



# Object-Oriented Software Engineering

David C. Kung 著 牟永敏 邢颖 译

# 面向对象 软件工程



# Object-Oriented Software Engineering

David C. Kung 著 牟永敏 邢颖 译

# 面向对象 软件工程



David C. Kung

**Object-Oriented Software Engineering, First Edition**

ISBN: 9780073376257

Copyright ©2014 by McGraw-Hill Education.

All Rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including without limitation photocopying, recording, taping, or any database, information or retrieval system, without the prior written permission of the publisher.

This authorized Chinese translation edition is jointly published by McGraw-Hill Education and Tsinghua University Press Limited. This edition is authorized for sale in the People's Republic of China only, excluding Hong Kong, Macao SAR and Taiwan.

Copyright © 2016 by McGraw-Hill Education and Tsinghua University Press Limited.

版权所有。未经出版人事先书面许可，对本出版物的任何部分不得以任何方式或途径复制或传播，包括但不限于复印、录制、录音，或通过任何数据库、信息或可检索的系统。

本授权中文简体字翻译版由麦格劳-希尔（亚洲）教育出版公司和清华大学出版社有限公司合作出版。此版本经授权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。版权©2016 由麦格劳-希尔（亚洲）教育出版公司与清华大学出版社有限公司所有。

北京市版权局著作权合同登记号 图字：01-2014-4570

本书封面贴有 McGraw-Hill 公司防伪标签，无标签者不得销售。

版权所有，侵权必究。侵权举报电话：010-62782989 13701121933

#### 图书在版编目（CIP）数据

面向对象软件工程 /（美）David C.Kung 著；牟永敏，邢颖译。—北京：清华大学出版社，2017  
（清华计算机图书译丛）

书名原文：Object-Oriented Software Engineering

ISBN 978-7-302-46094-7

I. ①面… II. ①D… ②牟… ③邢… III. ①面向对象语言-程序设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字（2016）第 312621 号

责任编辑：袁勤勇

封面设计：傅瑞学

责任校对：白蕾

责任印制：杨艳

出版发行：清华大学出版社

网 址：<http://www.tup.com.cn>, <http://www.wqbook.com>

地 址：北京清华大学学研大厦 A 座 邮 编：100084

社 总 机：010-62770175 邮 购：010-62786544

投稿与读者服务：010-62776969, [c-service@tup.tsinghua.edu.cn](mailto:c-service@tup.tsinghua.edu.cn)

质量反馈：010-62772015, [zhiliang@tup.tsinghua.edu.cn](mailto:zhiliang@tup.tsinghua.edu.cn)

课 件 下 载：<http://www.tup.com.cn>, 010-62795954

印 刷 者：北京富博印刷有限公司

装 订 者：北京市密云县京文制本装订厂

经 销：全国新华书店

开 本：185mm×260mm

印 张：35.5

字 数：840 千字

版 次：2017 年 1 月第 1 版

印 次：2017 年 1 月第 1 次印刷

印 数：1~2000

定 价：98.00 元

产品编号：057631-01

# 译者序

计算机已经渗透到社会生活的方方面面。计算机之所以能够被广泛使用,其背后的推动力就是市场经济。但是,实际上是软件在指挥着计算机按照人们想要的方式在工作。软件或者说计算机程序由成千上万条指令构成,这些指令指挥着计算机进行复杂的运算并且控制计算机硬件设备的运行。近些年对于计算机软件的需求快速增长。为了能够满足软件开发工程师或系统分析师的工作需求,学习软件工程的相关知识是非常必要的。

软件工程关注三个环节(软件开发过程、软件质量保证和软件项目管理)的行为,这些行为贯穿于软件生命周期中,并同时发生;而面向对象的软件工程(object-oriented software engineering, OOSE)是软件工程的一个专门学科。OOSE 将世界和各种系统看作是由相互联系和相互作用的对象构成。

在 20 世纪 80 年代,C++ 的迅速传播激起了对引导 OO 软件开发工作的开发方法的需求。提出了三个有影响的、在软件业内被广泛使用的 OO 软件开发方法,即 Booch 框图、对象建模技术(Object Modeling Technique, OMT)和用例工程。业界很快发现使用不同的方法将设计和实现的系统集成起来是一个非常大的挑战,原因就是不同的方法使用不同的建模概念和标记。为了解决这个问题,对象管理组织(Object Management Group, OMG)采用统一建模语言(Unified Modeling Language, UML)作为 OMG 标准。UML 是一个框图的集合,这些框图用于给一个 OO 系统的方方面面进行建模和设计。UML 框图用于需求分析阶段来帮助开发团队理解线性系统的业务流程,也作为设计说明的一部分用于设计阶段。本书大量展示了 UML 框图。

感谢德克萨斯大学阿灵顿分校的 David C. Kung 先生。敏捷过程、设计模式以及测试驱动的开发(test-driven development, TDD)激发了他写本书的巨大兴趣。敏捷过程强调团队合作、为改变而设计、对软件系统的每个小增量进行快速部署,以及与客户和用户共同开发。设计模式是对常见设计问题的有效解决方案,设计模式促进了软件重用并增进了团队沟通。总之,本文主要关注将 UML、设计模式、OO 软件测试、TDD 集成起来的敏捷统一方法学。

