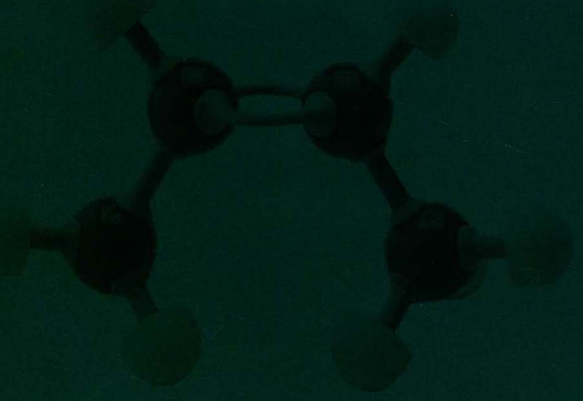
卓越工程师教育培养计算机类创新系列规划教材

UML面向对象需求分析与建模教程

主　编　邹盛荣
副主编　周　塔
顾爱华
彭昱静



科学出版社

卓越工程师教育培养计算机类创新系列规划教材

UML 面向对象需求分析与建模教程

主　编　邹盛荣

副主编　周　塔　顾爱华　彭昱静

科学出版社

北京

内 容 简 介

本书主要介绍基于 UML2.5 标准的系统建模基本理论、软件分析与设计方法，书中加强了软件案例的 UML 示例说明，以提高学生的软件分析与设计水平，进一步拓展学生分析问题、解决问题的能力，达到培养“厚基础、宽口径、会应用、能发展”的卓越人才培养宗旨。

全书共 13 章，内容包括绪论，面向对象方法、统一建模语言、RUP 统一过程、工具、UML 更多细节，系统的需求获取、分析、设计、实现和测试、UML 高级课题，案例介绍等。每章均有工程实践中的相关案例说明及实践应用的创意思考和提示，书的最后一章重点描述一个完整的 UML 建模课程设计案例。

图书在版编目(CIP)数据

UML 面向对象需求分析与建模教程 / 邹盛荣主编. —北京：科学出版社，2015.5
卓越工程师教育培养计算机类创新系列规划教材
ISBN 978-7-03-044467-7

Ⅰ. ① U… Ⅱ. ①邹…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(2015)第 114341 号

责任编辑：王正飞 / 责任校对：郑金红
责任印制：霍 兵 / 封面设计：迷底书装

科学出版社 出版
北京东黄城根北街 16 号
邮政编码：100717
http://www.sciencep.com

大厂书文印刷有限公司 印刷

科学出版社发行 各地新华书店经销

*

2015 年 6 月第 一 版 开本：787×1092 1/16
2015 年 6 月第一次印刷 印张：10 1/2
字数：248 000

定价：30.00 元

(如有印装质量问题，我社负责调换)

前　言

国家启动“卓越工程师教育培养计划”（简称“卓越计划”）的目的是加紧培养一批创新性强、能够适应经济和社会发展需求的各类工程技术人才，着力解决高等工程教育中的实践性和创新性问题，提高科技创新能力，这对于加快经济发展方式的转变、实现未来我国经济社会的持续发展具有重要意义。

“卓越计划”具有三个特点：一是行业企业深度参与培养过程；二是学校按通用标准和行业标准培养工程人才；三是强化培养学生的工程能力和创新能力。

为了配合该计划的实施，我们总结了近几年卓越工程师的教学经验，按照卓越工程师的要求编写了本书。本书主要介绍 UML 系统建模的基本理论、软件分析与设计方法，书中加强了软件案例的 UML 示例说明，以提高学生的软件分析与设计水平，进一步拓展学生分析问题、解决问题的能力，达到培养“厚基础，宽口径，会应用，能发展”的卓越人才的宗旨。

全书共 13 章，内容包括绪论，面向对象方法，统一建模语言，RUP 统一过程，ROSE 建模工具，UML 的进一步讨论，UML 系统建模过程的需求获取、需求分析、设计、实现、测试，UML 的形式化，综合案例等。每章均有工程实践中的相关案例说明及实践应用的创意思考和提示，本书最后一章重点描述了一个完整的 UML 建模课程设计案例。

