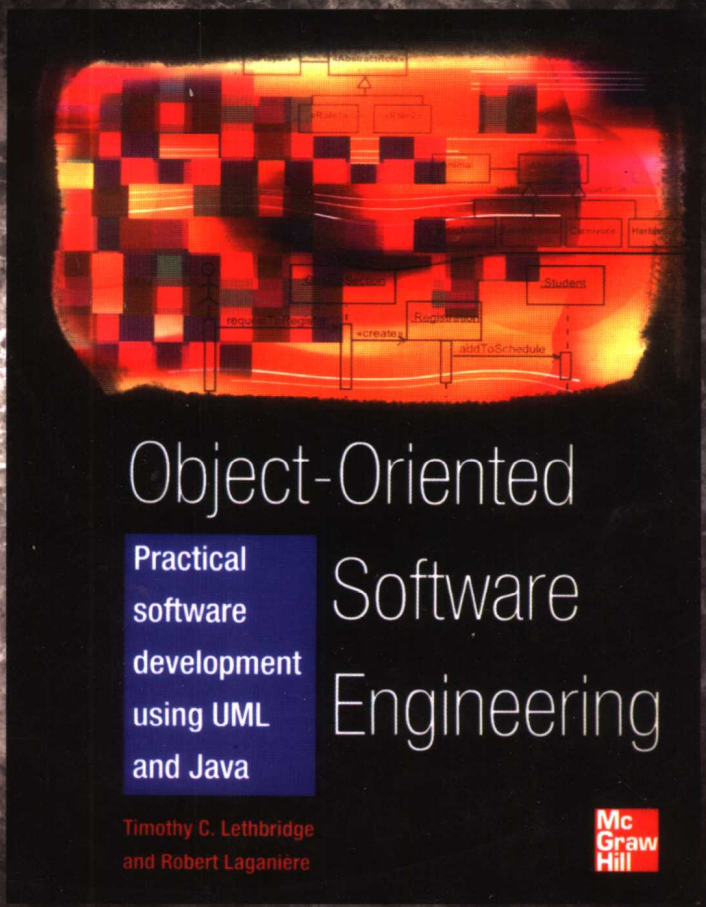
计 算 机 科 学 丛 书

面向对象软件工程

Timothy C. Lethbridge Robert Laganière 著 张红光 温遇华 徐巧丽 张楠 译



Object-Oriented Software Engineering

Practical Software Development Using UML and Java

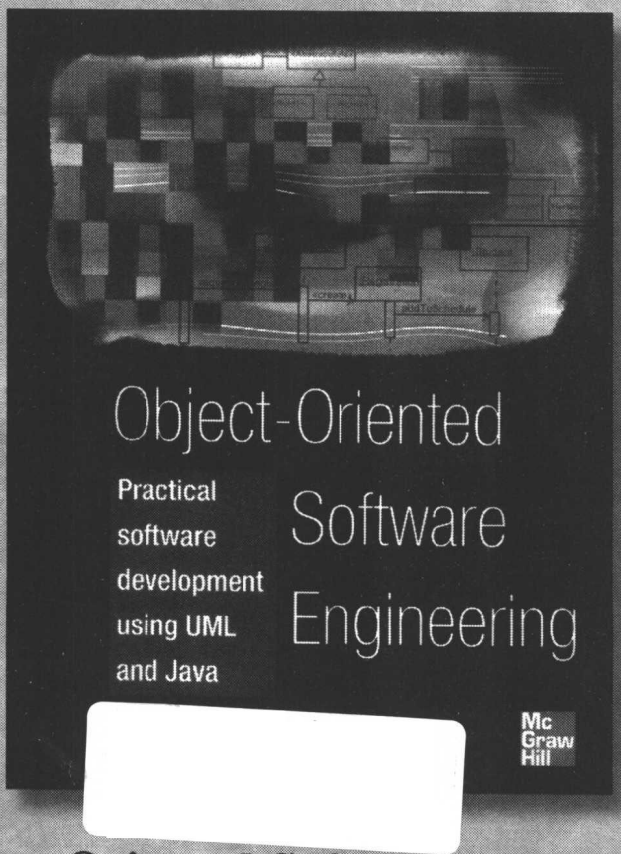
机械工业出版社
China Machine Press

Mc Graw Hill

计 算 机 科 学 丛

面向对象软件工程

Timothy C. Lethbridge Robert Laganière 著 张红光 温遇华 徐巧丽 张楠 译



Object-Oriented Software Engineering

Practical Software Development Using UML and Java

机械工业出版社
China Machine Press

本书深入地讲解了软件工程的主要原理。内容包括：基于可靠的原则和可重用技术进行开发、使用UML进行可视化建模、对需求分析和设计中各种方案进行评估、面向对象技术、迭代开发、风险管理等等。书中含有大量的练习与例子，读者可以将这些概念应用于实践中。

