实例讲解
实训强化
培养技能
面向就业

全国高等职业教育计算机类规划教材·实例与实训教程系列

软件工程基础与实训教程

◎杜文洁　白　萍　主编

- 紧扣教学，重点突出，教学内容实用
- 案例驱动，按项目运作所需的知识体系结构设置内容
- 突出实训，重在培养学生的专业能力和实践能力
- 教材配套齐全，提供相关教学资源

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

全国高等职业教育计算机类规划教材·实例与实训教程系列

软件工程基础与实训教程

杜文洁　白　萍　主　编

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

本书从结构化方法和面向对象方法两方面讲述软件工程的基本概念、原理和方法，系统地介绍了目前较成熟的、广泛使用的软件工程技术。本书内容包括：软件工程概述、软件需求分析、概要设计、详细设计、程序编码、软件测试、软件维护、面向对象技术、软件工程管理、综合实例——网上书店系统、实训指导。

本书采用案例式教学，既注重基本知识的表述，又注重内容的先进性、系统性和实用性，力求反映软件工程技术发展的最新成果。本书理论与实践相结合，内容翔实，实用性强，可操作性强。

本书可作为高等院校、高职高专院校及计算机相关专业的教材，也可作为社会软件工程培训的教材，同时还可供从事软件开发及应用的程序员参考。

图书在版编目（CIP）数据

软件工程基础与实训教程/杜文洁，白萍主编. —北京：电子工业出版社，2010.9
全国高等职业教育计算机类规划教材·实例与实训教程系列
ISBN 978-7-121-11770-1

Ⅰ. ①软… Ⅱ. ①杜… ②白… Ⅲ. ①软件工程－高等学校：技术学校－教材 Ⅳ. ①TP311.5

中国版本图书馆 CIP 数据核字（2010）第 174295 号

策划编辑：程超群
责任编辑：程超群
印　　刷：北京市顺义兴华印刷厂
装　　订：三河市双峰印刷装订有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
开　　本：787×1 092　1/16　印张：12.25　字数：310.4 千字
印　　次：2010 年 9 月第 1 次印刷
印　　数：4 000 册　　定价：22.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

软件工程是研究如何用工程化的理论、方法和技术，来研究和指导软件开发的一门交叉学科。随着软件应用日益广泛，软件规模日益扩大，软件工程技术已成为专业软件人员必须掌握的技术。因此我们依据高职高专软件工程课程教学大纲所规定的教学要求编写本书，把多年软件工程教学经验和教学实践成果融入本教材中，在内容分布上充分考虑理论与实践相结合的原则。

软件工程是一门理论与实践并重的课程。本书在讲述软件工程的基本概念、原理和方法的基础上，详细而全面地介绍了可以实际用于软件开发实践的各种技能。旨在通过本书的学习，学生不仅能对软件工程的原理有所认识，而且还能具备实际开发软件的各种技能。本书最后一章为实训指导，通过实训内容可以使学生掌握各种软件工程工具。

本书选材注意把握高职高专学生的专业知识背景与接受能力，以案例教学的方法激发学生的学习兴趣。在教材编写上，力求做到结合实际、注重应用、便于教学，注意内容的新颖、实用和系统性。在结构安排上，深入阐述软件工程的基础理论知识，循序渐进，做到理论和实际相结合。

本书共分为 11 章，内容涉及软件工程的基本原理和概念、软件开发生命周期的各个阶段、项目管理的相关内容。第 1 章软件工程概述，第 2 章软件需求分析，第 3 章概要设计，第 4 章详细设计，第 5 章程序编码，第 6 章软件测试，第 7 章软件维护，第 8 章面向对象技术，第 9 章软件工程管理，第 10 章综合实例——网上书店系统，第 11 章实训指导。

本书由杜文洁、白萍担任主编，马岩担任副主编。杜文洁负责全书的策划、修改、补充和统稿工作。各章编写分工如下：杜文洁编写第 1 章、第 5 章，白萍编写第 2 章、第 3 章、第 4 章、第 6 章、第 8 章、第 10 章、第 11 章，马岩编写第 7 章、第 9 章，王宗玉、徐春雨、刘明国、丛国凤、王志阳、郝蔷、李虹等老师也参加了部分内容的编写。

