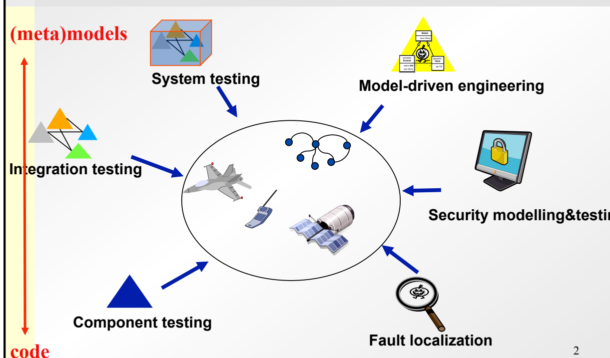
Design by contract for software vigilance and Diagnosability

Yves Le Traon
Professor, University of Luxembourg

team

securityandtrust.lu

Research domains



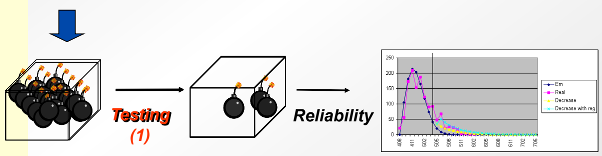
2

Software testing: cost and trust

« in God we trust, for the rest we test » (A. Petrenko)

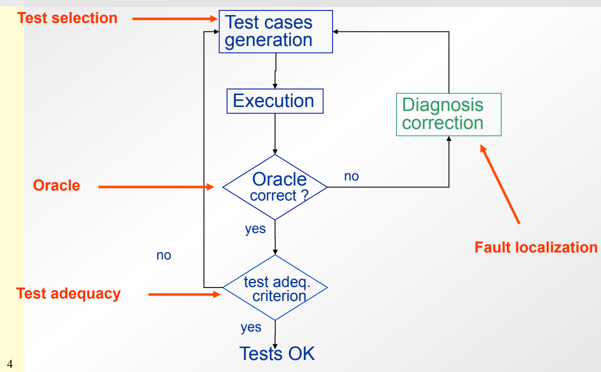
Testing → *Detecting inconsistencies between implementation and specification*

Design for testability
2) Design for trust



3

Testing issues



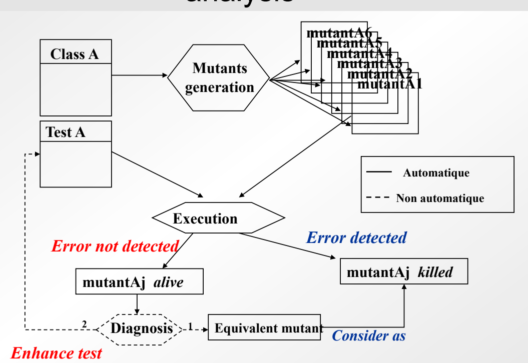
4

Trust the software => trust the tests

- Trust is higher if tests are good
- How « testing » tests ?
- A test is « good » if it has a high fault revealing power
- If tests are not able to detect faults we voluntarilly injected… be cautious

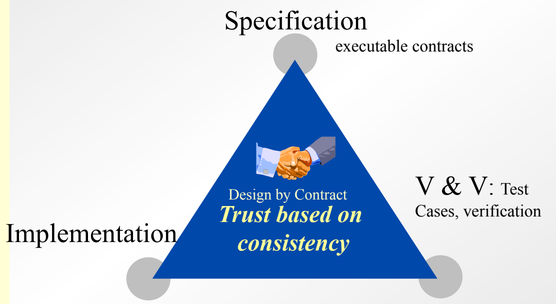
5

A test qualification technique: Mutation analysis



page 6

Design for trust



Reuse trustable components

7

Two examples of my research

Design by Contract

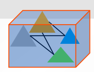
Contracts and reliability

contracts as embedded oracle functions

Y. Le Traon, B. Baudry and J-M. Jézéquel
"Design by Contract to improve Software Vigilance",
IEEE Transactions on Software Engineering, August 2006

Contracts as oracles

Evaluation of a design methodology



System testing

System testing

C. Nebut, F. Fleurey, J-M. Jézéquel and **Y. Le Traon**, "Automatic Test Generation: A Use Case-Driven Approach",
IEEE Transactions on Software Engineering, March 2006

S. Pickin, C. Jard, T. Jéron, J-M. Jézéquel, and **Y. Le Traon**.
"Test synthesis from UML models of distributed software",
IEEE Transactions on Software Engineering, April 2007

