# 面向对象程序设计

MIAN XIANG DUI XIANG CHENG XU SHE JI

主编　房晓溪

高等教育出版社
HIGHER EDUCATION PRESS

# 面向对象程序设计

主　编　房晓溪
副主编　杜小丹

主　审　梅　挺
副主审　黎　扬



高等教育出版社

## 内容简介

本书是按照本科院校学生的培养目标和基本要求，并结合多年来教学和软件工程实践的经验，为实施教学改革，使计算机教学进一步面向应用实践而编写的教材。

本书内容包括：从面向过程到面向对象、C++程序设计初步、表达式和引用、程序控制语句、类与对象、构造函数与析构函数、继承与派生、多态性和虚函数、进一步使用成员函数、运算符重载、流类库、模板、异常处理等。每章后都配有习题，书后附录提供了部分习题答案和课程设计，体现了教材的理论性和实用性的统一。本书可作为计算机及相关专业的教学用书，也可作为工程技术人员的参考用书。

**图书在版编目(CIP)数据**

面向对象程序设计/房晓溪主编.—北京：高等教育出版社，2003.7

ISBN 7-04-012008-9

Ⅰ.面… Ⅱ.房… Ⅲ.面向对象语言-程序设计-高等学校-教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2003)第 047650 号

**责任编辑** 陈 红 **封面设计** 吴 昊 **责任印制** 蔡敏燕

| | |
|---|---|
| **书　　名** | 面向对象程序设计 |
| **主　　编** | 房晓溪 |

| | | | |
|---|---|---|---|
| **出版发行** | 高等教育出版社 | **购书热线** | 010-64054588 |
| **社　　址** | 北京市西城区德外大街 4 号 | | 021-56964871 |
| **邮政编码** | 100011 | **免费咨询** | 800-810-0598 |
| **总　　机** | 010-82028899 | **网　　址** | http://www.hep.edu.cn |
| **传　　真** | 021-56965341 | | http://www.hep.com.cn |
| | | | http://www.hepsh.com |
| **排　　版** | 南京理工排版校对公司 | | |
| **印　　刷** | 商务印书馆上海印刷股份有限公司 | | |
| **开　　本** | 787×1092　1/16 | **版　　次** | 2003 年 6 月第 1 版 |
| **印　　张** | 19 | **印　　次** | 2003 年 6 月第 1 次 |
| **字　　数** | 451 000 | **定　　价** | 24.50 元 |

# 前　言

面向对象的程序设计近几年已经成为软件开发的主流方法和思想，这种方法通过增加软件的可重用性和可扩充性来改善并提高程序员的软件开发和设计能力。目前面向对象的程序设计中应用最广泛的语言就是C++语言，通过学习C++语言来学习面向对象的程序设计，不但能够掌握面向对象的程序设计思想，而且能够广泛地实际应用于实际。

本书较为全面地介绍了C++语言的各方面内容。第一章介绍了面向对象的概念和面向对象的设计，同时将C++语言和其他面向对象的程序设计语言作了比较分析。第二章开始实际进入C++程序的开发，介绍了C++程序的开发步骤，VC 6.0集成开发环境和预处理程序。第三章介绍C++程序语句中的具体要素，表达式和引用。第四章介绍了程序的控制语句。第五章到第八章比较系统的介绍了C++语言中面向对象的三个最主要的特性：封装(类与对象，对象构造与析构)，继承与派生，多态性与虚函数。第九章介绍了进一步使用成员函数，第十章介绍了运算符重载，第十一章介绍了流类库，第十二章介绍了模板，第十三章介绍了异常处理。每章后面都有习题，并且在书后附录中给出了部分习题答案，以便于教师授课和学生自学，最后，还结合C++的实际应用，提供了课程设计(Socket编程)。

本书由房晓溪主编，杜小丹副主编，第一章由刘文清编写，第二章、第八章和课程设计由陈祥曦编写，第三章由代群编写，第四章和第十章由杜小丹编写，第五章由刘仕筠编写，第六章由鄢涛编写，第九章由张弘编写，第十一章、第十二章和第十三章由赵卫东编写。第七章由房晓溪编写，房晓溪负责了全书的策划、统稿、主编和审核工作，主审梅挺，副主审黎扬负责了全书的审订工作。

