test cases on the test repository
- ⇒ Manage the test environment heterogeneity
- ⇒ Execute an log automatically the test sequences



## MaTeLo Test Campaign Analysis: Model Based Quality Insurance

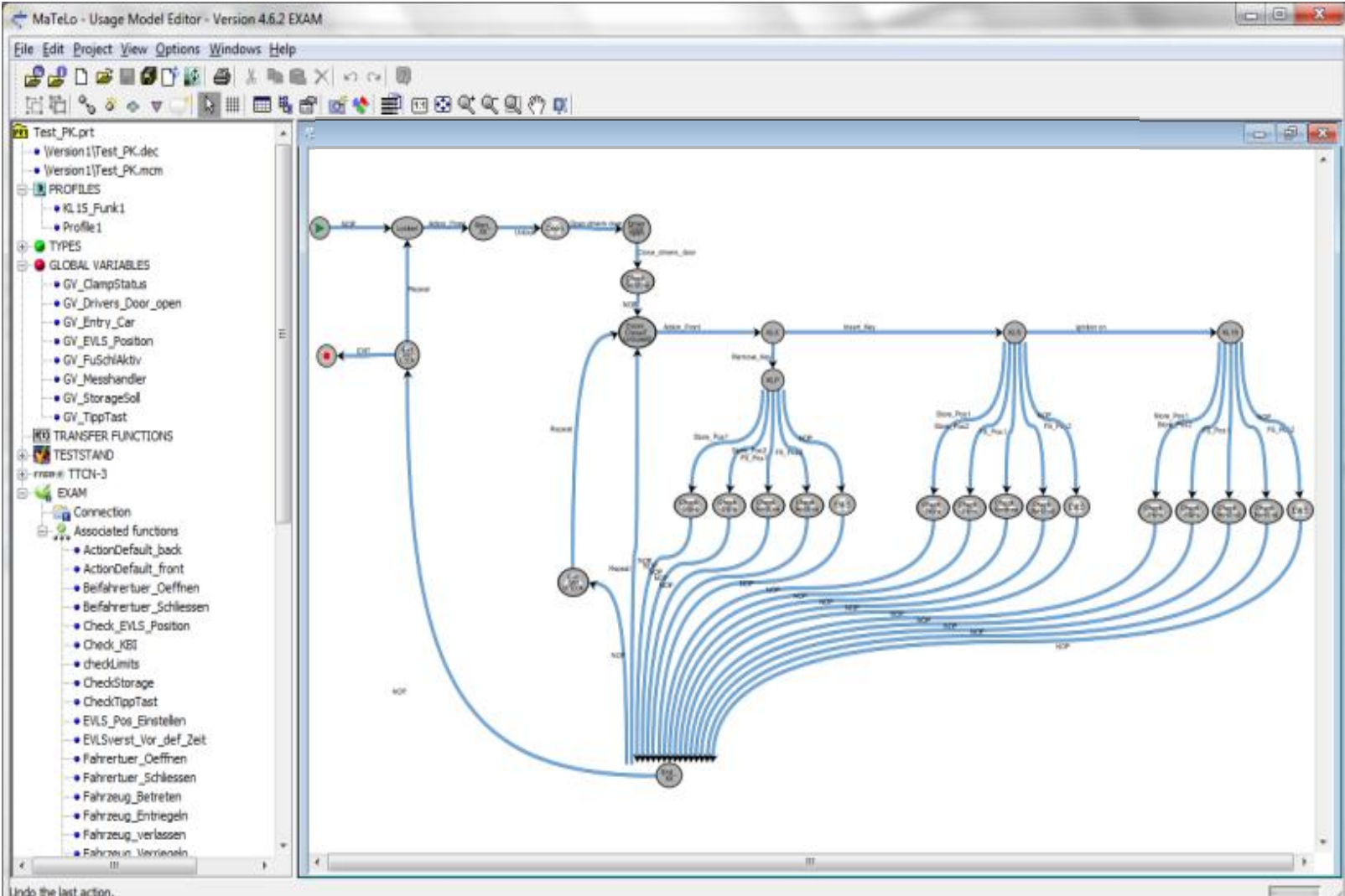
- ⇒ Analysis of the Test Campaign
- ⇒ Quality & Reliability assessment





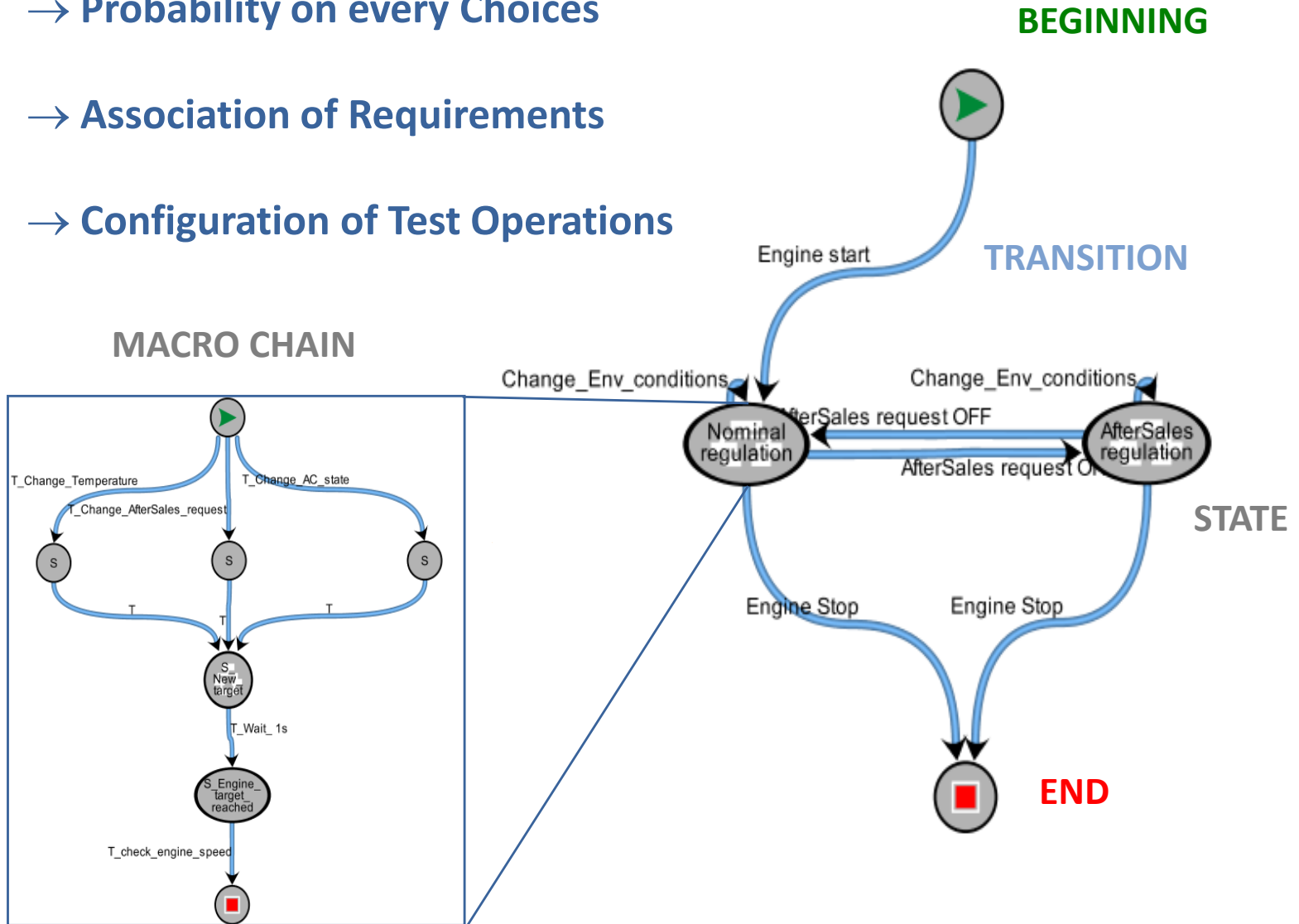
# MaTeLo EDITOR Usage Model Design

# MATELO SCREENSHOT

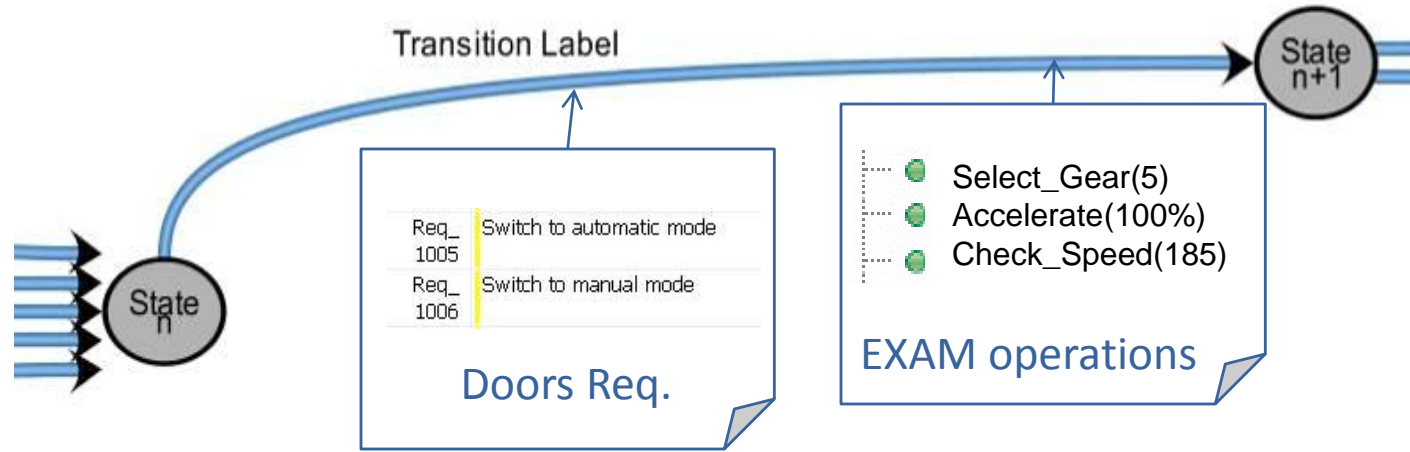


Undo the last action.

