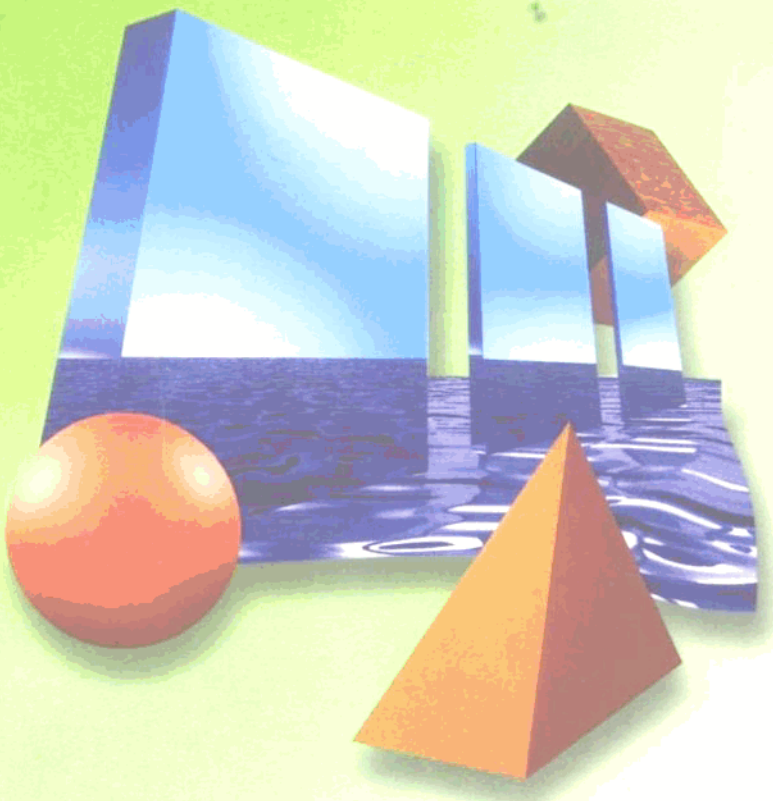


面向对象建模与设计

洪永清　黄德才　吕丽民　编著
吕丽民　审校

人民邮电出版社
PEOPLE'S POSTS & TELECOMMUNICATIONS PUBLISHING HOUSE

前　言

面向对象的建模设计是一种面向对象的软件开发方法，这种方法对实际应用的对象建模并利用这个模型去构造一种围绕那些对象且与程序语言无关的设计。面向对象建模和设计方法能为软件开发提供更容易理解的需求说明，更清楚的设计和更容易维护的系统。本书描述了一组面向对象概念和与程序语言无关的图形符号，即对象建模技术。它能被用来分析问题的需求，设计解决问题的方案，然后用程序语言或数据库实现。这种方法使得软件开发人员在分析、设计直到实现的整个软件开发过程中都能用同样的概念和符号，而不必象许多其它方法那样，需要在每个开发阶段将一个阶段的表示符号转换成另一阶段的符号。

书中展示了在整个软件生命周期(从分析、设计到实现)中如何使用面向对象概念。本书主要不是介绍面向对象语言和编码，而是强调编码是软件开发过程中的最后阶段，软件开发过程包括问题描述，需求分解，问题求解方法设计以及用特定程序语言实现。一个好的设计技术把详细实现推迟到设计的最后阶段，以保证可塑性。开发过程前期的错误对按时完成开发任务和最终产品都有很大的影响。

本书强调面向对象技术而不仅是编程方法。尤其重要的是用现实世界的概念而不是用计算机概念去抽象地思考问题。对于一些人来说这样做也许是十分困难的，因为旧的程序设计语言要求人们用计算机概念而不是实际应用概念去思考。许多注重面向对象编程的书往往不能成功地帮助程序员学会不用程序结构去抽象思维。本书所描述的图形符号能帮助软件开发者不至于过早地陷入实现去设想问题。

对大多数应用来说，面向对象技术提供了一种实用的、高产的软件开发方法，并且与最终实现用什么语言无关。本书采用非形式化方法，没有证明和希腊字母的形式定义。力求促进依据直觉得到问题求解图的实用方法。求解图是根据面向对象技术得到的并在实际问题中系统地使用的图符和方法学。书中提供了好的和坏的设计的范例和要点，使软件开发者不至于过早地陷入实现的危机。

本书适用于软件开发专业人员和学生。读者将学会如何将面向对象概念应用于软件开发生命周期的各个阶段。目前，覆盖整个软件生命周期来介绍面向对象的书很少，有的只针对编程，有的只针对分析。由于面向对象技术是当前的热门话题，而大多数读者经验有限，因此我们在编写这本书时不要求读者有任何关于面向对象概念的预备知识，只要求读者熟悉基本的计算机概念即可。然而由于本书介绍了许多新概念，即使是面向对象的程序员也能从学习系统设计的过程中获益。他们可能惊奇地发现某些以前普通的面向对象编码练习，其实违背了优良设计的原则。

