

UML基础 与Rose建模实用教程

- 最新的面向对象、UML、Rational Rose和Rational统一过程的相关知识
- 软件设计与开发人员一定要掌握的UML图和创建图中模型元素的基础理论
- 以图书管理系统和超市信息管理系统为例，介绍如何使用UML分析和设计一个实际的项目

谢星星 编著



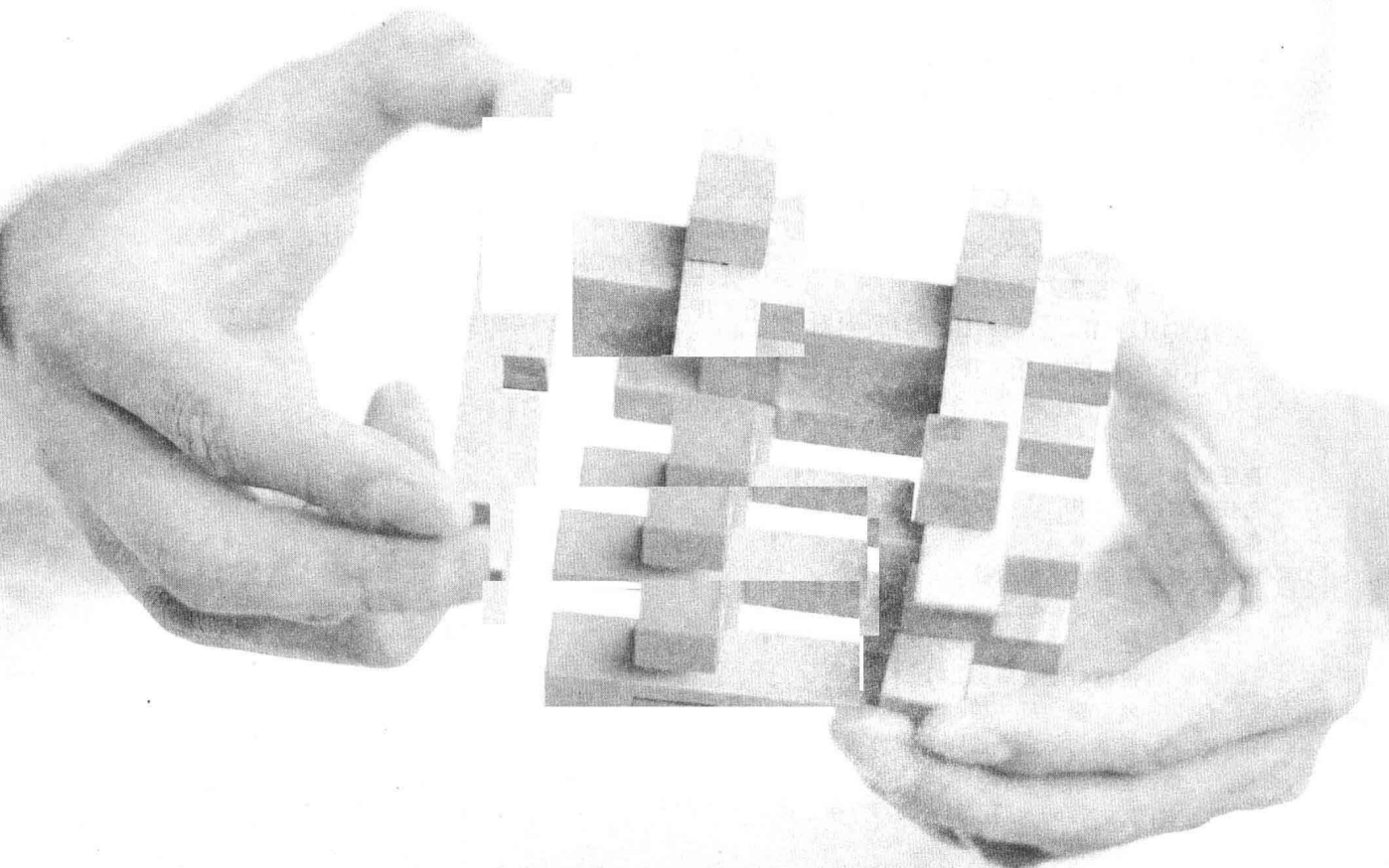
20小时全程视频讲解
PPT教学演示课件
案例素材与源程序

赠送 6 个完整系统的建模全过程：考试成绩管理系统、高校教材管理系统、网上教学系统、汽车租赁系统、ATM自动取款机系统、网上选课系统

清华大学出版社

UML基础 与Rose建模实用教程

谢星星 编著



清华大学出版社
北 京

内 容 简 介

本书系统地讲解了学习 UML 图和创建图中模型元素的基础理论, 并通过两个完整的建模案例讲解 Rose 建模工具的使用。

全书内容分为 4 部分: 入门基础(第 1~4 章), 介绍了面向对象、UML、Rational Rose 和 Rational 统一过程的相关知识; 图(第 5~12 章), 针对 UML 的各种图, 包括用例图、类图、对象图、序列图、协作图、状态图、活动图、包图、构件图和部署图进行介绍; 案例(第 13~14 章), 以图书管理系统和超市信息管理系统为例, 介绍如何使用 UML 分析和设计一个实际的项目; 附录(附录 A~附录 H)针对 Rational Rose 的安装和应用进行详细介绍, 并给出了 6 个系统的建模全过程, 最后提供章末练习答案。