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

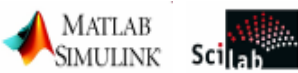


# MODEL TRANSITION = TEST STEP



**Stimulations**  
Inputs Stimulation  
Classes of Equivalence

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

**Test Oracle**   
Outputs = f(Inputs)

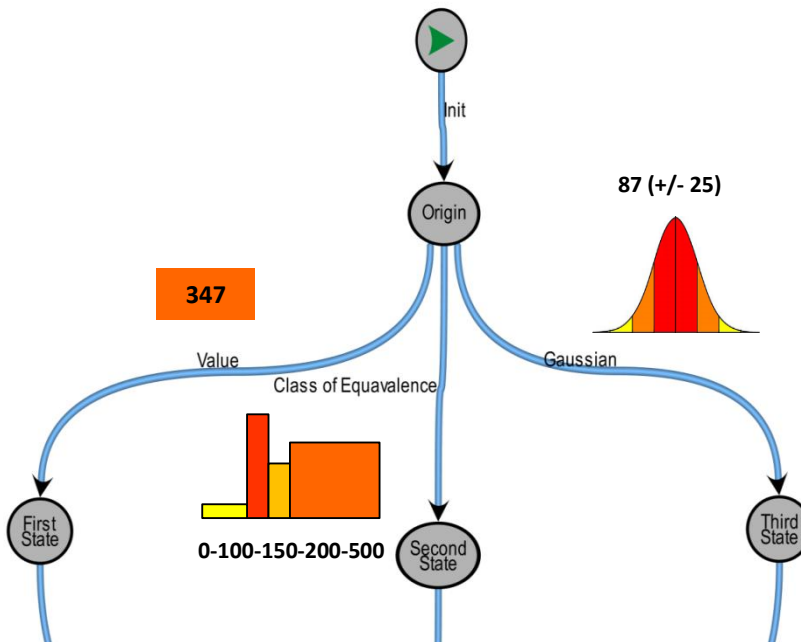
- Output ports
- Network signals
- Environ. Variables
- Wave Form
- ...

**Expected Results**  
Checks Outputs

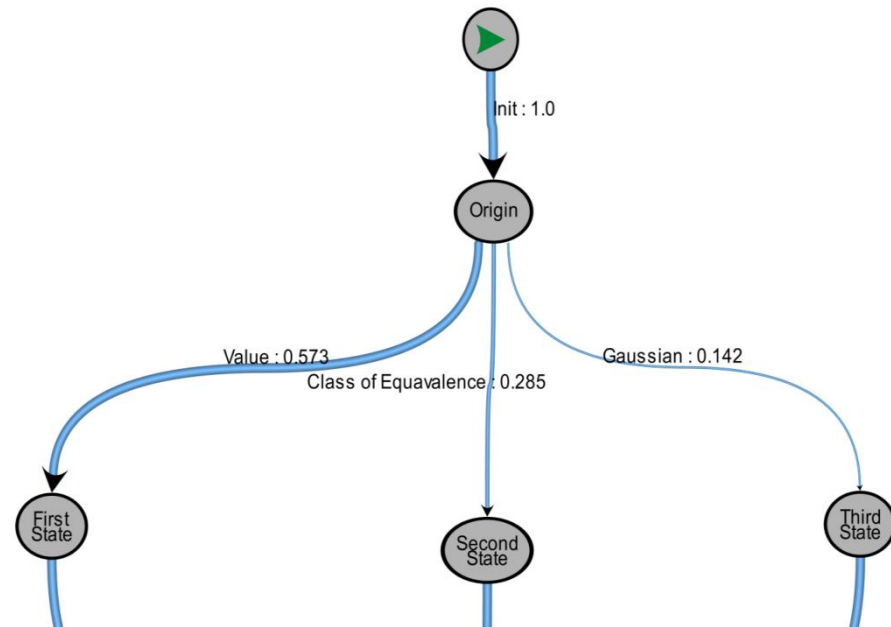
## Profiles can be embedded to qualify the usage model

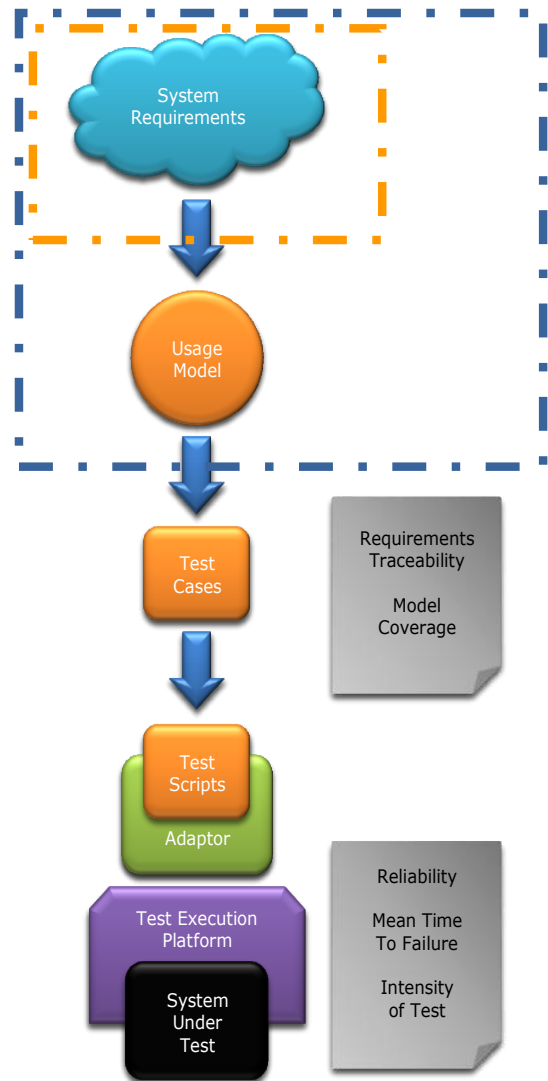
- ⇒ Operational profiles
- ⇒ Test profiles