软件是你的作品,需要用心血去创作。我们可以用编程语言来准确反映我们的思想,我们应该做得更好,也可以做得更好。只要遵循科学的方法学,一定可以创作出好的软件作品。

本书由邢颖翻译,牟永敏统稿。由于译者水平有限,译文中的不当之处在所难免,真诚希望广大读者和同行不吝赐教,我们将不胜感激。

译者

2016 年 11 月

# 序 言

## 背景

计算机被广泛应用于人类社会的方方面面,它与基于其运行的应用软件一起发挥了很多不同的作用。因此,软件工程师的岗位需求也在井喷式增长。2006年3月的 *Money* 杂志将软件工程师评为美国最好的50个工作岗位的第一名。根据美国劳工统计局(Bureau of Labor Statistics, BLS)2010—2020预测,应用程序开发工程师岗位将从520 800增长到664 500(27.6%),而系统分析师的岗位将从544 400增长到664 800(22.1%)。为了能够满足对应用程序开发工程师或系统分析师的要求,接受软件工程方面的教育是非常有必要的。但是根据BLS发布的数据,2006年在软件工程领域只授予了160个学士学位和600个硕士学位,而在计算机科学领域则授予了10 289个学士学位和4512个硕士学位。因此,在软件工程人才的需求和供给之间有巨大的鸿沟,尤其是研究生。

很多人都不清楚软件工程的研究范围以及它到底有什么用途,这门学科也经常被误解。许多媒体似乎将软件工程定义为写Java程序。有的学生认为软件工程包括与软件相关的一切。另外一些人则认为软件工程就是画UML框图,这个我们后面会提到。多年前,在第一节面向对象的软件工程(object-oriented software engineering, OOSE)课后,一位学生对我说:“教授,你应该知道这门课对于我来说很容易,因为我们以前都画过太多的UML图了。”学期末,那个学生又来找我,他说:“教授,我想说我们很努力地学习,但我们学的是OO设计。这门课并不是我想象的画UML图。”所以到底什么是软件工程呢?作为一门学科,它包含为了明显提高软件的生产率和质量,同时降低成本和投放到市场的时间所需的工程过程、方法、质量保证和项目管理研究、教育及应用。OOSE是软件工程的一个分支,其特点是将世界和系统看成是相关联系和相互作用的独特视角。20世纪80年代C++语言的出现标志着OOSE时代的到来。从那时起,软件产品开始了其前所未有的世界范围的增长,并因统一建模语言(unified modeling language, UML)和统一过程(unified process, UP)的产生以及世界范围的应用而进一步加速增长。严格来说,软件过程描述了一些阶段以及在各个阶段应该做什么。它不会(详细)定义在每个阶段如何实施各种行为。类似UML这样的建模语言会定义用于交流和记录分析及设计思路的标记、语法和语义。UML和UP都很好,也很必需,但是还不够。这是因为还没有介绍如何产生画UML图的分析 and 设计思路。

## 动机

为了填补上文提到的鸿沟,我们需要一个方法学,或者称之为“食谱”。与过程不同,方法学是对于一些步骤的详细描述,或者是如何实施一些行为使得初学者也可以学习从而生成并部署所需的软件系统。没有方法学,一名刚入门的软件工程师可能会花费数年在职的训练才能学会OO设计、实现和测试技巧。

写本书的动机也包括作者被敏捷过程、设计模式以及测试驱动的开发(test-driven development, TDD)所激发的巨大兴趣。敏捷过程强调团队合作、为改变而设计、对软件系

统的每个小增量进行快速部署,以及与客户和用户共同开发。设计模式是对常见设计问题的有效解决方案。设计模式促进了软件重用并增进了团队沟通。TDD 鼓励可测试的软件,并需要在软件实现前产生测试脚本从而使得软件可以立刻被测试,经常被测试。

这类似开发一个游乐场的思路。整个过程包括如下阶段:计划、公共支持、分析设计、筹措资金、画施工图纸、施工、采购设备、安装设备、试运行、剪彩。但是,光了解整个过程是不够的。开发团队必须知道如何实施各阶段的行为。例如,计划行为包括产生初步概念、可行性分析、总体规划。主题公园团队必须知道如何实施这些行为。分析设计行为包括从利益相关者那里获取需求、实地调查、公园布局设计、公园不同区域主题设计、建模从而研究布局设计和主题、进行总体规划。与描述不同阶段行为的过程不同,一套方法学详细描述了一些步骤,或者是如何实施这些行为。

游乐场的开发是一个数年的项目,且花费达数十亿美元。投资者希望公园越早产生收益越好,但是如果按照上述方法开发,投资者必须一直等到公园完工。由于传统过程所施加的约束,总体规划一旦结束就不能被轻易修改。而一旦公园完工,如果公园不能满足利益相关者的期待,则其改变的代价很大。

