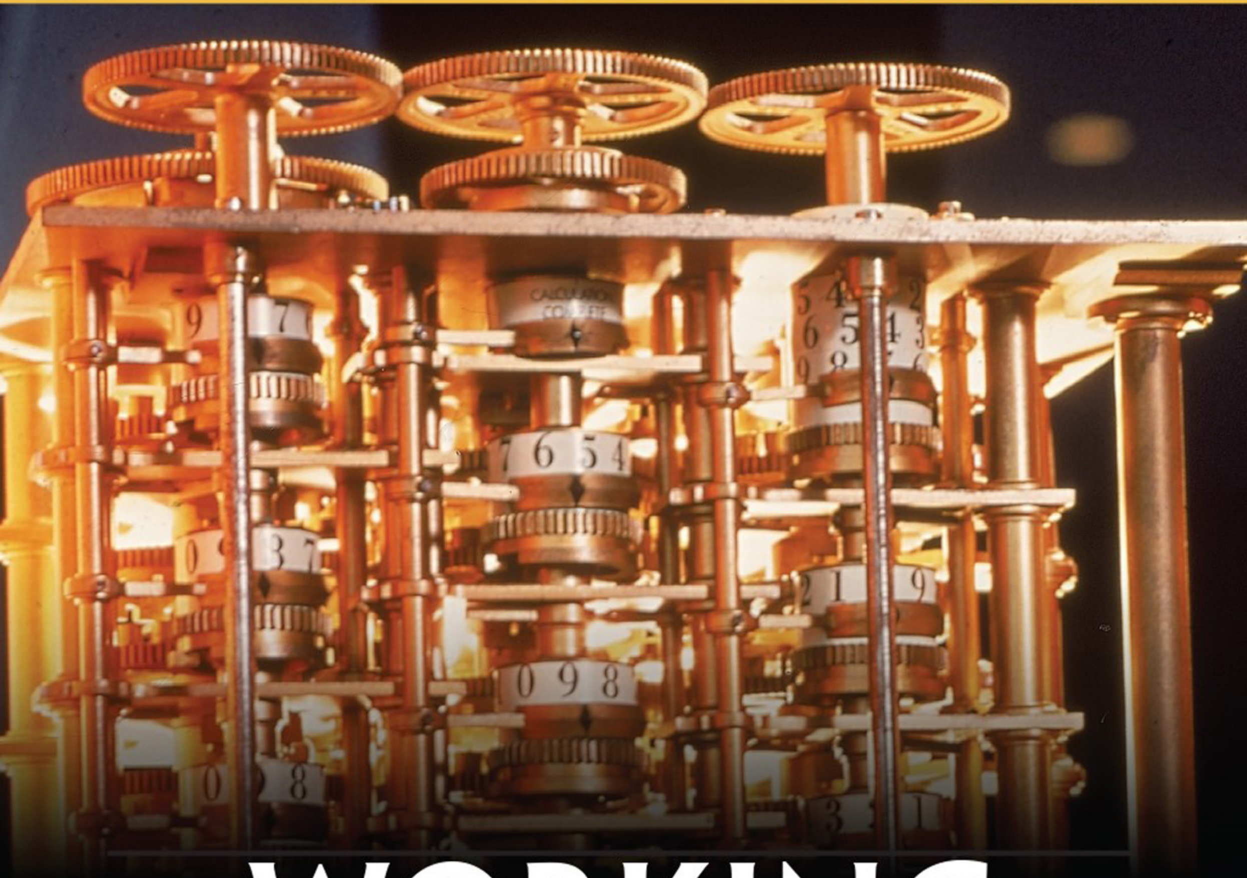
Robert C. Martin Series



WORKING EFFECTIVELY WITH LEGACY CODE

Michael C. Feathers

Working Effectively with Legacy Code

Working Effectively with Legacy Code

Table of Contents

Cover

Half Title

Series Page

Title Page

Copyright Page

Contents

Foreword

Preface

Introduction

PART I: The Mechanics of Change

- Chapter 1: Changing Software
  - Four Reasons to Change Software
  - Risky Change
- Chapter 2: Working with Feedback
  - What Is Unit Testing?
  - Higher-Level Testing
  - Test Coverings
  - The Legacy Code Change Algorithm
- Chapter 3: Sensing and Separation
  - Faking Collaborators
- Chapter 4: The Seam Model
  - A Huge Sheet of Text
  - Seams

