CHAPTER

第1章 概　述

进入21世纪，随着信息技术的快速发展，国民经济、国防建设、社会发展及人民生活都离不开计算机技术的广泛应用。软件是信息化的核心，软件产业关系到国家信息化和经济发展、文化与系统应用，体现了一个国家的综合实力，也决定着国际竞争地位。因此，提高软件的开发、维护、管理能力和掌握先进的软件技术，对国家信息化发展和信息技术应用水平及综合国力的提高至关重要。

教学目标

- 了解软件工程的产生和发展；
- 掌握软件工程的概念、内容和原理；
- 熟悉软件生存周期及阶段任务；
- 掌握软件开发模型。

【案例1-1】 IBM公司研发初期的OS/360共有约100万条指令，花费了5000人·年，经费达数亿美元，而结果却令人沮丧，错误多达2000个以上，系统根本无法正常运行。OS/360系统的负责人Brooks这样描述开发过程的困难和混乱："像巨兽在泥潭中垂死挣扎，挣扎得越猛，泥浆沾得越多，陷入越深，最后没有一只野兽能够逃脱淹没在泥潭中的命运。"

1.1 软件工程的发展

从"软件工程(Software Engineering)"的概念提出至今已经40多年，在解决20世纪60年代末所出现的"软件危机"过程中逐渐形成与发展，随着信息化建设、发展和信息技术的广泛应用，软件工程的理论、技术和应用都取得了很大的进展。

1.1.1 软件危机概述

20世纪60年代出现的软件危机是软件工程产生的直接原因。**软件危机**(Software crisis)是指在计算机软件开发、运行、维护和管理过程中所遇到的一系列严重问题。软件

危机主要包含两方面的问题：一是开发的软件如何满足社会对其日益增长的各种需求，二是怎样维护和管理不断快速增长的已有软件。

1. 软件危机产生的原因

软件开发过程是一项高度集成的脑力劳动，涉及很多方面。软件开发初期“生产作坊式”的模式和技术已完全无法适应软件快速发展的实际需要，致使大量质量低劣的软件产品投入使用，故障频发，一些开发过程中的大型软件系统遇到了许多棘手问题，有的甚至半途而废，有的比原计划推迟了多年，或费用大大超出预算，有的系统不能符合用户预期，有的无法进行修改更新和维护。

软件危机产生的**主要原因**有：

(1) 软件开发规模逐渐变大，复杂度和软件的需求量不断增加；

(2) 没有按照工程化方式运作，开发过程没有统一的标准和准则，以及科学规范的指导方法；

(3) 软件需求分析与设计考虑不周，软件开发、维护和管理不到位；

(4) 开发人员与用户或开发人员之间的相互交流沟通不够，文档资料不完备；

(5) 软件测试调试不规范、不细致，提交的软件质量不达标；

(6) 忽视软件运行过程中的正常维护和管理。

2. 软件危机的主要表现

出现的软件危机致使所研发软件的功能、性能和可靠性等细节难以保障，研发进度无法把握，成本增长难以控制，研发人员不断增加，软件运行维护和管理方面的工作量不断增大等问题。

【案例 1-2】 美国研发的阿波罗登月飞行计划的软件，堪称 20 世纪世界上最为精心设计的大型软件，花费了巨额投资和大量人力，最后仍然没能避免出错。例如，阿波罗 8 号由于太空飞船的一个计算机软件错误，造成存储器的一部分信息丢失；阿波罗 14 号在飞行的 10 天中，出现了 18 个软件错误。