由于水平和时间有限，书中难免存在错误和不足之处，敬请读者批评指正。

编　者

2010 年 5 月

目　录

第1章　软件工程概述

本章目标

- 了解软件的特点及解决软件危机的方法。
- 掌握软件工程的定义。
- 了解软件的开发方法。
- 掌握软件生命周期。
- 了解软件开发模型。

1.1　软件危机

迄今为止，计算机系统已经经历了四个不同的发展阶段，但是在此过程中，人们仍然没有彻底摆脱“软件危机”的困扰，软件已经成为限制计算机系统发展的瓶颈。

1.1.1　软件的定义及特点

随着计算机技术的发展，人们逐渐认识到：高质量的软件会使计算机系统的功能和效率大大地提高。高质量、多功能的软件使得计算机的应用从单一的科学计算扩展到数据处理、实时控制等多个领域。随着计算机应用的逐渐普及，软件变得越来越复杂，规模也越来越大，人们开始意识到软件并不仅仅是程序。目前对软件比较公认的解释为：软件在计算机系统中是与硬件相互依存的，它包括程序、相关数据及其说明文档。其中程序是按照事先设计的功能和性能要求而执行的指令序列；数据是程序运行过程中被处理的对象；文档是与程序开发、维护和使用有关的各种图文资料。

软件有如下特点：

（1）软件是一种逻辑实体，具有抽象性。人们无法看到软件本身的形态，只能通过观察、分析、思考、判断等方式才能了解它的功能和性能。

（2）软件在使用过程中没有磨损的问题。软件在使用过程中不会因为磨损而老化，但为了适应硬件和系统环境以及需求的变化，可能要不断地对软件的设计和编码进行改动，这些改动会不可避免地引入错误，导致软件失效率升高。

（3）软件一旦研制成功，就可以重复制造，其生产过程就变成复制过程。

（4）软件的开发和运行必须依赖于特定的计算机系统环境。为了适应计算机硬件和系统环境的不断升级，软件也需要不断地进行维护和更新。为了减少这种依赖性，在软件的开发过程中提出了软件的可移植性。

（5）软件开发至今尚未摆脱手工方式。虽然市场上出现了一些辅助开发工具，但是最终的核心代码仍然要程序员手工编写和组织。随着科学技术的进步，人们对计算机的依赖程度越来越高，对软件的需求数量和规模也越来越大，所以软件开发人员的工作压力也越来越大。

1.1.2 软件危机的产生原因

软件危机是指落后的软件生产方式无法满足迅速增长的计算机软件需求，从而导致软件开发与维护过程中出现一系列严重问题的现象。软件危机爆发于20世纪60年代中期，软件规模的扩大、复杂性的增加、功能的增强等使得高质量的软件开发变得越来越困难。在软件的开发过程中，经常会出现不能按时完成任务、产品质量得不到保证、工作效率低下和开发经费严重超支等情况。

软件危机的出现及其日益严重的趋势，促使人们去探究其产生的根本原因。最终，人们发现软件危机的产生有两方面的因素：一方面与软件本身的抽象性和复杂性有关，这是客观原因；另一方面则与软件开发和维护过程中使用的技术和方法有关，这是主观原因。根本原因是软件开发过程不成熟，主要表现为：

（1）忽视软件开发前期的调研和分析工作。只有用户最了解他们自己的需要，但许多用户开始并不能正确、具体地描述他们的要求，这要求软件开发人员需要做大量而且深入细致的调查工作，反复地与用户进行交流，才能全面、真实、具体地了解用户的需求。实践证明，在没有准确、完整地了解用户需求和定义问题的前提下就急于编程，是导致软件开发工程失败的主要原因之一。

（2）开发过程缺乏统一的规范。规模日益增大的软件往往需要很多人合作开发，这就需要用户和软件开发人员之间，软件开发人员之间进行及时、有效的沟通，以消除开发过程中对问题理解的差异，防止后续错误的发生。然而多年的“手工作坊”式的工作环境，使得软件开发人员更习惯于独自工作。开发过程中所采用的技术没有一个统一的规范和标准，势必会妨碍整个项目开发的团队合作，这也是导致软件危机的一个重要原因。