本书能用作研究生或高年级本科生的软件工程或面向对象技术课的教科书或作为数据库、程序设计语言的补充教材。也能作为软件开发人员的参考书。

本书共分两个部份，第一部份介绍高级的、与程序语言无关的面向对象概念，这些概念是本书其余部份的基础。第二部份逐步介绍从问题陈述到分析，系统设计和对象设计的软件开发的面向对象方法学。除了方法学的最后阶段外，其它阶段都是与程序语言无关的。

目　录

此为试读,需要完整PDF请访问: www.ertongbook.com

第1章 引 言

面向对象建模和设计是一种用根据实际应用概念建立的模型思考问题的新方法。其基本结构是对象，它将单个实体的数据结构和行为特性结合在一起。面向对象模型对于理解问题，与应用专家交流，建立企业模型，准备文档，设计程序和数据库都是十分有用的。本书提出了一种面向对象的软件开发方法学，即对象建模技术，它贯穿于从分析设计到实现的全过程。首先建立分析模型来抽象应用范畴的本质方面而不涉及最后实现。这种模型包含有应用领域建立的对象，包括对象属性和行为特性的描述。然后设计决策并在模型中加入详细内容来描述问题以及优化实现。应用中的对象形成设计模型的框架，但它们是以计算机中的对象来实现。最后，设计模型用程序设计语言，数据库或硬件来实现。

本书介绍一种表达面向对象模型的图符。能用相同的面向对象概念和符号对应用领域和计算机领域的对象进行建模、设计和实现。从分析设计到实现过程中，这些符号的连续使用保证在一个开发阶段加入的信息在下一阶段不会丢失和转变。

1.1 什么是面向对象

顾名思义，术语“面向对象”是指“用离散对象的集合来构造软件”，这些对象具有数据结构和行为特性两方面内容。这一点不同于数据结构和行为特征之间联系松散的传统的程序设计。关于面向对象方法究竟需要什么特征目前仍存在争论，但是，它们通常包括四个方面：标识、分类、多态和继承。

1.1.1 对象的特性

标识指量化的独立数据，可区分的实体称为对象。文档中的一段，工作站的窗口和国际象棋中的白方皇后都是对象的例子。图 1.1 给出了另外一些对象实例。对象是具体的，例如文件系统中的文件，或概念上的，例如多处理操作系统中的分配策略。每个对象都有自己固有的标识。换一句话说，即使两个对象所有的属性值（诸如名称和大小）一样，它们也是有明显区别的。

在实际应用中，一个对象只是简单的存在，但在程序设计语言中，每个对象只有一个用于引用的唯一句柄。这种句柄可以用不同方法实现，象一个地址，数组下标，或属性的唯一值。对象的引用是统一的，且独立于对象的内容，但可以产生混合对象集合。例如文件系统目录包含文件和子目录。

分类指的是将具有相同数据结构（属性）和行为特征（操作）的对象组合成类。文档的段，窗口和棋子是类的例子。类是一种抽象，它描述对应用来说十分重要的特性，而忽略其它因素。任何类的选择都是任意的，是根据应用而定的。

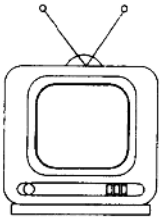
每个类描述独立对象可能的有限集合。可以说每个对象都是类的一个实例。类的每个实例对每个属性来说都有本身值，但与类中其它实例共享属性名和操作。图 1.2 给出了两个类和它们相应的实例对象。一个对象包含相对于所属类的一种隐含关系，即“知道自己是哪一类”。

多态是指一些操作对不同的类有不同的行为特征。例如，移动操作，在窗口和棋子类上有

变量名	地址
一份信用卡	10000007
一个借方	13537163
一个帐户	56826358
一个储蓄帐户	45205128

一个符号表

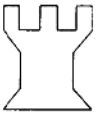
一个二叉树



灰色电视机

张红的自行车

李军的自行车



一个白色的棋子

图 1.1　对象

不同的行为特征。操作是一个对象完成或被完成的动作或转换。例如右对齐、显示和移动是一些操作实例。某一类对操作的特殊实现称之为方法。因为一个面向对象操作符是多态的，它可以有多种方法实现它。

在实际应用中，一个操作是通过不同种类对象的类似行为特征的简单抽象。每个对象知道如何执行它的操作。可是，在面向对象程序设计语言中，语言能根据操作名和所操作的对象所属的类名自动选择正确的方法实现操作。操作的使用者不必知道存在多少种方法实现所指定的多态操作。能增加新的类而不必改变已有的代码，所提供的操作也支持新类的应用操作。

