NSPYRE

making technology matter



Model based testing in case of complex systems with data dependency

Rachid Kherrazi

21 November 2013 Hampshire Hotel – Plaza Groningen

Dutch Testing Day



NSPYRE

making technology matter

Presentation outline

- Introduction
  - Philips Healthcare & typical products
  - FXD department & FD subsystem
- Model Based Testing
  - Motivation for Model Based Testing
  - Model-Based Testing with Spec Explorer
  - Device Under Test and Problem Statement
- Approach
  - Constraint Based Testing
  - Results
- Summary

Rachid Kherrazi	Rachid.Kherrazi@Nspyre.nl
Vivek Vishal	Vivek.Vishal@Nspyre.nl
Mehmet Kovacioglu	Mehmet.Kovacioglu@Philips.com
Mohammad Reza Mousavi	M.R.Mousavi@tue.nl



NSPYRE

making technology matter

Nspyre is the leading specialised IT service provider where technology matters.

MARKET SEGMENTS

- High Tech
- Traffic & Infra
- Industry

PHILIPS sense and simplicity

ASML

Rijkswaterstaat

VanDerLande INDUSTRIES

Defensie

AREA'S OF EXCELLENCE

Systems Engineering /Model Driven Engineering / **Model Based Testing** /Industrial Automation / Simulation / Big Data / Mobile Solutions



NSPYRE

making technology matter

PHILIPS

sense and simplicity

Philips Healthcare site in Best
(The Netherland)

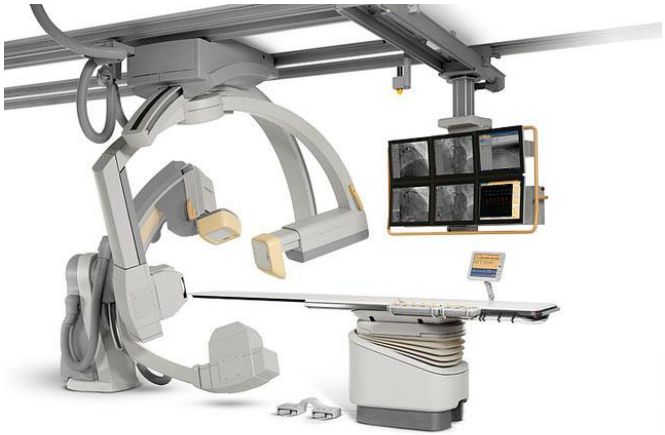
Approximately 3,000 people, of whom 1,000 are directly involved in research and innovation



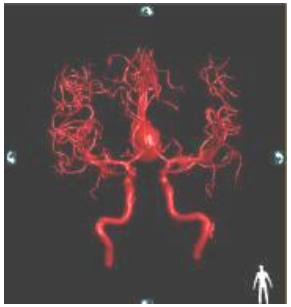
NSPYRE

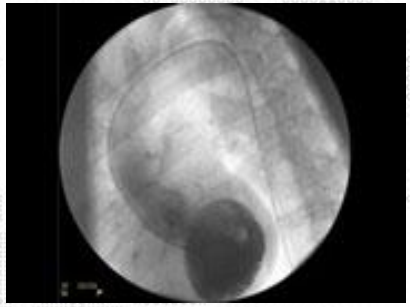
making technology matter

Philips Healthcare / typical products



X-ray **systems**






Interventional and diagnostic X-ray, Cardio Vascular, visualization of blood vessels….



NSPYRE

making technology matter

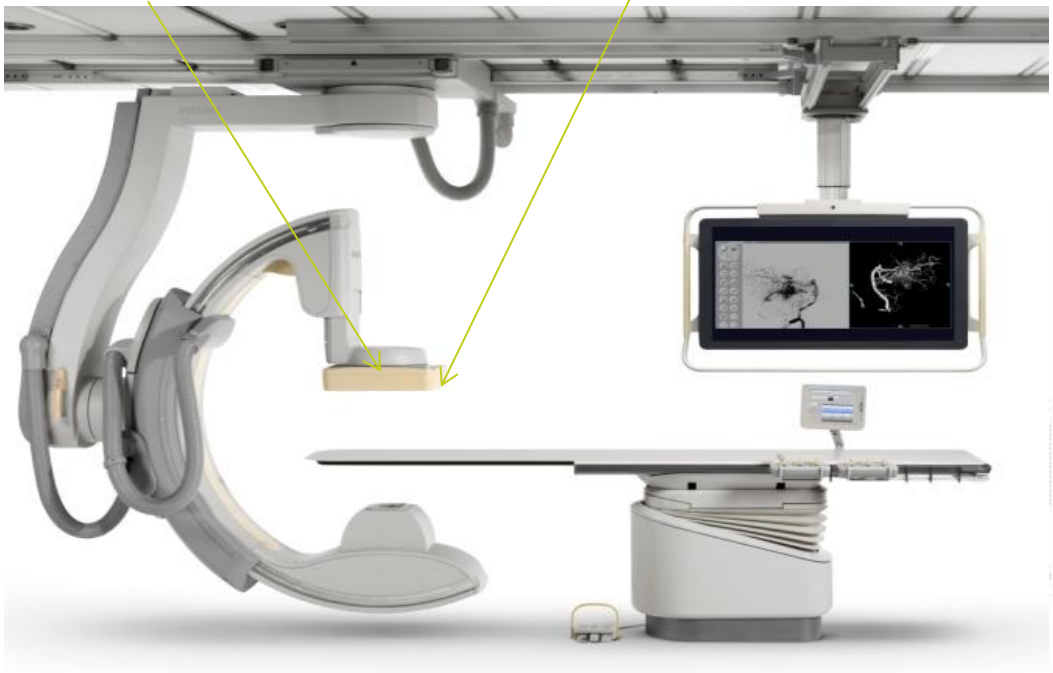
Philips Healthcare / FXD

Flat Detection **(FD) subsystem**

Flat Detector

Grids

Controller

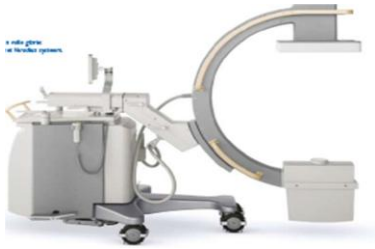


Flat X-Ray Detection (FXD) department develops Flat Detection (FD) subsystems including a flat detector, controllers and Grids.



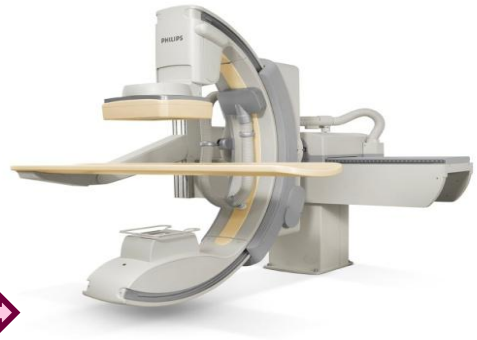
NSPYRE

making technology matter



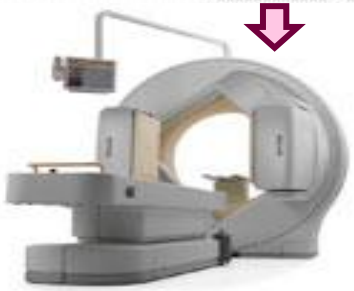
Mobile Surgery

FC Alpha



MultiDiagnost Eleva

Allura Xper



Bright View SPECT



NSPYRE

making technology matter

Philips Healthcare / product verification

In the past **subsystem** tests were performed manually on a **full system**

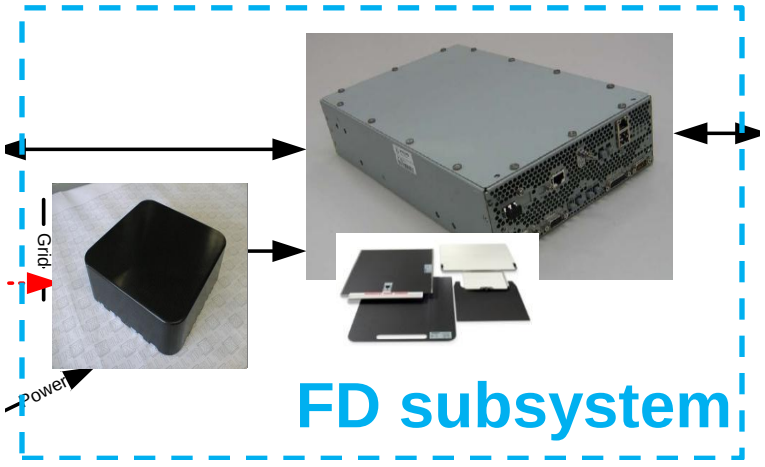
which results in **long test time** and **dependency** with system group