本书内容深入浅出，通俗易懂，具有很好的可读性，实用性强。案例引导为主，不讲太多 UML 理论；按软件系统的大小分类讲述建模，逐步引导和培养学生实践能力；结合 RUP 统一过程，符合软件工程的过程需要；本书配有精美 PPT，可作为大学本科软件工程类、计算机等专业的教材或参考书，可有针对性地应用于卓越工程师培养计划，可供各类研究生及科研人员参考使用，还可供从事软件开发应用的工程技术人员参考。

书中标有“*”的章节属于前沿课题，老师可不讲，有兴趣的学生可参考学习，并查找相关书籍或网上资源。

本书编写完成，向历届开课学生表示感谢，向参与其中的各位老师表示感谢，有了你们的支持和帮助，使本书编排更合理并容易接受，让读者更容易从中学会并理解理论知识，在问题的思考和讨论中学会创新思维能力，并能够运用到将来的工作中。

编者

2015.6

目　录

第一部分　建模理论概述

第二部分 UML 系统建模的过程

第三部分 高级课题

第四部分 实验案例

第1章 绪 论

“UML 系统建模”是一门与软件开发密切相关的建模课程。1968 年软件工程产生后，软件分析与设计的技术在 20 世纪 80 年代末至 90 年代中期出现了一个发展高潮，UML 是这个高潮的产物。它不仅统一了 Booch、Rumbaugh 和 Jacobson 的建模表示方法，而且使其有了进一步的发展，并最终统一为大众所接受的标准建模语言（unified modeling language，UML）。

1.1 UML 的发展史

工程师为什么要建造模型？航天工程师为什么要建造航天器的模型？桥梁工程师为什么要建造桥梁模型？建造这些模型的目的是什么？

工程师建造模型来查明他们的设计是否可以正常工作。航天工程师建造好了航天器的模型，然后把它们放入风洞中了解这些航天器是否可以飞行。桥梁工程师建造桥梁模型来了解桥是否稳固。建筑工程师建造建筑模型可以了解客户是否喜欢这种建筑样式。通过建立模型来验证建造事物的可行性。

UML 系统建模是一种与面向对象软件开发密切相关的建模方法。各种面向对象的分析与设计方法都为面向对象理论与技术的发展作出了贡献。这些方法各有自己的优点和缺点，同时在各自不同范围内拥有自己的用户群。各种方法的主导思想以及所采用的主要概念与原则大体上是一致的，但是也存在不少差异。这些差异所带来的问题是，不利于面向对象方法向一致的方向发展，也给用户的选择带来了一些困惑。为此，Rational 公司的 Booch 和 Rumbaugh 决定将他们各自的方法结合起来成为一种方法，并于 1995 年 10 月发布了第 1 个版本，称为“统一方法”（Unified Method 0.8）。此时 OOSE 的作者 Jacobson 也加入了 Rational 公司，于是也加入了统一行动。1996 年 6 月发布了第 2 个版本 UML 0.9。鉴于统一行动的产物只是一种建模语言，而不是一种建模方法（因为不包含过程指导），所以自 UML 0.9 起，改称“统一建模语言”。在此过程中，由 Rational 公司发起成立了 UML 伙伴组织。开始时有 12 家公司加入，共同推出了 UML 1.0 版，并于 1997 年 1 月提交到对象管理组织（OMG）申请作为一种标准建模语言。此后，又把其他几家向 OMG 提交建模语言提案的公司扩大到 UML 伙伴组织中，并汲取意见对 UML 进一步作了修改，产生了 UML 1.1 版。该版本于 1997 年 11 月 4 日被 OMG 采纳。此后 UML 还在继续改进，目前最新的版本是 UML 2.5（www.UML.org）。

OMG 提交给国际标准化组织（ISO）的 UML 1.4 已经通过审核成为国际标准（ISO/IEC 19501：2005）。UML 早期发展过程如图 1-1 所示。