继承是根据类之间的层次关系共享类间的属性和操作。一个类可以先粗略地定义，然后陆续改进成较好的子类。每个子类能合并、或继承超类的所有特性，并可增加自己独有的特性。每个子类不需重复超类的特性。例如，滚动窗口和固定窗口是窗口的子类。两个子类都继承了窗口的性质，如屏幕上的可见区域。滚动窗口增加了滚动条和偏移量。通过分析几个类的一般特性来建立一个公共超类并从超类继承特性的能力，能大大减少在设计和编程中的重复，这是面向对象系统的主要优点之一。

1.2　什么是面向对象开发

本书讨论的是关于面向对象的开发方法，它是基于抽象现实世界的一种构思软件的新方法。文中的"开发"指的是软件生命周期分析，设计的前期部分。面向对象开发的本质是应用领域中概念的识别与组织，而不是最终的编程语言的表达是否是面向对象。许多书认为软件开发的最难部分是由于问题的内在复杂性引起的本质的操作，而不是它映射到一种特殊语言时的偶然障碍，这种障碍是由于现有工具的暂时的不完善性引起的，而现有工具正迅速地得到改善。

本书不十分强调集成度、维护性和修改性，而是强调用一种精确符号表示更清晰的设计，这种表示更容易反映整个软件生命周期中的各个阶段。用于表达设计的面向对象概念和符号同样能够用在各阶段中建立文档。

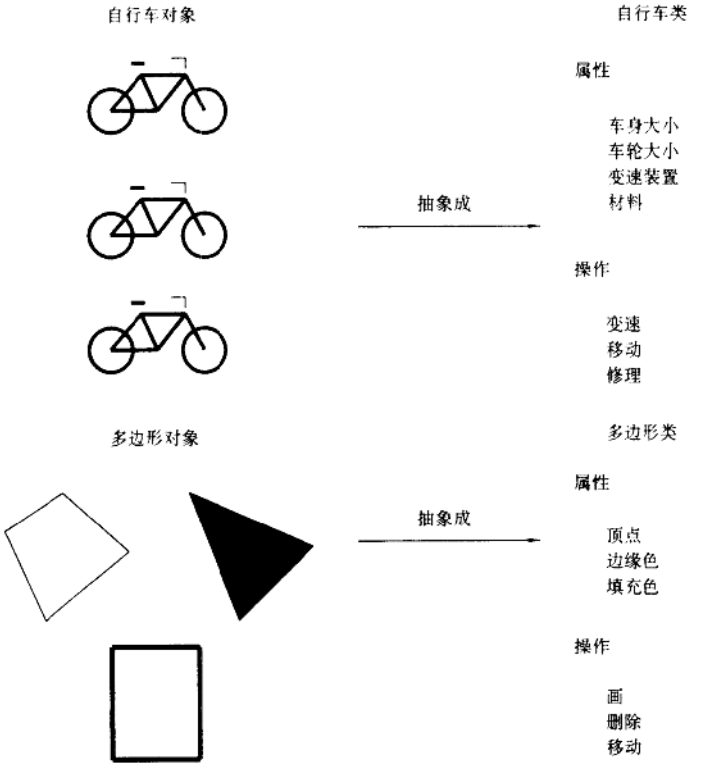


图 1.2　对象和类

1.2.1　建模概念而非实现

至今在面向对象领域，大多数工作都集中在编程语言的问题上。在文献中的共同重点是实现而不是分析和设计。面向对象编程语言对于消除传统程序设计语言的不可塑性引起的限制是有用的。可是从某种意义上讲，过于强调实现技巧而不是他们所支持的基本的思考过程，对软件工程来讲是一种倒退。

真正决定性因素来自前期概念问题的提出，而非后期的实现问题。在实现阶段弥补设计错误比在较早阶段发现并弥补的费用要大的多。过早注重实现会限制设计上的选择并经常导致劣质软件。面向对象的开发方法希望软件开发者在软件工程生命周期的各个阶段中，根据实际应用进行思考和工作。只有识别、组织和理解了应用问题的内在概念后，才能有效地提出数据结构和函数的实际细节。

面向对象的开发是独立于最后编程语言的思考过程。面向对象开发基本上是一种新的思考方法，而不是一种编程技术。它的最大的好处就是有助于说明者、开发者和用户清楚地表达抽象概念并相互交流。它能作为说明书、分析、文档、交互界面及编程的中间材料。正如一种编程工具，它具有各种实用目标，包括常用的编程语言、数据库和面向对象语言。

1.2.2 面向对象方法学

我们提出了面向对象开发的方法学和为表达面向对象概念提出的一套图符。这种方法学由建立一个实际应用模型，然后在系统的设计过程中在模型中加入一些实现细节所构成。我们称这种方法为对象模型技术，这种方法学包含以下步骤：

