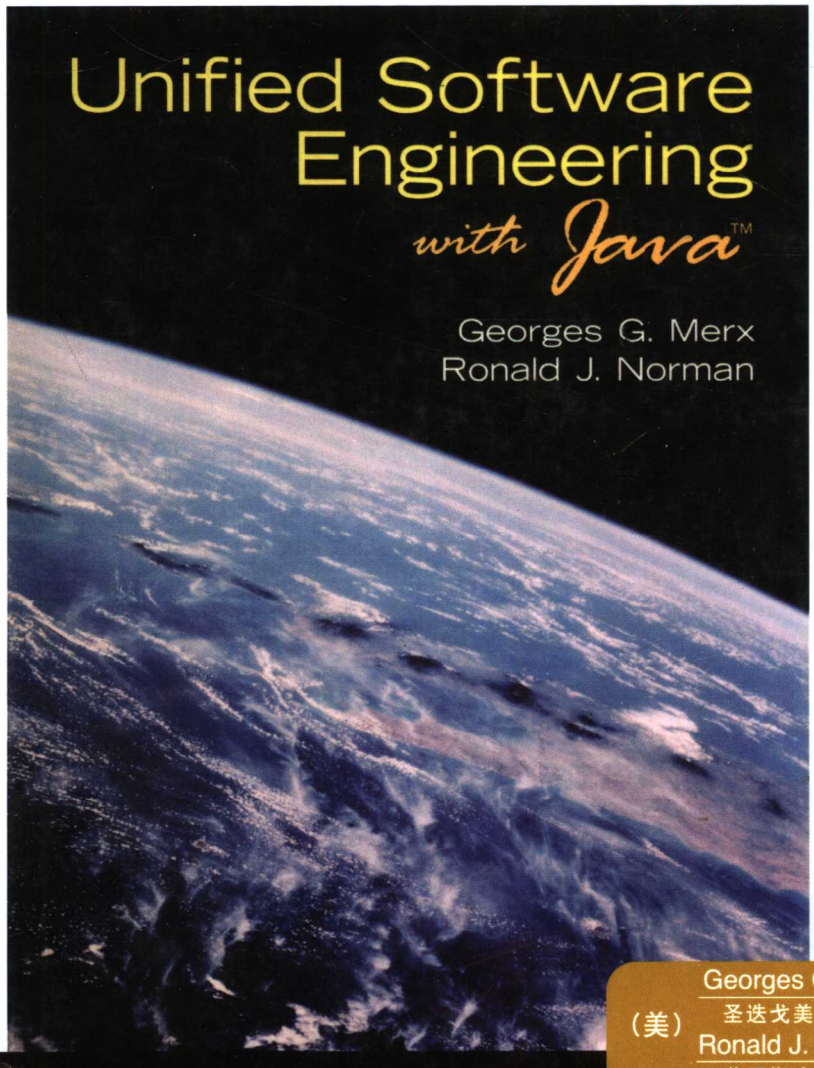
HZ BOOKS 华章教育
经 典 原 版 书 库
PRENTICE HALL

统一软件工程

（英文版）



Georges G. Merx
（美） 圣迭戈美萨学院 著
Ronald J. Norman
葛罗斯摩特学院

机械工业出版社
China Machine Press

经 典 原 版 书 库

统一软件工程

（英文版）

Unified Software Engineering with Java

TP312/X289
2008
江苏工业学院图书馆
藏书章

（美）Georges G. Merx 圣迭戈美萨学院
Ronald J. Norman 葛罗斯摩特学院 著

机械工业出版社
China Machine Press

English reprint edition copyright © 2008 by Pearson Education Asia Limited and China Machine Press.

Original English language title: *Unified Software Engineering with Java* (ISBN 0-13-047376-6) by Georges G. Merx and Ronald J. Norman, Copyright © 2007.

All rights reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as Prentice Hall.

For sale and distribution in the People's Republic of China exclusively (except Taiwan, Hong Kong SAR and Macau SAR).

