21st CENTURY
实用规划教材

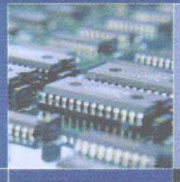
21世纪全国应用型本科计算机案例型规划教材

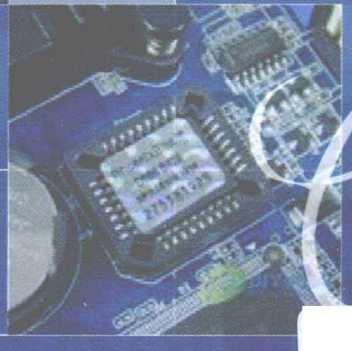
提供 电子课件 www.pup6.com

提供 素材 www.pup6.com

UML实用基础教程

赵春刚 主 编





Computer

- 基于UML统一建模语言，着重介绍基于面向对象思想的系统建模技术
- 主要章节每一章均设有经典的同步实验内容，理论与实践有机地结合
- 以完整案例做总结，提高综合运用所学理论解决实际完整案例的能力

北京大学出版社
PEKING UNIVERSITY PRESS

21 世纪全国应用型本科计算机案例型规划教材

UML 实用基础教程

主　编　赵春刚
副主编　才　智　高志军　宫　萍

北京大学出版社
PEKING UNIVERSITY PRESS

内 容 简 介

本书主要介绍了使用 UML 进行基于面向对象的软件建模的基础知识。主要包括面向对象开发方法及 UML 概述、UML 基础知识及相关案例分析、UML 与统一过程及 UML 建模工具等扩展内容 3 个重要组成部分。其中，第二部分采用以案例驱动的方式详细说明用例图、类图与对象图、包图、状态图、活动图、时序图、协作图、组件图和配置图建模的方法，并辅以实验指导，可用来指导相关的课程实验。

本书特点是基础与实例紧密结合，内容深入浅出、循序渐进、理论与实践相结合。

本书可作为高等院校计算机或软件工程相关专业本科生或研究生的教学用书，也可作为广大从事面向对象软件开发人员的学习指导用书。

图书在版编目(CIP)数据

UML 实用基础教程/赵春刚主编. —北京：北京大学出版社，2013.2

(21 世纪全国应用型本科计算机案例型规划教材)

ISBN 978-7-301-22119-8

Ⅰ. ①U… Ⅱ. ①赵… 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(2013)第 024980 号

书　　名：UML 实用基础教程

著作责任者：赵春刚　主编

策 划 编 辑：郑　双

责 任 编 辑：郑　双

标 准 书 号：ISBN 978-7-301-22119-8/TP•1272

出 版 发 行：北京大学出版社

地　　址：北京市海淀区成府路 205 号　邮编：100871

网　　址：http://www.pup.cn　　新浪官方微博：@北京大学出版社

电 子 信 箱：pup_6@163.com

电　　话：邮购部 62752015　发行部 62750672　编辑部 62750667　出版部 62754962

印 刷 者：三河市北燕印装有限公司

经 销 者：新华书店

787 毫米×1092 毫米　16 开本　18.25 印张　417 千字

2013 年 2 月第 1 版　　2013 年 2 月第 1 次印刷

定　　价：36.00 元

信息技术的案例型教材建设

(代丛书序)

刘瑞挺

北京大学出版社第六事业部在 2005 年组织编写了《21 世纪全国应用型本科计算机系列实用规划教材》，至今已出版了 50 多种。这些教材出版后，在全国高校引起热烈反响，可谓初战告捷。这使北京大学出版社的计算机教材市场规模迅速扩大，编辑队伍茁壮成长，经济效益明显增强，与各类高校师生的关系更加密切。

2008 年 1 月北京大学出版社第六事业部在北京召开了“21 世纪全国应用型本科计算机案例型教材建设和教学研讨会”。这次会议为编写案例型教材做了深入的探讨和具体的部署，制定了详细的编写目的、丛书特色、内容要求和风格规范。在内容上强调面向应用、能力驱动、精选案例、严把质量；在风格上力求文字精练、脉络清晰、图表明快、版式新颖。这次会议吹响了提高教材质量第二战役的进军号。