(1) 分析：从问题的陈述着手，分析员建立一个说明其重要特性的真实情况的模型。因为问题陈述最初很少是完整的和正确的，所以为理解问题，分析员必须与客户一起工作。分析模型是要求系统必须做什么的扼要的精确的抽象，而不是它将如何去实现。模型中的对象应当是实际应用的概念而不是象数据结构这些计算机概念。一个好的模型应当能被非程序员的应用专家所理解和评价。分析模型不应当含任何实现决策。例如，在工作站窗口系统中的窗口类应当按照用户可见的属性和操作来描述。关于分析的具体阐述见第 7 章。

(2) 系统设计：系统设计者对整个体系结构作高层决策。在系统设计中，根据分析结构和所提供的结构体系将目标系统组织成子系统。系统设计者必须决定优化哪些特征，选择解决问题的策略和暂时的资源分配。例如，系统设计者必须决定工作站屏幕在移动和擦除时的变化必须是快速的和圆滑平稳的、选择适当的通信协议和内存缓冲策略。第 8 章具体阐述系统设计。

(3) 对象设计：对象设计者按照分析模型建立设计模型，但在设计模型中包含实现细节。设计者按照系统设计阶段建立的策略在设计模型中增加一些细节。对象设计的关键是实现每个类所需的数据结构和算法。来自分析的对象类仍很有用，但增加了计算机中的数据结构和用于优化重要性能所选择的算法。尽管计算机中的对象和应用中的对象在概念上不同，但都用同样的面向对象概念和符号进行描述。例如，窗口类操作现在按照基本的硬件和操作系统来说明。第 9 章具体阐述对象设计。

(4) 实现：在对象设计阶段形成的对象类和关系最后被转换成特殊的程序设计语言、数据库或者硬件的实现。编程在开发过程中是相对较小的和机械的部份，因为在设计过程中已解决了所有困难的决策。目标语言在某种程度上会影响设计决策，但设计不应该依赖最后的程序设计语言细节。在实现过程中，按照好的软件工程方法去实施是重要的，这样能直接跟踪设计和保证实现的系统的可适应性和可扩充性。例如，用程序语言编码的窗口类应调用工作站的基本图形系统。

面向对象的概念能应用于整个系统开发的生命周期，从分析、设计到实现。尽管在后续阶段加入实现细节，但从一个阶段到另一个阶段能用同样的类而不改变符号。尽管窗口的分析观点和实现观点都是正确的，但它们服务于不同的目的和表示不同的抽象层。而标识、分类、多态和继承的相同的面向对象概念却应用于整个开发周期。

一些类不属于分析的内容，但在设计和实现中引入。例如，树、散列表和链表这样的数据结构不是实际问题的表示。引用它们是为了支持设计过程中的特殊算法。这样的数据结构对象在计算机中用来实现实际对象，并不是直接从实际应用问题中派生出它们的特性。

1.2.3 三种模型

OMT 方法学用三种模型描述一个系统：对象模型，描述系统中的对象和它们的关系；动态模型，描述系统中对象之间的互相作用；函数模型，描述系统的数据转换。每个模型在开发的所有阶段都是可应用的，且随着开发的进行逐渐获得实现的细节。一个完整的系统描述需要所有三种模型。

对象模型描述系统中的静态结构和它们的关系。对象模型包含对象图。对象图的结点是对象类，弧是类之间的关系。第3、4章具体阐述对象模型。

动态模型描述了系统在整个生存期的变化。动态模型用于说明和实现系统的控制内容。动态模型包含状态图。状态图的结点是状态，弧是由事件引起的状态间的变迁。第5章具体阐述动态模型。

函数模型描述了系统中数据值的变换。函数模型包括了数据流图。数据流图表示一种计算。数据流图的结点是处理而弧是数据流。第6章具体阐述函数模型。

三种模型是一个完整的系统描述的三个正交的部份且相互间是交连的。可是对象模型是最基本的，因为在描述什么变化和如何变化前必需描述是什么样的变化和转换。

1.2.4 与函数方法学的差异

面向对象方法学不同于以前的面向函数方法学。在以前的面向函数方法学中，重点在于说明和分解系统函数。这样一种方法似乎是实现所希望的目的最直接的方法，但是得到的系统可能是脆弱的。如果需求变化，基于函数分解的系统可能需要大量的重构造(公正地说：这些方法学比较复杂，更详细的内容参见第11章)。

相反，面向对象的方法首先着眼于根据实际应用去识别对象，然后围绕这些对象提出相应的过程。虽然这似乎是不够直接，但面向对象软件能较好地适应需求改变，因为它是建立在自己的应用领域的基本结构上，而不是单一问题的特定的函数需求。