（3）软件开发管理困难而复杂。大型软件项目需要规模较大的团队来共同完成，但多数管理人员缺乏大型软件系统的管理经验，多数软件开发人员又缺乏管理方面的经验。两者不能进行及时、准确的信息交流，甚至还会产生误解。软件开发人员不能独立、有效地处理软件开发过程中的各种工作流程和工作关系，因此容易产生疏漏和错误。另外，开发过程中忽视撰写和保存相关文档的工作，使文档缺乏一致性和完整性，从而导致开发者失去工作的基础、管理者失去管理的依据。

（4）没有完善的质量保证体系。建立完善的质量保证体系需要有严格的评审制度，同时还需要有科学的软件测试技术及质量维护技术。软件的质量得不到保证，开发出来的软件产品往往不能满足需求，甚至可能需要花费大量的时间、资金和精力去修复软件的缺陷，从而导致软件质量下降和开发预算超支等后果。

1.1.3 解决软件危机的方法

为了摆脱软件危机造成的困境，北大西洋公约组织（NATO）的科学委员会于1968年在联邦德国召开的有关研讨会上，第一次提出了“软件工程”（Software Engineering）的概念。从此，软件工程作为一门新兴的工程学科诞生了。

软件工程的主要思想是运用工程学的基本原理和方法来组织和管理软件生产。解决软件危机，既要有技术措施（包括方法和工具），又要有必要的组织管理措施。先进的方法和工具可以提高软件开发和维护的效率，更为软件质量提供保证；有效地组织管理措施可以评价、控制、管理整个开发流程，从而保证软件开发能够顺利有效地完成。软件工程正是从技术和管理两个方面，研究如何更好地发展计算机软件技术。

1.2 软件工程

软件工程是一门指导计算机软件开发和维护的工程学科。它运用工程学中的概念、原理、方法和技术来指导软件的开发和维护工作。

1.2.1 软件工程的定义及目标

关于软件工程，不同的学者和组织机构给出了不同的定义。

1993 年，电气电子工程师学会（Institute of Electrical and Electronics Engineers，IEEE）给出的定义是：软件工程是将系统化的、严格约束的、可量化的方法，应用于软件的开发、运行和维护过程，也就是将工程化应用于软件的开发和管理之中。

2001 年，软件工程大师 Roger S.Pressman 对软件工程的定义是：软件工程是一个过程、一组方法和一系列工具。

目前比较认可的一种定义是：软件工程就是采用工程的概念、原理、技术和方法来开发与维护软件，把经过时间考验而证明是正确的管理方法和最先进的软件开发技术结合起来，运用到软件的开发和维护过程中。

软件工程旨在开发满足用户需求、能及时交付、不超过预算和无故障的软件，其主要目标如下：

（1）合理预算开发成本，付出较低的开发费用；

（2）实现预期的软件功能，达到较好的软件性能，满足用户的需求；

（3）提高所开发软件的可维护性，降低维护费用；

（4）提高软件开发生产率，及时交付使用。

1.2.2 软件工程研究的内容

作为一门新兴的学科，软件工程研究的主要内容是软件开发技术和软件工程管理两方面。在软件开发技术方面，主要研究软件开发方法、软件开发过程、软件开发工具和环境。在软件工程管理方面，主要研究软件管理学、软件经济学、软件心理学等。

软件开发方法是指导软件开发的某种标准规程，它规定了明确的工作步骤和具体的描述方式。软件开发方法覆盖了软件开发过程中的一系列活动和任务，包括软件定义、软件开发和软件维护等。在软件开发方法的指导和约束下，面对每个环节的问题，所有开发人员都遵循统一的标准，按照统一的步骤和方式共同完成软件产品。软件开发方法作为软件生产的行为依据，有效地保证了软件的质量和开发效率。

