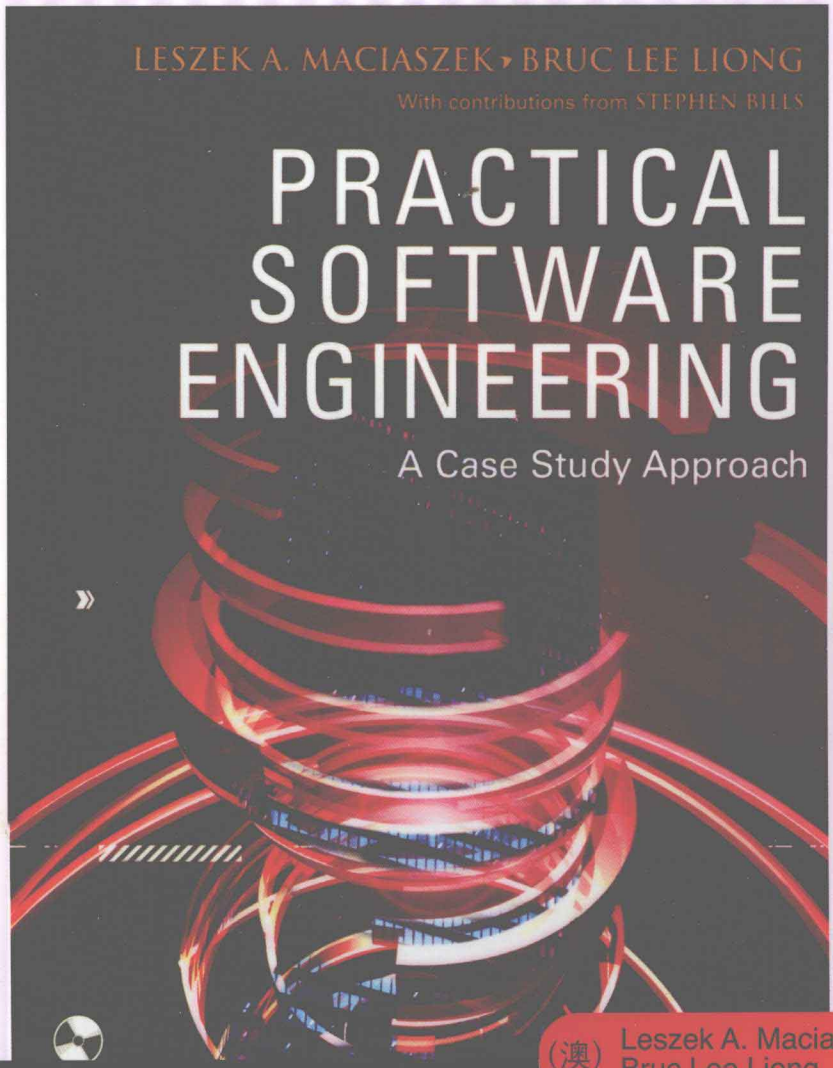
经 典 原 版 书 库

实用软件工程

（英文版）



附赠 CD-ROM

（澳）Leszek A. Maciaszek
Bruc Lee Liong 等著

机械工业出版社
China Machine Press

经 典 原 版 书 库

实用软件工程

（英文版）

Practical Software Engineering

A Case Study Approach

（澳）Leszek A. Maciaszek
Bruc Lee Liong 等著

机械工业出版社
China Machine Press

Leszek A. Maciaszek and Bruc Lee Liong, with contributions from Stephen Bills: Practical Software Engineering: A Case Study Approach (ISBN 0-321-20465-4).

Copyright © 2005 by Pearson Education Limited.

This edition of Practical Software Engineering: A Case Study Approach is published by arrangement with Pearson Education Limited. Licensed for sale in the mainland territory of the People's Republic of China only, excluding Hong Kong, Macau, and Taiwan.

