# Practical aspects for mutation testing of model transformations?

A. García-Domínguez and I. Medina-Bulo

UCASE
University of Cádiz, Spain

MT$^2$ 2014
July 22nd, 2014

# Who am I?

## Short history

- 2007–2008: Started work on mutation testing as a student research assistant
- 2008–2010: Substitute Lecturer at UCA
- 2010–2014: PhD on MDE for performance test generation
- 2014–ongoing: Research Assistant at UCA

## Participation in open source tool development

- MuBPEL: a mutation testing tool for Web Service compositions in BPEL
- EUnit: a unit testing framework for Epsilon
- Also, an Eclipse-based generic UI for mutation analysis tools (only MuBPEL so far)

# Some of the current mutation analysis lines in UCASE

## Mutation testing for WS-BPEL

- 30+ operators implemented and evaluated
- Current work: automatic test case generation

## Mutation testing for C++

- AST-based operators using Clang
- Tooling under development

## Mutation testing for model transformations

- We'd like to apply our techniques and generic tooling to a model transformation language
- But first, the operators need to be defined and implemented: which approach should be taken?

# Some of the current mutation analysis lines in UCASE

## Mutation testing for WS-BPEL

- 30+ operators implemented and evaluated
- Current work: automatic test case generation

## Mutation testing for C++

- AST-based operators using Clang
- Tooling under development

## Mutation testing for model transformations

- We'd like to apply our techniques and generic tooling to a model transformation language
- But first, the operators need to be defined and implemented: which approach should be taken?

# Previous works on mutation analysis for MT

## Mottu (2006): semantic mutation operators

- Mottu (2006) defined a set of "semantic" mutation operators
- Based on model navigation, filtering and creation/modification
- Original experiment done manually with Java implementations

## Fraternali (2009): stacked HOTs to implement operators

- A simple HOT is turned into a m. operator using a HOT
- Paper has example for Mottu's CFCD operator for ATL
- There is some code, but it seems like an early prototype

## Khan (2013): operators for ATL

- 10 concrete operators for ATL (seem to be focused on syntax)
- Prototype implementation: MuATL (availability?)

# Suggestions for discussion

## Requirements for a "practical" tool

- What do we want from mutation analysis?
  - Measuring test effectiveness?
  - Finding irrelevant code?
- What kind of UI/tool integration would be useful?
- Could HOTs be used to mutate any DSL too?

## Technical details about mutation

- How to define mutants: language spec, typical mistakes, emulation of structural coverage criteria?
- How to determine if a mutant is killed? Most approaches use one-to-one generic comparisons, but it might be useful to compare the interpretations of the models.

# End of the slides

## Thank you for your attention!

E-mail:
antonio.garciadominguez@uca.es

Twitter:
@antoniogado

# References



J.-M. Mottu, B. Baudry, Y. Le Traon
Mutation analysis testing for model transformations
Model Driven Architecture – Foundations and Applications, 2006, 376–390.


P. Fraternali, M. Tisi.
Mutation Analysis for Model Transformations in ATL.
MtATL 2009 Proceedings, pp. 145–149.


Y. Khan, J. Hassine.
Mutation Operators for the Atlas Transformation Language
ICSTW 2013 Proceedings, pp. 43–52.