普通高等教育“十一五”国家级规划教材

国家教学成果奖配套教材

高等学校计算机程序设计课程系列教材

Visual C++与面向对象程序设计教程（第3版）

冯博琴　主编

贾应智　姚全珠　吕　军　编

高等教育出版社
Higher Education Press

普通高等教育“十一五”国家级规划教材
国家教学成果奖配套教材
高等学校计算机程序设计课程系列教材

Visual C++与面向对象程序设计教程

Visual C++ yu Mianxiang Duixiang Chengxu Sheji Jiaocheng

（第3版）

冯博琴　主编
贾应智　姚全珠　吕　军　编

高等教育出版社·北京
HIGHER EDUCATION PRESS　BEIJING

内容提要

本书是普通高等教育“十一五”国家级规划教材，是国家教学成果奖配套教材。

本书是 Visual C ++ 的入门教材，面向程序设计“零起点”的学生，主要介绍如何使用 Visual C ++ 进行面向对象和可视化的程序设计。

本书在第 2 版的基础上，结合读者和教师反馈以及进一步的教学实践，对内容的选取、讲授方法、例题与习题等进行了全面地修订，从而更加适应该课程的教学要求。

本书主要内容共 12 章，分别是概论，数据和运算，控制结构，数组、字符串和结构体，函数，指针，类和对象，类的继承，多态性，异常处理和 I/O 流，C ++ 基本控件和数据库基础与应用。本书在讲授方式上注重结合应用开发实例、讲练结合、精讲多练，注重培养学生的程序设计和综合开发能力。

书中配有丰富的例题和习题，所有例题都已在 Visual C ++. NET 环境下调试并通过运行。

本书可作为高等学校计算机或相关专业的教材或参考书，也可供应用开发人员学习参考。

图书在版编目(CIP)数据

Visual C ++ 与面向对象程序设计教程/冯博琴主编. —3 版. —北京:高等教育出版社,2010. 2

ISBN 978 - 7 - 04 - 028112 - 5

Ⅰ. ① V… Ⅱ. ① 冯… Ⅲ. ① C 语言 - 程序设计 - 高等学校 - 教材 Ⅳ. ① TP312

中国版本图书馆 CIP 数据核字(2010)第 002671 号

策划编辑 耿 芳 **责任编辑** 许 可 **封面设计** 于文燕 **责任绘图** 尹 莉
版式设计 王艳红 **责任校对** 杨雪莲 **责任印制** 尤 静

出版发行	高等教育出版社	**购书热线**	010 - 58581118
社 址	北京市西城区德外大街 4 号	**咨询电话**	400 - 810 - 0598
邮政编码	100120	**网 址**	http://www.hep.edu.cn
总 机	010 - 58581000		http://www.hep.com.cn
		网上订购	http://www.landraco.com
经 销	蓝色畅想图书发行有限公司		http://www.landraco.com.cn
印 刷	潮河印业有限公司	**畅想教育**	http://www.widedu.com
		版 次	2000 年 7 月第 1 版
开 本	787×1092 1/16		2010 年 2 月第 3 版
印 张	19.25	**印 次**	2010 年 2 月第 1 次印刷
字 数	470 000	**定 价**	24.80 元

物料号 28112 - 00

第3版前言

本书的第2版是2003年出版的，通过对这几年教学工作的总结和读者反馈意见的归纳，我们深感需要对有关内容进行修订和调整。

在修订教材时，仍然明确本书的定位是作为Visual C++的入门教材，并且面向程序设计“零起点”的学生，目标还是第1版就明确的使学生掌握使用Visual C++设计应用程序的基本技能，以及能够编写、调试程序，而不是针对Windows编程的全面介绍。

为了和前两版的风格保持一致，每章的结构仍然主要由学习目标、授课内容、自学内容、调试技术、程序案例、上机练习题等栏目组成。新版的修订重点放在内容的更新、例题难度和跨度的调整上，各章修改工作采取了重写、合并充实或大部分保留等手段。