本书英文影印版由英国Pearson Education（培生教育出版集团）授权出版。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。

此影印版只限在中国大陆地区销售（不包括香港、澳门、台湾地区）。

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：01-2005-4509

图书在版编目（CIP）数据

实用软件工程（英文版）/（澳）马查斯泽克（Maciaszek. L. A.）等著. －北京：机械工业出版社，2006.1

（经典原版书库）

书名原文：Practical Software Engineering: A Case Study Approach

ISBN 7-111-17328-7

Ⅰ. 实…　Ⅱ. 马…　Ⅲ. 软件工程－英文　Ⅳ. TP311.5

中国版本图书馆CIP数据核字（2005）第102231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

北京京北制版厂印刷 · 新华书店北京发行所发行

2006年1月第1版第1次印刷

718mm × 1020mm 1/16 · 54印张

印数：0 001 - 3 000册

定价：69.00元（附光盘）

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”已经出版了近百个品种，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，在“华章教育”的总规划之下出版三个系列的计算机教材：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”；同时，引进全美通行的教学辅导书“Schaum's Outlines”系列组成“全美经典学习指导系列”。为了保证这三套丛书的权威性，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。

这三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

此为试读,需要完整PDF请访问: 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

MOTTO

Make everything as simple as possible,
but not simpler

– Albert Einstein

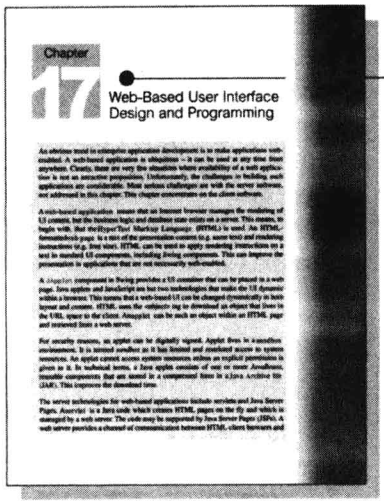
DEDICATIONS

For Diana, Dominika, and Tomasz
– Leszek A. Maciaszek

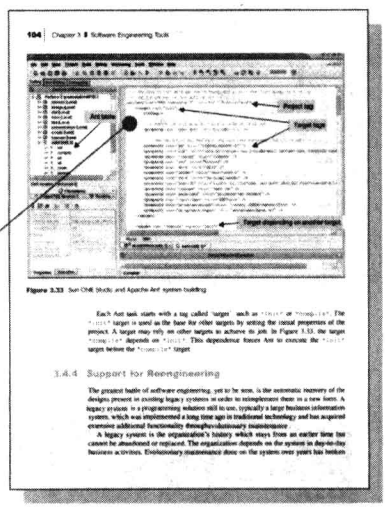
To my parents, Edison and Tina

– Bruc Lee Liong

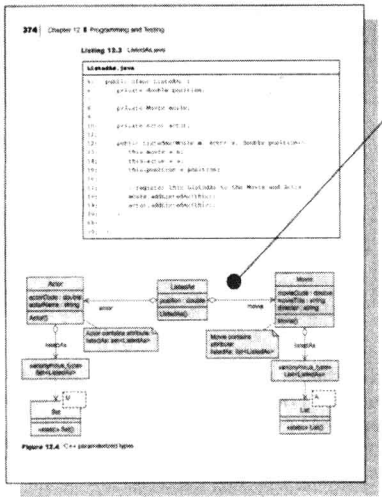
Guided Tour



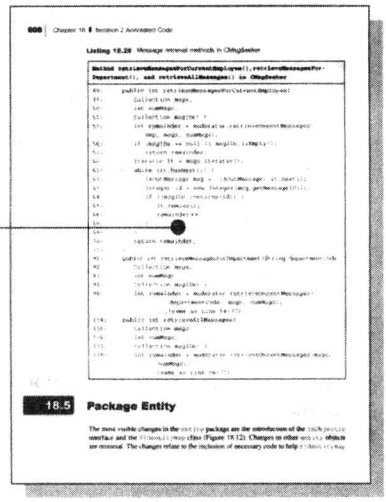
Chapter openings set the scene for more detailed discussion



