大学计算机优秀教材系列

面向对象分析与设计

第2版

麻志毅 编著

Object-Oriented Analysis and Design

Second Edition

机械工业出版社
China Machine Press

大学计算机优秀教材系列

面向对象分析与设计

第2版

麻志毅 编著

Object-Oriented Analysis and Design

Second Edition

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

面向对象分析与设计/麻志毅编著．—2版．—北京：机械工业出版社，2013.2
（大学计算机优秀教材系列）

ISBN 978-7-111-40751-5

Ⅰ．面…　Ⅱ．麻…　Ⅲ．面向对象语言－程序设计－教材　Ⅳ．TP312

中国版本图书馆CIP数据核字（2012）第311757号

本书是一本关于面向对象分析与设计的教材，讲述了面向对象的基本思想、主要概念以及相应的表示法，并给出了详细的建模过程指导。本书注重理论与实践相结合，通过给出大量的例题、内容较为详尽的案例分析以及对建模概念的详细剖析，阐明了如何进行面向对象的分析与设计。

本书适合作为高等院校计算机学院（或信息学院等）和软件学院的软件工程专业、计算机专业和相关专业的高年级本科生、工程硕士的教材，也可作为培训班师生以及从事软件开发的工程技术人员的参考书。

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：李　荣
北京瑞德印刷有限公司印刷
2013年3月第2版第1次印刷
185mm×260mm·14.75印张
标准书号：ISBN 978-7-111-40751-5
定　　价：35.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991　88361066　　投稿热线：（010）88379604
购书热线：（010）68326294　88379649　68995259　　读者信箱：hzjsj@hzbook.com

前　言

在20世纪90年代，面向对象技术以其显著的优势成为计算机软件领域的主流技术，随后该技术在大多数发达国家的软件开发中得到了相当广泛的运用。在我国的软件产业界，面向对象技术的学习与应用热潮出现于20世纪90年代后期，如今面向对象分析与设计技术也已经得到了广泛的应用。

当前，产业界需要大量掌握面向对象分析与设计技术的高级应用型开发人才。很多计算机学院和软件学院在软件工程教学中开设了相应的课程，旨在使学生不仅会使用一种或者几种面向对象编程语言来编程，更重要的是能运用面向对象方法进行系统建模，即通过面向对象分析（Object-Oriented Analysis，OOA）和面向对象设计（Object-Oriented Design，OOD）建立系统的分析模型和设计模型。

邵维忠教授和杨芙清院士合著的两本著作[17]~[18]在广泛借鉴国际上各种OOA与OOD方法的同时，根据作者长期的研究与实践形成了自己的方法特色。其中最主要的特色有三条：一是提倡充分运用面向对象方法的基本概念，限制扩充概念的引入，通过加强过程指导而保持建模概念的简练；二是对UML（Unified Modeling Language，统一建模语言）所采用的与面向对象有关的概念进行了深入的解析，给出了自己的见解；三是其OOD部分比以往的著作内容更为详细，并且更强调用OO概念表达各种全局性的设计决策。这两部学术专著作为教材适合于理论性强的研究生教学。

本书旨在提供一本更适合应用型人才培养的教材。在思想体系上，本书继承了参考文献［17］和［18］所提出的理论和方法。但是作为一本适合应用型人才培养的教材，本书与它们相比有以下不同：

- 减少了理论阐述和对不同学术观点的讨论，增加了对如何运用概念的讲解。
- 着重讲述了面向对象的应用技术。
- 在各章的正文部分增加了例题，在各章之后给出了习题。
- 通过案例讲述了如何运用面向对象方法进行分析与设计。

本书既是一本教材，也可作为从事软件开发的工程技术人员的参考书。由于以上几个特点，本书与参考文献［17］和［18］相比具有更强的普及性，适用于更广大的读者群。

UML是一个由国际对象管理组织（Object Management Group，OMG）采纳的建模语言规范，目前在软件工业界已经被广泛接受。但UML的内容过于庞大和复杂（这是UML本身的复杂性造成的），多数工程技术人员和读者反映其学习难度很大。UML中的许多内容是用于构造UML元模型的，对于大多数面向应用的软件开发者来说，这些概念是用不着的。还有一些概念在软件系统的建模中很少使用，这是因为UML是各方面成果