由于时间仓促，水平有限，在本书编写过程中，难免有不足和错误的地方，恳请读者提出批评和指正。

作　者

2003年4月

# 目　录

# 第 1 章

# 从面向过程到面向对象

**内 容 提 要**

本章讲述什么是面向对象，面向对象技术的产生，面向对象中的几个基本概念以及面向对象程序设计的理论基础，它们不依赖于具体的程序设计语言，也就是说，无论使用哪种面向对象语言进行面向对象程序设计，本章内容都具有指导意义。

## 1.1 面向对象技术的产生

在面向对象技术产生之前，一直使用结构化程序设计方法，也就是面向过程的程序设计方法。但是，随着计算机科学的发展和应用领域的不断扩大，对计算机技术的要求也越来越高。由于结构化程序设计存在可重用性差和难以维护等缺点，因此结构化程序设计语言和结构化分析与设计已无法满足用户需求的变化。提高软件质量，缩短软件开发周期，发展软件可靠性、可扩充性和可重用性迫使软件界人士不断研究新方法、新技术，探索新途径，由此推动了面向对象技术的出现。

### 1.1.1 面向过程方法面临的问题

面向过程程序设计诞生于 20 世纪 60 年代，它的主要思想是自顶向下、逐步求精，换句话说，就是将功能不断细分，直到程序容易实现为止。当程序功能十分复杂，无法直接用算法进行描述时，可以将这个大的功能拆分为一系列较小的功能模块；如果已拆分的功能模块仍然复杂，则再将其进行拆分，直到这些功能模块易于实现为止。例如，计算一个班级中每个学生的平均成绩是一项较为复杂的任务，可以将其进行如下分解：

(1) 找出一个学生的成绩；

(2) 计算总共有多少学生；

(3) 计算总成绩；

(4) 用总成绩除以学生人数。

计算总成绩本身又可以分为一系列子任务：

(1) 找出每个学生的档案；

(2) 读出成绩；

(3) 把成绩加到部分和上;

(4) 读出下个学生的档案。

类似地,读出每个学生档案中的记录又可以分解为一系列子任务:

(1) 打开学生的档案;

(2) 找出正确记录;

(3) 从磁盘读取数据。

面向过程程序设计为处理复杂问题提供了有力的手段。由于该程序设计方法采用了模块分解与功能抽象的手段,从而有效地将一个复杂的程序系统的设计任务分成许多易于控制和处理的子任务,这些子任务都是可独立编程的子程序模块。这些子程序都有各自清晰的界面,使用起来非常方便。

但是,人们在使用面向过程的方法进行程序设计的时候,逐渐感到该方法的局限性。到 20 世纪 80 年代末,它的一些缺点越来越突出。当数据量增大时,数据与处理这些数据的方法之间的分离使得程序的可读性和可重用性变差。对数据处理能力的需求越强,这种分离所造成的负作用就越显著。因此,人们试图开发一种新的程序设计思想。

### 1.1.2 面向对象的概念

"面向对象"是软件程序设计中的一种新思想,由于这种新思想的引入,使得程序设计能更加贴近现实,并且花费更小的精力。面向对象方法学的出发点和所追求的基本目标是使人们分析、设计与实现一个系统的方法尽可能接近人们认识一个系统的方法。也就是使描述问题的问题空间和解决问题的方法空间在结构上尽可能一致。其基本思想是:对问题空间进行自然分割,以更接近人类思维的方式建立问题域模型,以便对客观实体进行结构模拟和行为模拟,从而使设计出的软件尽可能直接地描述现实世界,构造出模块化的、可重用的、维护性好的软件,同时限制软件的复杂性和降低开发维护费用。

下面先来认识一下对象这个概念。在现实生活中,经常遇到对象,这个对象表示现实世界中的某个具体的事物。如一张课桌、一位同学等。而在面向对象的方法中,所谓的对象,则是指一组数据和使用该组数据的一组相关操作的集合的一个实体。对象具有如下一些特点:

