高等院校计算机应用系列教材

信息系统设计与建模

曹德胜　胡荷芬　贾海龙　主　编
狐为民　程　刚　王长利　王博玲　副主编

清華大學出版社
北　京

内 容 简 介

本书详细介绍了信息系统设计及 UML 系统建模的思想和具体方法，内容包括信息系统与面向对象技术概述、UML 概述、Rational Rose 的使用、用例图、类图与对象图、包图、顺序图与协作图、状态图、活动图、组件图、部署图和双向工程，最后以应急救援指挥调度系统和安全培训题库管理系统两个案例详解 UML 各种技术的综合应用。

本书采用理论结合案例的方法进行讲解，理论讲述清晰，技术讲解细致，案例丰富。在讲述 UML 案例时，结合了使用比较广泛的 UML 开发工具 Rational Rose。除了第 13、14 章，每章最后还提供了习题，附录还提供了 5 个课程实验，以供读者更好地了解和掌握 UML 技术。

本书可作为高等院校计算机及相关专业课程的教材，也可作为UML初学者和网站开发人员的参考书。

图书在版编目(CIP)数据

信息系统设计与建模 / 曹德胜，胡荷芬，贾海龙主编. —北京：清华大学出版社, 2023.12
高等院校计算机应用系列教材
ISBN 978-7-302-64795-9

I. ①信… II. ①曹… ②胡… ③贾… III. ①管理信息系统—系统设计—高等学校—教材 ②管理信息系统—系统建模—高等学校—教材 IV. ①TP399

中国国家版本馆 CIP 数据核字(2023)第 204795 号

责任编辑：刘金喜
封面设计：高娟妮
版式设计：孔祥峰
责任校对：成凤进
责任印制：宋　林

出版发行：清华大学出版社
网　　址：https://www.tup.com.cn，https://www.wqxuetang.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-83470000　　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：三河市君旺印务有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**15.5　　**字　　数：**397 千字
版　　次：2023 年 12 月第 1 版　　**印　　次：**2023 年 12 月第 1 次印刷
定　　价：59.80 元

产品编号：099583-01

前　言

UML(unified modeling language，统一建模语言)是当前比较流行的一种建模语言，可用于创建各种类型的项目需求、设计及上线文档。Rational Rose 是目前最受业界瞩目的可视化软件开发工具之一，通过 Rational Rose 能用一种统一的方式设计各种项目的 UML 图。

UML 的设计动机是让开发者用清晰和统一的方式完成项目的前期需求和设计文档，而这些需求和设计文档能够让项目的开发变得更加便捷和清晰。随着 UML 应用的逐渐深入，其获得了广泛的认同，目前已经成为主流项目需求和分析的建模语言。

本书之所以选择 Rational Rose 作为开发 UML 的工具结合信息系统设计，是因为它不仅提供了绘制所有 UML 图的功能，还完全支持“双向工程”，实现代码和模型的相互转换。

本书包含了 UML 的基础知识、基本元素及使用方法，在讲述 UML 的使用过程中结合了 Rational Rose，以便使读者从中感受到利用 Rational Rose 开发 UML 的便捷性和高效性。同时，在讲述 UML 的元素时，结合了大量的实战案例，并且为了提高学习效率，在除第 13、14 章外的各章后面还提供了一定数量的习题。

本书共分为 14 章及一个附录，各章内容遵循从简单到复杂、由浅入深的思路进行组织。由于本书案例基于实际项目，所以能让读者更快地掌握 UML 的基本元素和建模技巧，也能让读者学会通过 Rational Rose 开发 UML 的方法，是 UML 初学者必备的书籍。

1. 本书内容

第 1 章信息系统与面向对象技术概述，介绍了信息系统的基本概念、信息技术在生活中的应用，面向对象的基本概念、面向对象的软件开发，以及用面向对象思想建立软件过程模型的方法。

第 2 章 UML 概述，介绍了 UML 的通用知识，包括 UML 的各种常用元素和 UML 的通用机制。

第 3 章 Rational Rose 的使用，介绍了 Rational Rose 的安装和操作方法及 Rational Rose 的操作技巧。

第 4 章用例图，介绍了用例图的概念和构成要素、用例的重要元素、用例之间的各种重要关系和使用 Rose 创建用例图的步骤。

第 5 章类图与对象图，介绍了类图和对象图的基本概念，使用 Rose 创建类图和对象图的方式，以及案例分析。

第 6 章包图，介绍了包图的基本概念、使用 Rose 创建包图的方式及使用 Rose 在实际项目

中创建包图的具体案例。

第 7 章顺序图与协作图，介绍了顺序图与协作图的基本概念、顺序图与协作图的组成、顺序图与协作图中项目的相关概念、使用 Rose 创建顺序图与协作图的方式，以及使用 Rose 在实际项目中创建顺序图与协作图的具体案例。

第 8 章状态图，介绍了状态图的基本概念、构成状态图的元素、状态的组成，使用 Rose 创建状态图的方式，以及使用 Rose 在实际项目中创建状态图的具体案例。

第 9 章活动图，介绍了活动图的基本概念、活动图的组成，使用 Rose 创建活动图的方式，以及使用 Rose 在实际项目中创建活动图的具体案例。

第 10 章组件图，介绍了组件图的基本概念，使用 Rose 创建组件图的方式，以及使用 Rose 在实际项目中创建组件图的具体案例。

第 11 章部署图，介绍了部署图的基本概念，使用 Rose 创建部署图的方式，以及使用 Rose 在实际项目中创建部署图的具体案例。

第 12 章双向工程，介绍了正向工程和逆向工程的概念，以及使用 Rose 并以 Java 语言为例，介绍如何从图形生成代码和如何从代码生成图形的具体案例。

第 13 章和第 14 章从需求分析讲起，分别通过应急救援指挥调度系统、安全培训题库管理系统，介绍了创建系统用例图模型、静态模型、动态模型和部署模型的方式。

附录一共提供了 5 个完整的课程实验，可作为课程结束时的课程设计使用，有助于学生从整体上把握系统建模的技术和方法，方便教师课堂教学。

2. 本书特点

(1) 从入门到精通。本书遵循由浅入深、循序渐进的方式，按照知识点的梯度逐渐深入，这样编写的目的是让读者能快速地学习和掌握 UML 技术。

(2) 基于实战案例教学。本书的 UML 相关知识点都配套了实际的案例，能让读者了解现实项目中 UML 的具体应用。

(3) 用 Rational Rose 实现。目前有很多种 UML 的开发工具，但 Rational Rose 在业内使用比较广泛。通过学习本书，读者能了解 Rational Rose 的常规用法。