第3版主要进行了以下方面的修改和调整：

① 教学平台由原来的Visual Studio 6.0转变到了Visual C++.NET。

② 重写的第1章（概论）从软件工程的角度重点讲解了程序设计的基本概念，包括软件开发的范型、软件开发方法、算法的效率评价，最后是开发环境的使用。

③ 将第2版的第3章（数据类型）和第4章（表达式）调整合并为新的一章，即第2章（数据和运算），作为程序设计的基础。

④ 对第2版的第3~9章主要进行了内容的调整、重写和充实。

⑤ 第2版中第10~16章，内容分别是Windows编程、文档/视图结构、图形设备接口和资源、对话框、控件、文档读/写与打印、多文档界面程序。这几章的内容较多，在目前各学校对许多课程都在减少学时的情况下，很难有充分的时间完成和掌握，所以对这些章节的内容分别进行了删除、压缩，重新改写后形成现在的第10章（异常处理和I/O流）、第11章（C++基本控件）和第12章（数据库基础与应用）。

经过以上修改后，全书内容更加紧凑，目标也更明确：既突出了可视化程序设计的基础——控件，也增加了重要的应用之一——数据库的访问，所以，内容更加实用，也更易于入门。

⑥ 修订后的教材中有两类例题：一类是在“授课内容”栏目中新增的例题，主要是围绕该章节的语法讲解；另一类是放在“程序案例”栏目中的，这是综合例子，主要是第2版中“程序设计实例”栏目中的例子，所有程序均在.NET下调试并通过运行。

第3版的修订工作由冯博琴教授负责，并组织课程教学一线的教师共同完成。参加修订工作的还有姚全珠（第1、11、12章）和贾应智（第2~10章）。

本书是在前两版的基础上修订完成的，因此，书中也凝聚了参与前两版工作的多位教师的辛勤劳动，在此深表谢意。

由于编者水平有限，编写仓促，书中不足之处在所难免，希望读者批评指正。

编者于西安交通大学

2009年9月

第 2 版前言

本书自 2000 年 7 月面市，转眼已近三年了。在这三年里，仅在本校先后就有十余位教师、近万名学生参与了本课程的教学实践。在教学中，授课教师多次开展教学法活动，互相听课、研讨，并以各种形式听取学生的意见和建议。我们建立了一整套教学体系，并不断完善教学环节，包括多媒体授课、网络视频课堂、电子答疑、电子作业提交与批改以及上机编程的考核方法，以图贯彻“精讲多练”的教学方针。

在教学实践和教学法交流活动中，授课教师和学生对本书提出了大量建议。除了指出原书中存在的错误外，这些建议集中反映在学习难度上。作为 Visual C ++ 的入门教科书，并且面向“零起点”的学生，本书的目标在第一版就明确定位为“使学生掌握使用 Visual C ++ 设计应用程序的基本技能”，以及能够编写、调试程序，而不是对 Windows 编程的全面介绍。为了能在一本几十个课时的教材中涵盖 Visual C ++ 的基本技术，对于 C、C ++ 和 Windows 编程技术相关内容的选择一直是我们反复推敲的重点。经过三年的积累，我们深感需要对有关内容进行修订和调整，以反映这些来自教学一线的需求。

随着微软 .NET 的推广，有的教师建议将教学环境过渡到 .NET 框架，以适应技术发展的潮流。经过多次讨论，我们认为语言开发环境的变化是快速的，而语言本身在相当时间内会保持相对稳定。作为入门教材，我们应更注重对学生基本程序设计能力的培养，而不过分依赖于开发环境。为此，我们在新版中没有引入最新的开发环境，而把重点放在内容的取舍和例题难度、跨度的调整上。

除了修正原书中的错误外，第二版主要进行了以下方面的修改和调整：

(1) 加大了 C ++ 部分内容的分量，由原来的两章改为三章，并增加了相应例题，以期强化初学者面向对象程序设计的能力。为保持总课时数，Windows 编程部分相应地压缩了一章。

