Model Driven Mutation Applied to Adaptive Systems Testing

ICST Mutation Workshop

Alexandre Bartel[1], Benoit Baudry[2], Freddy Munoz[2], Jacques Klein[1], Tejeddine Mouelhi[1] and Yves Le Traon[1]

[1]University of Luxembourg, Luxembourg [2]INRIA Rennes, France

March 25, 2011

The present project is supported by the National Research Fund, Luxembourg

Motivation

- **Demand increases** for Dynamic Adaptive Systems (DAS) (smart home, Internet of things, ...)
- DAS need to **adapt** to changes in environment (high variability)
- DAS can be **more complex** that other "static" software and rely on an **adaptation logic** layer
- How to test them and in particulay the adaptation logic? How to evaluate test cases?

Outline

Outline

Outline

What a DAS is

- DAS adapt to change in the environment
  - what to monitor? (1)
  - what adaptation for a new behavior? (2)
  - what decision(s) to make to go from (1) to (2)? (3)
- (3) is called the Adaptation Logic (AL):
  - rule based (simple, rules defined at design time)
  - goal based (what, not how)
  - hybrid, ...
- Examples of DAS usage:
  - home automation
  - crisis management
  - air force campaign management
  - ecosystem monitoring
  - ...

Event Condition Action, a Rule Based Adaptation Logic

- **ECA** rules = **E**vent **C**ondition **A**ction rules

Listing 1: A ECA rule

```
1: when requestdensity is 'high' or 'medium'
2: if cacheHandler.size == 0
3: then utility of addCache is 'high'
```

DAS Architecture

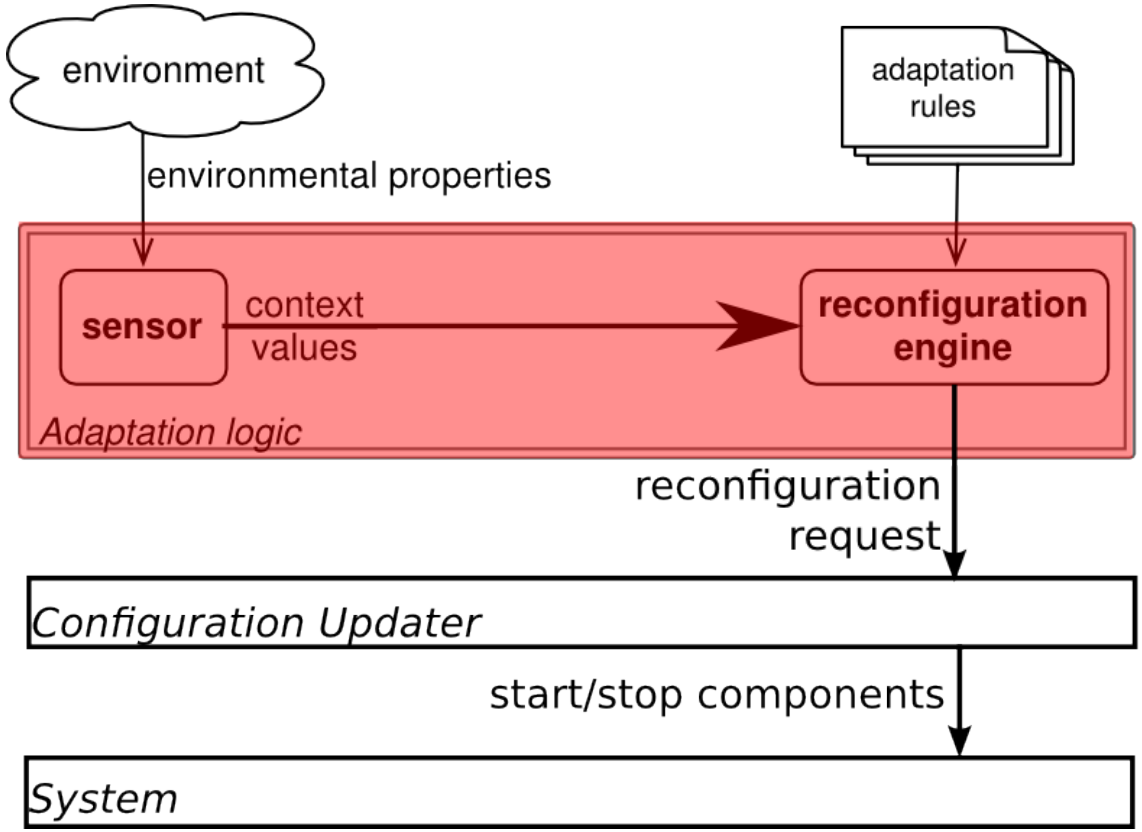

Figure: DAS Architecture (simplified)