(4) 习题配套。为了让读者快速掌握 UML 技术，除第 13、14 章外，每章后面都提供了相关的习题。

3. 学时安排

本课程总学时为 96 学时，各章学时分配见表 1(供参考)。

表 1　学时分配建议表

课 程 内 容	学 时 数		
	合　计	讲　授	实　验
第 1 章　信息系统与面向对象技术概述	1	1	
第 2 章　UML 概述	2	2	
第 3 章　Rational Rose 的使用	3	2	1

(续表)

课 程 内 容	学 时 数		
	合　计	讲　授	实　验
第 4 章　用例图	5	3	2
第 5 章　类图与对象图	6	4	2
第 6 章　包图	1	1	
第 7 章　顺序图与协作图	6	4	2
第 8 章　状态图	4	2	2
第 9 章　活动图	4	2	2
第 10 章　组件图	5	3	2
第 11 章　部署图	1	1	
第 12 章　双向工程	2	2	
第 13 章　应急救援指挥调度系统	1	1	
第 14 章　安全培训题库管理系统	1	1	
附录　课程实验	6	2	4
合计	48	31	17

本书可作为高等院校计算机及相关专业的 UML 课程教材，也可作为自学者及网站开发人员的参考书。

本书免费提供 PPT 教学课件、案例源文件和习题答案，可通过扫描下方二维码获取。

教学资源下载

本书由曹德胜、胡荷芬、贾海龙任主编，狐为民、程刚、王长利、王博玲任副主编。参与本书编写工作的还有周钰婷、李惟等，在此，编者对他们表示衷心的感谢。

在本书的编写过程中，借鉴了许多现行教材编写的宝贵经验，在此，谨向这些作者表示诚挚的感谢。

由于时间仓促，加之编者水平有限，书中难免有不足之处，敬请广大读者批评指正。

服务邮箱：476371891@qq.com。

编　者

2023 年 5 月

目　录

∞ 第1章 ∞

信息系统与面向对象技术概述

随着全球化带来的经济、文化和技术变革，信息系统已经渗透到我们的生活中，拥有操作计算机的技术与能力，也成为越来越多工作岗位的就职条件之一。在信息化时代，管理人员使用企业资源计划系统来管理业务运营，医生使用医疗信息系统分析患者数据、诊断病情，农民使用地理信息系统来减少肥料、优化植物产量。总而言之，信息系统已经成为人们生活和工作的重要组成部分。

现在，面向对象技术已经逐渐取代了传统的技术，成为当今计算机软件工程学中的主要开发技术。面向对象技术能够使计算机以更符合人的思维方式去解决一系列的编程问题，极大地提高了程序代码复用程度和可扩展性，大幅度提高了编程效率，并减少了软件维护的代价。面向对象技术发展的重大成果之一就是出现了统一建模语言(unified modeling language，UML)。UML是面向对象技术领域内占主导地位的标准建模语言，它统一了过去相互独立的数十种面向对象的建模语言共同存在的局面，通过统一语义和符号表示，系统地对软件工程进行描述和构造，形成了一个统一的、公共的、具有广泛适用性的建模语言。

1.1 信息系统的基本概念

信息系统领域是巨大的、多样化的、不断发展的，有大量的系统分析师、系统程序员等技术人员对信息系统进行构建、管理、使用和研究，他们所用的信息技术包括硬件、软件与通信网络。因而，我们将信息系统定义为创建、收集、处理、存储和传播有用数据的人员和信息技术的结合。如今，信息系统已逐渐成为我们学习、工作和生活中最基础、最重要的部分之一。

1.1.1 数据、信息与知识

在了解信息系统之前，必须区分原始的数据、信息和知识。

1. 数据

数据是原始符号、未经加工的事实及信息系统的根源，如字符、数字、图像、声音或它们的组合等都是数据。数据只是一个描述，其本身没有任何特定的背景和意义，在处理之前几乎没有价值。例如，1001 没有特定含义，可以表示日期、门牌号码、长度等。虽然数据没有固定的含义，但如果输入错误数据，输出得到的也会是错误数据。因此，评估数据对决策是否可靠的关键因素就是数据质量，包括数据的完整性、准确性、及时性、有效性和一致性。

2. 信息

信息(information)是经过格式化、组织化或处理加工而变得有用的数据，也是提供决策的有效数据，对特定用户有价值，并能传达意义。信息可以被定义为现实的代表，可以说明何时、何地、何人(或物)、何事。例如，没有经过处理的银行每天存取款的数据是无用的，但如果对数据进行统计分析，进行客户画像后展开精准营销，这些无用的数据就会变成对银行有很大帮助的有用信息。

除此之外，信息还有一个非常重要的特性——时效性。信息的时效性是指从信息源发送信息后经过接收、加工、传递、利用的时间间隔及其效率。时间间隔越短，使用信息越及时，信息的使用程度越高，时效性越强。我们在获取信息后，需要判断其时效性来确定该信息的价值，例如“今天下午三点到中央广场”，这种信息会在时效性消失后变得没有价值。

3. 知识

知识(knowledge)是使用信息、理解信息后形成的观点、认识或理解，是客观世界规律的总结。知识对每个人来说都是独一无二的，是过去经验和洞察力的积累让我们拥有解释信息并赋予信息意义的能力，与人们对事物的了解程度、谈论问题的角度、人的价值观念和所处环境等都有关系。对于地球是方还是圆的讨论，就是一个很好的例子。

知识具有：①隐性特征，具备较强的隐蔽性，需要进行归纳、总结与提炼；②行动导向特征，能够直接推动人的决策和行为，加速行动过程。若要使知识转化为行动，个人必须有权利和能力执行决定，用知识和权威来生成可产生影响的可操作信息。