NSPYRE

making technology matter

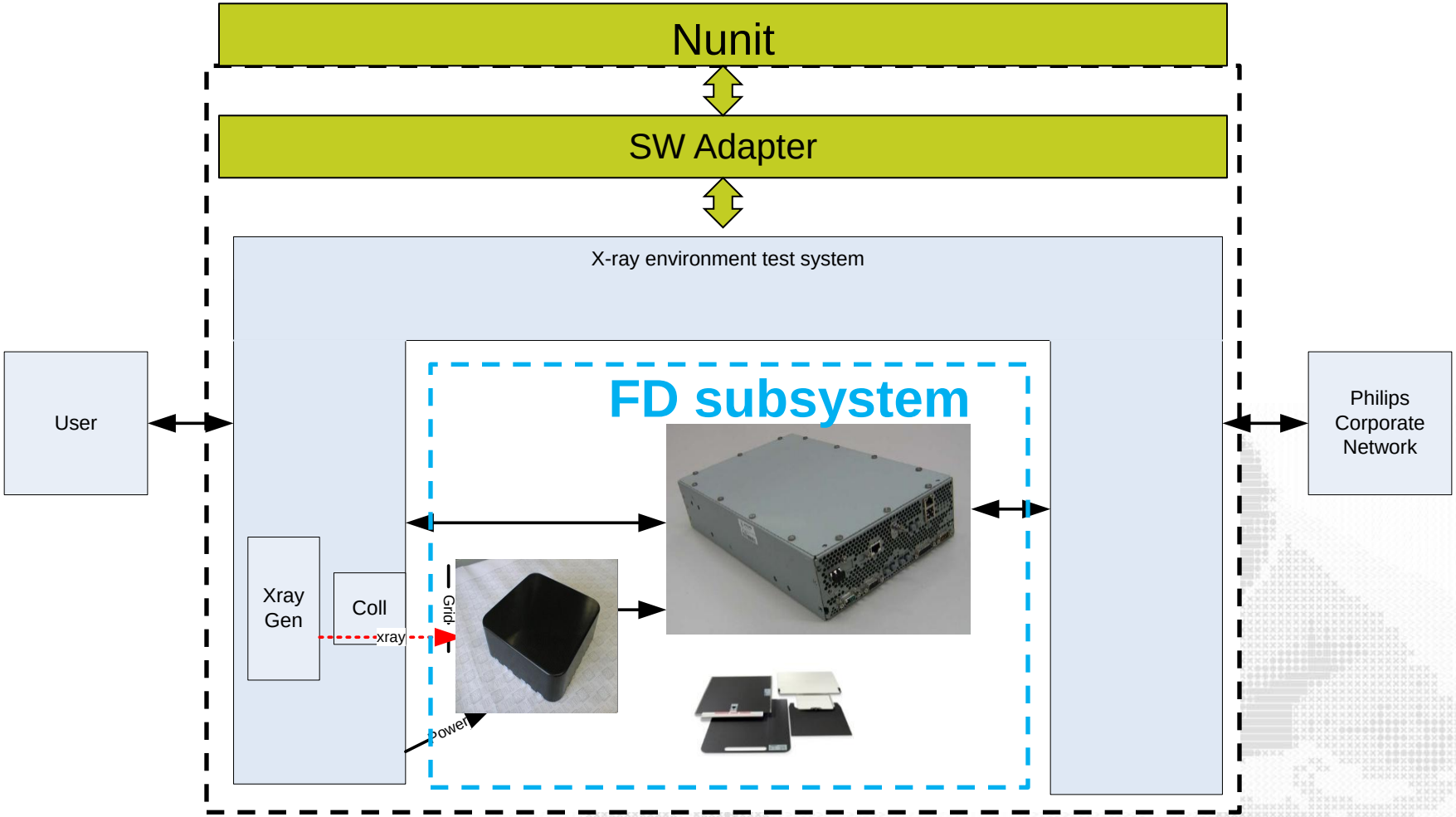
Now we isolate **subsystem (Device Under Test)** and simulate the rest.





NSPYRE

making technology matter



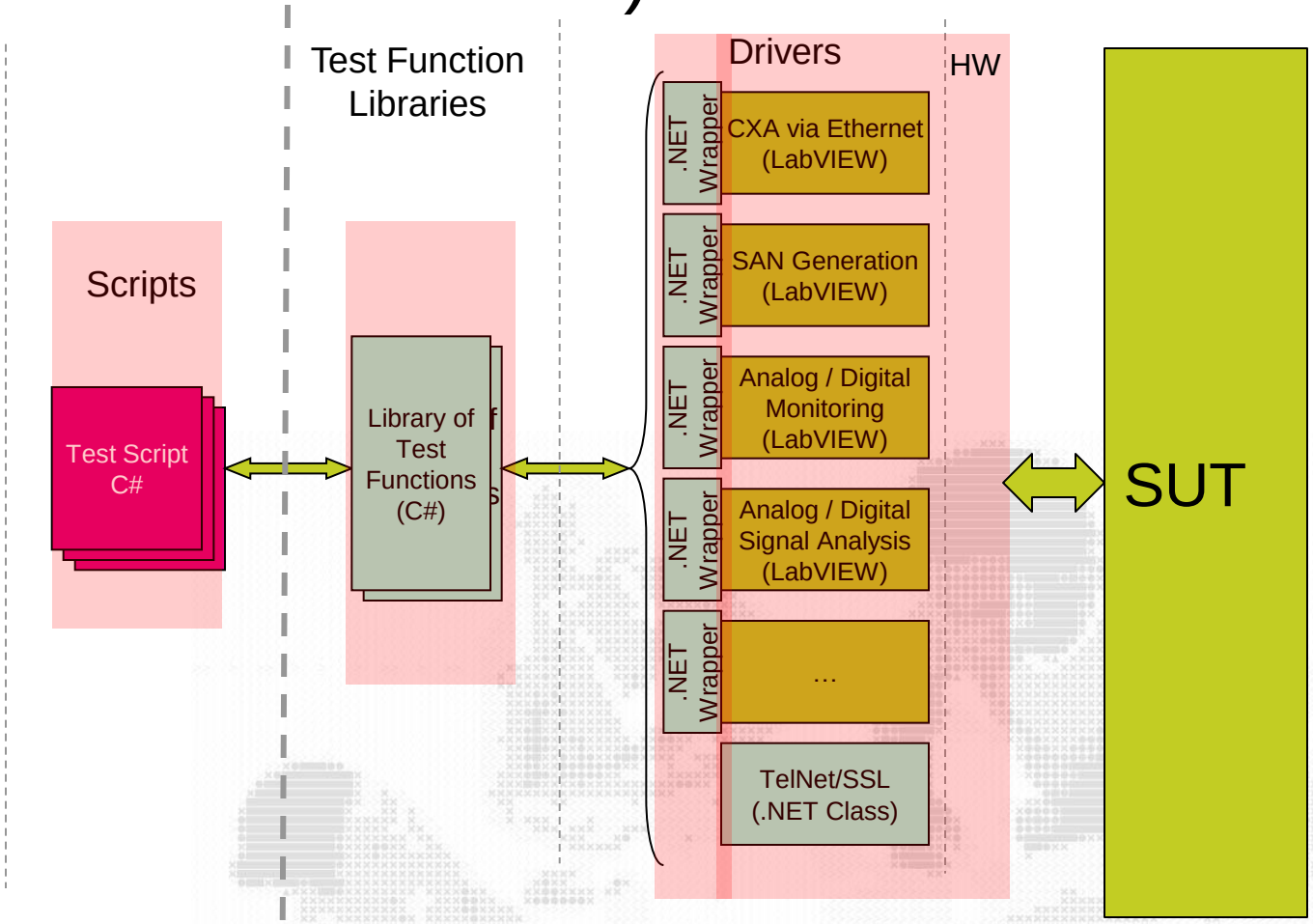
Our test framework is based on the well-known **Nunit** framework and contains everything (HW/SW) that is needed for testing our **DUT**, ➔ short test time and independent from system group



NSPYRE

making technology matter

Adapter (SW <-> HW interface)





NSPYRE

making technology matter

Manual testing

Scripts

Test Script C#

Test Function Libraries

Library of Test Functions (C#)

Drivers

HW

.NET Wrapper — CXA via Ethernet (LabVIEW)

.NET Wrapper — SAN Generation (LabVIEW)

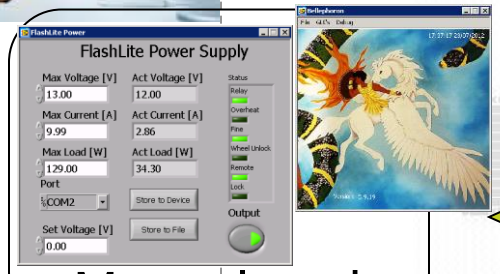
.NET Wrapper — Analog / Digital Monitoring (LabVIEW)

.NET Wrapper — Analog / Digital Signal Analysis (LabVIEW)