Table of Contents

Seam Types

Chapter 5: Tools

Automated Refactoring Tools

Mock Objects

Unit-Testing Harnesses

General Test Harnesses

PART II: Changing Software

Chapter 6: I Dont Have Much Time and I Have to Change It

Sprout Method

Sprout Class

Wrap Method

Wrap Class

Summary

Chapter 7: It Takes Forever to Make a Change

Understanding

Lag Time

Breaking Dependencies

Summary

Chapter 8: How Do I Add a Feature?

Test-Driven Development (TDD)

Programming by Difference

Summary

Chapter 9: I Cant Get This Class into a Test Harness

The Case of the Irritating Parameter

The Case of the Hidden Dependency

The Case of the Construction Blob

The Case of the Irritating Global Dependency

The Case of the Horrible Include Dependencies

The Case of the Onion Parameter

Table of Contents

- The Case of the Aliased Parameter
- Chapter 10: I Cant Run This Method in a Test Harness
  - The Case of the Hidden Method
  - The Case of the Helpful Language Feature
  - The Case of the Undetectable Side Effect
- Chapter 11: I Need to Make a Change. What Methods Should I Test?
  - Reasoning About Effects
  - Reasoning Forward
  - Effect Propagation
  - Tools for Effect Reasoning
  - Learning from Effect Analysis
  - Simplifying Effect Sketches
- Chapter 12: I Need to Make Many Changes in One Area
  - Interception Points
  - Judging Design with Pinch Points
  - Pinch Point Traps
- Chapter 13: I Need to Make a Change, but I Dont Know What Tests to Write
  - Characterization Tests
  - Characterizing Classes
  - Targeted Testing
  - A Heuristic for Writing Characterization Tests
- Chapter 14: Dependencies on Libraries Are Killing Me
- Chapter 15: My Application Is All API Calls
- Chapter 16: I Dont Understand the Code Well Enough to Change It
  - Notes/Sketching
  - Listing Markup

Table of Contents

Scratch Refactoring

Delete Unused Code

Chapter 17: My Application Has No Structure

Telling the Story of the System

Naked CRC

Conversation Scrutiny

Chapter 18: My Test Code Is in the Way

Class Naming Conventions

Test Location

Chapter 19: My Project Is Not Object Oriented How Do I Make Safe Changes?

An Easy Case

A Hard Case

Adding New Behavior

Taking Advantage of Object Orientation

Its All Object Oriented

Chapter 20: This Class Is Too Big and I Dont Want It to Get Any Bigger

Seeing Responsibilities

Other Techniques

Moving Forward

After Extract Class

Chapter 21: Im Changing the Same Code All Over the Place

First Steps

Chapter 22: I Need to Change a Monster Method and I Cant Write Tests for It

Varieties of Monsters

Tackling Monsters with Automated Refactoring Support

The Manual Refactoring Challenge

Table of Contents

Strategy

Chapter 23: How Do I Know That Im Not Breaking Anything?

Hyperaware Editing

Single-Goal Editing

Preserve Signatures

Lean on the Compiler

Chapter 24: We Feel Overwhelmed. It Isnt Going to Get Any Better

PART III: Dependency-Breaking Techniques

Chapter 25: Dependency-Breaking Techniques

Adapt Parameter

Break Out Method Object

Definition Completion

Encapsulate Global References

Expose Static Method

Extract and Override Call

Extract and Override Factory Method

Extract and Override Getter

Extract Implementer

Extract Interface

Introduce Instance Delegator

Introduce Static Setter

Link Substitution

Parameterize Constructor

Parameterize Method

Primitivize Parameter

Pull Up Feature

Push Down Dependency

Replace Function with Function Pointer

Replace Global Reference with Getter

Table of Contents

Subclass and Override Method

Supersede Instance Variable

Template Redefinition

Text Redefinition

Appendix: Refactoring

Extract Method

Glossary

Index

Pearson