4. 数据、信息与知识的比较

数据、信息与知识常被人们混淆，难以辨别。数据是记录信息的一种形式，信息由数据不断解析和加工而产生。例如，电子温度计上的数据 39.5℃，仅是对温度进行描述的数据符号，只是在人观察温度计上的数据判断出体温过高后，39.5℃这个数据才成为了信息。数据只有对实体行为产生影响时才成为信息。

而知识来源于信息，是基于某种角度的信息整合而形成的观点。由于信息的时效性，它的价值往往会在时间效用失效后开始衰减，只有通过人们参与对信息进行归纳、演绎、比较等手段提取有价值的部分信息，并与已经存在的人类知识体系相结合，这部分信息才会转变成知识。如上例，39.5℃是数据，通过“正常人类的体温在 36.0～37.3℃”是知识，以此判断出“体温过高”的信息。综上，同时拥有高质量的数据、充足的信息和相关的知识，才能做出高质量的决策，即决策依赖数据、信息和知识。

数据、信息与知识的关系如图 1-1 所示。

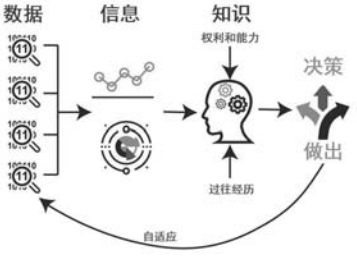

图 1-1　数据、信息与知识的关系

1.1.2　系统

系统(system)是内部相互依赖的各部分按照某种规则，为实现某一特定目标而联系在一起的合理的、有序的组合。

一般来说，系统具有以下 4 个特性。

(1) 整体性。系统内部的各部分是为了实现某一特定目标而联系在一起的，因此每部分都要服从整体，追求整体最优，而不是局部最优。即使一个系统中的各部分都不是最完善的，但通过综合与协调要使整个系统达到最好的效果。

(2) 层次性。复杂的系统总是由若干较为简单的子系统组成，这些较为简单的子系统又由更简单的子系统组成。对于一个复杂的系统可以采用分解的方法，利用系统的层次性由高到低、由表及里、由粗到细地进行分析。

(3) 关联性。系统的各部分在功能上相对独立，但彼此之间是相互联系、相互制约的。这些联系与制约决定了整个系统的运行机制，分析这些联系是构建一个系统的基础，实现一个系统的过程中不仅要考虑如何将系统分成若干子系统，而且要考虑这些子系统之间的相互制约关系。

(4) 目的性。任何一个系统都是为了完成某一特定目标而构造的，如学校的目标是培养经济建设人才、取得先进的科研成果，工厂的目标是生产出高质量、高销量的产品。因此，在建设系统之初就需要按照这个明确的目标设计系统的每一部分。

图 1-2 所示是系统的一般模型，系统边界是系统与环境分开的虚拟边界，在此边界实现物质、能量、信息的交换，边界之内是系统，边界之外是环境，环境和系统互相有一定的影响。反馈是系统设计中重要的策略之一，将观测到的输出取作反馈量以构成反馈律，形成对系统的闭环控制，以达到期望的对系统的性能指标要求。

系统无处不在，如大学就是一个常见的人造系统，学生、教师、管理人员、教材、设置等为系统输入，教学、研究、服务等为系统处理机制，受教育的学生、有意义的研究和提供服务为系统输出，知识与劳动成果的获得为输出反馈。

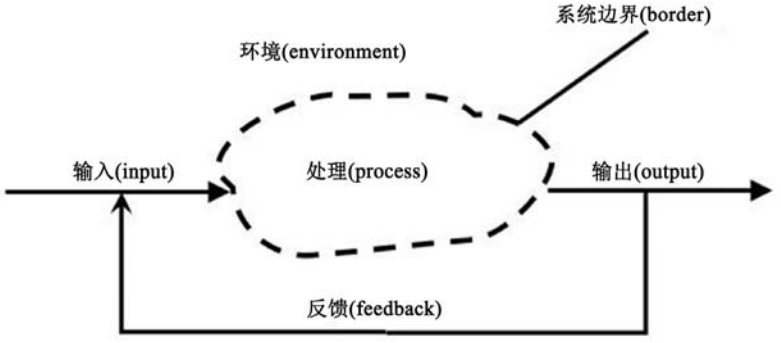

图 1-2　系统的一般模型

1.1.3 信息系统

人类有了生产活动时就有了简单的信息系统，即采用手工管理方式。但随着科学技术的发展与社会活动的复杂化，手工处理方式显然已经远远不能满足人类生产活动的需要，为了提高用于记录、查找和加工信息的效率，计算机已经成为信息处理的有力工具，目前人们所说的信息系统已经不是以往传统的手工管理信息系统，而是计算机化的信息系统。

于此给出信息系统的概念，信息系统是创建、收集、处理、存储和传播有用数据的人员和信息技术的结合，目的是实现组织中各项活动的管理、调节和控制。本质上，信息系统就是输入数据或信息，通过加工处理产生信息的系统，其运转情况与整个组织的效率密切相关。

信息系统自 20 世纪 50 年代以来经历了由单机到网络，由低级到高级，由电子数据处理系统到管理信息系统再到决策支持系统，由数据处理到智能处理的过程。这个发展过程大致经历了以下几个阶段。

(1) 20 世纪 50 年代中期，电子数据处理系统(electronic data processing system，EDPS)使用计算机代替以往人工进行事务性数据处理的系统，因此也被称为事务处理系统(transaction processing system，TPS)。但受限于当时计算机的能力与人们对计算机的认知，EDPS 完全模拟人工系统，数据收集速度慢且容易出错，是该系统最薄弱的环节。

(2) 20 世纪 70 年代，随着数据库技术、网络技术和科学管理方法的发展，人们开始不满足于仅用计算机来模拟简单的数据处理，便诞生了管理信息系统(management information system，MIS)。MIS 最大的特点是高度集中，它将组织中的数据集中起来，在中心数据库和计算机网络系统的基础上分布式处理数据。MIS 利用定量化的科学管理方法，通过预测、计划优化、管理、调节和控制等手段来支持决策。