Web Server DAS

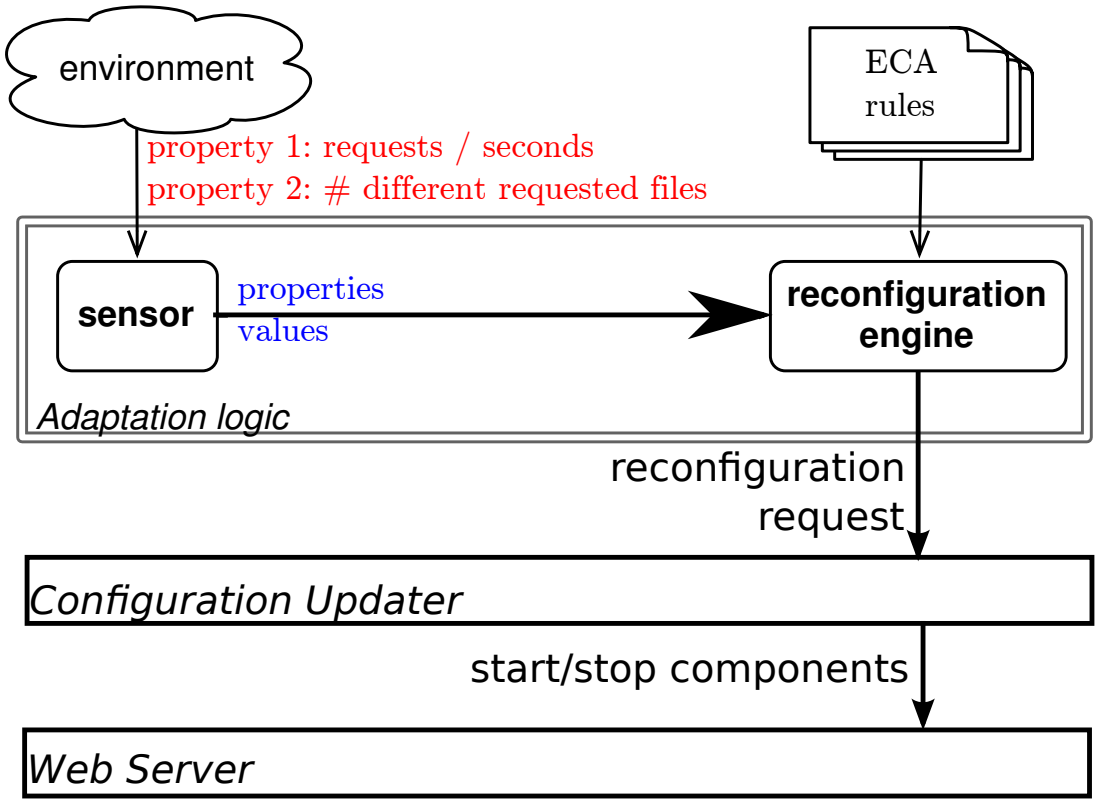

Figure: Web Server Architecture

Web Server DAS

Listing 2: Excerpt of the Webserver's ECA Rules

```
1: when requestdensity is 'high' or 'medium'
2: if cacheHandler.size == 0
3: then utility of addCache is 'high'

5: when requestdensity is 'low'
6: if cacheHandler.size == 0
7: then utility of addCache is 'low'

9: when LOAD is 'high'
10: if FileServers.size <= 10
11: then utility of addFileServer is 'high'

12: when LOAD is 'LOW'
13: if FileServers.size <= 10
14: then utility of addFileServer is 'low'
```

Outline

Brief Introduction to MDE

- Model: **abstraction** of a real-world entity [source code]
- Meta-model: describes **elements** of the model [grammar]
- Why MDE: to simplify, have a high level view of the system
- Software related to MDE:
  - Kermeta [2]: language to **manipulate models**
  - Sintkas [1]: text files to models **conversion** and vice versa