Numerous **annotated screenshots** to draw out key points

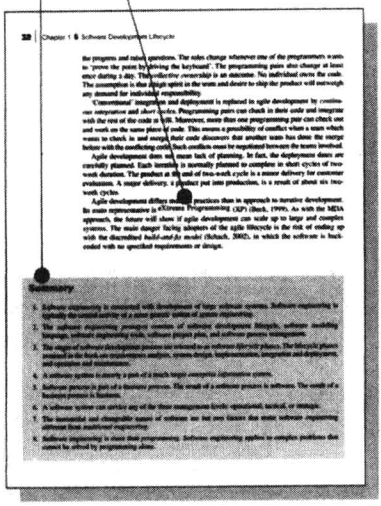


Extensive use of **UML diagrams**

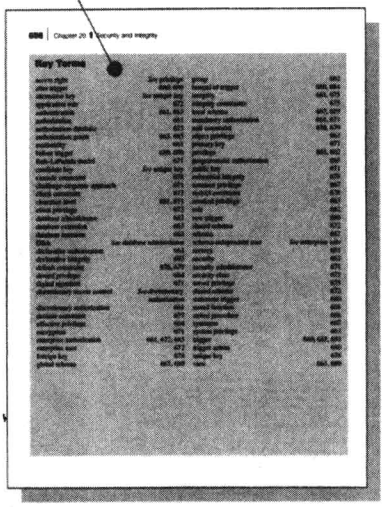


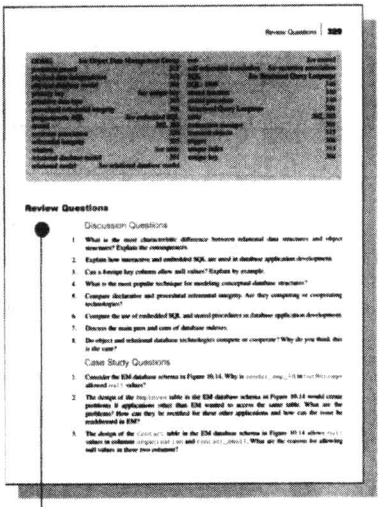
Real code shows readers how to put concepts into practice

Key terms highlighted within text
Summary at the end of each chapter reinforces readers' learning

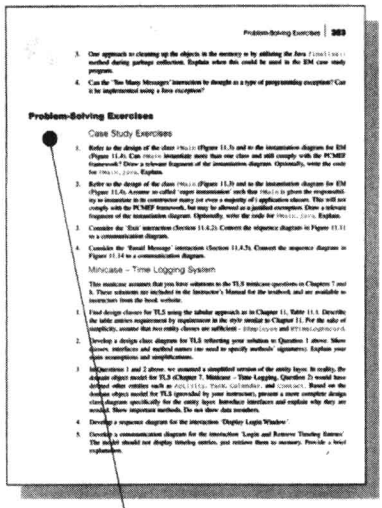


To aid revision, **key terms** and their **in-text definitions** are noted at the end of every chapter





General end-of-chapter **review questions**
End-of-chapter review questions specific to the **running case study**



End-of-chapter **problem-solving exercises** take readers deeper into the case study to ensure understanding
Additional end-of-chapter **minicase exercises** to test understanding further

Preface

The Book's Story

This book has a history of iterative and incremental development. It has certainly undergone all four major phases of one popular lifecycle model: project inception, elaboration, construction, and transition. The book has been an agile development by a pair of authors, with user stories as requirements, with continuous integration and lots of refactoring, but unfortunately not with short cycles to delivery.

