Book review

Effective Software Testing: 50 Specific Ways to Improve Your Testing
by Elfriede Dustin

Addison-Wesley, 2003
ISBN 0-201-79429-2
Cover price: US$34.99
304 pages

Some books are great reads, and some are chock-full of useful information. Sometimes you find a book that is both, but unfortunately, this book fits *neither* category.

Judging by the title, you might expect to discover between the covers a lot of tips and techniques to really help you and your team get better at testing. Unfortunately, if you have more than a little experience as a tester, you will find little here that is new to you. Conversely, if you are relatively new to testing, many techniques in Dustin's list of fifty might be new to you, but for the most part, you won't get enough detail to really learn them. For me, reading this book was about as helpful as reading the menu at my favorite take-out restaurant: I've seen all the choices before, and Dustin did not have much new to say about them.

Most of the suggestions about ways to improve your testing, which Dustin calls *items* in the book, are broad statements that, to an experienced tester, simply represent common sense. She describes some of these items in no more than a page, and the coverage is frustratingly skimpy.

Take Item 24, for example: Utilize System Design and Prototypes. An experienced tester might read this and say, "Okay that makes sense. Give me some details." But in the page devoted to this item, you will find only the following advice:

- Prototypes can be helpful in detecting inconsistencies in requirements.
- Prototyping high-risk and complex areas early allows the appropriate testing mechanism (e.g., a test harness) to be developed and refined early in the process.

- Designs and prototypes are helpful in refining test procedures, providing a basis for additions and modifications to requirements. They thus become a basis for creating better, more-detailed test procedures.
- Prototypes are also useful in the design of automated tests using functional testing tools.

As you can see, Dustin just keeps describing the high-level *benefits* of utilizing system design and prototypes -- she never gets down to specific actions a practitioner should take.

The biggest disappointment for me was the lack of good information about test automation. As Dustin authored a solid book called *Automated Software Testing* some time ago, I expected this new book to be strong in this area. But most of the items in the chapter on automated testing had few references to her other book. (That was not true for items in other chapters; in fact they contained so many references to her first book that I think the best strategy is to just go buy that book and forget about this one.)

On the bright side, this book is not *totally* devoid of practical advice. You just have to do some digging to find it. For example, under Item 12: Estimate Test Preparation and Execution Time, Dustin includes several methods for estimating the resources you need for a testing effort. If you have never had to do this, the guidelines here will be helpful. Even experienced testers will find something new here.

Item 13: Define Roles and Responsibilities will also be useful if you have not looked at a process that describes the different roles involved in testing software. Dustin defines these roles in quite a bit of detail, including even more roles and responsibilities than the Rational Unified Process,® or RUP.®

And even Item 31: Know the Different Types of Testing-Support Tools, though it has little explanation, does provide a table with the different type of tools you might want to consider for your project.

I wish I could say more positive things about this book, but at best, it provides a checklist of things to think about. When I compare it to a book like *Lessons Learned in Software Testing* by Kaner, Bach, and Pettichord, I find that it is severely lacking. Kaner and his co-authors provide concise ideas supported by an appropriate level of detail, and they present 293 items in 259 pages. That's a much greater density than Dustin's 50 items in 258 pages -- and testers appreciate that.

-**Gary Pollice**
Technical Marketing
Rational Software
IBM Software Group

For more information on the products or services discussed in this article, please click here and follow the instructions provided. Thank you!