本书适合软件设计与开发人员学习参考, 更适合作为高等院校计算机工程相关专业的教材或教学参考书。

本书封面贴有清华大学出版社防伪标签, 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

UML 基础与 Rose 建模实用教程/谢星星编著. —北京: 清华大学出版社, 2011.1
ISBN 978-7-302-23976-5

I. ①U… II. ①谢… III. ①面向对象语言, UML—程序设计—教材 IV. ①TP312
中国版本图书馆 CIP 数据核字(2010)第 205365 号

责任编辑: 夏非彼 张 楠

责任校对: 闫秀华

责任印制: 何 芊

出版发行: 清华大学出版社

地 址: 北京清华大学学研大厦 A 座

<http://www.tup.com.cn>

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

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者: 北京鑫海金澳胶印有限公司

经 销: 全国新华书店

开 本: 190×260 印 张: 22.5 字 数: 600 千字

附光盘 1 张

版 次: 2011 年 1 月第 1 版 印 次: 2011 年 1 月第 1 次印刷

印 数: 1~4000

定 价: 39.80 元

前言

Preface

本书将带领读者进入 UML 知识的殿堂，通过有效的学习掌握统一建模语言的基础内容，并以完整的开发过程为实例，介绍如何将 UML 运用到实际的项目开发中。

本书内容

本书分为入门基础、图、案例、附录共 4 个部分。

- 入门基础 (第 1~4 章): 着重介绍 UML 和 Rational Rose 的预备知识, 包括面向对象概述、UML 概述、Rational Rose 概述、Rational 统一过程。
- 图 (第 5~12 章): 着重介绍 UML 的各种图, 包括用例图、类图、对象图、序列图、协作图、状态图、活动图、包图、构件图与部署图等, 对 UML 图的介绍方式为: 图的基本概念→图的组成→图的创建概述→图的创建示例。通过这种方式能够使读者完整而系统地去把握和了解每一种 UML 图。
- 案例 (第 13~14 章): 着重介绍两个案例, 即图书管理系统和超市信息管理系统。通过这两个案例全面而系统地对如何使用 Rational Rose 进行建模给予说明。
- 附录 (附录 A~附录 H): 附录 A 介绍了 Rational Rose 的安装和应用, 目的是帮助读者熟练使用 Rational Rose 进行建模; 附录 B~附录 G 详细给出了 6 个系统的建模过程, 附录 H 是各章后练习的习题答案, 方便读者参考。

本书特点

- 理论实际, 紧密结合: 本书在介绍理论知识的同时, 每一章还给出了大量的案例讲解, 力求让读者在理解基础知识后, 就能学以致用, 快速上手。
- 配套教学, 源码视频: 在随书附赠的光盘中, 提供了每一章的源文件和多媒体教学视频, 整体的多媒体教学长达 20 个小时。读者可以随时观看视频, 从而同步学习。
- 图文并茂, 步骤详细: 在具体介绍 Rational Rose 软件功能的时候, 本书提供了详细的图例, 详尽说明了每一步功能的实现过程。

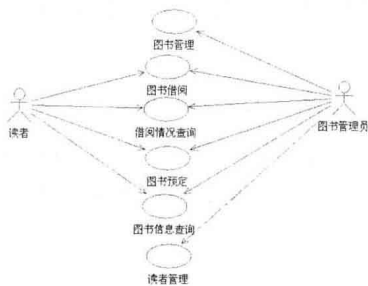
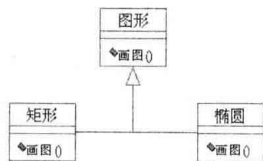
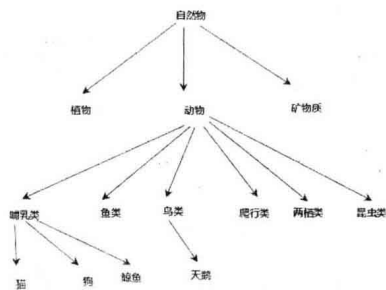
本书主要由谢星星、张华杰主持编写, 张华杰老师编写了其中的第 1~10 章, 此外, 胡书敏、王相羽、韩浩阳、张瑛、吴小莉、袁润非、叶明、崔宁、卢宏、程东、王勤、张锐、汪小锋、李葵、叶浩、肖飞、宋海剑、林勇、朱衡勇和许小荣等同志也参加了本书的编写, 在此, 编者对他们表示衷心的感谢。

编者

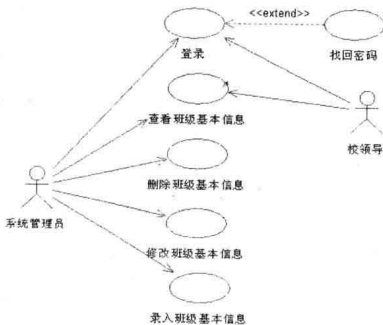
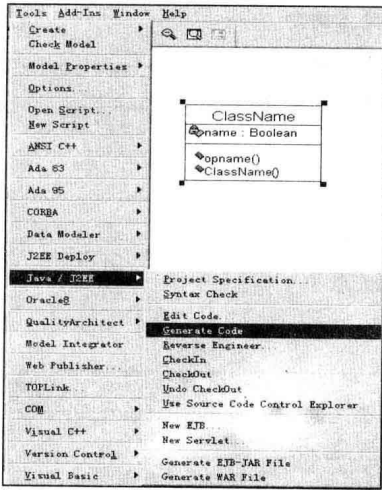
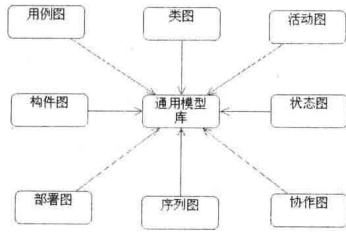
2010 年 11 月

