



## Point/Counterpoint

*Gerald D. Everett and Bertrand Meyer*

Vol. 26, No. 4  
July/August 2009

This material is presented to ensure timely dissemination of scholarly and technical work. Copyright and all rights therein are retained by authors or by other copyright holders. All persons copying this information are expected to adhere to the terms and constraints invoked by each author's copyright. In most cases, these works may not be reposted without the explicit permission of the copyright holder.

IEEE  computer society

© 2009 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

For more information, please see [www.ieee.org/web/publications/rights/index.html](http://www.ieee.org/web/publications/rights/index.html).



# Test Principles Revisited

Gerald D. Everett, *American Software Testing Qualifications Board*

*Bertrand Meyer's seven testing principles are worthwhile but insufficient.*

**B**ertrand Meyer recently proposed seven principles of software testing.<sup>1</sup> Other sets of principles embraced by testing professionals worldwide suggest that Meyer's list can be improved. One such set is the International Software Testing Qualifications Board Certified Tester Foundation Level Syllabus ([www.istqb.org/downloads/syllabi/SyllabusFoundation.pdf](http://www.istqb.org/downloads/syllabi/SyllabusFoundation.pdf)).

The ISTQB Syllabus more or less agrees with four of Meyer's principles. For example, it partially agrees with Meyer's Principle 5, "An effective testing process must include both manually and automatically produced test cases." However, it disagrees with "must include both." ISTQB test planning deliberately selects the testing approaches most appropriate for a specific project. If a project doesn't need automated tests, none are planned.

## Basic Disagreements

The ISTQB Syllabus has a basic disagreement with Meyer's Principle 1, "To test a program is to try to make it fail," and Principle 7, "A testing strategy's most important property is the number of faults it uncovers as a function of time." The Syllabus subsection 1.2, "What Is Testing," asserts that software testing's purpose is to reduce software user business risk. Meyer's Principle 1 fits the ISTQB Glossary term "negative testing"—purposefully attempting to show that software doesn't work. However, the ISTQB testing purpose also includes the term "positive

testing"—purposefully attempting to verify that the software behaves as required. The combined positive and negative testing results give a much more accurate assessment of the software's potential risk of failure.

Similarly, Meyer's Principle 7 emphasizes quickly finding the maximum number of defects. This emphasis easily diverts attention away from tracking test coverage of requirements, which is a major topic in ISTQB Syllabus subsection 5.3, "Test Progress Monitoring and Control." Some years ago, I worked for a company that internally deployed a new corporate-wide online accounting system. Three years after deployment, the accounting software crashed and the operations staff attempted to restore the system from routine backup tapes. They discovered to their chagrin that all the backup tapes were blank. The intense focus on finding all accounting-function defects resulted in nonfunctional testing (in this case, of backup and restore) being overlooked.

## Meyer's Omissions

Meyer's principles don't address 20 of the 27 ISTQB Syllabus subsections that constitute six major sections. Owing to publication space limitations, I offer four examples here as a representative sampling at the section level.

Meyer doesn't consider the psychology of testing. Section 1 of the Syllabus, "Fundamentals

*Continued on page 64*



## Reach for Standards— Good Ones

**Bertrand Meyer**, *ETH Zurich*

I am flattered by the attention and the reproach that I didn't say all about testing—a great compliment in its implicit suggestion that a 1,800-word column could have contemplated such an aim. Its title was “Seven Principles of Software Testing,” not “The Seven Principles ....” Mr. Everett rightly refers readers to seminal books.

I regretfully disagree with his three main points: that my definition of testing is too narrow, that it is too negative, and that the ISTQB document provides the answers.