### Data distribution



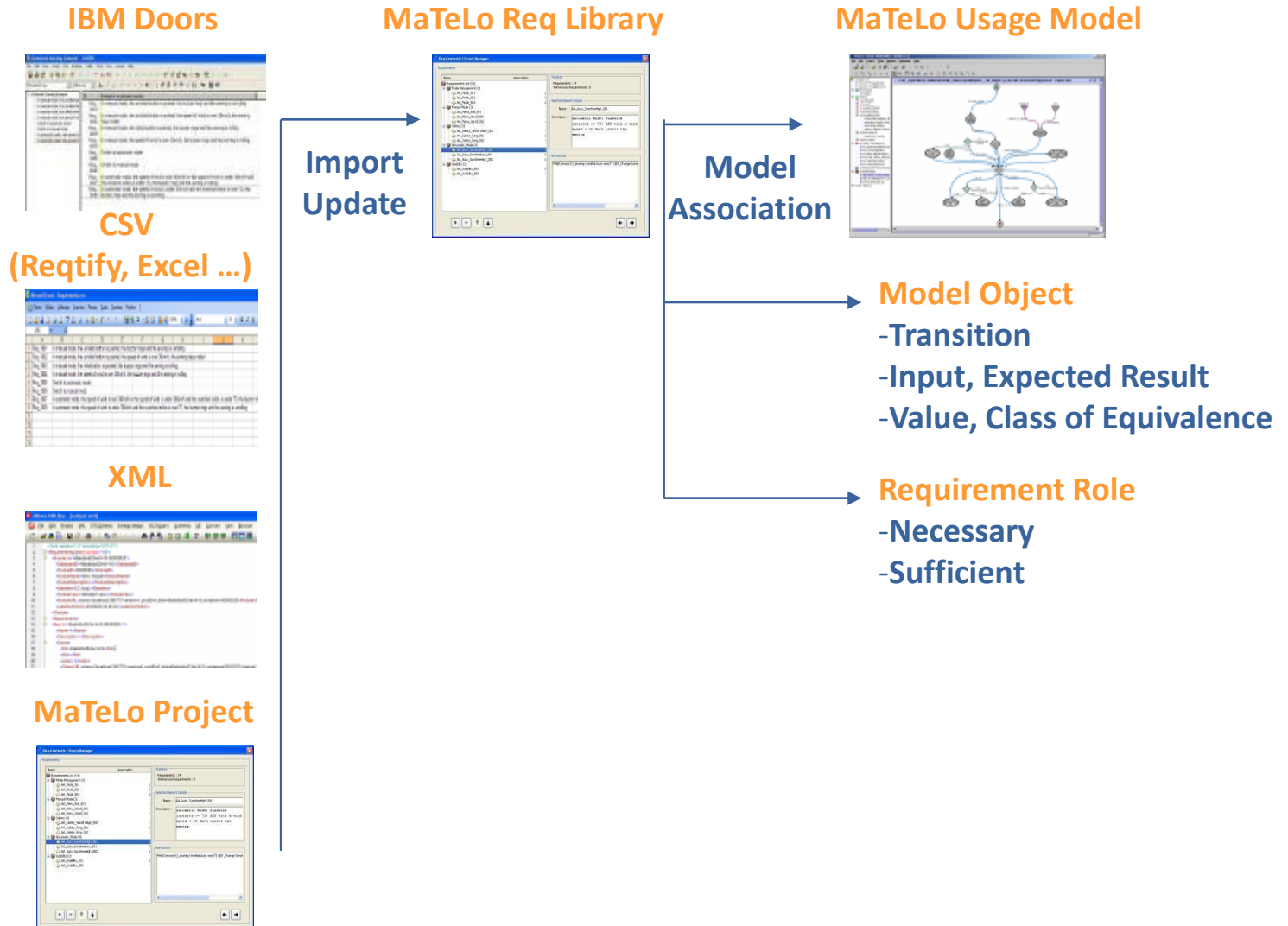
### Usage path probability



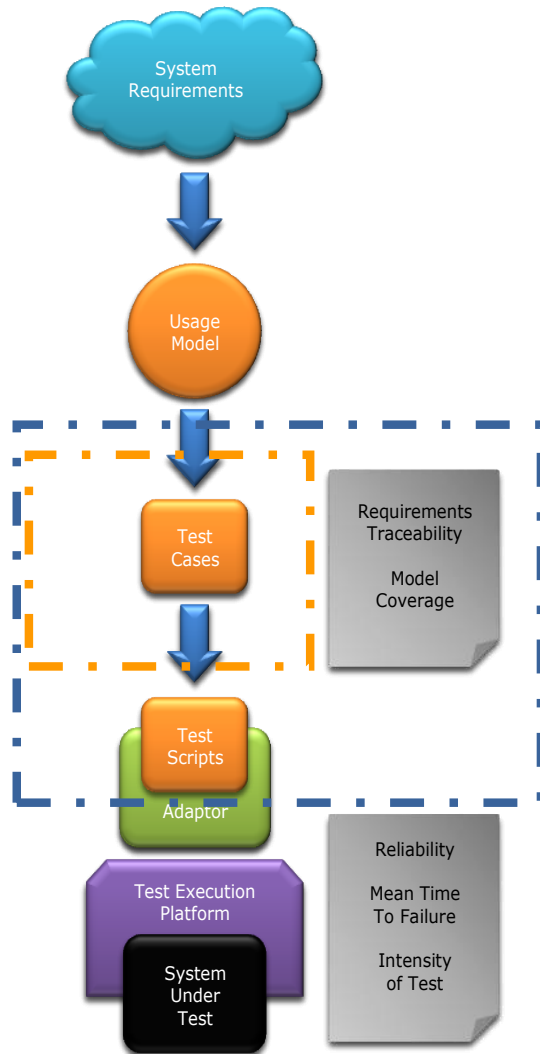


# MaTeLo EDITOR Requirements Management

# REQUIREMENTS MANAGEMENT

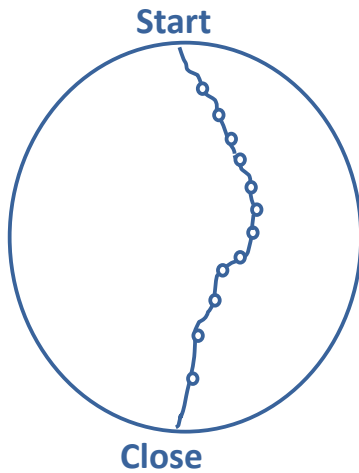






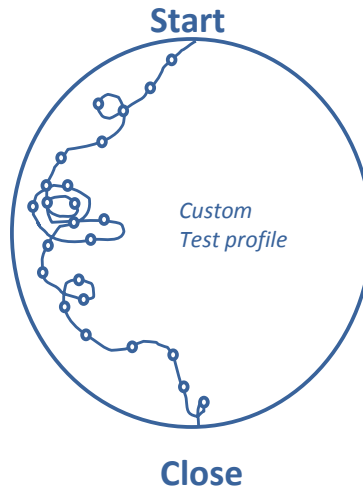
## MaTeLo TESTOR Test Cases Generation

## Most probable approach



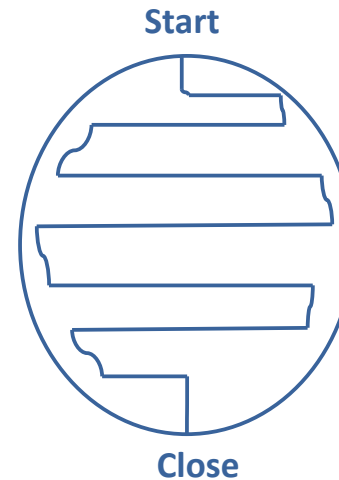
FREQUENCY

## Field application approach



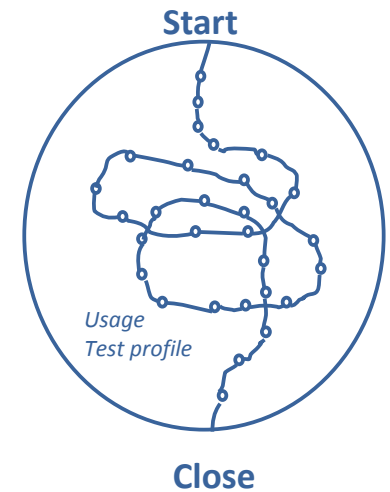
RISK, CRITICISM  
UPDATE...

## Arcs coverage approach



COVERAGE

## Usage approach



USAGE  
(Determinist, Stochastic)

## ❑ DEFINE THE TEST STRATEGY, BY CHOOSING

- ⇒ Test Algorithm
- ⇒ Test Profile
- ⇒ Part of model to test

Generate

# MaTeLo TESTOR: HTML TEST PLAN

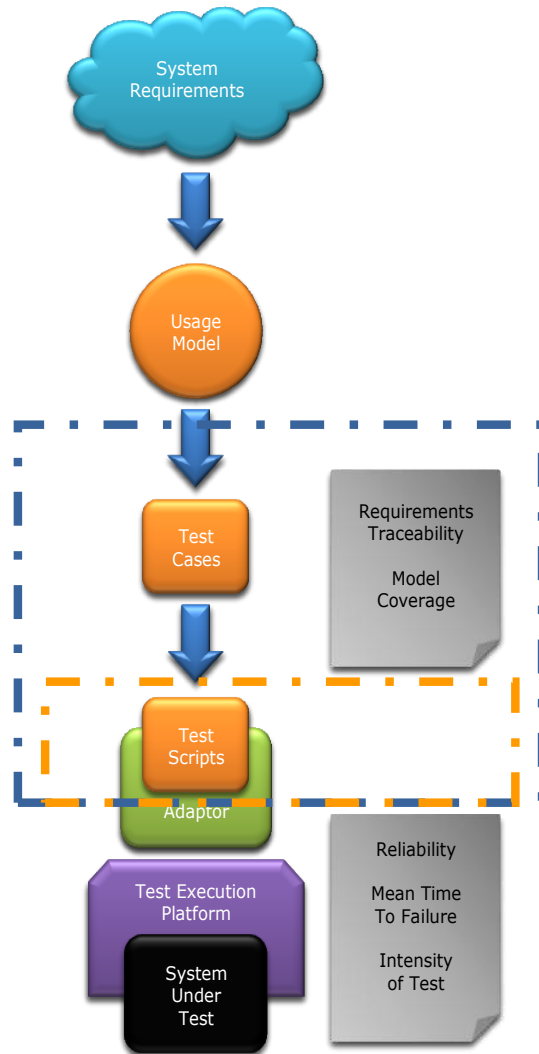
| State            | ID | Description                   | Input  | Requirements            | Expected Result                                   |
|------------------|----|-------------------------------|--|-------------------------|---|
| S_Awning Rolled  | 7  |                               |  |                         |   |
| S_Automatic Mode | 8  | T_Push Button Manual Mode     | Input : Button Manual Mode<br>• myCharString : Push Button Manual Mode         | Requirements : Req_1006 | ERA : State Mode<br>• myCharString == Manual Mode |
| S_Awning Rolled  | 10 | T_Push Unrolled Awning Button | Input : Unrolled Awning Button<br>• myCharString : Push Unrolled Awning Button |                         |   |

State

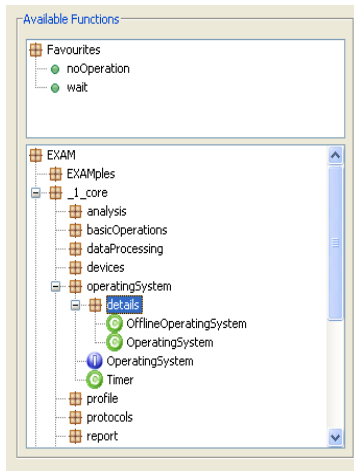
Input

Requirement

Expected Result

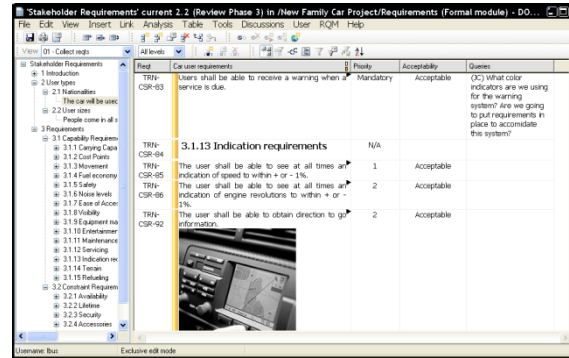


# MaTeLo EDITOR Test Bench Management



## Test Repository

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



## Requirements

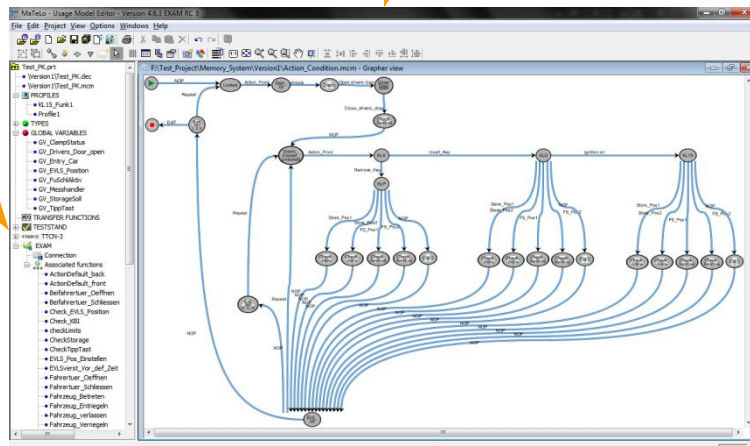
- Name
- UUID
- Description

Association

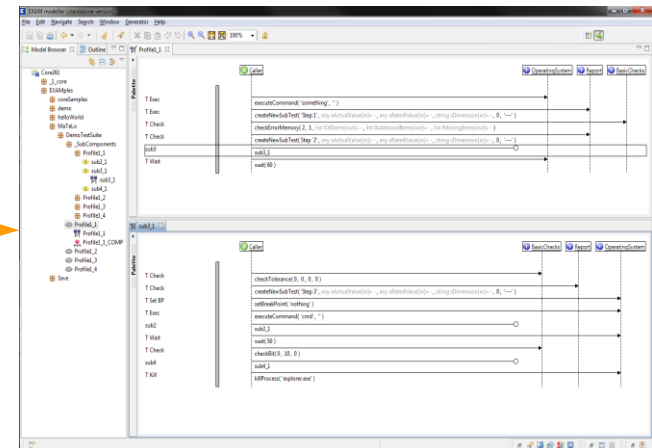
## Automatic Test Cases

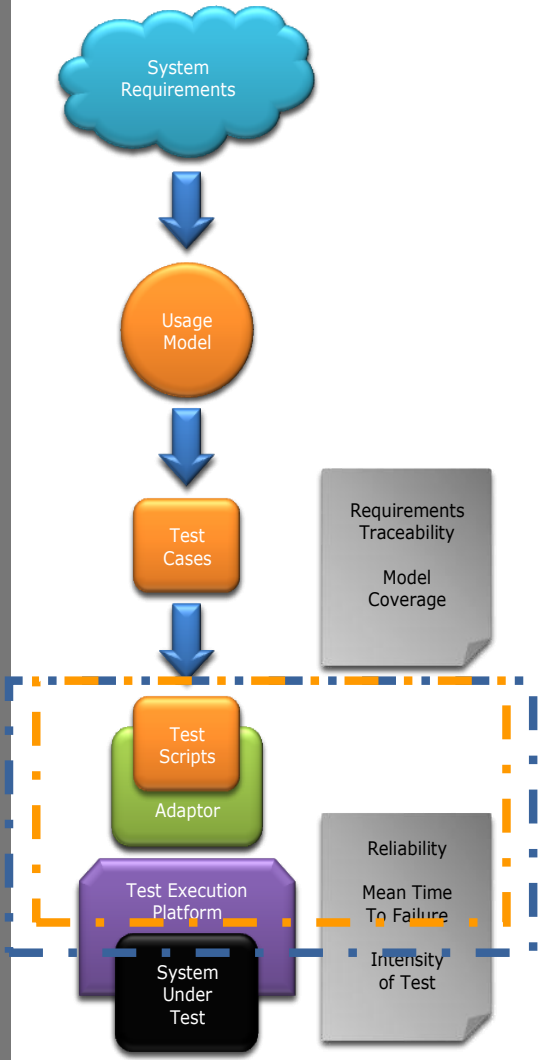
- Automatic call of Test Operation
- Automatic parameterisation
- Requirements association
- Test Case description generation
- Usage model respect

