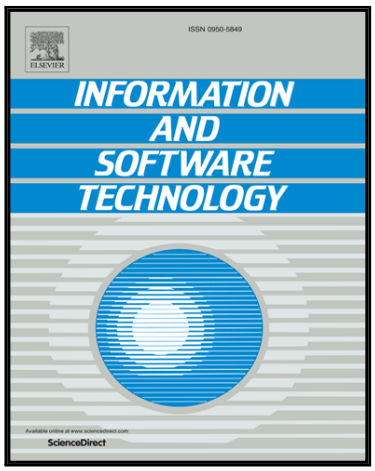

Using Mutation to Design Tests for Aspect-Oriented Models

Birgitta Lindström, Jeff Offutt, Daniel Sundmark, Sten F. Andler, Paul Pettersson

PII: S0950-5849(16)30063-5
DOI: 10.1016/j.infsof.2016.04.007
Reference: INFSOF 5717

To appear in: *Information and Software Technology*

Received date: 21 July 2015
Revised date: 24 March 2016
Accepted date: 11 April 2016

Please cite this article as: Birgitta Lindström, Jeff Offutt, Daniel Sundmark, Sten F. Andler, Paul Pettersson, Using Mutation to Design Tests for Aspect-Oriented Models, *Information and Software Technology* (2016), doi: 10.1016/j.infsof.2016.04.007

Using Mutation to Design Tests for Aspect-Oriented Models

Birgitta Lindström[a,*], Jeff Offutt[b], Daniel Sundmark[c], Sten F. Andler[a], Paul Pettersson[d]

[a] *University of Skövde, Skövde, Sweden*
[b] *George Mason University, Fairfax VA, USA*
[c] *Swedish Institute of Computer Science, Kista, Sweden*
[d] *Mälardalen University, Västerås, Sweden*

Abstract

Context: Testing for properties such as robustness or security is complicated because their concerns are often repeated in many locations and muddled with the normal code. Such "cross-cutting concerns" include things like interrupt events, exception handling, and security protocols. Aspect-oriented (AO) modeling allows developers to model the cross-cutting behavior independently of the normal behavior, thus supporting model-based testing of cross-cutting concerns. However, mutation operators defined for AO programs (source code) are usually not applicable to AO models (AOMs) and operators defined for models do not target the AO features.

Objective: We present a method to design abstract tests at the aspect-oriented model level. We define mutation operators for aspect-oriented models and evaluate the generated mutants for an example system. **Method**: AOMs are mutated with novel operators that specifically target the AO modeling features. Test traces killing these mutant models are then generated. The generated and selected traces are abstract tests that can be transformed to concrete black-box tests and run on the implementation level, to evaluate the behavior of the woven cross-cutting concerns (combined aspect and base models). **Results**: This paper is a significant extension of our paper at Mutation 2015. We present a complete fault model, additional mutation operators, and a thorough analysis of the mutants generated for an example system. **Conclusions**: The analysis shows that some mutants are stillborn (syntactically illegal) but none is equivalent (exhibiting the same behavior as the original model). Additionally, our AOM-specific mutation operators can be combined with pre-existing operators to mutate code or models without any overlap.

Keywords: Model-based testing, Aspect-oriented model, Mutation testing

1. Introduction and Background

Model-based development is gaining widespread use in the software industry. Models provide a graphical view of software behavior that developers find intuitive. In addition, certain types of models, such as state charts [23], Petri nets [47], and timed automata [5, 9] are useful for analysis and verification purposes. Such models can be used by model checkers to verify properties, e.g., to guarantee that a model is free from deadlocks, or to infer the correct ordering of certain events. Moreover, behavioral models can be used to generate test suites that cover the software with respect to model elements or sub paths [6, 48]. Consequently, developers can better understand and analyze complex behavior by modeling software behavior.

1.1. Aspect-Oriented Modeling

One proposed approach to managing complex behavioral models is to separate cross-cutting concerns from the main behavior by using *aspect-oriented modeling* [4, 13, 22, 28, 46]. A *cross-cutting concern* applies throughout multiple locations in the software, and may be crucial to the reliability, performance, security, or robustness of the system. Typical examples include events that require immediate attention, such as intrusion attempts or disturbances. Cross-cutting concerns have a tendency to clutter models, leading to complex models that are hard to analyze.

In aspect-oriented modeling, cross-cutting concerns are modeled as *aspects*, which are separated from the normal behavior, thus creating an *aspect-oriented model* (AOM). The general idea with an AOM is to model the

*Corresponding author: Birgitta Lindström, University of Skövde, Box 408, 541 28 Skövde, Sweden, Tel.:+46 500 448368
Email address: birgitta.lindstrom@his.se (Birgitta Lindström)