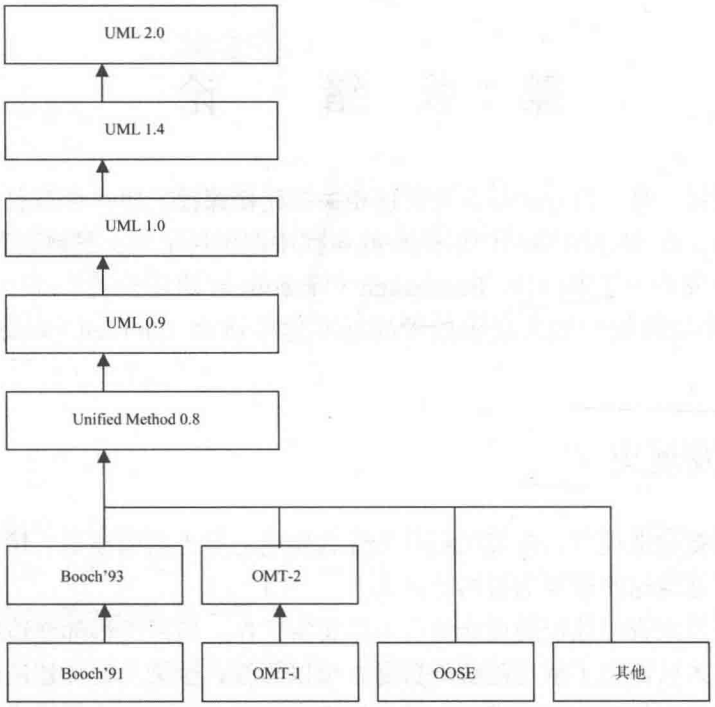


图 1-1 UML 的发展史

1.2 日常生活中的应用

模型是用某种工具对某个系统的表达方式。模型从某一个建模观点出发，抓住事物最重要的方面而简化或忽略其他方面。工程、建筑和其他许多需要创造性思想的领域中都使用模型。

表达模型的工具要便于使用。建筑模型可以是图纸上所绘的建筑图，也可以是用厚纸板制作的三维模型，还可以用存于计算机中的方程来表示。一个建筑物的结构模型不仅能够展示这个建筑物的外观，还可以进行工程设计和成本核算。

软件系统的模型用建模语言来表达，如 UML。模型包含语义信息和表示法，可以采取图形和文字等多种不同形式。

雾霾的研究可以建造模型吗？建造模型的好处有哪些？

UML 的目标是以面向对象各种相关图的方式来描述任何类型的系统。最常用的是建立软件系统的模型，但 UML 也可用来描述其他非计算机软件的系统或者商业机构或过程，以下是 UML 常见的应用。

信息系统：向用户提供信息的存储、检索、转换和提交处理存放在关系或对象数据库中大量具有复杂关系的数据。

技术系统：处理和控制技术设备，如电信设备、军事系统或工业过程，它们必须处理设计的特殊接口，标准软件很少，技术系统通常是实时系统。

嵌入式实时系统：在嵌入其他设备（如移动电话、汽车、家电）的硬件上执行的系统通常是通过低级程序设计进行的，需要实时支持。

分布式系统：分布在一组机器上运行的系统，数据很容易从一台机器传送到另一台机器，需要同步通信机制来确保数据完整性，通常是建立在对象机制上的，如 CORBA、COM/DCOM 或 Java Beans/RMI 上。

系统软件：定义了其他软件使用的技术基础设施，操作系统、数据库和在硬件上完成底层操作的用户接口等，同时提供一般接口供其他软件使用。

商业系统：描述目标、资源、人、计算机规则、法规、商业策略、政策等，以及商业中的实际工作、商业过程。商业系统是面向对象建模应用的一个重要的领域，引起了人们极大的兴趣，面向对象建模非常适合为公司的商业过程建模，运用全质量管理等技术，可以对公司的商业过程进行分析、改进和实现，使用面向对象建模语言为过程建模和编制文档，使过程更易于使用。