Association



Generate





# EXAM

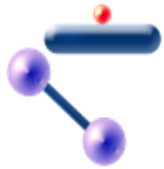
## Extended & Flexible Test Automation

# EXAM SCREENSHOT

The screenshot displays the EXAM modeller (standalone version) interface. The main window is titled "EXAM modeller (standalone version)" and features a menu bar (File, Edit, Navigate, Search, Window, Generator, Help) and a toolbar. The interface is divided into several panes:

- Model Browser:** A tree view on the left showing a hierarchical structure of models. The root is "Core", which contains a sub-model "\_1\_core". Under "\_1\_core", there are several sub-models including "analysis", "checks", "dataAnalysis", "basicOperations", "dataProcessing", "devices", "busDoctor", "climaticChamber", "dataLogger", "functionGenerator", "oscilloscope", "platform", "details", "core", "platform", "carts", "dSPACE", "modena", "novaSim", "stimulus", "mapping", "Platform", "RealTimeSequences", "Stimuli", "powerSupply", "operatingSystem", and "profile".
- Palette:** A central pane containing a list of modeling elements: Select, Marquee, Entity, Class/Interface, Empty Step, Control, alt, alt\_next, while, for, break, Call, and OperationCall.
- Diagram:** A central workspace showing a sequence diagram. It features two lifelines: "Caller" and "BasicMapping". The diagram contains two messages: "checkSystemConfiguration()" and "initMapping(«VariableMapping», 'NovaSim')".
- Properties:** A pane at the bottom showing the "Operations" list. The "Operations" section is expanded, displaying a list of available operations: initPlatform, initMapping, writeValue, readValue, setEvent, startMeasurement, stopMeasurement, downloadApplication, getMeasurementData, deinitPlatform, defineCaptureValueFile, and startStreamToDisk.

# ABSTRACTION LAYERS



MaTeLo

Usage Scenario Description

<< derive >>

Test Case Specification

<< generate >>

Test Flow Control Device Driver

<< control >>

System Under Test

Markov-Chain Usage Model

UML Sequence Diagram

Python Precompiled Application

Hardware In the Loop Test Bench



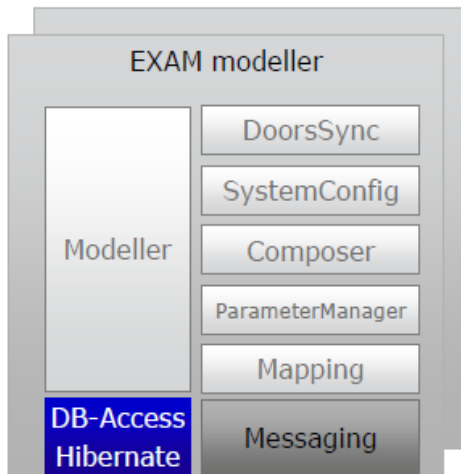
EXAM

Third Party Equipment

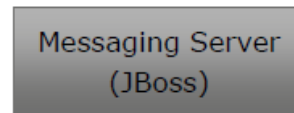
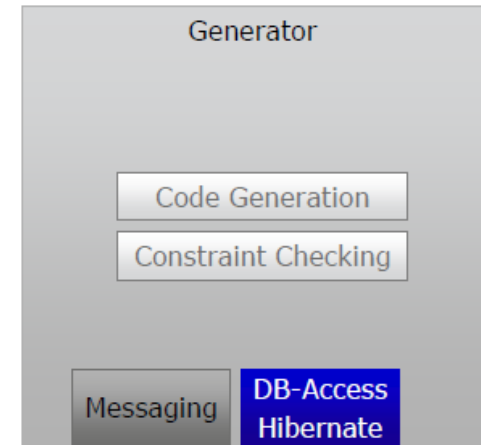




## Multi Users Test Environnement

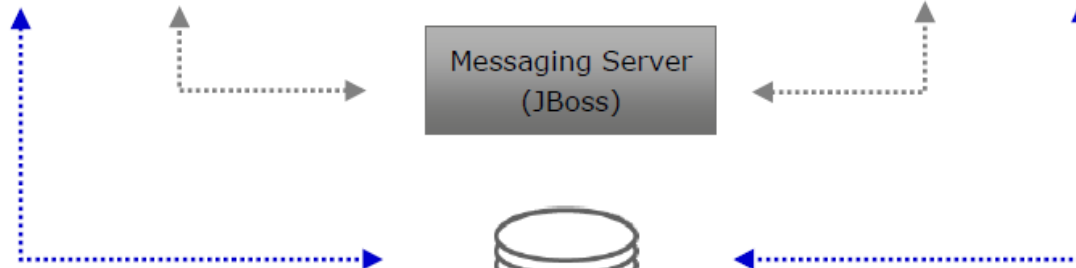


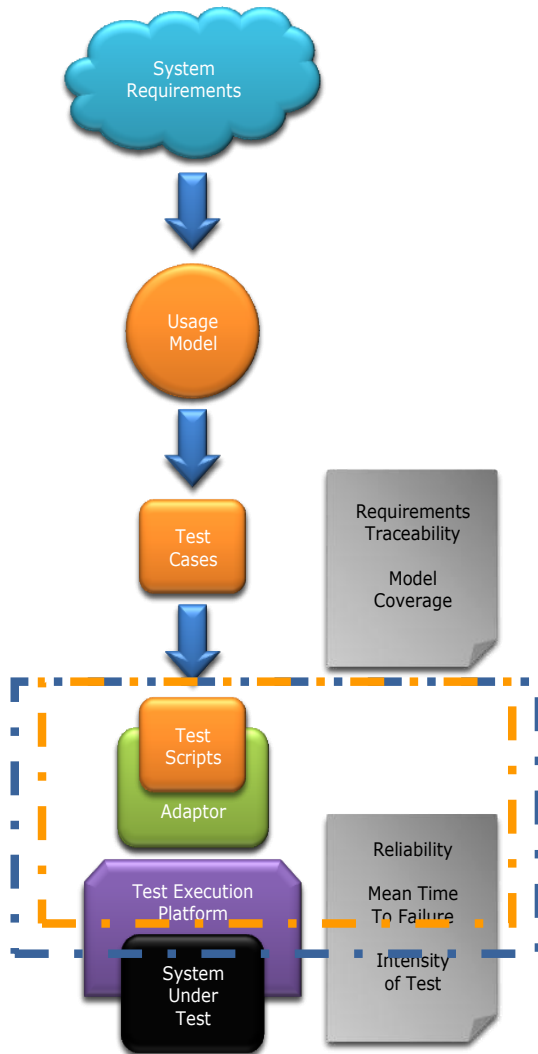
## Test Cases Implementer Python Code Generator



## Test Repository

Project, Libraries, Test Campaign

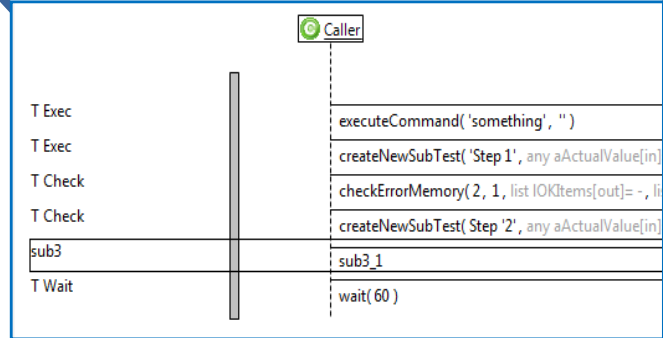
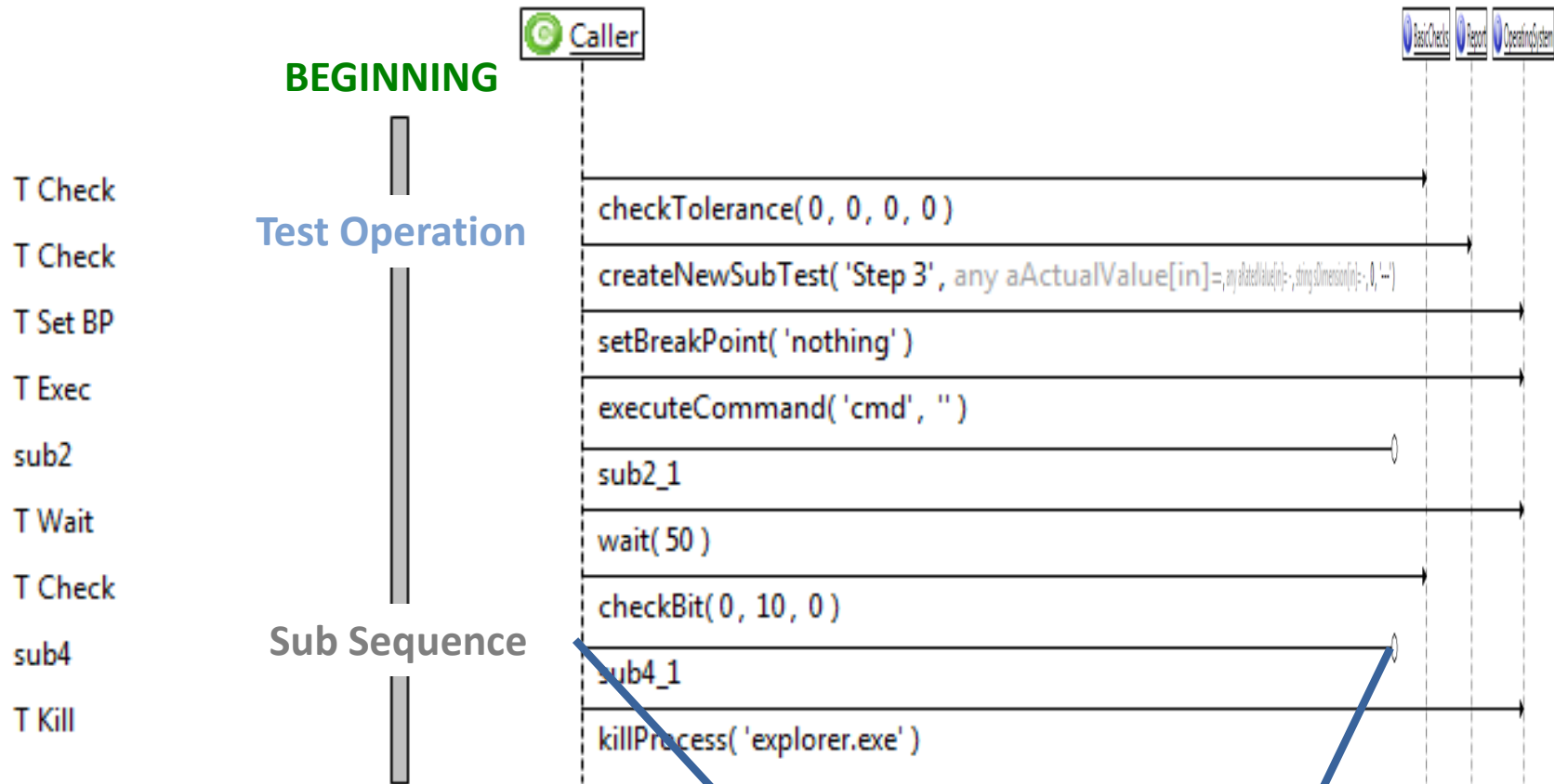