本书可以作为高校软件工程课程的教科书，也适用于软件开发技术人员参考。

Timothy C.Lethbridge & Robert Laganière:Object-Oriented Software Engineering(ISBN:0-07-709761-0)

本书版权登记号：图字：01-2002-4800

图书在版编目（CIP）数据

面向对象软件工程 /（加）莱斯布里奇（Lethbridge，T. C.），（加）拉格尼（Laganière，R.）著；张红光等译. -北京：机械工业出版社，2003

（计算机科学丛书）

书名原文：Object-Oriented Software Engineering

ISBN 7-111-11904-5

Ⅰ. 面… Ⅱ. ① 莱… ② 拉… ③ 张… Ⅲ. 软件工程 Ⅳ. TP311.5

中国版本图书馆CIP数据核字（2003）第023803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：张金梅

北京中加印刷有限公司印刷 · 新华书店北京发行所发行

2003年4月第1版第1次印刷

787mm × 1092mm 1/16 · 22.5印张

印数：0 001- 5 000册

定价：35.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

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，有的还专诚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：hzedu@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

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			

译 者 序

在软件开发中经常会碰到问题和困难，而且很难彻底解决，这一点与其他行业有很大的差异，为什么呢？通过分析软件开发所面临的现实，不难回答。软件是由计算机程序构成的，而计算机程序的本质是一组相互交错的概念集合，包括数据集、数据集之间的关系、算法及函数。这些概念是高度精确的，而且包含的内容非常丰富。软件开发中的困难包含识别、设计和检测这些概念本身；设计人员对软件系统本身有误解；开发者 、管理者、市场人员和客户关系紧张；非开发人员对软件开发本身的客观性认识不够，主观意识过强；开发人员无法与客户充分沟通等等。这些现实是造成软件开发问题的根源。

在软件开发过程中，设计人员是核心角色，即设计人员的工作质量直接影响着软件产品的质量。本书就是以培养软件设计工程师为目标的，描述软件开发者应该做什么及如何去做。书中包含了软件工程中研究的许多基础问题，如客户需求了解、设计原则和使用的开发技术、UML可视化建模、设计方案评估、面向对象设计原理、迭代开发、交流文档、软件风险管理等问题。

本书涉及的范围广泛，论述提纲挈领，资料丰富，非常适合用作少学时软件工程课程的教材。

本书的第1、2、4章由张红光翻译，第3、5、6、7章由温遇华翻译，第8、9、10章由徐巧丽翻译，第11、12章由张楠翻译，同时张楠还负责全书的版式调整及校对，张红光负责全书的整理与统稿。由于时间仓促及译者水平有限，错误和疏漏之处在所难免，敬请读者多提宝贵意见。

译 者

2003年3月于南开园

前　言

本书讲述了大量可以立即应用于实践的软件工程知识和技能，主要针对高校第二学年的软件工程课程。对于那些有一些程序设计背景，但是想更好地理解现代软件工程的读者来说，本书也有参考价值。

我们教授软件工程课程已经有十年的时间了，使用本书素材的早期版本作为教材也已经有三年了，并且得到了学生积极的反馈。此外，我们进行软件开发、咨询和专业培训的业界经验也使我们能够把本书的重点放在对于学生的未来雇主而言比较重要的内容上。

在软件工程学位项目中使用本书

软件工程正在成为一门成熟的学科，并逐步从计算机科学和计算机工程中分离出来。这种趋势的一个主要表现，就是全世界各大高校中纷纷建立了专门的软件工程学位项目。我们编写这本书的目的就是在这种学位项目中向学生介绍软件工程最重要的概念。

在渥太华大学，我们在第二学年的第一学期用12星期的时间教授本书的内容。这时，学生应该已经完成了两个学期的计算机科学课程——包括Java面向对象程序设计。在学习本课程的同时，他们还应该学习数据结构和算法等课程，然后再参加高级软件工程课程的学习，在高级软件工程课程学习时可以进一步扩展这里所学的内容。