(2) 降低学习难度，删除了原书中的“Win32 应用程序”编程模式、使用非模式对话框编程方法，并改写了相关例题，以单文档 (SDI) 编程模式为主线介绍 Windows 编程，以使学生更关注于程序设计本身，弱化对开发环境的学习。

(3) 增加部分贴近实际应用的例题，如应用数值分析及图示编程，为学生在后续课程及以后工作中应用编程技术打下良好基础。

(4) 对例题和概念的讲解进行了全面的润色，并从第十章开始，采用 step-by-step 的方法指导学生在向导生成的程序基础上进行编程，更有利于读者自学。原书中编程技巧较高的例题放在附录中，以满足学习进度较快的学生的要求。

第一版的主要作者刘璐放教授已经远在加拿大，第二版的修订工作是冯博琴教授应高等教育出版社要求，组织西安交通大学计算机教学实验中心多位在课程教学一线的教师共同完成的，这些教师中有些人还同时承担其他语言的教学工作。参加修订的有杨琦 (第 1 ~ 7 章)，

仇国巍（第8、9章），吕军（第10、11、15、16章），朱丹军（第12章），薛涛（第13、14章），崔舒宁（本书的部分例题）。本书由罗建军、杨琦统稿，全书由冯博琴教授主审。刘路放教授对本书的修改提出了重要的建议，在此致以诚挚的感谢。其他授课教师在百忙之中也对本书提出许多有益的建议，在此一并致谢。特别的致谢给予那些在调查问卷、课堂调查以及通过BBS、E-mail向教师提出建议的广大学生。

由于作者学识浅陋，编写时间仓促，书中错误在所难免，希望读者不吝赐教。

编者于西安交通大学

2003年6月

第 1 版前言

随着 Windows 操作系统的崛起，由传统的面向控制台的字符软件开发向面向窗口的可视化编程转化已成为必然趋势。而 Visual C ++ 正是 Windows 环境下最强大、最流行的程序设计语言之一。

Visual C ++ 支持面向对象的程序设计方法（Object-Oriented Programming，OOP），支持 MFC（Microsoft Foundation Class）类库编程，有强大的集成开发环境 Developer Studio（其中包括了程序自动生成向导 AppWizard、类向导 Class Wizard 和各种资源编辑器，以及功能强大的调试器等可视化和自动编程辅导工具）。Visual C ++ 可用来开发各种类型、不同规模和复杂程度的应用程序，开发效率很高，生成的应用软件代码品质优良。这一切都使得 Visual C ++ 成为许多专业程序开发人员的首选。

然而，Visual C ++ 一向有“难学”的名声，许多初学者视学习 Visual C ++ 为畏途。究其原因，一方面是 Visual C ++（包括 MFC 类库）的规模庞大，结构复杂，难于理出一条循序渐进的学习路线；另一方面是其 AppWizard 自动生成的程序专业化程度高，代码量大，结构复杂，以其为基础编写的例题难于为初学者理解和掌握。因此，目前的 Visual C ++ 教科书多是为已有 C 语言或 C ++ 语言编程基础的人准备的，起点较高。

本书是 Visual C ++ 入门教科书，适用于本科类计算机及相关专业学生程序设计能力的培养。为了克服上述困难，使基础不高的初学者也能很快地掌握程序设计方法，我们在确定教学目标、设计教学模式、编写教程内容等方面进行了一系列革新探索，以现代教育理论为指导，多媒体教学手段为基础，提出了“精讲多练”的教学模式。使用“精讲多练”模式进行 Visual C ++ 这类程序设计语言课程的教学，效果很好。

