An overview on testing automation techniques and strategies

Pedro Tavares

October 16, 2017

Abstract

Testing has long been considered as a time consuming activity that calls for enhanced and powerful techniques. In this paper, we compare some of the techniques and strategies used to take full advantage of testing automation.

Keywords: Software testing, automation, model-driven, data-driven, keyword-driven, hybrid-driven.

Contents

1 Overview

2 Model-Driven Testing

Model-based testing is the automation of black-box test design. A model-based testing tool uses various test generation algorithms and strategies to generate tests from a behavioral model of the system under test (SUT) [3].

With model-based testing we should be able to easily generate a larger test suite from the same model or regenerate the test suite each time the system requirements change.

This approach to software development is increasingly gaining the attention of both industry and academia. Actualy, is considered as leading-edge technology in industry. Unlike traditional development techniques which tend to focus on implementation, model-driven software development stresses the use of models at all levels of the software development process [1].

2.1 Types

There exist many model-driven testing approaches and tools, which vary significantly in their specific designs, testing target, tool support, and evaluation strategies.

2.2 Modelling

2.3 Challenges in Modelling

Model-driven testing can improve considerably the test efficiency and test quality when the models are light and available. The elaboration of a model is, probably, the key for the success of model-driven testing in practice, and if done wrongly can lead to a set of undesirable outcomes [2]:

1. If complex models have to be completed before testing can start, this induces an unacceptable delay for the proper test executions.
2. For complex SUT, like systems of systems, test models need to abstract from a large amount of detail, because otherwise the resulting test model would become unmanageable.
3. The required skills for test engineers writing test models are significantly higher than for test engineers writing sequential test procedures.

2.4 Model-Based Testing and Agile methods

In the eyes of the agile practitioners, writing acceptance tests that specify what the system is supposed to do brings lot of value:

- It forces the customer to specify precisely what is required;
- When the acceptance tests are green the customer has much more confidence that real and useful work has been done;
- An executable specification gives a clear and measurable objective to the development team.

Having the acceptance tests centered around a model it makes not only easier to develop a significant number of acceptance tests, but also to change the model and regenerate those same tests.

2.5 Benefits

Model-based testing can lead to less time and effort spent on testing if the time needed to write and maintain the model plus the time spent on directing the test generation is less than the cost of manually designing and maintaining a test suite.

2.6 Known Limitations

Model-based testing is not as trivial as other testing techniques, and requires a steep learning curve and different testing skills: modeling and programming skills. It is desirable to have a reasonably mature testing process and some experience with automated test execution before fully adopt this approach.

Up to date requirements plays a big role in model-based testing. The test cases are very coupled to the requirements, and requirements change, often. If the model is build based on outdated requirements, it will lead to unreliable results.

When one of these tests fails, it's hard to check if the failure is caused by the SUT, the adaptor, or even from the model itself. This process makes more difficult and time-consuming to find the root cause of the failed test.

Since this approach can generate huge numbers of tests, it becomes necessary to move toward other measurements of test progress instead of relying on the number-of-tests metric. Business code, requirements and model coverage are some of the alternatives.

3 Data-Driven

4 Keyword-Driven

5 Hybrid-Driven

6 Conclusion

References

[1] M. Mussa, S. Ouchani, W. A. Sammane, and A. Hamou-Lhadj. A survey of model-driven testing techniques. In *2009 Ninth International Conference on Quality Software*, pages 167–172, Aug 2009.

[2] Jan Peleska. Industrial-strength model-based testing - state of the art and current challenges. *Electronic Proceedings in Theoretical Computer Science*, page 328, 2013.

[3] Mark Utting and Bruno Legeard. *Practical Model-Based Testing: A Tools Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007.