(1) 对象以数据为中心。根据数据要求来确定所需要的操作,不设置与这些数据无关的操作,而操作的结果往往与当时数据的值有关。

(2) 对象是主动的。对象与传统的数据有本质的不同,它不是被动的接受外界对它的改变,而是根据外界的要求来能动地完成外界所需要的功能。

(3) 实现了数据封装。面向对象的对象机制有一个最大的特点就是外界对其功能实现不可见。也就是说这些实现细节被封装了起来,用户不必知道。

对象是面向对象中的一个非常重要的概念。下面,系统地看看面向对象中的几个核心概念:封装、继承和多态。

#### 1. 封装

封装,从字面上理解,就是将某事物包围起来,使外界不知道其实际内容。在面向对象的方法中,封装是指将一段程序代码"包装"起来,只需知道这段程序代码所完成的功

能，而不必弄懂完成这些功能的实现细节。这段程序代码包括一组数据和与这组数据相关的操作。例如，可以把理发师看成是一个封装体。如果需要理发，那么只需要告诉理发师所需要的想理发型，理发师则可以按提出的要求来完成这个操作。在这里，理发师相当于被包装起来的一段代码，理发是他所能完成的功能，而理发者则相当于使用这段代码的用户，理发者并不需要懂得如何理发。

类和对象是实现封装的重要机制。在前面，对对象已经有了一个大概的了解，现在要将对象和类放到一起来讨论。先回忆一下在 C 语言中所使用过的数据类型——整型(int)。它是系统已经定义好的，使用者只需要合法的使用即可。6 是整型的一个具体值，当然用户还可以定义整型的变量。如使用语句“int i;”用户就可创建一个整型变量 i。

类是一个类型，就相当于整型，是一个抽象的概念；而类类型的变量，就是对象，也就相当于整型的一个变量 i。因此，类和对象的关系就是：类是对象的抽象，对象是类的实例化(具体化)。

2. 继承

前面所讨论的类都是一些孤立的类，它们相互之间还没有建立关系，也就是说，这些类都处在同一级别上，是一种平坦结构。而这种没有相交关系的平坦结构限制了面向对象系统的设计，这是因为它不允许类之间实现信息共享。在系统中有些对象，它们有一些相同之处，但又有一些差别，如果不允许类之间建立相交关系，这些不同对象的相似之处就无法表示出来。

继承所表达的就是一种对象类之间的相交关系，它使得某类对象可以继承另外一类对象的特征和能力。例如要制造新的电视机，可以有两种选择：一是从草图开始，另一种是对现有的型号加以改进。也许现有的型号已经令人满意，但如果再加一个功能，会更加完美。电视机工程师肯定不想从头开始，而是希望制造另一种新型电视机，该机是在原有的型号基础上增加一组电路组成的。新的电视机很快就制造出来了，被赋予一种新的型号，于是新型电视机就诞生了。这是继承的实例。

C++采用继承支持重用的思想，程序可以在扩展现有类型的基础上声明新类型。新子类是从现有类型派生出来的，称为派生类。新型电视机是在原有型号的电视机上增加若干功能得到的，所以新型电视机是原有电视机的派生，继承了原有电视机的所有属性，并在此基础上增加了新的功能。

3. 多态

所谓多态，是指一个名字(函数、运算符)有多种语义；或一个相同界面，有多种实现。前者称为重载，后者称为虚函数。

重载的含义是指通过为函数和运算符创建附加定义而使它们的名字可以重载。重载有两种：函数重载和运算符重载。函数重载是指在同一作用域内的若干个参数特征不同的函数可以使用相同的函数名字；运算符重载是指同一个运算符可以施加于不同类型的操作数上面。也就是相同名字的函数或运算符在不同的场合可以表现出不同的行为。在C++语言中函数重载是通过静态联编实现的。

虚函数体现了另一种多态性。虚函数是指在类等级的不同层次中说明的名字、参数特征和返回值类型都相同但实现算法不同的函数。虚函数机制使得程序员能在一个类等

级中使用相同函数的多个不同版本,在运行时刻通过接收消息的对象所属的类来决定执行哪个版本的函数。运行时才能确定具体执行的情况称为动态联编。

### 1.1.3 程序设计语言的发展

程序设计语言(Programing Language)也称计算机语言,即编写计算机程序所用的语言。计算机语言是人和计算机交流信息的工具。程序设计语言沿着由难而易的方向发展,大致可以分为以下几个阶段:

1. 第一代语言——机器语言

机器语言是以二进制代码的形式组成的机器指令集合,它依赖于机器。不同的机器有不同的机器语言,存储安排也由语言本身控制。计算机可以直接识别和执行机器语言程序,因此用这种语言编制的程序运行效率极高。但是程序全部由0、1二进制组成,很不直观。编写很简单的功能就需要大量代码,而且编写起来效率比较低,很容易出现错误。此外,不同的计算机有不同的机器语言,不能通用,重用性差。