目录

Contents



第1章 面向对象概述	1
1.1 面向对象的基本概念	1
1.1.1 什么是对象	1
1.1.2 面向对象与面向过程的区别	2
1.1.3 对象与类的确定	4
1.1.4 消息和事件	5
1.2 面向对象的基本特征	5
1.2.1 抽象	5
1.2.2 封装	7
1.2.3 继承	7
1.2.4 多态	8
1.3 面向对象方法论	9
1.3.1 面向对象分析	9
1.3.2 面向对象设计	13
1.4 面向对象建模	15
1.4.1 为什么要用UML建模	15
1.4.2 以面向对象建模为基础的开发模式	16
1.5 本章小结	20
习题1	21
第2章 UML概述	22
2.1 UML的起源与发展	22
2.2 UML的概念范围	23
2.2.1 视图	24
2.2.2 图	27
2.2.3 模型元素	32
2.3 UML的公共机制	36
2.3.1 UML的通用机制	37
2.3.2 UML的扩展机制	38
2.4 UML的对象约束语言	40
2.5 UML的目标	40
2.6 本章小结	41



习题 2..... 41

第 3 章 Rational Rose 概述 43

- 3.1 Rational Rose 的起源..... 43
- 3.2 Rational Rose 对 UML 的支持..... 44
- 3.3 Rational Rose 的 4 种视图模型..... 46
 - 3.3.1 用例视图..... 47
 - 3.3.2 逻辑视图..... 50
 - 3.3.3 构件视图..... 52
 - 3.3.4 部署视图..... 54
- 3.4 Rational Rose 的其他技术..... 55
 - 3.4.1 Rational Rose 双向工程..... 55
 - 3.4.2 Rational Rose 与 XML..... 59
- 3.5 本章小结..... 59

习题 3..... 59

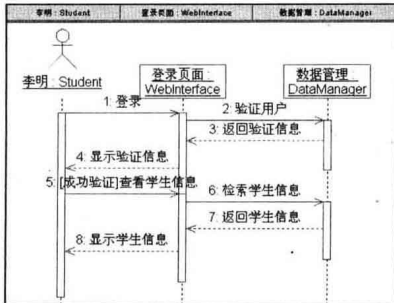
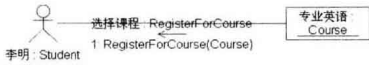
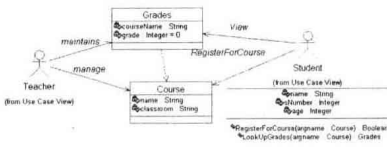
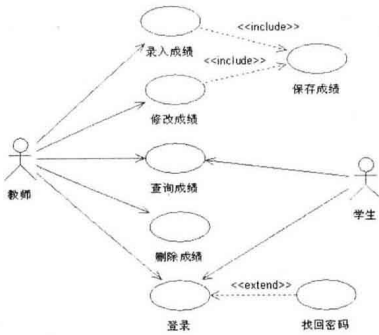
第 4 章 Rational 统一过程..... 61

- 4.1 什么是 Rational 统一过程..... 61
- 4.2 Rational 统一过程的结构..... 68
 - 4.2.1 统一过程的静态结构：过程描述..... 68
 - 4.2.2 统一过程的动态结构：迭代开发..... 70
 - 4.2.3 统一过程以架构为中心..... 72
- 4.3 Rational 统一过程的配置和实现..... 75
 - 4.3.1 配置 Rational 统一过程..... 75
 - 4.3.2 实现 Rational 统一过程..... 75
- 4.4 本章小结..... 77

习题 4..... 77

第 5 章 用例图..... 78

- 5.1 用例图的基本概念..... 78
 - 5.1.1 用例图的定义..... 78
 - 5.1.2 用例图的作用..... 79
- 5.2 用例图的组成..... 80
 - 5.2.1 参与者..... 80
 - 5.2.2 系统边界..... 82
 - 5.2.3 用例..... 83
 - 5.2.4 关联..... 85
- 5.3 用例图的创建概述..... 89
 - 5.3.1 创建用例图..... 89