本教程的目标是使学生掌握使用 Visual C ++ 设计应用程序的基本技能，了解面向对象的和结构化的程序设计方法，能够编写、调试和运行实用、规范、可读性好的 Visual C ++ 程序。不像其他 Visual C ++ 教材需要学习者具有一定的程序设计基础（如学过 C 语言或 C ++ 语言），本书“从零开始”，不要求学生有程序设计方面的先修课程。但在学习本课程时，学生最好对计算机的使用有一定了解（了解 Windows 的使用，具有键盘操作和文件处理的基础）。

我们在设计本教程内容时，以面向对象的和结构化的程序设计方法思想贯穿全书，并以大量篇幅介绍了 Visual C ++ 程序的调试技术和一些典型应用程序的设计思路，其中有些是作者在长期的编程和教学实践中摸索和总结出来的心得。

本教程共分 16 章，分别对应 16 个教学重点。这 16 个教学重点又可分为两组：前 8 章为一组，处理 C ++ 的基本内容，包括控制结构、基本数据类型、表达式、函数，指针和引用，以及类与对象的基本概念和封装、继承和多态性等面向对象程序设计的基础理论。在学习了这些内容之后，学生应能编写、调试和运行一般规模和难度的控制台应用程序（如数值计算类程序），并对面向对象的和结构化的程序设计方法有所了解，为编写较大规模的应用程序打下基础。后 8 章处理 Windows 编程技术，包括消息传递机制、MFC 应用程序框架、设备环境、

资源、文档/视图结构、对话框和控件等。在这一部分中，强调对基本概念的理解和掌握，以及在理解和掌握的基础上编写具有较复杂的窗口界面的 Windows 应用程序的能力。

为了便于教学，每章均按以下主题进行组织：

教学目标和学习要求 本书的特点是“精讲多练”，因此为教师和学生规定明确的教学和学习目标是非常重要的。

授课内容 是建议教师课堂讲授的内容。一般来说，授课内容是本章所有教学内容的“纲”，起着联系本章所有项目的作用。授课内容部分的分量按两学时组织。第 1 章的授课内容分量略轻，这是因为在第 1 章的授课时间中还应划分出部分时间用于介绍编辑、调试和运行应用程序项目的基本步骤。

自学内容 “自学内容”和“授课内容”部分一起组成了一个章的基本教学内容。这部分内容通常都是“授课内容”的延伸和继续，由学生在课外时间自学。必须强调的是自学部分并非不重要，也不能省略。一般来说，教师应在授课时间中抽出 5～10 分钟对自学内容略作导引，以便利学生自学。

调试技术 介绍 Developer Studio 集成开发环境的使用方法，以及如何调试、连接和运行 Visual C++ 应用程序项目。强调编程实践是本书的重要特色。第 1 章的调试技术中的部分内容可以在授课时间讲授，其他章的调试技术一般由学生自学，同时也可以作为学生上机的实验指导书。辅导教师在带学生上机时应对这些内容进行现场辅导。

程序设计举例 为了补充授课内容和自学内容部分的例题，我们设置了程序设计举例栏目。本栏目所有例题均与本章的授课、自学或调试技术等部分的内容相关，是学生学习复习本章的重要参考资料。

上机练习题 每章均配有若干上机练习题目，供学生上机练习。这些练习题目均为程序设计题目，传统的做法是先编程，再上机。由于 C++ 的特点，也可以在写出较详细的伪代码程序之后直接上机。“精讲多练”式教学方法的基本特点是上机时数较多，所以这部分的习题工作量较大，因此在上机时数不足的情况下可以酌情选做若干题目。

为了保证教学效果，在条件许可的情况下最好采用直接在计算机房进行的联机电化教学。在这种情况下，每个教学单元（即每章）可使用连续的 4 课时，先由教师讲解授课部分并对自学部分和调试技术等内容进行简短的指导（共 2 学时），然后学生即可在教师指导下上机练习（2 学时）。此外，如果能够提供一定数量的课外机时（如 20～30 小时）则更好。