案例型教材真能提高教学的质量吗？

是的。著名法国哲学家、数学家勒内·笛卡儿(Rene Descartes，1596—1650)说得好：“由一个例子的考察，我们可以抽出一条规律。(From the consideration of an example we can form a rule.)”事实上，他发明的直角坐标系，正是通过生活实例而得到的灵感。据说是在 1619 年夏天，笛卡儿因病住进医院。中午他躺在病床上，苦苦思索一个数学问题时，忽然看到天花板上有一只苍蝇飞来飞去。当时天花板是用木条做成正方形的格子。笛卡儿发现，要说出这只苍蝇在天花板上的位置，只需说出苍蝇在天花板上的第几行和第几列。当苍蝇落在第四行、第五列的那个正方形时，可以用(4，5)来表示这个位置……由此他联想到可用类似的办法来描述一个点在平面上的位置。他高兴地跳下床，喊着“我找到了，找到了”，然而不小心把国际象棋撒了一地。当他的目光落到棋盘上时，又兴奋地一拍大腿：“对，对，就是这个图”。笛卡儿锲而不舍的毅力，苦思冥想的钻研，使他开创了解析几何的新纪元。千百年来，代数与几何，井水不犯河水。17 世纪后，数学突飞猛进的发展，在很大程度上归功于笛卡儿坐标系和解析几何学的创立。

这个故事，听起来与阿基米德在浴池洗澡而发现浮力原理，牛顿在苹果树下遇到苹果落到头上而发现万有引力定律，确有异曲同工之妙。这就证明，一个好的例子往往能激发灵感，由特殊到一般，联想出普遍的规律，即所谓的“一叶知秋”、“见微知著”的意思。

回顾计算机发明的历史，每一台机器、每一颗芯片、每一种操作系统、每一类编程语言、每一个算法、每一套软件、每一款外部设备，无不像闪光的珍珠串在一起。每个案例都闪烁着智慧的火花，是创新思想不竭的源泉。在计算机科学技术领域，这样的案例就像大海岸边的贝壳，俯拾皆是。

事实上，案例研究(Case Study)是现代科学广泛使用的一种方法。Case 包含的意义很广：包括 Example 例子，Instance 事例、示例，Actual State 实际状况，Circumstance 情况、事件、境遇，甚至 Project 项目、工程等。

我们知道在计算机的科学术语中，很多是直接来自日常生活的。例如 Computer 一词早在 1646 年就出现于古代英文字典中，但当时它的意义不是“计算机”而是“计算工人”，

即专门从事简单计算的工人。同理，Printer 当时也是“印刷工人”而不是“打印机”。正是由于这些“计算工人”和“印刷工人”常出现计算错误和印刷错误，才激发查尔斯·巴贝奇(Charles Babbage，1791—1871)设计了差分机和分析机，这是最早的专用计算机和通用计算机。这位英国剑桥大学数学教授、机械设计专家、经济学家和哲学家是国际公认的“计算机之父”。

20 世纪 40 年代，人们还用 Calculator 表示计算机器。到电子计算机出现后，才用 Computer 表示计算机。此外，硬件(Hardware)和软件(Software)来自销售人员。总线(Bus)就是公共汽车或大巴，故障和排除故障源自格瑞斯·霍普(Grace Hopper，1906—1992)发现的“飞蛾子”(Bug)和“抓蛾子”或“抓虫子”(Debug)。其他如鼠标、菜单……不胜枚举。至于哲学家进餐问题，理发师睡觉问题更是操作系统文化中脍炙人口的经典。

以计算机为核心的信息技术，从一开始就与应用紧密结合。例如，ENIAC 用于弹道曲线的计算，ARPANET 用于资源共享以及核战争时的可靠通信。即使是非常抽象的图灵机模型，也受到二战时图灵博士破译纳粹密码工作的影响。

在信息技术中，既有许多成功的案例，也有不少失败的案例；既有先成功而后失败的案例，也有先失败而后成功的案例。好好研究它们的成功经验和失败教训，对于编写案例型教材有重要的意义。

我国正在实现中华民族的伟大复兴，教育是民族振兴的基石。改革开放 30 年来，我国高等教育在数量上、规模上已有相当的发展。当前的重要任务是提高培养人才的质量，必须从学科知识的灌输转变为素质与能力的培养。应当指出，大学课堂在高新技术的武装下，利用 PPT 进行的“高速灌输”、“翻页宣科”有愈演愈烈的趋势，我们不能容忍用“技术”绑架教学，而是让教学工作乘信息技术的东风自由地飞翔。

