普通高等教育
软件工程 “十二五”规划教材

12th Five-Year Plan Textbooks
of Software Engineering

UML与Rose建模实用教程

吕云翔 赵天宇 丛硕 ◎ 编著

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

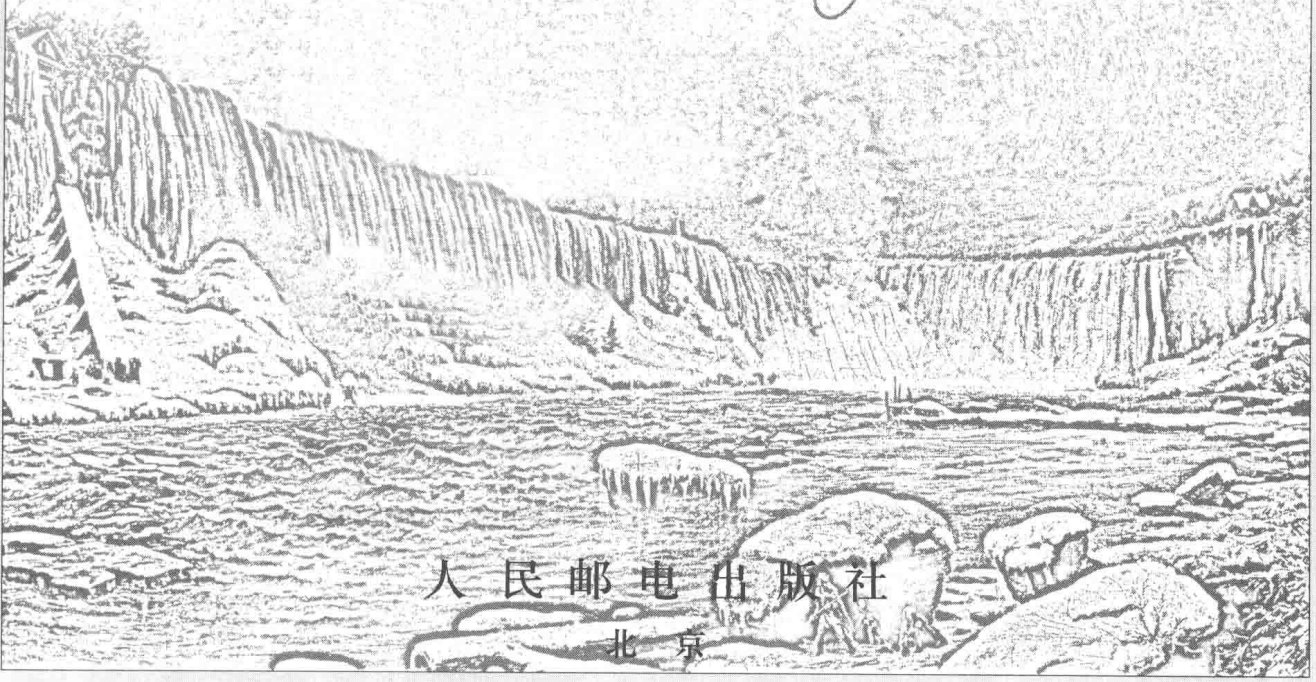
普通高等教育
软件工程“十二五”规划教材

12th Five-Year Plan Textbooks
of Software Engineering

UML与Rose建模实用教程

吕云翔 赵天宇 丛硕 ◎ 编著

UML and Rational Rose Modeling Tutorial



人民邮电出版社
北京

图书在版编目（CIP）数据

UML与Rose建模实用教程 / 吕云翔，赵天宇，丛硕编著. -- 北京 : 人民邮电出版社，2016.4
普通高等教育软件工程“十二五”规划教材
ISBN 978-7-115-41805-0

Ⅰ. ①U… Ⅱ. ①吕… ②赵… ③丛… Ⅲ. ①面向对象语言－程序设计－高等学校－教材 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2016)第031591号

内 容 提 要

本书介绍了使用 UML 进行软件建模的基础知识以及使用 Rational Rose 进行 UML 建模的基本方法。

本书主要分为三个部分。第一部分简要介绍了软件工程的产生、发展历史及重要作用，并对面向对象方法的概念和原则加以阐释，由此推出 UML 的概念和作用，还介绍了 UML 建模的重要工具——Rational Rose。第二部分从 UML 概念模型出发，对每种 UML 图进行了具体介绍，涵盖了 UML 中的用例图、包图、类图、对象图、协作图、顺序图、状态图、组件图、活动图、部署图。第三部分首先结合 UML 的实用过程，介绍了统一软件开发过程的相关概念；然后，通过小型网上书店系统、小型二手货交易系统、汽车服务管理系统三个具体案例，使读者更深刻地认识在实际开发过程中 UML 的使用。

