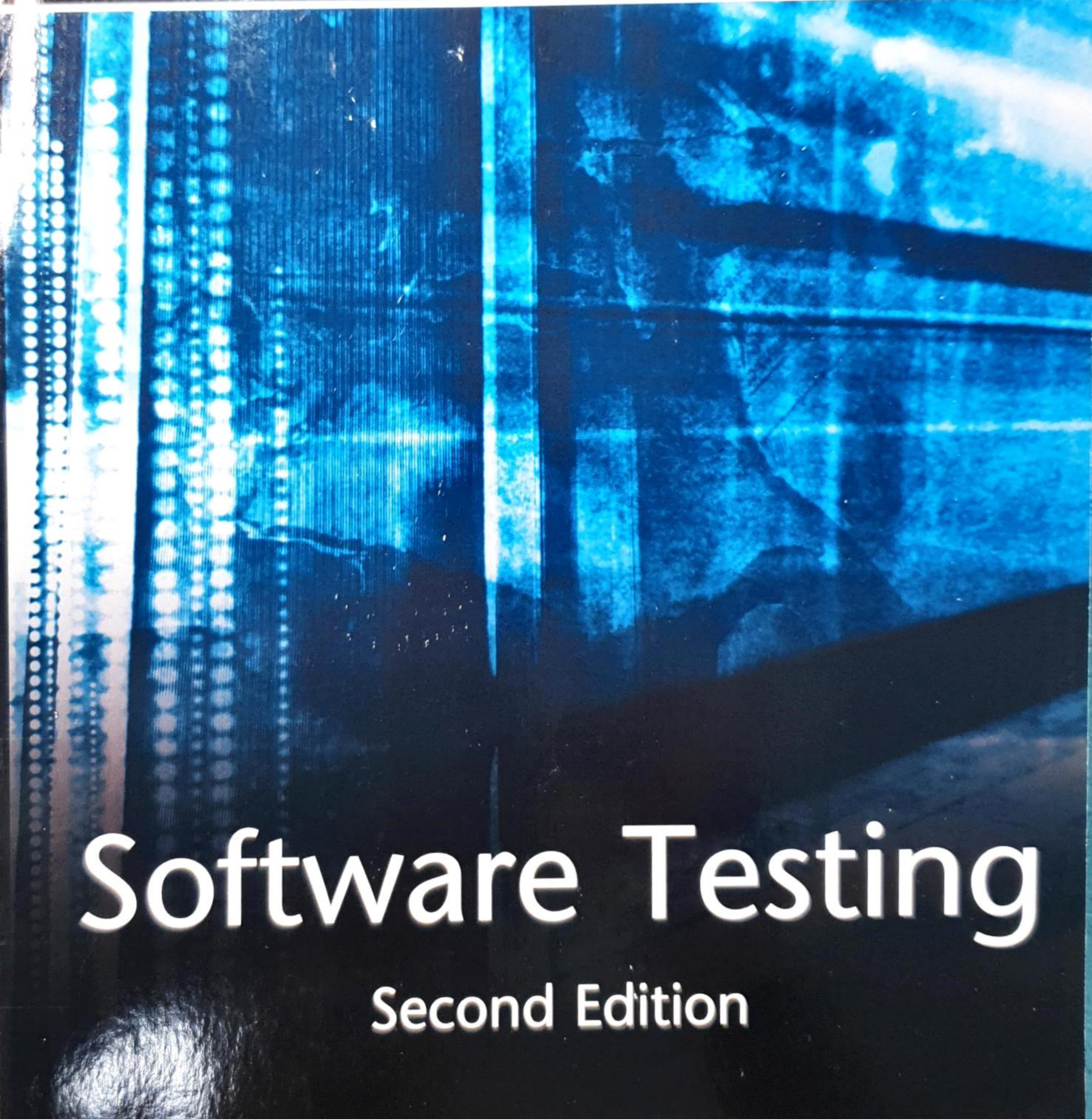
Ron Patton



Software Testing

Second Edition

SAMS

Ron Patton

Software Testing

SAMS
800 East 96th Street, Indianapolis, Indiana 46240

Contents at a Glance

Table of Contents

Introduction

It seems as though each day there's yet another news story about a computer software problem or security breach: a bank reporting incorrect account balances, a Mars lander lost in space, a grocery store scanner charging too much for bananas, or a hacker gaining access to millions of credit card numbers.