但传统的 MIS 缺乏对企业组织机构和不同阶层管理人员决策行为的深入研究，忽视了人在管理决策过程中不可替代的作用，在辅助企业高层的管理决策工作中常显得力不从心，而决策支持系统(decision support system，DSS)能较好地解决这一问题。相较于早期的 MIS，DSS 强调决策过程中人的主导作用，用户可以针对企业决策的问题，建立一个模型以考察一些变量的变化对决策结果的影响，在人和计算机交互的过程中帮助决策者探索可能的方案，为管理者提供决策所需要的信息，如用户可以观察利率变化对一个新建制造厂的投资的影响。

EDPS、MIS 和 DSS 代表了信息系统发展过程中的某一阶段，至今仍各自不断地发展着，并且是相互交叉的关系。随着网络技术、人工智能技术的发展，信息系统向网络化、智能化发展，此外，还出现主管信息系统(executive information system，EIS)、战略信息系统(strategic information system，SIS)、办公自动化系统(office automation system，OAS)等新概念。各类信息系统的开发应用，改善了人们的工作环境，提高了工作效率、扩大了人们思考问题的范围并增强了控制问题的能力。如果信息系统的构思、设计、使用和管理是有效且有战略性的，那么伴随着健全的商业模式，信息系统会大幅度提高公司的效率，拓展公司业务层面，并获得或保持竞争优势。

1.2　面向对象技术

信息系统在各行各业普遍应用的同时，软件工程和信息系统在自身开发技术方面也取得了一系列的成果，最早人们是从结构化开发方法开始从事信息化开发的，随后在计算机软件开发领域产生了软件工程，在管理信息系统开发领域产生了商务系统规划方法。20 世纪 80 年代后，由于第四代程序生成语言的出现，产生了原型方法及近来比较热门的面向对象(object-oriented)开发方法。

信息系统开发的成功与否与使用的开发工具和方法有直接联系。面向对象程序开发中的对象由数据和对数据执行的活动组成。因此，程序中的对象之间有相互作用的机会。例如，一个对象可能由员工信息和对员工执行的所有操作(如工资、福利等)方面的数据组成。

面向对象技术和过去的软件开发技术完全不同，是一种全新的软件开发技术。面向对象的概念从问世到现在，已经发展为一种非常成熟的编程思想，并且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)旨在创建软件重用代码，具备更好的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最佳选择。它通过给程序中加入扩展语句，把函数“封装”到编程所必需的“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。说它是一场革命，不是对对象本身而言，而是对它处理工作的能力而言。

1.2.1　面向对象技术的含义

面向对象方法的核心是面向对象技术，这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。从程序设计方法上来讲，面向对象设计是一种自下而上的程序设计方法，它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。

面向对象的程序设计方法能够使程序的结构清晰简单，大大提高了代码的重用性，有效减少了程序的维护量，提高了软件的开发效率。面向对象程序设计由类的定义和类的使用两部分组成，在程序中定义数个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。

面向对象开发过程中，系统由一系列计算对象组成，每个对象都封装了其自身的数据和逻辑程序，开发者通过定义一个类来定义程序逻辑的结构和数据字段，只有当程序开始执行时，对象才能存在，这个过程我们称为类的实例化。例如，面向对象方法不再把世界看成一个紧密关联的系统，而是看成一些相互独立的小组件，这些组件依据某种规则组织起来，完成特定功能。

1.2.2　对象、类、属性和操作

1. 对象

对象(object)是面向对象系统的基本构造块，是理解面向对象技术的关键。万物皆是对象，对象可以是有形的实体，如汽车、人、房子等，也可以是抽象的规则、计划或事件，如图书编号、学生学号、职员所属部门等。

2. 类

类(class)是对一组有相同属性和相同操作的对象的组合，而对象是类实例化的结果。也就是说，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。对象的特征称为属性，属性是对象从类中继承的或是自己拥有的属性。如图 1-3 所示，称为 Person 的类包括 Teacher 和 Student。Teacher 类和 Student 类都继承了 Person 类的 Name(姓名)、Gender(性别)属性，同时，Student 类也有不被 Person 类中的其他成员所共享的系部属性。面向对象设计将数据和对数据的操作封装到一起，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，最终将其抽象成一种新的数据类型——类。

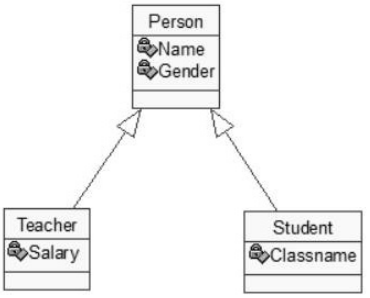

图 1-3　Person、Teacher 与 Student 类

类与类之间的联系及类的重用出现了类的继承、多态等特性。类的集成度越高，越适合大型应用程序的开发。类的思想使用有一个引用点，并根据具体对象与本类的相似性或区别对它进行描述，因此在面向对象开发过程中每次需要一个对象时，不必从头开始描述该对象，适当地继承其他类会大大减少工作量。

3. 属性

属性(attribute)是已命名的类的特性，它描述了该特性的实例可以取值的范围。类可以有任意数目的属性，也可以没有属性。属性描述了正被建模的事物的一些特性，这些特性为类的所有对象共有。如图 1-4 所示，每个顾客都有姓名、地址、手机号码和出生日期。因此，属性是对类的对象可能包含的数据种类或状态种类的抽象，在一个给定时间，类的一个对象将具有该类的每个属性的特定值。

图 1-4　属性

4. 操作

操作(operate)，也称方法，它定义了对象可以执行的任务，是一个服务的实现，该服务可以由任何类的对象来请求以影响其行为。也就是说，操作是能对一个对象所做的事情的抽象，并且它由这个类的所有对象共享。类可以有任意数目的操作，也可以没有操作。

例如“门”对象可以同假定能够在其上执行的行为相关联，如门可以打开、关闭、上锁、

开锁。这些行为都与“门”对象相关，并由该对象实现。调用对象的操作经常会改变对象的数据或状态。

通常，若操作名由一个单词组成，则小写；若操作名由多个单词组成，则第一个单词小写，第一个单词后面的每个单词的首字母大写，如 move 或 isEmpty，如图 1-5 所示。