敏捷开发旨在解决这些问题。使用敏捷开发,主题公园初步需求被迅速获取,并允许在开发过程中进行演化。然后从中提取出游乐和娱乐设施的需求,并将其认真归类。生成计划来在相对短的时期内开发和部署这些分类设施,即小增量的快速部署。因此,不同于一刀切的总体规划,开发过程每次只设计和部署一类设施。在这些设施部署好并投入运营后,就寻求反馈,并与利益相关者制订出关于开发计划、预算和时间表的需求变化,即联合开发。另外,应用架构设计模式来提高公园对改变的适应能力和质量,也即为改变而设计。强调团队合作是因为团队之间以及团队成员之间有效的协作能够确保这些设施按时无缝地开发和部署。敏捷过程有一些优点:投资者能较早获得收益,因为这些设施能按照计划投入运营而且是可行的;由于少量的设施是同时开发和部署的,可以容易改正错误并进行改变。

总之,本文主要关注将 UML、设计模式、OO 软件测试、TDD 集成起来的敏捷统一方法学。本书中的方法学叫做“统一方法学”,因为它使用 UML 作为建模语言并遵循敏捷统一过程。当然这并不意味着对于所有项目都将其他方法统一起来或被当成一个“统一”方法使用。

## 读者

本书的读者为计算机科学或软件工程的学生,以及软件开发的专门人士。本书尤其适合作为高年级本科生的基础教材,以及研究生课程和 IT 业专业培训课程的入门教材。本书包含过去十年间的很多内容,既有在美国以及全球的大学和公司里讲授的课程,也有行业内外软件工程项目的应用素材。这些素材让我能够近距离观察学生和软件工程师如何应用 UP、UML、设计模式和 TDD,以及他们所面临的困难。他们的反馈让我能够持续更新本书的内容。

## 组织

本书有 24 章,被分为以下 8 个部分:

**第 1 部分 引言与系统工程。**这部分由前 3 章构成,它提供了观察软件生命周期行为的视角。尤其是,它覆盖了软件过程模型、方法学的概念、过程与方法的区别以及系统工程。

**第 2 部分 分析和体系结构设计。**这部分介绍了计划阶段的行为,包括需求提取、领域建模和架构设计。

**第 3 部分 交互式系统的建模和设计。**这部分主要处理建模与交互系统设计问题,共包括 6 章。这 6 章介绍了如何根据需求识别出用例、如何对参与者-系统交互以及对象交互行为进行建模和设计、如何应用职责分配模式、如何得到作为设计蓝本的设计类框图,以及如何设计用户交互界面。

**第 4 部分 其他类型系统的建模和设计。**这部分包括 3 章,每章介绍一种类型系统的建模与设计。具体来说,第 13 章介绍事件驱动系统的建模与设计;第 14 章介绍迁移系统的建模与设计;第 15 章介绍基于业务规则的系统的建模与设计。

**第 5 部分 应用情景特定模式。**这部分由两章构成,主要介绍如何应用情景特定模式。其中使用了一个案例分析,即状态图编辑器的设计来帮助理解这个过程。

**第 6 部分 实现和质量保证。**这部分包括 3 章,包括实现时要考虑的问题、软件质量保证的概念和行为以及软件测试。

**第 7 部分 维护和配置管理。**这部分包括两章,内容覆盖了软件维护与软件配置管理。

**第 8 部分 项目管理和软件安全。**本书的最后一部分由两章构成,一章介绍软件项目管理,另一章介绍软件安全,包括涉及安全软件系统的建模与设计生命周期行为。

本书素材可以满足若干软件工程课程的需求,例如:

1. 第 1~3 部分与第 6~8 部分的一些主题对于面向对象的软件工程(OOSE)或软件工程课程的入门,都是一个很好的组合。既可以作为大学本科的课程,也可以作为入门的研究生课程。

2. 第 2 部分和第 5 部分以及其他部分的部分章节可以构成软件设计模式的课程。建议将上述 OOSE 课程作为这门课的先修课。但是可能有些国外的学生没学过 OOSE 这门课,在这种情况下,建议先使用 2~4 周的时间大概了解一下第 2 和第 3 部分介绍的方法,因为这些方法为应用模式提供了框架。

3. 第 6 和第 7 这两部分可以以很多种方式讲授。它们可以形成一门课——质量保证、测试和维护。也可以当成两门课来教,分别是软件质量保证,以及软件测试和维护。当然也可以作为三门课来教,分别是软件质量保证、软件测试和软件维护。

4. 第 13~15 章和第 19~20 章以及其他章的部分模式可以形成关于建模、设计、复杂系统的验证与确认这门课。

5. 第 1 和第 6~8 这几部分以及其他部分的部分章节可以形成软件项目管理这门课。

6. 最后,第 1 部分和第 2 部分以及第 24 章加上其他章节的部分模式和主题可以构成软件安全概论这门课程。教员可以增加素材使得课程内容更加丰富。

在网站 <http://www.mhhe.com/kung> 上可以找到教学辅助素材,包括 PowerPoint 教学幻灯片、随堂测试题和测试生成软件、测试问题的数据库、课程描述及要点、实验手册,以及软件工具。尤其对于没上过这门课的新教师来说,这些教学工具可以帮助减少备课的时间和 workload。