Contracts for test generation

8

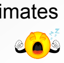
Overview

1. Contracts as embedded oracles
   - Vigilance
   - Diagnosability
2. Contracts for test generation
3. Conclusion about Design by Contract

9

Contracts as embedded oracles

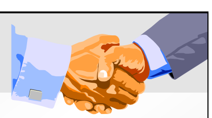
The objectives

- Quantitative estimate of what contracts really improve in the software
- We propose two estimates
  - Vigilance 
  - Diagnosability 
- Obtain general trends

«*Things must be as simple as possible, but no simpler*». A. Einstein

11

Design-by-contract™



- A design philosophy (B. Meyer)
- Component-based OO approach
- A component
  - is not responsible from its inputs consistency (caller responsibility)
    - may refuse to work if caller breaks the contract
  - is responsible from its result
- Specification is derived into executable contracts

12

Design-by-contract™: analysis

Contracts can detect faults: help to fault localization

Contracts may not detect all faults: contracts quality

BankAccount
{balance ≥ overdraft}

balance: Sum
overdraft: Sum

deposit (amount: Sum)
{pre: amount > 0}
{post: balance = balance@pre}+ amount}

balance = balance + amount

withdraw (amount: Sum)
{pre: amount > 0}
{post: balance = balance@pre - amount}

13

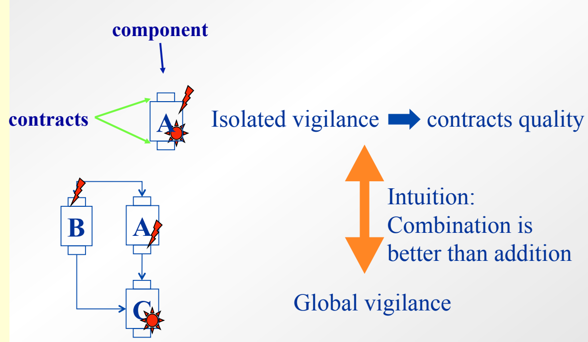
Overview

1. Contracts as embedded oracles
   - Vigilance
   - Diagnosability
2. Contracts for test generation
3. Conclusion about Design by Contract

14

Vigilance

Informally “the quality or state of being wakeful and alert”



15

Vigilance

- ***Vigilance (V):*** The vigilance expresses the probability that the system contracts dynamically detect faulty states that would have otherwise provoked a failure.
- Weakness is the contrary = 1 - V

16

Axiomatization: intuitive properties

Shepperd M. and Ince D. *Derivation and Validation of Software Metrics*. Oxford University Press N.Y., 1993.

Examples:

GVA2 - System concatenation. *The global vigilance of a system obtained by concatenation of two systems S1 and S2 cannot be lower than the lowest vigilance of S1 and S2*

GVP3 - Contract addition. For any system, its global vigilance cannot decrease by addition of a contract.

➔ Useful for formal validation

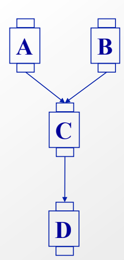
17

Vigilance: Test dependency

Y. Le Traon, T. Jéron, J-M. Jézéquel and P. Morel, **"Efficient OO Integration and Regression Testing "**, ***IEEE Transactions on Reliability***, March 2000

A component plugged into a system has a vigilance enhanced by its clients contracts

$\Rightarrow$ *test dependency* R_{TD}



Definition. Det(Ci, Cj)*:* *probability that Ci's contracts detect an error in Cj*

18

Global vigilance

- Let *Prob_error(i, S)* be the probability the failure in S comes from the component Ci

$$V(S) = 1 - Weak(S) = 1 - \sum_{i=1}^{n} Prob\_error(i,S) \cdot LocWeak(C_i,S)$$

$$LocWeak(C_i,S) = \underbrace{Weak_i}_{\text{prob. component i does not detect the faulty state}}$$

19

Formal and Empirical validation

Shepperd M. and Ince D. *Derivation and Validation of Software Metrics*. Oxford University Press N.Y., 1993.

GVP3 - Contract addition. For any system, its global vigilance cannot decrease by addition of a contract.