由于计算机只能存储和识别二进制的数据和指令,所以在机器语言中,每条指令的操作码和地址码都用二进制(或八进制)编码。存放数据和指令的地址也用二进制(或八进制)编码。数据需预先转换成二进制。因此,机器语言也称为二进制语言。

2. 第二代语言——汇编语言

汇编语言也称符号语言。它是用符号代替机器语言中的二进制编码,这样看起来比较直观,使用起来方便了很多,错误也相对减少。汇编语言增加了一些功能,如宏、符号地址等,存储空间的安排由机器完成。

计算机不能直接识别和执行汇编语言程序,它必须经过一个汇编程序转换成机器语言后才能执行。并且不同的指令集的机器仍有不同的汇编语言,程序重用性也很低。

3. 第三代语言——高级语言

高级语言是与机器不相关的一类程序设计语言,读写起来更接近人类的自然语言,因此,用高级语言开发的程序可读性较好,语言简洁。同时,由于高级语言并不直接和硬件相关,其编制出来的程序可移植性和可重用性也要好得多。这类语言大多采用结构化的程序设计思想,编程思路是自顶向下,逐步求精。

4. 第四代语言

一种还未成熟的语言。它具有一定的智能,更接近于日常语言,它对语言的概括更为抽象,从而使语言也更为简洁。

从程序设计语言的发展过程可以看到,人们在不断地简化编程的难度,使程序开发更快、更好。面向对象的程序设计思想也是在这种情况下产生的。

### 1.1.4 面向对象程序设计语言简介

面向对象程序设计语言是支持面向对象设计的语言工具,它必须支持抽象数据类型和继承性。面向对象的程序设计语言提供了特定的语言成分来保证和支持面向对象程序设计,并提供了继承性、多态性和动态链接机制,使得类和类库成为可重用的程序模块.

有人认为，用传统的程序设计语言加上特别的编程技术和约束规则就能在一定程度上实现抽象数据类型，而进行面向对象的程序设计。这种观点仅仅把面向对象编程看成是编程技巧，没有合理地使用面向对象的思想，因此，它并不是真正意义上的面向对象编程。

面向对象程序设计语言从产生到广泛应用经历了一个较长的发展过程。

1. 面向对象程序设计思想的出现

将面向对象的概念正式作为语言成分的是 20 世纪 60 年代开发出来的 Simula 语言。在 Simula 语言中引入了几个面向对象程序设计语言中的最重要的概念和特性，即，数据抽象的概念、类机构和继承性机制。Simula67 是 Simula 语言的一个具有代表性的版本，它从 Simula1 演化而来。虽然当今的面向对象程序设计的基本思想来源于 Simula 语言，但是现在 Simula 语言本身并不用于面向对象程序设计。

2. 面向对象程序设计语言的出现

Smalltalk 语言的问世标志着面向对象语言研究的开始。Smalltalk 是第一个真正的面向对象程序语言，它体现了纯粹的面向对象程序设计思想，是最纯的面向对象程序设计语言。Smalltalk 源于 Simula 语言，并受到 Lisp 语言的很大影响。Lisp 语言是 20 世纪 50 年代开发出来的一种语言，它以表处理为特色，是一种人工智能语言。

虽然 Smalltalk 是最纯的面向对象语言，且在大学里也很著名，但是由于它对计算平台的特殊要求，也使得它很难吸引公司和商用软件开发者，因此没有能得到广泛使用。

3. 面向对象程序设计语言的推广

在 20 世纪 80 年代，C 语言非常受欢迎，成为极其流行、应用很广的一种结构化语言。虽然 C 语言使用非常广泛，但是它并不是面向对象程序设计语言。在 80 年代早期，AT&T 贝尔实验室的 Bjarne Stroustrup 及其研究小组将 C 扩展成为 C++——一个支持面向对象概念的 C 语言。C++是一种混合型的面向对象程序设计语言，它在 C 语言的基础上增加了类、对象等机制，使之成为一种面向对象的语言，并且它提供了对传统 C 语言的向后兼容性。C++语言已经成为目前在实用中最有前途的面向对象程序设计语言。近年来，越来越多的面向对象语言被推向市场。目前最常用的面向对象的程序设计语言除了 C++还有 Visual C++和 JAVA 语言。