5.3.2 创建参与者 91

5.3.3 创建用例 92

5.3.4 创建用例之间的关联 93

5.4 用例图的创建示例 94

5.4.1 需求分析 94

5.4.2 识别参与者 95

5.4.3 确定用例 96

5.4.4 构建用例模型 97

5.5 本章小结 99

习题 5 100

第 6 章 类图与对象图 102

6.1 类图与对象图的基本概念 102

6.1.1 类图与对象图的定义 102

6.1.2 类图与对象图的作用 104

6.2 类图的组成 105

6.2.1 类 105

6.2.2 接口 110

6.2.3 类之间的关系 110

6.3 类图的创建概述 118

6.3.1 创建类 118

6.3.2 创建类与类之间的关系 124

6.4 类图的创建示例 130

6.4.1 确定类和关联 131

6.4.2 确定属性和操作 132

6.5 对象图 132

6.5.1 对象图的组成 132

6.5.2 对象图的创建 134

6.6 本章小结 135

习题 6 136

第 7 章 序列图 138

7.1 序列图的基本概念 138

7.1.1 序列图的定义 138

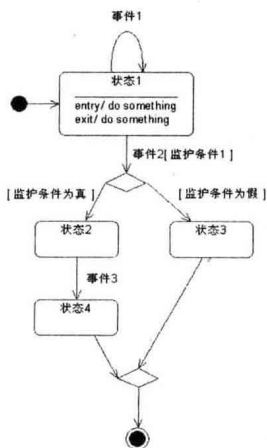
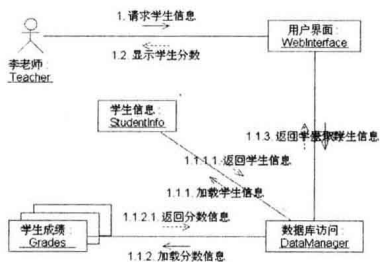
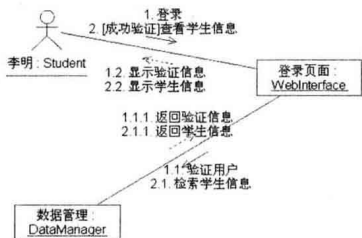
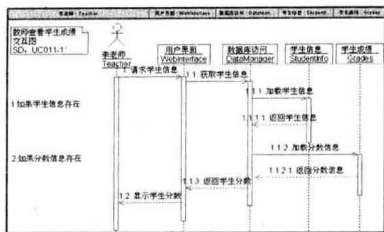
7.1.2 序列图的作用 139

7.2 序列图的组成 140

7.2.1 对象 (Object) 140

7.2.2 生命线 (Lifeline) 141

7.2.3 激活 (Activation) 141



7.2.4 消息 (Messages) 142

7.3 序列图的高级概念 143

7.3.1 创建与销毁对象 143

7.3.2 分支与从属流 144

7.3.3 帧化序列图 144

7.4 序列图的创建概述 145

7.4.1 创建序列图 145

7.4.2 创建生命线 148

7.4.3 创建消息 149

7.5 序列图的创建示例 151

7.5.1 确定工作流程 151

7.5.2 确定对象 152

7.5.3 确定消息和条件 152

7.5.4 绘制序列图总图 154

7.6 本章小结 154

习题 7 154

第 8 章 协作图 156

8.1 协作图的基本概念 156

8.1.1 协作图的定义 156

8.1.2 协作图的作用 157

8.2 协作图的组成 158

8.2.1 对象 158

8.2.2 消息 158

8.2.3 链 159

8.3 协作图的创建概述 159

8.3.1 创建对象 159

8.3.2 创建消息 163

8.3.3 创建链 163

8.4 协作图的创建示例 164

8.4.1 确定协作图的元素 164

8.4.2 确定元素之间的结构关系 165

8.4.3 细化协作图 165

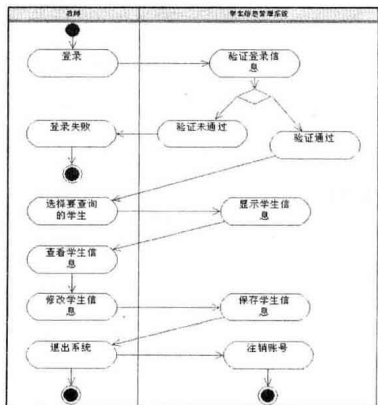
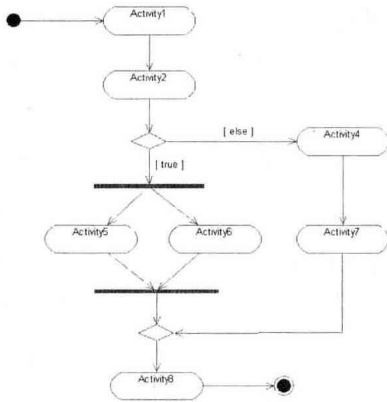
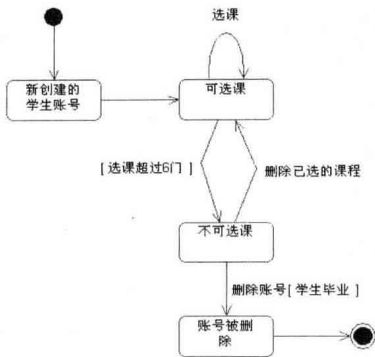
8.5 本章小结 166

习题 8 166

第 9 章 状态图 168

9.1 状态图的基本概念 168

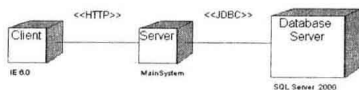
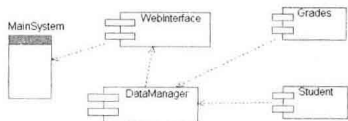
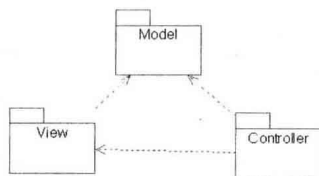
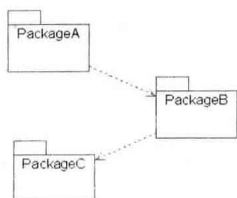
9.1.1 状态图的定义 168



- 9.1.2 状态图的作用 171
- 9.2 状态图的组成 172
 - 9.2.1 状态 172
 - 9.2.2 转换 174
 - 9.2.3 判定 177
 - 9.2.4 同步 178
 - 9.2.5 事件 178
- 9.3 组成状态 179
- 9.4 状态图的创建概述 181
 - 9.4.1 创建状态图 181
 - 9.4.2 创建初始和终止状态 182
 - 9.4.3 创建状态 183
 - 9.4.4 创建状态之间的转换 184
 - 9.4.5 创建事件 184
 - 9.4.6 创建动作 185
 - 9.4.7 创建监护条件 185
- 9.5 状态图的创建示例 186
 - 9.5.1 标识建模实体 186
 - 9.5.2 标识实体的各种状态 186
 - 9.5.3 标识相关事件并创建状态图 187
- 9.6 本章小结 187
- 习题 9 188