本书既可以作为软件从业人员的学习指导用书，也可以作为高等院校计算机与软件相关专业的教材。

◆ 编　　著　吕云翔　赵天宇　丛　硕
责任编辑　武恩玉
责任印制　沈　蓉　彭志环

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京昌平百善印刷厂印刷

◆ 开本：787×1092　1/16
印张：16　　2016 年 4 月第 1 版
字数：418 千字　　2016 年 4 月北京第 1 次印刷

定价：39.80 元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315

前　言

随着人类迈进信息时代，计算机软件越来越多地参与到系统控制、资源调配、数据分析等重要工作中来。因此，国家和社会对软件产品的安全、稳定和效率提出了更高的要求。而软件的开发又需要人的直接参与，很难工程化，这导致计算机软件开发和维护过程中难免会遇到一系列问题，因此软件工程学科应运而生。

在软件工程学科发展历程中，统一建模语言（Unified Modeling Language，UML）作为一种定义良好、易于表达、功能强大且使用范围广泛的标准建模语言，获得了工业界、科技界和应用界的广泛支持。它融入了软件工程领域许多新兴的思想、方法和技术，支持从需求分析开始的软件开发的全过程。时至今日，UML 已然成为世界公认的面向对象标准建模语言。熟练使用 UML 建模，是软件工程师、软件设计师的基本要求。它可以起到引领读者认识软件工程核心要素，体系化、工程化思考软件开发过程、清晰各部分逻辑的重要作用。而 Rational Rose 作为绘制 UML 最得心应手的工具，学习它将简化读者的学习过程，并规范读者的绘图习惯，最终提升读者分析和设计的效率。

本书旨在让有一定程序制作能力、未来有志从事软件开发和设计工作的读者学习标准的建模语言，获得设计软件的逻辑能力以及对软件内部各种组织结构的表达能力，为今后的课程学习和职业前途打下坚实的基础。与此同时，本书结合了互联网时代软件思想更新快的特点，坚持使用最新的 UML 标准；重点介绍使用新技术的案例，避免使用老式的、可能导致潜在风险的分析和设计方法。

本书的最后几章，给出了更多的软件工程具体实例，使读者能从实际应用中体会 UML 的简洁性和重要性，加深对 UML 的印象；让读者能够了解到更多的工程相关知识，提高软件分析和设计的综合能力。

本书中大多数的图均为 Rational Rose v7.0 绘制，少量 Rose 无法绘制的 UML2 标准的图例是使用 Enterprise Artifact 绘制的，在文中已经注明。

由于 UML 和 Rational Rose 的教学方法本身还在探索之中，加之我们的水平和能力有限，本书难免有疏漏之处。恳请广大读者给予批评指正，也希望各位能将实践过程中的经验和心得与我们交流（yunxianglu@hotmail.com）。

编　者

目 录

第一部分 概述

第 1 章 软件工程与面向对象方法 … 2

1.1 软件工程简介 …… 2
1.1.1 软件工程的发展过程 …… 2
1.1.2 软件工程的目标和原则 …… 3
1.2 面向对象方法简介 …… 3
1.2.1 什么是面向对象方法 …… 3
1.2.2 面向对象方法的发展历史 …… 4
1.2.3 面向对象方法的基本概念 …… 4
1.2.4 面向对象方法的优势 …… 6
小结 …… 7
习题 …… 7

第 2 章 统一建模语言 UML …… 9

2.1 软件建模简介 …… 9
2.1.1 什么是模型 …… 9
2.1.2 建模的重要性 …… 9
2.1.3 建模的基本原理 …… 10
2.2 UML 简述 …… 10
2.3 UML 的发展历史 …… 11
2.3.1 UML 的出现背景 …… 11
2.3.2 UML 的诞生及其标准化 …… 12
2.3.3 UML 2 规范 …… 12
2.4 UML 的目标与应用范围 …… 13
2.4.1 UML 的目标 …… 13
2.4.2 UML 的应用范围 …… 14
小结 …… 14
习题 …… 14

第 3 章 Rational Rose 工具概述 … 16

3.1 Rose 简述 …… 16
3.1.1 何谓 Rose …… 16
3.1.2 Rational Rose 对 UML 的支持 …… 17
3.2 Rational Rose 的安装 …… 18
3.2.1 安装前的准备 …… 18
3.2.2 安装过程 …… 19
3.3 Rational Rose 的使用 …… 25
3.3.1 Rational Rose 界面介绍 …… 25
3.3.2 Rose 的基本操作 …… 31
小结 …… 38
习题 …… 38

第二部分 UML 概念详解

第 4 章 UML 概念模型 …… 40

4.1 构造块 …… 40
4.1.1 事物 …… 40
4.1.2 关系 …… 42
4.1.3 图 …… 43
4.2 通用机制 …… 45
4.2.1 规格说明 …… 45
4.2.2 修饰 …… 45
4.2.3 通用划分 …… 45
4.2.4 扩展机制 …… 46
4.3 “4+1”架构 …… 47
4.3.1 “4+1”架构的概念和组成 …… 47
4.3.2 “4+1”架构要解决的问题 …… 48
4.3.3 运用“4+1”视图方法进行软件架构设计 …… 49
小结 …… 50
习题 …… 50