Why does this happen? Can't computer programmers figure out ways to make software just plain work? Unfortunately, no. As software gets more complex, gains more features, and is more interconnected, it becomes more and more difficult—actually, mathematically impossible—to create a glitch-free program. Despite how competent the programmers are and how much care is taken, there will always be software problems.

This is where software testing comes in. We've all found those little *Inspector 12* tags in the pockets of our new clothes. Well, software has *Inspector 12s*, too. Most large software companies are so committed to quality they have one or more testers for each programmer. These jobs span the software spectrum from computer games to factory automation to business applications.

This book, *Software Testing*, will introduce you to the basics of software testing, teaching you not just the fundamental technical skills but also the supporting skills necessary to become a successful software tester. You will learn how to immediately find problems in any computer program, how to plan an effective test approach, how to clearly report your findings, and how to tell when your software is ready for release.

About the Second Edition

When I wrote the first edition of *Software Testing*, software security issues were just beginning to make the headlines. Hackers and security problems had always been a problem, but with the interconnectivity explosion that was about to occur, few in the industry could predict the impact that security bugs would have on developers and users of computer software.

In this second edition I've revisited every chapter to emphasize software security issues and point out how the basic testing techniques covered throughout the book can be used to prevent, find, and fix them. I've also added a chapter that specifically addresses how to test for software security bugs.

If you're a reader of the first edition, you know that no matter what you do, your software will still be released with bugs. As you'll learn in the second edition, this axiom still holds true—even for security problems. However, by applying the lessons taught in this book you'll go a long way towards assuring that the most important bugs don't slip through and that your team will create the highest quality and most secure software possible.

Who Should Use This Book?

This book is written for three different groups of people:

- Students or computer hobbyists interested in software testing as a full-time job, internship, or co-op. Read this book before your interview or before your first day on the job to really impress your new boss.
- Career changers wanting to move from their field of expertise into the software industry. There are lots of opportunities for non-software experts to apply their knowledge to software testing. For example, a flight instructor could test a flight simulator game, an accountant could test tax preparation software, or a teacher could test a new child education program.
- Programmers, software project managers, and other people who make up a software development team who want to improve their knowledge and understanding of what software testing is all about.

What This Book Will Do for You

In this book you will learn something about nearly every aspect of software testing:

- How software testing fits into the software development process
- Basic and advanced software testing techniques
- Applying testing skills to common testing tasks
- Improving test efficiency with automation
- Planning and documenting your test effort
- Effectively reporting the problems you find
- Measuring your test effort and your product's progress
- Knowing the difference between testing and quality assurance
- Finding a job as a software tester

Software Necessary to Use This Book

The methods presented in this book are generic and can be applied to testing any type of computer software. But, to make the examples familiar and usable by most people, they are based on simple programs such as Calculator, Notepad, and WordPad included with Windows XP and Windows NT/2000.

Even if you're using a Mac or a PC running Linux or another operating system, you will likely have similar programs available on your computer that you can easily adapt to the text. Be creative! Creativity is one trait of a good software tester.

NOTE

The examples used throughout this book of various applications, software bugs, and software test tools are in no way intended as an endorsement or a disparagement of the software. They're simply used to demonstrate the concepts of software testing.

How This Book Is Organized

This book is designed to lead you through the essential knowledge and skills necessary to become a good software tester. Software testing is not about banging on the keyboard hoping you'll eventually crash the computer. A great deal of science and engineering is behind it, lots of discipline and planning, and there can be lots of fun, too—as you'll soon see.

Part I: The Big Picture

The chapters in Part I lay the foundation for this book by showing you how software products are developed and how software testing fits into the overall development process. You'll see the importance of software testing and gain an appreciation for the magnitude of the job.

- Chapter 1, "Software Testing Background," helps you understand exactly what a software bug is, how serious they can be, and why they occur. You'll learn what your ultimate goal is as a software tester and what traits will help make you a good one.
- Chapter 2, "The Software Development Process," gives you an overview of how a software product is created in the corporate world. You'll learn what components typically go into software, what types of people contribute to it, and the different process models that can be used.
- Chapter 3, "The Realities of Software Testing," brings a reality check to how software is developed. You'll see why no matter how hard you try, software can never be perfect. You'll also learn a few fundamental terms and concepts used throughout the rest of this book.