本系列教材的编写，以学生就业所需的专业知识和操作技能为着眼点，在适度的基础知识与理论体系覆盖下，突出应用型、技能型教学的实用性和可操作性，强化案例教学。本套教材将会有机融入大量最新的示例、实例以及操作性较强的案例，力求提高教材的趣味性和实用性，打破传统教材自身知识框架的封闭性，强化实际操作的训练，使本系列教材做到“教师易教，学生乐学，技能实用”。有了广阔的应用背景，再造计算机案例型教材就有了基础。

我相信北京大学出版社在全国各地高校教师的积极支持下，精心设计，严格把关，一定能够建设出一批符合计算机应用型人才培养模式的、以案例型为创新点和兴奋点的精品教材，并且通过一体化设计、实现多种媒体有机结合的立体化教材，为各门计算机课程配齐电子教案、学习指导、习题解答、课程设计等辅导资料。让我们用锲而不舍的毅力，勤奋好学的钻研，向着共同的目标努力吧！

刘瑞挺教授 本系列教材编写指导委员会主任、全国高等院校计算机基础教育研究会副会长、中国计算机学会普及工作委员会顾问、教育部考试中心全国计算机应用技术证书考试委员会副主任、全国计算机等级考试顾问。曾任教育部理科计算机科学教学指导委员会委员、中国计算机学会教育培训委员会副主任。PC Magazine《个人电脑》总编辑、CHIP《新电脑》总顾问、清华大学《计算机教育》总策划。

前　言

UML 是面向对象可视化系统分析的一种功能较强的建模语言，采用较先进的建模技术，为软件开发的各个阶段提供了模型化和可视化的技术支持。它的各个模型可以帮助开发人员更好地理解业务流程，建立更可靠、更完善的系统模型，从而使用户和开发人员对问题的描述达到相同的理解，以减少语义差异，保障分析的正确性。

全书共分 14 章，以循序渐进的顺序介绍 UML，从基础知识开始，然后是分析、构建和部署。第 1 章主要介绍了面向对象技术和开发方法，包括面向对象的一些基本概念、面向对象分析、面向对象设计、面向对象编程语言和几种典型的面向对象方法等；第 2 章主要介绍了 UML 的基础理论，包括 UML 的特点、体系结构、模型图和模型元素、通用机制和扩展机制等；第 3 章主要介绍了需求建模技术，包括参与者和用例的概念、确定和描述用例技术、识别参与者技术及用例图建模技术等；第 4 章主要介绍了静态建模技术，包括类图、对象图和包图及静态建模技术等；第 5 章主要介绍了动态建模技术——状态图，包括状态图及状态图建模技术等。第 6 章主要介绍了动态建模技术——活动图，包括活动图及活动图建模技术等；第 7 章主要介绍了动态建模技术——时序图，包括时序图及时序图建模技术等；第 8 章主要介绍了动态建模技术——协作图，包括协作图及协作图建模技术等；第 9 章主要介绍了构架建模技术——组件图，包括组件图、组件图建模及应用技术等；第 10 章主要介绍了构架建模——配置图，包括配置图、配置图建模及应用技术等；第 11 章主要介绍了 UML 与统一过程基础知识，包括软件开发过程、Rational 统一过程和核心工作流程等；第 12 章主要介绍了相关的 UML 建模工具，包括 Rational Rose 2003、Microsoft Visio 2003、JUDE 工具等；第 13 章以 Rational Rose 为建模工具，以图书管理系统为例，将前面章节中所介绍的 UML 的各种图形及模型元素综合起来，形成一个对图书管理系统的建模实例；第 14 章以超市信息管理系统为例介绍基于 UML 的面向对象系统分析与设计过程。

本书由黑龙江科技学院的赵春刚、才智、高志军和宫萍共同编写完成。其中第 1、3、4 和 13 章由赵春刚编写；第 2、12 和 14 章由宫萍编写；第 5～8 章由才智编写；第 9～11 章由高志军编写。

在编写本书的过程中参考了一些文献资料，在此对这些文献的作者表示衷心的感谢。

由于编者水平有限，书中难免存在不妥之处，恳请广大读者批评指正。

编　者

2012 年 9 月

目　录

第 1 章　面向对象的开发方法

面向对象思想最初出现在由挪威奥斯陆大学和挪威计算机中心共同研制的仿真语言 Simula 67 中。之后，美国 Xerox 研究中心推出了面向对象的 Smalltalk-76 和 Smalltalk-80 语言，使得面向对象的程序设计方法首次得以比较全面地实现。