1.3 面向对象的主题

目前的面向对象技术有几个主题。虽然这些主题对面向对象系统来说并不唯一，但它们能很好支持面向对象系统。

1.3.1 抽象

抽象的构成集中在一个实体的重要的、内在的方面而不考虑它的偶然特性。在系统开发过程中，这意味着在决定应该怎样实现对象前，将重点放在该对象是什么和做什么。通过避免过早地确定细节，抽象的运用保证尽可能自由地作出决定。现代语言具有数据抽象，但继承和多态的使用提供了更强的能力。在分析阶段运用抽象意味着在理解问题之前只处理实际应用的概念，而不是设计、实现决策。正确使用抽象可以在分析、高层设计、程序结构、数据库结构和文档中使用同样的模型。一种独立于语言的设计风格将编程细节推迟到相对来说较为机械的开发最后阶段。

1.3.2 封装

封装，也即信息隐藏，是将其它对象可访问的外部内容与对象的隐藏的内部细节分开。封装避免了程序间过多依赖，而这种依赖导致一个小小的改变会引起很大的波动效应。一个对象实现的改变不影响这个对象的应用。人们往往希望改进对象的操作性能，弥补缺陷，合并代码和代码移植。封装不是面向对象语言所唯一的，但在单个实体中复合数据结构和行为特性的能力使封装比在分离数据结构和行为特性的传统语言中更清晰和更有效。

1.3.3 复合数据和行为特性

实际调用者不必考虑给定操作有多少种实现存在。操作符的多态性减轻了从访问代码到类层次过程确定使用什么实现的难度。例如，显示窗口内容的非面向对象代码必须区分每个图形的类型，如多边形、圆或文本，并调用相应的过程去显示它。而面向对象程序将简单地调用每个图的画操作；根据它的类，使用哪个过程的决策由每个对象隐含。当应用程序调用操作时不必要地次重复选择过程。因为加入新类时，不必修改调用代码，所以维护也容易得多。在一个面向对象系统中，数据结构体系和操作的继承体系是一致的(见图 1.3)。

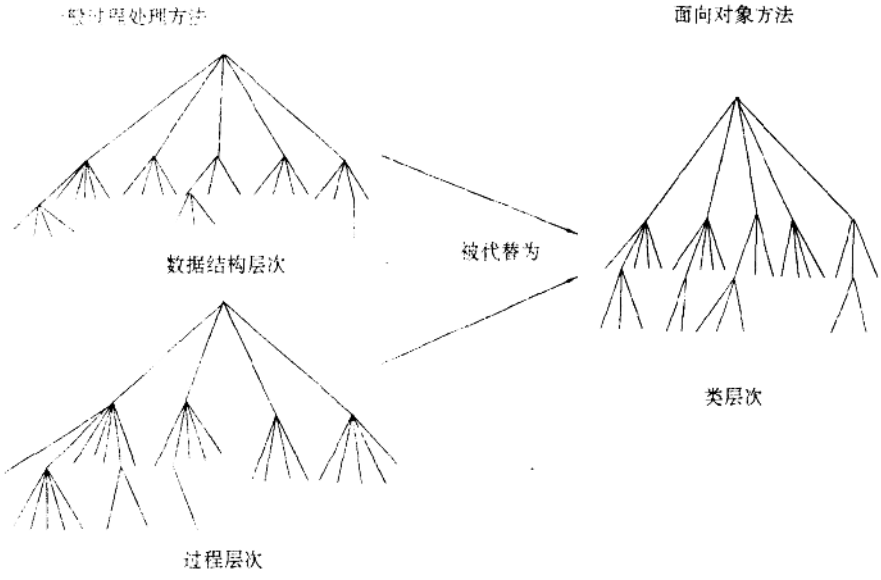


图 1.3 具有统一层次结构的面向对象方法

1.3.4 共享

面向对象技术加强了几个不同层次的共享。数据结构和行为特性的继承允许几个类似的子类无冗余地对公共结构共享。运用继承进行代码共享是面向对象语言的主要优点之一。比节省代码更为重要的是识别实际同一事情而不同操作的概念上的清晰性。这就从数量上减少了必须理解和分析的不同情况。

面向对象开发不仅允许在一个应用中共享信息，而且给将来的项目提供了重复利用设计和代码的可能。尽管这一点作为面向对象技术强调有些过份，但面向对象开发技术提供了这些可用来构建可重用部件库的工具：如抽象、封装和继承。可是，面向对象并不是一个能确保重用性的万能公式。重用并不是要发生就能发生的；它的设计必须不只限于考虑即时的应用而要额外努力使设计更为通用。

1.3.5 重点在于对象结构，而不在过程结构

面向对象技术强调对象是什么，而不是如何使用该对象。对象的使用十分依赖于实际应用

此为试读，需要完整PDF请访问：www.ertongbook.com

细节并且在开发过程中经常变化。当需求改变时，对象所提供的特征比它所用的方法更为稳定。因此，从长远观点来看，按照对象结构开发的软件系统更为稳定。与传统的函数分解方法学相比，面向对象开发更着重数据结构而不是过程结构。尽管面向对象开发增加了类依赖特性，但在这方面，面向对象开发类似于数据库设计中的信息建模技术。