Outline

Outline

Event Condition Action (ECA) Metamodel

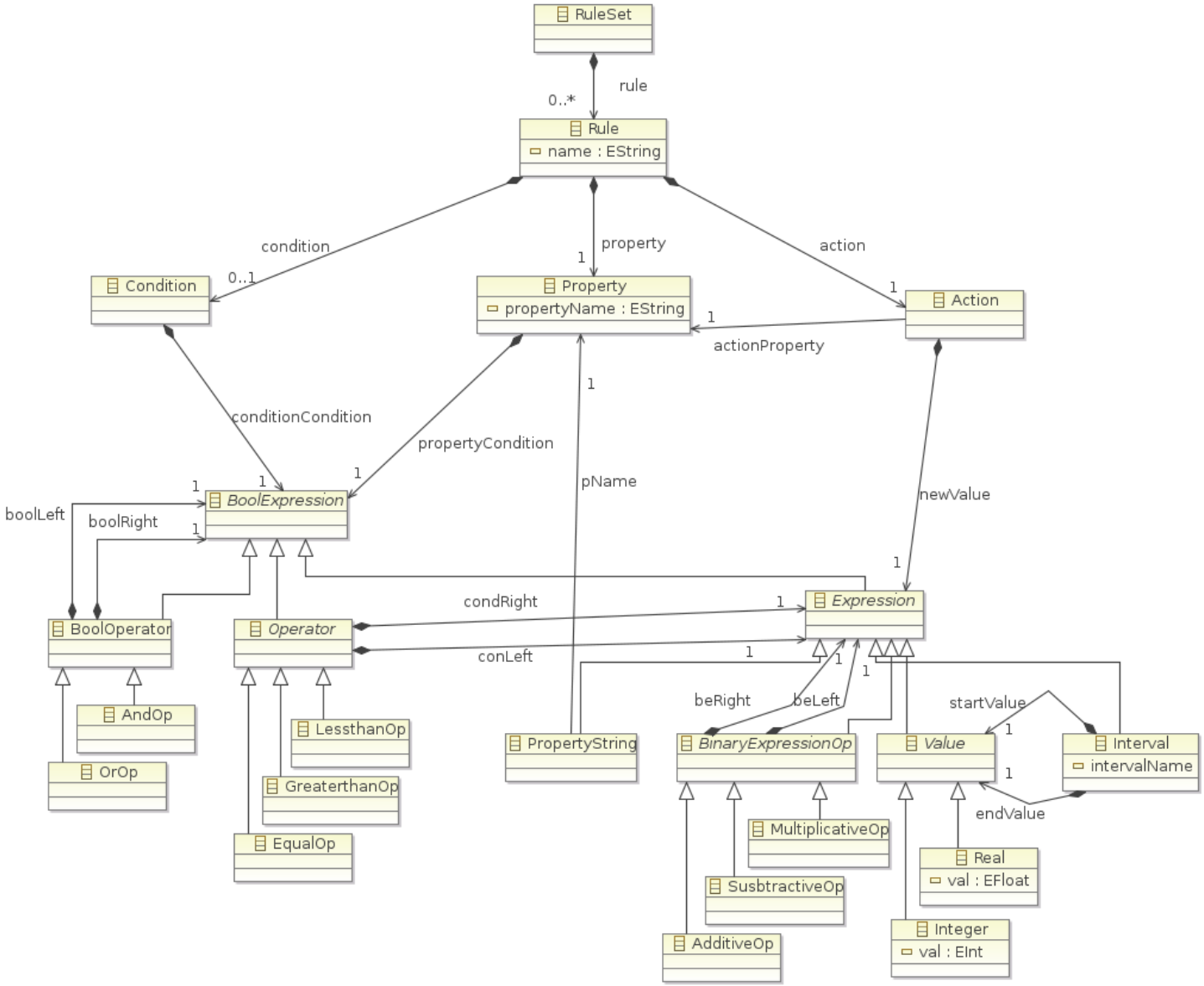

Figure: ECA metamodel

Generic Metamodel

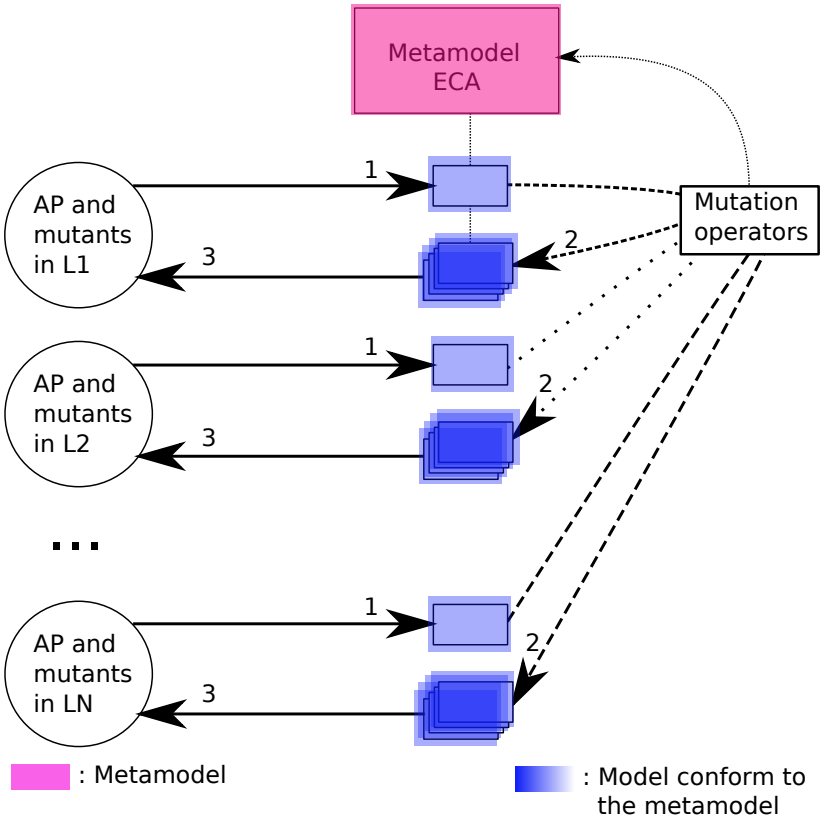

Figure: Generic Metamodel†

† AP: Adaptation Policy = ECA Rules

Outline

Mutant Generation

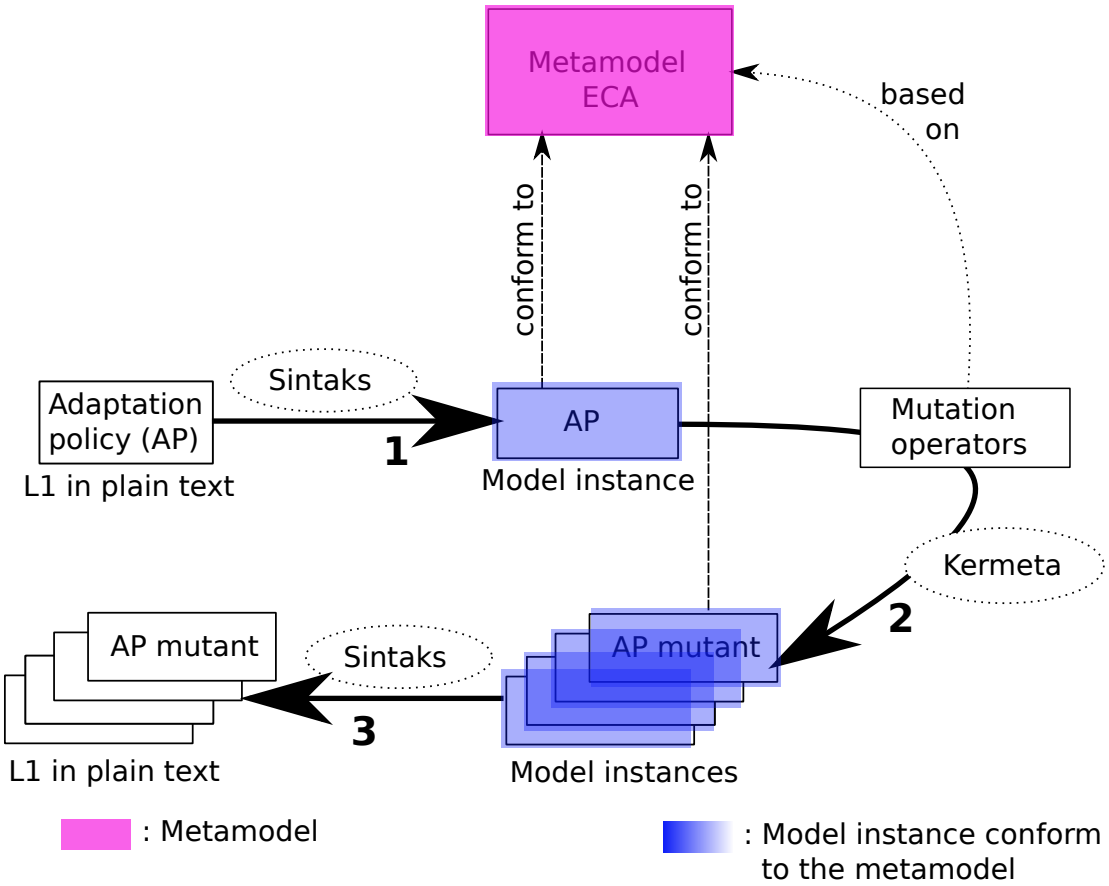

Figure: Mutant Generation

Mutation Operators (1/3): Environmental Completeness Faults

→ Detect neglected environment property (EP) values / EP

1. ICP - Ignore Context Property

   For a given property p, delete each rule that can be executed on p.

2. ISV - Ignore Specific Context Property Value

   For a given couple (property p, value v), delete each rule that can be executed when p equals v.

Listing 3: A Rule

```
1: when requestdensity is 'high' or 'medium'
2: if cacheHandler.size == 0
3: then utility of addCache is 'high'
```