近年来，我中心在计算机基础教育的理论和实践等方面进行了一系列探索和革新，其成果（“精讲多练”的教学模式是其中之一）荣获了 1997 年度国家级教学成果一等奖。这些成果都是在冯博琴教授的领导下完成的，本课程的建设也不例外。本教程的构思和编写得到了冯博琴教授的多方指导，并由他审核了书稿，在此向冯老师表示深深的谢意。在本书编写过程中，曾与李波、罗建军、卫颜俊、杨琦、吕军和张伟诸同事进行了多次交流，受益匪浅。以上同事还提供了一些有用的材料；杨琦同志为本书绘制了部分插图，在此一并表示感谢。由于作者学识浅陋，编写时间仓促，书中错误在所难免。希望读者不吝指教。

编者于西安交通大学

2000 年 4 月

目　录

第1章 概 论

学习目标

通过本章学习，要求了解常用的软件工程范型和常见的软件开发方法，理解面向对象程序设计的基本概念，学会在 Visual Studio 环境下编写简单程序。

授课内容

本章首先介绍软件开发的范型和方法，然后介绍算法效率的评价方法，最后介绍有关 C ++ 的基本概念及编程环境。

随着计算机科学与技术的发展，计算机应用已步入社会的各个角落。要想用计算机解决实际问题，就需要人与计算机进行交互。首先，人将需解决的问题告诉计算机，并说明要解决这一问题的方法、步骤，然后，计算机才能按照“预定”的策略解决问题。因此，用计算机解决问题时涉及两方面的问题：一方面是解决这一问题的方法、策略，即算法；另一方面是如何把算法告诉计算机，即人机交互语言。要想制定一个高效的解决问题的算法涉及软件开发方法学的问题，这属于软件工程的范畴，本章旨在对这一问题做简单的介绍，使读者对这一问题有一个初步的了解。本书的重点在于介绍人机交互的语言。作为人机交互的语言有机器语言、汇编语言和高级语言。机器语言是由 0、1 序列构成的，例如在 IBM PC 所执行的指令系统中，指令：

0 0 0 0 0 0 1 0 1 1 0 0 1 1 1 1　　(02CFH)

操作码　　操作数

表示把计算机中寄存器 CL 和 BH 的内容相加，并把结果存入 CL 中，同样的指令：

0 0 0 0 0 0 1 0 1 1 1 1 1 1 0 0 1　　(02F9H)

则表示把寄存器 BH 和 CL 的内容相加，并把结果存入 BH 中。机器语言便于在机器中表示和运算，但对于用户来说难读、难记，出了错难于修改，因而就有了汇编语言。汇编语言用英文单词作为助记符，用面向机器的方式来表达操作。例如，上述两条指令用汇编语言可表示为：

```
ADD CL,BH      ;表示把寄存器 CL 和 BH 的内容相加，结果存入 CL 中
ADD BH,CL      ;表示把寄存器 BH 和 CL 的内容相加，结果存入 BH 中
```

显然汇编语言比机器语言易读、易懂，也容易修改。但编程者需要了解机器结构，知道有哪些寄存器，机器字长是多少，有哪些寻址方式等，这给普通用户带来不便；这种语言与人类的自然

语言仍有较大差距。因此,便有了高级语言的诞生。目前已有近百种高级语言,如 Pascal、FORTRAN、BASIC、C、COBOL 等语言。学会了高级语言便能方便地与计算机进行沟通,可以把解决问题的"算法"通过高级语言以程序的方式向计算机描述。为了高效地描述解决问题的方法步骤,在本章中首先介绍软件开发的范型、软件开发的方法以及 C ++ 语言的基础知识。

1.1 软件开发的范型

要想迅速地开发出一个高效的软件系统,以实现某一特定的需求,则开发此软件时应遵循一定的方法、步骤。一个软件工程师或一组工程师必须制定一个开发策略。这个策略包括如何构造软件的技术(方法),支持软件分析、设计和实现的语言,为方法和语言提供自动化或半自动化支持的工具,以及把方法、语言和工具连接在一起的工具。这个策略常常被称为过程模型或软件工程范型。