面向对象方法是一种较新的软件开发方法，基于一种面向对象的方法论，其基本观点是：

(1) 把世界上的任何事物都看做对象，整个世界就是由众多的简单对象逐步构成越来越复杂的对象，可见对象是分层次的。

(2) 所有对象被划分成各种各样的对象类，各种对象类之间可构成各种层次关系。

(3) 对象之间都是通过“消息传递”来互相联系的。

(4) 对象类把其所含对象及其属性和方法等封装在一起。

本章主要从面向对象的基本概念、面向对象分析与设计、面向对象程序设计、面向对象建模等方面进行介绍。

1.1　面向对象技术概述

面向对象的基本思想是从现实世界中客观存在的事物(即对象)出发来构造软件系统，并在系统构造中尽可能运用人们的自然思维方式。同时，面向对象从一开始就支持软件重用，这为其发展奠定了基础。软件的可重用将直接导致软件开发成本的降低和软件质量的提高。另外，由于对象本身具有的“自治”特点，使得面向对象系统的可扩展性和可维护性大大提高。

1.1.1　面向对象技术的由来

面向对象方法的形成最初是从面向对象程序设计语言开始的，随之逐步形成面向对象的分析和设计方法。面向对象方法与技术的发展经历了 3 个阶段。

1. 雏形阶段

最早的面向对象语言是 20 世纪 60 年代的 Simula 67 语言。虽然它是一种通用的仿真建模语言，但其所使用的对象概念和方法，给人们以软件设计的新启示，该语言的诞生是面向对象发展史上的第一个里程碑。20 世纪 70 年代的 CLU、并发 Pascal、Ada 等语言对抽象数据类型理论的发展起到了重要的作用，它们支持数据与操作的封装。1972 年 Palo Alno 研究中心发布了 Smalltalk-72，正式使用了“面向对象”这个术语。Smalltalk 的问世标志着面向对象程序设计方法的正式诞生，但这个时期的 Smalltalk 语言还不够完善。

2. 完善阶段

1981 年 Smalltalk-80 的推出，使面向对象程序设计方法得到比较完善的实现。Smalltalk 的目标是使软件尽可能以“自治”的单元来设计。迄今为止绝大部分面向对象的基本概念及其支持机制在 Smalltalk-80 中都已具备。但是由于一种新的软件方法被广泛地接受需要一定的

时间，同时商品化软件开发工作到 1987 年才开始，所以在 20 世纪 80 年代 Smalltalk 的应用尚不够广泛。

3. 繁荣阶段

Smalltalk 的发布引起了人们的广泛关注，随后产生了包括 C++、Java 在内的数十种面向对象的程序设计语言，以及相应的程序设计环境，先后成为面向对象技术发展的重要里程碑。各种面向对象语言的出现直接导致了面向对象的广泛应用，面向对象技术也很快被应用到系统分析和系统设计方法中。

1.1.2 面向对象的基本思想

面向对象的基本思想如下：

(1) 从问题域中客观存在的事物出发来构造软件系统，用对象作为对客观事物的抽象表示，并以此作为系统的基本构成单位。

(2) 事物的静态特征(即可以用一些数据表达的特征)用对象的属性来表示，事物的动态特征(即事物的行为)用对象的操作表示。对象的属性与操作结合为一体，成为一个独立的实体。

(3) 相同属性和相同操作的对象归为一类，对象是类的一个实例。

(4) 通过在不同程度上运用抽象原则(忽略事物之间的一些差异)，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和操作，面向对象方法支持这种继承关系的描述与实现，从而简化系统的构造过程及其文档。

(5) 复杂对象可以用简单的对象作为其构成部分(称为聚合)。

(6) 对象之间通过消息进行通信，以实现对象之间的动态联系。

(7) 通过关联表达对象之间的静态关系。

从以上几点可以看出，面向对象强调以问题域中的事物为中心来思考问题、认识问题，并根据这些事物的本质特征，把它抽象地表示为系统中的对象，作为系统的基本构成单位。面向对象方法可以使系统直接地映射问题域，保持问题域中事物及其相互关系的本来面貌。面向对象方法是一种运用对象、类、继承、封装、聚合、关联、消息和多态性等概念和原则来构造系统的软件开发方法。