试读结束：需要全本请在线购买： www.ertongbook.com

第 5 章　用例图 …… 52

5.1　用例图的基本概念 …… 52
5.2　参与者 …… 53
5.2.1　参与者的概念 …… 53
5.2.2　确定参与者 …… 53
5.2.3　参与者的泛化关系 …… 54
5.3　用例 …… 55
5.3.1　用例的概念 …… 55
5.3.2　用例与参与者 …… 55
5.3.3　用例的特征 …… 56
5.3.4　用例的粒度 …… 57
5.4　用例之间的关系 …… 58
5.4.1　泛化关系 …… 58
5.4.2　依赖关系 …… 59
5.5　用例描述与文档 …… 60
5.5.1　用例描述概述 …… 60
5.5.2　前置条件与后置条件 …… 61
5.5.3　事件流 …… 61
5.5.4　补充约束 …… 62
5.5.5　用例文档实践 …… 62
5.6　应用用例图建模 …… 63
5.6.1　用例图建模技术 …… 63
5.6.2　用例图使用要点 …… 65
5.7　实验：使用 Rose 绘制用例图 …… 65
5.7.1　用例图的 Rose 操作 …… 65
5.7.2　绘制机票预订系统的用例图 …… 68
小结 …… 70
习题 …… 70

第 6 章　类图与对象图 …… 73

6.1　类图的基本概念 …… 73
6.2　类图的组成元素 …… 74
6.2.1　类 …… 74
6.2.2　接口 …… 77
6.2.3　类图中的关系 …… 77
6.2.4　涉及类的其他概念 …… 83
6.3　类图的实例——对象图 …… 85
6.3.1　对象图概述 …… 86
6.3.2　对象图的组成元素 …… 86
6.4　类图与对象图的建模技术 …… 87
6.4.1　类图的建模技术 …… 87
6.4.2　正向工程与逆向工程 …… 88
6.4.3　对象图的建模技术 …… 89
6.4.4　面向对象设计的原则 …… 89
6.5　实验：使用 Rose 绘制类图 …… 92
6.5.1　类图的 Rose 操作 …… 92
6.5.2　绘制机票预订系统的类图 …… 97
小结 …… 101
习题 …… 101

第 7 章　包图 …… 104

7.1　包图的基本概念 …… 104
7.2　包 …… 104
7.2.1　包的概念 …… 105
7.2.2　包的作用 …… 107
7.2.3　元素的分包原则 …… 107
7.3　包的依赖关系 …… 107
7.4　包图的建模技术 …… 109
7.5　实验：使用 Rose 绘制包与包图 …… 110
7.5.1　包图的 Rose 操作 …… 110
7.5.2　使用包组织 UML 图中的元素 …… 112
小结 …… 112
习题 …… 113

第 8 章　顺序图 …… 115

8.1　顺序图的概念 …… 115
8.2　顺序图的组成元素 …… 116
8.2.1　对象 …… 116
8.2.2　生命线 …… 116
8.2.3　激活 …… 117
8.2.4　消息 …… 117
8.3　（*）UML 2 中的“片段”概念 …… 119
8.4　顺序图建模技术 …… 120
8.5　（*）顺序图的变体——时间图 …… 121
8.6　实验：使用 Rose 绘制顺序图 …… 122
8.6.1　顺序图的 Rose 操作 …… 122
8.6.2　绘制登录用例的顺序图 …… 124
小结 …… 125
习题 …… 125

第 9 章　协作图 ……127

9.1　协作图的概念 ……127
9.2　协作图的组成元素 ……128
9.2.1　对象 ……128
9.2.2　链 ……129
9.2.3　消息 ……129
9.3　协作图与顺序图 ……130
9.4　协作图建模技术 ……131
9.5　(*) UML 2 中的通信图 ……131
9.6　实验：使用 Rose 绘制协作图 ……132
9.6.1　协作图的 Rose 操作 ……133
9.6.2　绘制查询航班用例的协作图 ……134
小结 ……135
习题 ……135

第 10 章　状态图 ……138

10.1　状态图的基本概念 ……138
10.1.1　状态机 ……138
10.1.2　状态图 ……139
10.2　状态图的组成 ……140
10.2.1　简单状态 ……140
10.2.2　转换 ……141
10.2.3　伪状态 ……144
10.3　复合状态 ……145
10.4　状态图的建模技术 ……147
10.5　实验：使用 Rose 绘制状态图 ……147
10.5.1　状态图的 Rose 操作 ……148
10.5.2　绘制航班类的状态图 ……152
小结 ……154
习题 ……154