1.3.6 协作

标识、分类、多态和继承是面向对象语言的主要特征。其中每一种都能单独使用，但一起使用能互相补充。面向对象方法的好处远比初看它时多得多。更多地强调对象的本质特征迫使软件开发人员更仔细地思考对象是什么和做什么等问题，结果产生的系统比只强调数据和操作使用的系统更清晰、更适用、更坚固。按 Thomas 的观点，这些不同特点的综合运用必将创建一种不同的编程风格。而 Cox 则主张封装是面向对象方法的基础，它将重点从编码技术转移到封装技术上，而根据封装建立继承使代码重用成为现实。本章主要概念如图 1.4 所示。

抽象	函数模型	对象设计
分析	标识	面向对象
分类	实现	面向对象建模技术
动态模型	继承	多态
压缩	对象模型	系统设计

图 1.4 第 1 章的关键概念

练 习

说明：每道练习题括号中的数表示练习的难度，从 1(容易)到 10(非常难)。

1. 在过去的软件课题中你碰到的主要问题是什么？估计你花在分析、设计、编码和测试的时间百分比。你如何评估一个课题所需的工作量。(2)

2. 回顾你在过去生成的系统。简要描述此系统。在设计中碰到的障碍是什么？如果有的话，你使用什么软件工程方法学？你选择或不选择一种方法学的理由是什么？你对所存在的系统满意吗？对系统增加新的特点困难性如何？可维护性如何？(3)

3. 描述一个最近五年开发的不能按期完成，超预算的或不能实现预期性能的大型软件。应改进的因素是什么？如何才能避免失败？(4)

4. 从用户的观点，评论特别使你困惑的硬件和软件的缺陷。描述一个系统，其缺陷如何被忽视，且在设计期间如何作更多考虑能够避免它。(3)

5. 所有对象具有标识并且是可区分的。可是，对于大的对象集合，设计一个区分它们的方案是十分烦琐的工作。此外，一种方法也许与区分的目的有关。对于下面收集的每一个对象，说明如何才能区分它们：(5)

a. 所有在邮寄的人。

b. 所有在作犯罪研究的人。

c. 所有在给定的一个银行有保险箱的人。

d. 所有在进行呼叫的电话。

e. 所有在电话公司结算的客户。

f. 全世界所有电子邮寄地址。

g. 由于安全原因限制访问的公司职员。

6. 设计一张下面每一个系统都能操作的对象表：(4)

a. 报纸排版程序。

b. 计算和存储滚木游戏分数的程序。

c. 电话应答机。

d. 录象机控制器。

e. 目录储存顺序输入系统。

7. 下面有两张表。第一张是描述实现对象的类表。第二张是操作表。对于每一个类选择一个此类上某对象有响应的操作。讨论每个类中每个操作的特性。(6)

类：

变长数组——对象的有序集合，下标用正整数，它的大小在程序运行中能够变化。

符号表——一张将文本键入符映象成描述符的表。

集合——不重复对象的无序集。

操作：

添加——在集合的尾部增加一个对象。

拷贝——对集合进行复制。

统计——对集合的元素数目进行统计。

索引——按给定的位置从集合中恢复对象。

交——求两个集合的公共元素。

插入——将所给对象按指定位置放置进集合。

修改——在集合中增加一个成员，无论它是否存在都重写。

8. 讨论下列每一表中有什么样的共同对象。对每一表你可增加更多的类。(4)

a. 扫描电子显微镜，眼镜，望远镜，轰炸瞄准器，双筒镜

b. 管道，单向阀，水龙头，过滤器，压力计

c. 自行车，帆船，小汽车，卡车，飞机，滑翔机，摩托车，马

d. 铁钉，螺钉，螺栓，铆钉

e. 帐蓬，岩洞，棚子，车库，堆房，摩天楼

f. 平方根，指数，正弦，余弦

第 2 章　建模——一种设计技术

模型是为了在构造事物前理解事物而对事物作出的一种抽象。因为模型省略了一些不必要的细节，所以对模型操作要比对原始实体操作更加容易。抽象是人类处理复杂事务的基本能力。几千年来，工程师、艺术家和工匠等在实现他们的设计前，都是用建立模型进行试验。软、硬件系统的开发也不例外。为了建造复杂的系统，开发者必须从不同角度对系统进行抽象，用精确的符号建立模型，检验模型是否满足系统的需要，然后逐步增加细节将模型变成实现。

本书的第 1 部分描述了涉及面向对象建模的概念和图形符号。这些概念将被用于本书的第 2 部分的分析和设计。本章首先从总体上讨论建模思想，然后介绍组成对象建模技术的三种面向对象模型：描述静态结构的对象模型、描述瞬时关系的动态模型和描述属性值间的函数关系的函数模型。

2.1　建　模

设计者出于各种目的在构建实物前创建了许多不同模型，如给顾客看的建筑模型、作为风洞测试而建立的飞机模型、油画作品的铅笔素描、机器部件的兰图、广告板和书的提纲等。使用模型有以下几个目的：