The *inception* of the book dates back to the publication in 1990 of Maciaszek's *Database Design and Implementation* (Prentice Hall). Many readers of that book requested a follow-up text with more complete case studies, short and long examples, and with a stepwise (i.e. iterative and incremental) increase of technical difficulty and content sophistication. The business case for the book was made and the project entered the *elaboration* phase – the vision was refined, the risks resolved, the requirements and scope identified, but the target platform ended up to be . . . lots of industry training and consulting instead of a book.

Ten years later, and after countless industrial projects, the amount of practical material collected was screaming for publishing to a wider audience. But the audience has changed – the industry entered the Internet age. A new textbook was needed to define the prerequisite knowledge demanded by modern software engineering. That textbook was Maciaszek's *Requirements Analysis and System Design: Developing Information Systems with UML* (Addison-Wesley, 2001), going into its 2nd edition concurrently with this book. Soon after, the book you are holding entered the construction phase.

The *construction* phase was bumpy. Originally the book was perceived as a companion to Maciaszek's 2001 textbook or any similar book. Later the emphasis shifted to a stand-alone textbook that uses an iterative case study approach to teaching practical software engineering. The book concentrates on software design, programming and management. It emphasizes modern development practices, methods, techniques, and tools.

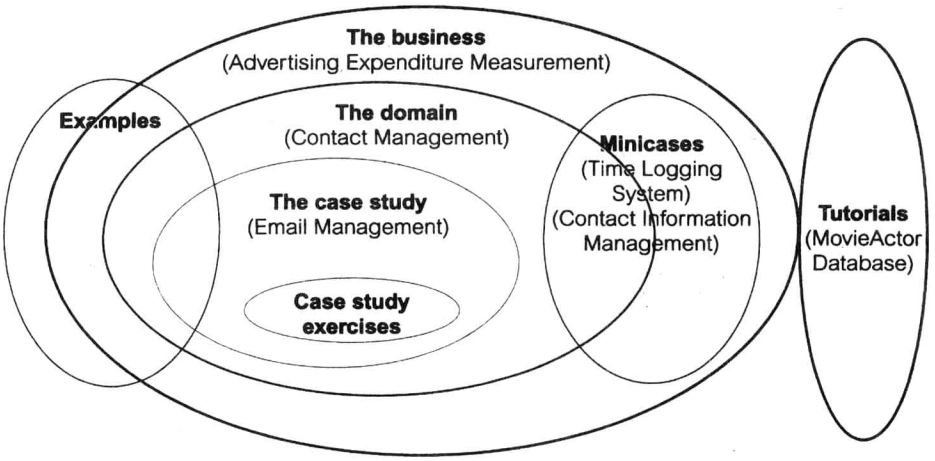
The *transition* phase of this book is in your – the reader's – hands. The beta-tests of this book were conducted in classrooms and on software projects. The deployment is at the reader's mercy. Please submit change requests, defects and enhancements to the development team.

Book Outline and Organization

The distinctive character of this book stems from two endeavors. Firstly, this book is about the way software engineering is done *in practice*. Secondly, it is about software engineering for *enterprise applications*. The following description of what enterprise applications include and exclude applies fully to this book: 'Enterprise applications include payroll, patient records, shipping tracking, cost analysis, credit scoring, insurance, supply chain, accounting, customer service, and foreign exchange trading. Enterprise applications don't include automobile fuel injection, word processors, elevator controllers, chemical plant controllers, telephone switches, operating systems, compilers, and games' (Fowler, 2003, p.3).

The book is pivoted on one main **case study**, two **minicases** with related exercises, a large number of supporting **examples**, **tutorials** to review basic modeling and programming concepts, and end-of-chapter **problem-solving exercises** which contain mostly **case study exercises**. The organization that is a reference for the case study, and for some minicases and examples, is a company specializing in *advertising expenditure measurement*. The book names this organization after its core business activity – AEM (in reality the organization is ACNielsen's Nielsen Media Research in Sydney, Australia). The case study is *Email Management* (EM) – a subsystem of AEM's *Contact Management* (CM) system.