### **Too Narrow?**

On the first: Mr. Everett and the ISTQB broaden the definition of testing to cover essentially all of quality assurance. In science one is free to use any term, with a precise definition, to denote anything, but it makes no sense to contradict established practice. The ISTQB's definition goes beyond dynamic techniques commonly known as testing to encompass static ones. Hundreds of publications discuss static analysis, including proofs, versus tests. Such comparisons are of great relevance (including to me as the originator, with Yuri Gurevich, of the Tests and Proofs conferences, <http://tap.ethz.ch>), but the differences remain clear. Ask practitioners or researchers about testing; most will describe dynamic techniques. If the ISTQB wants to extend its scope to quality assurance, it should change its name, not try to redefine decades-old terminology.

### **Too Negative?**

The second criticism targets my thesis that one should direct testing at finding faults. Mr. Everett calls it “confrontational.” I did not address psychology. Does effective testing imply confrontation? I am writing this while on a plane; whether the flight software's tester hurt the programmer's feelings worries me less, right now, than whether he exercised best efforts to uncover deficiencies. (Actually, the informational display has been stuck for the past hour at 958 km from our starting point—I hope critical systems were tested more confrontationally.)

But psychology is not the key consideration. Some of my best friends are testers, and I even knew one who, most years, called her mother on her birthday. Rather, think game theory, with a two-player game: the programmer strives to minimize bugs produced; the tester, to maximize bugs found. Each needs the other: the programmer likes that someone strenuously searches for imperfections (it beats having customers find them); the tester likes that the programmer designs for testability. They might even have coffee together—decaffeinated, to dampen confrontational impulses.

### **Not Quite a Standard Yet**

The third reproach is that I should have relied on the ISTQB's document. Definitely “worthwhile” (the term that Mr. Everett kindly applies

*Testing is not the same as quality assurance.*

*Continued on page 64*

of Testing,” recommends that the test team partner with the developers and business sponsors so that the inevitable discovery of defects is perceived as supportive rather than confrontational. Nothing appears more adversarial than a tester who always tells you what’s wrong (negative testing) and never tells you what’s right (positive testing).

The key concept in section 2, “Testing throughout the Software Life Cycle,” is the number and variety of good testing techniques, tool testing among them, that testers must thoughtfully apply during different phases of the software development life cycle. I don’t recall Meyer relating any of his testing techniques to specific software development phases, yet that’s a substantial challenge for an experienced testing professional.

Section 3, “Static Techniques,” deals with an area of software testing in which

test tools are making their newest and most beneficial impact: testing software documentation. Meyer doesn’t mention automated documentation testing. In contrast, Capers Jones asserts that more than 85 percent of software defects originate in documentation (requirements, specifications, code walkthroughs, and user manuals),<sup>2</sup> which makes static testing more than of passing interest.

Meyer might consider extending his principles to encompass defect management, found in section 5, “Test Management.” Total discovered defects is definitely a quality indicator for the test team, developers, and business sponsors. The types of defects found can be a powerful predictor of software stability. Furthermore, patterns of defects found over time can forecast the numbers of expected defects.<sup>3</sup>

Meyer’s testing principles are worth-

while. But it’s important to note that prior work in this subject area has left a richer legacy than the principles that Meyer proposes.

## References

1. B. Meyer, “Seven Principles of Software Testing,” *Computer*, Aug. 2008, pp. 99–101.
2. C. Jones, *Applied Software Management: Assuring Productivity and Quality*, 2nd ed., McGraw-Hill, 1996.
3. G. Everett and R. McLeod Jr., *Software Testing: Testing across the Entire Software Development Life Cycle*, Wiley-Interscience, 2007, pp. 176–202.

**Gerald D. Everett** is a retired IBM testing expert, a member of the American Software Testing Qualifications Board, and the coauthor of *Software Testing: Testing across the Entire Software Development Life Cycle* (Wiley-Interscience, 2007). Contact him at [gerverett@gvvc.com](mailto:gerverett@gvvc.com).

to my column), it provides some excellent advice from people with extensive testing experience but is not the kind of industry standard that Mr. Everett suggests. It falls short of the quality of IEEE or Ecma standards and often reads like a manifesto—a different genre.

