



**Effective Methods
for Software Testing**

Third Edition

William E. Perry



WILEY

Wiley Publishing, Inc.

Effective Methods for Software Testing

Third Edition



Effective Methods for Software Testing

Third Edition

William E. Perry



WILEY

Wiley Publishing, Inc.

Effective Methods for Software Testing, Third Edition

Published by

Wiley Publishing, Inc.

10475 Crosspoint Boulevard

Indianapolis, IN 46256

www.wiley.com

Copyright © 2006 by Wiley Publishing, Inc., Indianapolis, Indiana

Published simultaneously in Canada

ISBN-13: 978-0-7645-9837-1

ISBN-10: 0-7645-9837-6

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

3MA/QV/QU/QW/IN

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Legal Department, Wiley Publishing, Inc., 10475 Crosspoint Blvd., Indianapolis, IN 46256, (317) 572-3447, fax (317) 572-4355, or online at <http://www.wiley.com/go/permissions>.

Limit of Liability/Disclaimer of Warranty: The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Website is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or Website may provide or recommendations it may make. Further, readers should be aware that Internet Websites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services or to obtain technical support, please contact our Customer Care Department within the U.S. at (800) 762-2974, outside the U.S. at (317) 572-3993 or fax (317) 572-4002.

Library of Congress Control Number: 2005036216

Trademarks: Wiley and related trade dress are registered trademarks of Wiley Publishing, Inc., in the United States and other countries, and may not be used without written permission. All other trademarks are the property of their respective owners. Wiley Publishing, Inc., is not associated with any product or vendor mentioned in this book.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

This book is dedicated to my wife Cynthia, who for many years has been "testing" my ability to live in accordance with our marriage vows. She taught me that testing is a lifelong process, that testing is necessary to ensure that you are meeting your objectives, and that testing can be fun if it is performed correctly. Thank you, Cynthia. What you have taught me is incorporated into many of the concepts in this book.



About the Author

William E. Perry holds degrees from Clarkson University, University of Rochester, and Rochester Institute of Technology. Bill also holds the following professional certifications: CPA (Certified Public Accountant), CIA (Certified Internal Auditor), CISA (Certified Information Services Auditor), CSQA (Certified Software Quality Analyst), and CSTE (Certified Software Tester). He has been an examiner for the Malcolm Baldrige National Quality Award, and served on standards committees for NIST (National Institute of Standards and Technology), IEEE (Institute of Electrical and Electronics Engineers), AICPA (American Institute of Certified Public Accountants) and ISACA (Information Systems Audit and Control Association).

In 1980, Bill founded the Quality Assurance Institute (QAI), a professional association for testers. QAI offers professional certification for Quality Assurance, Software Testing, Software Project Leaders and Business Analyst Professional. More than 27,000 individuals have been certified since the inception of the program.

Bill has authored more than 50 books, many published by John Wiley & Sons. He recently founded the Internal Control Institute (ICI). ICI and St. Petersburg College recently formed the Internal Control Center of Excellence to share best internal control practices, hold conferences on emerging internal control practices, and to offer e-learning courses and a professional certification in internal control.