1.1.1 瀑布模型

软件工程的传统途径是生存周期方法学(线性顺序模型)。它是从时间角度对软件开发和维护的复杂问题进行分解,把软件生存的漫长周期依次划分为若干个阶段,每个阶段有相对独立的任务,然后逐步完成每个阶段的任务。以控制开发工作的复杂性,通过有限的确定步骤,把用户需求从抽象的逻辑概念逐步转化为具体的物理实现。

软件生存周期方法把软件开发划分为三大阶段,每个阶段又可进一步细分。其具体内容划分如图 1.1 所示。

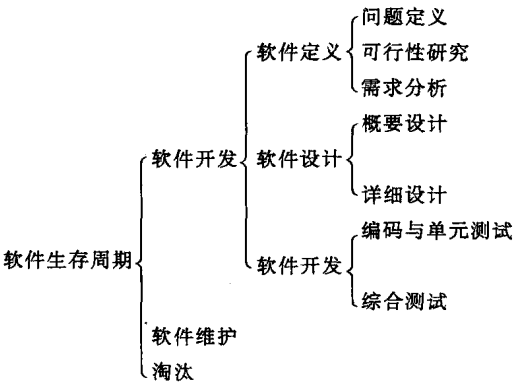

图 1.1 生存周期方法中软件开发阶段的划分

这种传统的软件生存周期方法(也称为软件开发的瀑布模型)中每个阶段完成一定的工作,且有一个里程碑。例如,在问题定义阶段的工作就是"要解决的问题是什么",其成果为关于规模和目标的报告书;在可行性研究阶段的工作是分析"此问题有可行解吗",其成果为系统的高层逻辑模型(数据流图)和成本效益分析报告。

其缺点在于:对于初次进行软件开发的人来说,最容易犯的错误就是急于求成,不愿意仔细地去做前期工作,过早地动手编写程序。结果经常出现所开发的程序在功能或者性能上满足不了用户的要求;或者所开发的系统对运行环境要求很高,无法投入实际运行;或者所开发的系统文档不规范,程序的可读性差,模块的独立性差等缺点。一旦用户需求、支撑平台或维护人员发

生变化，系统便无法修改，导致大量返工或过早地结束生存周期。

上述问题都是真实存在的。但不管怎样，传统的生存周期模型在软件工程中仍占有重要的位置。它提供了一个模板，使分析、设计、编码、测试和维护的方法可以在该模板的指导下展开。传统的生存周期模型仍然是软件工程中应用最广泛的过程模型。虽然它确实有不少缺陷，但很显然，它比软件开发中随意的状态要好得多。

1.1.2 原型模型

产生原型化方法的原因很多，主要是随着系统开发经验的增多，软件开发人员也发现并非所有的需求都能够预先定义，而反复修改是不可避免的。当然能够采用原型化方法还是因为开发工具的快速发展，比如用 VB、PorwerBuider、Delphi、C ++ 等工具，就可以迅速地开发出一个让用户“看得见、摸得着”的系统框架，这样，对于计算机不是很熟悉的用户就可以根据这个样板提出自己的需求。主要有以下两种实现原型法的途径。