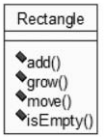

图 1-5　操作

可以通过阐明操作的特征标记来详细描述操作，特征标记包含所有参数的名称、类型和默认值，如果是函数，还包括返回值类型，如图 1-6 所示。

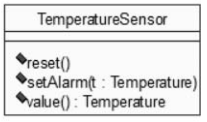

图 1-6　含有特征标记的操作

1.2.3　消息与事件

消息是对象间的通信，它传递了要执行动作的信息，并能触发事件。接收到的一个消息通常被认为是一个事件。一个消息主要由消息的发送对象、消息的接收对象、消息的传递方式、消息的内容(参数)、消息的返回五部分组成。传入消息内容的目的有两个：一个是让接收请求的对象获取执行任务的相关信息；另一个是行为指令。

事件通常是指一种由系统预先定义而用户或系统发出的动作。事件作用于对象，对象识别事件并做出相应的响应。一个事件具有原子性(不可中断)且在概念上无持续，如“已插卡”“已输入密码”“钱已取出”等都是事件。事件会相互依赖，例如在一个给定的事件序列中，“已输入密码”总是在事件“钱已取出”之前发生。现代高级语言中可以通过一些其他技术在类中加入事件，如我们通常所熟悉的一些事件：Click，单击对象时发生的事件；Load，当界面被加载到内存中时发生的事件等。

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该对象执行某个方法时，就向其发送一个消息，该对象响应请求，并完成指定的操作。程序的执行取决于事件发生的顺序，由顺序产生的消息来驱动程序的执行，而不必预先确定消息产生的顺序。

1.2.4　封装、继承和多态

封装、继承、多态是面向对象程序的三大特征。随着信息技术的发展，这三大特征已被应用于硬件、数据库、人工智能技术、分布式计算、网络、操作系统等领域。三大特征相互配合，使得面向对象程序具有更好的组织结构和可维护性：封装保护了对象的内部细节，继承提供了代码的重用性，多态使代码可以在不同的对象和场景中重用。通过合理应用三大特征，可以设计出更加灵活可靠的面向对象程序。

1. 封装

封装(encapsulation)就是把对象的状态和行为绑到一起的机制，使对象形成一个独立的整体，并且尽可能地隐藏对象的内部细节。封装有两个含义：一是把对象的全部状态和行为结合在一起，形成一个不可分割的整体。对象的私有属性只能由对象的行为来修改和读取。二是尽可能隐蔽对象的内部细节，与外界的联系只能通过外部接口来实现。

从对象外部来看，良好的封装能隐藏大量细节。隐藏是使用封装将某些信息或实现方法限制在封装结构内部，以约束外部的可见性。隐藏有两种形式：信息隐藏和实现隐藏。信息隐藏就是使封装单元内的信息不被外界察觉，而实现隐藏是指外界不能察觉单元内的实现细节。例如，使用手机时，我们关注的通常是这部手机能实现什么功能，而不太会去关心这部手机是怎么被制造出来的。

但是在实际项目中，如果一味地强调封装，对象的任何属性都不允许外部直接读取，反而会增加许多无意义的操作，为编程增加负担。为避免这一点，在语言的具体使用过程中，应该根据需要和具体情况，来决定对象属性的可见性。例如，房子就是一个类的实例，室内的装饰和摆设只能被室内的居住者欣赏和使用，如果没有四面墙的遮挡，室内的所有活动在外人面前将一览无遗。由于有了封装，房屋内的所有摆设都可以被随意改变且不影响他人，然而，如果没有门窗，即使它的空间再宽阔，也没有实用价值。房屋的门窗，就是封装对象暴露在外的属性和方法，专供人进出，以及空气流通和带来阳光。

2. 继承

继承(inheritance)是利用可重用软件构造系统的有效机制，它可以使开发人员在已有类的基础上定义和实现新的类。继承意味着自动地拥有或隐含地复制，它是一种连接类与类之间的层次模型，是指特殊类的对象拥有其一般类的属性和行为。例如，交通工具可以分为车、船、飞机等，我们通过抽象的方式实现一个交通工具类以后，可以通过继承的方式分别实现车、船、飞机等类，并且这些类包含交通工具的特性。交通工具类继承结构示例如图 1-7 所示。

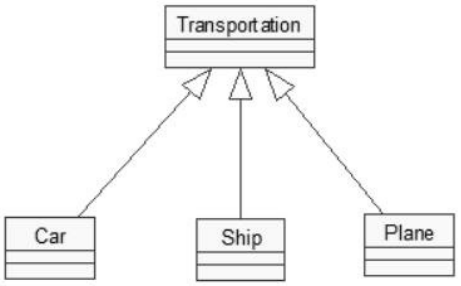

图 1-7　交通工具类继承结构示例

继承性表示较一般的类与较特殊的类之间的关系，在此关系中，一般的类通常被称为“超类”“基类”“父类”等，而较特殊的类被称为“子类”“衍生类”“派生类”等，UML 中将这种关系称为“泛化”或“一般化”。

继承是传递的，当一个特殊类被它更下层的特殊类继承时，它继承来的及自定义的属性和行为又被下一层的特殊类继承下去。通过继承机制，子类可以自动拥有(隐含复制)父类的全部

属性与操作，继承机制简化了对现实世界的认识和描述，在定义子类时，不必重复定义那些已在父类中定义过的属性和操作，只要声明自己是某个父类的子类，把精力集中在定义子类所特有的属性和操作上即可，所以继承机制提高了软件的可复用性。

3. 多态

多态(polymorphism)是指两个或多个属于不同类的对象对于同一个消息或方法调用所做出不同响应的能力。在面向对象技术中，多态性是指在父类中定义的属性和操作被其子类继承后，可以具有不同的数据类型或表现出不同的行为。例如，我们在“动物”基类中定义了“进食”行为，派生类“猫”和“狗”都继承了动物类的进食行为，但其进食的事物却不一定相同，猫喜欢吃鱼，而狗喜欢啃骨头。该进食的消息发出以后，猫类和狗类的对象接收到消息后各自执行不同的进食行为。图 1-8、图 1-9 所示为多态性示例。