1.1.3 传统开发方法存在的问题

传统开发方法(结构化方法)存在以下几个问题。

(1) 软件重用性差。重用性是指同一事物不经修改或稍加修改就可多次重复使用的性质。软件重用性是软件工程追求的目标之一。

(2) 软件可维护性差。软件工程强调软件的可维护性，强调文档资料的重用性，固定最终的软件产品应该由完整、一致的配置成分组成。在软件开发过程中，始终强调软件的可读性、可修改性和可测试性是软件重要的质量指标。实践证明，用传统方法开发出来的软件，维护时其费用和成本仍然很高，其原因是可修改性差、维护困难，导致可维护性差。

(3) 开发的软件不能满足用户需要。用传统的结构化方法开发大型软件系统涉及各种不同领域的知识，在开发需求模糊或需求动态变化的系统时，所开发出的软件系统往往不能真正满足用户的需要。

用结构化方法开发的软件，其稳定性、可修改性和可重用性都比较差。这是因为结构化

方法的本质是功能分解，从代表目标系统整体功能的单个处理着手，自上而下不断把复杂的处理分解为子处理，这样一层一层地分解下去，直到分解为若干个容易实现的子处理功能为止，然后用相应的工具来描述各个底层的处理。因此，结构化方法是围绕实现处理功能的“过程”来构造系统的。然而，用户需求的变化大部分是针对功能的，因此，这种变化对于基于过程的设计来说是灾难性的。用这种方法设计出的系统结构常常是不稳定的，用户需求的变化往往造成系统结构的较大变化，从而需要花费很大代价才能实现这种变化。

1.1.4　面向对象方法的优点

面向对象方法主要有以下优点。

(1) 与人们习惯的思维方法一致。面向对象方法以对象为核心，提供了对现实世界更加完整的描述，涉及数据、行为和通信等。在面向对象方法中，计算机观点已被淡化，现实世界模型成为构造系统的最重要的依据。该方法鼓励开发人员，能够更多地使用用户应用领域中的概念去思考问题。在开发过程中自始至终都在考虑如何建立问题域对象模型；从对问题域进行自然的对象分解，到确定需要使用的类、对象；再到在对象之间建立消息通道以实现对象间的联系等。面向对象技术帮助人们先设计出由抽象类(概念类)构成的系统框架，然后随着认识的深入，逐步派生出更加具体的派生类。这样的开发过程与人们在解决复杂问题时逐步深化的渐进过程是一致的。

(2) 可使软件系统结构更加稳定。与传统方法不同，面向对象方法以对象为中心来构造软件系统，它的基本做法是用对象模拟问题域中的实体，并以对象之间的联系来表现领域实体之间的联系。因此当对系统的功能需求改变时，一般仅需要对一些局部对象进行修改，而并不需要改变整个软件结构。例如，可以从已有的类中派生出一些新的子类，以实现系统功能的扩充或修改。由于现实世界中的实体是相对稳定的，因此，以对象为中心来构造的软件系统也比较稳定。

(3) 软件具有更好的可重用性。软件重用是提高软件生产率的最主要的途径之一。面向对象技术可使软件系统具有更大的灵活性，其主要的重用机制是利用类的继承性，可以通过上级父类派生出下级子类，这不仅可以重复使用父类的数据结构和程序代码，并且可以在其父类代码的基础上，方便地进行子类的修改和扩充。面向对象技术所实现的可重用性是自然的和准确的，在重用技术中面向对象技术是最成功的一个。

(4) 软件更加便于维护和扩充。利用面向对象方法开发的软件系统便于维护与扩充。当软件需求发生改变时，通常不会引起软件体系发生整体变动，只需要逐个改变局部类模块，而不影响其他部分。类是独立性较强的模块，因此当修改该类时，不易引起“波动效应”，只要接口不变，就不影响其他部分。另外，使用面向对象方法开发的软件比较容易理解，因为软件系统结构与问题域空间结构具有较好的一致性。

1.2　面向对象的基本概念

要深入理解面向对象的思想和原理，首先必须了解面向对象的基本概念。本节着重介绍对象、类、封装、继承、聚合、关联、消息、多态性等面向对象的基本概念，这些概念是系统建模的基础。

1.2.1 对象

对象无处不在，组成了整个世界。可以从两个角度来理解对象：一个角度是现实世界，另一个角度是所建立的系统。按照人们的认识角度，从客观世界中任何有确定边界、可触摸、可感知的事物，到某种可思考、可认识的概念(如速度、时间等)均可认为是对象。例如，在客观世界中，学生、教师、电视、汽车等都是对象的例子，表示客观世界中的某些概念。每个对象有其自身的属性。例如，学生有学号、性别、年龄、年级、专业、成绩等。对象的属性值可因施加于该对象上的行为动作而变更。例如，根据学生升留级的情况，改变学生的年级属性值。对于所要建立的特定系统模型来说，现实世界中的有些对象是有待于抽象的事物。