.NET Wrapper — …

TelNet/SSL (.NET Class)

SUT



Manual testing



NSPYRE

making technology matter

Automatic test execution (Nunit)

Execution

Test automation

Test Function Libraries

Scripts

Test Script C#

Library of Test Functions (C#)

Drivers

HW

.NET Wrapper — CXA via Ethernet (LabVIEW)

.NET Wrapper — SAN Generation (LabVIEW)

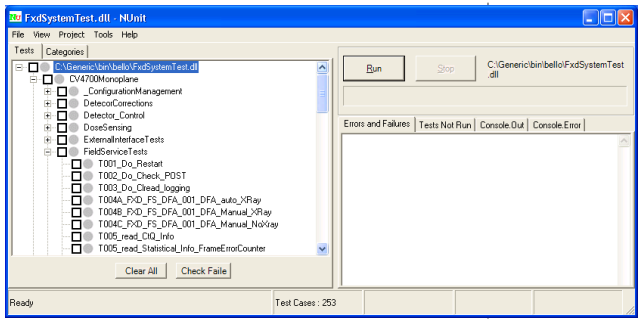
.NET Wrapper — Analog / Digital Monitoring (LabVIEW)

.NET Wrapper — Analog / Digital Signal Analysis (LabVIEW)

.NET Wrapper — …

TelNet/SSL (.NET Class)

SUT





NSPYRE
making technology matter

Context regarding V-Model

3 main steps in our subsystem test process

System

Subsystem

Module

Test case specification
(design of logical test case)
(reduction of number of test cases by application of test techniques

Test execution
(reporting)

Test case generation
(design of physical test case)
(selection of input values and calculation of expected results)



NSPYRE
making technology matter

Motivation for MBT Flashback and some historical data

	Manual testing	Automated test execution Using Nunit
Test case specification (design of logical test case)	(3 man weeks)	(3 man weeks)
Test case generation (design of physical test case) (Actions & expected results)	(3 man weeks)	(C# scripting for Nunit) (4 man weeks)
Test case execution (and reporting)	(6 man weeks)	(1 man weeks)
	average 12 man weeks /release	average 8 man weeks /release



NSPYRE

making technology matter

Test (Automation) Improvement Roadmap

Efficiency
Effectiveness

- Manual testing
- 12 man weeks

- Automated test execution and reporting
- 8 man weeks

- Automated test case generation, test execution and reporting
- Target: < 4 man weeks
- Improved test effectiveness and efficiency

Past

Present

Future



NSPYRE

making technology matter

Model based testing is the automation of test case generation

Manual — Automatic

Test case specification
(design of logical test cases)

Test case generation
(design of physical test case)
(Actions & expected results)

Test case execution
(and reporting)

Traditional manual testing

Traditional automatic testing

Manual C# Scripting

Model based testing

Modeling
C# or UML

Generate
Scripting

Testing skills

+ Scripting skills

+ Modeling skills



NSPYRE

making technology matter

Testing using Nunit

User/Architect

Requirements
Interface
Specifications

Test
design

Tester

design

SW/FW developer

Test
report

feedback

Implementation
code

1

SW adapter

2

Manual creation of Nunit test cases
(C# programming)

Test framework

Nunit dll

MS Visual Studio + Nunit

Development
environment

SUT

SW/FW Implementation



NSPYRE

making technology matter

MBT with Spec Explorer

User/Architect

Requirements
Interface
Specifications

Test
design

Tester

design

SW/FW developer

Test
report

feedback

Implementation
code

1
SW adapter

2
Manual create of test Model
(C# programming)

Test framework

Nunit dll

Automatic generation of
Test cases

Development
environment

MS Visual Studio + Nunit + Spec Explorer

SUT

SW/FW Implementation



NSPYRE

making technology matter

Modeling with Spec Explorer: Look and Feel

C# Model (or other .Net Language)

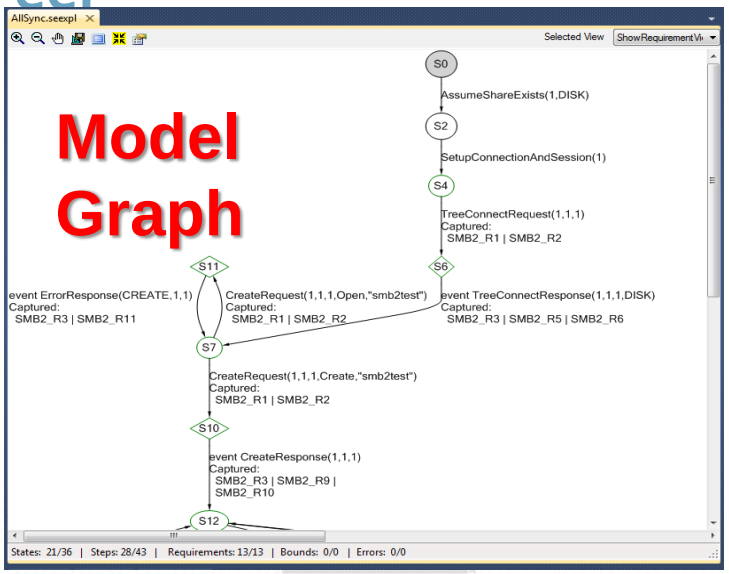
```
Model.cs
SMB2.Model.Model
TreeConnectRequest(int sequenceId, int creditRequest, int
#region Tree Connect

/// <summary>
/// Describes a tree connect request.
/// </summary>
[Action]
static void TreeConnectRequest(int sequenceId, int creditRequest, [Dom
{
    Contracts.Requires(treeIds.Count - [illegible]
    CheckRequest(sequenceId, creditRequ[illegible]
    inflight.Add(new TreeConnectRequest(sequenceId, shareId));
    treeIdsInFlight++;
}

/// <summary>
/// Describes a tree connect response.
/// </summary>
[Action]
static void TreeConnectResponse(int sequenceId, [Domain("CreditDomain"
                                int treeId, ShareType shareType)
{
    TreeConnectRequest request =
        (TreeConnectRequest)CheckResponse(Command_Values.TREE_CONNECT,
    Capture(5, "tree connect request must be responded");
    Requires(shares.ContainsKey(request.shareId) && shares[request.sha
                    6, "only existing share should have successful
    Share share = shares[request.shareId];
```

Explore & Analyze

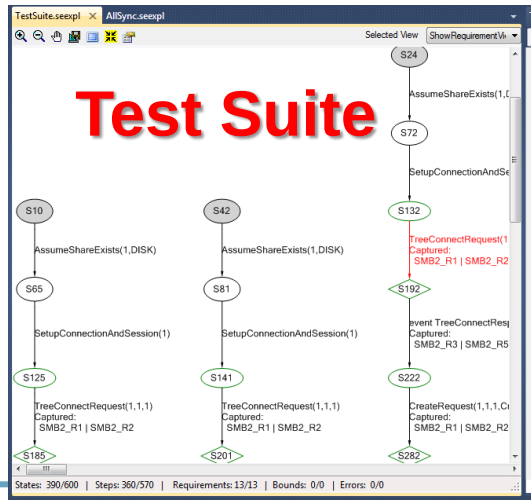
Remodel

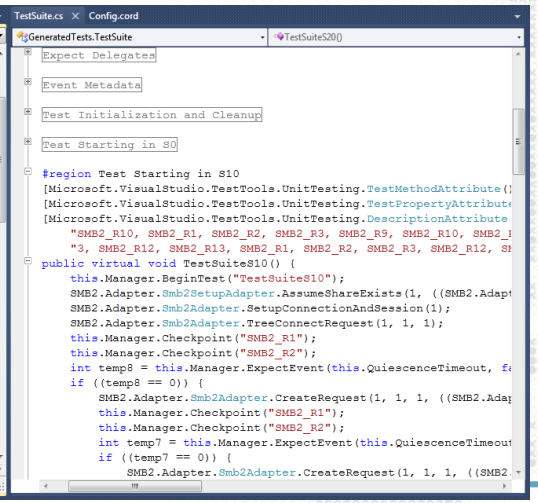
Generate

Model Graph

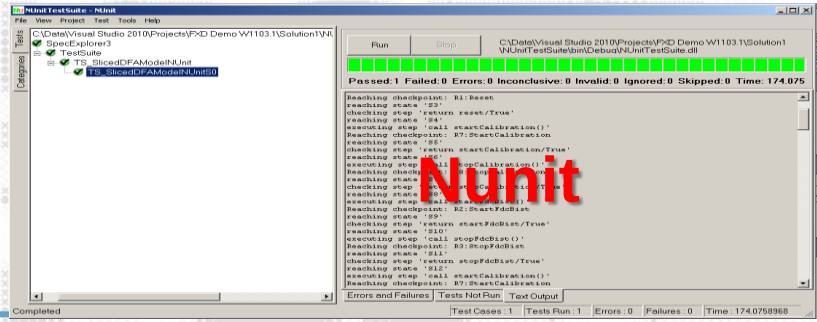


Test Suite





Execute



Nunit

http://www.nunit.com/



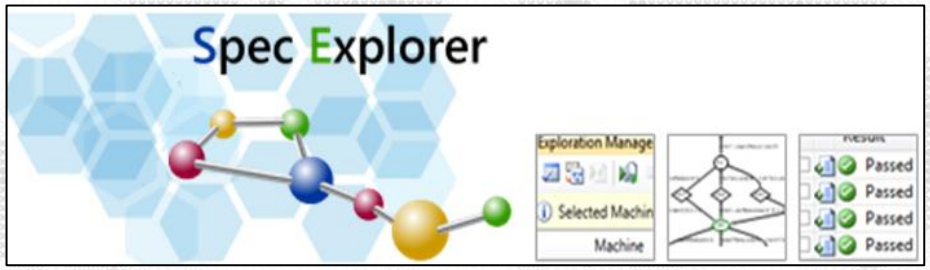
NSPYRE

making technology matter

The core idea behind Spec Explorer is:

- To **encode** a system's intended behavior (its specification) in machine-executable form (as a "model program").
- To **explore** the possible runs of the specification-program as a way to systematically generate test suites.
- To **compare** the behavior of the model program to the system's implementation in each of the scenarios discovered by algorithmic exploration

www.microsoft.com

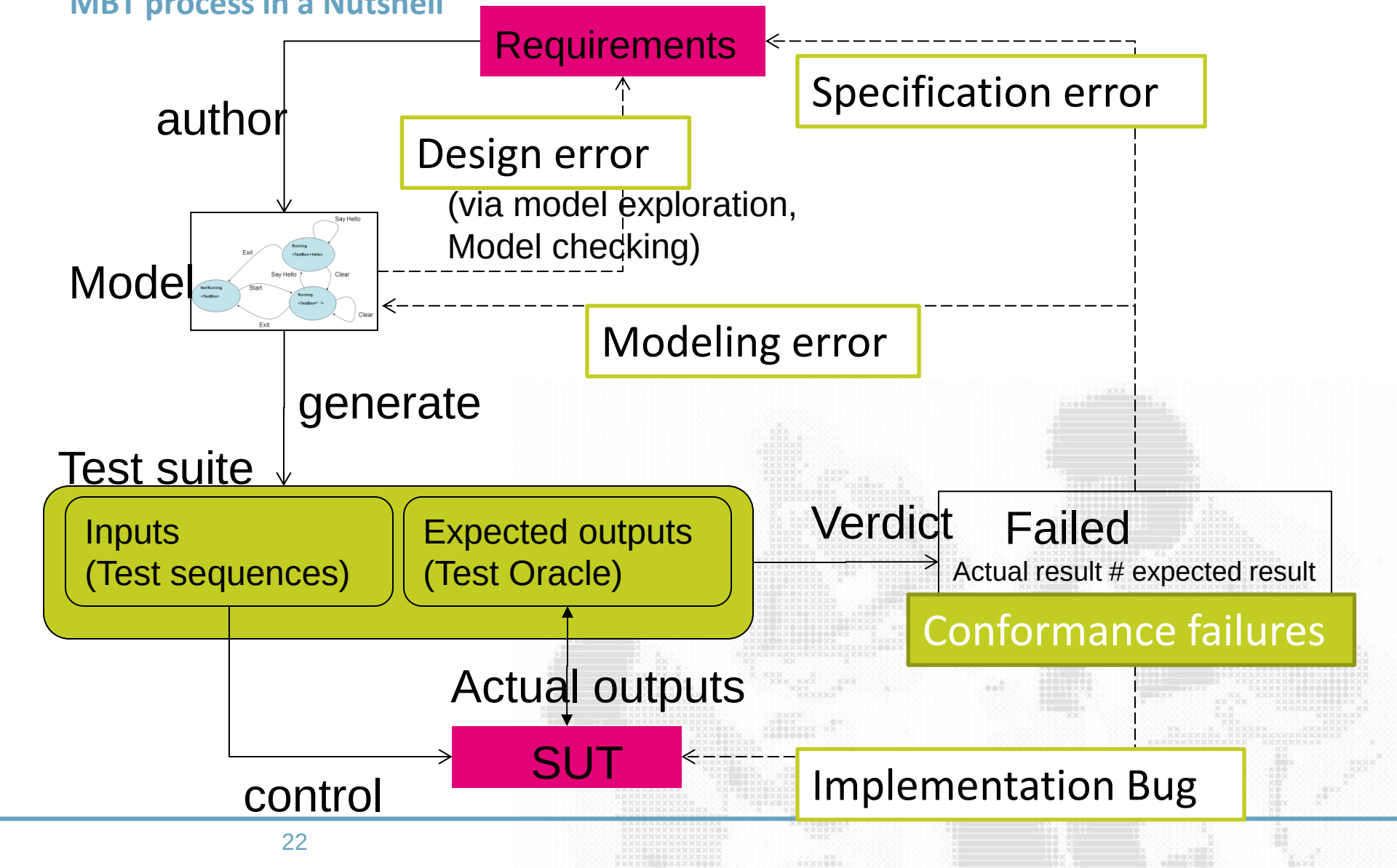




NSPYRE

making technology matter

MBT process in a Nutshell

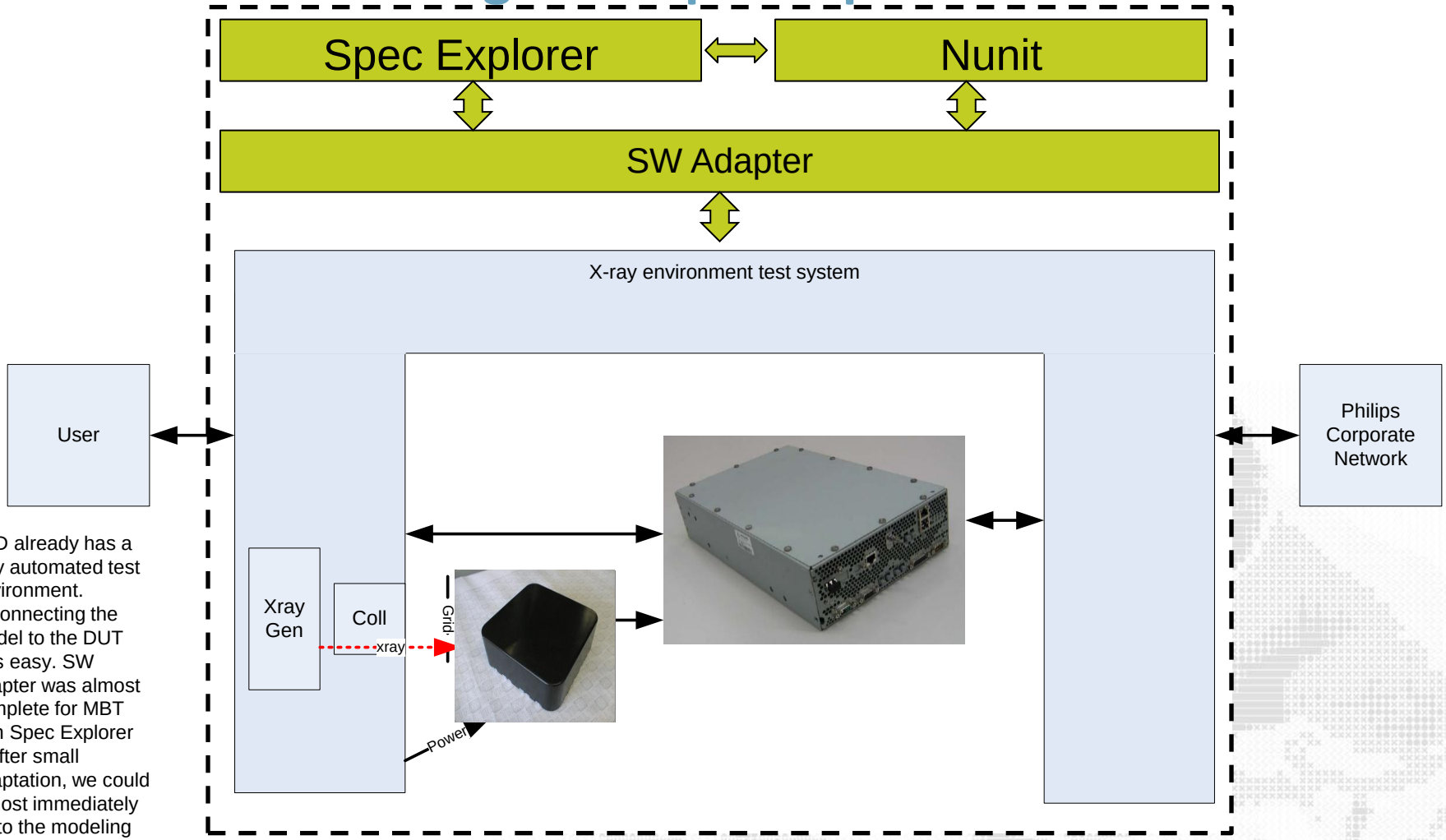




NSPYRE

making technology matter

Model Based Testing with Spec Explorer



FXD already has a fully automated test environment.
– Connecting the model to the DUT was easy. SW adapter was almost complete for MBT with Spec Explorer
– After small adaptation, we could almost immediately go to the modeling stage