As shown in the Venn diagram, AEM is the business, CM is one of the business domains, and EM is the case study. Examples and minicases are drawn from a cross-section of domains of the AEM business, including CM. Some examples are not related to AEM. The case study is enriched by examples and by case study exercises. Tutorials are used to quickly teach introductory topics with relation to UML modeling, Java programming, relational databases, GUI (graphical user interface) construction, and working with business components.



此为试读,需要完整PDF请访问: www.ertongbook.com

The book endeavors to give broad software engineering knowledge and to provide background information prior to presenting case study solutions. However, a distinguishing emphasis of the book is to show how to apply this knowledge on software projects. For given requirements, the book iteratively develops design and implementation models. Case study, minicases, examples and problem-solving exercises are carefully selected to emphasize various aspects of software development as necessitated by the unique characteristics of different applications and target software solutions.

The book consists of four parts. Part 1 (**Software Projects**) discusses software lifecycle, modeling languages, engineering tools, project planning, and process management. The next three parts (2, 3, and 4) introduce the case study, minicases and examples. The discussion in these three parts concentrates on the methods, techniques, processes, and development environments of software engineering.

Parts 2, 3 and 4 correspond to three project (case study) iterations. Each iteration starts with use case specifications enriched by an initial object model. The generic theory and practical knowledge underpinning each iteration are explained prior to demonstrating the case study design and programming solutions. Any knowledge specific to the case study solutions, and without significant generic appeal, is presented within or as a subsection of the case study discussion. Each iteration results in a complete solution and concludes with a chapter that contains the source code with necessary annotations and references to explanations in prior chapters.

Part 2 (**From Requirements via Architectural Design to Software Release**) starts by giving the business context for the EM case study. The first two chapters in this part present the business object model for AEM and the domain object model for CM. Next, the EM requirements are defined and the EM Iteration 1 is successively developed. The cornerstone of the first iteration is a sound architectural design amenable to successive stepwise enhancements. The 'deliverable' of the first iteration is the software release to the users (i.e. the readers of this book).

Part 3 (**Software Refactoring and User Interface Development**) concentrates on determining the front end of the system and on the presentation and domain layers of the application. It discusses the graphical user interface (GUI) design, including a web-enabled front end. The transformation from Iteration 1 to Iteration 2 is achieved through architectural refactoring and the development of an attractive user interface.

Part 4 (**Data Engineering and Business Components**) moves the focus point from the system front end to its backbone and to the middle tier. This part discusses the storage and manipulation of data, implementation of business rules, transaction processing, and security control. It explains also how the application logic can be moved to an application server in the middle tier.

Although Iteration 3 of the case study is developed from Iteration 2, Parts 3 and 4 of the book can be studied relatively independently. A reader can elect to concentrate on one of these parts and only skim through the other part. For example, a database designer/programmer may have a marginal interest in Part 3, whereas a GUI designer or Java programmer may have a marginal interest in Part 4. An expectation is that some readers will concentrate on Parts 1 and 2 of the book and will fully identify with Parts 3 and 4 after a period of experimentation and gaining better appreciation of the knowledge contained in earlier parts. It is also possible, in more advanced project-oriented courses, to begin using the book from Part 2.

From the total number of 23 chapters in the book, 6 are dedicated to the EM case study (these are Chapters 8, 13, 14, 18, 19, and 23). The educational value of these six chapters is through understanding and analyzing the case study. This is truly learning by example. By contrast, the first five chapters (Part 1) explain the foundations of software engineering and they do not refer to the case study. The remaining 12 chapters have both theoretical parts and the parts that link the theory to the case study, minicases, or other examples.

Distinguishing Features

The main distinguishing feature of the book is in its subtitle: **A Case Study Approach**. If you believe – as many educators do – that the best teaching formula is to *teach by example*, then this book is for you. If you want to be challenged and invited to *learn from mistakes*, then this book gives you plenty of opportunity to experiment with your solutions and compare them with the authors' answers and explanations. If on top of that, you would like to *customize learning to your current needs and level of knowledge* then each iteration has different emphasis, different modeling difficulty and may demand a different subset of development techniques and models.