融合的产物，它要尽量地适合各领域。特别是 UML 不仅仅是用于面向对象开发的软件建模语言，它还可用于其他方面的建模，例如建筑业或机器制造业也可用它进行建模。基于上述因素，本书选用了 UML 中常用的概念来控制技术的复杂性。由于本书加强了运用基本概念解决各种复杂的分析与设计问题的过程指导，因此所选用的概念和表示法仍能保持表达能力的完整性。对本书而言，有些概念并非必不可少的，但为了方便读者理解这些常见概念，本书也适当地进行了讲解，同时也给出了一些运用基本的 OO 概念代替这些概念的方法。

本书所采用的概念和表示法与 UML 2.4 保持一致。在中文术语方面，本书与我国的行业规范《面向对象的软件建模规范》完全一致。作为该规范的主要起草者，本书作者曾经与国内很多专家、学者和企业界的专业人士进行过反复研究讨论，从而对该规范达成共识。

进行软件开发，应该遵循一定的过程指导。过程指导为完成软件系统开发的步骤提供详细指导，其中包括模型、工具和技术。本书所讲述的过程指导的思想来自参考文献[17]和[18]，即本书所采用的开发过程，是在借鉴了较为流行的多种开发过程的基础上，根据青鸟工程的成果和作者的科研及工程实践的经验总结出来的。

像使用其他开发方法一样，用面向对象方法进行软件系统建模的目的是要建立相应的模型。总的来讲，本书把模型分为功能需求模型、分析模型和设计模型。针对建立各种模型所使用的图以及其中的一些具体的模型元素，本书还给出了相应的规约。

对于面向对象的软件建模，需要有建模工具的支持。本书对此类工具所应具有的主要功能进行了讲述，并介绍了两款面向对象的软件建模工具。

与第 1 版相比，本版进行了如下改进：

- 对面向对象概念的定义更为准确，对概念的解释更加丰富和深入，对建模指导方面的内容进行了充实。
- 第 1 版中的建模语言采用 UML 2.0，然而至本版写作时 OMG 发布了 UML 2.4，其中模型图的种类、图元素的表示法以及一些解释都发生了变化，因此本版的建模语言采用了 UML 2.4。
- 解决了作者和热心的读者在第 1 版的使用中发现的一些问题。
- 为了加强对分析与设计建模策略和技巧的理解，本版给出了更多的应用实例。

以下简要地介绍本书的概貌，使读者对它有一个提纲挈领的了解。

第 1 章集中介绍了面向对象方法的基本思想和原则，解释了它的基本概念，论述了它的主要优点，并简单介绍了它的发展历史和现状，以及与本书密切相关的 UML 2.4。

第 2 章首先概述了面向对象分析所面临的问题，然后对其进行了综述，在综述中阐述了面向对象分析模型和过程模型。

第 3 章全面地讲解了建立功能需求模型所需要使用的概念及其表示法，并详述了如何使用它们来建立功能需求模型。

第 4 章详细地讲述了类图中所使用的概念与表示法，并详述了如何使用它们来建立类图。

第 5 章讲述了建立辅助模型所用到的几种图：顺序图、通信图、活动图、状态机图和

包图，其中详细地讲述了这些图中所使用的概念与表示法。

第 6 章说明了面向对象分析与设计的关系，并阐述了面向对象设计模型和过程模型。

第 7 章详述了如何针对实现条件对分析模型进行补充与调整，完成问题域部分设计。

第 8 章详述了进行人机交互设计所需要考虑的因素，并从分析和设计两个方面详述了如何进行人机交互设计。

第 9 章详述了什么是控制流，如何识别与定义控制流，以及如何协调控制流之间的同步。

第 10 章讲述了数据管理部分的设计。本章首先对数据库进行了简介，然后详细讲解了如何使用关系数据库系统对永久对象及它们之间的关系进行存储与检索。

第 11 章针对如何描述与构造系统的构件，详细讲解了构件图及其应用。本章还讲述了制品图和部署图。

第 12 章讲述了一些在面向对象设计中经常使用的设计模式。

第 13 章从耦合、内聚和复用等方面讲述了如何评价面向对象的设计模型。

第 14 章首先讲述如何把一个较为复杂的系统划分成一系列子系统，然后说明了如何对系统或子系统进行可视化建模，以及从那些方面建立系统的模型，此外还阐述了如何保证模型的一致性。