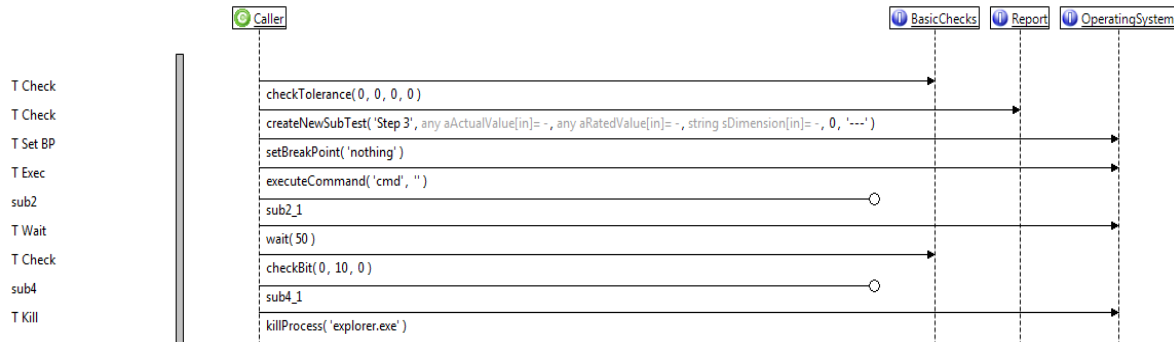
# EXAM Modeller

## Test Cases Description

# EXAM UML TEST SEQUENCE

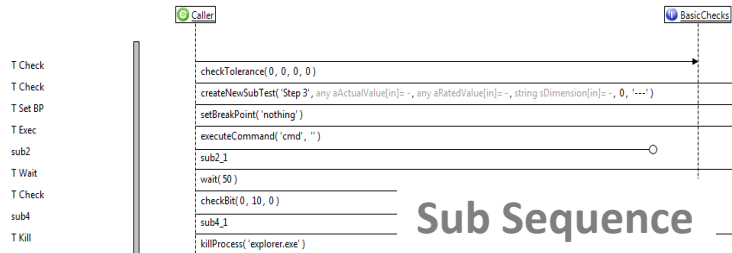


## Test Case Description



INTERFACE

## Test Case Implementations



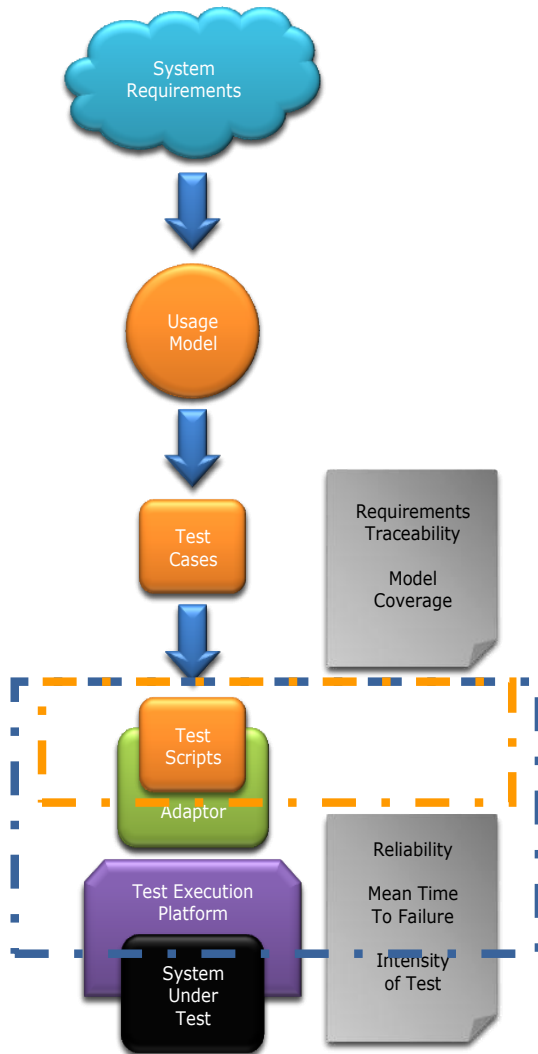
CLASS

```

260 def readValue(self, cTH, sVariable):
261     cTH.methodEntry("__c_r_d.UmlClass_de2a27d0_bfad_43cc_a48e_e96bcc7a...
262
263     sVariable = copy.deepcopy(sVariable)
264
265     '''
266     description:
267     Before the value of the mapped variable is read, the exit
268     simulator variable is checked. The simulator variable check
269     simple read operation, that fails with an exception if the
270     implemented in the method checkSetMapping.
271     '''
272
273     # defines
274     variable = sVariable
275
276     # check the mapping object
277     cTH.checkSetMappingObject(variable, 'car
278
279     # read value from platform
    
```

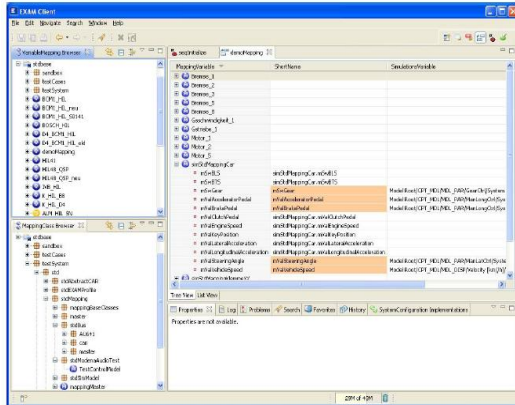
Python Code

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



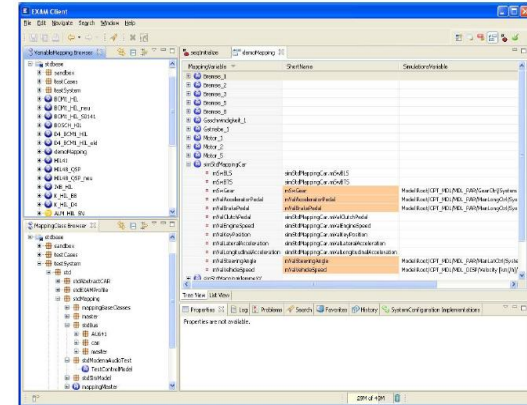
## EXAM Modeller Test Cases Management

## Variable Mapping



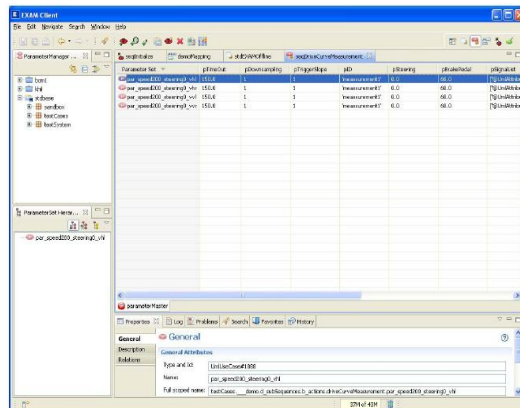
Simulation Model Abstraction

## Short-name Mapping



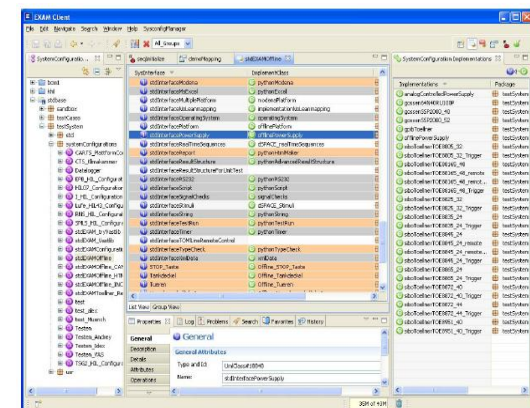
Embedded Network Abstraction

## Parameter Set Composing



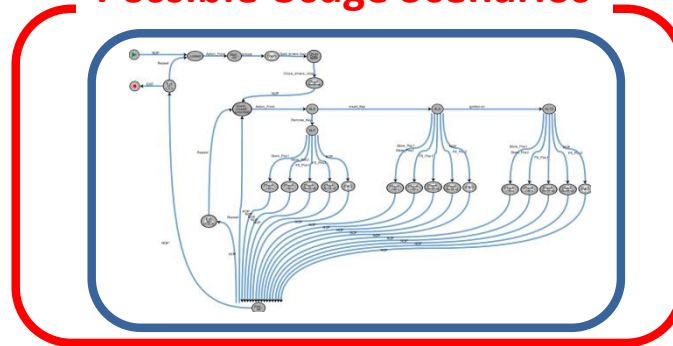
Concrete Test Case Abstraction

## System Configuration

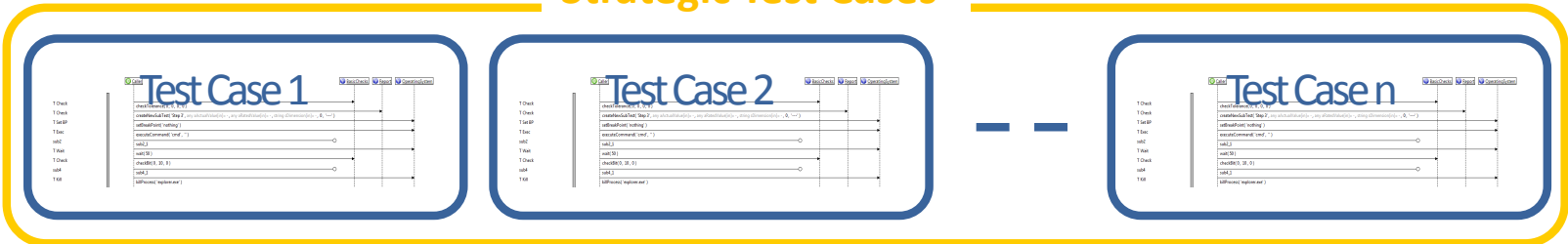


Test Platform Abstraction

## Possible Usage Scenarios



## Strategic Test Cases



## Available Test Configurations

### HIL Platform

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  
...

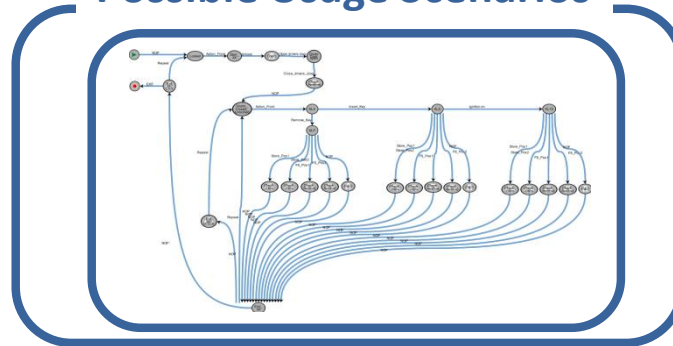
### Test Tools

CANoe  
CANape  
INCA  
MS Excel  
Diag Tool  
...