## 致谢

我想感谢我的众多学生,他们的提问、反馈、热情,以及将课程内容应用于实际项目的努

力都在不断激励我。学生们也阅读和使用本书以及教学素材。他们提供给我宝贵的反馈和改进的建议。有些人实地参加到了本书中某些方法的设计和实现中。有些人参加到了评估本书方法效果的实验中。很多人毕业后继续将本书方法应用于行业内,并和我分享他们的宝贵经历。我想感谢我的女儿 Jacquelyn Kung 帮我编辑草稿的部分章节。感谢 McGraw-Hill 环球出版社的工程和计算机科学部的 Raghu Srinivasan 提供给我宝贵的改进建议和对出版流程的指导。感谢评论家们提出的意见和建议。这些意见和建议对于本书的结构组织、内容显示以及很多方面都有极大的帮助。在漫长的写作过程中,我的妻子 Cindy Kung 和很多学术界及业内的同事都给予我持续的鼓励和无私的帮助,在此也对他们表示感谢。

本书的出版,首先要感谢我的妻子 Cindy Kung,她不仅在我写作本书的过程中给予了极大的支持和鼓励,还承担了大部分的家务,让我能够专心致志地完成本书的写作。同时,我也要感谢我的女儿 Jacquelyn Kung,她不仅在我写作本书的过程中给予了极大的支持和鼓励,还承担了大部分的家务,让我能够专心致志地完成本书的写作。此外,我还要感谢我的同事和朋友们,他们在我写作本书的过程中给予了极大的支持和鼓励,还承担了大部分的家务,让我能够专心致志地完成本书的写作。最后,我要感谢 McGraw-Hill 环球出版社的工程和计算机科学部的 Raghu Srinivasan,他不仅在我写作本书的过程中给予了极大的支持和鼓励,还承担了大部分的家务,让我能够专心致志地完成本书的写作。



# 目 录

<b>第 1 部分 引言与系统工程</b>	
<b>第 1 章 引言</b> .....	3
1.1 什么是软件工程 .....	3
1.2 为什么要用软件工程 .....	4
1.3 软件生命周期行为 .....	5
1.3.1 软件开发过程 .....	5
1.3.2 软件质量保证 .....	8
1.3.3 软件项目管理 .....	9
1.4 面向对象的软件工程 .....	10
1.4.1 面向对象的建模和设计语言 .....	10
1.4.2 面向对象的开发过程 .....	10
1.4.3 面向对象的开发方法 .....	11
1.4.4 OO 会取代传统的方法吗 .....	11
1.5 软件工程和计算机科学 .....	11
小结 .....	12
深入阅读 .....	12
章节复习问题 .....	13
练习 .....	13
<b>第 2 章 软件过程和方法</b> .....	14
2.1 系统开发的挑战 .....	14
2.2 软件过程 .....	15
2.3 瀑布模型的优势和问题 .....	16
2.4 软件开发是一个险恶问题 .....	16
2.5 软件过程模型 .....	17
2.5.1 原型过程 .....	18
2.5.2 演化过程 .....	18
2.5.3 螺旋模型 .....	18
2.5.4 统一过程 .....	19
2.5.5 个人软件过程 .....	20
2.5.6 团队软件过程 .....	24
2.5.7 敏捷过程 .....	26
2.6 软件开发方法 .....	30
2.6.1 过程和方法的区别 .....	30
2.6.2 方法的好处 .....	31
2.6.3 结构化方法 .....	32
2.6.4 经典的面向对象方法 .....	32
2.7 敏捷方法 .....	32
2.7.1 动态系统开发方法(DSDM) .....	34
2.7.2 Scrum .....	35
2.7.3 特征驱动的开发 .....	35
2.7.4 极限编程 .....	36
2.7.5 敏捷还是计划驱动 .....	36
2.8 本书中的过程和方法一览 .....	37
小结 .....	41
深入阅读 .....	41
章节复习问题 .....	42
练习 .....	42
<b>第 3 章 系统工程</b> .....	43
3.1 什么是系统 .....	43
3.2 什么是系统工程 .....	44
3.3 系统需求定义 .....	47
3.3.1 识别业务需求 .....	47
3.3.2 定义系统需求 .....	48
3.4 系统结构设计 .....	48
3.4.1 系统分解 .....	49
3.4.2 需求分配 .....	51
3.4.3 结构设计图 .....	52
3.4.4 子系统功能和接口规格说明 .....	55
3.5 子系统开发 .....	56
3.5.1 面向对象的上下文图 .....	56