第 15 章通过一个具体的案例分析，说明如何用面向对象方法进行建模。

最后本书给出了两个附录。附录 A 讲述了两款面向对象的软件建模工具。附录 B 给出了对用面向对象方法进行软件系统建模所生成的文档的主要编制要求。

本书的研究工作和写作得到了北京大学邵维忠教授的大力帮助。邵维忠教授对书稿提出了十分难得的宝贵修改意见，本书的很多内容也来自他作为第一作者的著作[17,18]。邵维忠教授具有严谨的治学态度、深厚的学术功底以及敏锐的洞察力，他的指导使我受益良多。在此致以衷心的感谢！

本书的研究工作和写作得到了杨芙清院士和梅宏院士所领导的学术队伍的支持，也得到了国家自然科学基金（61272159）、北京市自然科学基金资助项目（4122036）、国家重点基础研究发展规划（2011CB302604）和国家 863 高技术研究发展计划项目（2012AA011202）的资助。在此谨向上述单位表示衷心的感谢！

对书中存在的错误和疏漏之处，恳请各位读者给予批评指正，并通过电子邮件（mzy@sei.pku.edu.cn）或其他方式进行更有意义的讨论。

麻志毅

2013 年 1 月于北京大学

教学建议

教学章节	教学要求	课时
第 1 章 面向对象方法概论	初步掌握面向对象方法的基本思想和主要概念，领会面向对象的基本原则 与传统软件开发方法对比，理解面向对象方法的主要优点 了解面向对象方法的发展历史和现状 了解 UML 与面向对象方法的关系	4～6
第 2 章 什么是面向对象分析	深刻地认识面向对象分析所面临的问题 掌握面向对象分析模型和过程模型	2
第 3 章 建立需求模型——用况图	掌握用况图的概念及其表示法，能熟练地运用它们建立软件系统的需求模型	3～4
第 4 章 建立基本模型——类图	掌握类图的概念与表示法，能熟练地运用它们建立软件系统的基本模型	8
第 5 章 建立辅助模型	掌握顺序图、通信图、活动图、状态机图和包图中的概念与表示法，能熟练地运用它们来建立软件系统的辅助模型	4～6
第 6 章 什么是面向对象设计	明确面向对象分析与面向对象设计的关系 掌握面向对象设计模型和过程模型	1～2
第 7 章 问题域部分的设计	掌握对典型问题的处理方法	4
第 8 章 人机交互部分的设计	掌握分析与设计人机交互部分的技术	2
第 9 章 控制驱动部分的设计	理解什么是控制流 掌握控制驱动部分的设计技术 掌握进程间和线程间的通信以及控制流间的同步技术	2～4
第 10 章 数据管理部分的设计	掌握针对关系数据库的数据存取设计 了解针对面向对象数据库的数据存取设计和针对文件的数据存取设计	3～4
第 11 章 构件及部署部分的设计	掌握构件、连接件、端口和接口等概念以及它们之间的关系 掌握基于上述面向对象设计模型对系统的构件以及构件之间的关系建模的技术 掌握对构件的客观建模的技术 了解如何使用部署图对系统的部署进行设计	3～4
第 12 章 若干典型的设计模式	领会设计模式中强调的原则 掌握典型的设计模式 具备学习其他设计模式的能力	2
第 13 章 OOD 的评价准则	了解如何从耦合、内聚和复用等方面评价面向对象的设计模型	1～2

（续）

教学章节	教学要求	课时
第 14 章 系统与模型	掌握系统、子系统、模型和视图等概念以及相关概念间的关系 掌握运用这些概念划分系统和对系统建模的技术	2～3
第 15 章 案例：教学管理系统	体会用面向对象方法进行系统建模的整个环节以及一些图的具体用法	4～6
总课时	第 1～15 章建议课时	45～59
	综合练习建议课时	20～30