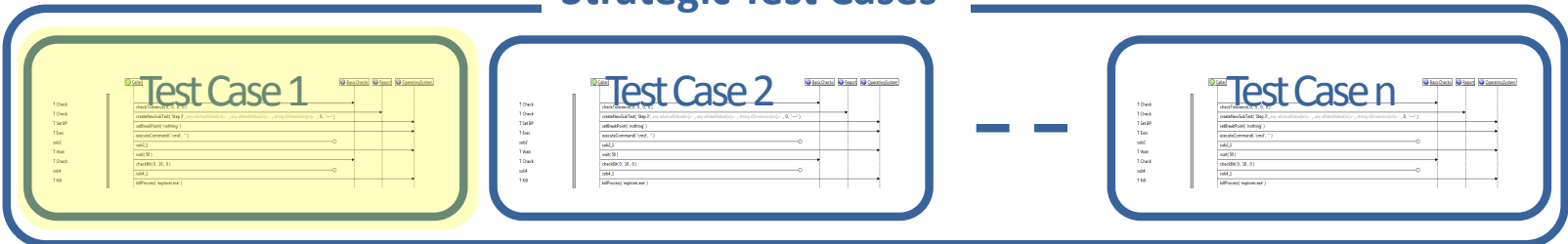
### Env. Model

Gasoline  
Diesel  
Turbo  
Hybrid  
Electric  
...

## Possible Usage Scenarios



## Strategic Test Cases

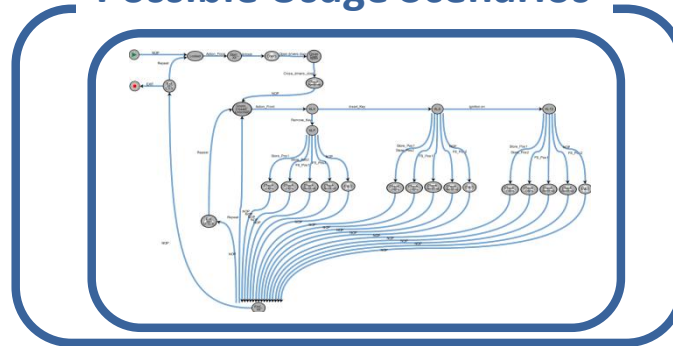


## Available Test Configurations

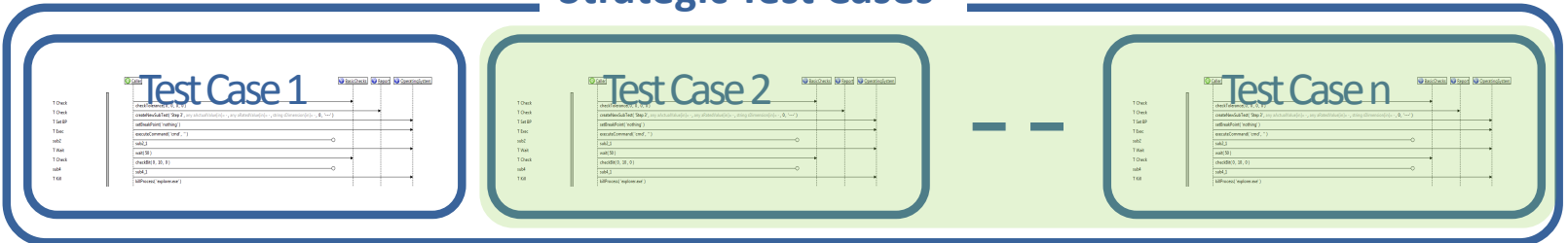
| HIL Platform   | Network  | Functions  | Test Tools  | Env. Model   |
|--|--|--|---|--|
| dSpace<br>NovaSim<br>Carts<br>ASAM HIL API<br>Proprietary<br>... | CAN_1<br>CAN_2<br>LIN_X<br>FlexRay_1<br>FlexRay_2<br>... | Manual Gear<br>Auto Gear<br>Hand Free<br>ACC<br>StartStop<br>... | CANoe<br>CANape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



## Possible Usage Scenarios

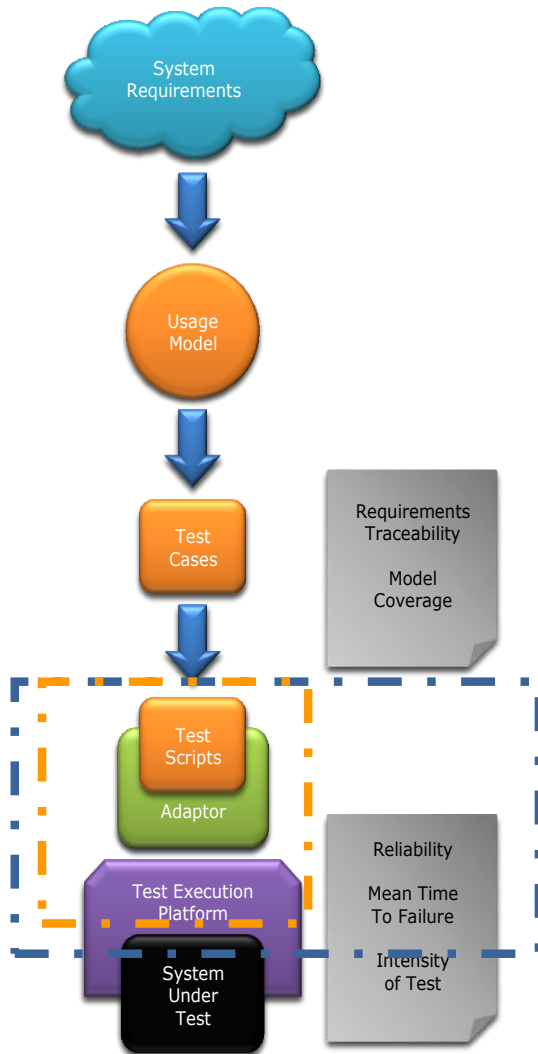


## Strategic Test Cases



## Available Test Configurations

| HIL Platform | Network   | Functions   | Test Tools | Env. Model |
|--------------|-----------|-------------|------------|------------|
| dSpace       | CAN_1     | Manual Gear | CANoe      | Gasoline   |
| NovaSim      | CAN_2     | Auto Gear   | CANape     | Diesel     |
| Carts        | LIN_X     | Hand Free   | INCA       | Turbo      |
| ASAM HIL API | FlexRay_1 | ACC         | MS Excel   | Hybrid     |
| Proprietary  | FlexRay_2 | StartStop   | Diag Tool  | Electric   |
| ...          | ...       | ...         | ...        | ...        |



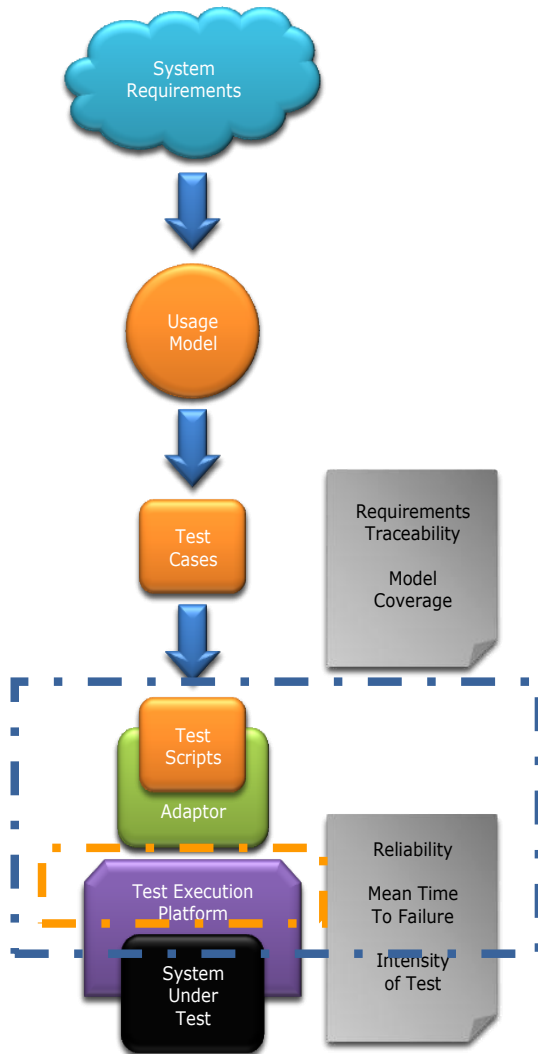
## EXAM Test Runner Test Cases Automatic Execution

## AUTOMATIC TEST SCHEDULER

The screenshot displays the EXAM TestRunner application interface. The main window is titled "EXAM TestRunner" and contains several panels:

- ExchangeData:** A tree view showing the test structure, including folders like "testRun", "typeCheck", "metaData", "dataProcessing", and "convertData".
- ExchangeBrowser:** A tree view showing the project structure, including folders like "bcm1", "stdbase", "sandbox", "testCases", and "std".
- Console:** A window displaying the test run log, starting with "TestRun: Started at: Thu Nov 13 18:12:03 CET 2008". It shows a long list of numerical values and a summary of the test case: "syncCanLog".
- TestRun State:** A panel showing the current test state, including "LogLevel: DEBUG", "TestCase: 90 of 312", and "38 errors".
- General Test Data:** A panel showing an overview of the current TestElements, including "samples\_SUITE", "3de329a7-45d1-4a2e-9e66-3820a55673c3 stdEXAMOffline", and "syncCanLog".

The bottom status bar shows the test run progress: "00:00:00:25", "38 errors", and "TestRun: (28%)".