软件开发工具是辅助和支持软件开发全过程的一系列软件，是在高级程序设计语言的基础上，为提高软件开发的质量和效率而从定义、分析、设计、编码、测试、归档和管理等各方面，为软件开发人员提供各种帮助的一类软件。其目的是为了提高软件的生产率、改进软件质量。软件开发工具所涵盖的范围，不仅包括编程阶段，还包括如需求分析、系统设计、软件测试和软件维护等软件开发的各个阶段。

软件工程管理是一门新兴的管理学科，它通过管理风险、平衡冲突目标、克服各种限制、合理配置和使用资源等一系列活动，达到为用户提供满足应用需求的软件的目标。软件工程管理主要包括以下内容：软件项目管理、软件风险管理、软件质量管理、软件配置管理、软件进度管理。

总的来说，软件工程是一门交叉学科，涉及的范围很广泛。

1.3 软件的开发方法

软件工程学的最终目的是以较少的投资获得较高质量的软件产品。研究软件开发方法的目的是使开发过程“纪律化”，也就是寻找一些规范的“求解过程”，使开发工作能够有计划、有步骤地进行。

软件开发方法又称为软件工程方法论。目前，常用的软件开发方法有面向过程的开发方法、面向数据的开发方法和面向对象的开发方法。在本书的后续章节中，将详细介绍面向过程、面向数据和面向对象的开发方法。

1.3.1 面向过程的方法

面向过程的方法包括面向过程需求分析、面向过程设计、面向过程编程、面向过程测试、面向过程维护、面向过程管理。面向过程的方法又称结构化方法，包括结构化分析、结构化设计、结构化编程、结构化测试、结构化维护。面向过程的方法的特点是：程序的执行过程不由用户控制，完全由程序控制，故其优点是简单实用，缺点是维护困难。

面向过程的方法开始于 20 世纪 60 年代，成熟于 70 年代，盛行于 80 年代。该方法的基本特点是，分析设计中强调“自顶向下、逐步求精”，编程实现时强调程序的“单入口和单出口”。这种方法在国内曾经大量应用，非常普及。

对于软件行业来说，某一种方法论往往来自于某一类程序设计语言。面向过程的方法，来自于 20 世纪 60 至 70 年代流行的面向过程的程序设计语言，如 ALGOL、Pascal、BASIC、FORTRAN、COBOL、C 语言等。这些语言的特点是：用“顺序、选择（if-then-else）、循环（do-while 或 do-until）”三种基本结构来组织程序编制，实现设计目标。

面向过程的方法，在军事上的实时跟踪监控系统中有很好的应用。如我方侦察卫星发射后其飞行轨迹的捕获、测量、跟踪和预报，导弹防御系统中敌方导弹发射后飞行轨迹的捕获、测量、跟踪和预报，其软件系统都是采用面向过程的方法设计和实现的。使用面向过程的方法，系统的执行路径可由系统自动控制，也就是程序自动控制，这是一切自动控制与跟踪系统所必需的。

1.3.2 面向数据的方法

面向数据的方法，也称为面向元数据（Metadata）的方法。元数据是关于数据的数据，组织数据的数据。例如，数据库概念设计中的实体名和属性名，数据库物理设计中的表名和字段名，它们就是元数据。而具体的某一个特定的实例，就不是元数据，它们叫做对象或记录，是被元数据组织或统帅的数据。

面向数据的方法开始于 20 世纪 80 年代，成熟于 90 年代。20 世纪 80 年代中期，美国学者 James Martin 在《信息系统宣言》中提出了“以数据为中心”的学说，它是这种设计方法的萌芽与起源。90 年代中期，Sybase 和 Oracle 公司的 CASE 工具 Power Designer 和 Designer/2000（以后叫做 Oracle Designer）的出现，宣告这种设计方法已经进入工程化、规范化、自动化和实用化阶段，因为 CASE 工具中隐含了这种方法。概括起来，面向数据方法的要点是：