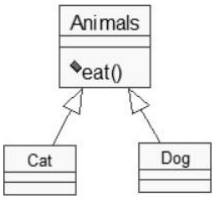

图 1-8　多态性示例 1

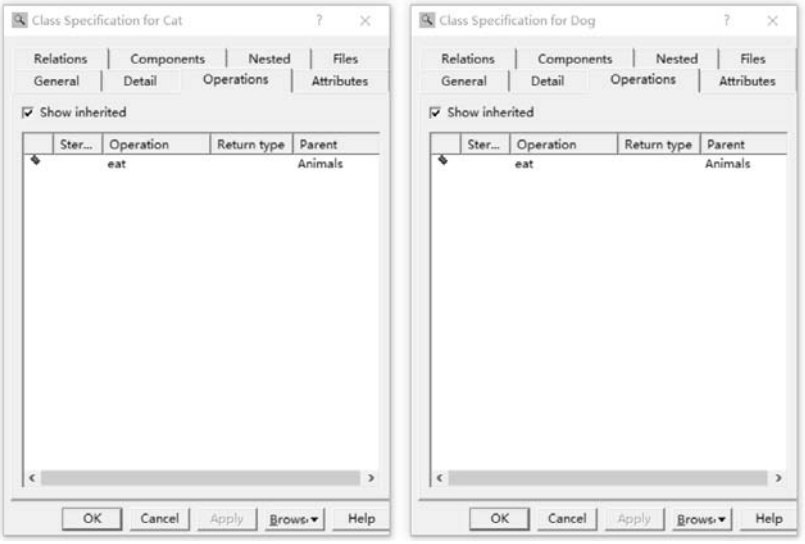

图 1-9　多态性示例 2

在一般与特殊关系的类层次结构中，利用多态性可以使不同层次的类共享一个操作名称，而各自有不同的实现，当一个对象接收到一个请求进行某项服务的消息时，将根据该对象所属的类，动态地选用在该类中定义的操作。子类继承父类的属性或操作名称，而根据子类对象的特性修改属性的数据类型或操作的内容，这称为重载(overloading)，它是实现多态性的重要方法之一。

继承性和多态性的结合可以生成一系列虽然类似但独一无二的对象：由于继承性，这些对象共享许多相似的特征；由于多态性，针对相同的消息，不同的对象可以有独特的表现方式，实现个性化的设计。

1.3 面向对象的软件开发

软件开发是一项系统工程，它的真正决定性因素来自前期对所解决问题的分析、抽象和概念问题的提出，而非后期的程序源代码的实现，只有正确识别并深刻理解了目标问题的内在逻辑和本质特征，才可能圆满地解决问题，设计出优秀的软件。综上，在软件开发与设计的全过程中，程序代码设计只是相对较小的一项工作。

面向对象的程序设计方法是一种新型、实用的程序设计方法。该方法的主要特征在于支持数据抽象、封装和继承等概念。借助数据抽象和封装，可以抽象地定义对象的外部行为而隐蔽其实现细节，从而达到规约和实现的分离，有利于程序的理解、修改和维护，对系统原型速成和有效实现大有帮助；支持继承则可以在原有代码的基础上构造新的软件模块，从而有利于软件的复用。在采用面向对象的程序设计方法开发系统时，系统实际上是由许多对象构成的集合。

1.3.1 面向对象分析

面向对象分析的目的是认知客观世界的系统并对系统进行建模，主要过程为对系统进行评估，采集和分析系统的需求，理解系统要解决的问题，重点是充分考虑系统的实用性。这一阶段的结果可以建立用例模型，描述系统需求。在开发过程中，随着对系统的认识不断加深，用例模型可以自顶向下不断精化，演化出更为详细的用例模型。

在这一过程中，抽象是最本质、最重要的方法，针对不同问题性质选择不同的抽象层次，过简或过繁都会影响对问题本质属性的了解和解决。

面向对象的分析方法是在一个系统的开发过程中进行了系统业务调查后，按照面向对象的思想来分析问题。面向对象分析与结构化分析有较大的区别，面向对象分析所强调的是在系统调查资料的基础上，针对面向对象方法所需要的素材进行的归类分析和整理，而不是对管理业务现状和方法的分析。

面向对象分析一个事物时的基本步骤如下。

1) 确定对象和类

这里所说的对象是对数据及其处理方式的抽象，它反映了系统保存和处理现实世界中某些事物的信息的能力。类是对多个对象的共同属性和方法集合的描述，包括如何在一个类中建立一个新对象的描述。

2) 确定结构

结构是指问题域的复杂性和连接关系。例如，类成员结构反映了泛化与特化的关系，整体和部分结构反映了整体和局部之间的关系。

3) 确定主题

主题是指事物的总体概貌和总体分析模型。

4) 确定属性

属性就是数据元素，可用来描述对象或分类结构的实例，在对象的存储中指定。

5) 确定方法

方法是在收到消息后必须进行的一些处理操作。对于每个对象和结构来说，那些用来增加、修改、删除和选择一个方法本身的操作都是隐含的，而有些操作则是显示的。

1.3.2　面向对象设计

1. 面向对象设计的准则

面向对象设计的准则包括模块化、抽象、信息隐藏、低耦合和高内聚等。

1) 模块化

面向对象开发方法很自然地支持了把系统分解成模块的设计原则，即对象就是模块。它是把数据结构和操作这些数据的方法紧密地结合在一起所构成的模块。类的设计要很好地支持模块化这一准则，这样能使系统有更好的维护性。

2) 抽象

面向对象方法不仅支持对过程进行抽象，而且支持对数据进行抽象。抽象方法的好坏及抽象的层次都对系统的设计有很大影响。

3) 信息隐藏

在面向对象方法中，信息隐藏是通过对象的封装性来实现的。对象暴露接口的多少及接口的好坏都对系统的设计有很大影响。

4) 低耦合

在面向对象方法中，对象是最基本的模块，因此耦合主要是指不同对象之间相互关联的紧密程度。低耦合是设计的一个重要标准，因为这有助于使系统中某一部分的变化对其他部分的影响降到最低限度。低耦合的程序有助于类的维护，也是衡量类质量的一个很重要的指标。