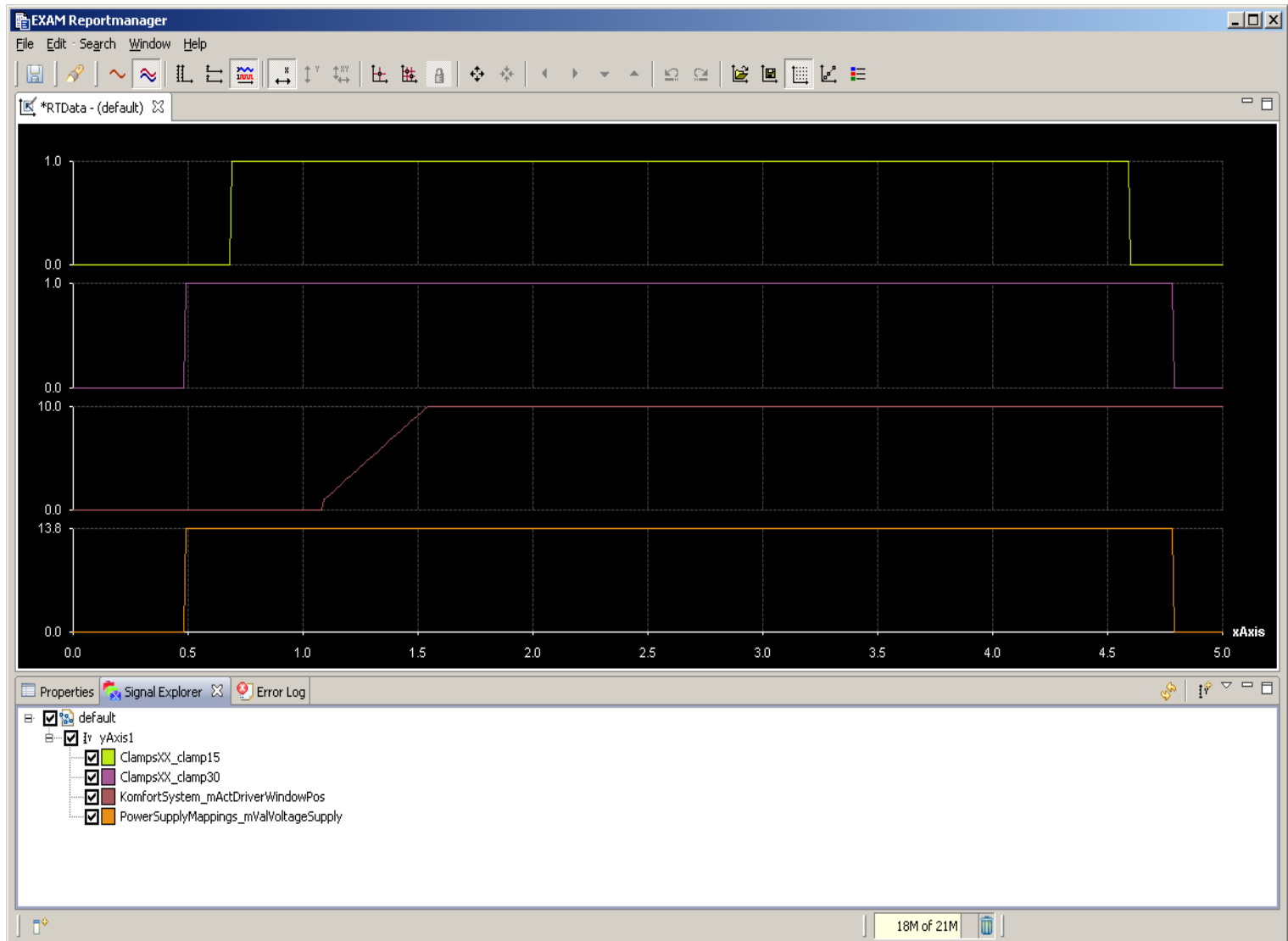
## EXAM Report Manager Verdict and Analyze Test Runs

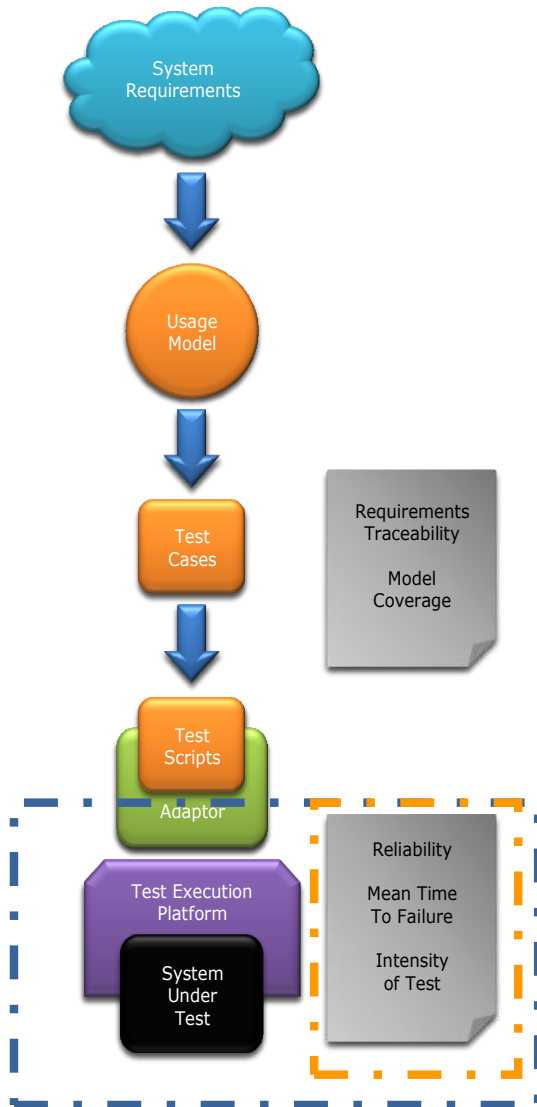
## TEST CAMPAIGN RESULT & VERDICT MANAGEMENT

The screenshot displays the EXAM ReportManager application window. The interface is divided into several sections:

- Left Panel (Tree View):** Shows a hierarchical view of test results. The selected test case is `TimeOut_ESP_01`, which is highlighted in blue. Other tests listed include `TimeOut_Diagnose_01`, `TimeOut_Gateway_05`, `TimeOut_Klemmen_Status_01`, `TimeOut_Kombi_01`, and `TimeOut_VIN_01`.
- Top Panel (SubTests):** Displays the results for the selected test case. It shows a list of subtests with their status (Pass/Fail) and a description. The subtests are:
  - Ausfall der Botschaft TX\_ECAN\_mESP\_01 fü (Info icon)
  - 84: BotschaftsTimeOut (Fail icon)
  - Fehler aufgetreten! NV hatte bereits vor Einl (Fail icon)
  - NW\_KD\_Fehler (Pass icon)
  - NW\_Text (Pass icon)
  - TimeOut Qualifizierung (Fail icon)
  - Aktiv Qualifizierung (Fail icon)
- Right Panel (NewSubTest Details):** Provides details for the selected subtest. The details include:
  - Creation Time: (empty field)
  - Valuation: PASS (dropdown menu)
  - Label: NW\_Text (text field)
  - Actual-Value: 3 (text field)
  - Rated-Value: 3 (text field)
  - Dimension: (empty field)
  - Comment: --- (text area)
- Bottom Panel (Details):** Shows the overall details for the test case. The details include:
  - Name: TimeOut\_ESP\_01 (text field)
  - ID: 0002fcab C898F47B-2CB4-4CF1-838F-09743D49E84F (text field)
  - Comment: --- (text area)

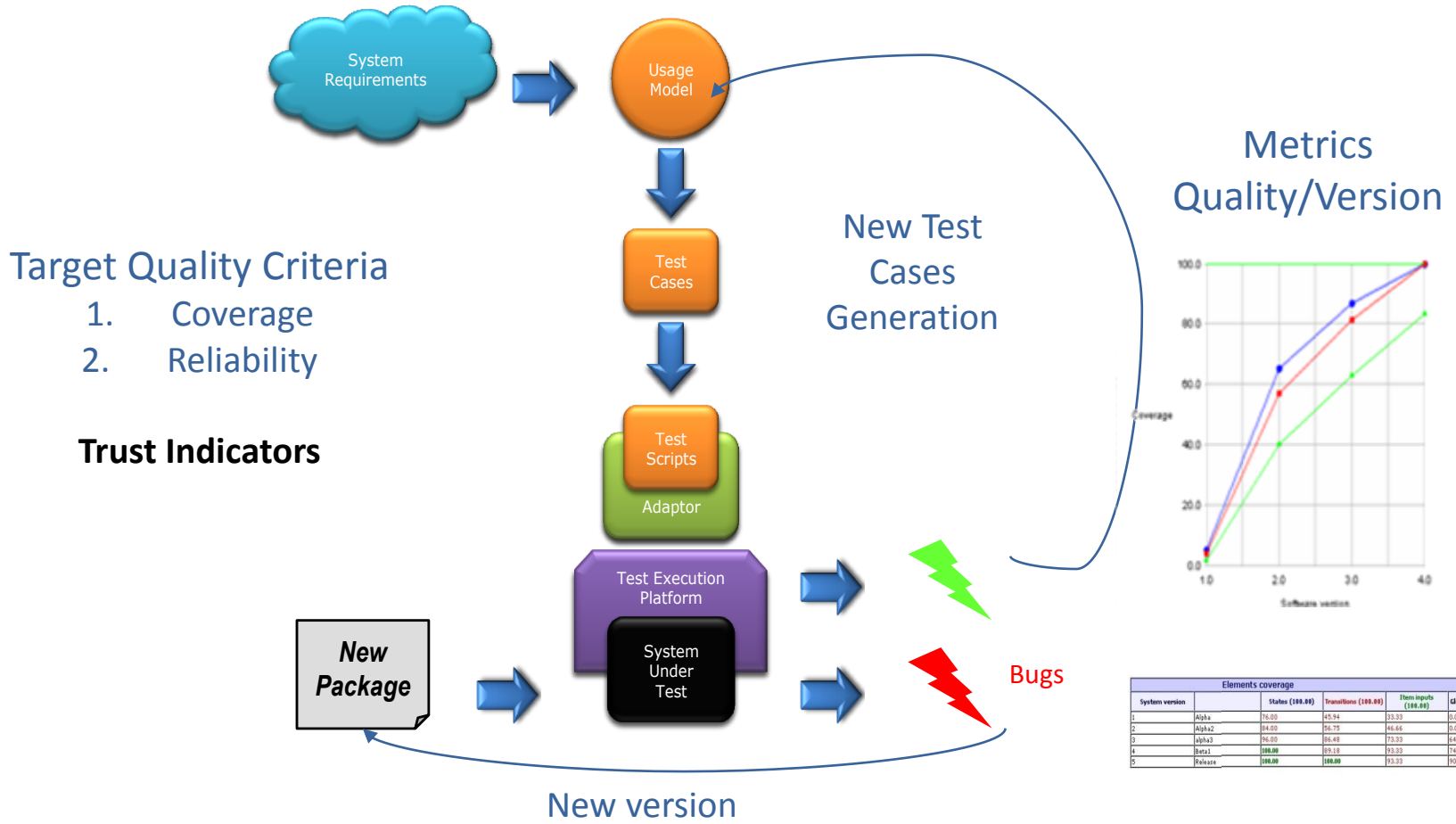
## REAL TIME RECORD ANALYSIS AND VERDICTS