UML 具有描述以上这些类型系统的能力。

1.3 本课程学习中需要注意的问题

建模是计算机学科所特有的吗？其他领域是否需要建模？设想造一幢摩天大厦，如果没有图纸将会是怎样的结果？

教师可按照章节进行一步步的理论教学，实践性的案例贯穿在相关章节中，有兴趣的学生可主动按照案例进行模仿建模，探索学习，主动学习，理论联系实际学习效果将更好。

（1）简单的案例描述在第 3 章讲述，主要引导学生观察了解 UML 常用的 9 种图的画法及作用，图形简单易用，便于学习，可立即上手模仿实践。

（2）中型案例在全书的第二部分按章节详细讲述，按照统一过程的各过程流在第二部分的每章结尾处展开说明。

图 1-2 是按软件 RUP 过程设置建模的知识点概况。

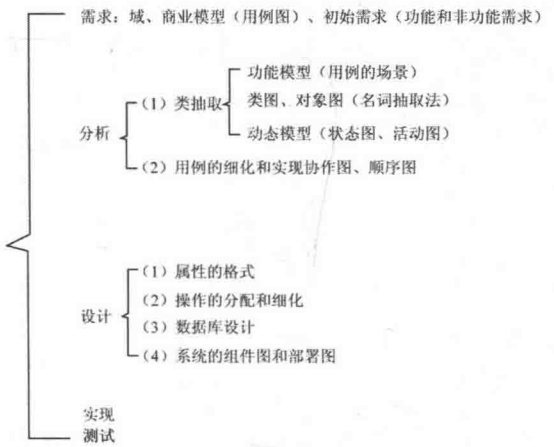


图 1-2 建模的知识点框图

学期开始，学生可在老师的协助指导下明确一个感兴趣的软件案例作为学期学习目标，教师讲理论的同时学生自己动手一步步完成案例。教师课上讲解的时候，学生也可举一反三，并结合自己所做案例思考自己的经验和教训。图 1-3 是软件人才的发展途径。

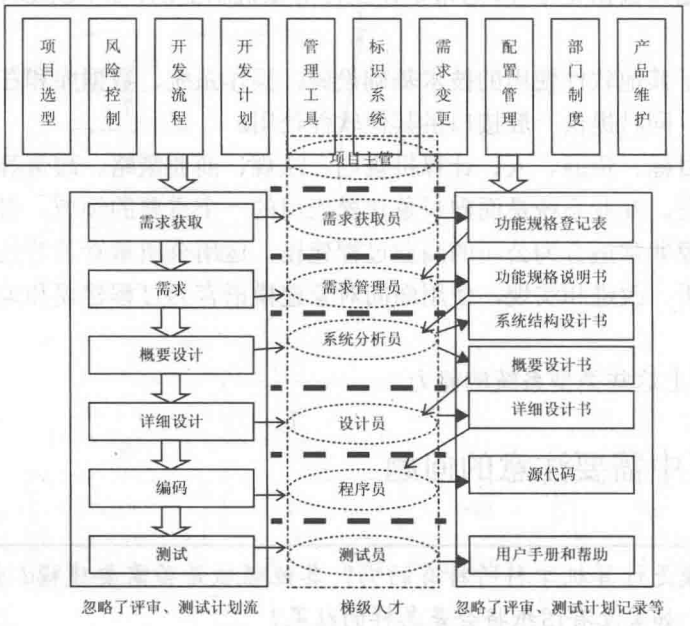


图 1-3　软件工程师发展途径

（3）复杂的大型案例在第三部分高级课题中讲述，大型软件可能是涉及人身财产安全的系统，还有一些大型系统的需求很难说清或抽取，第 12 章用案例详细讲述了这种系统建模过程中的复杂网络需求获取方法和形式化方法技术的补充作用。