本书英文影印版由Pearson Education Asia Ltd.授权机械工业出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

仅限于中华人民共和国境内（不包括中国香港、澳门特别行政区和中国台湾地区）销售发行。

本书封面贴有Pearson Education（培生教育出版集团）激光防伪标签，无标签者不得销售。

版权所有，侵权必究。

本书法律顾问　北京市展达律师事务所

本书版权登记号：图字： 01-2007-4208

图书在版编目（CIP）数据

统一软件工程（英文版）/（美）默克斯（Merx, G. G.）等著. —北京：机械工业出版社，2008.1
（经典原版书库）
书名原文：Unified Software Engineering with Java

ISBN 978-7-111-23164-6

Ⅰ. 统…　Ⅱ. 默…　Ⅲ. 软件开发—英文　Ⅳ. TP311.52

中国版本图书馆CIP数据核字（2007）第205697号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）
责任编辑：迟振春
北京瑞德印刷有限公司印刷·新华书店北京发行所发行
2008年1月第1版第1次印刷
170mm × 242mm　·　39.5印张
标准书号：ISBN 978-7-111-23164-6
定价：69.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线：（010）68326294

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭橥了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短、从业人员较少的现状下，美国等发达国家在其计算机科学发展的几十年间积淀的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出Tanenbaum，Stroustrup，Kernighan，Jim Gray等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及庋藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近260个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都步入一个新的阶段。为此，华章公司将加大引进教材的力度，除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”。为了保证这两套丛书的权威性，同时也为了更好地为学校和老师们服务，华章公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空

航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这两套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经三十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证，但我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。教材的出版只是我们的后续服务的起点。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

电子邮件：hzjsj@hzbook.com
联系电话：(010) 68995264
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

试读结束：需要全本请在线购买： www.ertongbook.com

专家指导委员会

（按姓氏笔画顺序）

尤晋元	王　珊	冯博琴	史忠植	史美林
石教英	吕　建	孙玉芳	吴世忠	吴时霖
张立昂	李伟琴	李师贤	李建中	杨冬青
邵维忠	陆丽娜	陆鑫达	陈向群	周伯生
周克定	周傲英	孟小峰	岳丽华	范　明
郑国梁	施伯乐	钟玉琢	唐世渭	袁崇义
高传善	梅　宏	程　旭	程时端	谢希仁
裘宗燕	戴　葵			

My wife Jin always supports and encourages my projects—even one as major as writing this book—although they take away from our time together. I am deeply grateful for this irrefutable evidence of true love.

I hope that my efforts will serve to provide some inspiration for my daughter London to find her own path to success, wherever that path may lead her.

A good portion of this work was completed at the Fenton Place Starbucks in San Diego. I express my gratitude to the staff there, and to Starbucks as one of the great American companies, for creating an environment where those of us with short attention spans can be productive.

All the people at Prentice Hall—especially our editor, Tracy Dunkelberger, and her indefatiguable assistant, Christianna Lee—with whom I have had the privilege to work are immensely supportive, helpful, and competent: my appreciation is heartfelt. A special note of appreciation to Irwin Zucker, our production editor, for getting us through the production phase of this project with patience and much good will.

Finally, the indefatigable encouragement and contributions of my co-author, Ron, have been indispensable ingredients to making this project workable.

Georges G. Merx

There are so many to dedicate this book to that I cannot name them individually. Literally hundreds of software engineers (generic title) and academics/researchers around the globe have contributed to making me the professional I am today through the many publications, conferences, seminars, and workshops that I have either physically or virtually attended or led. Their influence has been profound in my life, and I am deeply grateful for the experiences and interaction with each of these professional women and men.

I also dedicate this book to those who will advance their academic and/or professional knowledge through the use of this book. It is truly a privilege to contribute something back to you, since so many individuals have profoundly influenced me.

Thank you, Georges, for allowing me to take this book journey with you—you are a gifted writer and seasoned professional/academic.

Finally, thank you to my life-partner, wife, and best friend: Caralie.

Ronald J. Norman

Preface

Creating commercial software requires excellent knowledge and skills in a number of areas, not just programming language syntax and semantics. We have therefore written a book that teaches the fundamentals of Java programming in the context of object-oriented software engineering and a Unified-Process-based software development methodology. Today's programmers need to be software engineers who knw their languages and tools, certainly, but who also understand object-oriented analysis and design, software quality asurance, and software project management. In fact, the best antidote to the *outsourcing* of software development jobs overseas is to elevate the profession above the specialist tasks of code development. Software engineers need to have the skills to deliver quality software on time, on budget, and according to stakeholder requirements. Our book puts the study of Java in this meaningful, valuable context.