3.5.2 面向对象的上下文图 的用途 .....	57	5.3.5 多重性和角色 .....	89
3.5.3 工程团队的协作 .....	58	5.3.6 聚合 .....	90
3.6 系统集成、测试和部署 .....	58	5.3.7 继承 .....	90
3.7 系统配置管理 .....	58	5.3.8 继承和多态 .....	91
小结 .....	60	5.3.9 关联类 .....	91
深入阅读 .....	60	5.4 领域建模的步骤 .....	93
章节复习问题 .....	60	5.4.1 收集应用领域信息 .....	94
练习 .....	60	5.4.2 头脑风暴 .....	94
<b>第 2 部分 分析和体系结构设计</b>		5.4.3 对头脑风暴结果 分类 .....	95
<b>第 4 章 获取软件需求 .....</b>	<b>65</b>	5.4.4 领域模型可视化 .....	98
4.1 什么是需求获取 .....	65	5.4.5 领域建模审查 清单 .....	102
4.2 获取需求的重要性 .....	67	5.5 综合 .....	103
4.3 获取需求的挑战 .....	67	5.6 领域建模的方针 .....	105
4.4 需求的类型 .....	68	5.7 应用敏捷原则 .....	106
4.5 获取需求的步骤 .....	69	5.8 领域建模的工具支持 .....	107
4.5.1 收集信息 .....	70	小结 .....	108
4.5.2 构建分析模型 .....	73	深入阅读 .....	109
4.5.3 获取需求和约束 .....	74	章节复习问题 .....	109
4.5.4 需求规格标准 .....	77	练习 .....	109
4.5.5 进行可行性研究 .....	77	<b>第 6 章 架构设计 .....</b>	<b>111</b>
4.5.6 审查需求规格说 明书 .....	77	6.1 什么是架构设计 .....	111
4.6 应用敏捷原则 .....	79	6.2 架构设计的重要性 .....	112
4.7 需求管理和工具 .....	80	6.3 架构设计过程 .....	112
小结 .....	81	6.3.1 确定架构设计 目标 .....	113
深入阅读 .....	81	6.3.2 确定系统类型 .....	114
章节复习问题 .....	81	6.3.3 应用架构样式 .....	117
练习 .....	82	6.3.4 进行定制的架构 设计 .....	124
<b>第 5 章 领域建模 .....</b>	<b>84</b>	6.3.5 明确子系统功能和 接口 .....	124
5.1 什么是领域建模 .....	84	6.3.6 审查架构设计 .....	125
5.2 为什么要进行领域建模 .....	85	6.4 架构样式和包图 .....	125
5.3 面向对象和类图 .....	85	6.5 应用软件设计准则 .....	126
5.3.1 外延定义和意向 定义 .....	85	6.5.1 什么是软件设计 准则 .....	127
5.3.2 类和对象 .....	86	6.5.2 为改变而设计 .....	127
5.3.3 对象和属性 .....	87		
5.3.4 关联 .....	88		

6.5.3 关注点分离	128	步骤	160
6.5.4 信息隐藏	129	8.3.1 初始化一个两列的表	161
6.5.5 高内聚	129	8.3.2 明确参与者-系统交互的步骤	161
6.5.6 低耦合	129	8.3.3 检查参与者-系统交互的规约	161
6.5.7 保持简单和直接	130	8.4 明确可替换的流	162
6.6 架构设计的方针	131	8.5 使用用户界面原型	163
6.7 架构设计和设计模式	131	8.6 不要显示异常处理	166
6.8 应用敏捷原则	132	8.7 用例的先决条件和后置条件	166
小结	132	8.8 包含其他用例	167
深入阅读	132	8.9 用其他用例继续	167
章节复习问题	133	8.10 常见问题	168
练习	133	8.11 应用敏捷原则	170
<b>第3部分 交互式系统的建模和设计</b>			
<b>第7章 从需求获取用例</b> 137			
7.1 什么是参与者	137	小结	170
7.2 什么是用例	138	深入阅读	171
7.3 业务过程、操作和活动	138	章节复习问题	171
7.4 从需求获取用例的步骤	140	练习	171
7.4.1 识别用例	140	<b>第9章 对象交互建模</b> 172	
7.4.2 明确用例范围	146	9.1 什么是对象交互建模	172
7.4.3 用例上下文可视化	147	9.2 UML 顺序图	173
7.4.4 检查用例规约	151	9.2.1 概念和表示法	173
7.4.5 将用例分配给迭代	152	9.2.2 展示类的实例	173
7.5 导出用例的方针	153	9.2.3 顺序图说明	174
7.6 应用敏捷原则	155	9.2.4 顺序图用于分析和设计	176
7.7 用例建模的工具支持	155	9.2.5 正确使用表示方法	178
小结	157	9.3 对象交互建模的步骤	179
深入阅读	157	9.3.1 收集业务过程的信息	180
章节复习问题	157	9.3.2 识别非普通步骤	180
练习	158	9.3.3 为非普通步骤书写场景	181
<b>第8章 参与者-系统交互建模</b> 159			
8.1 什么是参与者-系统交互建模	159	9.3.4 构建场景表	183
8.2 参与者-系统交互建模的重要性	160	9.3.5 怎么写场景	183
8.3 参与者-系统交互建模的步骤	160	9.3.6 从场景表得到顺	