Mutation Operators (2/3): Environmental Completeness Faults

- IMV - Ignore Multiple Context Property Values

For a given set of couples (property p_i, value v_i), delete each rule that can be executed when any p_i (i $\in$ {1,2,...,N}) equals v_i. (At least two rules with different properties are modified/deleted).

Listing 4: A Rule

```
1: when requestdensity is 'high' or 'medium'
2: if cacheHandler.size == 0
3: then utility of addCache is 'high'
```

securityandtrust.lu

UNIVERSITÉ DU LUXEMBOURG

Mutation Operators (3/3): Adaption Correctness Faults

→ Detect false adaptation productions

1. SRA - Swap Rule Action

   The action values from two rules modifying the same property are swapped.

2. Modify Rule Condition Value

   The condition value (always on the right part of the condition), for a condition which uses operator $>$ or $<$, in a rule is decreased or increased, respectively.

Listing 5: A Rule

```
1: when requestdensity is 'high' or 'medium'
2: if cacheHandler.size == 0
3: then utility of addCache is 'high'
```

Mutation Operator in Kermeta

Listing 6: Mutation operator in Kermeta

```
method delPropertyMutant(p:Policy): set Policy[*] is do

  var mutant : Policy
  result := Set<Policy>.new

  // for each properties delete all related
  // rules and create new mutant
  p.properties.each{property |
    // create mutated policy by deleting all
    // rules with property
    mutant := p.copy
    mutant.rules.remove(mutant.rules.detect{x |
      x.containsProperty(property)})
    result.add(mutant)
  }
```

Outline

Adaptive Web Server Architecture

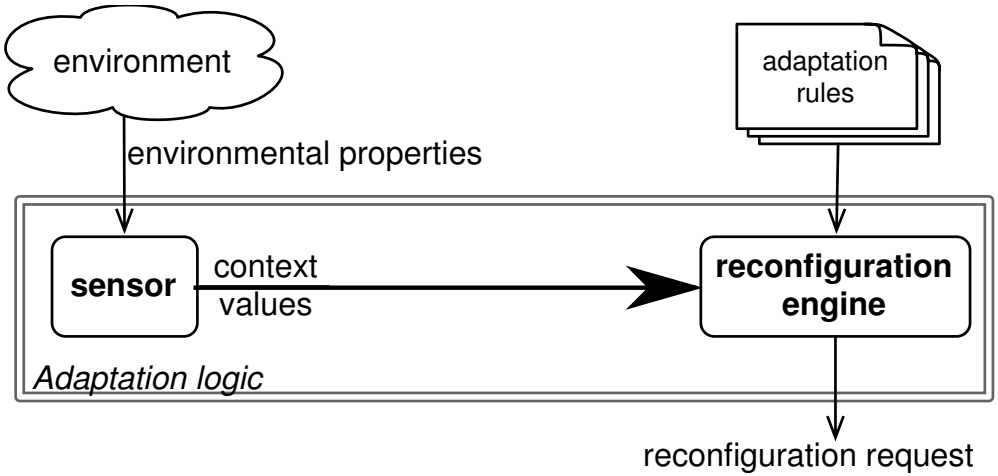

Figure: Architecture of the adaptive web server adaptation logic

- How do we test such a system?
- How good are randomly generated tests for DAS?