NSPYRE

making technology matter

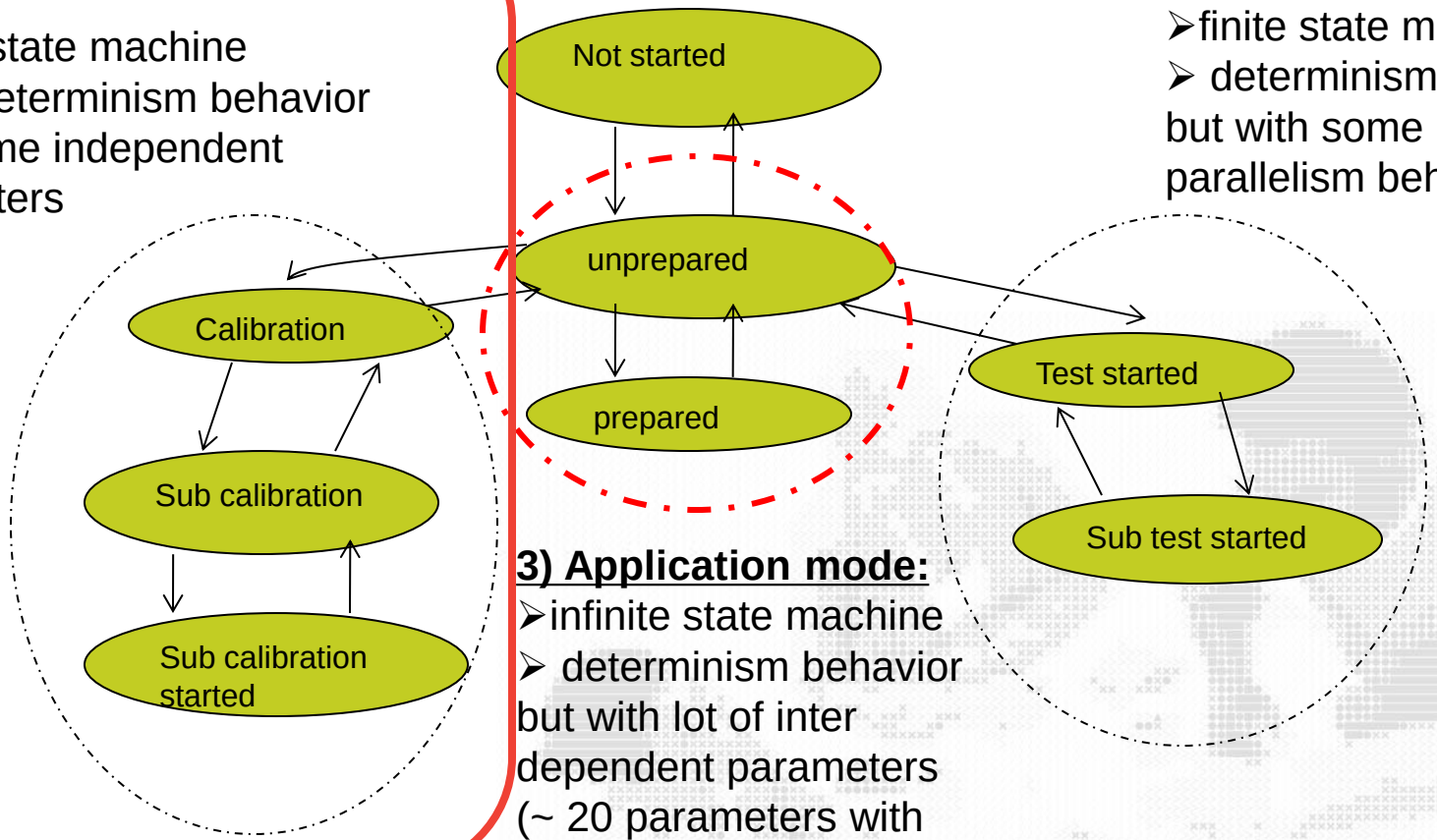
Modeling of our Device Under Test

– 3 main parts of the DUT state model:

1) Detector Calibration mode:

➢finite state machine

➢non-determinism behavior with some independent parameters



2) Field Service mode:

➢finite state machine

➢ determinism behavior but with some parallelism behavior

3) Application mode:

➢infinite state machine

➢ determinism behavior but with lot of inter dependent parameters (~ 20 parameters with different types and ranges)



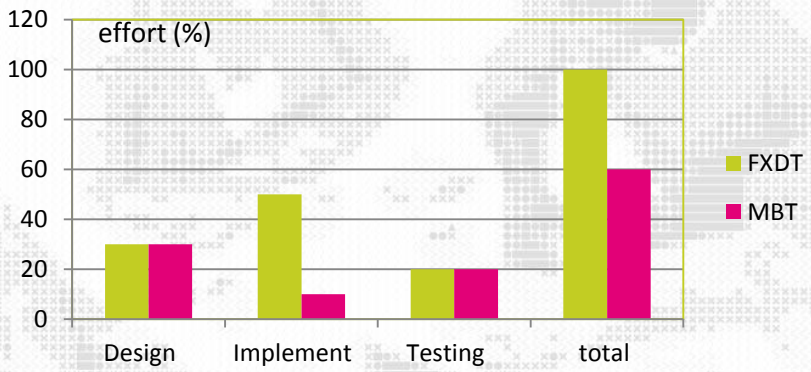
NSPYRE

making technology matter

First results:

- MBT is a technique that should apply more widely, Higher coverage, easier to maintain and suitable for (random) reliability testing.

- MBT very suited for state machine and interface compliance testing





NSPYRE
making technology matter

Modeling of our Device Under Test

– 3 main parts of the DUT state model:

1) Detector Calibration mode:

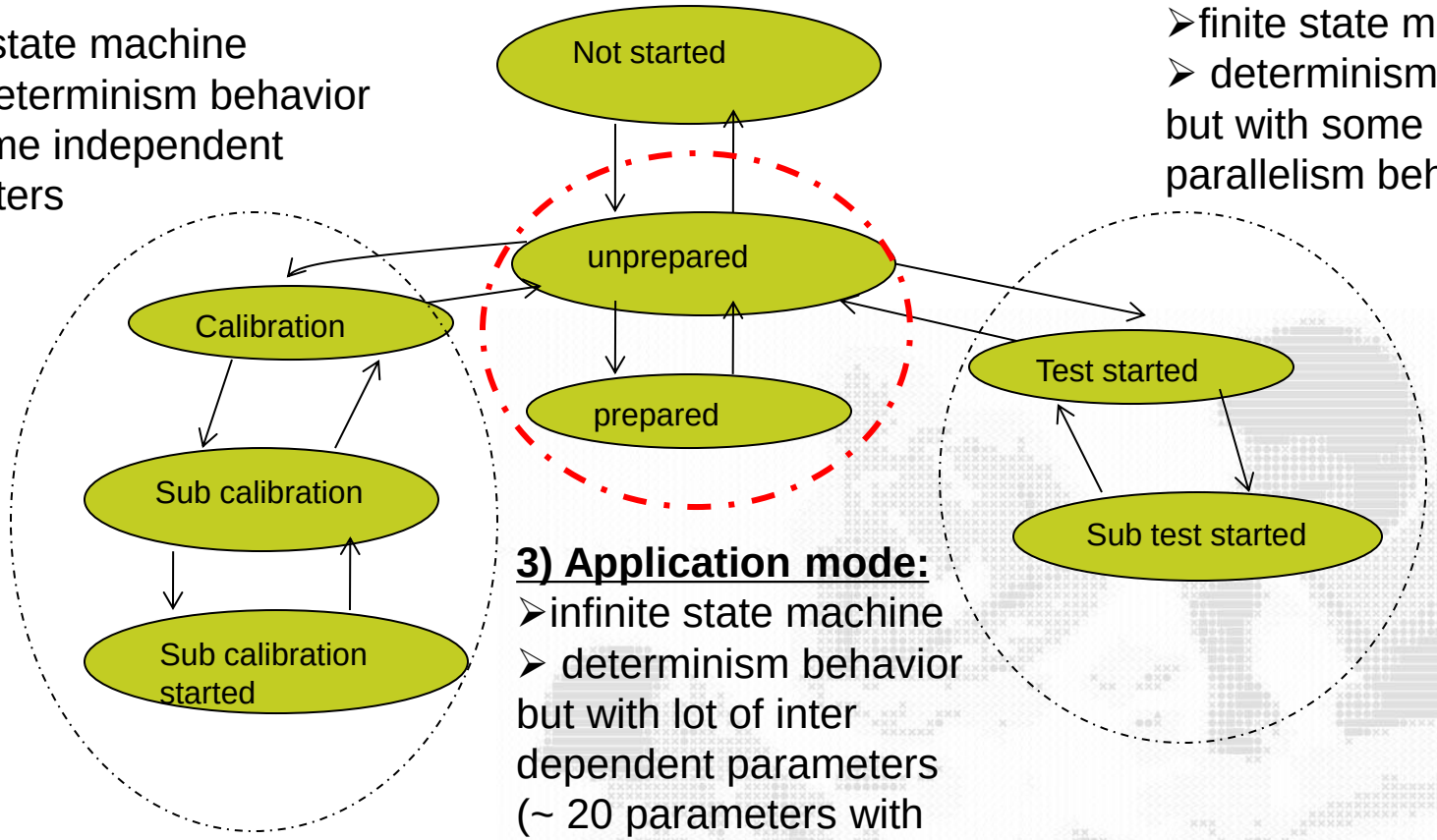
➢finite state machine

➢non-determinism behavior with some independent parameters

2) Field Service mode:

➢finite state machine

➢ determinism behavior but with some parallelism behavior



3) Application mode:

➢infinite state machine

➢ determinism behavior but with lot of inter dependent parameters (~ 20 parameters with different types and ranges)



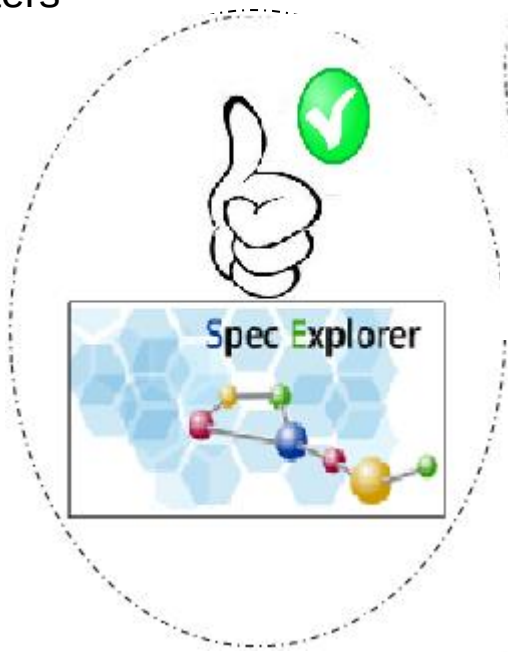
NSPYRE
making technology matter

Modeling of our Device Under Test

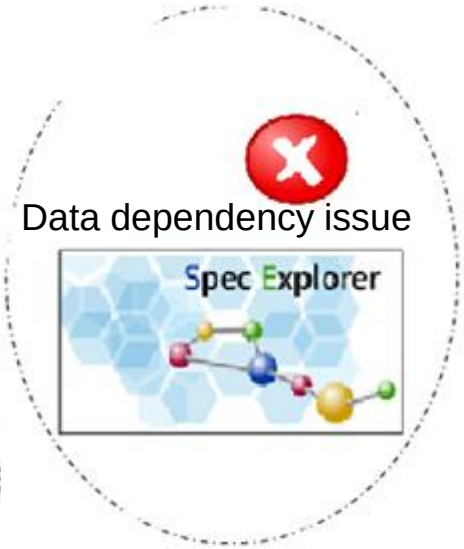
- 3 main parts of the DUT state model:

1) Detector Calibration mode:

- finite state machine
- non-determinism behavior with some independent parameters



Data dependency issue



2) Field Service mode:

- finite state machine
- determinism behavior but with some parallelism behavior

3) Application mode:

- infinite state machine
- determinism behavior but with lot of inter dependent parameters (~ 20 parameters with different types and ranges)



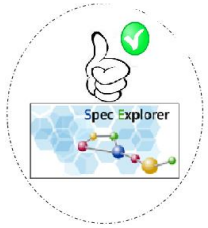
NSPYRE

making technology matter

Data dependency issue

- In case of limited number of **independent** parameters, SpecExplorer provides various built-in techniques to model and generate manageable and meaningful test cases with a limited effort and complexity.

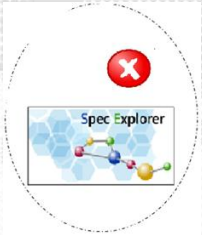
➔ Slicing, Scenario control, data combination (e.g. pair wise), data abstraction,….



- In case of high number of parameters that depend on each others, the built-in combination strategies of SpecExplorer do not provide any means for reflecting dependencies among parameters.

➔ Lot of **invalid test case** and little of **meaningful test cases**, unmanageable and complex test suite.

Parameter dependencies: a, b, c, =f(a, b, c, ..)





NSPYRE

To solve the data dependency issue Philips started a Research in cooperation with Technical University of Eindhoven under the leading of professor Mohammad Reza Mousavi and Vivek Vishal.

TU/e Technische Universiteit Eindhoven University of Technology

Where innovation starts

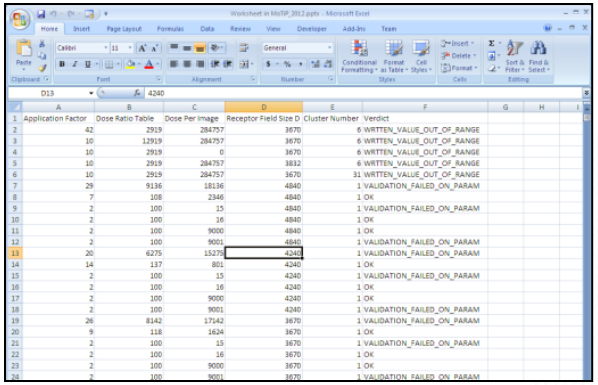


NSPYRE

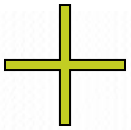
making technology matter

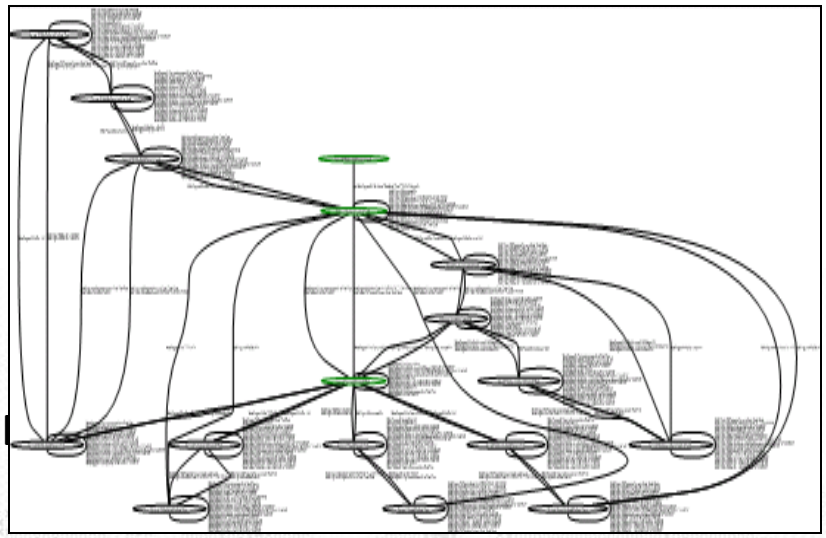
Approach to solve data dependency issue

- Compute data abstraction model independent from the behavioral model



Application Factor	Dose Ratio Table	Dose Per Image	Receptor Field Size D	Cluster Number	Verdict
42	2919	284757	3670	6	WRITTEN_VALUE_OUT_OF_RANGE
10	12919	284757	3670	6	WRITTEN_VALUE_OUT_OF_RANGE
10	2919	0	3670	6	WRITTEN_VALUE_OUT_OF_RANGE
10	2919	284757	3832	6	WRITTEN_VALUE_OUT_OF_RANGE
10	2919	284757	3670	31	WRITTEN_VALUE_OUT_OF_RANGE
29	9136	18136	4840	1	VALIDATION_FAILED_ON_PARAM
7	108	2346	4840	1	OK
2	100	15	4840	1	VALIDATION_FAILED_ON_PARAM
2	100	16	4840	1	OK
2	100	9000	4840	1	OK
2	100	9001	4840	1	VALIDATION_FAILED_ON_PARAM
20	6275	15275	4240	1	VALIDATION_FAILED_ON_PARAM
14	137	801	4240	1	OK
2	100	15	4240	1	VALIDATION_FAILED_ON_PARAM
2	100	16	4240	1	OK
2	100	9000	4240	1	OK
2	100	9001	4240	1	VALIDATION_FAILED_ON_PARAM
26	8142	17142	3670	1	VALIDATION_FAILED_ON_PARAM
9	118	1624	3670	1	OK
2	100	15	3670	1	VALIDATION_FAILED_ON_PARAM
2	100	16	3670	1	OK
2	100	9000	3670	1	OK
2	100	9001	3670	1	VALIDATION_FAILED_ON_PARAM





- Data model

Behavior model

- We used a **Constraint Solver** to generate the optimized data model.