第一部分 建模理论概述

本部分介绍面向对象方法的产生、UML 的发展、RUP 统一过程的模型、Rose 工具及 UML 的更多技术细节。

第 2 章　面向对象方法

软件工程从 1968 年产生以来，经历了传统软件工程阶段并发展到现在的高级软件工程阶段，面向对象的方法（简称 00 方法）在传统软件工程方法的基础上产生，因其一些显著特点解决了一些软件问题，是目前主流的软件开发方法。有了面向对象的思想才产生了后面所要介绍的 UML。

请回忆传统软件工程课上完成软件案例作业的过程，完成情况如何？该过程中存在哪些问题？

本章知识要点

（1）面向对象产生的原因。

（2）面向对象的思考方式。

兴趣实践

现实生活中哪些软件基于面向对象方法开发？试举一例。

探索思考

计算机硬件有摩尔定律，为什么软件没有呢？

预习准备

复习传统的结构化软件工程方法的知识及面向对象程序语言相关的三个特性。

2.1　了解面向对象产生的原因

俗话说：天下大势，分久必合，合久必分。计算机软件技术的发展也是一个“分久必合，合久必分”的过程。

从 1946 年 2 月 14 日第一台计算机诞生之日起，软件应运而生。

起初软件是手工作坊的生产方式，没有标准化的过程、工具和技术，从而导致大量软件错误。之后计算机专家提出了各种语言和方法，但还是不能避免错误的发生。小型软件（5000 行代码以下的软件）基本能正确地生产出来，但大中型软件（5 万行代码以上的软件是大型软件，其间的为中型软件）项目就很难保证。到目前为止约 1/3 的软件项目是失败的。

20 世纪 60 年代起随着计算机硬件性能的不断提高和价格的不断下降，其应用领域也在不断扩大。人们在越来越多的领域把更多、更难的问题交给计算机解决。这使得计算机软件的规模和复杂性与日俱增，从而使软件技术不断受到新的挑战。60 年代软件危机的出现就是因为系统的复杂性超出了人们在当时的技术条件下所能解决的程度。此后在软件领域，从学术界到工业界，人们一直在为寻求更先进的软件方法与技

术而奋斗。每当出现一种先进的方法与技术，都会使软件危机得到一定程度的缓和。然而这种进步又立刻促使人们把更多、更复杂的问题交给计算机解决。于是又需要更先进的方法与技术。

1968 年，计算机专家集中讨论了软件危机产生的原因，并达成共识，要用工程化的管理方法来解决软件问题，从而形成软件工程的研究新领域。

开发一个具有一定规模和复杂性的软件系统和编写一个简单的程序大不一样。其间的差别，借用 Booch 的比喻，如同建造一座大厦和搭一个狗窝的差别。大型的、复杂的软件系统的开发是一项工程，必须按工程学的方法组织软件的生产与管理，必须经过分析、设计、实现、测试、维护等一系列软件生命周期阶段。这是人们从软件危机中获得的最重要的收获。这一认识促成传统软件工程学的诞生。编程仍然是重要的，但是更具有决定性意义的是系统建模。只有在分析和设计阶段建立了良好的系统模型，才有可能保证工程的正确实施。正是基于这一原因，许多在编程领域首先出现的新方法和新技术总是很快被拓展到软件生命周期的分析与设计阶段。

开发 Windows 系统需要多少程序员？

在传统的结构化软件工程的指导下，软件生产普遍采用结构化的语言和方法，取得了很大的成就，但大型软件还是频频遇到问题。于是计算机专家又分头行动，想出了各种语言、方法、工具等，以期解决软件问题。

面向对象方法正是在新的技术创意的浪潮中产生的，并经历了这样的发展过程：它首先在编程领域兴起，作为一种崭新的程序设计范型引起世人瞩目。

面向对象方法起源于面向对象的编程语言（简称 OOPL）。20 世纪 50 年代后期，在用 FORTRAN 语言编写大型程序时，常出现变量名在程序不同部分发生冲突的问题。鉴于此，ALGOL 语言的设计者在 ALGOL 60 中采用了以“Begin…End”为标识的程序块，使块内变量名是局部的，以避免它们与程序块外的同名变量相冲突。这是编程语言中首次提供封装保护的尝试。此后程序块结构广泛用于高级语言如 Pascal、Ada、C 之中。