- Axiomatization
- Mathematical modelling
- Theoretical validation of the model
- Vali
- Res

GVP3 - Contract addition. For any system, its global vigilance cannot decrease by addition of a contract.

$$V(S) = 1 - Weak(S) = 1 - \sum_{i=1}^{n} Prob\_err(C_i,S) \cdot LocWeak(C_i,S)$$

$$LocWeak(C_i,S) = Weak_i \prod_k (1 - Det(C_k,C_i)), k \ni C_k R_D C_i$$

	Minimum	Maximum	average
% mutants killed (initial contracts)	17%	83%	58,5%
% mutants killed after contracts improvement	72%	100%	87,5%

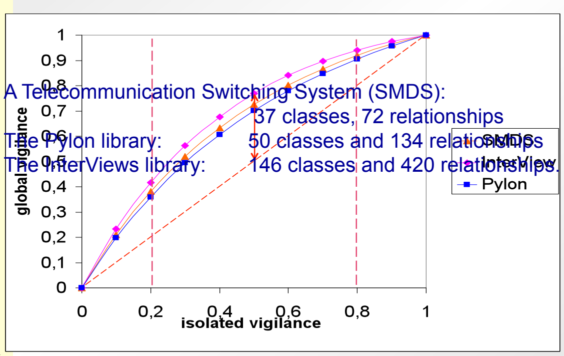
Isolated Vigilance

	Min.	Max.	Average
% mutants of provider killed by client's contracts	50%	84%	69%

Det(Ci,Cj)

20

Vigilance: Empirical results and interpretation



A Telecommunication Switching System (SMDS): 37 classes, 72 relationships

The Pylon library: 50 classes and 134 relationships

The InterViews library: 146 classes and 420 relationships

21

Vigilance: Conclusion

- About the results:
  - no contracts ⇒ system not vigilant
  - vigilance improves rapidly with contracts quality
  - very high vigilance is very expensive: almost 40% more contracts to improve from 80% to 100% vigilance

22

Overview

1. Contracts as embedded oracles
   - Vigilance
   - Diagnosability
2. Contracts for test generation
3. Conclusion about Design by Contract

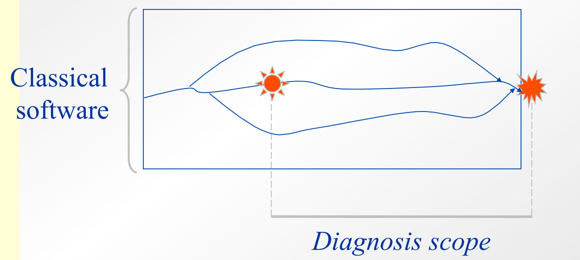
23

Diagnosability

- Diagnosability expresses the effort for the localization of a fault.

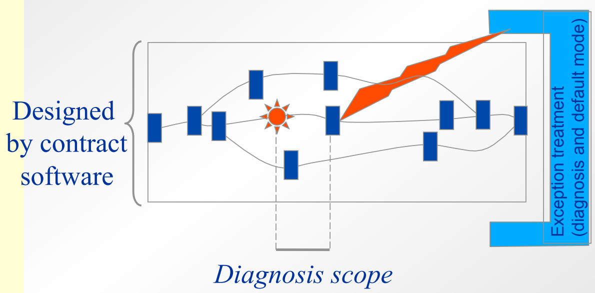
24

Diagnosability: the help of contracts



25

Diagnosability: the help of contracts

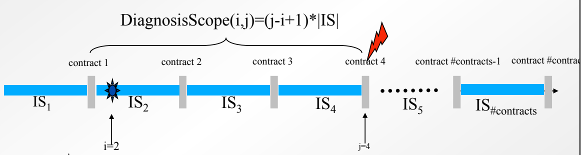


26

Diagnosability

$$\delta = \sum_{i=1}^{\#contracts} (Pfaulty_i \times \delta_i)$$

DiagnosisScope(i,j)=(j-i+1)*|IS|

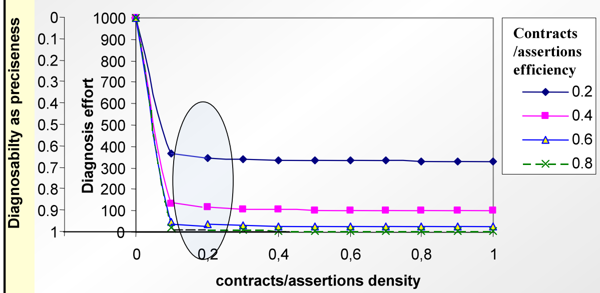


Det_i^j is the probability that contract j detects a faulty statement in ISi

$$Det_i^j = p_j \cdot \prod_{k=1}^{j-1} (1 - p_k)$$

27

Diagnosability: trends



28

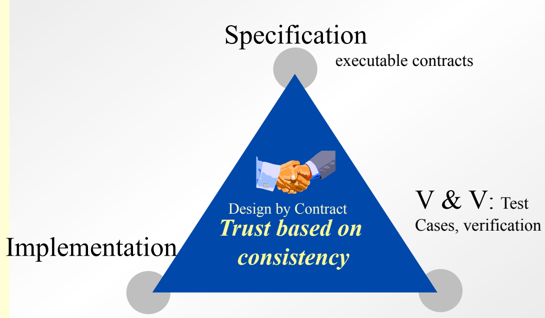
Conclusion Vigilance & Diagnosability

- Measures estimate the contribution of contract quality and density
- The quality of contracts is more important than their quantity
- Related work
  - Automated fault localization (ICSE 2006)