第 11 章　活动图 ……156

11.1　活动图基本概念 ……156
11.2　活动图组成元素 ……157
11.2.1　动作和活动节点 ……157
11.2.2　开始和终止 ……158
11.2.3　控制流 ……158
11.2.4　判断节点 ……158
11.2.5　合并节点 ……159
11.2.6　泳道 ……159
11.3　活动图的高级概念 ……160
11.3.1　并发 ……160
11.3.2　分叉节点 ……161
11.3.3　结合节点 ……161
11.3.4　对象流 ……161
11.3.5　扩展区域 ……162
11.4　活动图建模技术 ……162
11.5　活动图的进一步说明 ……163
11.6　实验：使用 Rose 绘制活动图 ……163
11.6.1　活动图的 Rose 操作 ……163
11.6.2　绘制用户购票的活动图 ……166
小结 ……167
习题 ……167

第 12 章　组件图 ……170

12.1　组件图的基本概念 ……170
12.2　组件图的组成元素 ……171
12.2.1　组件 ……171
12.2.2　接口 ……172
12.2.3　组件图中的关系 ……173
12.2.4　(*) Rose 中的特殊组件 ……173
12.2.5　(*) UML 2 中组件的嵌套 ……175
12.3　组件图的建模技术 ……175
12.4　实验：使用 Rose 绘制组件图 ……176
12.4.1　组件图的 Rose 操作 ……176
12.4.2　绘制机票预订系统的组件图 ……178
小结 ……179
习题 ……179

第 13 章　部署图 ……181

13.1　部署图的基本概念 ……181
13.2　部署图的组成元素 ……181
13.2.1　节点 ……182
13.2.2　部署图中的关系 ……182
13.3　部署图建模技术 ……183
13.4　实验：使用 Rose 绘制部署图 ……183

13.4.1 部署图的 Rose 操作……183
13.4.2 绘制机票预订系统的部署图……185
小结……185
习题……186

第三部分 建模过程剖析

第 14 章 统一软件开发过程……189

14.1 统一软件开发过程概述……189
14.1.1 什么是软件开发过程……189
14.1.2 统一软件开发过程简介……190
14.1.3 统一软件开发过程发展历程……190
14.2 过程总览……191
14.3 阶段和迭代——时间维度……191
14.3.1 起始阶段……192
14.3.2 细化阶段……193
14.3.3 构建阶段……194
14.3.4 转化阶段……195
14.3.5 迭代……195
14.4 过程的静态结构……196
14.4.1 工作者……196
14.4.2 活动……196
14.4.3 制品……197
14.4.4 工作流……197
14.5 核心工作流……197
14.6 在统一软件开发过程中使用 UML……200
14.6.1 起始阶段常用 UML 图……201
14.6.2 细化阶段常用 UML 图……202
14.6.3 构建阶段常用 UML 图……202
14.6.4 转化阶段常用 UML 图……203
小结……204
习题……205

第 15 章 小型网上书店系统……207

15.1 小型网上书店系统的需求分析……207
15.1.1 项目背景描述……207
15.1.2 系统需求分析……207
15.1.3 用户管理模块……208
15.1.4 订单管理模块……208
15.1.5 书目管理模块……208
15.2 系统的 UML 基本模型……209
15.2.1 需求分析阶段模型……209
15.2.2 基本动态模型……210
15.3 类的设计与实现……212
15.3.1 系统设计类……213
15.3.2 类的实现……214
15.4 系统的组件图和部署图……215
15.4.1 系统的组件图……215
15.4.2 系统的部署图……215

第 16 章 小型二手货交易系统……217

16.1 需求分析部分……217
16.1.1 子系统划分……218
16.1.2 系统功能需求……218
16.1.3 非功能需求……221
16.2 系统设计部分……221
16.2.1 系统设计类图……221
16.2.2 关键用例的动态模型……224
16.2.3 类的代码框架……227

第 17 章 汽车服务管理系统……229

17.1 汽车服务管理系统的需求分析……229
17.1.1 系统功能需求……229
17.1.2 车辆及路线管理模块……230
17.1.3 人员管理模块……230
17.1.4 信息管理模块……230
17.2 系统的 UML 基本模型……230
17.2.1 需求分析阶段模型……230
17.2.2 基本动态模型……231
17.3 系统中的类……234
17.3.1 系统类图……234
17.3.2 生成类的代码框架……236
17.4 系统的划分与部署……237
17.4.1 系统的包图……237
17.4.2 系统的部署图……237

附录 习题答案……238

参考文献……248

第一部分
概　　述

第 1 章 软件工程与面向对象方法

20 世纪 60 年代中期以来，软件系统规模越来越大，系统的复杂性也不断增加。由于人们当时难于掌握软件开发过程和开发方法，最终出现了软件危机。为了解决这一问题，软件工程的思想提出并逐步为人们所重视，至今已取得了令人瞩目的成就。然而，由于当时主流的开发方法是结构化的分析与设计方法，这种方法的缺陷也逐渐暴露出来。因此，人们逐渐寻找到了一种新的途径——面向对象方法。面向对象的概念自 20 世纪 60 年代后期出现以来，已经发展成为一种完整的思想与方法体系并且获得了广泛应用。其在软件开发方面呈现出的巨大优越性，使得人们将其作为一个解决软件危机的突破口。如今，面向对象技术已经延伸到了软件开发的各个过程并逐步形成了一个完善的机制。