下面介绍目前最为常用的几种面向对象程序设计语言：

(1) C++语言简介。

大约在 1980 年，AT&T 的 Bjarne Stroustrup 设计了 C++语言，最初的目的是创建一种模拟语言，具有面向对象的程序设计特色。在当时，面向对象的编程还是一个比较新的理念。Stroustrup 没有从头开始设计新的语言，而是在 C 语言的基础上进行创建，他设计了一个转译程序，能够把 C++源程序处理成 C 源程序，这个程序称为 c front，c front 程序后来得到不断的移植和扩充。1985 年之后，C++开始在 AT&T 外面流行，ANSI 和 ISO 于 1989 年着手为 C++制定标准，1994 年出版了第一份非正式的草案，1998 年正式推出 C++的国际标准。

C++支持数据封装。支持数据封装就是支持数据抽象。在 C++中，类是支持数据封装的工具，对象则是数据封装的实现。面向过程的程序设计方法与面向对象的程序设计方法在对待数据和函数关系上是不同的，在面向对象的程序设计中，将数据和对该数据

进行合法操作的函数封装在一起作为一个类的定义，数据将被隐藏在封装体中，该封装体通过操作接口与外界交换信息。对象被说明具有一个给定类的变量，类类似于 C 语言中的结构，在 C 语言中可以定义结构，但这种结构中包含数据，而不包含函数。C++中的类是数据和函数的封装体。在 C++中，结构可作为一种特殊的类，它虽然可以包含函数，但是它没有私有或保护的成员。

C++类中包含私有、公有和保护成员。C++类中可定义三种不同访问控制权限的成员。一种是私有(Private)成员，只有在类中说明的函数才能访问该类的私有成员，而在该类外的函数不可以访问私有成员；另一种是公有(Public)成员，类外面也可访问公有成员，成为该类的接口；还有一种是保护(Protected)成员，这种成员只有该类的派生类可以访问，其余的在这个类外不能访问。

C++中通过发送消息来处理对象。C++中是通过向对象发送消息来处理对象的，每个对象根据所接收到的消息的性质来决定需要采取的行动，以响应这个消息。响应这些消息的是一系列的方法，方法是在类定义中用函数来定义的，使用一种类似于函数调用的机制把消息发送到一个对象上。

C++中允许友元破坏封装性。类中的私有成员一般是不允许该类外面的任何函数访问的，但是友元可以打破这条禁令，它可以访问该类的私有成员(包含数据成员和成员函数)。友元可以是在类外定义的函数，也可以是在类外定义的整个类，前者称友元函数，后者称为友元类。友元打破了类的封装性，它是 C++另一个面向对象的重要性。

C++允许函数名和运算符重载。C++支持多态性，C++允许一个相同的标识符或运算符代表多个不同实现的函数，这称为标识符或运算符的重载，用户可以根据需要定义标识符重载或运算符重载。

C++支持继承性。C++中可以允许单继承和多继承。一个类可以根据需要生成派生类。派生类继承了基类的所有方法，另外派生类自身还可以定义所需要的不包含在父类中的新方法。一个子类的每个对象包含有从父类那里继承来的数据成员以及自己所特有的数据成员。

C++支持动态联编。C++中可以定义虚函数，通过定义虚函数来支持动态联编。

C++支持异常处理。异常处理能够使程序更好地处理出错的情况，以一种有序的、有条理的、一致的方法来处理。异常处理允许程序中的一个部分去发现并且通知错误情况，另外一个部分去处理错误。

C++支持名字空间。在 C 中，外部和全局的标识符的作用域都是整个程序。它们对于应用程序、第三方函数库和编译器的系统库的所有函数而言都是可见的。在面对数百种函数库的时候，用户自己的程序中的函数名或者变量名很可能和库中的函数和变量重复。名字空间的出现就是为了避免这种情况，将库中的全局变量限制在一个名字空间之内。

(2) Java 语言简介。

Java 是由 SUN 公司开发，该语言有下面一些特点：简单、面向对象、分布式、解释执行、安全、体系结构中立、可移植、高性能、多线程以及动态性。Java 现在已经成为一种非常流行的语言。

Java 语言是一种面向对象的语言，它通过提供最基本的方法来完成指定的任务，只需理解一些基本的概念，就可以用它编写出适合于各种情况的应用程序。Java 略去了运算符重载、多重继承等模糊的概念，并且通过实现自动垃圾收集大大简化了程序设计者的内存管理工作。