5) 高内聚

在面向对象方法中，高内聚也是必须满足的条件。高内聚是指在一个对象类中应尽量多地汇集逻辑上相关的计算资源。如果一个模块只负责一件事情，则说明这个模块有很高的内聚度；如果一个模块负责了很多相关的事情，则说明这个模块的内聚度很低。内聚度高的模块通常容易理解，很容易被复用、扩展和维护。较低的耦合度和较高的内聚度，也即我们常说的“低耦合、高内聚”，是所有优秀软件的共同特征。

2. 系统设计

系统设计阶段主要确定系统的高层次结构，包括子系统的分解、系统的固有并发性、子系统如何分配给硬软件、数据存储管理、资源协调、软件控制实现、定义人机交互接口等，需要做出如下决策。

1) 将系统划分为子系统

对于每个子系统，都必须建立该子系统与其他子系统之间的定义良好的接口，接口的建立使不同子系统的设计可以独立进行。如果必要，还可以不断地将子系统进一步分解为更小的子系统，直到将子系统分解为模块。

2) 识别并发

若要识别出系统固有的并发，可以通过分析状态图来完成这个任务。为了定义并发任务，需要检查系统中不同的、可能的控制线程，并将这几个控制线程合并为一个。

3) 将子系统和任务分配给处理器

将子系统分配给处理器是从估计所需要的硬件资源开始的，同时设计者还必须决策哪个子

系统由硬件实现，哪个子系统由软件实现。

4) 选择实现数据存储的策略

在系统设计阶段，必须完成关于数据库的决策，即决定是使用文件还是数据库管理系统存储数据。

5) 识别出全局资源，并确定控制访问全局资源的机制

必须明确定义对全局资源(如物理单元、逻辑名和共享数据等)的使用和访问。

6) 选择实现软件控制的方法

用软件实现控制又分为外部控制和内部控制两种。外部控制(external control)是系统中对象间的外部可见的事件流。处理外部控制有 3 种方式：过程驱动系统、事件驱动系统和并发系统。内部控制(internal control)是进程内的控制流，可以被看作程序语言中的过程调用(所提到的系统并不是唯一的选择，还可以采用基于规则的系统或其他非过程的系统)。

7) 考虑边界条件

描述各种边界条件也是很重要的，包括系统的初始化、系统的结束和系统的失败。

8) 建立折中的优先级

系统的所有目标并不是都可以达到，所以要分析系统的所有目标，然后进行折中，为不同的目标设置不同的优先级。

1.4 软件过程模型

面向对象的建模以面向对象开发者的观点创建所需要的系统。事实上，选择创建什么样的模型，对如何解决问题和形成解决方案有深远的影响，可跨越整个生存期的系统开发、运作和维护所实施的全部过程、活动和任务的结构框架。

软件过程模型能清晰、直观地表达软件开发的全过程，明确规定了要完成的主要活动和任务，用来作为软件项目开发工作的基础。对于不同的软件系统，可采用不同的开发方法，使用不同的程序设计语言和不同技能的人员，以及不同的管理方法和手段等，它还允许采用不同的软件工具和不同的软件工程环境。

常见的软件过程模型有瀑布模型、喷泉模型、基于构件的开发模型等。

1.4.1 瀑布模型

瀑布模型，也被称为生存周期模型，是历史上第一个正式使用并得到业界广泛认可的软件开发模型，其核心思想是按照相应的工序将问题进行简化，将系统功能的实现与系统的设计工作分开，便于项目之间的分工与协作，即采用结构化的分析与设计方法将逻辑实现与物理实现分开。瀑布模型将软件的生命周期划分为系统工程、需求分析与规约、设计与规约、编码与单元测试、集成测试与系统测试及运行与维护 6 个阶段，并且规定了它们自上而下的次序，如同瀑布一样下落，每个阶段都是依次衔接的。采用瀑布模型的软件开发过程如图 1-10 所示。

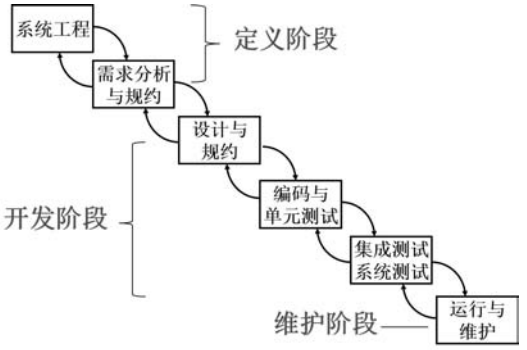

图 1-10　采用瀑布模型的软件开发过程

瀑布模型是最早出现的软件过程模型，在软件工程中占有重要的地位，它提供了软件开发的基本框架。瀑布模型的本质是一次通过，即每个活动只做一次，最后得到软件产品，因此也称作“线性顺序模型”或“传统生命周期”。

瀑布模型的主要优点是：整个开发过程阶段和步骤明确，每一阶段均有明确的成果，这些成果以可行性分析报告、系统说明书、系统设计说明书等形式表现出来，并作为下一阶段工作的依据。整个项目按阶段和步骤可以划分为许多组成部分，各部分可以独立地开展工作，这有利于整个项目的管理与控制。

然而软件开发的实践表明，瀑布模型也存在一定的缺点：由于开发模型呈线性，所以当开发成果尚未经过测试时，用户无法看到软件的效果，使软件与用户见面的时间间隔较长，增加了一定的风险；在软件开发前期未发现的错误传递到后面的开发活动中时，可能会扩散，进而可能会造成整个软件项目开发失败；在软件需求分析阶段，要完全确定用户的所有需求是比较困难的，甚至可以说是不太可能的。