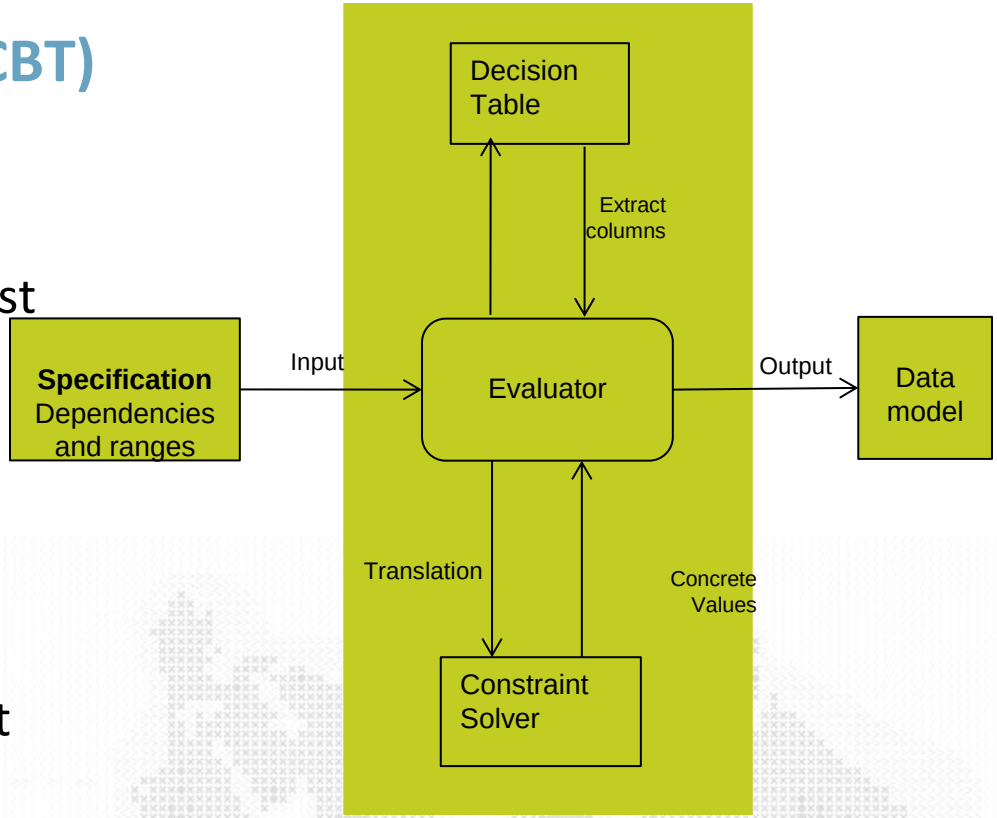


NSPYRE

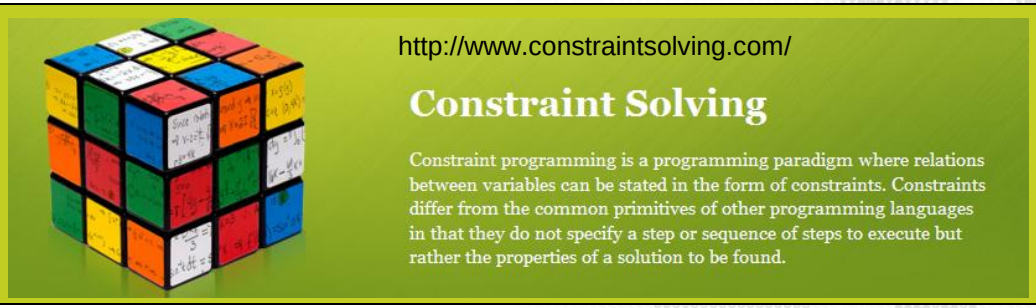
making technology matter

Constraint Based Testing Tool (CBT)

- Our developed CBT tool uses **decision table** and **constraint solver** to generate meaningful test data reflecting dependencies among the input parameters



- The **evaluator** performs the translation between the different components of the CBT.

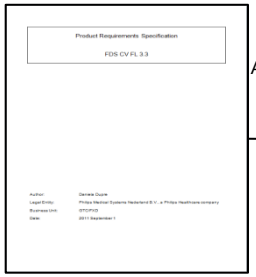




NSPYRE

making technology matter

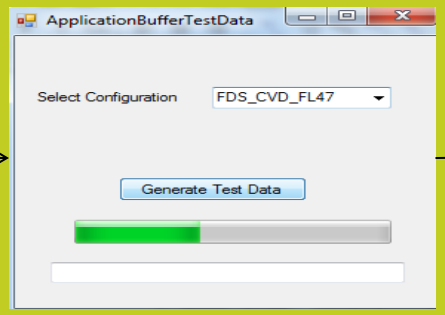
Work Flow



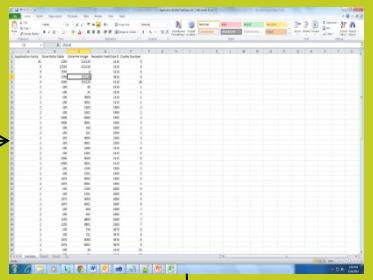
Automatic

Specification
(Dependencies and ranges)