1.1 软件工程简介

软件工程以系统性的、规范化的、可定量的过程化方法去开发和维护软件，以及研究如何把经过时间考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来。本节将主要介绍软件工程的发展过程及相关概念。

1.1.1 软件工程的发展过程

从 20 世纪 60 年代中期开始，软件行业进入了一个大发展时期。软件开始作为一种产品被使用，同时也产生了许多软件公司。然而，随着软件规模的扩大，复杂性的增加，功能的增强，使用早期的自由软件开发方式来开发高质量的软件变得越来越困难。在软件开发的过程中，经常会出现不能按时完成任务、产品质量得不到保证、工作效率低下和开发经费严重超支等情况，失败的软件项目比比皆是。这一系列问题导致了“软件危机”的产生。

软件危机的出现及其日益严重的趋势，充分暴露了软件产业在早期发展过程中存在的各种各样的问题。可以说，人们对软件产品认识的不足以及对软件开发的内在规律的理解偏差是软件危机出现的根本原因。为了解决软件危机，人们逐渐认识了软件的特性以及软件产品开发的内在规律，并尝试用工程化的思想去指导软件开发，于是软件工程诞生了。

1968 年，在北大西洋公约组织举行的一次学术会议上，该组织的科学委员们在开会讨论软件的可靠性与软件危机的问题时，首次提出了“软件工程”的概念，并将其定义为“为了经济地获得可靠的和能在实际机器上高效运行的软件，而建立和使用的健全的工程原则”。这个定义肯定了

工程化的思想在软件工程中的重要性，但是并没有提到软件产品的特殊性。

经过四十多年的发展，软件工程已经成为一门独立的学科，人们对软件工程也逐渐有了更全面、更科学的认识。在现代，软件工程是指应用计算机科学技术、数学和管理学的原理，运用工程学的理论、方法和技术，研究和指导软件开发和演化的一门交叉学科。它强调按照软件产品的特殊性质，采用工程化的思想来指导软件开发，在高效的软件生产和科学的项目管理的基础上得到高质量的软件产品。

相对于其他学科而言，软件工程是一门比较年轻的学科，它的思想体系和理论基础还有待进一步修整和完善。软件工程学包括的内容有软件工程原理、软件工程过程、软件工程方法、软件工程模型、软件工程管理、软件工程度量、软件工程环境和软件工程应用等。

1.1.2　软件工程的目标和原则

一般来说，软件工程的目标主要包括以下几点。

- 使软件开发的成本能够控制在预计的合理范围内。
- 使软件产品的各项功能和性能能够满足用户需求。
- 提高软件产品的质量。
- 提高软件产品的可靠性。
- 使生产出来的软件产品易于移植、维护、升级和使用。
- 使软件产品的开发周期能够控制在预计的合理时间范围内。

为了达到上述目标，软件工程设计、工程支持以及工程管理在软件开发过程中必须遵循一些基本原则。著名软件工程专家鲍伊姆（B.W.Boehm）综合有关意见并总结了多年来软件开发的经验，提出了软件工程的 7 条基本原则。

- 用分阶段的生命周期计划进行严格的管理。
- 坚持进行阶段评审。
- 实行严格的产品控制。
- 采用现代程序设计技术。
- 软件工程结果应能清楚地审查。
- 开发小组的人员应该少而精。
- 承认不断改进软件工程实践性的必要性。

1.2　面向对象方法简介

一切事物皆对象。通过面向对象的方式，将现实世界的事物抽象成对象，现实世界中的关系抽象成类、继承，帮助人们实现对现实世界的抽象与数字建模。面向对象的概念极大地丰富了软件开发方法，也是 UML 出现的背景和基础。本节将主要介绍面向对象的相关内容。

1.2.1　什么是面向对象方法

如今，面向对象方法（Object-Oriented Method，OOM）已深入计算机软件领域的各个分支。它不仅是一些具体的软件开发技术与策略，而且是一整套关于如何看待软件系统与现实世界的关系，用什么观点来研究问题并进行问题求解，以及如何进行软件系统构造的软件方法学。

试读结束：需要全本请在线购买：　www.ertongbook.com

面向对象方法解决问题的思路就是主张从客观世界固有的事物出发来构造系统，提倡用人类在现实生活中常用的思维方法来认识、理解和描述客观事物，强调最终建立的系统能够映射问题域，也就是说，系统中的对象以及对象之间的关系能够如实地反映问题域中固有事物及其关系。一般来说，面向对象方法是一种运用一系列面向对象的指导软件构造的概念和原则（如类、对象、抽象、封装、继承、多态、消息等）来构造软件系统的开发方法。从本质上讲，面向对象方法是对这些概念和原则的应用。

面向对象的思想已经被软件开发过程的各个阶段所应用。例如，面向对象分析（Object Oriented Analysis，OOA）、面向对象设计（Object Oriented Design，OOD），以及面向对象编程实现（Object Oriented Programming，OOP）。