20 世纪 60 年代中后期，Simula 语言在 ALGOL 基础上研制开发，它将 ALGOL 的块结构概念向前发展了一步，提出了对象的概念，并使用了类，也支持类继承。20 世纪 70 年代，Smalltalk 语言诞生，它以 Simula 的类为核心概念，它的很多内容借鉴了 Lisp 语言。由 Xerox 公司经过对 Smalltalk 72、Smalltalk 76 持续不断的研究和改进之后，于 1980 年推出了商品化的应用，它在系统设计中强调对象概念的统一，引入对象、对象类、方法、实例等概念和术语，采用动态联编和单继承机制。

面向对象源出自 Simula，真正的 OOPL 由 Smalltalk 奠基。Smalltalk 现在被认为是最纯的 OOPL。

正是通过 Smalltalk 80 的研制与推广应用，人们注意到面向对象方法所具有的模块化、信息封装与隐蔽、抽象性、继承性、多样性等独特之处，这些优异特性为研制大型软件，提高软件可靠性、可重用性、可扩充性和可维护性提供了有效的手段和途径。分解和模块化可以通过在不同组件设定不同的功能，把一个问题分解成多个小的独立且互相作用的组件来处

理复杂、巨型的软件。

继 Smalltalk 80 之后，20 世纪 80 年代又有一大批面向对象的编程语言问世，标志着面向对象方法走向成熟和实用。此时，面向对象方法开始向系统设计阶段延伸，出现了如 Booch 86、GOOD（通用面向对象的开发）、HOOD（层次式面向对象的设计）、OOSD（面向对象的结构设计）等一批面向对象设计（简称 OOD）方法。但是这些早期的 OOD 方法不是以面向对象分析（OOA）为基础的，而主要是基于结构化分析。到 1989 年之后，面向对象方法的研究重点开始转向软件生命周期的分析阶段，并将 OOA 和 OOD 密切地联系在一起，出现了一大批面向对象的分析与设计（OOA&D）方法，如 Booch 方法、Coad/Yourdon 方法、Firesmith 方法、Jacobson 的 OOSE、Martin/Odell 方法、Rumbaugh 等的 OMT、Shlaer/Mellor 方法等。截至 1994 年，公开发表并具有一定影响力的 OOA&D 方法已达 50 余种。这种繁荣的局面表明面向对象方法已经深入分析与设计领域，并随着面向对象的测试、集成与演化技术的出现而发展为一套贯穿整个软件生命周期的方法体系。目前，大多数较先进的软件开发组织已经从分析、设计到编程、测试阶段全面地采用面向对象方法，使面向对象无可置疑地成为当前软件领域的主流技术。

直到 1995 年，面向对象的 UML 产生标志着面向对象的思想方法得到了人们的一致认可，大型软件的生产才有了更好的解决方式。

2.2 面向对象方法基本概念与特征

用计算机解决问题需要用程序设计语言对问题求解加以描述（编程），实质上，软件是问题求解的一种表述形式。显然，假如软件能直接表现人求解问题的思维路径（求解问题的方法），那么软件不仅容易被人理解，而且易于维护和修改，从而可保证软件的可靠性和可维护性，并能提高公共问题域中的软件模块和模块重用的可靠性。面向对象的概念和机制恰好可以使人们按照通常的思维方式来建立问题域的模型，设计出尽可能自然地表现求解方法的软件。

面向对象方法是一种把面向对象的思想应用于软件开发过程中，指导开发活动的系统方法，是建立在“对象”概念基础上的方法学。对象是由数据和容许的操作组成的封装体，与客观实体有直接对应关系，一个对象类定义了具有相似性质的一组对象。而继承性是对具有层次关系的类的属性和操作进行共享的一种方式。所谓面向对象就是基于对象概念、以对象为中心、以类和继承为构造机制，来认识、理解、刻画客观世界并设计、构建相应的软件系统。