序图·····	188	10.7 创建者模式·····	213
9.3.7 对象交互建模检查		10.7.1 什么是创建者·····	213
清单·····	194	10.7.2 应用创建者	
9.4 应用敏捷原则·····	195	模式·····	214
9.5 对象交互建模的工具支持·····	197	10.7.3 创建者模式的	
小结·····	197	优势·····	215
深入阅读·····	197	10.7.4 何时应用创建者	
章节复习问题·····	197	模式·····	215
练习·····	198	小结·····	216
<b>第 10 章 应用职责分配模式·····</b>	<b>199</b>	深入阅读·····	216
10.1 什么是设计模式·····	199	章节复习问题·····	216
10.2 为什么要用设计模式·····	200	练习·····	217
10.3 情景特定模式和职责分配		<b>第 11 章 获取设计类图·····</b>	<b>219</b>
模式·····	200	11.1 什么是设计类图·····	220
10.4 模式规约·····	201	11.2 设计类图的用途·····	220
10.5 控制器模式·····	202	11.3 获取设计类图的步骤·····	221
10.5.1 激励示例·····	202	11.3.1 识别类·····	221
10.5.2 什么是控制器·····	204	11.3.2 识别方法·····	222
10.5.3 应用控制器		11.3.3 识别属性·····	224
模式·····	204	11.3.4 类之间的关系·····	226
10.5.4 控制器的类型·····	206	11.3.5 识别关系·····	226
10.5.5 跟踪记录用例		11.3.6 设计类图检查	
状态·····	206	清单·····	228
10.5.6 臃肿的控制器·····	207	11.4 用包图组织类·····	228
10.5.7 比较不同的		11.5 应用敏捷原则·····	230
设计·····	208	11.6 设计类图的工具支持·····	231
10.5.8 何时应用控制器		小结·····	231
模式·····	209	深入阅读·····	231
10.5.9 使用控制器的		章节复习问题·····	231
方针·····	210	练习·····	231
10.6 专家模式·····	210	<b>第 12 章 用户接口设计·····</b>	<b>232</b>
10.6.1 信息专家·····	210	12.1 什么是用户接口设计·····	232
10.6.2 应用专家模式·····	211	12.2 用户接口设计为什么	
10.6.3 包含不止一个对象		重要·····	233
的专家模式·····	212	12.3 图形用户接口组件·····	234
10.6.4 何时应用专家		12.3.1 容器组件·····	235
模式·····	213	12.3.2 输入、输出和信息	
10.6.5 使用专家模式的		展示组件·····	235
方针·····	213	12.3.3 使用 GUI 组件的	

方针.....	237	用途.....	261
12.4 用户接口设计过程.....	237	13.4.5 将状态迁移表转换	
12.4.1 案例研究：为状态		为分析状态图.....	262
图编辑器设计用户		13.4.6 将分析状态图转换	
接口.....	238	为设计状态图.....	264
12.4.2 识别系统主要显示		13.4.7 状态建模检查	
方式.....	238	清单.....	265
12.4.3 生成布局设计		13.5 状态模式.....	265
草图.....	240	13.5.1 约定的方法.....	265
12.4.4 明确交互行为.....	242	13.5.2 什么是状态	
12.4.5 构建原型.....	242	模式.....	266
12.4.6 用户评估用户接口		13.5.3 应用状态模式.....	266
设计.....	243	13.6 实时系统的建模与设计.....	269
12.4.7 用户接口设计检查		13.6.1 转换图式.....	270
清单.....	244	13.6.2 定时状态机.....	271
12.5 设计用户支持功能.....	245	13.6.3 中断处理.....	272
12.6 用户接口设计的方针.....	245	13.7 应用敏捷原则.....	273
12.7 应用敏捷原则.....	247	13.8 对象状态建模的工具	
12.8 用户接口设计的工具		支持.....	274
支持.....	247	小结.....	274
小结.....	248	深入阅读.....	275
深入阅读.....	248	章节复习问题.....	275
章节复习问题.....	248	练习.....	275
练习.....	249	<b>第 14 章 转换型系统的活动建模</b> .....	278
<b>第 4 部分 其他类型系统的建模和设计</b>		14.1 什么是活动建模.....	278
<b>第 13 章 事件驱动系统的对象状态</b>		14.2 为什么使用活动建模.....	279
<b>建模</b> .....	253	14.3 活动建模：技术背景.....	279
13.1 什么是对象状态建模.....	253	14.3.1 流程图.....	280
13.2 为什么要进行对象状态		14.3.2 佩特里网.....	280
建模.....	254	14.3.3 数据流图.....	281
13.3 基本定义.....	254	14.4 UML 活动图.....	282
13.4 对象状态建模的步骤.....	255	14.5 活动建模的步骤.....	283
13.4.1 收集和分类状态		14.5.1 识别活动和工	
行为信息.....	256	作流.....	284
13.4.2 构建领域模型来		14.5.2 产生一个初步活	
展示上下文.....	258	动图.....	286
13.4.3 构建状态迁移表 .....	260	14.5.3 引入条件分支、分叉	
13.4.4 状态迁移表的		和连接.....	287
		14.5.4 精炼复杂的活动 .....	287