说明：

1）若学时数少，可不讲或选讲通信图、活动图、构件及部署部分的设计、若干典型的设计模式、OOD 的评价准则以及系统与模型。

2）可随着课程的进度按相关章节的内容分开讲解第 15 章的内容。对于该章给出的综合性习题，建议学生在学完第 3 章后，根据所选择的习题的难易程度组成 2～4 人的小组，随着课程的进展在 20～30 学时内逐步完成综合练习。

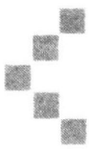
3）本书各章都附有习题，任课教师可以根据情况留课外作业。建议安排 1～2 次习题课，其中重点讲解作业中存在的带有普遍性的问题。也可以安排 1～2 次课堂综合性习题讨论课。

目　录

第三部分 面向对象设计

第一部分
PART ONE

概　　述



CHAPTER 1

第 1 章

面向对象方法概论

本章首先简要地回顾传统软件开发方法中存在的问题，然后重点讨论面向对象的基本思想、主要概念和基本原则，论述面向对象方法的主要优点，并对面向对象方法的发展史和现状以及统一建模语言（Unified Modeling Language，UML）进行简介。

通过对本章的学习，读者要了解面向对象方法的主要内容，掌握基本知识，为进一步学习与应用面向对象分析和设计方法打下基础。

1.1 传统软件开发方法中存在的问题

20 世纪 60 年代以前，软件开发者构造的软件系统的规模大多较小，且构造相对简单。那时所使用的编程语言和编程环境也相对简单，常见的编程语言有汇编语言以及随后出现的一些高级编程语言（如 FORTRAN 和 COBOL 等）。当时人们认为软件开发是一项强烈依赖个人技巧和技术能力的艺术性劳动，崇尚程序员的个人技能，没有认识到需要使用什么方法。那时产生的代码，按现在的人们所形容的，是意大利细面条式的，那是因为代码中含有较多的 GOTO 语句。

随着软件复杂性的增长，那种随心所欲的做法会带来问题，一个典型的问题是代码难以维护。一些高级编程语言试图解决所出现的问题，但这些语言并不能充分解决问题，因为软件开发也需要方法。

随后出现了多种软件开发方法，这些开发方法都能解决一些问题，但也都有一定的局限性。下面对三种典型的开发方法进行简要分析，以找出其中存在的主要问题。

1. 功能分解法

早期的一种开发方法称为功能分解法，它是以系统需要提供的功能为中心来开发系统的。它的基本思想为：首先定义顶层功能，然后把功能分解为子功能，同时定义功能之间的接口。对较大的子功能进一步分解，直到可给出明确的定义，进而根据功能/子功能设计数据结构和算法。

在那时，人们都认为功能分解法非常自然，因为它以系统需要提供的功能为中心来组织系统。此外，功能分解法也较好地运用了过程抽象原则。当时，计算机的应用还不是很普及，只有特定的用户有着软件需求，而且要求规模并不是很大。功能分解法的发明，在很大程度上解决了以前存在的问题，开发效率也有了很大的提高。特别是提出了模块化思想，并与模块化编程相结合，使得软件维护更加有效。这些是功能分解法在当时大受欢迎的主要原因。

使用这种以功能为中心的方法开发软件系统，一个显著特点是开始容易深入难。因为一开始按照功能需求进行自顶向下的功能分解是很直接的，但功能和功能接口这些系统成分却不能

直接地映射到问题域中的事物，这导致所建立的功能模型难以准确而深入地描述问题域，而且也难以检验所建立的模型的正确性。特别是，该方法对需求变化性的适应能力差：需求的变化必定导致功能模块发生变化，一个功能模块的变化往往引起其接口发生变化，这又致使其他模块发生变化，最终的结果经常是局部的变化导致全局性的影响。

2. 结构化方法

结构化方法包括结构化需求分析、设计、编程和测试方法等。结构化需求分析使用数据流图、加工说明和数据字典来构造系统的需求分析模型。结构化需求分析方法比较严谨，使用它可避免很多错误和疏漏。此外，该方法也运用了逐步求精的原则，把加工逐步细化。结构化设计在需求分析的基础上，要针对给定的问题给出软件解决方案。结构化设计中的总体设计部分要给出被建系统的模块结构，详细设计部分要为各模块提供关于算法的详细描述。