（1）数据（Data）位于企业信息系统的中心。信息系统就是对数据的输入、处理、传输、

查询和输出。

（2）只要企业的业务方向和内容不变，企业的元数据就是稳定的，由元数据构成的数据模型（Data Model）也是稳定的。

（3）对元数据的处理方法是可变的。用不变的元数据支持可变的处理方法，即以不变应万变，这就是企业信息系统工程的基本原理。

（4）企业信息系统的核心是数据模型。数据模型包括概念数据模型 CDM（Conceptual Data Model）和物理数据模型 PDM（Physics Data Model）。数据模型的表示形式是 E-R 图，E-R 图要用 CASE 工具设计。例如，Power Designer，Oracle Designer 或 ERwin，它们不但具有正向设计功能，而且具有逆向分析功能，这样才能实现快速原型法。

（5）信息系统的实现（编码）方法主要是面向对象，其次才是面向数据和面向过程。

（6）用户自始至终参与信息系统的分析、设计、实现与维护。

面向数据方法的特点是：程序的执行过程中，根据数据流动和处理的需要，有时由程序员控制（如数据库服务器上触发器和存储过程的执行），有时由用户控制（如用户浏览层上控件的选择与执行）。

面向数据方法的优点是通俗易懂，特别适合信息系统中数据层（数据库服务器）上的设计与实现，缺点是实现窗口界面较困难。

面向数据的方法来自于 20 世纪 80 年代开始流行的关系数据库管理系统 RDBMS，以及关系数据库程序设计语言。例如，Oracle，Sybase 关系数据库语言，这种关系数据库语言或命令，提供了强大的面向关系表中数据的编程能力，典型的例子就是编写存储过程和触发器。Oracle 数据库管理系统自带的编程工具 Developer 2000，首先是一个面向数据的编程工具，其次才是一个面向对象的编程工具。Oracle Designer 加上 Developer 2000，便构成了一个完整的面向数据的信息系统开发环境。

面向数据的方法与关系数据库管理系统紧密地捆绑在一起，只要面向对象数据库不能完全替代关系数据库，这种方法就不会终结。目前数据库管理系统的发展趋势是：在关系型数据库的基础上，将面向对象的某些特性（如继承）添加上去，称为“对象-关系型数据库”，但本质上仍然是一个关系型数据库。正如美国数据库专家 David M.Kroenke 所说的，“面向对象这样的数据库只是概念上的兴趣，他们在商用数据库处理中只起很小的作用。”

面向数据的方法在电子商务中的应用有：网站后台数据库服务器上的数据处理和数据传输，其软件都是利用面向数据的方法设计与实现的。实际上，不管网络应用系统结构是两层结构还是三层结构，在数据库服务器上对数据的分析、设计和实现，都自觉或不自觉地使用了面向数据的方法。

1.3.3 面向对象的方法

面向对象方法起源于面向对象编程语言。20 世纪 80 年代中期到 90 年代，面向对象的研究重点已从编程语言转移到设计方法学上来，其中有代表性的工作有：

（1）B. Henderson-sellers 和 J. M. Edwards 提出的面向对象软件生存周期的喷泉模型，以及面向对象系统开发方法；

（2）G. Booch 提出面向对象开发方法等；

（3）P. Coad 和 E. Yourdon 提出面向对象分析（OOA）和面向对象设计（OOD）；

（4）J. Rumbaugh 等人提出的对象建模技术（OMT）；

（5）Jacobson 提出的面向对象软件工程（OOSE）；

（6）由 G. Booch，J. Rumbough 和 Jacobson 等人发起，在 Booch 方法、OMT 方法加 OOSE 方法的基础上推出了统一的建模语言（UML）。

面向对象方法包括分析、设计和实现活动。它是一种把面向对象的思想运用于软件开发过程，指导开发活动的系统方法，建立在“对象”概念（对象、类和继承）基础上的方法学基于对象概念，以对象为中心，以类和继承为构造机制来认识、理解、刻画客观世界和设计、构建相应的软件系统。面向对象方法的应用，目前有两个方面：

（1）在分析、设计、实现活动中完全采用面向对象的技术；

（2）传统的功能分解方法与面向对象方法结合使用，即功能分析、面向对象的设计和实现，以及面向对象分析和设计、实现用过程式语言。

面向对象分析的任务就是通过分析问题域建立系统的概念模型，并用相应的符号系统表示。而模型一般由五个层次构成，即主题层、类及对象层、结构层、属性层和服务层，因此其步骤也是按其五个层次逐步展开的。