· 在建立物理实体前进行测试。中世纪的泥工不知道现代物理，但他们按比例构造了哥特式教堂的模型以测试结构的强度。飞机、汽车和船的比例模型在风洞或水池里进行测试，以改进它们的空气动力学性能。近年来计算机的飞速发展已经允许不建立实际模型而模拟许多物理结构。这种模拟不但经济，而且提供了测试物理模型时转眼即逝的或者得不到的信息。物理模型和计算机模型通常都比建一个完整的系统要经济，并且能较早地纠正错误。

· 同顾客交流。建筑师和产品设计师造出模型给顾客观看。样品模型是模仿一个系统的一些或所有外部行为的演示产品。

· 可视性。电影图片、电视画面和广告都允许作者观察到他们的设计思路。在开始详细制作之前，作者可以改变蹩脚的情节、多余的结尾和不必要的段落。艺术家的草稿允许他们草拟思想并在作品定型前修改它。

· 降低复杂度。也许是建模的主要原因（综合前面的所有理由）是要处理那些过于复杂以至不能直接理解的系统。人类思维每次只能处理有限的信息。而模型通过每次处理分解出的少量重要事物，从而降低了复杂性。

2.1.1　抽象

抽象是问题的某些方面的选择性检查。抽象的目的主要是为了提炼出相对某种目的是重要的方面，而忽略次要的方面。抽象总出于某一目的，因为目的决定了哪些方面是重要的，哪些方面是不重要的。根据不同的抽象目的，同一事物可有不同的抽象。

所有的抽象都是不完全的，不精确的。现实就像一张无缝的网，我们所谈的和所描述的一切都只是它的片段。所有人类的词汇和语言都是抽象——客观世界的不完全的描述。但这并

不影响抽象的有用性。抽象的目的就是为了将事物限制到我们能力所及范围。因此在建造模型时，我们用不着寻找绝对的真实，而只需找出一些足以说明某一目标的内容就行了。世上没有唯一"正确"的模型，只有充分与不充分之分。

一个好的模型能反应问题的主要方面而省略次要方面。如多数计算机语言对建模算法来说基本是无用的，因为它们强调的是和算法无关的实现细节。一个包含涉及无关细节的模型将限制你的设计决策和分散对真正问题的注意力。

2.2 对象建模技术

我们发现从三个不同但又相关的角度去建立系统模型是很有用的，这三个角度各自都抓住了系统的一些重要方面，但对于系统的完整描述来说都是需要的。对象建模技术是综合这三个观点建立系统模型的方法学的名字。"对象模型"代表了系统静态的、结构化的数据；"动态模型"代表了系统暂行的，行为的控制方式；"函数模型"则代表了系统的变化函数。一个普通的软件过程包括所有三个方面：它运用数据结构(对象模型)，按时间调整操作顺序(动态模型)和改变值(函数模型)。每个模型都包括对应于另外模型中的实体。例如，操作可归并到对象模型中的对象，但在函数模型中则更充分的展开。

这三种模型将一个系统分成正交视图，并能用统一符号表示和操作。不同的模型并不完全独立，一个系统并不仅仅是独立部分的集合 但在很大程度上，每个模型自身都能进行测试和理解。不同模型间的相互联系是有限的，明确的。当然，也有可能形成不理想的设计，三个模型交错混杂以致不能分离，然而，一个理想的设计能区分一个系统的不同方面并且限制它们之间的耦合。

在开发周期中，每一种模型都不断发展。在分析阶段，在不考虑最终实现的基础上创建实际应用的模型。在设计阶段，在原有模型上增加了应用问题的解决方法。在实现阶段，则对应用域和求解领域的结构进行编码。单词"模型"有两层意思——系统的观点(对象模型，动态模型或函数模型)和开发的阶段(分析，设计或实现)，根据前后关系，它的含义一般是很明显的。

2.2.1 对象模型

对象模型描述了系统中对象的结构 — 它们的标识、与其它对象的关系、属性、和操作。对象模型提供了动态和函数模型都适用的基本框架。如果不存在将改变或转换的事物，那么该改变和转换是无意义的。对象是划分事件的单元，是模型的组成成份。

构造对象模型的目的是想捕获那些来自实际事件且对应用很重要的概念。在建立工程问题模型中，对象模型应该包括熟悉的工程术语；在建立商业问题的模型中，术语应该来自商业；在建立用户界面的模型中，术语应该来自应用领域。除非建立的模型是计算机问题(如编译成操作系统)，一个分析模型不应当包含计算机结构。设计的模型描述如何去解决一个问题，它可以包含计算机结构。