在面向对象的系统模型中，对象是构成系统的一个基本单位，是指问题域中某些事物的一个抽象，反映该事物在系统中需要保存的信息和发挥的作用，是由数据(属性)及其上的操作(也称为服务、方法或行为)组成的封装体。对象的属性是用来描述对象的静态特征的一个数据项，可以是系统或用户定义的数据类型，也可以是一个抽象的数据类型。对象的行为是定义在对象属性上的一组操作方法的集合，是用来描述对象动态特征的一个动作序列，可以从以下 4 个方面来认识对象。

(1) 从动态的观点看，对象的操作就是对象的行为。

(2) 从存储的角度看，对象是私有存储，其中有数据也有操作。其他对象的方法不能直接操纵该对象的私有数据，只有对象私有的方法才可以操纵它。

(3) 从实现的机制看，对象是一个自动机，其中私有数据表示了对象的状态，该状态只能由私有的方法改变它。

(4) 在面向对象的程序设计中，对象是系统中的基本运行实体。

系统中的每一个对象，在软件生命周期的各个阶段可能有不同的表现形式。在分析阶段对象主要是从问题域中抽象出来的、反映概念的实体对象；在设计阶段则要结合实现环境增加用户界面对象、数据存储对象等；到实现阶段则要用一种程序设计语言写出详细而确切的源程序代码。这说明，系统中的对象要经过若干演化阶段，其表现重点、形式各异，但在概念上是一致的——都是问题域中某一事物的抽象表示。

“对象”一词在许多场合用法含糊。有时对象表示单一物体，有时它指的是一组相似的物体，通常根据上下文可以确定它的意思。当要求比较精确或明确指某件物体时，可用词组“对象实例”，而“对象类”指一组相似的物体。

图 1.1 “汽车”类图

1.2.2 类

类是指具有相同属性和操作的对象的集合，它代表一种抽象，作为具有类似特征与共同行为的对象的模版，可用来产生对象。类的概念使人们能对属于该类的全部个体事物进行统一的描述。例如，人类、汽车、树等都是一些抽象概念，它们是一些具有共同特征的事物的集合。图 1.1 是用于描述“汽车”的类图，其类名是“汽车”，它具有“类别”、“生产厂家”、“出厂日期”等属性，操作有“驱动”、“制动”等，这些特征适合所有的汽车。

把具有共同性质的事物划分为一类，得出一个抽象的概念，是人们在认识客观世界时经常采用的思维方法。分类所依据的原则是抽象，即忽略事物的非本质的特征，只注意那些与

当前目标有关的本质特征，从而找出事物的共性。

UML 对类的定义是：说明一系列拥有相同的属性、操作、方法、关系、行为的对象集。在软件开发中，一个类就代表了该建模系统中的一个概念。根据模型种类的不同，此概念可能是现实世界的(对于分析模型)，或含有算法和计算机实现的概念(对于设计模型)。

1.2.3　封装

封装是面向对象的一个重要原则。封装是指把对象的外部特征与内部实现细节分开，使得一个对象的外部特征对其他对象来说是可访问的，而它的内部细节对其他对象是隐藏的。从外部只能看到对象的行为(操作)，而并不知道那些存在的操作是怎样工作的。例如，用“售报亭”对象描述现实中的一个售报亭。它的属性是售报亭内各种报纸杂志的名称、定价和钱箱(总金额)，有两个服务(操作)：报刊零售和货款清点。封装意味着售报亭对象是由属性和操作结合成的一个整体，它通过售报亭的窗口对外提供零售服务。顾客只能从这个窗口请求服务，而不能自己到售报亭内去拿报纸或找零。货款清点则是一个内部操作，不向客户开放。