Executive Editor

Robert Elliott

Production Editor

Felicia Robinson

Editorial Manager

Mary Beth Wakefield

Production Manager

Tim Tate

Vice President and Executive Group

Publisher

Richard Swadley

Vice President and Executive Publisher

Joseph B. Wikert

Project Coordinator

Michael Kruzil

Graphics and Production Specialists

Carrie Foster

Mary J. Gillot

Lauren Goddard

Denny Hager

Joyce Haughey

Stephanie D. Jumper

Rashell Smith

Quality Control Technicians

John Greenough

Brian H. Walls

Proofreading and Indexing

Techbooks



Contents

Introduction	xxv
Part I Assessing Testing Capabilities and Competencies	1
Chapter 1 Assessing Capabilities, Staff Competency, and User Satisfaction	3
The Three-Step Process to Becoming a World-Class Testing Organization	3
Step 1: Define a World-Class Software Testing Model	5
Customizing the World-Class Model for Your Organization	7
Step 2: Develop Baselines for Your Organization	8
Assessment 1: Assessing the Test Environment	8
Implementation Procedures	9
Verifying the Assessment	13
Assessment 2: Assessing the Capabilities of Your Existing Test Processes	13
Assessment 3: Assessing the Competency of Your Testers	14
Implementation Procedures	14
Verifying the Assessment	16
Step 3: Develop an Improvement Plan	16
Summary	18
Part II Building a Software Testing Environment	35
Chapter 2 Creating an Environment Supportive of Software Testing	37
Minimizing Risks	38
Risk Appetite for Software Quality	38
Risks Associated with Implementing Specifications	39
Faulty Software Design	39
Data Problems	39

Risks Associated with Not Meeting Customer Needs	40
Developing a Role for Software Testers	43
Writing a Policy for Software Testing	45
Criteria for a Testing Policy	45
Methods for Establishing a Testing Policy	46
Economics of Testing	47
Testing—An Organizational Issue	50
Management Support for Software Testing	50
Building a Structured Approach to Software Testing	51
Requirements	54
Design	54
Program	55
Test	55
Installation	55
Maintenance	55
Developing a Test Strategy	56
Use Work Paper 2-1	58
Use Work Paper 2-2	58
Summary	60
Chapter 3 Building the Software Testing Process	63
Software Testing Guidelines	63
Guideline #1: Testing Should Reduce Software Development Risk	64
Guideline #2: Testing Should Be Performed Effectively	65
Guideline #3: Testing Should Uncover Defects	65
Defects Versus Failures	65
Why Are Defects Hard to Find?	66
Guideline #4: Testing Should Be Performed Using Business Logic	67
Guideline #5: Testing Should Occur Throughout the Development Life Cycle	68
Guideline #6: Testing Should Test Both Function and Structure	69
Why Use Both Testing Methods?	69
Structural and Functional Tests Using Verification and Validation Techniques	69
Workbench Concept	71
Testing That Parallels the Software Development Process	72
Customizing the Software-Testing Process	74
Determining the Test Strategy Objectives	74
Determining the Type of Development Project	75
Determining the Type of Software System	76
Determining the Project Scope	77
Identifying the Software Risks	77
Determining When Testing Should Occur	79
Defining the System Test Plan Standard	79

	Defining the Unit Test Plan Standard	83
	Converting Testing Strategy to Testing Tactics	83
	Process Preparation Checklist	86
	Summary	86
Chapter 4	Selecting and Installing Software Testing Tools	103
	Integrating Tools into the Tester’s Work Processes	103
	Tools Available for Testing Software	104
	Selecting and Using Test Tools	108
	Matching the Tool to Its Use	109
	Selecting a Tool Appropriate to Its Life Cycle Phase	109
	Matching the Tool to the Tester’s Skill Level	111
	Selecting an Affordable Tool	114
	Training Testers in Tool Usage	116
	Appointing Tool Managers	117
	Prerequisites to Creating a Tool Manager Position	118
	Selecting a Tool Manager	118
	Assigning the Tool Manager Duties	119
	Limiting the Tool Manager’s Tenure	120
	Summary	120
Chapter 5	Building Software Tester Competency	125
	What Is a Common Body of Knowledge?	125
	Who Is Responsible for the Software Tester’s Competency?	126
	How Is Personal Competency Used in Job Performance?	126
	Using the 2006 CSTE CBOK	127
	Developing a Training Curriculum	128
	Using the CBOK to Build an Effective Testing Team	129
	Summary	131
Part III	The Seven-Step Testing Process	151
Chapter 6	Overview of the Software Testing Process	153
	Advantages of Following a Process	153
	The Cost of Computer Testing	154
	Quantifying the Cost of Removing Defects	155
	Reducing the Cost of Testing	156
	The Seven-Step Software Testing Process	156
	Objectives of the Seven-Step Process	159
	Customizing the Seven-Step Process	160
	Managing the Seven-Step Process	161
	Using the Tester’s Workbench with the Seven-Step Process	162
	Workbench Skills	163
	Summary	164
Chapter 7	Step 1: Organizing for Testing	165
	Objective	165
	Workbench	166
	Input	167