学习过这门课程的学生应该能够胜任在企业中的暑期实习工作。老板们期望学生具有如下能力：了解好的需求中都包含什么，能够运用基本的设计原则，能正确地使用UML，能够把需求和设计转变为高质量的程序并且有效地测试这些程序。本书给出了所有这些技能的实践基础。

本书是这样组织的，在12星期的课程或单元中，可以使用每星期3个小时的课堂教学加上定期的实验练习。每年我们都布置一些练习，其中的许多内容要求学生分组共同完成。

使用本书提高技能和知识水平

除了学习软件工程的学生以外，本书还适用于那些具有一些编程背景、需要阅读软件工程实用入门资料的读者。这些人可能需要通过更新技能来适应知识经济的需求，或者经历若干年的编码工作后想提高自己的知识水平。

建议具备的知识背景

在学习本书之前，读者应该了解面向对象编程的基本概念，尽管第2章概述了这些概念。由于Java是完全的、易于掌握的而且流行的面向对象语言，所以我们选择它来作为本书的编程范例语言。了解其他面向对象语言的读者能够从第2章和本书的网站上以及在完成练习的过程中获得必要的Java知识。

网站上的资料

我们准备了一个资源丰富的网站，地址是：www.mcgraw-hill.co.uk/textbooks/lethbridge，读者和教师可以从那里得到进一步的支持。

在网站上可以找到授课的幻灯片、源代码、练习的答案、网上参考资料的链接、总结了许多概念的知识库以及多种其他的辅助学习工具，以及收集反馈信息的表单。

我们期盼着收到读者的反馈信息。

贯穿全书的主题

本书主要论述8个主题，我们认为它们是21世纪软件工程的基础。这些主题在许多章节中都被重复介绍，并且在相关的具体例子和练习中进行了讲解。

1) **了解客户和用户**。我们强调领域分析与需求收集和需求确认一样重要。我们把这些内容放在用例分析和可用性的背景中讲述。我们要求读者按照这样的思路去思考：客户问题到底是什么？现实情况又怎么样，等等。在本书的一开始，我们就把软件工程的目的描述为解决客户的问题，而不是解决软件开发自身的问题。

2) **基于可靠的原则和可重用技术进行开发**。我们强调，软件工程师在着手进行一个项目前需要理解设计原则并彻底掌握适用的技术。为了保证在本书的设计工作中确实如此，我们首先回顾了面向对象设计原则，接着讨论了框架、设计原则和设计模式。

3) **使用UML进行可视化建模**。我们介绍了UML的关键元素，特别是类图和交互图，此外还较为简略地讲到了状态图。由于UML没有覆盖所有软件工程的内容，本书没有讲述UML的所有内容，也没有把讨论只限制在UML本身上，我们强调的是UML图本身不能解决问题，但它是软件工程师应该在日常工作中使用的工具之一。

4) **对需求和设计中各种方案的评估**。在整本书中，我们介绍了各种方案和它们的优缺点，以及进行每个选择的基本原理。学生应该对方案的评估进行实践。

5) **面向对象**。本书包含了面向对象开发的所有方面，包括分析、设计和编程。确保读者能够看到一个项目的完整实现过程，可以使他们除了能抽象地了解开发过程之外，还能获得更多的东西，并且理解许多设计原则的成因。

6) **迭代开发**。我们着重强调了读者应该遵循迭代方法。在项目练习中，全书一直要求读者完成需求、设计和实现。为达到这一点，在第3章介绍了一个完整的项目。最开始，我们只要求读者对项目做一些小改动以初步理解项目。然后在第4章要求读者为系统的新功能编写并评审需求——他们又一次设计和实现这些功能。接下来，读者了解了主题的更多细节，如设计和质量保证，并且要求把所学的内容用于项目的不断升级改进中。

7) **使用文档有效地交流**。我们鼓励读者练习编写内容丰富的文档，本书提供了每一种类型文档的模板和实例。

8) **在所有软件工程活动中的风险管理**。全书讨论了风险管理的许多方面，包括定期评估潜在成本和风险、权衡风险和效益、避免事倍功半，以及随着对项目的深入了解不断改进计划。我们指出，从上面其他主题学到的知识可以用于降低风险。

本书的组织

篇幅。本书篇幅不长，教师可以要求学生在12星期的课程中读完。我们在下面给出了时间安排的建议。

深度。本书没有讲述软件工程的所有方面，而是在合理的深度上介绍了一系列相互联系紧密的内容，为读者提供了这一领域中心主题的知识基础。我们的精力主要放在那些可以立即应用到工业级软件项目的内容。