1. 抛弃原型法

其目的是要评价目标系统的某些特性，以便更准确地定义需求，使用之后就把这种原型抛弃。

2. 演化原型法

演化原型法是一个多次迭代的过程，每次迭代都由以下几个阶段组成：

① 确定用户需求。

② 开发原始模型（快速设计、建立原型）。

③ 征求用户对初始原型的改进意见。

④ 修改原型。

这一过程如图 1.2 所示。

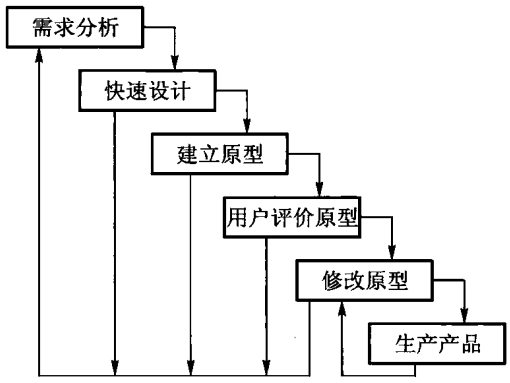

图 1.2 原型开发模型

原型化开发比较适合于用户需求不明确、业务不确定、需求经常变化的情况。当系统规模不是很大也不太复杂时采用该方法是比较好的。原型法在使用过程中要解决的问题如下：

① 用户看到的是一个可运行的软件版本，但不知道这个原型是临时搭建的，也不知道为了使其尽快运行，还没有考虑软件的整体质量及今后的可维护性问题。用户往往会认为此系统只

要经过小的调整即可投入使用，软件开发管理者可能也就不坚持质量原则了。

② 为使原型系统尽快投入运行，开发人员经常采用一些折中的解决方法。如使用一些不当的操作系统或编程语言，仅仅因为他们对此比较熟悉；采用一些效率不高的算法。但经过一段时间，开发人员可能熟悉了自己的这些选择，却忘记它们不适合的原因。这些不明智的选择很可能就成了系统集成的一部分。

3. 演化软件过程模型

人们已经越来越认识到软件就像所有复杂系统一样要经过一段时间的演化。业务和产品需求随着开发的发展常常发生改变，想找到最终产品的一条直线路径是不可能的；紧迫的市场期限使开发人员难以完成一个完善的软件产品，但可以提交一个有限的版本以对付竞争或商业的压力；又要核心产品或系统需求能够被很好地理解，而产品或系统的细节部分可以进一步定义。在这些情况及其他类似情况下，软件工程师需要一个过程模型，以便明确设计，同时又能适应随时间演化的产品的开发。

线性顺序模型支持直线开发，本质上，瀑布方法是假设当线性序列完成之后就能够交付一个完善的系统。原型模型的目的是帮助用户（或开发者）理解需求。总体上讲，它并不是交付一个最终产品系统。而软件的变化特征在这些传统的软件工程范型中都没有加以考虑。

演化模型是利用一种迭代的思想方法。它的特征是使软件工程师渐进地开发，逐步完善软件版本。演化模型可细分为以下几种类型。

（1）增量模型

增量模型融合了线性顺序模型的基本成分（重复的应用）和原形的迭代特征。如图 1.3 所示，增量模型采用随着日程时间的进展而交错的线性序列。每一个线性序列产生软件的一个可发布的“增量”。例如，使用增量范型开发的字处理软件，可能在第 1 个增量中发布基本的文件管理、编辑和文档生成功能；在第 2 个增量中发布更加完善的编辑和文档生成能力；第 3 个增量实现拼写和文法检查功能；第 4 个增量完成高级的页面布局功能。

