Executable Specifications in Software Reliability Engineering

Bernhard K. Aichernig
Technical University Graz, Institute for Software Technology (IST),
Münzgrabenstr. 11/II, A-8010 Graz, Austria
E-mail: aichernig@ist.tu-graz.ac.at

In the Handbook of Software Reliability, Lyu mentions the use of formal methods for fault removal [9]:

“When formal methods are in full swing, formal design proofs might be available to achieve mathematical proof of correctness for programs. Also, fault-monitoring assertions could be employed through executable specifications, and test-cases could be automatically generated to achieve efficient software verification.”

In the following, so called light-weight applications of formal methods to Software Reliability Engineering (SRE) [10] are examined. 'Light-weight' means that not correctness proofs are in the center of interest, but the modeling process and the execution of functional specifications. The Vienna Development Method (VDM) [7, 3] will serve to demonstrate that today’s formal methods are mature enough to serve SRE as a complementary tool.

VDM is a widely used formal method, and it can be applied to the construction of a large variety of systems. It is a model-oriented method, i.e. its formal descriptions (specifications) consist of an explicit model of the system being constructed. The 'M' in VDM is the stepwise refinement of a system’s abstract specification towards its implementation, with the benefit that each refinement step can be verified by proving the correctness with respect to its specification.

However, formal proofs need experts and are often out of an industrial project’s scope. Therefore, we envisage the use of formal notations to support the testing process. In [1] we have presented how VDM’s specification language and notion of correctness can be used to synthesize black-box testing oracles. Moreover, [2] explores several possibilities to automate and integrate the approach through available CASE tools, providing e.g., specification interpretation, code-generation and the linking of executable specifications with CORBA objects. In Figure 1 the data-flow of the testing scenario is presented. An implementation is executed with a concrete test input (state). An implemented retrieve function maps the concrete input and output to its abstract representations. A precondition check validates the input and feeds it into the oracle which checks the relation to the produced output. If the postcondition evaluates to true, the test passed.

This black-box technique based on formally specified oracles forms the kernel of an adapted

![Figure 1. DFD of the testing scenario.](image-url)
approach, where refinement steps are verified through testing in accordance to an operational profile. This **reliable refinement**, in contrast to correct refinement, is a pragmatic and still well-founded approach in order to raise the level of reliability throughout the whole development process of software.

In the first step a highly abstract model is created, where operations are specified by means of pre- and postconditions. Industrial applications showed the benefits gained through formally specifying: (1) Validation: The need to find a mathematical model raises questions about unclear, imprecise and missing parts in the user requirement documents. (2) Consistency: The formulation itself reveals inconsistencies in the requirements and thus leads to reliable specifications the developer can depend on. (3) Oracles: Modern tools support the evaluation of the pre- and postcondition functions, and thus the usage of the model as a testing-oracle. In this phase the operational profile is determined, following the method proposed in [10].

In the second phase, a more concrete explicit and executable model is specified, which is tested against its abstract specification. Here, the testing-input, chosen in accordance to the operational profile, is checked against the pre- and postcondition oracles. Like in traditional SRE, a target reliability determines when to stop testing. Still, for critical operations formal proofs might be carried out in order to get a reliability of 1.0 which is correctness. In [8] the benefits of such a combination of proofs and testing are mathematically analyzed.

An executable specification can be viewed as a formal prototype, to which structural test-strategies could be applied. As proposed in [4], test-coverage information could be used in order to measure the adequacy of test-cases synthesized from the operational profile. Additional test-cases may be derived to cover the specified behavior completely, with respect to the chosen test-strategy.

The result of a recent industrial project in the area of flight-control indicates that such assessments of test-cases are necessary [5]. Besides the expected improvement of the informal specification documents, 64 defects have been found, the efficiency of the system test-cases to cover the functionality of a safety critical voice communication system has been analyzed. In order to get a test-coverage measure, the formal specification has been animated with existing system test-cases using IFAD’s VDMTools [6]. The included VDM-interpreter is able to record and visualize the test-coverage information. An overview of this assessment process is given in Fig. 2. A main result of this work was the realization that only 80% of the system’s radio functionality had been covered by the former existing test-cases.

In future, existing specification based test-case generators [11] could be modified in order to create test-cases with respect to both, the operational profile and the functional coverage of the specification. Such a tool should have the availability to generate and execute test-cases incrementally, until a certain reliability has been reached. Here, a **lazy-evaluation testing architecture** comes into our mind, where infinite test-case streams, implemented as lazy lists, are formulated, but only computed until the target reliability is reached. This lazy-evaluation techniques are well-known from functional programs, which themselves can be considered as executable specifications, too.

The described approach highlights the achievable positive synergy effects of combining VDM with SRE. The perspective is twofold: First, from the formal method engineer’s point of view, the operational profile and reliability measures guide the recourses to be spent on various verification techniques ranging from proofs over inspections to specification based testing techniques. Second, the software reliability engineer overcomes the disadvantage that SRE has to be applied rather late.

---

**Figure 2. Verification of test-adequacy.**
in the development process. The application of formal specification techniques facilitates the fault detection in the phase where they are introduced — and early found faults are cheap faults.

References