例子和练习。本书提供了大量的例子和练习，读者可以将这些概念应用于实践中。深入的项目练习是以我们提供的一个完全实现的小系统为基础的。这就意味着不用总是从头编程，

读者能够把时间花在考虑更高级的分析和设计问题上，而且还能实际实现自己的想法。读者借此还可以理解重用，因为该系统是基于框架的，而此框架可以在各种客户机-服务器系统内应用。练习在难度上变化很大。一些比较简单的练习只是鼓励读者去思考他们所学过的内容，而另一些则为了激发专业程度较高的读者的主动性。在我们的网站上，许多练习都有带完整解释的答案。

先后顺序。本书中内容的顺序安排循序渐进，理论联系实际，因此可以使学生迅速地将需求分析、设计和实现应用于现实问题。例如，本书首先介绍了理解项目工作中所需的有关面向对象的知识和体系结构，然后进行需求和面向对象分析，然后是静态建模，然后是介绍用例建模和动态建模。与此同时练习也随之更难。

在12星期的课程中使用本书

下面是在本科第二学年课程中使用本书的进度安排的建议。对于本书的主体内容——第3章到第10章，安排的时间大致与各章的长度成正比。

作者在12星期的课程中使用这本书时，每星期有三个小时的授课和三个小时的实验和辅导时间。要求学生阅读所有章节，尽管讲授重点集中在从第3章到第10章的核心内容上，尤其是第3章、第5章、第8章和第9章。

我们还预定学生在课程过程中完成所选练习和项目活动。我们已经在许多章的末尾提供了对项目活动的建议。

第1星期：　第1章和第2章——介绍和回顾（1星期）

第2～3星期：　第3章——重用和客户机—服务器框架（1.5星期）。

项目工作：通过稍加修改使用客户机-服务器框架实现的一个系统，学会使用这种框架。

第3～4星期：　第4章——领域分析和需求（1星期）。

项目工作：在需求分析之后增加新的功能。

第4～5星期：　第5章——面向对象分析和建模（1.5星期）。

项目工作：增加需要大量建模工作的功能。

第6星期：　第6章——设计模式（1星期）。

第7星期：　第7章——用例和用户界面（1星期）。

项目工作：增加一个图形用户界面。

第8～9星期：　第8章——动态建模（1.5星期）。

第9～10星期：　第9章——设计原则和体系结构（1.5星期）。

项目工作：一些功能的详细设计。

第11星期：　第10章——测试（1星期）。

项目工作：　准备一个测试计划。

第12星期：　第11~12章——项目管理的介绍和回顾（1星期）。

也可以按照其他顺序进行学习，尤其是从第6章到第11章的顺序可以灵活安排。还有，为了更强调其他内容，许多章的部分内容可以略过。

感谢

我们要感谢下面这些人，他们帮助我们改进了本书的早期草稿：

- 早期草稿的评论者，他们提了许多有用的建议。这些人包括：Hausi Müller、Mike Lutz、Carol Greswell、Rohit Bahl、Bob Probert、Lionel Briand和Mike Bennett。
- 仔细编辑了本书中许多内容的K. Teresa Khidir。
- 在我们精炼规格说明时辛苦地反复修改代码的Francois Bélanger。
- 完成了相关网站知识库的工作并帮助精炼词汇表的Judy Kavanagh。
- 渥太华大学参加了SEG 2100和SEG 2500课程的学生，在他们身上我们实验了本书的早期版本和软件，并且从他们那里得到了许多有用的反馈。特别感谢Dan Danis、Ali Echihabi、Stephane Jacoby、Meng Han、Mudassar Hayee、Patrice LaFlamme、Aleksandar Lukic、Vinh Mai、Ryan Rebello、Jim Sellers、James Ward、Karen Williamson和那些不愿透露姓名的人们。

目　录

第1章　软件和软件工程

软件工程师的工作是开发出高质量的软件来经济地解决实际问题。在第1章中将介绍一些每个软件工程师都应当了解的重要主题。

本章内容：

- 软件与其他产品有什么不同？随着时间的改变软件将如何变化？高质量软件的含义是什么？软件有哪些类型，它们之间有什么主要区别？
- 软件项目是怎样组织的？典型项目的成效如何？
- 如何定义软件工程？为什么遵循软件工程的规范方法有助于开发成功的软件系统？
- 哪些活动会在每一个软件项目中都出现？
- 在进行任何一项软件工程活动时，应当谨记哪些要点？