14.5.5 活动建模审核检 查表.....	288		
14.6 与其他图之间的关系.....	288	<b>第 5 部分 应用情景特定模式</b>	
14.7 应用敏捷原则.....	289	<b>第 16 章 应用模式来设计状态图</b>	
14.8 活动建模的工具支持.....	289	编辑器 .....	307
小结.....	289	16.1 应用模型的过程.....	308
深入阅读.....	290	16.2 案例研究：状态图编 辑器.....	310
章节复习问题.....	290	16.3 处理复杂的结构.....	311
练习.....	290	16.3.1 表示递归的整体- 部分结构.....	311
<b>第 15 章 基于规则的系统的建模与     设计 .....</b>	291	16.3.2 使用策略提供布局 选择.....	315
15.1 什么是决策表.....	292	16.3.3 用迭代器访问复杂 的结构.....	316
15.2 决策表的用处.....	293	16.3.4 通过访问者分析复 杂的结构.....	317
15.3 系统化的决策表构造.....	293	16.3.5 通过备忘录存储和 恢复对象状态.....	320
15.4 渐进式决策表构造.....	294	16.4 创建和构造复杂的对象.....	321
15.5 检查所需的属性.....	295	16.4.1 创建系列产品.....	321
15.6 决策表合并.....	296	16.4.2 构建大型的复杂 对象.....	324
15.7 根据决策表生成代码.....	296	16.4.3 通过享元重用 对象.....	326
15.8 应用解释器模式.....	297	16.5 图形用户界面的设计和 显示.....	327
15.8.1 定义业务规则 语法.....	297	16.5.1 跟踪编辑状态.....	327
15.8.2 在类图中表示 规则.....	298	16.5.2 响应编辑事件.....	328
15.8.3 构建解析器和变量 查找上下文.....	299	16.5.3 转换接口.....	330
15.8.4 解释业务规则.....	299	16.5.4 提供上下文相关的 帮助.....	333
15.8.5 动态更新规则.....	300	16.5.5 使用装饰增强显示 能力.....	335
15.8.6 解释方法的 优点.....	300	16.6 应用敏捷原则.....	338
15.9 在测试驱动开发中使用 决策表.....	300	小结.....	338
15.10 决策树 .....	301	深入阅读.....	338
15.11 应用敏捷原则 .....	301	章节复习问题.....	339
小结.....	302	练习.....	339
深入阅读.....	302		
章节复习问题.....	302		
练习.....	303		

<b>第 17 章 应用模式来设计持久性</b>	
<b>框架</b> .....	340
17.1 直接访问数据库会出现的 问题 .....	340
17.2 通过桥隐藏持久存储 .....	341
17.3 将数据库请求封装为 命令 .....	344
17.4 用远程代理隐藏网络 访问 .....	348
17.5 用模板方法共享通用 代码 .....	351
17.6 通过工厂方法检索不同 的对象 .....	353
17.7 用原型减少类的数量 .....	355
17.8 应用敏捷原则 .....	358
小结 .....	358
深入阅读 .....	358
章节复习问题 .....	358
练习 .....	358
 <b>第 6 部分 实现和质量保证</b>	
<b>第 18 章 实现方面的考虑</b> .....	363
18.1 编码标准 .....	363
18.1.1 什么是编码 标准 .....	363
18.1.2 为什么要建立 编码标准 .....	367
18.1.3 代码审查清单 .....	367
18.1.4 编码标准实施 准则 .....	367
18.2 组织实现部件 .....	369
18.3 根据设计生成代码 .....	370
18.3.1 实现类和接口 .....	370
18.3.2 从顺序图到方法 代码骨架 .....	370
18.3.3 实现关联关系 .....	371
18.4 给团队成员分配任务 .....	372
18.5 结对编程 .....	372
18.6 测试驱动开发 .....	373
18.6.1 测试驱动开发的 工作流程 .....	373
18.6.2 测试驱动开发的 优点 .....	375
18.6.3 潜在问题 .....	375
18.7 应用敏捷原则 .....	376
18.8 对实现的工具支持 .....	376
小结 .....	376
深入阅读 .....	377
章节复习问题 .....	377
练习 .....	377
<b>第 19 章 软件质量保证</b> .....	379
19.1 软件质量保证的益处 .....	379
19.2 软件质量属性 .....	379
19.3 质量测量和度量 .....	381
19.3.1 质量测量和度量的 实用性 .....	382
19.3.2 常规质量度量 .....	382
19.3.3 对面向对象软件 重用传统度量 .....	387
19.3.4 面向对象质量 度量 .....	387
19.4 软件验证与确认技术 .....	389
19.4.1 审查 .....	389
19.4.2 走查 .....	390
19.4.3 同行审查 .....	391
19.5 生命周期中的验证和 确认 .....	392
19.6 软件质量保证功能 .....	393
19.6.1 过程和标准的 定义 .....	394
19.6.2 质量管理 .....	396
19.6.3 过程改进 .....	397
19.7 应用敏捷原则 .....	398
19.8 SQA 的工具支持 .....	399
小结 .....	399
深入阅读 .....	400
章节复习问题 .....	400
练习 .....	400

