foreword to the second edition

The year must have been 2009. I was speaking at the Norwegian Developers Conference in Oslo. (Ah, Oslo in June!) The event was held in a huge sports arena. The conference organizers divided the bleachers into sections, built stages in front of them, and draped them with thick black cloth in order to create eight different session "rooms." I remember I was just about finished with my talk, which was about TDD, or SOLID, or astronomy, or something, when suddenly, from the stage next to me, came this loud and raucous singing and guitar playing.

The drapes were such that I was able to peer around them and see the fellow on the stage next to mine, who was making all the noise. Of course, it was Roy Osherove.

Now, those of you who know me know that breaking into song in the middle of a technical talk about software is something that *I* might just do, if the mood struck me. So as I turned back to my audience, I thought to myself that this Osherove fellow was a kindred spirit, and I'd have to get to know him better.

And getting to know him better is just what I did. In fact, he made a significant contribution to my most recent book *The Clean Coder* and spent three days with me co-teaching a TDD class. My experiences with Roy have all been very positive, and I hope there are many more.

I predict that your experience with Roy, in the reading of this book, will be very positive as well because this book is something special.

Have you ever read a Michener novel? I haven't; but I've been told that they all start at "the atom." The book you're holding isn't a James Michener novel, but it does start at the atom—the atom of unit testing.

Don't be misled as you thumb through the early pages. This is *not* a mere introduction to unit testing. It starts that way, and if you're experienced you can skim those first chapters. As the book progresses, the chapters start to build on each other into a rather startling accumulation of depth. Indeed, as I read the last chapter (not knowing it was the last chapter) I thought to myself that the next chapter would be dealing with world peace—because, I mean, where else can you go after solving the problem of introducing unit testing into obstinate organizations with old legacy systems?

This book is technical—deeply technical. There's a lot of code. That's a good thing. But Roy doesn't restrict himself to the technical. From time to time he pulls out his guitar and breaks into song as he tells anecdotes from his professional past or waxes philosophical about the meaning of design or the definition of integration. He seems to relish in regaling us with stories about some of the things he did really badly in the deep, dark past of 2006.

Oh, and don't be too concerned that the code is all in C#. I mean, who can tell the difference between C# and Java anyway? Right? And besides, it just doesn't matter. He may use C# as a vehicle to communicate his intent, but the lessons in this book also apply to Java, C, Ruby, Python, PHP, or any other programming language (except, perhaps COBOL).

If you're a newcomer to unit testing and test-driven development, or if you're an old hand at it, you'll find this book has something for you. So get ready for a treat as Roy sings you the song "The Art of Unit Testing."

And Roy, please tune that guitar!

ROBERT C. MARTIN (UNCLE BOB)
CLEANCODER.COM