Do Procedures	167
Task 1: Appoint the Test Manager	167
Task 2: Define the Scope of Testing	168
Task 3: Appoint the Test Team	168
Internal Team Approach	169
External Team Approach	170
Non-IT Team Approach	170
Combination Team Approach	170
Task 4: Verify the Development Documentation	171
Development Phases	171
Measuring Project Documentation Needs	174
Determining What Documents Must Be Produced	175
Determining the Completeness of Individual Documents	179
Determining Documentation Timeliness	180
Task 5: Validate the Test Estimate and Project Status	
Reporting Process	181
Validating the Test Estimate	182
Testing the Validity of the Software Cost Estimate	185
Calculating the Project Status Using a Point System	189
Check Procedures	200
Output	200
Summary	200
Chapter 8 Step 2: Developing the Test Plan	209
Overview	209
Objective	210
Concerns	210
Workbench	211
Input	212
Do Procedures	212
Task 1: Profile the Software Project	212
Conducting a Walkthrough of the Customer/User Area	212
Developing a Profile of the Software Project	213
Task 2: Understand the Project Risks	215
Task 3: Select a Testing Technique	222
Structural System Testing Techniques	223
Functional System Testing Techniques	229
Task 4: Plan Unit Testing and Analysis	235
Functional Testing and Analysis	236
Structural Testing and Analysis	238
Error-Oriented Testing and Analysis	240
Managerial Aspects of Unit Testing and Analysis	243
Task 5: Build the Test Plan	244
Setting Test Objectives	245
Developing a Test Matrix	245
Defining Test Administration	250
Writing the Test Plan	251

Task 6: Inspect the Test Plan	254
Inspection Concerns	255
Products/Deliverables to Inspect	256
Formal Inspection Roles	256
Formal Inspection Defect Classification	258
Inspection Procedures	259
Check Procedures	262
Output	262
Guidelines	262
Summary	263
Chapter 9 Step 3: Verification Testing	291
Overview	292
Objective	293
Concerns	294
Workbench	294
Input	296
The Requirements Phase	296
The Design Phase	296
The Programming Phase	297
Do Procedures	298
Task 1: Test During the Requirements Phase	298
Requirements Phase Test Factors	299
Preparing a Risk Matrix	302
Performing a Test Factor Analysis	310
Conducting a Requirements Walkthrough	312
Performing Requirements Tracing	314
Ensuring Requirements Are Testable	315
Task 2: Test During the Design Phase	316
Scoring Success Factors	316
Analyzing Test Factors	318
Conducting a Design Review	320
Inspecting Design Deliverables	322
Task 3: Test During the Programming Phase	323
Desk Debugging the Program	325
Performing Programming Phase Test Factor Analysis	326
Conducting a Peer Review	328
Check Procedures	330
Output	331
Guidelines	331
Summary	332
Chapter 10 Step 4: Validation Testing	409
Overview	409
Objective	410
Concerns	410
Workbench	410
Input	411

Do Procedures	412
Task 1: Build the Test Data	412
Sources of Test Data/Test Scripts	412
Testing File Design	413
Defining Design Goals	414
Entering Test Data	414
Applying Test Files Against Programs That Update	
Master Records	414
Creating and Using Test Data	415
Payroll Application Example	416
Creating Test Data for Stress/Load Testing	430
Creating Test Scripts	430
Task 2: Execute Tests	434
Task 3: Record Test Results	436
Documenting the Deviation	437
Documenting the Effect	438
Documenting the Cause	438
Check Procedures	439
Output	439
Guidelines	439
Summary	440
Chapter 11 Step 5: Analyzing and Reporting Test Results	459
Overview	459
Concerns	460
Workbench	460
Input	461
Test Plan and Project Plan	461
Expected Processing Results	461
Data Collected during Testing	461
Test Results Data	462
Test Transactions, Test Suites, and Test Events	462
Defects	462
Efficiency	463
Storing Data Collected During Testing	463
Do Procedures	463
Task 1: Report Software Status	464
Establishing a Measurement Team	465
Creating an Inventory of Existing Project Measurements	465
Developing a Consistent Set of Project Metrics	466
Defining Process Requirements	466
Developing and Implementing the Process	466
Monitoring the Process	466
Task 2: Report Interim Test Results	470
Function/Test Matrix	470
Functional Testing Status Report	471
Functions Working Timeline Report	472
Expected Versus Actual Defects Uncovered Timeline Report	472