Section 1.2, “What Is Testing?” starts, “A common perception of testing is that it only consists of running tests, i.e. executing the software. This is part of testing, but not all.” The section’s remainder is in the same style, discursive and defensive, but contains no definition of testing; a standard would include a carefully worded definition reflecting expert consensus. Here we read “Debugging and testing are different,” and a few lines down, “The responsibility for each activity is very different, i.e. testers test and developers debug.” This is a textbook

example of how not to write a standard or other reference text: be repetitive, weaken the focus through such words as “very,” and waste the reader’s attention on platitudes (miners mine and programmers program—what should we expect testers and debuggers to do?).

Problems of form and substance permeate the document. To cite just one more, section 1.3 describes testing principles. Some are “worthwhile,” but take Principle 4: “A small number of modules contain most of the defects.” This is an empirical observation, not a principle. A principle should guide action, as Principle 3 (start testing early) does. “Some people forget to call their mothers” is not a principle, although one might work out a principle from it, such as “Don’t forget to call your mom on her birthday.” Similarly, a true principle may

lurk behind Principle 4—perhaps, “Apply to each module a testing effort commensurate with its expected fault density.” The ISTQB offers a collection of gems of testing wisdom, ready neither as a standard (which it does not claim to be) nor as a syllabus (which it does).

**W**e need good testing syllabi and standards. They might include my principles, or some of them, or none at all. Whatever the case, I hope my column brought its contribution, however minute, and thank Mr. Everett for continuing the discussion. ☺

**Bertrand Meyer** is a professor of software engineering at ETH Zurich and chief architect at Eiffel Software (Santa Barbara, California). Contact him at [bertrand.meyer@inf.ethz.ch](mailto:bertrand.meyer@inf.ethz.ch).

## Everett Responds

The first priority of a new test team on a development project is to gain agreement about the testing terminology because software testing terminology remains a professional tripwire. The British Computing Society developed a testing syllabus in the mid-1990s that included both “static testing” and “dynamic testing.” The ISTQB adopted that syllabus. Software testing publications abound that demonstrate how and why static and dynamic testing are just some of the components of “software quality assurance” (A. Spillner, T. Linz, and H. Schaefer, *Software Testing Foundations*, 2nd ed., Rocky Nook Computing, p. 10).

A palpable professional tension exists between software developers and testers; therefore, the test team confrontational risk is real. The “coffee break” approach to building trust and cooperation between developers and testers can work with small development teams as the agile methodology advocates. I’ve discovered through consultative experience that this approach doesn’t scale at all when you’re attempting to avert confrontation between 200 developers and 30 testers on a project.

I’m not surprised at Mr. Meyer’s dissatisfaction with the ISTQB Syllabus’s academic quality. It was written by a professional society, not an academic committee, to help prospective testing professionals learn the full scope of the profession and help experienced testing professionals hone their skills—not earn a degree. In 2008, the ISTQB certified its 100,000th tester worldwide, including the 3,000th in Switzerland ([www.istqb.org/news.htm](http://www.istqb.org/news.htm)). Although Mr. Meyer doesn’t recognize the ISTQB certification as the industry testing standard, it has definitely gained traction in the professional testing community.

## Meyer Responds

“Academic quality”? Quality is not an academic matter—especially not for testing, which is all about software quality, and especially not for a standard, which is all about best practices. Successful IEEE and other standards are mostly the work of industry people (not academics, to whom participation in standards efforts hardly brings any rewards). The reason for their success is devotion to quality, including precise terminology. Despite its merits, the ISTQB Syllabus is not a usable standard, whether according to academic or industrial criteria.

Neither the word “confrontation” (I am addressing Mr. Everett’s points in reverse order) nor the concept figured in my article. Team spirit is clearly important. The goal, however, is not to protect egos but to get the bugs out. Compare testers with security personnel (when you fly out) and customs officers (when you fly in). They do not have to be “confrontational,” but their job is clear: detecting anything that you’re carrying and shouldn’t be. We accept that any ensuing “confrontation” is of lesser concern than the reliability and security benefits. The same holds for testing.