2.2.1 面向对象的概念

对象是要研究的任何事物。对象可以是一个物理实体，如桌子、书等，也可以是特定的实体，如我的桌子，他的 UML 系统建模书；另外，对象也可以是无形的事物，如经济效益、交易等，虽然无形，但仍然可以描述、创建和销毁。对象由数据（描述事物的属性）和作用于数据的操作（体现事物的行为）构成一个独立整体。从程序设计者角度来看，对

像是一个程序模块，从用户角度来看，对象为他们提供所希望的行为。在对象内的操作通常称为方法。

类是对象的模板，即类是对一组有相同数据和相同操作的对象的定义，一个类所包含的方法和数据描述一组对象的共同属性和行为。类是在对象之上的抽象，对象则是类的具体化，是类的实例。类可有其子类，也可有其父类，形成类层次结构。

消息是对象之间进行通信的一种规格说明。它一般由三部分组成：接收消息的对象、消息名及实际变元。

2.2.2　面向对象的特征

封装性：封装是一种信息隐蔽技术，它体现于类的说明，使数据更安全，是对象的重要特性。封装使数据和加工该数据的方法（函数）封装为一个整体，以实现独立性很强的模块，使得用户只能见到对象的外部特性（对象能接收哪些消息，具有哪些处理能力），而对象的内部特性（保存内部状态的私有数据和实现加工能力的算法）对用户是隐蔽的。封装的目的在于把对象的设计者和对象的使用者分开，使用者不必知晓行为实现的细节，只需用设计者提供的消息来访问该对象。通过封装，可以把类作为软件中的基本复用单元，提高其内聚度，降低其耦合度。图 2-1 所示为录音机封装性示例。

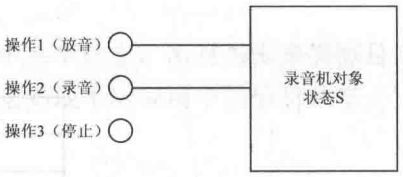


图 2-1　封装的示例

继承性：继承性是子类自动共享父类数据和方法的机制，它由类的派生功能体现。一个类直接继承其父类的全部描述，同时可修改和扩充，其示例见图 2-2。

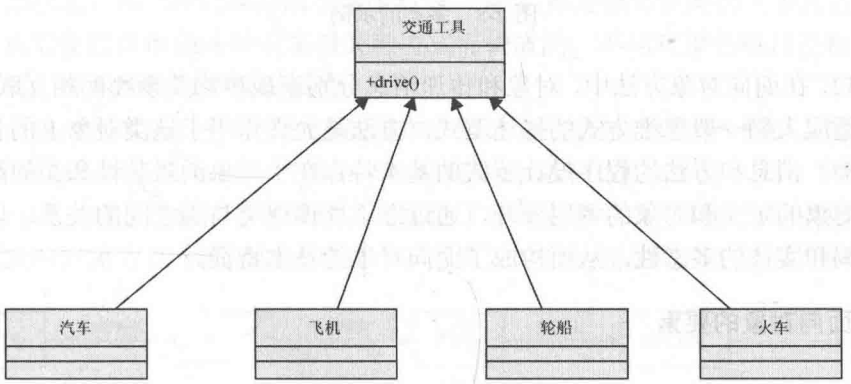


图 2-2　继承的示例

继承具有传递性。如果 B 类继承了 A 类，而 C 类又继承了 B 类，就可以说，C 类在继承了 B 类的同时，也继承了 A 类。C 类中的对象可以实现 A 类中的方法。通常所说的多继承是指一个类在继承其父类的同时，可实现其他类或接口。类的对象是各自封闭的，如果没有继承性机制，则类对象中的数据、方法就会出现大量重复。继承支持系统的可重用性，从而达到减少代码量的作用，而且可以促进系统的可扩充性。