Defects Uncovered Versus Corrected Gap Timeline Report	473
Average Age of Uncorrected Defects by Type Report	475
Defect Distribution Report	475
Normalized Defect Distribution Report	476
Testing Action Report	477
Interim Test Report	478
Task 3: Report Final Test Results	478
Individual Project Test Report	480
Integration Test Report	480
System Test Report	480
Acceptance Test Report	482
Check Procedures	482
Output	482
Guidelines	482
Summary	483
Chapter 12 Step 6: Acceptance and Operational Testing	491
Overview	491
Objective	492
Concerns	493
Workbench	494
Input Procedures	495
Task 1: Acceptance Testing	496
Defining the Acceptance Criteria	497
Developing an Acceptance Plan	498
Executing the Acceptance Plan	499
Developing Test Cases (Use Cases) Based on How Software Will Be Used	500
Task 2: Pre-Operational Testing	503
Testing New Software Installation	509
Testing the Changed Software Version	509
Monitoring Production	512
Documenting Problems	513
Task 3: Post-Operational Testing	513
Developing and Updating the Test Plan	514
Developing and Updating the Test Data	515
Testing the Control Change Process	517
Conducting Testing	518
Developing and Updating Training Material	518
Check Procedures	522
Output	522
Is the Automated Application Acceptable?	522
Automated Application Segment Failure Notification	523
Is the Manual Segment Acceptable?	523
Training Failure Notification Form	524
Guidelines	524
Summary	525

Chapter 13	Step 7: Post-Implementation Analysis	571
	Overview	571
	Concerns	572
	Workbench	572
	Input	574
	Do Procedures	574
	Task 1: Establish Assessment Objectives	574
	Task 2: Identify What to Measure	575
	Task 3: Assign Measurement Responsibility	575
	Task 4: Select Evaluation Approach	575
	Task 5: Identify Needed Facts	576
	Task 6: Collect Evaluation Data	577
	Task 7: Assess the Effectiveness of Testing	577
	Using Testing Metrics	577
	Check Procedures	580
	Output	580
	Guidelines	581
	Summary	581
Part IV	Incorporating Specialized Testing Responsibilities	583
Chapter 14	Software Development Methodologies	585
	How Much Testing Is Enough?	585
	Software Development Methodologies	586
	Overview	586
	Methodology Types	587
	Software Development Life Cycle	588
	Defining Requirements	592
	Categories	592
	Attributes	593
	Methodology Maturity	596
	Competencies Required	598
	Staff Experience	600
	Configuration-Management Controls	600
	Basic CM Requirements	600
	Planning	602
	Data Distribution and Access	602
	CM Administration	602
	Configuration Identification	603
	Configuration Control	605
	Measuring the Impact of the Software Development Process	605
	Summary	606
Chapter 15	Testing Client/Server Systems	611
	Overview	611
	Concerns	612
	Workbench	613
	Input	614

Do Procedures	614
Task 1: Assess Readiness	614
Software Development Process Maturity Levels	615
Conducting the Client/Server Readiness Assessment	621
Preparing a Client/Server Readiness Footprint Chart	621
Task 2: Assess Key Components	622
Task 3: Assess Client Needs	622
Check Procedures	624
Output	624
Guidelines	624
Summary	624
Chapter 16 Rapid Application Development Testing	633
Overview	633
Objective	634
Concerns	634
Testing Iterations	634
Testing Components	635
Testing Performance	635
Recording Test Information	635
Workbench	635
Input	636
Do Procedures	636
Testing Within Iterative RAD	636
Spiral Testing	638
Task 1: Determine Appropriateness of RAD	639
Task 2: Test Planning Iterations	640
Task 3: Test Subsequent Planning Iterations	640
Task 4: Test the Final Planning Iteration	642
Check Procedures	642
Output	643
Guidelines	643
Summary	643
Chapter 17 Testing Internal Controls	655
Overview	655
Internal Controls	657
Control Objectives	657
Preventive Controls	658
Source-Data Authorization	658
Data Input	659
Source-Data Preparation	659
Turnaround Documents	659
Prenumbered Forms	659
Input Validation	659
File Auto-Updating	661
Processing Controls	661

