Integrated Automated Test Case Generation and Static Analysis – Extended Abstract

Jan Peleska
University of Bremen
jp@tzi.de

Cornelia Zahlten
Verified Systems International GmbH
cmz@verified.de

In the general literature discussing software quality assurance, there is a common understanding that the software verification process should be supported by both tests and static analyses [Lig02], the latter ranging from informal inspections to rigorous application of formal methods [CCF$^+$06]. This understanding is also reflected by the applicable standards for – potentially safety-related – software development in avionics and railway control [SC-92, ECfES01]. Moreover, it is advisable that the persons performing tests simultaneously perform the associated analyses, since the more intimate knowledge of the unit under test (UUT), which is usually gained from the inspections, helps to specify more relevant test data and more comprehensive test oracles. As a consequence, integrated tool support for software testing and static analysis is desirable from the perspective of verification experts responsible for performing these tasks within a software development project.

From the tool builders' perspective, it turns out that test automation and static analysis share a considerable amount of common methodology as well as concrete techniques and algorithms. Therefore, the objective of this article is twofold:

- First, we illustrate how fundamental techniques from static static analysis, in particular abstract interpretation, are important pre-requisites for the solution of the test case/test data generation problem.
- Second, it is shown how an integrated test case/test data generation component can be applied to improve static analysis tools by confirming potential UUT errors with concrete test data leading to erroneous program states, or uncovering false alarms by disproving the reachability of these undesired states.

A major portion of the results presented here has been implemented in the RT-Tester tool developed by Verified Systems International GmbH in cooperation with the first author's research group at the University of Bremen. While the tool supports all testing stages – from module testing via software and HW/SW integration testing to system integration testing (see, for example, [Pel02]) – the techniques described here are typically applied on module or software integration level, in order to provide test and static analysis support for the earlier development phases. It should be emphasised that while this presentation refers to a specific tool, our objective is to point out the general architectural and algorithmic features which – according to our experience – should be present in every test automation tool suitable for module testing and static analysis of a wide range of applications.

References

[CCF$^+$06] P. Cousot, R. Cousot, J. Feret, L. Mauborgne, A. Miné, D. Monniaux, and X. Rival. Combination of Abstractions in the ASTRÉE Static Analyzer. In M. Okada and I. Satoh, editors, *Eleventh Annual Asian Computing Science Conference (ASIAN'06)*, pages 1–24, Tokyo, Japan, LNCS, December 6–8 2006. Springer, Berlin. (to appear).

[ECfES01] European Committee for Electrotechnical Standardization. *EN 50128 – Railway applications – Communications, signalling and processing systems – Software for railway control and protection systems*. CENELEC, Brussels, 2001.

[Lig02] Peter Liggesmeyer. *Software-Qualität*. Spektrum Akademischer Verlag, Heidelberg, Berlin, 2002.

[Pel02] Jan Peleska. Formal Methods for Test Automation - Hard Real-Time Testing of Controllers for the Airbus Aircraft Family. In *Proc. of the Sixth Biennial World Conference on Integrated Design & Process Technology (IDPT2002), Pasadena, California, June 23-28, 2002*. Society for Design and Process Science, June 2002. ISSN 1090-9389.

[SC-92] SC-167. *Software Considerations in Airborne Systems and Equipment Certification*. RTCA, 1992.