Audience

College students with a previous course in programming or software engineering learning their first or second computer programming language are the primary audience for this textbook. Some previous exposure to principles of information systems and computer science is desirable, but not required. Other likely readers are software development professionals who are looking for a methodical approach to learning object-oriented software development using Java, especially those who only have experience in procedural programming.

Course Definition

This book is recommended for use in information systems or computer science courses at the college level and targets students pursuing an interest in computer science, information systems, or software engineering. In addition to delivering solid programming language instruction, it lays a broad foundation in object-oriented methodology, based on best practices and proven principles developed by Grady Booch, Jim Rumbaugh, Ivar Jacobson, Peter Coad, Barry Boehm, Kent Beck, and other recognized software engineering thought leaders. Based on a complete, object-oriented life-cycle view of the software design and development process, software engineering as defined

and described in this book embraces the use of Java for the development of robust, commercially viable, and eminently usable software solutions.

From initial concept to deployment, all aspects of software engineering project design, development, and management will accompany the students' learning experience. They will understand how rigorous iteration-based requirements management (using stakeholder and use case analysis), conceptual and physical design (using the Unified Modeling Language and Design Patterns), component-based implementation, and well-planned deployment contribute to transitioning software development from an art form to an engineering discipline.

For professors and instructors, this book and accompanying website constitute solid teaching aids, providing not only Java language training, but also work process-based instruction, including a clear and practical introduction to object-oriented design and development. Written with the understanding that the introduction to software engineering and Java can be a daunting experience for many inexperienced readers, this book delivers its instructional content with a strong emphasis on illustrative examples and a firm grounding in real-life applications.

Courses on Java and object-oriented programming are mainstay offerings on many college campuses. This book seeks to support and deepen the interest of students, teachers and administrators in an area of computer science critical to the development of core skills sought after by the high-technology industry. Courses built to leverage the contents of this book will help students advance their understanding of object-oriented software engineering using Java to a level where they can either move on to more advanced course work, or apply their new, practical knowledge in entry-level work positions.

Another Java Book?

The idea for this book arose originally from my search (Merx) for appropriate textbooks for use in my own Java courses. It appeared that available textbooks either focus on Java syntax and structure at the expense of methodology and process, or emphasize "analysis and design," while lacking the practical context of a modern object-oriented programming language and toolset. Both of us have extensive experience both in academia (SDSU, UCSD, Mesa College, University of Phoenix, National University, Grossmont College, and University of Maryland University College) and in business (Borland, TogetherSoft, NCR, AT&T, QUALCOMM, ICL/Fujitsu, etc.). This background convinces us that as educators we need to do better in training our students for the multi-disciplinary effort required to develop valuable, high-quality software that solves difficult problems in a world-class fashion.