1.2.2 面向对象方法的发展历史

人们普遍认为，第一种面向对象的编程语言是 1967 年诞生的 Simula-67。它由挪威奥斯陆国家计算中心的 Kristen Nygaard 与 Ole-Johan Dahl 设计开发。尽管这种语言没有后继版本，却首先引入了类、对象、继承等概念，对后来的许多面向对象语言的设计者产生了很大的影响。

使面向对象技术进入实用化的标志是 19 世纪 70 年代诞生的 Smalltalk，它是由美国施乐帕洛阿尔托研究中心（Xerox Palo Alto Research Center，PARC）的 Alan Kay 设计实现的。最初的版本 Smalltalk-72 在 1972 年被发布，其中正式使用了“面向对象”这个术语。PARC 先后发布了 Smalltalk-72、76 和 78 等版本，直至 1981 年推出该语言完善的版本 Smalltalk-80。Smalltalk-80 的问世被认为是面向对象语言发展史上最重要的里程碑。Smalltalk-80 提供了比较完整的面向对象技术解决方案，诸如类、对象、封装、抽象、继承、多态等。它是第一个完善的、能够实际应用的面向对象语言。

从 20 世纪 80 年代中期到 90 年代，是面向对象的程序设计语言走向繁荣的阶段。比较实用的面向对象编程语言大量涌现，如 Objective C、C++、Eiffel 和 CLOS 等语言。其中诞生于 1983 年的 C++语言对面向对象技术的发展起到了重要作用。面向对象技术能够发展到今天，正是因为 C++语言的广泛应用，它使得面向对象技术真正从实验室阶段走到了商业化阶段，当今的 Java、C#等面向对象的编程语言都有 C++的影子。

在 20 世纪 80 年代末 90 年代初，随着软件工程技术的日益成熟，面向对象的软件工程也得到了迅速发展，面向对象方法从编程发展到设计、分析，进而发展到整个软件生命周期。在此期间，大量关于面向对象方法的著作问世，并各有自己的一套概念、定义、表示法、术语和适用的开发过程。于是在它们中间出现了许多差异，这给普通用户的使用带来了很大的困惑。一些人尝试将各种方法中使用的各种概念进行统一，最终由 Grady Booch、James Rumbaugh 和 Ivar Jacobson 发起的统一建模语言（UML）在 1997 年 11 月被对象管理组织（Object Management Group，OMG）全体成员一致通过，并被采纳为标准。UML 的产生标志着面向对象方法学的统一，从而为面向对象方法的应用扫清了最后一个障碍。

1.2.3 面向对象方法的基本概念

面向对象方法以类和对象作为核心，并存在着很多与之相关的原则。这些面向对象的基本概念决定了面向对象方法的本质特征。使用这些概念并遵循相关原则，才是一个真正符合面向对象思想的解决方案。下面依次对这些概念进行介绍。

1. 对象

对象（object）有着广泛的含义，难以被精确地定义。一般来说，世界上万事万物都可以看作是一个对象。这些现实中的实体也被称作客观对象。客观对象可以抽象其某些属性和方法来研究在某个问题或场景中的性质，这被称为问题对象。抽象出来的问题对象通过封装等过程成为计算机中的一个包含有数据和操作的集合体，这被称为计算机对象。三种对象之间的关系如图 1-1 所示。

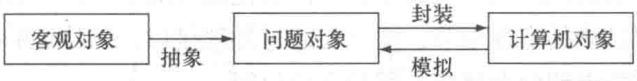


图 1-1　三种对象之间的关系

一个对象应该是一个具有状态、行为和标识符的实体，并且对象之间往往可以通过通信互相交互。一般来说，对象可以被归纳为下列四个特性。

- 自治性：即对象有一定的独立处理问题的能力，且对象自身的状态变化是不直接受到外界干预的。
- 封闭性：对象是相对封闭的。外部对象只能通过发送消息来对其施加影响，却无法直接对对象进行修改。对象隐藏了属性与操作的实现方法，只有操作声明对外部可见。
- 通信性：对象具有能与其他对象进行通信的能力。具体来说，就是对象能向其他对象发送消息，也能接受其他对象发送来的消息。对象通过通信来建立联系，并协作完成系统的某项任务。
- 被动性：虽然对象自身状态变化不受外界的直接影响，但是对象的存在和状态转换都是由外界驱动的。即只有在对象接收到外界消息后，自身才可以进行某种转换。

2. 类

对象是存在于某个时空的具体实体，而类（class）则是拥有共同的结构、行为和语义的一组对象的抽象。例如，我们可以定义一个“哺乳动物”类，则所有满足全身披毛、恒温胎生、体内有膈的脊椎动物对象都属于该类。类可以作为对象的一种描述机制，用来刻画一组对象的公共属性与公共行为，也可以用来作为程序的一个单位，用来形成程序中更大的模块。