结构化方法比功能分解法更强调对问题域的分析，但所使用的建模概念仍然不能直接地映射到问题域中的事物。需求的变化往往会引起相应的加工和数据流的变化，进而影响与之相关的其他加工和数据流的变化。系统复杂时也不能检验分析模型的正确性。此外，结构化需求分析与后续的结构化设计所采用的概念与表示法是不一致的（基于不同的概念体系），且转换规则不严格、具体，仅是指导性的，这致使从需求分析模型过渡到设计模型较为困难。

人们用功能分解法和结构化方法已经开发了很多软件系统，但是同时由于上述原因，对系统的开发与维护问题也日益显现出来。对于功能稳定的应用领域，如某些科学计算，上述方法是适用的。但对于众多的领域而言，它们的需求是易变的，如企业管理和商业管理领域就是如此。因为随着市场的变化，要对这些领域的管理模式不断地进行调整。对于较为复杂的系统，用上述方法进行软件开发，容易导致模块的低内聚和模块间的高耦合，从而使得系统缺乏良好的灵活性和可维护性。加上当时团队的开发与管理方法的不足，这些因素使得在 20 世纪 70 年代的软件危机情况更加严重。为了解决软件危机，人们对开发技术进行了一定的改进，对编程语言也进行了革新，如产生了用于软件开发的 4GL、CASE 工具、原型技术和代码生成器。这些努力取得了一定的成就，但没有从根本上解决问题。

3. 信息建模方法

信息建模方法是在实体联系模型（entity relationship model）的基础上发展起来的。该方法以称为实体的数据集合作为系统的构造块，即以数据结构为中心来开发软件。因为有相当多的人认为实体是稳定的，并且实体联系模型有相当好的理论基础，所以当时该方法为很多开发团队所采用。

对于数据及其关系比较复杂的系统来说，信息建模方法很有用。但它也存在弱点，即它仅对问题域中事物的数据方面进行了建模，而对功能行为在模型中没有体现。这也是信息建模方法常常与其他开发方法相结合使用的一个原因。

包括上述方法在内的几乎所有的传统方法都只注重于系统的一个或少数几个方面，对系统的其他方面建模的能力都很弱。典型地，功能分解法集中于将功能作为系统的构造块，对数据组织的功能较弱，即使是在结构化方法中，对数据组织的支持也不是很强；在信息建模方法中的构造块是实体，强调对数据的组织，但在该方法中忽略了系统功能。此外，上述方法都没有较强的描述系统的动态行为的能力。

软件学术界和产业界尝试了数十年，一直在寻找有效的开发复杂软件系统的方法。经过坚持不懈的努力，形成了面向对象方法。

面向对象方法是在传统方法的基础上发展起来的，例如仍使用抽象和模块化等概念。然而，面向对象方法与传统方法相比发生了根本性的变化，主要在于面向对象方法具有从多维度把所建立的模型与问题域进行直接映射的能力，在整个开发过程中均采用一致的概念和表示法，采用诸如封装、继承和消息等机制使得问题域的复杂性在模型上得以控制。

1.2　面向对象的基本思想

面向对象方法已深入到计算机软件领域的几乎所有分支。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。因而，面向对象方法有着自己的基本思想。

面向对象方法解决问题的思路是从现实世界中的客观对象（如人和事物）入手，尽量运用人类的自然思维方式从不同的抽象层次和方面来构造软件系统，这与传统开发方法构造系统的思想是不一样的。特别是，面向对象方法把一切都看成是对象。下面以开发一个开发票的软件为例来说明这种观点。发票的样本如图 1-1 所示。

编号	名称	规格	单位	数量	单价	金额
合计						

图 1-1　发票样本

按非面向对象思路，要定义数据结构（如 C 中的结构或 Pascal 中的记录）以及编写根据数据结构进行计算的函数或过程。而按面向对象思路，先把发票看成一个对象，其中有若干属性，如编号、名称和规格等，还有若干操作，如计算一种商品金额的操作“单项金额计算”和计算金额合计的操作“发票金额合计”，然后再根据具体编程语言考虑怎样实现这个对象。