面向对象设计是在面向对象分析的基础上进行系统设计，包括交互过程和用户接口、任务管理、全局资源协调并确定边界、各个类的存储和数据格式。

面向对象实现就是用面向对象程序设计语言来实现面向对象设计，因为该类语言支持对象、运行多态性和继承，因此比较容易，如果使用非面向对象程序设计语言，则需要特别注意和规定保留程序的面向对象式的程序结构。

用面向对象方法开发的软件，是基于客观世界界定的对象结构，与传统的软件相比较，软件本身的内容结构发生了质的变化，因而易复用性和易扩充性都得到了提高，而且能适应需求的变化。

谈到软件开发方法，软件复用技术是必不可缺的。当今国际上十分重视软件复用，提出了构件、体系结构和框架等一系列方法。我们将在本书的其他章中加以介绍。

面向对象的方法在电子商务中也有应用，网站前台界面的制作、信息的发布和处理、用户在网上浏览和录入信息等应用软件都是利用面向对象的方法设计与实现的，除此之外，个人网页的制作也是面向对象方法的应用例子。窗口操作系统与互联网的出现，为面向对象方法的前景提供了无限的空间。

1.3.4 三种开发方法的比较

面向过程的方法、面向数据的方法和面向对象的方法这三种开发方法的比较如表 1-1 所示。

表 1-1 三种开发方法的比较

方法名称	优点	缺点	适用情况
面向过程的方法	简单易学	不适应窗口界面、维护困难	大型工程计算、实时数据跟踪处理、各种自动化控制系统、系统软件实现等领域
面向数据的方法	通俗易懂	不适应窗口界面	以关系数据库管理系统为支撑环境的信息系统建设
面向对象的方法	功能强大、易于维护	不易掌握	互联网时代，完全由用户交互控制程序执行过程的应用软件和系统软件的开发

由表 1-1 可知，这三种开发方法各有优缺点、生存时间和适用空间，所以它们在信息系统领域能和平共处、互相促进，共同构成一个多极化的世界。

1.4 软件生命周期

如同人的一生，软件也有一个孕育、诞生、成长、衰亡的生存过程，这个过程称为软件的生命周期。

软件生命周期是指软件产品或软件系统从产生、投入使用到被淘汰的全过程。软件生命周期大致可以分为六个阶段：可行性研究、需求分析、软件设计、编码、测试和维护。

1. 可行性研究

可行性研究是从技术、经济等角度判断软件系统的开发目标是否可以实现。如果目标不可行，或软件系统没有可行的解决办法，系统分析员应当建议停止开发软件项目。

可行性研究的目的是在最短的时间内、以最少的成本确定已定义的问题是否值得解决。必须强调的是，可行性研究的目的不是解决问题，而是确定问题是否可解或是否值得去解。可行性研究是在技术、经济、操作和法律等方面寻求可行的解决方案，然后对各个方案进行比较，并完成可行性研究报告的过程。

2. 需求分析

需求分析是一个很复杂的过程，需求分析是否准确和成功，直接关系到软件开发的成败。需求分析的任务是对待开发软件在功能、性能、界面、数据等方面的要求进行详细具体的描述，并编写软件需求规格说明书。该文档既是开发者和用户之间的合同书，也是后续软件设计的依据，更是用户验收系统的标准。

3. 软件设计

软件设计分概要设计和详细设计两部分。

概要设计也称为总体设计，是在软件需求规格说明书的基础上，对软件的总体结构进行规划，主要完成软件架构设计、模块分解、模块功能定义和模块接口描述等工作。

详细设计在概要设计的基础上，对模块进行具体、详细的过程性描述，用各种工具表示模块的结构、过程、功能和对外接口。详细设计是编码的依据。

4. 编码

在编码阶段，开发人员根据设计阶段制订出的设计方案，编写程序代码。简单地说，编码的过程就是把详细设计文档中对每个模块实现过程的算法描述转换为能用某种程序设计语言来实现的程序。

