Testing and Verification Problems in Industry: Technology Transfer

Position Statement

Dr. Edward Miller,
President
Software Research, Inc.
625 Third Street
San Francisco, California 94107-1997

What IS the Problem?
Within the software test R&D community the technology available is believed to have very positive potential benefits in software engineering; but matching the promise of these extremely difficult and intricate technologies to industrial needs seems to have eluded us in all but a few cases. For software test technology and associated methods to become widely used in practice in the software industry will require changes in the way the technology is moved from the research stage to the application stage.

Software testing gets "bad press": few except the enlightened pay more than lip service to systematic test methods. Yet the truth is that the errors that software test methods can uncover are the 5% of product functionality that make more than 55% of the difference in users' perceptions of quality. We have heard it before, and we know it's true: "Testing is HARD WORK!" (this was first used by my firm's former German Office director, Mr. Harry Sneed, around 1980). Testing is easy to talk about, but hard to implement. And, perhaps more important to this forum, extremely difficult to automate in a truly effective way.

The technology transfer problem seems to be twofold: (1) how to mechanize, and (2) how to transfer laboratory-scale effects of mechanization to users.

What Methods Really Work? How Do We Know?
After nearly twenty plus years of research, and a full decade after the Florida Software Test Conference, testing methods still appear to be a mixed bag. We know that: (1) there must be multiple filters to find subtle defects; (2) you must use mechanized methods to overcome problems with combinatorics and complexity; and, (3) you must get high-level management commitment to even have the chance to try (1) and (2). Even if you use a sophisticated five-filter process — module plus system-level software inspection, functional testing, structural (branch) coverage, and system (call-pair) coverage — you cannot yet really guarantee reduced errors, even though you can easily believe it.

Tools that mechanize some of this kind of methodology exist — in growing numbers, in fact — and their effectiveness is believed, and in many quarters even well understood, but not unambiguously demonstrated. The methods adopted by managers truly work, when they have the enlightenment and the budgets. At slightly higher costs, to be sure, but with significantly higher quality.

Formal and Advanced Methods
The even-better test methods — those using formal techniques such as symbolic evaluation/symbolic execution and all of the associated path-oriented analyses that go with this — suffer an even worse fate. Without effective tools, even in R&D forms of "prototype demonstration of effectiveness" there is no serious acceptance by any part of industry.

Compare the advanced methods' prospects with those more mundane methods mentioned above, based on the 1986 Banff Workshop Proceedings to take an example. For a manager in the field, the advanced techniques don't really offer anything other than an interesting diversion, and possibly the speculation, "Well, if we had that we might use it!" And, the manager making decisions doesn't see any of it as available. At least not today.

What's on the Agenda?
The action agenda is a clear one. It has three elements: effectiveness demonstration, further mechanization, and enhanced technical communication.

Effectiveness of test methods needs to be demonstrated, first in laboratory settings and then in the field, in a way that makes clear what the true costs are and what the payoffs are likely to be. The recent work at NASA/Goddard (with various Universities in support) and the software reliability programs at RADC are important first steps. More steps like them need to be taken. And, the data needs to be ruthlessly front lines where it can be used immediately.

Mechanization of test methods always appears simple. Simple tools are devilishly difficult to field successfully, enough so that trying to field complex ones is just plain scary! The functionalities of certain main-line test tools are now rather well established. Yet an installed base in the hundreds nationwide hardly seems like a big penetration of the market, nor an effective attack on the needs! The tools need to be simpler, easier, more powerful, and have demonstrated effectiveness.

Communication may be the most challenging action area, requiring responses within the test community that are somewhat different from the normal R&D cycle. The test methods, with their supporting tools, empirical studies, and positive effects, must be taught to the wider community. Not just talked about among the software testing experts.

This will mean: (1) more workshops or short conferences focused on applications and technology transfer; (2) seminars or short courses of all kinds and levels within the professional community's educational structure; (3) more publishing about "software testing methods that really work"; and (4) getting larger organizations to be more forthcoming with case studies and "plain truth" about effectiveness. Many organizations don't need and can't use software test methods, and the community should admit that. Meanwhile, some sectors are virtually begging for support and are getting far, far too little.

Summary
Difficult though it may be, the opportunity and challenge of exporting software test methods to the wider world are there. Continued leadership by this group — perhaps especially by this particular group — is clearly the main ingredient!

BIBLIOGRAPHY