可以用包含对象类的对象图来表示对象模型。类按层次排列，并共享公共结构和行为特征，类与别的类相关联。类定义了每个对象实例所取的属性值和每个对象执行或经历的操作。

2.2.2 动态模型

动态模型描述了系统中与时间和操作顺序有关的内容 — 标记变化的事件、事件的顺序、

定义事件背景的状态以及事件和状态的组织。动态模型着眼于控制：描述系统中发生操作的顺序，但不考虑操作做什么、对什么进行操作或如何实现。

动态模型用状态图表示。每张状态图说明了系统中对象类所允许的状态和事件序列。状态图也涉及到别的模型。状态图中的动作对应于函数模型中的函数；状态图中的事件为对象模型中对对象的操作。

2.2.3 函数模型

函数模型描述了着眼于系统中与值转换有关的那些方面——函数、映射、约束和函数作用范围等。函数模型只着眼于系统做什么，而用不着考虑怎样做，何时做。

函数模型用数据流图表示。数据流图表示根据输入值和函数进行的输出值的计算与值之间的相关性。而不考虑函数是否执行和什么时候执行。象表达树这些传统的计算机概念是函数模型的例子。扩展表则是较新的函数模型例子。在动态模型中，函数作为动作被唤醒；而在对象模型中则作为对对象的操作。

2.2.4 模型间的关系

每个模型描述系统的一个方面，但包含和其它模型的联系。对象模型描述了动态模型和函数模型操作的数据结构。对象模型中的操作对应于动态模型中的事件和函数模型中的函数。动态模型描述对象的控制结构。它表示依赖于对象值并导致改变对象值和唤醒函数的动作的决策。函数模型描述由对象模型中的操作和动态模型中的动作唤醒的函数。函数对对象模型指定的数据值操作。函数模型也给出对象值上的约束。

各模型应该包含哪些属性，还存在一定程度的模糊性。这是自然的，因为任何抽象都只是粗略地反应现实；不可避免地存在一些越界的抽象。系统的某些性质不能通过模型表示，这也是正常的，因为没有一个抽象能概括一切。目的是简化系统描述，而不希望模型具有很多结构，以至变成大而无助的模型。对那些模型不能充分说明的事物，自然语言或特殊的应用符号仍是一种好的描述工具。

2.3 本章小结

模型是在实现问题求解之前，为了理解而创建的抽象描述。所有的抽象都是为特殊目的选择的实际事物的子集。

对象建模技术由三种模型组成。对象模型描述一个根据对象和相对于实体关系的系统的静态结构。动态模型是根据事件和状态描述的系统控制结构。函数模型是根据值和函数来描述系统的计算结构。不同的问题对三种模型的侧重也不同，但对任何大系统而言，三种模型都是必需的。本章主要概念如图 2.1 所示。

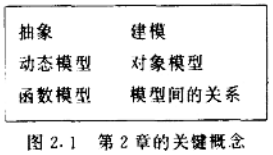


图 2.1 第 2 章的关键概念

练　习

1. 一个轮胎的属性有尺寸，原料，内部结构（例如斜折，铁片带等），花纹设计，价格，过期年限和重量。当你在决定是否为你的汽车买轮胎时，哪些因素是重要的呢？对于某些用计算机模拟汽车防滑系统性能的人，哪些因素有关？（1）

2. 假如你洗澡间的下水道堵住了，你不得不用电线去疏通它。而在房间里有好几种电线，绝缘的和不绝缘的。请问当你在选择电线时，需考虑电线的哪几个属性？请选择并解释。（2）

- 对电子噪声的抗干扰性
- 绝缘体的颜色
- 该绝缘体在盐水中的阻抗
- 该绝缘体加热时的阻抗
- 价格
- 硬度
- 剥去绝缘体的容易程度
- 重量
- 可用性
- 强度
- 高温时的阻抗
- 抗拉力

3. 电线常常用在以下几个方面。对每个应用，请准备相关的电线属性表，并解释每种属性为什么重要？（3）

- 用于横渡大西洋的电线
- 用于创作彩色艺术作品的电线
- 设计飞机上电子系统的电线
- 用于悬挂鸟进料器的电线
- 设计钢琴的电线
- 用于造灯丝的电线

4. 如果你正在设计通过电话线把文件从一台计算机传送到另一台计算机的系统，在下面的属性中你会选哪些？并解释你为什么选择这种属性以及与你要解决的问题有关。（3）

- 通讯线上的电噪声
- 串行数据的传输速度，一般是每分钟 300，1200，2400，4800 或 9600 比特。
- 一个好的关系数据库。
- 一个好的全屏幕编辑器
- 诸如用于调节数据输入流的 XON/XOFF 协议的缓冲和信息流控制。
- 在硬、软磁盘上的磁道、扇区。
- 字符（象特殊的控制字符）的翻译。
- 文件组织，如作为一个记录的字节串。
- 数字协处理器。

5. 有好几个模型可用于电动机的分析和设计。一台电子模型涉及电压、电流、电磁场、电

此为试读，需要完整PDF请访问：www.ertongbook.com