Detective Controls	662
Data Transmission	663
Control Register	663
Control Totals	664
Documenting and Testing	664
Output Checks	664
Corrective Controls	665
Error Detection and Resubmission	665
Audit Trails	665
Cost/Benefit Analysis	666
Assessing Internal Controls	666
Task 1: Understand the System Being Tested	666
Task 2: Identify Risks	668
Task 3: Review Application Controls	668
Task 4: Test Application Controls	668
Testing Without Computer Processing	669
Testing with Computer Processing	669
Transaction Flow Testing	672
Objectives of Internal Accounting Controls	673
Results of Testing	677
Task 5: Document Control Strengths and Weaknesses	677
Quality Control Checklist	678
Summary	678
Chapter 18 Testing COTS and Contracted Software	685
Overview	686
COTS Software Advantages, Disadvantages, and Risks	686
COTS Versus Contracted Software	686
COTS Advantages	687
COTS Disadvantages	687
Implementation Risks	688
Testing COTS Software	689
Testing Contracted Software	690
Objective	691
Concerns	691
Workbench	692
Input	693
Do Procedures	693
Task 1: Test Business Fit	693
Step 1: Testing Needs Specification	693
Step 2: Testing CSFs	695
Task 2: Test Operational Fit	696
Step 1: Test Compatibility	697
Step 2: Integrate the Software into Existing Work Flows	698
Step 3: Demonstrate the Software in Action	700
Task 3: Test People Fit	701

Task 4: Acceptance-Test the Software Process	702
Step 1: Create Functional Test Conditions	702
Step 2: Create Structural Test Conditions	703
Modifying the Testing Process for Contracted Software	704
Check Procedures	705
Output	705
Guidelines	706
Summary	706
Chapter 19 Testing in a Multiplatform Environment	717
Overview	717
Objective	718
Concerns	718
Background on Testing in a Multiplatform Environment	718
Workbench	719
Input	720
Do Procedures	721
Task 1: Define Platform Configuration Concerns	721
Task 2: List Needed Platform Configurations	723
Task 3: Assess Test Room Configurations	723
Task 4: List Structural Components Affected by the Platform(s)	723
Task 5: List Interfaces the Platform Affects	725
Task 6: Execute the Tests	726
Check Procedures	726
Output	726
Guidelines	726
Summary	727
Chapter 20 Testing Software System Security	733
Overview	733
Objective	734
Concerns	734
Workbench	734
Input	735
Where Vulnerabilities Occur	735
Functional Vulnerabilities	736
Vulnerable Areas	737
Accidental Versus Intentional Losses	738
Do Procedures	739
Task 1: Establish a Security Baseline	739
Why Baselines Are Necessary	740
Creating Baselines	740
Using Baselines	749
Task 2: Build a Penetration-Point Matrix	751
Controlling People by Controlling Activities	751
Selecting Security Activities	752
Controlling Business Transactions	755

Characteristics of Security Penetration	756
Building a Penetration-Point Matrix	757
Task 3: Analyze the Results of Security Testing	760
Evaluating the Adequacy of Security	761
Check Procedures	762
Output	762
Guidelines	762
Summary	762
Chapter 21 Testing a Data Warehouse	765
Overview	765
Concerns	765
Workbench	766
Input	767
Do Procedures	768
Task 1: Measure the Magnitude of Data Warehouse Concerns	768
Task 2: Identify Data Warehouse Activity Processes to Test	769
Organizational Process	769
Data Documentation Process	769
System Development Process	770
Access Control Process	771
Data Integrity Process	771
Operations Process	772
Backup/Recovery Process	773
Performing Task 2	774
Task 3: Test the Adequacy of Data Warehouse Activity	
Processes	774
Check Procedures	780
Output	780
Guidelines	780
Summary	780
Chapter 22 Testing Web-Based Systems	799
Overview	799
Concerns	800
Workbench	800
Input	801
Do Procedures	802
Task 1: Select Web-Based Risks to Include in the Test Plan	802
Security Concerns	803
Performance Concerns	803
Correctness Concerns	804
Compatibility Concerns	804
Reliability Concerns	806
Data Integrity Concerns	806
Usability Concerns	806
Recoverability Concerns	807