人们已经形成共识：面向对象方法是一种运用对象、类、继承、聚合、关联、消息和封装等概念和原则来构造软件系统的开发方法。下面具体地阐述面向对象方法的基本思想：

1）客观世界中的事物都是对象（object），对象间存在一定的关系。面向对象方法要求从现实世界中客观存在的事物出发来建立软件系统，强调直接以问题域（现实世界）中的事物以及事物间的联系为中心来思考问题和认识问题，并根据这些事物的本质特征和系统责任，把它们抽象地表示为系统中的对象，作为系统的基本构成单位。这可以使系统直接映射到问题域，保持问题域中的事物及其相互关系的本来面貌。

2）用对象的属性（attribute）表示事物的数据特征；用对象的操作（operation）表示事物的行为特征。

3）对象把它的属性与操作结合在一起，成为一个独立的、不可分的实体，并对外屏蔽它的内部细节。

4）通过抽象对事物进行分类。把具有相同属性和相同操作的对象归为一类，类（class）是这些对象的抽象描述，每个对象是它的类的一个实例。

5）复杂的对象可以用简单的对象作为构成部分。

6）通过在不同程度上运用抽象原则，可以得到较一般的类和较特殊的类。特殊类继承一般类的属性与操作。

7）对象之间通过消息进行通信，以实现对象之间的动态联系。

8）通过关联表达类之间的静态关系。

图 1-2 为上述部分思想的一个示意图。

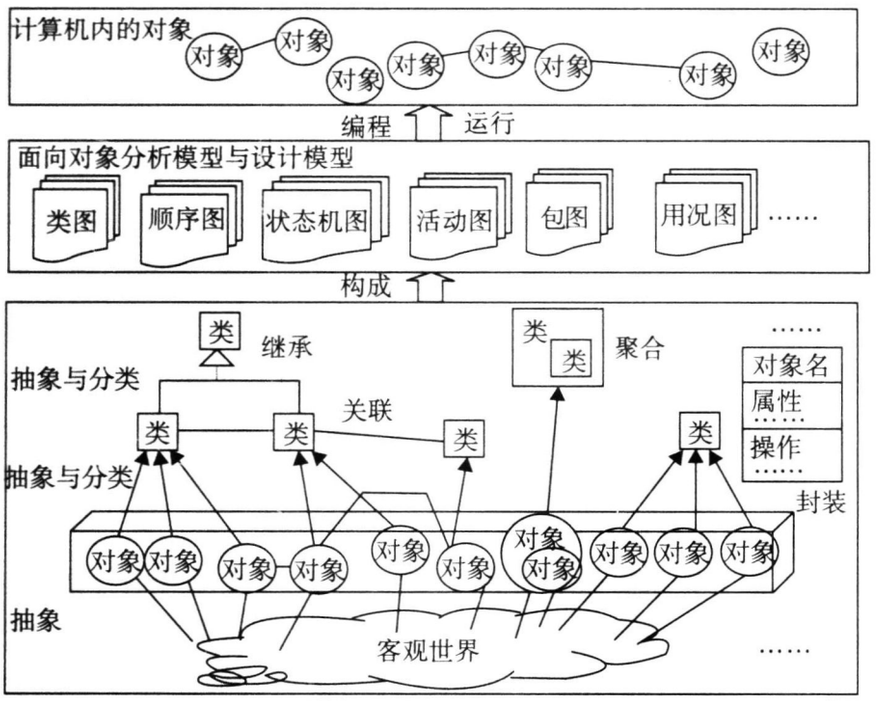


图 1-2　面向对象基本思想（部分）的示意图

利用抽象原则从客观世界中发现对象以及对象间的关系，其中包括整体对象和部分对象，进而再把对象抽象成类，把对象间的关系抽象为类之间的关系。通过继续运用抽象原则，确定类之间存在的继承关系。上述简略地说明了建立系统的静态结构模型的思想，系统其他模型的建立原则也与此类似，这些内容将是本书讲述的重点。通过以图形的方式作为建模的主要方式之一，分别建立系统的分析与设计模型，进而得到可运行的程序。正是通过面向对象建模，对所要解决的问题有了深刻且完整的认识，进而把其转换成可运行的程序，使得程序所处理的对象是对现实世界中对象的抽象。

