May/June 2005 $9.95 www.StickyMinds.com

BETTER SOFTWARE

The Print Companion to StickyMinds.com

DON'T SELL YOURSELF SHORT
Staying essential in the workplace
PAGE 26

A ROSE BY ANY OTHER NAME
Writing sweet-smelling comments
PAGE 18

High-Performance Testing

Executives and managers, get your performance testing teams out of the pit and ahead of the pack PAGE 32

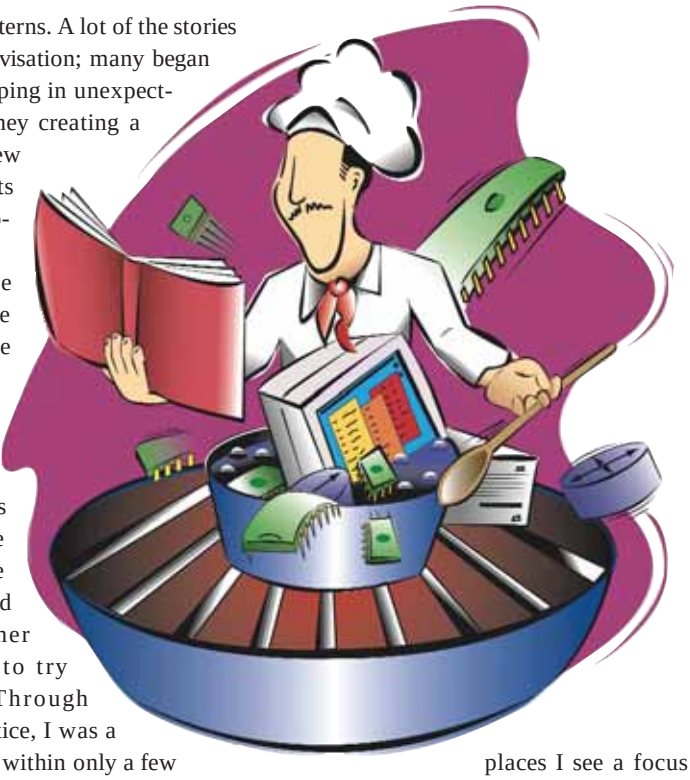
Do You Want Fries With That Test?

by Michael Bolton

In my early 20s, I decided that if I were to be a well-rounded young man (and appeal to young women), a good start would be for me to learn to cook. Like most young men, I didn't want to spend a lot of time and effort every time I walked into the kitchen, but I still wanted to impress. Above all, I wanted to learn skills so I could deal with whatever the situation required: people with different tastes and special dietary needs, the bachelor's meager refrigerator, and the unexplored territory of someone else's kitchen. One of my standard approaches to learning is to head for the bookstore and browse. I spotted a copy of *The 60-Minute Gourmet*, by Pierre Franey. Perfect, I thought.

In the introduction, M. Franey described his philosophy of cooking. He stressed speed and simplicity, which surprised me. I had assumed that all French cooking was elaborate and complex, but true to its title, each recipe required an hour or less to prepare. He also recommended some tools, emphasizing important characteristics and qualities—a heavy saucepan and a balanced, substantial chef's knife. He included a modest list of basic foodstuffs and seasonings to keep on hand. He recommended gathering the tools and measuring out the ingredients before turning on the heat, and he advised continuously tasting the work in progress.

Each recipe was the centerpiece of a meal; each included an accompanying side dish that was simultaneously pleasing not only to the eye and the taste buds but also to the nutritional balance of the meal. In addition, each recipe began with an introductory essay so engaging that, even if I weren't really interested in the dish, I was interested in M. Franey's stories and lore about it. After only a few pages, I had learned a whole slew of new things. The dishes complemented each other so gracefully that, although I didn't recognize the art of it at first, I gradually began to pick up on the patterns. A lot of the stories were about improvisation; many began with friends dropping in unexpectedly and M. Franey creating a new dish with a few simple ingredients from a mostly empty refrigerator.



Darryl Shelton/Illustration Works/Getty Images

In fact, those stories taught me far more than the recipes. While the recipes focused on techniques, the essays taught me skills and made me think, and the results provided me with further encouragement to try new things. Through reading and practice, I was a much better cook within only a few weeks—even when I wasn't working directly from recipes. By experimenting and tasting the results, I was getting fast feedback (literally!). Family and friends (and the aforementioned young ladies) approved.