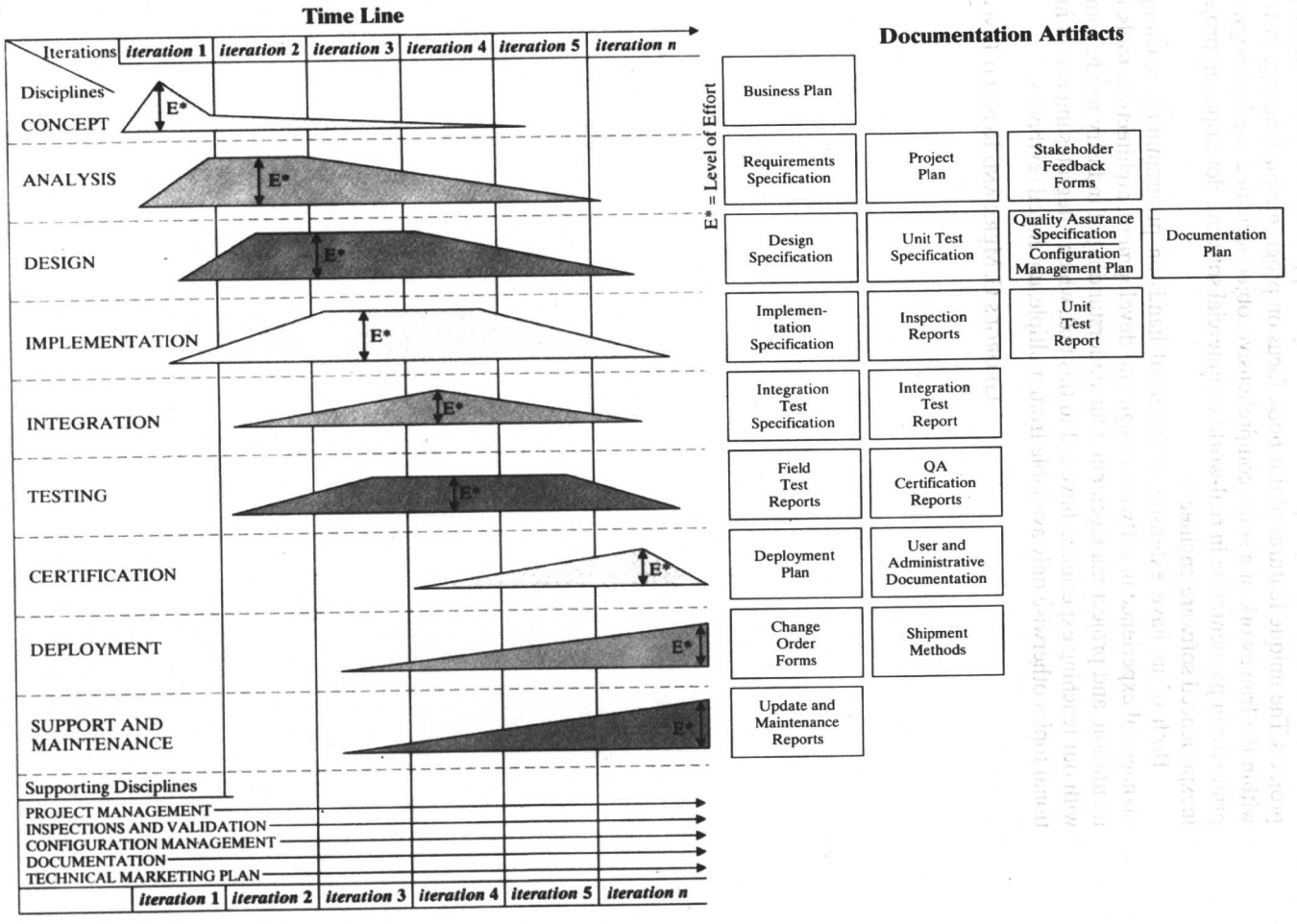
While many excellent text and reference books on Java are available, most are lacking key features deemed essential for practical instruction of effective software engineering using Java. From our perspective, a key area of progress in software development is the recasting of professional programming as a *software engineering discipline*, with its implications of reusability, quality controlled rigor, focus on architecture and process, and project management. Software engineering, as described and standardized in the Carnegie-Mellon University Software Engineering Institute's Capability

Maturity Model, for example, promotes a life-cycle approach to software development projects. The unique features of this book focus on programming language instruction within the framework of a solid, comprehensive, object-oriented methodology, appropriate for implementation in real-world commercial software development projects by inexperienced software engineers.

Both of us have extensive commercial hands-on programming, teaching, and managerial experience in software design and development, architecture, tools, implementation, and project management. Our understanding of industry needs, combined with our teaching experience, have led to this text, which integrates important instructional topics otherwise only available from multiple, unrelated textbooks.

GEORGES G. MERX AND RONALD J. NORMAN

Extended Unified Process Model



试读结束：需要全本请在线购买：www.ertongbook.com

Contents

Preface vii

1 Introduction to Java in the Context of Software Engineering 1

1.1 Getting Acquainted 1
1.2 What Is Java Programming? 1
1.2.1 What is Software Engineering? 2
1.3 Learning Objectives 6
1.3.1 Learning Layout 7
1.3.2 Learning Connections 8
1.4 Executive Summary 9
1.5 Learning Modules 10
1.5.1 Concepts 10
1.5.2 Unified Process–Based Methodology Overview 12
1.5.3 Position in Process 14
1.5.4 Domain Model 17
1.5.5 Scenarios 17
1.5.6 The Unified Modeling Language 21
1.6 The Java Programming Language 67
1.6.1 Historical Perspective on Java 21
1.6.2 Java Basics 23
1.7 Relationships 28
1.7.1 Caveats and Complexities 30
1.8 Example: The Voting Program 30
1.8.1 Project (System) Vision 31
1.8.2 Project Description 31
1.8.3 Stakeholder Analysis 32
1.8.4 Customer Profile 32
1.8.5 Market Analysis 32
1.8.6 Risk Analysis 33
1.8.7 Business Use Case Model and Use Cases 33
1.8.8 Competitive Analysis 36
1.8.9 Distribution Plan 41