An overriding objective of this book is to relate theories to reality by giving special attention to software design and implementation (while not neglecting analysis) and by addressing non-trivial practical problems. In its objective of 'exemplifying to explain', the book is unique in a number of ways:

1. *Education in mind.* The book was written with education in mind. The case study, examples and problem-solving exercises are not just plainly taken from real-world solutions; they are molded to suit educational needs. Real-world solutions are part of a complex business and software implementation context. That context is likely to be overwhelming and uninteresting to the reader, so it is simplified as much as possible. Presentation of GUI and database designs as well as programming examples eliminates unnecessary dependencies, 'information noise', and repetitive tasks.
2. *Annotated solutions.* There are no black-or-white, true–false, zero–one solutions in information systems. Frequently, a solution serves a particular purpose and may look plainly wrong when analyzed from a different perspective. Therefore, answers and solutions are carefully annotated.
3. *Alternative solutions.* Sometimes a single solution, no matter how annotated and explained, is not distinctly better than other potential solutions. To this end, alternative solutions are frequently provided and explained.
4. *Lists of key terms* at the end of each chapter compiled as indexes with references to page numbers. The lists can be used for self-study reviews of the understanding of the basic terminology introduced in each chapter. They can also be used by instructors to query the students' knowledge of each chapter.
5. *Review questions* to reinforce the reader's knowledge by insightful questions to each chapter. The questions are divided, when appropriate, into discussion questions and case study questions. Answers to all review questions are available to instructors from the book's website.

6. *Problem-solving exercises* to challenge the reader to research the issues before attempting a solution and to attempt extended or alternative solutions to the case study, minicases or other examples. Sample solutions to all review questions are available to instructors from the book's website.
7. *Website* with complete set of supporting material, including models and programming code (mostly UML, Java and database (Oracle) code). All programming code, including code not presented in the text, is available on the book's website.
8. *Emphasis on principles.* There are some well-defined principles (patterns, frameworks, standards, libraries, etc.) of good software engineering and system development. The book identifies and explains these principles and makes linkages to sources of information.
9. *Balanced mixture of professional depth and educational benefit.* In general, writing software and writing educational books are somewhat disjoint activities. Hopefully, this book contradicts this observation.
10. *Substituting for professional education and training courses.* Busy professionals tend to perform routine tasks and they can quickly fall behind the state-of-art and frequently the state-of-practice in the discipline. Finding time and funds to attend expensive professional education and training courses with case studies similar to those in this book may be difficult. Perhaps this book can give professionals an opportunity to catch up on latest developments at the time of their choice or between normal work duties.

Intended Readership

This book is aimed at a wide readership of students and IT professionals. An ideal reader is a student of a software engineering course or a software developer (or project leader/manager). The book is written so that it can be fully understood by students and professionals who possess basic knowledge of information systems and basic programming skills (hopefully in an object-oriented language and with some database use). For most readers, this corresponds to the first year of a university degree in computer science or informatics (information systems, information technology).

For *students*, the primary use of this textbook is in software engineering courses with a software development component. The book can also be used as a textbook in courses in information systems development, software projects, or systems analysis and design when taught in higher years or to more mature students. Moreover, the book can be used as a recommended reading in courses on object technology, object programming, web-based systems, database design and programming, and similar.

Practitioners most likely to benefit from the book include system designers, programmers, software architects, business and system analysts, project leaders and managers, web and content developers, reviewers, testers, quality assurance managers and industry trainers. The book can be used for professional education and training courses and workshops. It can also be adopted as a source of information for project teams. For practitioners already using UML, Java and relational databases on software projects, this book can serve as a validation of their software development practices and as a source of development ideas and directions.