与对象相似，类也应该具有数据、操作及标识符，并且类之间通过接口使一些操作对外可见。一般来说，类可以从以下四个角度来理解。

- 类是面向对象程序中的构造单位。一个面向对象程序就是一组相关的类。
- 类是面向对象程序设计语言的基本成分。在面向对象编程语言中，类内的成分是无法单独构成程序的，程序应该至少包含一个完整的类。
- 类是抽象数据类型的具体表现。类可以表示一种数据类型的抽象并给出了具体的数据结构表示与操作的实现方法。
- 类刻画了一组相似对象的共同特性。在面向对象程序的运行时，对象是根据类的定义而创建的，同类的对象具有相同的属性与操作。对象又被称为类的实例。

3. 抽象

抽象（abstraction）就是揭示一个事物区别于其他事物的本质特征，去除从某一个角度看来不重要的细节的行为。抽象是一个分析与理解问题的过程，它取决于使用者的目的，应该包括使用者所需要的那些问题，而忽略掉其他不相关的部分。因此抽象过程并没有唯一的答案，同一个实体在不同的场景中可能有不同的抽象。从对象到类的过程就是抽象的过程，即将所见到的具体实

体抽象成概念，从而可以在计算机中进行描述并对其采取各种操作。

在面向对象中，抽象具有静态与动态的属性。例如，一个文件对象有文件名与文件内容，这些是静态的属性。而同时，在对象的生命周期中，这些属性的值是动态的——文件名与内容都可能被改变，这些就被抽象为对象的操作。

4. 封装

封装（encapsulation），即对其客户隐藏对象的属性和实现细节，仅对外公开接口，并控制在程序中属性的读和修改的访问级别。封装是软件模块化思想的体现，其目的是增强安全性和简化编程，使用者不必了解具体的实现细节，而只要通过外部接口，以特定的访问权限来使用类的成员即可。简单来说，封装强调两个概念，即独立和封闭。

- 独立是指对象是一个不可分割的整体，它集成了事物全部的属性和操作，并且它的存在不依赖于外部事物。
- 封闭是指与外部的事物通信时，对象要尽量地隐藏其内部的实现细节，它的内部信息对外界来说是隐蔽的，外界不能直接访问对象的内部信息，而只能通过有限的接口与对象发生联系。

可以说，类是数据封装的工具，而对象是封装的实现。类的成员又分为公有成员、私有成员和保护成员，它们分别有不同的访问控制机制。封装是软件模块化思想的重要体现。

5. 泛化

泛化（generalization）是类元的一般描述和具体描述之间的关系，具体描述建立在一般描述的基础之上，并对其进行了扩展。具体描述完全拥有一般描述的特性、成员和关系，并且包含补充的信息。例如，中学生是学生中具体的一种，中学生保持了学生的基本特性并加入了附加特性，二者就构成了泛化关系。实现泛化关系的机制为继承（inheritance）。一个子类（subclass）继承一个或多个父类（superclass），从而实现了不同的抽象层次，实现两者之间的泛化关系。通过这种关系，子类可以共享其父类的结构和行为，从而容易复用已经存在的数据和代码，并实现多态处理。

6. 多态

多态（polymorphism）是在同一接口下表现多种行为的能力，是面向对象技术的根本特征。具体来说，多态允许属于不同类的对象对同一消息做出不同的响应。当一个对象接收到进行某项操作的消息时，多态机制将根据对象所属的类，动态地选用该类中定义的操作。面向对象方法正是利用多态提供的动态行为特征来封装变化，适应变更，以达到系统的稳定。首先需要有泛化关系的支持，然后才能表现多态。例如，先定义一个父类“几何图形”，它具有“计算面积”的操作，然后再定义一些子类，如“三角形”“长方形”和“圆形”，它们可继承父类“几何图形”的各种属性和操作，并且在各自的定义中要重新描述“计算面积”的操作。这样，当有计算几何图形面积的消息发出时，对象会根据类的类型做出不同的响应，采用不同的面积计算公式。

1.2.4 面向对象方法的优势

面向对象方法是在传统的结构化设计方法出现很多问题的情况下应运而生的。它之所以能够被广泛认可和应用，是因为其自身的优势，这些优势体现在很多方面。

传统的结构化设计方法侧重于计算机处理事情的方法和能力，面向对象方法则从客观世界存在的事物进行抽象，更符合人类的思维习惯。这一特点能够让软件开发设计人员更有效地实现业务和系统之间的理解和转换，从而更快速、有效地解决用户问题。

在软件开发过程中，需求的不稳定性是影响软件工程的一个非常重要的因素。在现实使用时，

数据和功能最容易发生改变，而对象一般则是相对稳定的。为此，使用面向对象方法可以用较稳定的对象将易变的功能和数据进行封装，从而保证较小的需求变化不会导致系统结构大的改变。