Task 2: Select Web-Based Tests	807
Unit or Component	807
Integration	807
System	807
User Acceptance	808
Performance	808
Load/Stress	808
Regression	808
Usability	808
Compatibility	808
Task 3: Select Web-Based Test Tools	809
Task 4: Test Web-Based Systems	809
Check Procedures	809
Output	810
Guidelines	810
Summary	811
Part V Building Agility into the Testing Process	817
Chapter 23 Using Agile Methods to Improve Software Testing	819
The Importance of Agility	819
Building an Agile Testing Process	820
Agility Inhibitors	821
Is Improvement Necessary?	822
Compressing Time	823
Challenges	824
Solutions	825
Measuring Readiness	826
The Seven-Step Process	826
Summary	827
Chapter 24 Building Agility into the Testing Process	831
Step 1: Measure Software Process Variability	831
Timelines	832
Process Steps	833
Workbenches	833
Time-Compression Workbenches	834
Reducing Variability	835
Developing Timelines	836
Improvement Shopping List	841
Quality Control Checklist	841
Conclusion	842
Step 2: Maximize Best Practices	842
Tester Agility	842
Software Testing Relationships	843
Tradeoffs	845
Capability Chart	847
Measuring Effectiveness and Efficiency	848

Improvement Shopping List	856
Quality Control Checklist	856
Conclusion	857
Step 3: Build on Strength, Minimize Weakness	857
Effective Testing Processes	857
Poor Testing Processes	860
Improvement Shopping List	860
Quality Control Checklist	860
Conclusion	861
Step 4: Identify and Address Improvement Barriers	861
The Stakeholder Perspective	861
Stakeholder Involvement	863
Performing Stakeholder Analysis	863
Red-Flag/Hot-Button Barriers	864
Staff-Competency Barriers	865
Administrative/Organizational Barriers	865
Determining the Root Cause of Barriers/Obstacles	866
Addressing the Root Cause of Barriers/Obstacles	867
Quality Control Checklist	869
Conclusion	869
Step 5: Identify and Address Cultural and Communication Barriers	869
Management Cultures	870
Culture 1: Manage People	871
Culture 2: Manage by Process	873
Culture 3: Manage Competencies	874
Culture 4: Manage by Fact	876
Culture 5: Manage Business Innovation	878
Cultural Barriers	879
Identifying the Current Management Culture	879
Identifying the Barriers Posed by the Culture	879
Determining What Can Be Done in the Current Culture	879
Determining the Desired Culture for Time Compression	879
Determining How to Address Culture Barriers	880
Open and Effective Communication	880
Lines of Communication	881
Information/Communication Barriers	882
Effective Communication	882
Quality Control Checklist	884
Conclusion	885
Step 6: Identify Implementable Improvements	885
What Is an Implementable?	885
Identifying Implementables via Time Compression	886
Prioritizing Implementables	888
Documenting Approaches	890
Quality Control Checklist	890
Conclusion	890

Step 7: Develop and Execute an Implementation Plan	891
Planning	891
Implementing Ideas	891
Requisite Resources	893
Quality Control Checklist	894
Conclusion	894
Summary	895

Index	929
--------------	------------



Introduction

Most books about software testing explain “what” to do. This book, on the other hand, takes more of a “how-to” approach. It provides the procedures, templates, checklists, and assessment questionnaires necessary to conduct effective and efficient software testing.