1.8.10 Financial Plan 42
1.8.11 High-Level Project Plan 43
1.8.12 Recommendations 43
1.9 Ongoing Case Study 43
1.9.1 Introduction 43
1.9.2 Initial Concept 44
1.9.3 Business Justification 45
1.9.4 Stakeholder Analysis 46
1.9.5 Case Assignments 46
1.10 Resources: Connections • People • Companies 46
1.11 Summary and Recommendations 47
1.12 Review Questions 47
1.13 Glossary – Terminology – Concepts 48
1.14 Exercises 50
1.15 Setting up a Java Development Environment 50
1.15.1 Versions of Java 51
1.15.2 Class and Classpath Setup 51
1.16 Java Programming Exercises 51

2 Experimenting with Classes and Objects 52

2.1 Learning Objectives 54
2.1.1 Learning Layout 55
2.1.2 Learning Connections 56
2.2 Executive Summary 57
2.3 Learning Modules 58
2.3.1 Concepts 58
2.3.2 Position in Process 73
2.4 The Purpose of Object Orientation in Software Engineering 74
2.5 Problems with Procedural Programming 75
2.6 How O-O Solves Software Development Problems 76
2.7 Understanding Object Orientation 77
2.8 Object-Orientation in Java 79
2.8.1 Java Classes and Objects 79
2.9 Architecture and Class Hierarchy 82
2.10 Economies of Reuse 82
2.10.1 Quality 83
2.10.2 Consistency 83
2.10.3 Implement Once 83
2.10.4 Flexibility 83
2.11 Use Case Models and Classes 84
2.12 "Real-Life" Variations 85
2.13 Translating Generic Class Descriptions into Java Classes 85

2.14 Unified Modeling Language Perspective 86
2.15 A Simple Java Program: The Voting Program Prototype 86
2.16 Relationships 87
2.16.1 Caveats and Complexities 87
2.17 Example: The Voting Program 88
2.17.1 The Domain Model 88
2.17.2 Requirements Specification Outline 89
2.17.3 Deliverables 103
2.17.4 Other Requirements 103
2.18 Ongoing Case Study 104
2.18.1 Market Analysis 104
2.18.2 Risk Management 104
2.18.3 Business Use Case Model and Business Use Cases 105
2.18.4 Competitive Analysis 105
2.18.5 Distribution Plan (Pricing, Packaging, Promotion, Positioning) 105
2.18.6 Financial Plan (Revenue Plan, Budget, Cash Flow Analysis, ROI Analysis) 106
2.18.7 High-Level Project Plan 106
2.18.8 Recommendations 106
2.18.9 Case Assignments 106
2.19 Resources: Connections • People • Companies 107
2.20 Summary and Recommendations 107
2.21 Review Questions 107
2.22 Glossary – Terminology – Concepts 108
2.23 Exercises 108

3 The Structure and Syntax of Java 109

3.1 Learning Objectives 111
3.1.1 Learning Layout 111
3.1.2 Learning Connections 111
3.2 Executive Summary 112
3.3 Learning a Programming Language 114
3.3.1 For the Novice 114
3.3.2 For the Experienced Software Engineer 119
3.3.3 Similarities to Other O-O Programming Languages 120
3.4 Learning Modules 122
3.4.1 Concepts 122
3.5 The Java Family of Classes and Packages 143
3.6 Third-Party Components 144
3.7 Software Quality Assurance 144

3.8 Position in Process 146
3.8.1 Design Model 147
3.8.2 Component Design 147
3.8.3 Class Hierarchy 147
3.8.4 System Architecture 148
3.8.5 Prototyping 150
3.9 Relationships 150
3.9.1 Caveats and Complexities 152
3.10 Example: The Voting Program 152
3.10.1 Component Design 152
3.10.2 Class Hierarchy 153
3.10.3 System Architecture 153
3.10.4 Prototype 153
3.10.5 Design Specification, Quality Assurance Specification, and Configuration Management Plan 153
3.11 Ongoing Case Study 155
3.12 Resources: Connections • People • Companies 158
3.13 Summary and Recommendations 158
3.14 Review Questions 159
3.15 Glossary – Terminology – Concepts 159
3.16 Exercises 159
3.17 Optional Chapter Addendum: LibraryManager, an Application Example 160
3.17.1 Program Code 161
3.17.2 Analysis 179