Adaptive Web Server Instrumented Architecture

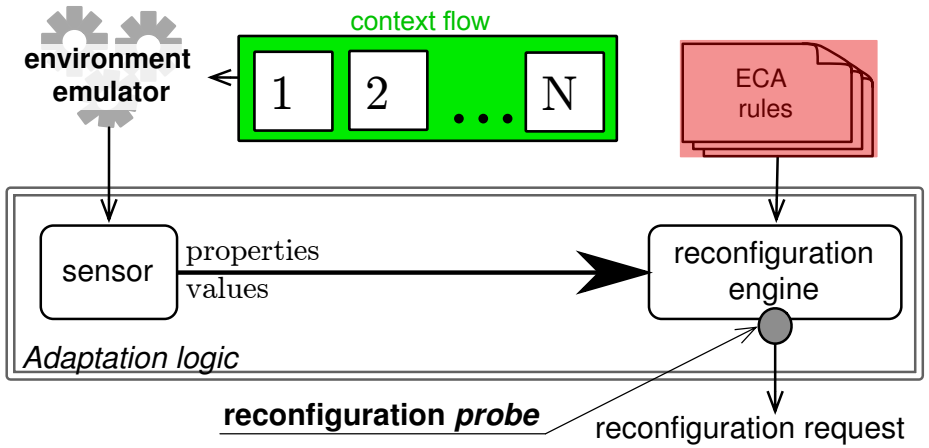

Figure: Instrumented architecture of the adaptive web server adaptation logic

- Each of {1, 2, ..., N} is a set of property values or **context**
- A **context flow** is a sequence of context
- A **new configuration** is generated for each context

Experiment Set-up and Execution

Table: Experiment set-up and execution

# of test suites	30
# of context flows per test suite	10
# of context per flow	20
# of mutants of the adaptation logic	130
Total number of simulations	39.000 $(30 \cdot 10 \cdot 20 \cdot 130)$

- Note: context flows are randomly generated
- Oracle: test suites are "played" a first time to gather generated new configurations

Results

Table: Experiment results

Test suite	Random
minimum mutation score	$91/130 \approx 70\%$
maximum mutation score	$96/130 \approx 74\%$
average mutation score	$93/130 \approx 71\%$

- Open questions:
  - what is the part of equivalent mutants?
  - what is the impact of the (simple) oracle we use?
  - what is the impact of the "reconfiguration engine"?

Outline

Conclusion

- Mutation operators:
  - offer a **qualification environment** for comparing testing techniques applied to action-based adaptative systems
- MDE:
  - provides a **common framework** for such test cases qualification
- Case study:
  - shows the feasibility of the approach
  - other testing techniques should be considered

Future Work

- Future work:
  - completing the set of mutation operators
  - experiments on other case studies (to compare several test generation techniques)
    - much larger case study
    - several environmental properties and interactions
  - studying and specializing our fault model to other adaptation logic technologies (ex: goal oriented)
  - answer open questions

Thank you for your attention.
Questions?


Pierre-Alain Muller, Franck Fleurey, Frédéric Fondement, Michel Hassenforder, Rémi Schneckenburger, Sébastien Gérard, and Jean-Marc Jézéquel.
Model-driven analysis and synthesis of concrete syntax.
In *Proceedings of the MoDELS/UML 2006*, Genova, Italy, October 2006.


Pierre-Alain Muller, Franck Fleurey, and Jean-Marc Jézéquel.
Weaving executability into object-oriented meta-languages.
In S. Kent L. Briand, editor, *Proceedings of MODELS/UML'2005*, volume 3713 of *LNCS*, pages 264–278, Montego Bay, Jamaica, October 2005. Springer.