MS VS 2010 SDE

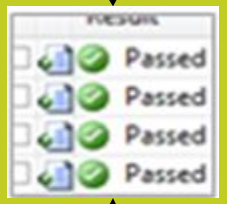


CBT



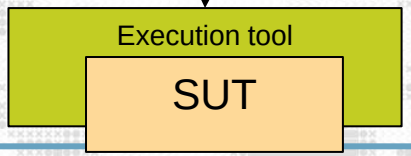
Data model

Behavior model



Test Cases

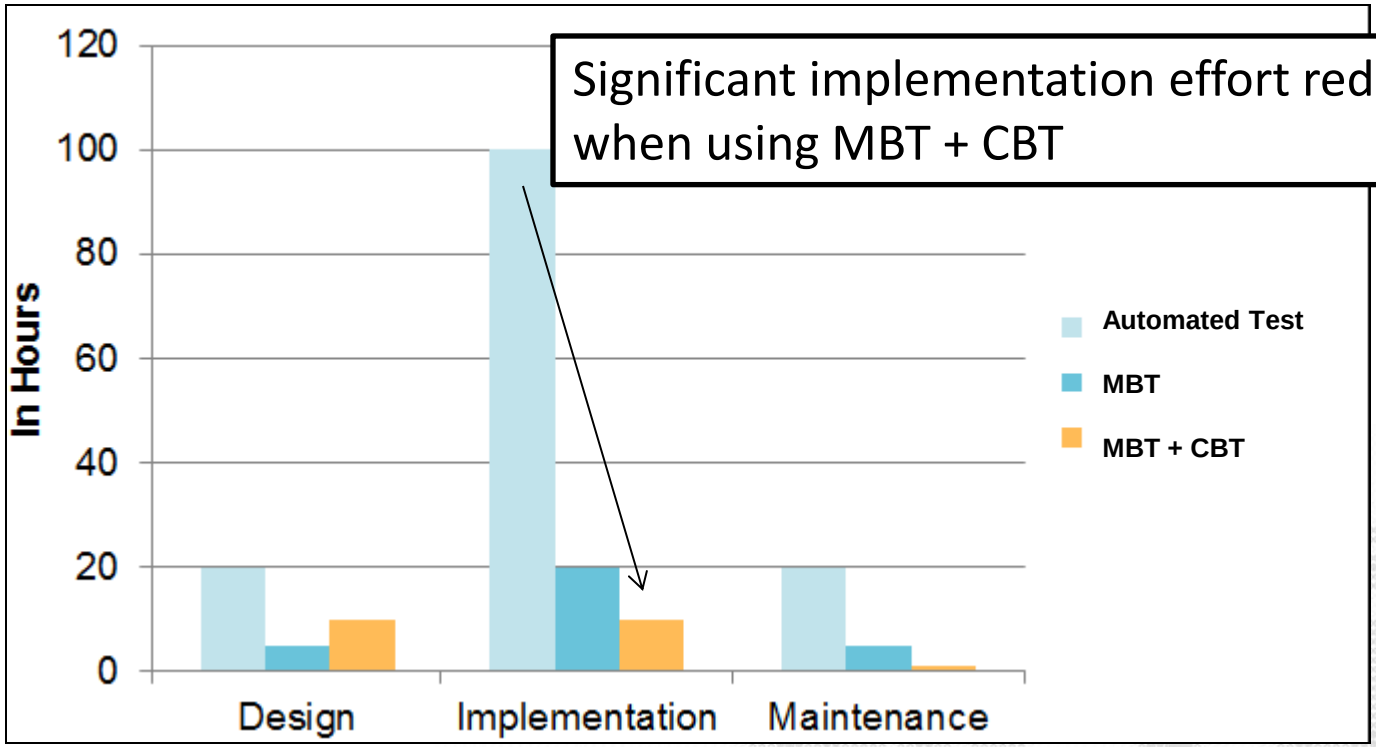
SUT + execution tool





NSPYRE
making technology matter

Results : Effort Involved



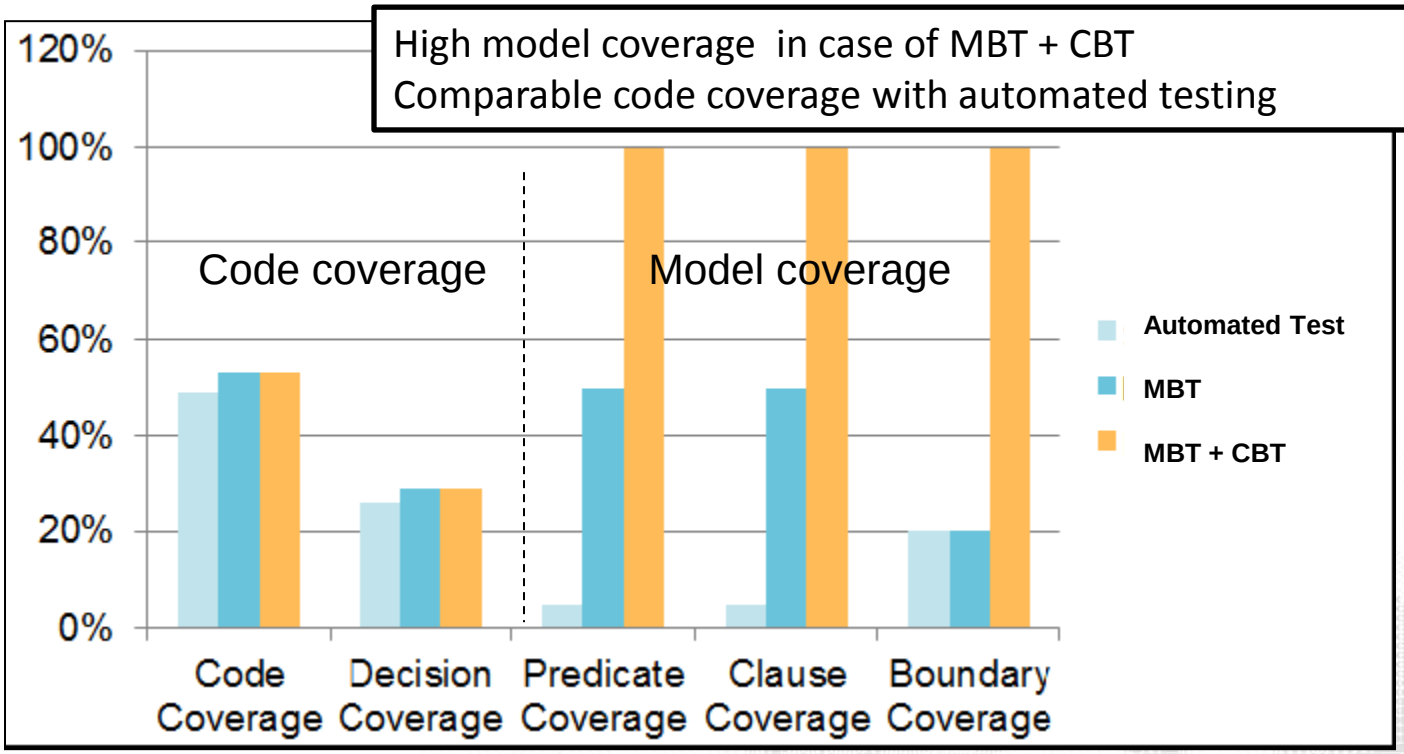
Design: creation of logical test case specification

Implementation: creation of physical test cases (in case of the semi-automatic method), and behavioral models (in case of MBT and MBT+CBT).



NSPYRE
making technology matter

Results : Coverage Achieved



Predicate, Clause and boundary coverage are a model coverage metric
Predicate coverage = %of the boolean expressions (in the data model) evaluated both to true and false.
Clause coverage = % of atomic boolean expressions (not containing any logical operators) that are evaluated both to true and false.
Boundary coverage = % of the boundaries that have been tested.
Code and decision coverage are measured code coverage analyzer tool Bullseye (http://www.bullseye.com/).



NSPYRE
making technology matter

Results : Other Metrics

Metrices	Automated Test	MBT	MBT+CBT
Testing Technique	BVA + ECT	PairWise	BVA+ DT+ Constraint solver
Test cases generated	~100	~13000 Lot of invalid test case and little of meaningful test cases	~1000 Manageable number of test cases (especially valid and meaningful test cases)
Test execution time	1 minute	43 minutes	7 minutes
Perceived effectiveness	Low	Medium	High
Additional bugs found	-	2	2

The effectiveness of MBT + CBT is considered high as a reasonable number of “smart” test cases are generated with very low effort.



NSPYRE

making technology matter

Summary

- We have developed a simple but effective tool to solve data dependency problem and increase the effectiveness and efficiency of our test process.

- MBT has obvious advantages over traditional test automation. It helps to increase the effectiveness and efficiency of test process.

- Spec Explorer shows some shortcoming in case of SUT with lot of parameters that depend on the values of each others. We solved this weakness by applying constraint based testing in combination with Spec Explorer.

- MS Spec Explorer team is interested in our work and will probably integrate this feature in next release.

- This work is published at the ISREE 2012 in Dallas

IEEE
Advancing Technology for Humanity



NSPYRE

making technology matter

Published at the Motip Workshop in Dallas USA as part of the ISSRE 2012
27 Nov 2012
Integrating Model-Based and Constraint-Based Testing Using SpecExplorer
Vivek Vishal, Mehmet Kovacioglu, Rachid Kherazi and Mohammadreza Mousavi

http://2012.issre.net/

http://2012.issre.net/content/4th-workshop-model-based-testing-practice-motip

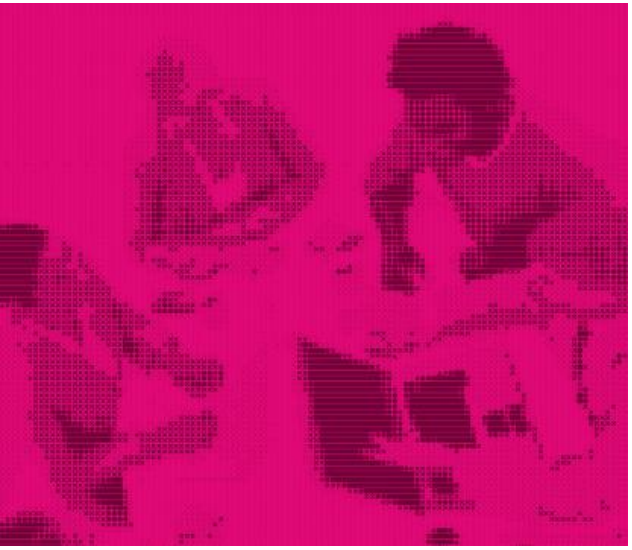
IEEE Advancing Technology for Humanity

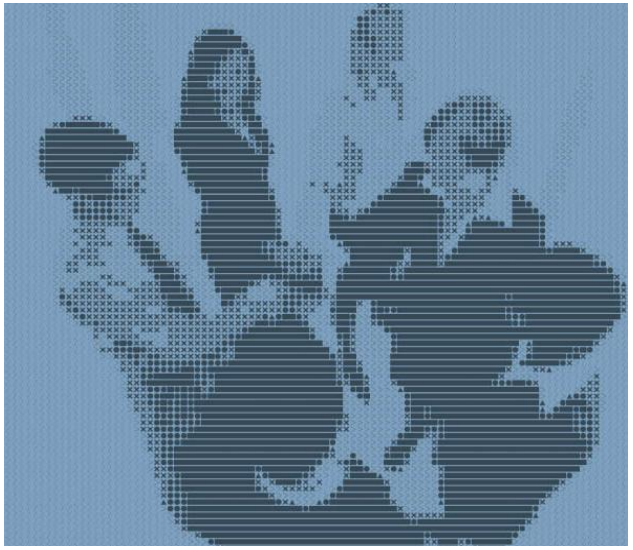
MoTiP

ReliabilitySociety

NSPYRE

making technology matter





Questions?

For more info contact Rachid.kherrazi@Nspyre.nl

www.nspyre.nl



NSPYRE

making technology matter

Problems of MBT

- Process shift
  - Up front investment in test
- Personnel shift
  - Higher education and sophistication