第 10 章 活动图 190

- 10.1 活动图的基本概念 190
 - 10.1.1 活动图的定义 190
 - 10.1.2 活动图的作用 191
- 10.2 活动图的组成 192
 - 10.2.1 动作状态 192
 - 10.2.2 活动状态 192
 - 10.2.3 组合活动 193
 - 10.2.4 分叉与结合 193
 - 10.2.5 分支与合并 194
 - 10.2.6 泳道 195
 - 10.2.7 对象流 195
- 10.3 活动图的创建概述 197
 - 10.3.1 创建活动图 197
 - 10.3.2 创建初始和终止状态 198
 - 10.3.3 创建动作状态 198



10.3.4 创建活动状态 199

10.3.5 创建转换 200

10.3.6 创建分叉与结合 200

10.3.7 创建分支与合并 200

10.3.8 创建泳道 201

10.3.9 创建对象流的状态与对象流 201

10.4 活动图的创建示例 202

10.4.1 标识活动图的用例 202

10.4.2 建模用例的路径 203

10.4.3 创建活动图 203

10.5 本章小结 204

习题 10 204

第 11 章 包图 206

11.1 模型的组织结构 206

11.2 包图的基本概念 207

11.3 包图的创建概述 210

11.4 包图的创建示例 213

11.5 本章小结 214

习题 11 214

第 12 章 构件图与部署图 216

12.1 构件图与部署图的基本概念 216

12.1.1 构件的基本概念 216

12.1.2 构件图的基本概念 218

12.1.3 部署图的基本概念 219

12.2 构件图与部署图的创建概述 221

12.2.1 创建构件图 221

12.2.2 创建部署图 228

12.3 构件图与部署图的创建示例 231

12.3.1 创建构件图 232

12.3.2 创建部署图 233

12.4 本章小结 234

习题 12 234

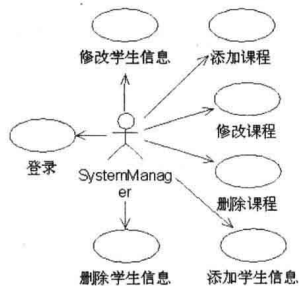
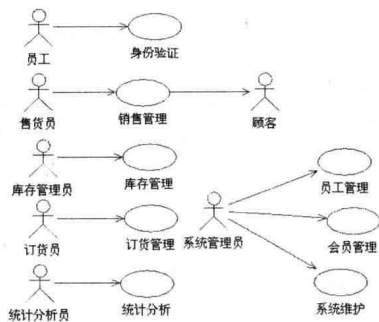
第 13 章 图书管理系统 236

13.1 需求分析 236

13.2 系统建模 237

13.2.1 创建系统用例模型 238





13.2.2 创建系统静态模型 240
 13.2.3 创建系统动态模型 241
 13.2.4 创建系统部署模型 264

13.3 本章小结 265

第 14 章 超市信息管理系统 266

14.1 需求分析 266

14.2 系统建模 267

14.2.1 创建系统用例模型 267

14.2.2 创建系统静态模型 271

14.2.3 创建系统动态模型 272

14.2.4 创建系统部署模型 278

14.3 本章小结 279

附录 A Rational Rose 的安装与应用 280

附录 B 考试成绩管理系统 309

附录 C 高校教材管理系统 314

附录 D 网上教学系统 319

附录 E 汽车租赁系统 324

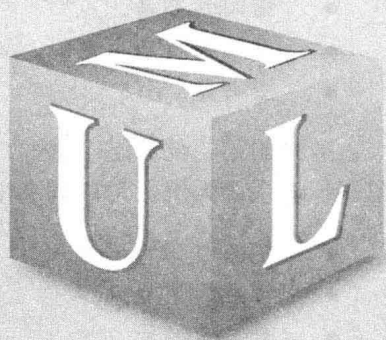
附录 F ATM 自动取款机系统 330

附录 G 网上选课系统 334

附录 H 参考答案 338

第 1 章

面向对象概述



目前，面向对象技术已经成为计算机软件开发中的一种主流技术了，随着其研究内容不断深化、应用领域不断扩大，特别是工业界对面向对象技术研究与产品化方面的支持，使得面向对象技术体现出越来越强大的生命力。面向对象技术作为一种全新的设计和构造软件的技术，使得计算机解决问题的方式更加符合人类的思维方式，更能直观地描述客观世界。

本章首先介绍了面向对象的基本概念，并与面向过程进行对比，然后具体到面向对象的基本特征和实际的方法论技术，最后对为什么使用 UML 建模进行简要介绍。

1.1 面向对象的基本概念

面向对象是一种全新的软件技术，其概念来源于程序设计本身。从 20 世纪 60 年代提出面向对象的概念到现在，面向对象已经发展成为一种比较成熟的编程思想了，并且逐步成为软件开发领域的主流技术。面向对象的程序设计（Object-Oriented Programming, OOP）立意于创建软件重用代码，具备更好地模拟现实世界环境的能力，这使它被公认为是自上而下编程的优胜者。它通过在程序中加入扩展语句把函数“封装”进编程所需要的“对象”中。面向对象的编程语言使得复杂的工作条理清晰、编写容易。