4 Design and Development of Java Applications 183

4.1 Learning Objectives 185
4.1.1 Software Engineering Methodology 186
4.1.2 Java Syntax and Structure 187
4.1.3 Object Orientation 190
4.1.4 Software Quality Assurance 191
4.1.5 Attitude and Motivation 192
4.1.6 Learning Layout 193
4.1.7 Learning Connections 193
4.2 Executive Summary 193
4.3 Learning Modules 196
4.3.1 Software Engineering History 196
4.3.2 Process Models 205
4.3.3 "Object Thinking" 209
4.3.4 Object Orientation 210
4.3.5 Java Control Structures 214

4.4 Position in Process 227
4.4.1 Design Specification 227
4.4.2 Unit Test Specification 229
4.4.3 Quality Assurance Plan 229
4.4.4 Configuration Management Plan 230
4.4.5 Documentation Plan 230
4.5 Example: The Voting Program 231
4.5.1 Introduction 231
4.5.2 Functional Overview 234
4.5.3 System Architecture 235
4.5.4 Class Hierarchy 236
4.5.5 Component Definition and Design 238
4.5.6 Prototype Description and Evaluation 238
4.5.7 Environment 239
4.5.8 Supporting Disciplines 240
4.6 Ongoing Case Study 245
4.7 Resources: Connections • People • Companies 246
4.8 Summary and Recommendations 247
4.9 Review Questions 247
4.10 Glossary – Terminology – Concepts 247
4.11 Exercises 248
4.12 Optional Chapter Addendum: Pattern-Driven Design 248
4.12.1 Pattern Principle 1: High Cohesion 248
4.12.2 Pattern Principle 2: Low Coupling 249
4.12.3 Most Popular Patterns 249

5 Architecture-Driven Component Development 251

5.1 Learning Objectives 251
5.1.1 Revisiting System and Software Architecture 252
5.1.2 Java Component Interaction and Integration 255
5.1.3 Learning Layout 260
5.1.4 Learning Connections 260
5.2 Executive Summary 260
5.3 Learning Modules 262
5.3.1 Concepts 263
5.3.2 Architectural Perspectives 264
5.3.3 Developing Java Components 269
5.3.4 Java Class Interaction 276
5.3.5 Java Objects 278
5.3.6 Methods and Constructors 278
5.3.7 Polymorphism: Method Overloading 279

5.3.8 Polymorphism: Method Overriding 280
5.3.9 Inheritance: Extending Classes 280
5.3.10 Inheritance: Implementing Interfaces 280
5.3.11 User Interface: An Introduction 281
5.3.12 User Input and User Input Validation 282
5.4 Position in Process 283
5.4.1 Component Implementation 284
5.4.2 Unit Testing 284
5.4.3 Build Management 285
5.5 Example: The Voting Program 287
5.5.1 Components 288
5.6 Ongoing Case Study 310
5.6.1 Some Notes on the Model Home Interior Design Business 311
5.7 Resources: Connections • People • Companies 312
5.8 Summary and Recommendations 312
5.9 Review Questions 312
5.10 Glossary – Terminology – Concepts 313
5.11 Exercises 313

6 Introduction to Distributed Computing Concepts 314

6.1 Learning Objectives 315
6.1.1 Creating Value 315
6.1.2 Agile Techniques 318
6.1.3 Learning Layout 319
6.1.4 Learning Connections 319
6.2 Executive Summary 320
6.3 Learning Modules 322
6.3.1 Concepts 322
6.3.2 Agile Methods and Rapid Application Development 324
6.3.3 Distributed Java Applications 325
6.3.4 Methodology, Tools, and Distributed Solutions 327
6.3.5 Information Persistence 331
6.4 Position in Process 333
6.4.1 Class and Object Integration 333
6.4.2 Package Integration 334
6.4.3 Subsystem Integration 335
6.4.4 System Integration 335
6.4.5 Integration Testing 335
6.5 Iterative Improvements 337

试读结束：需要全本请在线购买： www.ertongbook.com