Java 语言的设计集中于对象及其接口，它提供了简单的类机制以及动态的接口模型。对象中封装了它的状态变量以及相应的方法，实现了模块化和信息隐藏；而类则提供了一类对象的原型，并且通过继承机制，子类可以使用父类所提供的方法，实现了代码的复用。

Java 是面向网络的语言。通过它提供的类库可以处理 TCP/IP 协议，用户可以通过 URL 地址在网络上很方便地访问其他对象。

Java 在编译和运行程序时，都要对可能出现的问题进行检查，以消除错误的产生。它提供自动垃圾收集来进行内存管理，防止程序员在管理内存时容易产生的错误。通过集成的面向对象的例外处理机制，在编译时，Java 提示出可能出现但未被处理的例外，帮助程序员正确地进行选择以防止系统崩溃。另外，Java 在编译时还可捕获类型声明中的许多常见错误，防止动态运行时不匹配问题的出现。

Java 不支持指针，一切对内存的访问都必须通过对象的实例变量来实现，这样就防止程序员使用“特洛伊”木马等欺骗手段访问对象的私有成员，同时也避免了指针操作中容易产生的错误。

Java 解释器生成与体系结构无关的字节码指令，只要安装了 Java 运行时系统，Java 程序就可在任意的处理器上运行。这些字节码指令对应于 Java 虚拟机中的表示，Java 解释器得到字节码后，对它进行转换，使之能够在不同的平台运行。与平台无关的特性使 Java 程序可以方便地被移植到网络上的不同机器。同时，Java 的类库也实现了与不同平台的接口，使这些类库可以移植。另外，Java 编译器是由 Java 语言实现的，Java 运行时系统由标准 C 实现，这使得 Java 系统本身也具有可移植性。

(3) C# 语言简介。

C# 是微软开发的，基于微软下一代操作平台. NET，面向对象的全新的开发语言。C# 出现在 C++ 和 Java 的后面，它综合了两者的优点。由于微软宣布不支持 Java，C# 更像 Java，它们许多特性都是一样的，比如，把程序编译成中间代码，通过解释器运行，并且不支持多继承，具有垃圾回收机制等。

C# 语言由 C/C++ 演变而来。C# 是现代的编程语言，它简化了 C++ 在类、命名空间、方法重载和异常处理等领域的操作；摈弃了 C++ 的复杂性，使它更易用、更少出错。在 C# 中去掉了宏、模板和多重继承等机制。

C# 中没有指针，所有的工作都默认放在受管理的代码中，并且不允许进行直接存取内存等不安全的操作。在 C++ 中，有“::”、“.”、“->”操作符，它们用于命名空间、成员和引用。对于初学者来说，操作符至今仍是学习的一道难关。C# 不再使用更多的操作符，而仅使用单个操作符“.”。

C# 支持所有关键的面向对象的概念，完整的 C# 类模式构建在 NGWS 运行时的虚拟对象系统(VOS，Virtual Object System)的上层。对象模式只是基础的一部分，不再是

编程语言的一部分。使用C#编写类,程序员可以使用四种访问权限:private、protected、public和internal。

C#取消了不安全的类型转换。在C#中不能把一个整型强制转换成一个引用类型(如对象);但是将引用类型转换成整型时,C#验证这种转换是正确的。C#提供边界检查,避免了数组越界使用等情况出现。

C#并没有存在于一个封闭的世界中。它允许使用最先进的NGWS的通用语言规定(CLS,Common Language Specification)访问不同的API。CLS规定了一个标准,用于符合这种标准语言内部之间的操作。为了加强CLS的编译,C#编译器检测所有的公共出口编译,并在编译通不过时列出错误。C#允许程序员用C原型的API实现内部操作。可以从程序员的应用程序访问任何DLL中的入口点。用于访问原始API的功能称做平台调用服务(Platform Invocation Services)。

## 1.2 面向对象程序设计的特点

### 1.2.1 面向对象方法面临的问题

在面向对象方法中,寻找合适的对象来使问题空间直接映射到程序空间非常重要。面向对象的基本想法就是要把待构造的系统表示为对象的集合。这里所说的系统不仅包括程序和软件,而且也包括计算模型以及人工智能中的知识表示形式等。但本书主要讨论程序和软件。