一个对象具有封装性的条件如下：

(1) 有一个清楚的边界，所有私有数据和操作的代码都被封装在这个边界内，从外面看不见更不能访问。

(2) 有确定的接口，接口是可见的，这些接口描述这个对象和其他对象之间的相互作用。

(3) 受保护的内部实现，实现对用户来说是不可见的，这个实现给出了由软件对象提供的功能细节，实现细节能在定义这个对象的类的外部访问。

通过封装及信息隐藏就避免了外部错误对它的“交叉感染”，另外对象内部修改对外部的影响很小，减少了修改引起的“波动效应”。

封装的另一个目的在于将对象的使用者和开发人员分开。当对某一对象进行修改时，不影响其他对象的使用。加工操作是对象的一部分，可通过向对象传送一个消息而启动。对象之间只能通过消息进行通信。消息是一个对象为实现其责任而与其他对象的通信。

严格的封装也会带来问题，如编程麻烦、执行效率受损。有些语言不强调严格的封装和信息隐藏，而实现可见性控制，以此来解决问题，如 C++和 Java，通过定义对象的属性和操作的可见性，对外规定了其他对象怎样获得对其属性和操作的访问。

1.2.4　继承

继承是面向对象描述类之间相似性的重要机制。在现实世界中大量的实体都存在一定程度的相似性，人们总是希望能够最大程度地利用种种相似性，不仅在管理系统的类的时候，而且在定义新的类的时候也希望通过利用这种相似性来简化工作，并重用之前的工作。继承刻画了一般性和特殊性，在软件开发中定义特殊类时，不需要把它的一般类已经定义过的属性和服务重复地书写一遍，只需要声明它是某个类的特殊类即可。继承具有“是一种”的含义。例如，人们认识了汽车的特征之后，在考虑货车时只要知道货车也是一种汽车这个事实，就理所当然地知道货车具有汽车的全部特征，只要把精力用于发现和描述客车独有的那些特征即可。如图 1.2 所示，在类的继承层次结构中，位于较高层次的类称为一般类，而位于较低层次的类称为特殊类。继承还具有传递性。例如，客车具有汽车的全部特征。

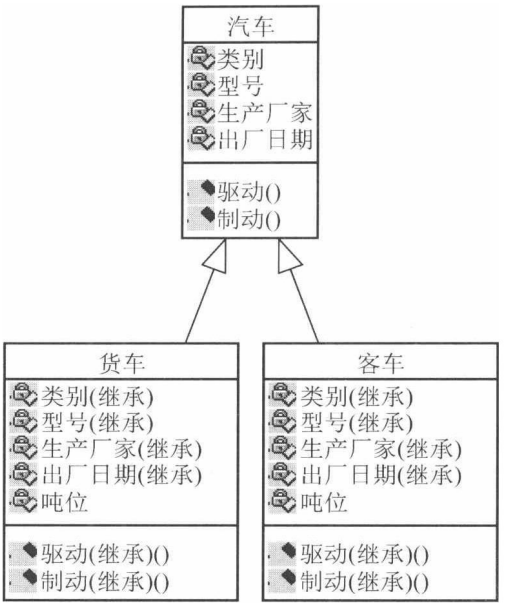


图 1.2　类的继承

继承的重要意义在于它简化了人们对事物的认识和描述。利用继承可以达到重用公共描述的目的，这将产生更容易理解的、更小的模型。另外，当人们想修改或增加新的特征时，只需要修改相关地方即可。例如，增加汽车的“颜色”属性，只需在类“汽车”中进行修改即可。

继承也体现了类的层次关系。在已有的类的基础上构造新的类，前者称为父类(或超类、基类、一般类)，后者称为子类(或派生类、特殊类)。子类除了自动拥有父类的全部属性和操作外，还可以进一步定义新的属性和操作。例如，固定电话、手机、小灵通等都属于电话机类产品，而电视机、电话机、洗衣机等都是电子产品，它们具有电子产品的公共特征。为了避免重复编码，可以将这些电子产品的公共特性部分定义为电子产品类，而将电视机、电话机、洗衣机定义为电子产品的子类，将固定电话、手机、小灵通等定义为电话机产品的子类。

继承可以分为单继承和多继承。如果子类只从一个父类继承则称为单继承；如果子类从多个父类继承则称为多继承。例如，两栖交通工具同时继承地面交通工具和水上交通工具的特征。多继承虽然比较灵活，但多继承可能会带来“命名冲突”的问题。在实际阶段，不同的程序设计语言可能有不同的解决方法。例如，C++采用成员名限定方法来解决，而 Java 不支持多继承，如果要实现类似于多继承的功能，则采用接口来实现。

覆盖和重载是面向对象技术中常用的两个术语，很容易混淆。在子类中可以增加或重新定义所继承的属性或方法，如果是重新定义，则称为覆盖。与覆盖很类似的一个概念是重载。重载指的是一个类中有多个同名的方法，但这些方法在操作数或操作数的类型上有区别。