B. Baudry, F.Fleurey and Y. Le Traon, "Improving Test Cases for Accurate Diagnosis", *in proceedings of the 28th Int. Conference on Software Engineering (ICSE 2006), Shanghai, May 2006.*

29

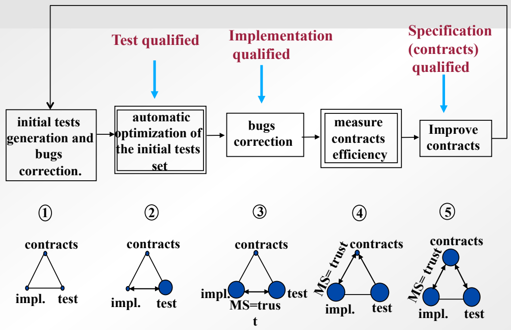
Design for trust



Reuse trustable components

30

Design for trust loop



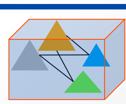
page 31

Overview

1. Contracts as embedded oracles
   - Vigilance
   - Diagnosability
2. Contracts for test generation
3. Conclusion about Design by Contract

32

Contracts for test generation



Problem analysis: Model-Based System Testing (for product lines)

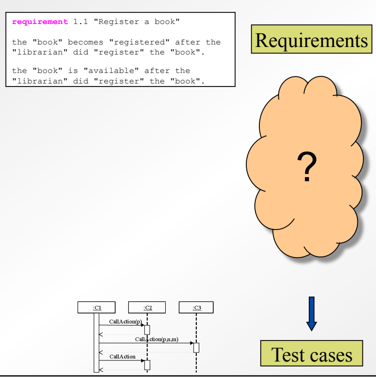
- System requirements ...
  - … evolve very often
    - Nokia : 69% of the requirements modified, 22% modified twice

    → need to build quickly new tests from the new requirements
  - … are in natural language

    → need of a formalization to apply automatic test generation techniques

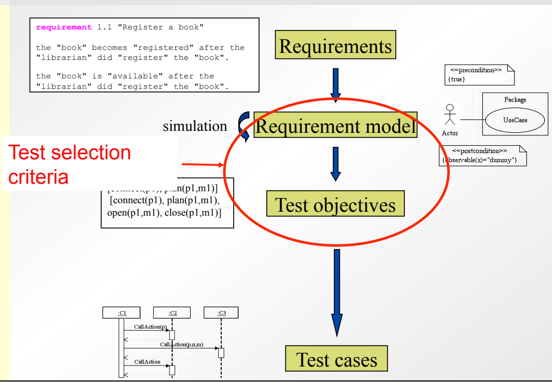
34

The problem: a gap to bridge



35

The problem: a gap to bridge



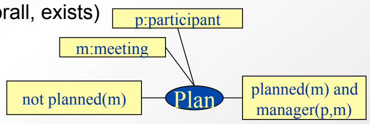
36

A use case contract language

Requirements models

- First order logic expressions
  - Boolean properties (predicates) = name+typed parameters
    - Ex: planned(m:meeting)

      manager(u:participant,m:meeting)
  - Enumerated properties
  - Classical boolean operators (and, or, implies, not)
  - Quantifiers (forall, exists)



- Benefits:
  - formalization of the use cases
  - dependencies between the use cases can be deduced

37

A use case contract language : Deducing dependencies

#use case OPEN

UC open(u:participant;m:mtg)

pre created(m) and moderator(u,m) and not closed(m) and not opened(m) and connected(u)

post opened(m)

#use case CLOSE

UC close(u:participant; m:mtg)

pre opened(m) and moderator(u,m)

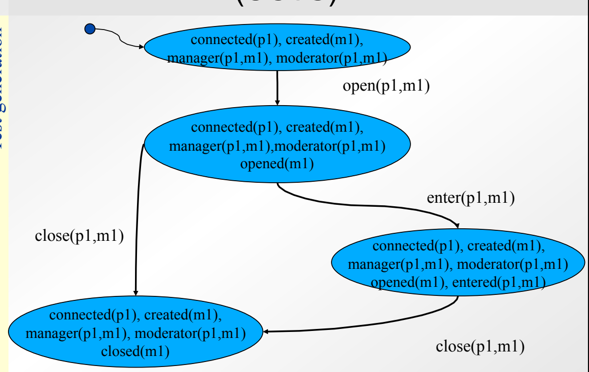
post ...