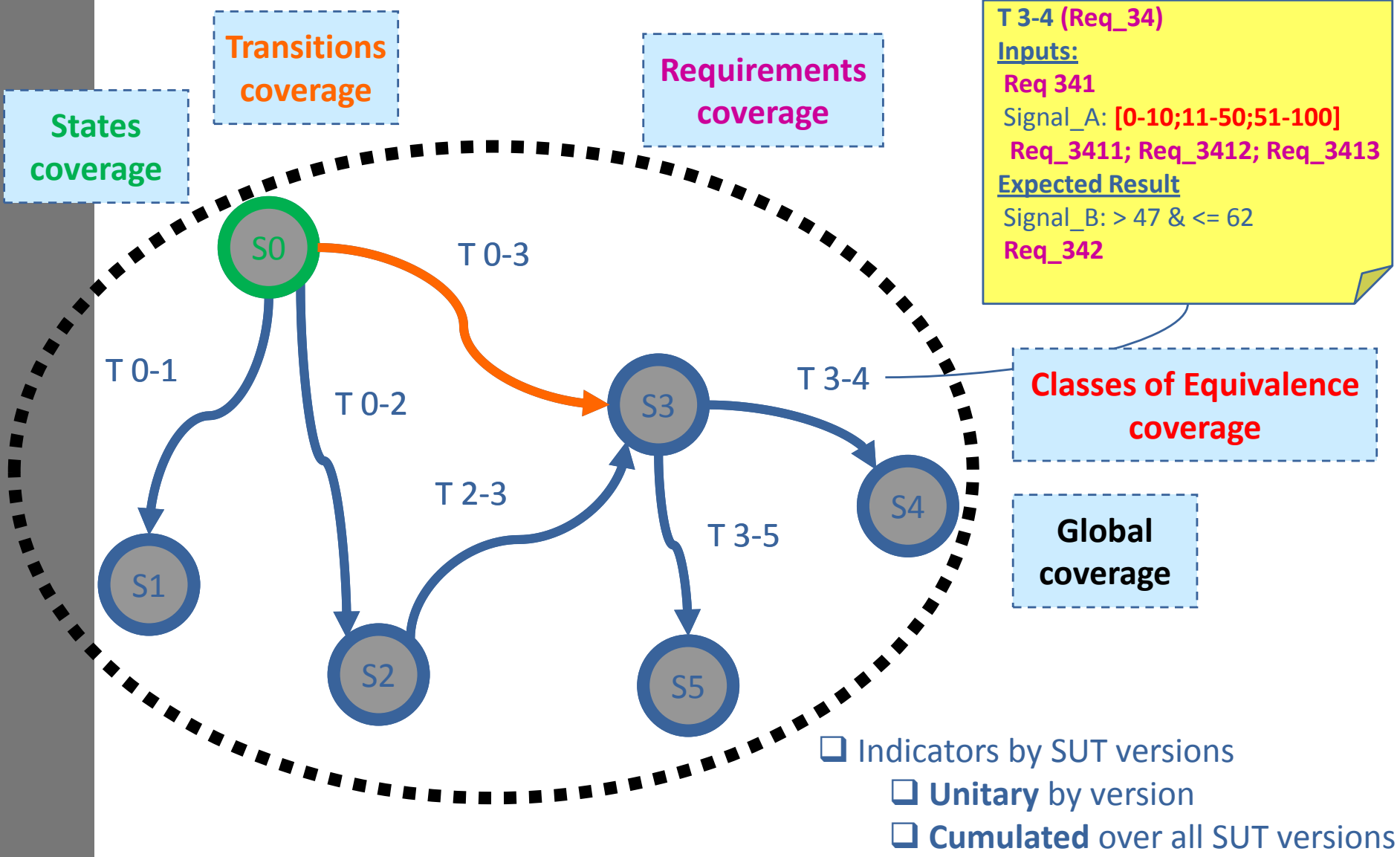
## MaTeLo TCA Test Campaign Analysis

# TEST CAMPAIGN PROCESS

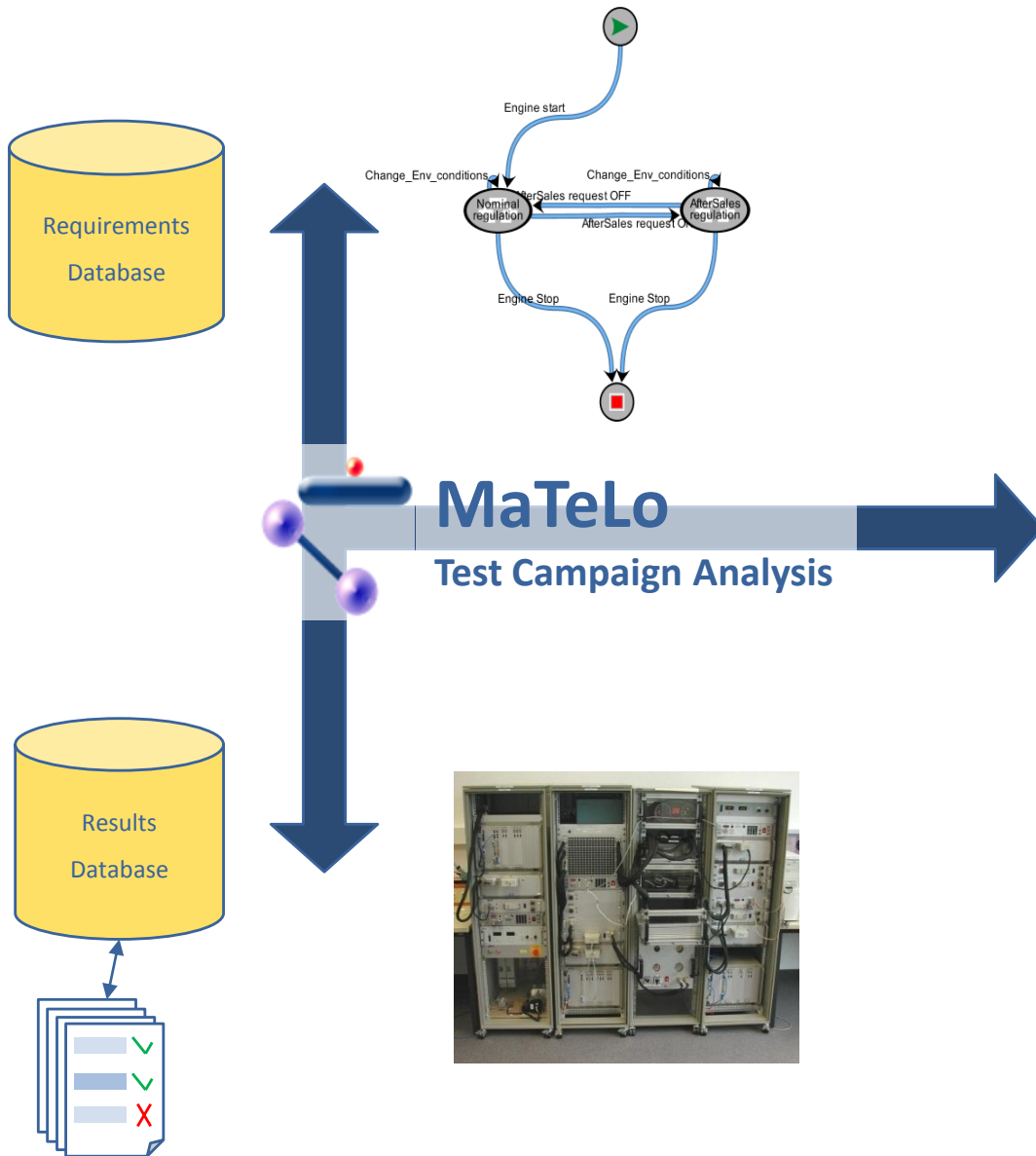




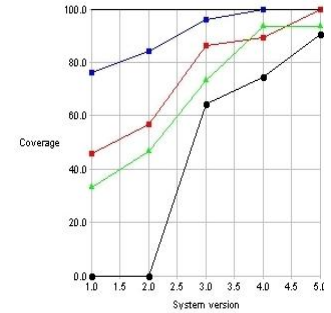
# COVERAGE CAPABILITIES



# TEST CAMPAIGN REPORT



## Model Coverage



## Requirements Coverage

| List of covered requirements by the TestSuite |                            |                           |                |
|---|----------------------------|---------------------------|----------------|
| RequirementName                               | Requirements specification | Coverage                  | Coverage Ratio |
| Req_006                                       | Function01_MotorBlockade   | 100%                      | 2              |
| Req_002                                       | Function01_MotorBlockade   | 100%                      | 4              |
| Req_004                                       | Function01_MotorBlockade   | 100% (Full Exhaust Valve) | 2              |
| Req_001                                       | Function01_MotorBlockade   | 100% (Full Exhaust Valve) | 2              |
| Req_008                                       | Function01_MotorBlockade   | 100% (Full Exhaust Valve) | 2              |
| Req_009                                       | Function01_MotorBlockade   | 100% (Full Exhaust Valve) | 2              |

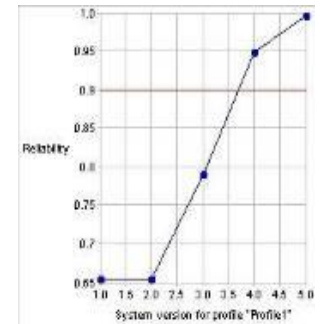
  

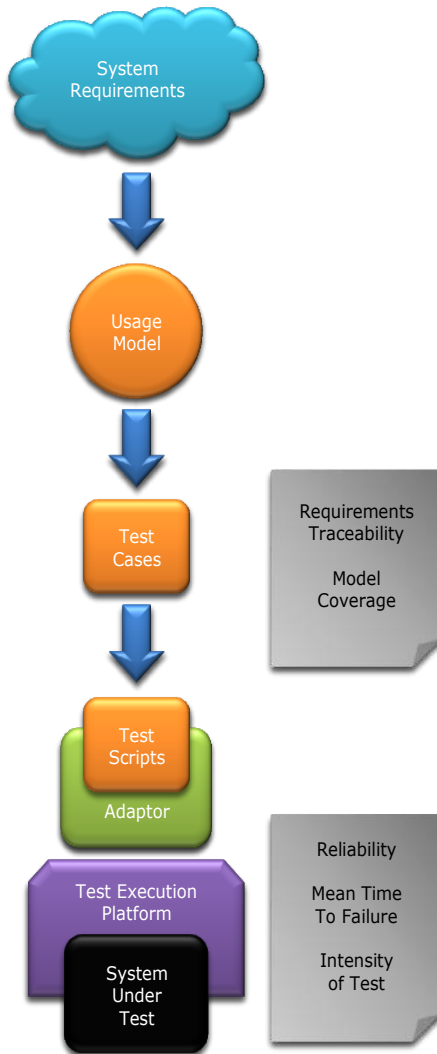
| List of partially covered requirements by the TestSuite |                            |          |                |
|---|----------------------------|----------|----------------|
| RequirementName   | Requirements specification | Coverage | Coverage Ratio |
| Req_007   | Function01_MotorBlockade   | 100%     | 30%            |

| List of uncovered requirements by the TestSuite |                            |
|---|----------------------------|
| RequirementName                                 | Requirements specification |
| Req_003   | Function01_MotorBlockade   |

## 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](http://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](http://www.all4tec.net)

[www.exam-ta.de](http://www.exam-ta.de)