1.1　软件的特性

机械工程师设计机械系统、电气工程师设计电气系统——与之相似，软件工程师设计的是软件系统。然而，软件与其他类型的工程师设计的产品存在重要的差别：

- 软件是*无形的*（intangible），比其他工程产品更加不可捉摸。你无法感觉软件产品的形状，它的设计也难以直观表示。因此确定软件产品的质量或者估计其开发工作量是非常困难的。这就是人们总是低估系统开发时间的一个原因。
- 软件副本的大批量生产轻而易举。绝大多数其他类型的工程师非常关心每一个部件耗费的成本与劳动量。换句话说，对于有形产品，设计完成之后的过程往往是代价昂贵的部分。与之相反，软件产品能通过网络下载或制作CD以极低的成本进行复制。因此，几乎全部的软件成本都存在于它的开发过程中，而不是制造过程中。
- 软件业是劳动密集型的。通过使用机器，制造业与建筑业的许多领域已经实现了自动化；因此其他工程分支可以用更少的劳动量生产出更多的产品。但是，使软件的设计与编程完全实现自动化，需要真正“智能”的机器。目前这个方向上的尝试离成功还很遥远。
- 一个没有经过充分训练的软件开发人员很容易编写出难以理解和修改的软件。编程新手可以创建复杂的系统完成有用的功能，但它的设计却可能极其紊乱的。其他工程领域也可能产生不良的设计，但是这些缺陷通常比较容易发现，因为它们不会隐藏在上千页的源代码中。例如，如果土木工程师设计了一座不安全的桥梁，审查人员一般会比较容易地发现设计中的缺陷，因为他们清楚地知道在每张图纸和每次计算中应当寻找什么。通常，设计不良的软件系统至少在一定程度上还可以运行，而拙劣的设计会导致许多其他类型的工程产品根本无法工作。
- 软件本身很容易修改，但由于它的复杂性，又很难正确地修改。人们总是试图在尚未完全了解软件的时候就去修改它，而这些修改又会带来新的错误。
- 软件不像其他的工业产品那样会因使用而磨损，但随着反复修改，它的设计会逐渐退化。上一点提到，对软件的修改很容易引入新的缺陷，因此修改过的软件从设计角度讲容易

变差。随着时间的流逝，软件后续版本的设计可能会表现出严重的退化，这时就需要完全重新设计了。

总而言之，以上这些特性说明，大部分现有软件的质量都相对较差，而且还在不断地变得更差。同时，对新的、修改过的软件的需求旺盛，客户希望它们质量高，并且能够很快地生产出来。因此，软件开发人员常常满足不了经理与客户的期望——许多软件项目不是根本没有交付，就是推迟交付而且超出了预算。此外，许多已交付的软件系统的问题太多，从未投入使用；而其他一些软件在使用前需要做较大的修改。

所有这些问题称为*软件危机*（software crisis），尽管事实上这种危机已经存在几十年了。选用“危机”这个术语，是希望这个随着软件产业的发展而产生的问题会通过实施改进的软件工程方法得到解决。尽管这种观点依然适用，我们现在也认识到，软件产业的困难在一定程度上是软件复杂特性的自然结果，经济学规律与捉摸不定的人类心理也在起作用。

本书的目的是教会你怎样“以工程化的方式设计”软件，使软件可以满足预期需求而不导致危机。为了做到这一点，需要学习一些技术，能够减少或隐藏复杂性，并解释经济学和心理学方面的实际问题。

软件的类型及其区别

软件有很多不同的类型。一种重要的分类方法是将软件分为*定制软件*（custom software）、*通用软件*（generic software）和*嵌入式软件*（embedded software）。

定制软件主要是为了满足特定客户的特殊需要，这种软件对其他客户的用处不大（尽管在某些情况下，定制软件的开发可能会揭示出几个类似机构所共有的问题）。大多数定制软件是在使用该软件的机构*内部*（in-house）开发的；在其他情况下，开发被*外包*（contract out）给咨询公司。定制软件通常只由少数人使用，其成功与否依赖于软件是否能满足这些人的需要。