OPEN(u1,m1);CLOSE(u1,m1) is a correct sequence

38

The Use Case Transition System (UCTS)

Test generation



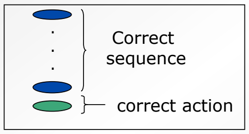
39

OPEN(p1,m1);ENTER(p1,m1); CLOSE(p1,m1) is a correct sequence

Test selection criteria

Nominal behaviors

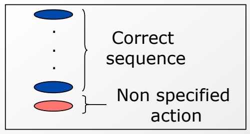
Generate sequences leading to all licit application of the use case



"all configurations making its **precondition true**"

Robustness behaviors

Generate sequences leading to an invalid application of the use case



"all configurations making its **precondition false**"

40

Hypothesis

- H1: Test cases produced from requirements are « efficient » to test the overall system.
  - Adequacy criterion from the industry : code coverage
  - Comparing test criteria
- H2: Most real-industrial requirements can be treated with such an approach

  % operational requirements which can be covered by the approach

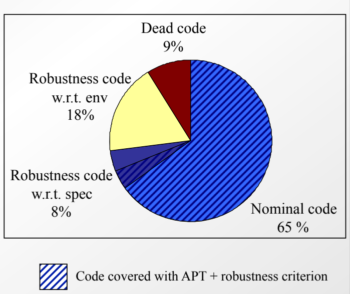
41

H1: Academic experiments

- 3 case studies
  - FTP client
  - ATM
  - Virtual meeting
- Code repartition
- Code Coverage

Code coverage

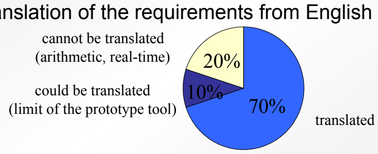
for the virtual meeting example



42

H2: Experiments with TAS and France Telecom

- TAS: Two components of weapon navigation system (Mirage 2000-9 and Rafale).
- France telecom: Three services on the livebox 2 modem
- Translation of the requirements from English to RDL



43

New issues

- Initiate many researches:
  - MDE for Requirements
  - Product lines testing and verification
- Commercial tools are now available based on similar principles
- Empirical validation of Model-based testing
  - ALCATEL: testing new distributed telecom services
  - French Defense Department: testing cryptographic components

44

Overview

1. Contracts as embedded oracles
   - Vigilance
   - Diagnosability
2. Contracts for test generation
3. Conclusion about Design by Contract

45

Conclusion

- Design by Contract 
  - an instrument to build trust in a system
  - Declarative approach
  - Lightweight
- Can be used for
  - Fault localization
  - Test generation
  - Security
  - Vigilance → adaptive resilient systems

46

Creation of the main conference on testing, verification and validation

- Gang of Four : L. Briand, J. Offutt, B. Baudry, Y. Le Traon
- 1st edition : 303 abstracts, 224 full papers
- 250 participants
- 8 associated workshops
- Selection rate: 20-25 %

« intelligently react to abnormal situations and ensure the quality of the information » (P1 conclusion)

Questions ?

48

Questions

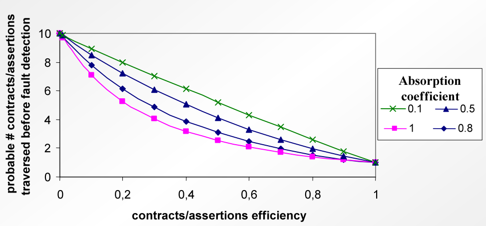
Questions

49

…

50

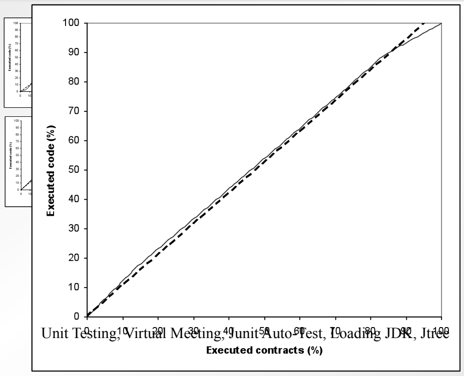
Threats to validity: sensitivity analysis



never contradict the axioms

51

Threats to validity: contracts repartition



Unit Testing, Virtual Meeting, Junit Auto-Test, Loading JDK, Jtree

52

Threats to validity: contracts repartition

Table 1 contracts distribution

	#statements	#contracts	#stmts / #contracts
List Class	1391	505	2,75
Virtual Meeting Server	2291	1171	1,96
Junit Auto-Test	19419	10801	1,8
Loading JDK	111730	40751	2,74
Jtree	1970745	885001	2,22

53

Proof for GV3

GVP3 - Contract addition. For any system, its global vigilance cannot decrease by addition of a contract.