面向对象方法主要需要考虑两个问题:一是怎样克服软件的复杂性;二是怎样将现实世界模型在计算机中自然地表示出来。

作为克服软件复杂性的手段,在面向对象中利用了对象的两个性质:第一,将密切相关的数据和过程封装起来定义为一个实体。第二,定义了一个实体后,即使不知道此实体的功能是怎样实现的,也能使用它们。

在将现实世界模型在计算机中自然地表示出来问题上,客观世界的问题都是由客观世界及其相互间的联系构成的,我们把客观世界的实体称之为问题空间的对象。对象由于问题的不同而不同,不同的对象之间的相互通信构成了完整的客观世界。从思维模型的角度,面向对象很自然地与客观世界相对应。

### 1.2.2 面向对象方法的分析

面向对象方法的分析,就是抽取和整理用户需求并建立问题域精确模型的过程。分析工作主要包含3项内容:理解、表达和验证。

通常,面向对象分析过程从分析陈述用户需求的文件开始。在分析需求陈述的过程中,系统分析员反复多次地与用户讨论、协商,还需通过调研来了解现有的类似系统。快速建立起一个可以在计算机上运行的原型系统,让用户试用并听取用户反馈意见,能更正确地提炼出用户的需求。

系统分析员在深入理解用户需求的基础上，抽象出目标系统的本质属性，并用模型准确地表示出来。分析模型应该成为对问题的精确而又简洁的表示。

面对复杂问题时，上述理解和建模工作通常不能一次就达到理想的效果。必须经常验证分析工作的正确性、完整性和有效性，如果发现问题则及时修正。理解和验证工作通常交替进行，反复迭代。

### 1.2.3　面向对象方法的实现

对系统分析完成后，就要进入系统的实现阶段，进而转入程序编写中。面向对象实现主要包括两项工作：

(1) 把面向对象分析的结果翻译成用某种面向对象程序语言书写的面向对象程序；

(2) 测试和调试编写出的面向对象程序。

为有效使用面向对象的程序方法，首先需要解决程序的结构设计问题。在程序设计过程中最重要的是抽象，也就是说，从现实世界中抽象出合理的对象结构。在面向对象思想中，抽象决定了对象的对外形象、内部结构以及处理对象的外部接口，其关键是处理对象的可见外部特征。封装是与抽象紧密联系的概念，它需要两个基本前提：首先，对象必须能够表现一个完备的概念，例如可以将一条道路的纵断面设计作为一个对象，它与外部的联系限于纵断面地面线、纵断面设计参数。当这些外部条件给定之后，对象可以独立计算任意桩号的设计高程、地面高程、设计高差、填挖面积等，也可以根据给定的比例和初始坐标计算设计线和地面线的图形坐标。第二，对象的私有性，例如上述示例中，对象的内部数据结构在外部是不可见的，其他程序员并不需要了解这种数据结构就可以使用对象的功能。对于对象的接口设计是十分重要的，它必须给出必要的访问渠道，同时必须尽可能地将内部细节隐藏起来。可以将接口看作是在屏蔽墙上打开的一些出入通道，通道过多了就失去了屏蔽墙存在的意义，通道过少又由于限制过强而行动不便。一般情况下，这种通道表现为一组接口函数，事实上也可以将一些变量作为对外开放的，但这并不是一种很好的方法。

### 1.2.4　面向对象方法的优点

面向对象方法学的出发点和基本原则，是尽可能模拟人类习惯的思维方式，是开发软件的方法与过程尽可能接近人类认识世界时解决问题的方法与过程，也就是使问题域与求解域在结构上尽可能一致。

面向对象的方法具有以下主要优点：

(1) 面向对象方法与人类所习惯的思维方法较为一致。面向过程的程序设计方法以算法为核心。面向过程程序设计站在计算机的立场上，以算法为核心，数据与过程相对分离，这不符合人类所习惯的思维方式。而面向对象的方法克服这方面的缺点，使用户可以直接使用某功能而不必知道其具体实现。

(2) 面向对象方法开发的软件可重用性和维护性好。由于面向对象将一些常用的功能创建为类，对数据和操作进行了封装，所以可以只定义一次而多次使用这些资源，因此它具有良好的可重用性。类是独立的，只要类的接口不改变，改变类功能的实现并不会影