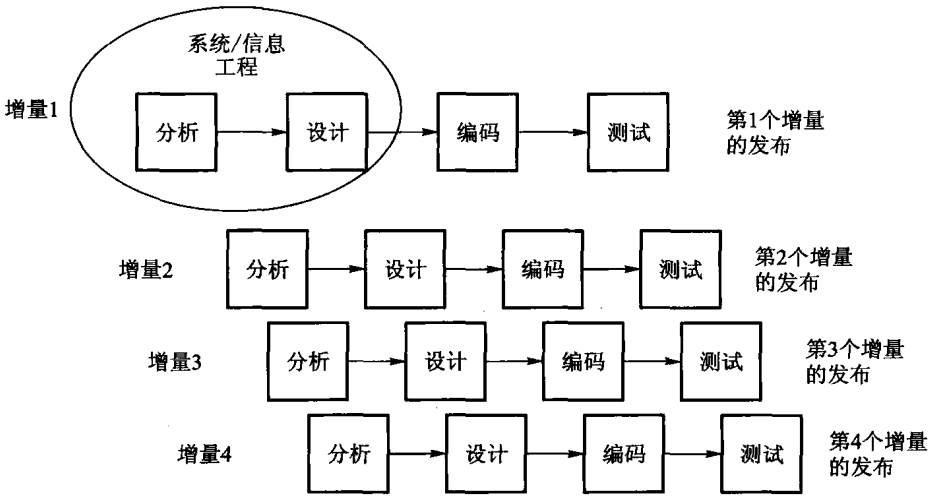

图 1.3　增量模型

增量过程模型,像原型和其他演化方法一样,具有迭代的特征。但与原型不同之处是,增量模型强调每一个增量均发布一个可操作产品。早期的增量是最终产品的“可拆卸”版本,但它们确实提供了为用户服务的功能,并且提供了给用户评估的平台。

增量开发是很有用的,尤其是当配备的人员不能在为该项目设定的市场期限之前实现一个完全的版本时。早期的增量可以由较少的人员实现。如果核心产品很受欢迎,可以增加新的人手(如果需要)实现下一个增量。此外,增量能够有计划地管理技术风险,例如,系统的一个重要部分需要使用正在开发的且发布时间尚未确定的新硬件,有可能计划在早期的增量中避免使用该硬件。这样,就可以先发布部分功能给用户,以缩短系统的问世时间。

增量模型的缺点是:由于各个构件是逐渐并入已有的软件体系结构中,所以加入构件必须不破坏已构造好的系统部分,这需要软件具备开放式的体系结构;在开发过程中,需求的变化是不可避免的。增量模型的灵活性可以使其适应这种变化的能力大大优于瀑布模型和快速原型模型,但也很容易退化为边做边改模型,从而使软件过程的控制失去整体性。

(2) 螺旋模型

螺旋模型最早是由 Boehm 提出来的,是一个演化软件过程模型,它将原型的迭代特征与线性顺序模型中控制和系统化的方面结合起来,使软件的增量版本的快速开发成为可能。在螺旋模型中,软件开发是一系列的增量发布。在早期的迭代中,发布的增量可能是一个纸上的模型或原型;在以后的迭代中,被开发系统更加完善的版本才逐步产生。

螺旋模型被划分为若干框架活动,也称任务区域。一般情况下,有 3 ~ 6 个任务区域,图 1.4 表示包含 6 个任务区域的螺旋模型:

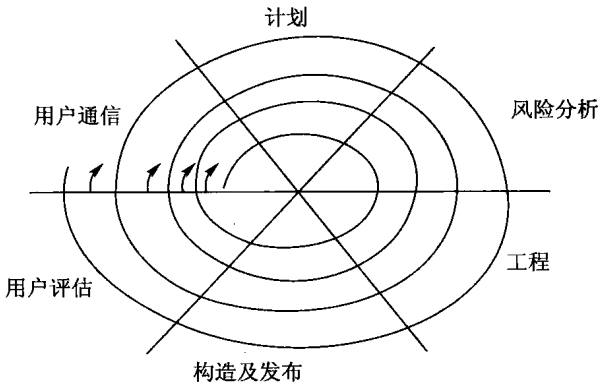

图 1.4　一个典型的螺旋模型

- 用户通信:建立开发者和用户之间有效通信所需要的任务。
- 计划:定义资源、进度及其他相关项目信息所需要的任务。
- 风险分析:评估技术及管理的风险所需要的任务。
- 工程:建立应用的一个或多个表示所需要的任务。
- 构造及发布:构造、测试、安装和提供用户支持(如文档及培训)所需要的任务。
- 用户评估:基于在工程阶段产生的或在安装阶段实现的软件表示的评估,获得用户反馈所需要的任务。

每一个区域均含有一系列适应待开发项目的特点的工作任务。对于较小的项目,工作任务