“Static testing”: mainstream software engineering literature covers testing as the main dynamic quality-assurance technique, contrasting it with static techniques such as proofs and static analysis. I just received a new textbook, *Introduction to Software Testing* (Paul Ammann and Jeff Offutt, Cambridge Univ. Press, 2008), which defines testing as “Evaluating software by observing its execution.” This is a useful convention, and I don’t see the point of muddying the waters. On the other hand, who am I to deny some group the right to its own terminology? In the end, all that counts is the goal of quality—as dear, I am sure, to my contradictor as it is to me.

## ADVERTISER INFORMATION

JULY/AUGUST 2009 • IEEE SOFTWARE

### Advertiser

**Elsevier**  
**ICSE 2010**  
**Nu Info Systems, Inc.**  
**Seapine Software**  
**STPCon 2009**

Advertising Personnel  
Marion Delaney,  
IEEE Media, Advertising Dir.  
Phone: +1 415 863 4717  
Email: [md.ieeemedi@ieee.org](mailto:md.ieeemedi@ieee.org)  
Marian Anderson,  
Sr. Advertising Coordinator  
Phone: +1 714 821 8380  
Fax: +1 714 821 4010  
Email: [manderson@computer.org](mailto:manderson@computer.org)  
Sandy Brown, Sr. Business Dev.Mgr.  
Phone: +1 714 821 8380  
Fax: +1 714 821 4010  
Email: [sb.ieeemedi@ieee.org](mailto:sb.ieeemedi@ieee.org)

**Page**  
**Cover 2**  
**1**  
**Cover 4**  
**Cover 3**

### Advertising Sales Representatives

#### Recruitment:

Mid Atlantic  
Lisa Rinaldo  
Tel: +1 732 772 0160  
Fax: +1 732 772 0164  
Email: [lr.ieeemedi@ieee.org](mailto:lr.ieeemedi@ieee.org)

New England  
John Restchack  
Tel: +1 212 419 7578  
Fax: +1 212 419 7589  
Email: [j.restchack@ieee.org](mailto:j.restchack@ieee.org)

Southeast  
Thomas M. Flynn  
Tel: +1 770 645 2944  
Fax:  
+1 770 993 4423  
Email: [flynniom@mindspring.com](mailto:flynniom@mindspring.com)

Midwest/Southwest  
Darcy Giovingo  
Tel: +1 847 498 4520  
Fax: +1 847 498 5911  
Email: [dg.ieeemedi@ieee.org](mailto:dg.ieeemedi@ieee.org)

Northwest/Southern CA  
Tim Matteson  
Tel: +1 310 836 4064  
Fax: +1 310 836 4067  
Email: [tm.ieeemedi@ieee.org](mailto:tm.ieeemedi@ieee.org)

Japan  
Tim Matteson  
Tel: +1 310 836 4064  
Fax: +1 310 836 4067  
Email: [tm.ieeemedi@ieee.org](mailto:tm.ieeemedi@ieee.org)

Europe  
Hilary Turnbull  
Tel: +44 1875 825700  
Fax: +44 1875 825701  
Email: [impress@impressmedia.com](mailto:impress@impressmedia.com)

#### Product:

US East  
Joseph M. Donnelly  
Tel: +1 732 526 7119  
Email: [jmd.ieeemedi@ieee.org](mailto:jmd.ieeemedi@ieee.org)

US Central  
Darcy Giovingo  
Tel: +1 847 498 4520  
Fax: +1 847 498 5911  
Email: [dg.ieeemedi@ieee.org](mailto:dg.ieeemedi@ieee.org)

US West  
Lynne Stickrod  
Tel: +1 415 931 9782  
Fax: +1 415 931 9782  
Email: [ls.ieeemedi@ieee.org](mailto:ls.ieeemedi@ieee.org)

Europe  
Sven Anacker  
Tel: +49 202 27169 11  
Fax:  
+49 202 27169 20  
Email: [sanacker@intermediapartners.de](mailto:sanacker@intermediapartners.de)