1.1.1 什么是对象

对象（Object）是面向对象（Object-Oriented, OO）系统的基本构造块，是一些相关的变量和方法的软件集。对象经常用于建立对现实世界中的一些对象模型。对象是理解面向对象技术的关键。

在教室里的桌子、椅子、电脑等都可以认为是对象。根据《韦氏大词典》的解释，对象是某种可被人感知的事物，也是思维、感觉或动作所能作用的物质或精神体。

该解释的第一部分“某种可被人感知的事物”是指对象有可以看到和感知到的“东西”，



而且可以占据一定事物的空间。以图书管理系统为例,想象一下图书管理系统中围绕图书管理概念中应该有哪些物理对象:

- 到图书馆借书的学生。
- 管理借阅的老师。
- 管理图书信息的计算机。
- 借阅的书籍。
- 存放书籍的书架。
- 图书馆。

解释的第二部分是“思维、感觉或动作所能作用的物质或精神体”,即“概念性对象”,以图书管理系统为例,可以列举出以下对象:

- 学生所在的院系。
- 学生的学号。
- 图书的编号。

对象是不能看到的、听到的,但是在描述抽象模型和物理对象时,仍然起着很重要的作用。

由上可知,软件对象可以这样定义:所谓软件对象,是一种将状态和行为有机结合起来形成的软件构造模型,它可以用来描述现实世界中的一个对象。

可以利用软件对象来代表现实世界中的对象,如用一个动画程序来代表现实世界中正在飞行的飞机,或者利用可以控制电子机械的程序来代表现实世界运行的机械车,也可以使用软件对象来描述抽象的概念,如按键事件就是一个用在 GUI 窗口系统的公共对象,它可以代表用户按下鼠标按钮或者键盘上的按键的反应。

1.1.2 面向对象与面向过程的区别

在面向对象程序设计(Object Oriented Programming, OOP)方法出现以前,结构化程序设计占据着主流。结构化程序设计是一种自上而下的设计方法,通常使用一个主函数来概括出整个程序需要做的事情,而主函数是由一系列子函数所组成。对于主函数中的每一个子函数,又都可以被分解为更小的函数。结构化程序设计思想就是把大的程序分解为具有层次结构的若干个模块,每个模块再分解为下一层模块,如此自顶向下、逐步细分,从而把复杂的大模块分解为许多功能单一的小模块。结构化程序设计的特征就是以函数为中心,也就是以功能为中心来描述系统,用函数来作为划分程序的基本单位,数据在过程式设计中往往处于从属的位置。结构化程序设计的优点是易于理解和掌握,这种模块化、结构化、自顶向下与逐步求精的设计原则,与大多数人的思维和解决问题的方式比较接近。

然而,对于比较复杂的问题或在开发中需求变化比较多的情况下,结构化程序设计往往显得力不从心。这是因为结构化程序设计是自上而下的,这要求设计者在一开始就要对解决的问题有一定的了解。在问题比较复杂时,要做到这一点会比较困难,而当开发中的需求变化时,以前对问题的理解也许会变得不再适用。事实上,开发一个系统的过程往往也是一个对系统不断了解和学习的过程,而结构化程序设计方法忽略了这一点。结构化程序设计的方法把密切相关、相互依赖的数据和对数据的操作相互分离,这种实质上的依赖与形式上的分离使得大型程

程序的编写比较困难，并难于调试和修改。在多人进行协同开发的项目组中，程序员之间很难读懂对方的代码，代码的重用变得十分困难。由于现代应用程序的规模越来越大，对代码的可重用性和易维护性的要求也越来越高，面向对象技术对以上问题提供了很好地支持。

面向对象技术是一种以对象为基础、以事件或消息来驱动对象执行处理的程序设计技术。它是一种自下而上的程序设计方法，它不像面向过程程序设计那样一开始就需要使用一个主函数来概括出整个程序，面向对象程序设计往往从问题的一部分着手，一点一点地构建出整个程序。面向对象设计是以数据为中心，使用类作为表现数据的工具，类是划分程序的基本单位。而函数在面向对象设计中成了类的接口，以数据为中心而不是以功能为中心来描述系统，相对来讲，更能使程序具有稳定性。它将数据和对数据的操作封装到一起，这种作为一个整体进行处理并且采用数据抽象和信息隐藏技术最终被抽象成一种新的数据类型——类。类与类之间的联系以及类的重用使得类出现了继承、多态等特性。类的集成度越高，越适合大型应用程序的开发。另外，面向对象程序的控制流程运行时是由事件进行驱动的，而不再由预定的顺序进行执行。事件驱动程序的执行围绕消息的产生与处理，靠消息的循环机制来实现。更加重要的是，可以利用不断成熟的各种框架，如.NET的.NET Framework等，在实际的编程过程中迅速地将程序构建起来。面向对象的程序设计方法还能够使程序的结构清晰简单，从而大大提高代码的重用性、有效地减少程序的维护量、提高软件的开发效率。