5. 软件测试

软件测试的目的是发现软件产品中存在的软件缺陷，进而保证软件产品的质量。软件缺陷发现得越晚，修复缺陷所需的成本就越高，损失也就越大。软件缺陷是不可避免的，所以软件测试是保证软件质量的关键步骤。

6. 软件维护

软件维护是软件生命周期的最后一个阶段，同时也是持续时间最长的一个阶段。在软件产品的使用过程中，用户会不断地发现产品中所隐藏的各种缺陷。同时，随着用户需求的增加或改变，或市场环境的变化，软件产品的功能需要不断更新，版本需要不断升级。所以，在使用软件产品的过程中，软件开发人员需要对产品进行维护，以保证软件产品的正常运行。

1.5 软件开发模型

在软件工程中，人们通过建立抽象的软件开发模型，把软件生命周期中的各个活动或步骤安排到一个框架中，将软件开发的全过程清晰且直观地表达出来。可以说，软件开发模型是软件工程思想的具体化，它反映了软件在其生命周期中各阶段之间的衔接和过渡关系以及软件开发的组织方式，是人们在软件开发实践中总结出来的软件开发方法和步骤。

常见的软件开发模型有瀑布模型、原型模型、增量模型、螺旋模型以及统一软件开发过程。

1.5.1 瀑布模型

瀑布模型是出现得比较早的软件开发模型。瀑布模型将软件生命周期划分为软件计划、需求分析、软件设计、软件编码、软件测试、软件运行和维护六个阶段，各个阶段之间如同瀑布流水，自上而下，逐级下落，形成相互衔接的固定次序。每一个阶段的工作都以上一个阶段工作的结果为依据，同时为下一个阶段的工作提供前提和基础。瀑布模型如图 1-1 所示。

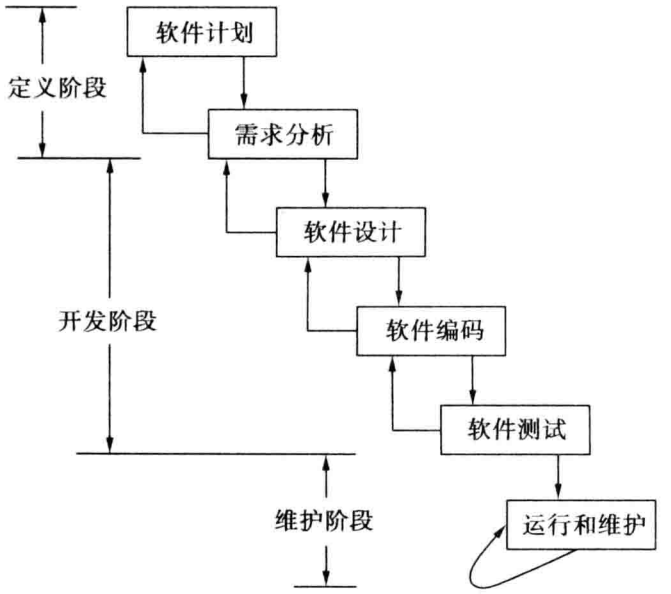

图 1-1 瀑布模型

瀑布模型具有以下几个特点：

（1）瀑布模型强调软件开发过程的阶段性，每个阶段完成特定的任务。软件开发人员可以按阶段制定项目计划、核算成本、分配资源和评审标准等，这为软件项目的管理提供了便利，增强了项目管理的可操作性。

（2）瀑布模型是一种基于里程碑的阶段过程模型。开发人员为每个阶段设立里程碑或基线，并组织好对基线的评审与审计。阶段评审通过后，才能开始下一阶段的工作。

（3）瀑布模型是一种线性的软件开发模型，回溯性很差。只有某一个阶段的活动完成后，开发过程才会进入下一个阶段，各阶段之间按单向顺序逐级过渡。如果在后续阶段中发现前一阶段的错误，那么修复此错误将会造成巨大的人力以及财物损失。

瀑布模型为软件的开发和维护提供了一种有效地管理模式。在瀑布模型中，各个阶段之间的关系清晰、易懂；原理简单，易于掌握；每个阶段中都有验证和确认环节，以便进行质量管理；在下一阶段开始前，该模型会通过项目管理来控制本阶段工作的完成。