总之，瀑布模型代表了一种直观的、切合实际的、通用的工程方法在软件工程中的应用，它强调在执行活动之前先进行设计的重要性，并因此提供了一种有价值的平衡，平衡软件开发中对总体体系结构或程序结构不做任何考虑就开始编码的普遍倾向。但实际上，瀑布模型的应用并没有真正做到它所承诺的那样，提供一种组织软件项目的方法，使得项目将会是可控制的，按时产生在预算内交付功能正确的系统。造成这种失败的两个主要原因是，该模型管理项目中涉及的风险的方式和模型中对系统需求的处理。

1.4.2　喷泉模型

喷泉模型是典型的面向对象开发模型，着重强调不同阶段之间的重叠，认为面向对象软件开发过程不需要或不应该严格区分不同的开发阶段。喷泉模型是一种以用户需求为动力、以对象作为驱动的模型，类似一个喷泉，水喷上去又可以落下来。各个开发阶段没有特定的次序要求，可以交互进行，并且可以在某个开发阶段中随时补充其他任何开发阶段中的遗漏。喷泉模型克服了瀑布模型不支持软件重用和多项开发活动集成的局限性，具有迭代性和无间隙性。采用喷泉模型的软件开发过程如图 1-11 所示。

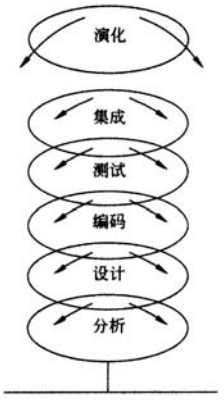

图 1-11　采用喷泉模型的软件开发过程

喷泉模型不像瀑布模型需要分析活动结束后才开始设计活动，设计活动结束后才开始编码活动，该模型的各个阶段没有明显的界限，开发人员可以同步进行开发。

喷泉模型的优点是：可以提高软件项目的开发效率，节省开发时间，适用于面向对象的软件开发过程。

喷泉模型的缺点是：由于喷泉模型在各个开发阶段是重叠的，因此在开发过程中需要大量的开发人员，不利于项目的管理。此外该模型要求严格管理文档，使得审核的难度加大，尤其是面对可能随时加入各种信息、需求与资料的情况。

1.4.3　基于构件的开发模型

基于构件的开发模型是在面向对象技术的基础上发展起来的，是利用模块化方法将整个系统模块化，并在一定构件模型的支持下复用构件库中的一个或多个软件构件，通过组合手段高效率、高质量地构造应用软件系统的过程。基于构件的开发模型由软件计划、需求分析和定义、软件快速原型、原型评审及软件设计和实现 5 个阶段组成，采用这种开发模型的软件开发过程如图 1-12 所示。

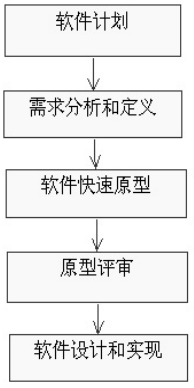

图 1-12　采用基于构件的开发模型的软件开发过程

基于构件的开发方法使软件开发不再是一切从头开始，开发的过程就是构件组装的过程，维护的过程就是构件升级、替换和扩充的过程。基于构件的软件工程在特定的应用领域内标识、

构造、分类和传播一系列软件构件，这些构件经过了合格性检验、适应性修改，并集成到新系统中。对于每个应用领域，应该在建立了标准数据结构、接口协议和程序体系结构的环境中设计可复用构件。

基于构件的开发模型的优点是：构件组装模型使软件可复用，提高了软件开发的效率。构件可由一方定义其规格说明，被另一方实现，然后供给第三方使用。构件组装模型允许多个项目同时开发，降低了费用，提高了可维护性，可实现分步提交软件产品。

基于构件的开发模型的缺点是：由于采用自定义的组装结构标准，缺乏通用的组装结构标准，因而引入了较大的风险，可重用性和软件高效性不易协调，需要精干的、有经验的分析和开发人员。客户的满意度低，并且由于过分依赖于构件，所以构件库的质量影响产品质量。

【本章小结】

本章首先介绍了信息系统的基本概念，并介绍了有关面向对象技术的大体概念，这有助于我们使用面向对象技术实现软件系统的建模工作。其次介绍了面向对象分析和设计的一般步骤。最后对软件过程模型进行了简要的介绍。本章是对信息系统与面向对象的概念等进行全景式的描述，重点是信息系统的基本概念与面向对象的特征及面向对象设计的方法。

习题 1

1. 填空题

(1) _____是面向对象技术领域占主导地位的标准建模语言，它统一了过去相互独立的数十种面向对象的建模语言共同存在的局面，形成了一个统一的、公共的、具有广泛适用性的建模语言。

(2) 面向对象方法中的_____机制使子类可以自动地拥有(复制)父类的全部属性和操作。

(3) 面向对象程序的三大特征是_____、_____和_____。

(4) _____是内部相互依赖的各个部分按照某种规则，为实现某一特定目标而联系在一起的、合理的、有序的组合。

(5) _____是创建、收集、处理、存储和传播有用数据的人员和信息技术的结合。

2. 选择题

(1) 当问题比较复杂且做出最佳决策的信息难以获得时，应使用(　　)。

A. 事务处理系统　　B. 决策支持系统

C. 管理信息系统　　D. 人工智能

(2) 对象表示的含义是(　　)。

A. 如果两个对象的属性值相同，则这两个对象就是一样的

B. 每个对象的类有唯一的属性

C. 所有的对象都彼此相同

D. 每个对象都有唯一的标识，以彼此区分

(3) 封装的含义是(　　)。

A. 封装后对象不能与外界联系

B. 确保对象中的数据只能通过操作来访问

C. 密封对象的状态，使之不能改变

D. 把对象放在集合中

(4) 下列中关于类与对象的关系说法正确的是(　　)。

A. 有些对象是不能被抽象成类的

B. 类给出了属于该类的全部对象的抽象定义

C. 类是对象集合的再抽象

D. 类是用来在内存中开辟一个数据区，存储新对象的属性

(5) (　　)模型的缺点是缺乏灵活性，特别是无法解决软件需求不明确或不准确的问题。

A. 瀑布　　B. 增量　　C. 原型　　D. 螺旋

3. 简答题

(1) 试述对象和类的关系。

(2) 请简述面向对象的概念。

(3) 请简述数据、信息和知识的区别。

(4) 请简述系统的 4 个特性。