Part II: Testing Fundamentals

The chapters in Part II teach you the fundamental approaches to software testing. The work of testing software is divided into four basic areas, and you will see the techniques used for each one:

- Chapter 4, "Examining the Specification," teaches you how to find bugs by carefully inspecting the documentation that describes what the software is intended to do.
- Chapter 5, "Testing the Software with Blinders On," teaches you the techniques to use for testing software without having access to the code or even knowing how to program. This is the most common type of testing.
- Chapter 6, "Examining the Code," shows you how to perform detailed analysis of the program's source code to find bugs. You'll learn that you don't have to be an expert programmer to use these techniques.
- Chapter 7, "Testing the Software with X-Ray Glasses," teaches you how you can improve your testing by leveraging information you gain by reviewing the code or being able to see it execute while you run your tests.

Part III: Applying Your Testing Skills

The chapters in Part III take the techniques that you learned in Part II and apply them to some real-world scenarios that you'll encounter as a software tester:

- Chapter 8, "Configuration Testing," teaches you how to organize and perform software testing on different hardware configurations and platforms.
- Chapter 9, "Compatibility Testing," teaches you how to test for issues with different software applications and operating systems interacting with each other.
- Chapter 10, "Foreign-Language Testing," shows you that a whole world of software is out there and that it's important to test for the special problems that can arise when software is translated into other languages.
- Chapter 11, "Usability Testing," teaches you how to apply your testing skills when checking a software application's user interface and how to assure that your software is accessible to the disabled.
- Chapter 12, "Testing the Documentation," explains how to examine the software's documentation such as help files, user manuals, even the marketing material, for bugs.
- Chapter 13, "Testing for Software Security," shows you how to find bugs that allow hackers to gain access to (supposedly) secure computer systems and data.

- Chapter 14, "Website Testing," takes everything you've learned so far and applies it to a present-day situation. You'll see how something as simple as testing a website can encompass nearly all aspects of software testing.

Part IV: Supplementing Your Testing

The chapters in Part IV show you how to improve your test coverage and capability by leveraging both technology and people to perform your testing more efficiently and effectively:

- Chapter 15, "Automated Testing and Test Tools," explains how you can use computers and software to test other software. You'll learn several different methods for automating your tests and using tools. You'll also learn why using technology isn't foolproof.
- Chapter 16, "Bug Bashes and Beta Testing," shows you how to use other people to see the software differently and to find bugs that you completely overlooked.

Part V: Working with Test Documentation

The chapters in Part V cover how software testing is documented so that its plans, bugs, and results can be seen and understood by everyone on the project team:

- Chapter 17, "Planning Your Test Effort," shows you what goes into creating a test plan for your project. As a new software tester, you likely won't write a test plan from scratch, but it's important to know what's in one and why.
- Chapter 18, "Writing and Tracking Test Cases," teaches you how to properly document the test cases you develop so that you and other testers can use them.
- Chapter 19, "Reporting What You Find," teaches you how to tell the world when you find a bug, how to isolate the steps necessary to make it recur, and how to describe it so that others will understand and want to fix it.
- Chapter 20, "Measuring Your Success," describes various types of data, charts, and graphs used to gauge both your progress and success at testing and your software project's steps toward release.

Part VI: The Future

The chapters in Part VI explain where the future lies in software testing and set the stage for your career:

- Chapter 21, "Software Quality Assurance," teaches you the big difference between software testing and quality assurance. You'll learn about different software industry goals such as ISO 9000 and the Capabilities Maturity Model and what it takes to achieve them.
- Chapter 22, "Your Career as a Software Tester," gives you that kick in the behind to go out and be a software tester. You'll learn what types of jobs are available and where to look for them. You'll also find many pointers to more information.

Appendix

Each chapter in this book ends with a short quiz where you can try out the testing concepts that you learn. The answers appear in Appendix A, "Answers to Quiz Questions."

Conventions Used in This Book

This book uses several common conventions to help teach software testing topics. Here's a summary of those typographical conventions:

- New terms are emphasized in *italics* the first time they are used.
- Commands and computer output appear in a special `monospaced` font.
- Words you type appear in a **`monospaced bold`** font.

In addition to typographical conventions, the following special elements are included to set off different types of information to make them easily recognizable.

NOTE

Special notes augment the material you read in each chapter. These notes clarify concepts and procedures.

TIP

You'll find various tips that offer shortcuts and solutions to common problems.

REMINDER

Reminders refer to concepts discussed in previous chapters to help refresh your memory and reinforce important concepts.