While I looked at other cookbooks, too, most of them left a lot to be desired. They didn't have M. Franey's personality and didn't convey a love and respect for food; instead, they just laid out the recipes, often poorly. A lot of the methods were ponderous. They contained a lot of directives, but little in the way of advice or experience. They taught me the techniques needed to put together a meal but not the skills necessary to become a better cook.

Several years ago, James Bach challenged me to think about the differences between testing skills and techniques. That has helped to condition my observations as a consultant and testing trainer. When I visit organizations or read or chat with peers about testing, I'm sad to say that in many places I see a focus on techniques over skills. Unskilled testing can be like working behind the counter at a burger joint. You're given a procedure—a technique—that you follow rigorously. It's deliberately hard for you to screw up, and you can consistently produce what the customers are willing to pay for. You can sometimes do a good job by sticking with a specific technique—but *only* if your environment is highly controlled. Software development is never so controlled, so applying a fast-food production model seems inappropriate—although there's still a nagging concern that the customers aren't really thriving on our product.

So what are we to do if we don't have skills? Learn. Unlike burger flippers, most of us have at least some control over our time and some control over the quality of our work. We can use that time and that freedom to build on our skills, and we don't need a mandate from anyone other than ourselves to do it.

In the columns following this one, I'll

show you how to develop skills. The particular skills I'll concentrate on are those that James Bach and I call Rapid Testing. We focus on skills that derive from critical thinking, scientific thinking, general systems thinking, heuristics, and rapid learning. Those skills help us to assess the environment in which we're testing, to identify what we need to do to satisfy our customers, to ask important questions about the product, and to select and apply techniques to answer those questions—all at the maximum possible velocity.

If you ask a tester without skills to test something, no matter how explicit your instructions, you can expect him to give you unexceptional results. If you were to suggest that he do GUI testing, he might not know what that means to you; he might not know what it means to him. Either way, he won't stop to ask if you mean to test the operation of the GUI itself, or to test the application by using the GUI, or something else entirely. He'll click a few buttons or enter some text and some numbers into a dialog. Although he'll know that boundary testing is a technique, he won't have a theory of error or a sense of purpose behind his tests, so he'll likely miss some boundaries that are real, but not obvious. He might remember that preparing state diagrams is a testing technique, but he won't consider the enormous complexity of a state diagram for even a simple program. He might remember that test automation is a technique, but he'll likely get stuck on counting test cases rather than thinking of the value of each one. He'll know that exploratory testing is the name of a technique but, rather than exploring, he'll simply wander, and as a consequence he might bring the technique itself into disrepute.

If he has skills, our tester will begin quickly by clarifying his mission with you. After that, he'll act independently but appropriately. When the specification is unclear or unavailable, he'll be able to make reasoned inferences about the product and to perform fast and focused research. He'll select and craft tests that will highlight the risks that he has helped you to identify and prioritize. Rather than using the specification as his only source of information, he'll be able to identify and select other references that will help him to recognize problems. While performing functional tests, he'll also pay attention to other quality criteria—usability, performance, reliability, and testability. He'll provide you with rapid feedback and he'll report quickly and credibly. Rather than thinking about "the user," he'll consider a variety of user profiles. He'll look at the product as part of a larger system and think expansively about the testing mission.

That expansiveness is important. If you think about coverage solely in terms of code coverage or specification coverage, you may ignore many other important quality dimensions. If your oracles (principles or mechanisms by which we recognize a problem) are limited, there are going to be problems whose significance you might exaggerate or miss altogether. In the next few columns, we'll focus on critical thinking, heuristics, oracles, and coverage, and on developing skills and techniques that will allow us to think quickly but thoroughly about them.

Why? Because, like cooking, testing is something that we do to serve and to satisfy other people. Do you want to be a thoughtful testing cook who expertly uses the tools and ingredients available, or just some guy sitting behind a terminal flipping "testburgers"? **{end}**

Michael Bolton lives in Toronto and teaches James Bach's Rapid Software Testing course all over the world. Contact Michael at mb@developsense.com.

Postscript: As I write this column, some very exciting things are coming from two key skills-based testing educators. Cem Kaner is in the process of posting online the entire Black Box Software Testing course, which he and James Bach wrote. See http://www.testingeducation.org. James has posted the course notes for his Rapid Software Testing course at http://www.satisfice.com. All of this valuable material is being provided free of charge.

Don't Stop Now!

Log on to **StickyMinds.com** and join Michael Bolton and your peers in a conversation about this issue's topic. At the end of the digital column, add your views or just read what others have to say.