Supplementary Materials

A comprehensive package of supplementary material is provided for the companion website. Most of the website content is freely available to all readers, but some material is password-protected for the benefit of instructors who have adopted the book in their teaching. The home page for this book is maintained at:

http://www.comp.mq.edu.au/books/pse
http://www.booksites.net/maciaszek

The web package for this book contains two sets of resources: for all readers and for instructors who adopted the textbook for teaching. The instructor resources are password-protected.

Instructor resources on the web include (but are not limited to):

1. *Instructor's Manual* with:
   (a) *questions and answers* to all end-of-chapter review questions and problem-solving exercises
   (b) extra *projects and solutions* – not contained in the textbook and available from the website to assist instructors in setting up student assignments and projects
2. *Lecture slides* – in PowerPoint and in modifiable Acrobat pdf formats
3. *UML models and Java/Database source code* for solutions to:
   (a) exercises and minicases
   (b) projects provided on the book's website
   (c) alternative design/programming approaches to the book's case study

Resources for all readers include (but are not limited to):

1. *Lecture slides* – in printable-only Acrobat pdf format
2. *Errata and addendum* document
3. *UML models and Java/Database source code* for the book's case study and examples, complete with instructions on how to compile and run the code.

Your comments, corrections, suggestions for improvements, contributions, etc. are very much appreciated. Please, direct any correspondence to:

Leszek A. Maciaszek
Department of Computing
Macquarie University
Sydney
NSW 2109, Australia
email: leszek@ics.mq.edu.au
web: http://www.comp.mq.edu.au/~leszek
phone: +61 2 9850-9519
facsimile: +61 2 9850-9551
courier: North Ryde, Herring Road, Bld. E6A, Room 319

Acknowledgements

Author's Acknowledgements

This book took a considerable time to write but that time shrinks to insignificance in comparison with the time taken to gain the knowledge and skills necessary for its writing. Our special gratitude goes to our friends and colleagues at **ACNielsen, Sydney, Australia**, who provided the initial 'testbed' for much of the knowledge transferred to the readers in this book. Our thanks go in particular to Stephen Bills (who is also this book's named contributor) and his team of developers, including Paul Antoun, Bruno Beira, Sue Dayes, Steven Grotte, Jeff Hong, Yijun Li, Kevin Mathie, Denise McCrae, Chantal O'Connell, James Rees, Jovan Spoa, Eric Zurcher.

Writing a book is a considerable project. As explained in Chapter 4 and elsewhere in the book, a successful project requires that sufficient work and material resources be allocated to project tasks. Work resources consist of people and equipment, including hardware and software. Material resources are consumables and supplies. The authors of this book allocated themselves to the tasks, but the project would dismally fail without all accompanying work and material resources. The resources were provided by **Macquarie University, Sydney, Australia**, and **University of Economics, Wrocław, Poland**. We are indebted to our friends, colleagues and students at these two universities for their advice, support, and assistance, for all the resources that they have provided to this project.

Talking about resources, this book could not have been written without intensive and wide-ranging use of software tools and environments. The software necessary for the book was obtained by various means, from purchase via downloads of demonstration copies to using open software sources. People behind software obtained through such means are anonymous to us, so acknowledgements are not possible. Our acknowledgements are, however, in place to these software vendors who responded to our requests to provide software free of charge for educational use by us and our students. We are particularly indebted to Oracle Corporation (Oracle and JDeveloper), Rational Software Corporation, currently IBM (Rational Suite), Sybase (PowerDesigner), and yWorks (yDoc).

Most significantly, we are grateful to **Keith Mansfield** of Pearson Education, the editor of this book as well as the editor of Maciaszek's *Requirements Analysis and Systems Design*. Thank you, Keith, for having the vision for both books and for your hard work from inception to production and deliverables. Book production is a real team effort. Risking the omission of quite a few names, we would like to thank especially: **Anita Atkinson** (senior desk editor), **Helen MacFadyen** (proofreader), **Ruth Freestone King** (copyeditor)

and **Owen Knight** (editorial assistant). Thank you all for your corrections, improvements, insights, advice and great cooperation.