定制软件的例子有网站、空中交通管制系统和大型机构的财务管理软件系统。

另一方面，开发通用软件是为了在开放的市场上出售，它们在通用计算机上完成许多人需要的功能。软件的需求主要由市场调研确定。现在业界有用通用软件代替定制软件的趋势，因为通用软件远比定制软件便宜而且也更可靠。其中的主要困难在于，通用软件不一定能够完全满足软件使用机构的特定需要。通用软件被称为*商业成品*（Commercial Off-The-Shelf，COTS）*软件*，因为它们常常装在塑封的盒子中出售，所以有时也称作*塑封*（shrink-wrapped）软件。通用软件生产者希望能够卖出许多拷贝，但是他们的成功与否受市场力量的支配。

通用软件的例子有字处理器、电子表格软件、编译器、Web浏览器、操作系统、计算机游戏及小型企业财务管理软件包。

嵌入式软件运行于通常在开放市场出售的特定硬件设备中。这样的设备包括洗衣机、录像机、微波炉和汽车。与通用软件不同，如果不同时更换硬件，用户通常不能更换或升级嵌入式软件。硬件设备的开放市场特性意味着开发嵌入式软件与开发通用软件有相似之处，但是因为开发方法存在明显的区别，我们将它归入了另一个类别。

由于嵌入式系统正在逐渐进入数量巨大的消费和商业产品中，它们占据了现有软件拷贝数量的很大份额。另一方面，在通用计算机（如PC）上运行的绝大部分软件都是通用系统。尽管定制软件不会像其他两种软件类型卖出那么多的拷贝，但是更多的系统上运行的都是定制软件，因此绝大多数开发人员开发的是定制软件。

定制通用软件是可能的。但是，这样做存在风险，当通用软件的新版本发布时，定制工作必须重新进行。

也可以将定制软件修改为通用软件，然而如果软件的设计不够灵活，这可能是一个非常复杂的过程。

表1-1总结了定制、通用和嵌入式软件的重要特性。

表1-1 定制、通用和嵌入式软件之间的区别

	定制	通用	嵌入式
正在使用的拷贝数量	低	中	高
运行该类软件的总的处理能力（processing power）	低	高	中
世界范围内每年投入的开发力量（development effort）	高	中	低

另一种重要的总体分类方法将软件分为实时（real-time）软件和数据处理（data processing）软件。大多数实时软件用于对特定用途的硬件进行操作，如前面描述的嵌入式系统，又如工厂和电话网。实时软件最独特的特征是，它必须立即（也就是实时地）对环境中的请求信号（例如用户按下的一个键或从传感器上传来的信号）做出反应。设计中投入了大量精力来保证系统应答的及时性。

通用应用程序，如电子表格和计算机游戏，也具有一定的实时特性，因为它们必须响应用户的输入。但是这通常是软实时（soft real-time）特性：当不能满足定时限制时，这些系统仅仅是变得很慢而已。与之相反，大多数嵌入式系统具有硬实时（hard real-time）特性，如果实时限制无法满足，系统就会彻底失败。因此安全性通常是设计这类系统时要考虑的关键问题。

数据处理软件通常用于业务运营中，它完成诸如销售记录、账目管理、票据打印等功能。这里最大的设计问题是如何组织这些数据，并为用户提供有用的信息，使用户能够有效地完成他们的工作。数据的准确性和安全性是至关重要的指标，因为系统收集的可能是人们的私人信息。传统数据处理任务的一个关键特征是，系统不是在数据可用时就对其进行处理，而是将数据集中按批（batch）留待以后处理。

有的软件会同时兼有实时与数据处理两种特性，例如电话系统，它的电话通话管理是实时的，而对这些通话记账是数据处理活动。

软件的“年龄”各不相同。许多20世纪60、70年代编写的定制软件今天仍在使用。这些软件与近期开发的软件在程序设计语言、数据存储技术、用户界面技术和设计技巧上都不相同。

“software”一词的用法
——母语不是英语的人常犯的一个错误

许多母语不是英语的人经常会说出这样的错误句子：“I will create a software to update the database”（我们将创建一个软件去修改数据库）。错误在于不能说“a software”。“software”用做名词时，是物质名词（mass noun），就像“water”（水）和“sand”（沙）一样，不能在它前面使用不定冠词“a”。所以应当说“I will create some software to update the database”，或者“I will create a piece of software to update the database”。也可以将“software”一词作为形容词使用，如“I will create a software system to update the database”。这种情况下不定冠词修饰的是“system”，而不是“software”。

练习1.1 将以下软件分类，说明它们更接近于定制软件、通用软件还是嵌入式软件，是数据处理软件还是实时软件。