Implementation-based Refinements for Equivalence Class Testing

Masoumeh Taromirad, Mohammad Reza Mousavi

Centre for Research on Embedded Systems (CERES),
Halmstad University, Sweden
[m.taromirad,m.r.mousavi]@hh.se

It is extremely laborious and ineffective to manually test complex software. Hence, test automation (automated test case generation and execution) has received significant attention. To be able to automatically generate test cases, a model (specification) of the system has to be available. Test cases are then automatically derived from this (preferably formal) model, and are executed on the system under test (SUT). This approach is typically known as model-based testing (MBT) [5].

The key artifact in MBT is a (formal) model that is used to describe a specification of the system. In an MBT process: (1) a model of the SUT is defined, (2) test selection criteria are specified, (3) test cases are automatically generated and executed, and (4) the final verdict of testing is produced.

A substantial challenge associated with MBT is that it is a black-box testing technique, in which the implementation is only accessible through its interfaces. Therefore, generated test suites may leave some untested gaps in a given SUT.

This is also a concern in test data selection. A common practice in generating specification-based test cases is to partition the input domain and then select test data from partitions, which are assumed to contain equally useful values from the testing perspective [1]. The theoretical foundations for dealing with the reduction of test suites have been laid out in [2] as the *regularity-* and the *uniformity hypothesis*. There are different testing techniques developed for partitioning and selecting representatives from the large input domain of parameters, such as equivalence-class based testing [3] and category-partition method [4]. In all of these techniques, partitions are derived from the reference model.

A promising approach to address this issue is to enrich initial test models with structural information exploited from the implementation domain, and then effectively generate concrete test cases and choose test data. With such test suites the coverage of the specification model and the implementation model would be complemented to each other.

We propose an approach to generate test cases considering both specification models and implementation models. The proposal is based on the input equivalence class partition (IECP) testing strategy presented by Huang and Peleska in [3]. We show that, under certain conditions, implementation models can be used in the equivalence class partition testing.

Our approach goes beyond the existing approaches by refining the initial in-out partitions obtained the specification using the structural information from the implementation. The advantages of this approach are twofold: firstly, we

cover the two models in tandem and hence avoid gaps in the specification or implementation. Secondly, we detect their possible structural differences (at interface level) during test case generation and steer the test case to exercise such possibly deviating paths of behavior.

References

1. Paul Ammann and Jeff Offutt. *Introduction to Software Testing.* Cambridge University Press, 2008.
2. Marie-Claude Gaudel. Testing can be formal, too. *TAPSOFT. Lecture Notes in Computer Science*, 1995.
3. Wen ling Huang and Jan Peleska. Complete model-based equivalence class testing. *International Journal on Software Tools for Technology Transfer*, 2014.
4. Thomas J. Ostrand and Marc J. Balcer. The category-partition method for specifying and generating functional tests. *Communications of the ACM*, 1988.
5. Mark Utting, Alexander Pretschner, and Bruno Legeard. A taxonomy of model-based testing. *Journal Software Testing, Verification & Reliability*, 2012.