在结构上，面向对象程序设计和结构化程序设计也有很大不同。结构化程序设计首先应该确定的是程序的流程怎么走、函数间的调用关系怎么样、函数间的依赖关系是什么。一个主函数依赖于其子函数，子函数又依赖于更小的子函数，而在程序中，越小的函数处理的往往是细节实现，具体的实现又常常变化。这种变化的结果就是程序的核心逻辑依赖于外延的细节，程序中本来应该也是比较稳定的核心逻辑，也因为依赖于易变化的部分而变得不稳定起来，一个细节上的小改动也有可能依赖关系上引发一系列变动。可以说这种依赖关系也是过程式设计不能很好处理变化的原因之一，而一个合理的依赖关系应该由细节实现依赖于核心逻辑。面向对象程序设计由类的定义和类的使用两部分组成，主程序中定义对象并规定它们之间消息传递的方式，程序中的一切操作都是通过面向对象的发送消息机制来实现的。对象接收到消息后，启动消息处理函数完成相应的操作。

仍然以图书管理系统为例，使用结构化程序设计的方法时，首先需要在主函数中确定图书管理系统要做哪些事情，并使用函数将这些事情进行表示、使用一个分支选择程序进行选择，然后将这些函数进行细化实现，并确定调用的流程等。使用面向对象技术来实现图书管理系统时，以学生为例，要了解图书管理系统中学生的主要属性（如学号、院系等）、学生要进行什么操作（如借书、还书）等，并且把这些当成一个整体进行对待，形成一个类，即学生类。使用这个类可以创建不同的学生实例，即创建许多具体的学生模型，每个学生拥有不同的学号，都可以在图书馆借书和还书。学生类中的数据和操作都是可共享的，可以在学生类的基础上派生出大学生类、大专生类、研究生类等，从而实现代码的重用。

类与对象是面向对象程序设计中最基本和最重要的概念，也是创建和使用UML图的基础，有必要仔细理解和掌握，并且在学习中不断深化。





1.1.3 对象与类的确定

面向对象技术认为客观世界是由各种各样的对象组成的，每个对象都有自己的数据和操作，不同对象之间的相互联系和作用构成了各种系统。在面向对象程序设计中，系统被描绘成由一系列完全自治、封装的对象组成，对象和对象之间是通过对象暴露在外的接口进行调用的。对象是组成系统的基本单元，是一个组织形式的含有信息的实体，如一个叫张三的人，“人”作为实体对象，包含姓名为“张三”的信息。而类是创建对象的模板，在整体上可以代表一组对象，如创建“人”此类，它就代表人这个概念，可以使用此类来表达张三、李四等。设计类而不是设计对象，这样可以避免重复编码，类只需要编码一次就可以实例化这个类的对象。

对象 (Object) 是由状态 (State) 和行为 (Behavior) 构成的。事实上，属性 (Property)、状态 (State)、数据 (Data) 这些在各种书中提到的概念，意思都相似，是用于描述一个对象的数据元素，这些概念具体到各种语言便有不同的叫法。对象的状态值用来定义对象的状态，如当判断学生是否能够借书时，可以利用学生的学号 (称为第一个状态) 和学生目前的借书数量 (称为第二个状态) 进行判断。行为 (Behavior)、操作 (Operation) 以及方法 (Method) 这些在各种书中提到的概念，是用于描述访问对象的数据或修改 (维护) 数据值的方法，如“告诉图书管理员你的学号”和“选择需要借阅的图书”等。对象只有在具有状态和行为的情况下才有意义，状态用来描述对象的静态特征，行为用于描述对象的动态特征。对象是包含客观事物特征的抽象实体，封装了状态和行为，在程序设计领域可以用“对象=数据+数据的操作”来进行表达。

类 (Class) 是具有相同属性和操作的一组对象的组合，即抽象模型中的“类”描述了一组相似对象的共同特征，为属于该类的全部对象提供了统一的抽象描述，如学生类被用于描述能够到图书馆借阅图书的学生对象。

类的定义需要包含以下要素：

- 定义该类对象的数据结构 (属性的名称和类型)。
- 对象所要执行的操作，即类的对象要被调用执行哪些操作以及进行这些操作时对象还要执行哪些操作，如数据库操作等。

类是对象集合的再抽象，类与对象的关系如同一个模具和使用这个模具浇注出来的铸件一样，类是创建软件对象的模板——一种模型。类给出了属于该类的全部对象的抽象定义，而对象是符合这种定义的一个实体。类用来在内存中开辟一个数据区，用于存储新对象的属性或把一系列行为和对象关联起来。

一个对象又被称作类的一个实例，即实体化 (Instantiation)。实体化 (Instantiation) 是指对象在类声明的基础上创建的过程，如声明了一个学生类，可以在这个基础上创建一个姓名叫张三的学生对象。

类的确定和划分没有一个统一的标准和方法，基本上依赖于设计人员的经验、技巧以及对实际项目中问题的把握。通常的标准是“寻求共性、抓住特性”，即在一个大的系统环境中，寻求事物的共性，将具有共性的事物用一个类进行表述。确定一个类的步骤通常包含以下方面：

- 确定系统的范围，如图书管理系统需要确定和图书管理相关的内容。
- 在系统范围内寻找对象，该对象通常具有一个和多个类似的事物，如图书管理系统中，计算机系有一个名叫张三的学生，英语系名叫李四的人和张三一样都是学生。

- 将对象抽象成为一个类，按照上面的类的定义确定类的数据和操作。

在面向对象的程序设计中，类和对象的确定非常重要，是软件开发的第一步，软件开发中类和对象的确定将直接影响软件的质量。如果划分得当，对于软件的维护、扩充以及软件的重用性方面都有很大帮助。

1.1.4 消息和事件

当使用某一个系统时，单击一个按钮后通常会显示相应的信息，以图书管理系统为例，单击图书管理系统界面的某一个按钮时会显示出当前的图书信息。那么当前的程序是如何运行的呢？