- Consider that we add a contract to one component, for instance, C_1

$\mathrm{Det}(C'1, C'1) \geq \mathrm{Det}(C1, C1)$ (1)

$\forall\, k \,/\, C_1\, R_{TD}\, C_k,\ \mathrm{Det}(C'_1, C_k) \geq \mathrm{Det}(C_1, C_k)$

==> $\prod_k (1 - Det(C'_1, C_k)) \leq \prod_k (1 - Det(C_1, C_k))$

==> $\forall\, k \,/\, C_1\, R_{TD}\, C_k\ \mathrm{LocWeak}(C_k, S') \leq \mathrm{LocWeak}(C_k, S)$ (2)

Consider that C_1 has q-1 servers, we denote [C2, C3 … Cq]

From (2) $\forall\, k \in [2 .. q],\ \mathrm{LocWeak}(C_k, S') \leq \mathrm{LocWeak}(C_k, S)$

$$V(S') = 1 - [\sum_{i=(q+1)}^{|S'|} (Prob\_Err(C'_i, S') \times LocWeak(C'_i, S')) + \sum_{i=1}^{q} (Prob\_Err(C'_i, S') \times LocWeak(C'_i, S'))]$$

$$V(S') \geq 1 - [\sum_{i=q+1}^{|S|} (Prob\_Err(C_i, S) \times LocWeak(C_i, S)) + \sum_{i=1}^{q} (Prob\_Err(C_i, S) \times LocWeak(C_i, S))]$$

==> $V(S') \geq V(S)$

54

Diagnosability: Measures

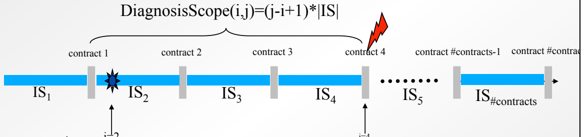
- Assumptions:
  - the contracts repartition in a flow is uniform
    - Each IS has same size ISsize (=#stat div #contracts),
  - the closer a contract is to the faulty statement i the more probable it can detect the fault
  - the contracts have an equal probability p to detect a fault coming from the statements they are directly consecutive to
  - each statement has the same probability to be faulty equal to 1/Nstat

Diagnosability

$$\delta_i = \sum_{j=1}^{n\_contract} \mathit{DiagnosisScope}(i,j) \times \mathrm{Det}_i^j + \mathrm{Pfailure} \times \mathit{DiagnosisScope}(i, n\_contracts)$$

$$\delta = \sum_{i=1}^{\#contracts} (\mathrm{Pfaulty}_i \times \delta_i)$$

DiagnosisScope(i,j)=(j-i+1)*|IS|



Det_i^j is the probability that contract j detects a faulty statement in ISi

$$\mathrm{Det}_i^j = \mathrm{p}_j \cdot \prod_{k=1}^{j-1} (1 - \mathrm{p}_k)$$

$$\mathrm{Det}_i^j = \alpha^j p \cdot \prod_{k=i}^{j-1} (1 - \alpha^{k-i} p)$$

Absorption coefficient α: $p_j = \alpha^j.p$

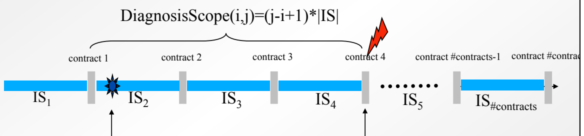
$$\delta_i = \mathrm{ISsize}.[\sum_{j=i}^{\#contracts} (j-i+1).\mathrm{Det}_i^j + (1 - \sum_{j=i}^{\#contracts} \mathrm{Det}_i^j) \times (\#contracts - i + 1)]$$