Publisher's Acknowledgements

We are grateful to the following for permission to reproduce copyright material:

Table 1.1 based on information from *Fundamentals of Software Engineering*, Prentice Hall (Pearson Education, Inc.), (Ghezzi, C., Jazayeri, M. and Mandrioli, D., 2003); Figure 1.10 adapted from figure of *Rational Unified Process*, reprinted by permission from http://www.rational.com/products/rup/, © Copyright 2003 by International Business Machines Corporation. All Rights Reserved; Figures 3.1, 4.3, 4.4, 4.5, 4.7, 4.8, 4.9, 4.10, 4.11, 4.13, 4.14, 4.16, 4.17, 4.19, 4.20, 4.21, 4.22, 4.23, 4.25, 4.26, 4.28 and 4.29 from screen shots of Microsoft® Office Project (2003), reprinted by permission from Microsoft Corporation, Copyright © 1998–2003 Microsoft Corporation; Figure 3.2 from screen shot of ManagePro™ 6.1 from www.managepro.net, March 2004, Performance Solutions Technology, LLC, reproduced by kind permission of Performance Solutions Technology, LLC; Figure 3.3 from screen shot of eRoom from www.eroom.net/eRoomNet/, July 2003, Documentum, Inc., reproduced by kind permission of Documentum, Inc.; Figure 3.4 from a screen shot of eProject Enterprise, July 2003, reproduced by kind permission of eProject, Inc.; Figures 3.5 and 3.6 from screen shots of *Small Worlds*, reprinted by permission from www.thesmallworlds.com/, © Copyright 2003 by International Business Machines Corporation. All Rights Reserved; Figure 3.8 from screen shot of @Risk from www.palisade-europe.com, reproduced by kind permission of the Palisade Corporation; Figures 3.9, 3.10, 3.12, 3.14, 3.16, 3.30, 3.31, 5.14, 5.15 from screen shots of *IBM Rational Suite*, reprinted by permission from *Rational Suite Tutorial, Version 2002.05.00*, © Copyright 2002 by International Business Machines Corporation. All Rights Reserved; Figure 3.9 from screen shot of Microsoft® Word, reprinted by permission from Microsoft Corporation, Copyright © 1998–2003 Microsoft Corporation; Figure 3.11 from screen shot of DOORS® from www.telelogic.com, reprinted by kind permission of Telelogic UK Ltd., Copyright © 2004 Telelogic AB; Figure 3.13 from screen shot of Enterprise Architect from www.sparxsystems.com.au, August 2003, Sparx Systems Pty Ltd., reproduced by kind permission of Sparx Systems Pty Ltd.; Figure 3.15 from a screen shot of Gentleware's Poseidon Sales Application model, August 2003, reproduced by kind permission of Gentleware AG; Figures 3.16, 3.34 and 3.35 from screen shots of No Magic's MagicDraw™ from www.magicdraw.com, July 2003, © Copyright No Magic, Inc. 1998–2004, reproduced with permission. All Rights Reserved; Figure 3.17 from screen shot of Sybase® PowerDesigner® version 9.5, from www.sybase.com, July 2003, © Copyright 2002, Sybase, Inc., All Rights Reserved; Figure 3.25 from screen shot of *Borland® Together® ControlCenter® example* (2003), www.borland.com, reprinted by permission of Borland Software Corporation; Figure 3.27 from screen shot of Oracle JDeveloper from www.otn.oracle.com, reproduced by permission of Oracle Corporation; Figure 3.29 from screen shot of Perforce, from www.perforce.com, reproduced by kind permission of Perforce Software, Inc.; Figure 3.32 from screen shot of Microsoft® Visual SourceSafe from http://msdn.microsoft.com/ssafe/default.asp, (2003), reprinted by permission from Microsoft Corporation, Copyright ©

此为试读,需要完整PDF请访问: www.ertongbook.com