- 01 图书管理系统界面的某一个按钮发送鼠标单击事件的消息给相应的对象。
- 02 对象接收到消息后有所反应，它提供了图书的相关信息给界面。
- 03 界面将图书的相关信息显示出来，从而完成任务。

在这个过程中首先要触发一个事件，然后发送消息，那么消息是什么呢？所谓消息（Message）是指描述事件发生的信息，是对象间相互联系和作用的方式。一个消息主要由 5 部分组成：消息的发送对象、消息的接收对象、消息的传递方式、消息的内容（参数）、消息的返回。传入消息内容的目的有两个，一个是让接受请求的对象获取执行任务的相关信息，另一个是行为指令。

什么是事件呢？所谓事件通常是指一种由系统预先定义而由用户或系统发出的动作。事件作用于对象，对象识别事件并作出相应反应。与对象的方法集可以无限扩展不同，事件的集合通常是固定的，用户不能随便定义新的事件。但是现代高级语言中可以通过一些其他技术在类中进行事件的加入，如 Click（单击对象时发生的事件）、Load（当界面被加载到内存时发生的事件）等。对象通过对外提供的方法在系统中发挥自己的作用，当系统中的其他对象请求一个对象执行某个方法时，就向该对象发送一个消息，对象响应这个请求，并完成指定的操作。程序的执行取决于事件发生的顺序，并由顺序产生的消息来驱动程序的执行，不必预先确定消息产生的顺序。

1.2 面向对象的基本特征

面向对象技术强调的是在软件开发过程中，对于客观世界或问题域中的事物的认知，采用人类在认知客观世界的过程中普遍运用的思维方法，可直观、自然地描述有关事物。

抽象、封装、继承、多态是面向对象程序的基本特征。正是这些特征使得程序的安全性、可靠性、可重用性和易维护性得以保证。随着技术的发展，可把这些思想用于硬件、数据库、人工智能技术、分布式计算、网络、操作系统等领域，越来越显示出其优越性。

1.2.1 抽象

人们每天都要应对身边的各种各样的信息，如电子邮件、新闻信息等，但是也有办法从这些信息中提取出精华，大脑懂得如何去简化所接收的信息，让信息细节通过抽象（Abstraction）过程进行管理，能够通过抽象进行以下操作。

1. 将需要的事物进行简化

大家都熟悉地球仪，作为一种抽象，地球仪上标明了地球信息的特征，在地球仪上可以清楚地看到各大洲的位置、各大洲的具体形状等。就必要性而言，地球仪并不包括真实世界中的所有存在物，如你在什么位置、学校在什么位置、学校的教学楼是什么样等。如果地球仪上真的有些东西，将很难进行表达，如果制作的是一个大型的地理信息系统，可能关注的东西会更多，总之，抽象真实世界的不同特征是根据使用对象的不同或者使用者的类型不同进行确定的。

以图书管理信息系统为例，从学生的角度来看，关注的是能否进行借书、能否还书。对于图书管理员来说，关注的往往是能否更好地维护图书等。他们都不会关注图书馆有多少个卫生间等特征。

通过抽象能够识别和关注当前状况或物体的主要特征，淘汰掉所有非本质信息，即通过抽象可以忽略事物中与当前目标无关的非本质特征、强调与当前目标有关的特征。

2. 将事物特征进行概括

如果能够从一个抽象模型中剔出足够多的细节，那它将变得非常通用，能够适应于多种情况或场合。这样的通用抽象相当有用，如可抽象出“学号”、“姓名”、“学院”、“年级”等来描绘学生的一般功能，但是要具体到如学生在教室里的座位号等，描述力就差了一些。

抽象模型越简单、展示的特点越少，它就越通用，也越具有普适性。抽象越复杂、越具有限制性，用于描述的情况也就越少。

3. 将抽象模型组织为层次结构

能够通过抽象按照一定的标准将信息系统地进行分类处理，以此来应付系统的复杂性，这一过程被称为分类（Classification）。

如在科学领域将自然物区分为动物、植物和矿物质。动物应该满足有生命、能够自主运动、对外界刺激能够产生反应等条件。可以按照这个条件将一些自然物划分为动物，如飞翔的天鹅、水中游的鲸鱼、狗和猫等。当有了这个规则并且按照这个规则进行分类时，将一个物体划分为适当的类就相当简单了。仍然可以定义出不同的规则将动物继续划分，直到能够构造出一个自顶向下渐趋复杂的抽象层次结构为止，如可以把飞翔的天鹅划分为鸟类，将猫和狗划分为哺乳类。图 1-1 是一个抽象层次结构的例子。

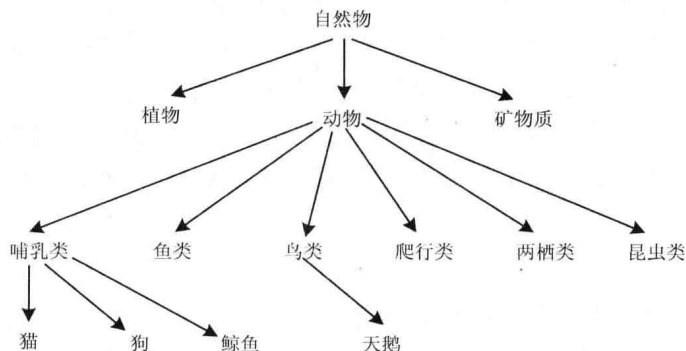


图 1-1 自然物的抽象层次结构示例