Diagnosability: simplified

$$\delta_i = \sum_{j=1}^{n\_contract} \mathit{DiagnosisScope}(i,j) \times \mathrm{Det}_i^j + \mathrm{Pfailure} \times \mathit{DiagnosisScope}(i, n\_contracts)$$

$$\delta = \sum_{i=1}^{\#contracts} (\mathrm{Pfaulty}_i \times \delta_i)$$

DiagnosisScope(i,j)=(j-i+1)*|IS|



Det_i^j is the probability that contract j detects a faulty statement in ISi

$$\mathrm{Det}_i^j = \mathrm{p}_j \cdot \prod_{k=1}^{j-1} (1 - \mathrm{p}_k)$$

Cursus and diploma

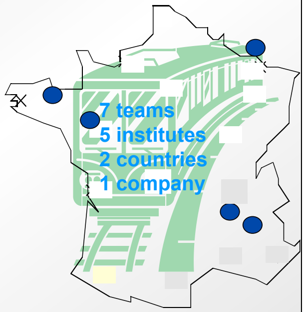
1994. Master in computer science (DEA) from INPG, Grenoble.
1994. Engineering Degree - ENSIMAG.

1994-1997. PhD at INPG- Grenoble
1997-1998. PostDoc at LCIS-INPG lab (Valence)

1998-2004. Assistant professor, Univ. of Rennes 1 - IRISA lab.
2004. Authorization for the management of research (HDR).

2004-2006. France Télécom R&D

Nov. 2006. Associate professor – Telecom Bretagne
Sept. 2008. Professor
Head of the SERVAL team
Julyt. 2009. Professor
Univ. Luxembourg



58

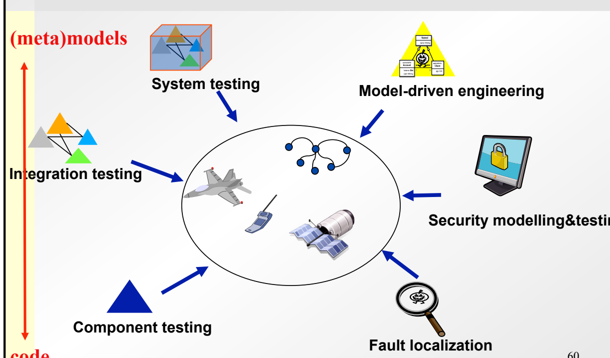
Some results and ongoing work

- Unit component testing
  - Self-testable components (IEEE Software 01)
  - Evolutionary algorithms (IEEE Software 05, STVR 05)
- Integration testing (best paper ISSRE 09, IEEE Trans. on Reliability, ECOOP 01)
- System Testing (IEEE TSE 06, IEEE TSE 07)
- Testability Analysis
  - Refactoring of UML models (Best paper UML 01)
  - Measurements (Information Software and Technology 05, IEEE TSE. 06)
- Security
  - Modeling (ICST 08, best paper Models 08)
  - Testing (ISSRE 07, ISSRE 08, ICST 08, ICST 09)
- Communication and networks
  - P2P system testing (Best paper ISSRE 08)
- MDE and Barriers to Systematic Model Transformation Testing (SoSym journ 07, Communications of the ACM 2010)

… Aspect Oriented Programming and testing
… Requirements and Model-driven engineering
… ad-hoc network testing
… security contracts

59

Research domains



60

Some results and ongoing work

- Unit component testing
  - Self-testable components (IEEE Software 01)
  - Evolutionary algorithms (IEEE Software 05, STVR 05)
- Integration testing (best paper ISSRE 09, IEEE Trans. on Reliability, ECOOP 01)
- System Testing (IEEE TSE 06, IEEE TSE 07)
- Testability Analysis
  - Refactoring of UML models (Best paper UML 01)
  - Measurements (Information Software and Technology 05, IEEE TSE. 06)
- Security
  - Modeling (ICST 08, best paper Models 08)
  - Testing (ISSRE 07, ISSRE 08, ICST 08, ICST 09)
- Communication and networks
  - P2P system testing (Best paper ISSRE 08)
- MDE and Barriers to Systematic Model Transformation Testing (SoSym journ 07, Communications of the ACM 2010)

… Aspect Oriented Programming and testing
… Requirements and Model-driven engineering
… ad-hoc network testing
… security contracts

61

Industrial partnerships and valorization

- Contracts
  - European fundings
    - 2000-2004: Café, Families :Product lines, OO modeling
      - NOKIA, Ericsson, Philips …
    - 2005-2006: Modelware :Model-driven Engineering
  - French fundings
    - Cote, Politess, Dali
  - Direct contracts
    - 1995-1998: PEA Aérospatiale/Airbus
    - 2002-2004 :Caroll (INRIA, CEA, THALES)
    - 2008+ : French Defense Department (Security Testing)
- Main french partners
  - THALES TRT and TAS,
  - France Telecom R&D,
  - EADS Test & Services,
  - French Defense Department (DGA)