<b>第 20 章 软件测试</b> .....	401	20.8.4 用 httpUnit 进行 Web 应用程序测试.....	422
20.1 什么是软件测试.....	402	20.9 非功能性需求的测试.....	422
20.2 为什么进行软件测试.....	402	20.9.1 性能和压力 测试.....	422
20.3 传统的黑盒测试.....	403	20.9.2 安全测试.....	423
20.3.1 功能性测试: 一个 例子.....	403	20.9.3 测试用户界面.....	423
20.3.2 等价类划分.....	404	20.10 生命周期中的软件测试 ..	424
20.3.3 边界值分析.....	405	20.11 回归测试 .....	426
20.3.4 因果分析.....	407	20.12 什么时候停止 测试 .....	426
20.4 传统的白盒测试.....	407	20.13 应用敏捷原则 .....	427
20.4.1 基路径测试.....	408	20.14 测试的工具支持 .....	427
20.4.2 圈复杂度.....	409	小结.....	427
20.4.3 流图测试覆盖 准则.....	409	深入阅读.....	428
20.4.4 测试循环.....	410	章节复习问题.....	428
20.4.5 数据流测试.....	411	练习.....	428
20.4.6 数据流测试的覆盖 准则.....	412		
20.4.7 过程间数据流 测试.....	412		
20.5 测试覆盖率.....	413	<b>第 7 部分 维护和配置管理</b>	
20.6 一个通用的软件测试 过程.....	413	<b>第 21 章 软件维护</b> .....	433
20.7 面向对象的软件测试.....	415	21.1 什么是软件维护.....	433
20.7.1 基于用例的 测试.....	415	21.2 软件变化的因素.....	434
20.7.2 用 ClassBench 进行 对象状态测试.....	416	21.3 系统演化的雷曼定律.....	434
20.7.3 测试类的层次 结构.....	418	21.4 软件维护的类型.....	435
20.7.4 测试异常处理 能力.....	419	21.5 软件维护过程和活动.....	435
20.8 测试 Web 应用程序 .....	420	21.5.1 维护过程模型.....	436
20.8.1 Web 应用程序测试 的面向对象模型 ..	420	21.5.2 理解程序.....	437
20.8.2 使用面向对象模型进 行静态分析.....	421	21.5.3 变更识别和 分析.....	437
20.8.3 使用面向对象模型生 成测试用例.....	421	21.5.4 配置变更控制.....	439
		21.5.5 变更实施、测试和 交付.....	440
		21.6 逆向工程.....	440
		21.6.1 逆向工程 workflow .....	440
		21.6.2 逆向工程的用途 .....	440
		21.6.3 逆向工程: 案例 研究.....	441



21.7 软件再工程.....	441	23.1.1 项目形式.....	466
21.7.1 再工程的目标.....	441	23.1.2 团队结构.....	467
21.7.2 软件再工程的 过程.....	442	23.2 工作量评估方法.....	468
21.7.3 软件再工程: 案例 研究.....	443	23.2.1 功能点方法.....	468
21.8 软件维护的模式.....	444	23.2.2 COCOMO II 模型 .....	469
21.8.1 用外观模式简化 客户端接口.....	444	23.2.3 Delphi 评估方法 .....	474
21.8.2 用中介者模式简化 组件交互.....	445	23.2.4 敏捷评估.....	474
21.8.3 软件维护的 模式.....	445	23.3 项目计划和调度.....	476
21.9 应用敏捷原则.....	449	23.3.1 PERT 图.....	476
21.10 软件维护的工具支持 .....	449	23.3.2 Gantt 图和人员 分配.....	477
小结.....	450	23.3.3 敏捷规划.....	478
深入阅读.....	450	23.4 风险管理.....	479
章节复习问题.....	450	23.4.1 风险识别.....	479
练习.....	450	23.4.2 风险分析和优先级 划分.....	480
<b>第 22 章 软件配置管理</b> .....	452	23.4.3 风险管理规划.....	482
22.1 软件生命周期的基准.....	452	23.4.4 风险解决和监控 .....	482
22.2 什么是软件配置管理.....	453	23.5 过程改进.....	482
22.3 为什么需要软件配置 管理.....	454	23.6 应用敏捷原则.....	484
22.4 软件配置管理的功能.....	454	23.7 项目管理的工具支持.....	484
22.4.1 软件配置识别.....	455	小结.....	485
22.4.2 软件配置变更 控制.....	456	深入阅读.....	485
22.4.3 软件配置审核.....	457	章节复习问题.....	486
22.4.4 软件配置状态的 审计.....	458	练习.....	486
22.5 敏捷项目中的配置管理.....	458	<b>第 24 章 软件安全</b> .....	488
22.6 软件配置管理工具.....	458	24.1 什么是软件安全.....	489
小结.....	460	24.2 安全要求.....	489
深入阅读.....	460	24.3 安全软件设计原则.....	490
章节复习问题.....	460	24.4 安全软件设计模式.....	491
练习.....	461	24.5 软件安全的 7 个最佳 实践.....	493
<b>第 8 部分 项目管理和软件安全</b>		24.6 通过攻击树进行风险 分析.....	494
<b>第 23 章 软件项目管理</b> .....	465	24.7 生命周期中的软件安 全性.....	494
23.1 项目组织.....	466	24.7.1 规划阶段的安全 .....	495