如果把用面向对象方法开发的类作为可重用组件提交到组件库，那么在开发新软件时不仅可以直接地重用这个类，还可以把它作为一般类，通过继承而实现重用，从而大大扩展了重用的范围。

1.2.5　聚合

现实世界中既有比较简单的事物，也有比较复杂的事物。人们在认识比较复杂的事物时，

常常采用这样一种思想方式：把复杂事物看做由若干比较简单的事物构成。在一个复杂事物的内部识别出若干相对独立的组成部分，把它们作为一些比较简单的事物，认识它们的特征，然后由它们构成复杂事物。这种思想方法反映在面向对象方法中就是聚合。

一个复杂的对象以若干比较简单的对象作为其组成部分称为聚合。

聚合是面向对象方法的基本概念之一。它是一种系统构造原则，即由比较简单的对象构成比较复杂的对象。同时它也是对象之间的一种关系。即整体对象和部分对象之间的关系。整体对象和部分对象也是相对而言的，前者描述了一个复杂事物的整体，后者则描述复杂事物中的一个相对独立的局部。整体对象和部分对象之间的关系便是聚合关系，又称整体-部分关系。

在现实世界中，有些事物之间的组成关系是紧密而固定的。例如，人体的四肢和内脏是一个人与生俱来、不可分割的组成部分。有些事物之间的组成关系则是松散而灵活的，如一个公司和这个公司的经理，虽然也是整体与部分的关系，但是这种关系并不是一成不变的，反映在面向对象方法中，聚合关系也分为紧密固定的和松散灵活的两种情况。在 UML 中把紧密固定的聚合关系称为组合(Composition)，而聚合则泛指所有的情况。在以往的某些面向对象分析与设计方法中组合这个术语也被用做聚合的同义词。

聚合关系有两种实现方式。第一种方式是用部分对象作为整体对象的一个属性(这个属性的数据类型是部分对象的类，因此它不是一个普通数据，而是一个对象，即部分对象)，从而构成一个嵌套对象。在这种方式下，一个部分对象只能隶属于唯一的整体对象，并与它同生同灭。第二种方式是独立地定义、创建整体对象和部分对象，并在整体对象中设置一个属性，它的值是部分对象的对象标识，或者是一个指向部分对象的指针。在这种方式下，整体对象和部分对象可以有不同的生存期。整体对象中的一个部分可以在不同的时刻更换成不同的对象；一个部分对象在不同的时刻可以属于不同的整体，而且可以同时属于多个整体。虽然，前一种实现方式便于表示紧密、固定的聚合关系(即组合关系)；后一种实现方式便于表示松散、灵活的聚合关系。

整体-部分结构是把一组具有聚合关系(即整体-部分关系)的类组织在一起所形成的结构。它是一个以类为结点，以聚合关系为边的连通有向图。

整体-部分结构描述了对象之间的组成关系，即一些对象是另一些对象的组成部分。在模型中，整体对象和部分对象都是通过它们的类来表示的，二者之间的这种组成关系也是通过在它们的类之间画出的关系表示符号来表达的。这就是说，整体-部分结构的表达是在类的抽象层次上进行的，而它的语义则是描述了对象实例之间的组成关系。一个整体-部分结构可以包含两个或者两个以上的类，其中所包含的聚合关系可以是一个或多个。图 1.3 显示了魔术师(juggler)的一个简单例子。图中的菱形表示一个对象由另一个对象组成。数字表示有多少个部分。一个魔术师 i 有两只手(hand)和两只脚(foot)，他用手抓住(catch)、抛出(throw)球，也可以用脚踢(kick)球。

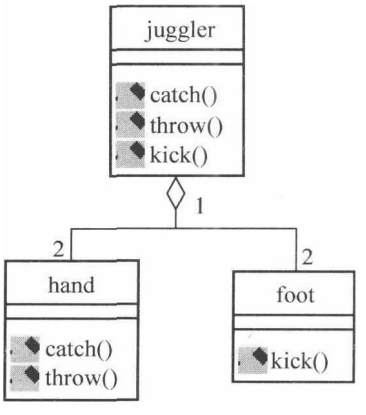


图 1.3　聚合的一个简单例子

1.2.6　关联

面向对象的思路是把事物之间的各种关系归结为泛化、