62

Industrial partnerships and valorization

- Tools developments and valorization
  - UCTSystem
  - Kermeta (metamodeling language for executability)
  - Many prototype tools (AOP testing, Security Testing)
- Two years at France Télécom R&D
  - Real world projects
    - IS Migration, new telecom services modelling and testing (MDE)
- Courses for companies
  - The « Test essentials » program for ALCATEL
  - Thomson, Mitsubishi, EDS…

63

International visibility

- 9 past PhDs, 3 running PhDs
- +90 int. papers (15 journals)
  - Communications of the ACM (CACM)
  - (3) IEEE Trans. on Software Engineering,
  - IEEE Trans. on Reliability,
  - (2) Software, Testing, Verification & Reliability journal (STVR)
  - (2) IEEE Software,
  - IEEE Design & Test,
  - SoSym,
  - Information & Software Technology.
- PC member
  - IEEE ICST, IEEE ISSRE, IEEE Metrics, ICSOFT, ICFI…
- Steering committees
  - Testing: IEEE ICST, Mutation, IWoTA, SecTest
  - Reliability : IEEE ISSRE

64

Cursus and diploma

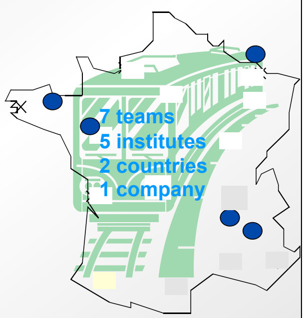
1994. Master in computer science (DEA) from INPG, Grenoble.
1994. Engineering Degree - ENSIMAG.

1994-1997. PhD at INPG- Grenoble
1997-1998. PostDoc at LCIS-INPG lab (Valence)

1998-2004. Assistant professor, Univ. of Rennes 1 - IRISA lab.
2004. Authorization for the management of research (HDR).

2004-2006. France Télécom R&D

Nov. 2006. Associate professor – Telecom Bretagne
Sept. 2008. Professor
Head of the SERVAL team
Julyt. 2009. Professor
Univ. Luxembourg



65

Industrial partnerships and valorization

- Contracts
  - European fundings
    - 2000-2004: Café, Families :Product lines, OO modeling
      - NOKIA, Ericsson, Philips …
    - 2005-2006: Modelware :Model-driven Engineering
  - French fundings
    - Cote, Politess, Dali
  - Direct contracts
    - 1995-1998: PEA Aérospatiale/Airbus
    - 2002-2004 :Caroll (INRIA, CEA, THALES)
    - 2008+ : French Defense Department (Security Testing)
- Main french partners
  - THALES TRT and TAS,
  - France Telecom R&D,
  - EADS Test & Services,
  - French Defense Department (DGA)

66

Industrial partnerships and valorization

- Tools developments and valorization
  - UCTSystem
  - Kermeta (metamodeling language for executability)
  - Many prototype tools (AOP testing, Security Testing)
- Two years at France Télécom R&D
  - Real world projects
    - IS Migration, new telecom services modelling and testing (MDE)
- Courses for companies
  - The « Test essentials » program for ALCATEL
  - Thomson, Mitsubishi, EDS…

67

International visibility

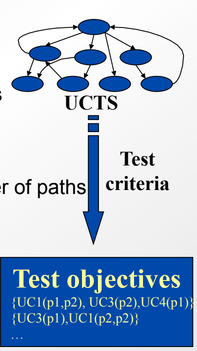
- 9 past PhDs, 3 running PhDs
- 100 int. papers (18 journals)
  - Communications of the ACM (CACM)
  - (3) IEEE Trans. on Software Engineering,
  - IEEE Trans. on Reliability,
  - (2) Empirical Software Engineering
  - (2) Software, Testing, Verification & Reliability journal (STVR)
  - (2) IEEE Software,
  - IEEE Design & Test,
  - (2) SoSym,
  - Information & Software Technology.
- PC member
  - IEEE ICST, IEEE ISSRE, IEEE Metrics, ICSOFT, ICFI…
- Steering committees
  - Testing: IEEE ICST, Mutation, IWoTA, SecTest
  - Reliability : IEEE ISSRE

68

Hard point 2: test objective generation

Test generation

- Test objective
  = path of the UCTS
  = sequence of instantiated use cases
- Generating test objectives
  - Extracting short paths in the UCTS
  - Extracting a « reasonable » number of paths
  →Test criteria
    - 4 structural criteria
    - 1 semantic criterion
    - 1 robustness criterion



69