从上述可以看出，面向对象方法强调充分运用人类在日常逻辑思维中经常采用的思想方法与原则，如抽象、聚合、封装和关联等。这使得软件开发者能更有效地思考问题，并以其他人也能看得懂的方式把自己的认识表达出来。为了更全面和清楚地表达认识，面向对象方法要用多种图来详述模型，即从多方面来刻画模型。像一些开发方法一样，面向对象方法也要求从分析、设计和实现等不同抽象层次（开发阶段）来开发复杂的软件系统。

面向对象方法也是多种多样的，尽管各种面向对象方法不同，但都是以上述基本思想为基础的。还要指出的是，一种方法要包含一组概念和相应的表示法以及用其构造系统的过程指导，面向对象方法也不例外。贯穿于本书的面向对象概念及表示法均取自 UML 2.4；至于过程指导，国内外尚无统一标准，本书给出的是基于特定活动而组织的，其特点是易学易用，且不失应用的普遍性。

1.3　面向对象的基本原则

面向对象的基本原则主要有抽象、分类、封装、消息通信、多态性、行为分析和复杂性控制。

（1）抽象

抽象（abstraction）是指从事物中舍弃个别的、非本质的特征，而抽取共同的、本质特征的思维方式。在面向对象方法中，可从几个方面来理解抽象：

1）编程语言的发展呈现抽象层次提高的趋势。例如，用C++编程，不用考虑CPU寄存器和堆栈中存放的内容，因为大多数编程语言都是从这些细节中抽象出来的。对于一个完成确定功能的语句序列，其使用者都可把它看作单一的实体（如函数），这种抽象就是过程抽象。在面向对象编程语言中，存在着过程抽象和数据抽象。在类的范围内，使用过程抽象来形成操作。数据抽象是指把数据类型和施加在其上的操作结合在一起，形成一种新的数据类型。类就是一种数据抽象，栈也是一种数据抽象。

2）在面向对象方法中，对象是对现实世界中事物的抽象，类是对对象的抽象，一般类是对特殊类的抽象。有的抽象是根据开发需要进行的。例如，就对象是对现实世界中的事物的抽象而言，高校中的学籍管理系统和伙食管理系统中所使用的学生的信息就是不一样的；再如，一个现实事物可能要担任很多角色，只有与问题域有关的角色，在系统中才予以考虑。

3）在面向对象的不同开发阶段需要进行不同程度的抽象。典型地，在面向对象分析阶段，先定义类的属性和操作，而与实现有关的因素在设计阶段再考虑。例如，对自动售货机建模，在分析阶段先定义一个类“自动售货机”，根据其收钱和发货的职责定义其属性和操作，其中对外提供的操作为收钱口、选择按钮和发货口（三者形成一个接口），而对于如何根据实现条件来设计它的内部细节是设计阶段的任务。这与现实生活中一样，我们可以在较高的抽象层次上分析与解决问题，然后再逐步地在较低抽象层次上予以落实。

从上述自动售货机的例子中能看到，使用抽象至少有如下好处：一是便于访问，外部对象只需知道有限的几个操作（作为接口）即可使用自动售货机对象；二是便于维护，如自动售货机的某部分有变化而其接口没有发生变化，只需在机器内部对该部分进行修改。甚至可用更优的具有相同接口的售货机对其进行替换，而不影响使用者的使用方式。

（2）分类

分类（classification）的作用是按照某种原则划分出事物的类别，以有助于认识复杂世界。

在OO中，分类就是把具有相同属性和相同操作的对象划分为一类，用类作为这些对象的抽象描述。如果一个对象是分类（类）的一个实例，它将符合该分类的模式。分类实际上是把抽象原则运用于对象描述时的一种表现形式。在OO中，进一步地还可以运用分类原则，通过不同程度的抽象，形成一般/特殊结构。

运用分类原则，清楚地表示了对象与类的关系，以及特殊类与一般类的关系。

（3）封装

封装（encapsulation）有两个含义：①把描述一个事物的性质和行为结合在一起，对外形成该事物的一个界限。面向对象方法中的封装就是用对象把属性和操纵这些属性的操作包装起来，形成一个独立的单元。封装原则使对象能够集中而完整地对应并描述具体的事物，体现了事物的相对独立性。②信息隐蔽，即外界不能直接存取对象的内部信息（属性）以及隐藏起来的内部操作，外界也不用知道对象对外操作的内部实现细节。在原则上，对象对外界仅定义其