The book is divided into five parts, as follows:

- **Part One: Assessing Testing Capabilities and Competencies.** It is difficult to make any significant change until you know where you are. A baseline tells not only where you are, but lets you measure your progress as your testing strategies and techniques improve. Part One provides three baseline assessments: the capabilities of your software testing group, the competencies of your individual testers, and the effectiveness of your test processes.
- **Part Two: Building a Software Testing Environment.** Software testers are most effective when they work in an environment that encourages and supports well-established testing policies and procedures. The environment includes the procedures and tools for testing, as well as the support and encouragement of management. Part Two begins by describing how to build an environment conducive to testing, and then expands the discussion by describing how to develop a testing process, select testing tools, and build the competency of your testers.
- **Part Three: The Seven-Step Testing Process.** Part Three comprises the core material in the book. It defines a world-class software testing process, from its initiation through testing changes made to operational software systems. This material can be used two ways. First, it contains sufficient procedures and templates so that an organization can use the process as their own. Of course, most organizations inevitably will make some changes to accommodate local vocabulary, specific needs, and customs. This customization process, the seven-step process in this book becomes “owned” by the software testers.

- **Part Four: Incorporating Specialized Testing Responsibilities.** The seven-step testing process is a generic process that almost all software testing organizations can use. However, the mission of software testers may incorporate specialized activities, such as testing security. Rather than incorporating these specialized testing activities directly into the seven-step process, they are presented as individual, specialized activities. As appropriate, they can be incorporated into the seven-step process.
- **Part Five: Building Agility into the Testing Process.** Part Five, which draws on what you've learned earlier in the book, is designed to help you identify the strengths and weaknesses of your current software testing process, and then modify it to become more usable or agile.

Getting the Most Out of This Book

This book is not designed to be read like a novel, from beginning to end, nor is it filled with human interest stories about testers. The book focuses on how to conduct software testing. It is designed to help you improve your testing competencies and processes. The self-assessments in Part One will help you identify which parts of the book you need to read first.

The following guidelines will help you maximize the benefit from this book:

- **Establish a baseline of current performance.** Part One of this book (and Chapter 5) contains four self-assessments for establishing baselines. You need to know where you are so that you can develop a good plan for moving forward.
- **Define the software testing organization you would like to have.** It has been said that if you do not know where you're going, all roads lead there. Too many software testing groups just add new testing programs, processes, and tools without knowing if they will integrate effectively.
- **Develop a plan for moving from your baseline to your goal.** Few organizations can quickly and effectively install an entirely new software testing process. Gradual change is normally much better than radical change. Therefore, identify the gaps between where you are and where you want to be. Determine which of those gaps if closed would provide the greatest benefit to your organization. That becomes the part of the plan you implement first. Over time you will move the entire testing process from your current baseline to your desired goal.

For additional information on software testing conferences and training programs, visit www.taiworldwide.org. For information on software testing certifications, visit www.softwarecertifications.org.

What's New in the Third Edition

The core of this book is the step-by-step process for testing software. This edition has simplified that process from 11 steps to 7 steps.

A major addition to this edition is the self-assessment in Chapter 5, which testers can use to identify their strengths and weaknesses and then build a personal improvement plan. The self-assessment is based on the Common Body of Knowledge (CBOK) for the Certified Software Tester (CSTE).

Other significant additions include

- A new chapter on testing internal control
- An expanded chapter on testing security
- A new chapter on adapting testing to the developmental methodology used to build the software
- Two new chapters on how to incorporate agile methods into the testing process

What's on the CD

This book includes a CD that contains the work papers and quality control checklists to help you implement the software testing process.

To use the CD, first you need to select a software testing activity that you want to implement in your organization—for example, test planning. Then, from the chapter on test planning, identify those work papers and checklists that you believe would be beneficial to your organization. You can extract those work papers and checklists from the CD and begin a customization process. For example, you can include the name of your organization, add or delete portions of the work papers, and change the terminology to be consistent with your organization.

After you have used the work papers for conducting a software test, you should bundle the work papers into a case study for new testers. If they use the book to learn the basics of software testing and then can cross reference what they have learned to examples of how the work papers are actually used in software testing, learning should be accelerated.