复用性也是面向对象方法带来的优势之一。面向对象方法通过封装、继承、聚合等手段，在不同层次上提供各种代码复用，以此提高软件的开发效率。

除此之外，面向对象方法还有改善软件结构、增强扩展性、支持迭代式开发等优势。这些优势使得面向对象方法得到了更广泛的应用。

小　结

本章简要介绍了软件工程的概念以及面向对象的概念。软件危机的出现促进了软件工程的兴起，同时结构化方法的不足也促进了面向对象思想的诞生。如今，面向对象的思想已经进入到计算机软件领域的许多方面，占据着不可替代的位置。同时，面向对象的兴起也促进了 UML 的诞生和发展。

习　题

1. 选择题

（1）软件工程的概念是在（　　）年被首次提出的。

A. 1949　　B. 1968　　C. 1972　　D. 1989

（2）下列不属于软件工程的目标的一项是（　　）。

A. 提高软件产品的质量　　B. 提高软件产品的可靠性

C. 减少软件产品的需求　　D. 控制软件产品的开发成本

（3）软件危机产生的主要原因是（　　）。

A. 软件工具落后　　B. 软件生产能力不足

C. 对软件认识不够　　D. 软件本身的特点及开发方法

（4）人们公认的第一门面向对象编程语言是（　　）。

A. Simula　　B. Smalltalk　　C. C++　　D. Java

（5）下列编程语言中不支持面向对象的特性的是（　　）。

A. C++　　B. ANSI C　　C. Java　　D. Objective C

（6）下列选项中不是面向对象方法的相关原则的是（　　）。

A. 封装　　B. 继承　　C. 多态　　D. 结构

（7）（　　）是面向对象方法中用来描述“对客户隐藏对象的属性和实现细节”的概念。

A. 封装　　B. 继承　　C. 多态　　D. 抽象

（8）下列选项中不属于面向对象方法的优势之一的是（　　）。

A. 复用性强　　B. 改善了软件结构

C. 软件的执行效率更高　　D. 抽象更符合人类的思维习惯

2. 判断题

（1）软件就是程序，编写软件就是编写程序。　　（　　）

（2）软件危机的主要表现是软件需求增加，软件价格上升。（　　）

（3）C 语言对面向对象的发展起到了重要作用。（　　）

（4）面向对象方法中的对象是从客观世界中抽象出来的一个集合体。（　　）

（5）面向对象可以保证开发过程中的需求变化完全不会导致系统结构的变化。（　　）

（6）面向对象方法就是使用面向对象的程序设计语言进行编程。（　　）

（7）对象的自治性指的是对象是完全封闭的，不受任何外界影响。（　　）

（8）类是面向对象程序中的构造单位，也是面向对象程序设计语言的基本成分。（　　）

3. 简答题

（1）简述软件危机产生的原因和可能的解决方案。

（2）软件工程的目标有哪些？

（3）什么是面向对象方法？简述其优势。

（4）简述对象、类、抽象、封装、泛化与多态的概念。

第2章 统一建模语言 UML

随着面向对象方法的出现和种类的不断增多，使用何种开发方法往往成为软件设计人员的一大问题，这也妨碍了不同项目开发组之间的交流。因此，一种标准统一的、综合了各种开发方法长处的建模语言 UML 应运而生。本章主要讲解软件建模的相关概念，并对 UML 进行简要概述。通过对本章的阅读，读者可以对软件建模和 UML 有一个总体的认识。

2.1 软件建模简介

建模是研究系统的重要手段和前提。建模用于定义应用程序的要求，确定可能被其他企业级应用程序重复使用的数据和服务，并为将来扩展奠定了强有力的基础。本节将简要介绍软件建模的相关内容。

2.1.1 什么是模型

模型（model）是用某种媒介对相同媒介或其他媒介里的一些事物的表现形式。从一个建模角度出发，模型就是要抓住事物的最重要方面而简化或忽略其他方面。简而言之，模型就是对现实的简化。建立模型的过程，称为建模。

模型提供了系统的蓝图。它既可以包括详细的计划，也可以包括系统的总体计划。每个系统都可以从不同方面用不同模型来描述刻画，每个模型都是在一个特定语义上闭合的系统抽象。模型可以是结构性的或行为性的，这对应着静态与动态的两种建模机制。

软件系统的模型用建模语言来表达，包括语义信息和表示法，可以使用图形和文字等多种不同形式。本书中讨论的 UML 就是以图形作为表现形式的一种建模语言。

2.1.2 建模的重要性

建模可以帮助理解正在开发的系统，这是需要建模的一个基本理由。人对复杂问题的理解能力是有限的。建模可以帮助开发者缩小问题的范围，每次着重研究一个方面，进而对整个系统产生更加深刻的理解。可以明确地说，越大、越复杂的系统，建模的重要性也越大。建模对一个系统主要有以下几点作用。

- **捕获和精确表达项目的需求和应用领域的知识，以使全部涉众能够理解并达成一致。**通过建模，可以捕获关于这个软件的应用领域、使用方法、模块拆分和构造模式等方面的需求信息。这里的涉众包括软件架构师、系统分析员、程序员、项目经理、顾客、投资者、最终用户和使用

试读结束：需要全本请在线购买： www.ertongbook.com