软件危机**主要表现**在以下 7 个方面：

(1) 已完成的软件系统时常出现功能、性能不满意或出现故障等现象。

(2) 软件产品的可靠性和质量安全等方面时常达不到标准。软件产品质量难以保证，甚至在开发过程中就被迫中断。

(3) 软件开发管理差，对成本和进度的估计时常不准确。

(4) 系统时常出现无法维护、升级或更新的现象。

(5) 软件开发没有标准、完整、统一规范的文档资料。计算机软件不仅仅是程序，还应当有一整套规范的文档资料和售后服务。

(6) 软件开发效率低，无法满足计算机应用迅速发展的实际需要。

(7) 软件研发成本在计算机系统总成本中所占的比例逐年上升。

3. 解决软件危机的措施

随着信息技术的日益发展和广泛应用，软件在现代信息化社会主要用于信息处理和

管理等方面。而计算机硬件本身的基本功能只适用于数值和逻辑运算，只能依靠各种计算机软件来满足各种需求，同时也致使软件变得更加复杂庞大，因此，必须采取有力措施。

解决软件危机的**主要措施**包括以下 3 个方面。

(1) 技术方法。运用软件工程的技术、方法和标准规范。

(2) 开发工具。选用先进高效的软件工具，同时采取切实可行的实施策略。

(3) 组织管理。研发机构应当做到组织高效、管理制度和标准，严格规范、职责明确、质量保证、团结互助、齐心协力，注重文档及服务。

为了避免在软件开发中出现软件危机，不仅需要标准规范的技术措施，更需要强有力的组织管理保障。只有各方面密切配合、齐抓共管，切实以软件工程方式方法和规程进行运作，才能确保软件质量和信息化的健康发展。

【案例 1-3】 某企业投资 32 万元用于网络销售软件的开发和建设，软件开发者为某高校计算机学院的项目研发小组，在软件开发前的需求调研分析阶段，该系的教师组织有关师生在商厦设备处的计算机室负责人的陪同下对各业务部门进行了调研，并根据各业务部门的需要编制了按业务部门划分的系统功能模块需求说明书。后来，将师生编成若干个软件开发小组，分别负责各个功能模块的研发。两年后，大部分的功能模块开发完毕，但发现各模块之间的数据不能很好地共享和传输，与系统有关的各类单证的录入、校对和传输比原来的手工处理过程还复杂，而且随着企业经营规模的扩大和经营方式及业务的变化，原有的业务部门也做了一些调整，所开发的功能模块只有 55%能勉强使用。由于大部分学生毕业离校，各模块的开发文档资料保存不全，最后，项目无法继续进行而终止，并因为没有按期达到合同规定要求而赔偿损失。

1.1.2 软件工程的发展过程

软件工程的发展过程与软件的发展过程紧密相关。从 1946 年电子计算机诞生以来，计算机软件随着信息技术的快速发展得到了广泛应用。最初只有程序的概念，逐渐出现软件的概念，后来由于软件需求量及市场化规模的快速扩大，用户将软件视为产品，并确定了软件开发各阶段的编程技术和称为“文档”的图文资料及服务。

计算机软件从数值计算到广泛应用于各行各业，软件技术的发展经历了程序设计阶段、程序系统阶段、软件工程阶段和创新完善软件工程阶段，其典型技术如表 1-1 所示。

表 1-1 软件技术各发展阶段的典型技术

阶段	程序设计阶段	程序系统阶段	软件工程阶段	创新完善软件工程阶段
软件典型技术	面向批处理 有限的分布 自定义软件	多用户 实时处理 数据库 软件产品	分布式系统 嵌入“智能” 低成本硬件 消费者的影响	强大桌面系统面向对象技术、专家系统、神经网络、并行计算、网格计算等高新技术

在软件技术发展过程中，由于受到软件危机的重要影响，逐步形成了研究如何避免和解决软件危机的技术和方法，产生了科学合理地开发和维护软件的学科——软件工程学。

在此过程中，“软件工程”的**发展经历**了 4 个重要阶段。

1. 传统软件工程

传统软件工程是指软件工程产生的初期，也称为第一代软件工程。20 世纪 60 年代末到 70 年代，软件主要采用“生产作坊”的开发方式，随着软件需求量、规模及复杂度的快速增大，不断出现各种难以解决的软件问题，生产作坊的开发方式已无法适应软件开发的需要，导致出现了“软件危机”，使软件开发效率低、成本高、进度及质量失控，很多技术问题难以及时解决，大量无标准开发的低劣软件涌入市场，“软件危机”不断扩大。

1968 年，北大西洋公约组织(NATO)在联邦德国召开的一次会议上首次提出“软件工程”这一术语，并专门讨论了软件危机问题。从此将软件开发纳入了工程化的轨道，基本形成了软件工程的概念、框架、技术和方法。

2. 对象工程

对象工程也称为第二代软件工程。20 世纪 80 年代中到 90 年代，以 Smalltalk 为代表的面向对象的程序设计语言相继推出，使面向对象的方法与技术得到快速发展。从 20 世纪 90 年代起，研究的重点从程序设计语言逐渐发展到面向对象的分析与设计技术，形成了一种完整的软件开发方法和一套系统的技术体系，之后出现了许多面向对象的开发方法，使面向对象的开发技术和方法逐渐得到完善和广泛流行。

3. 过程工程

过程工程也称为第三代软件工程。随着计算机网络等高新技术的出现及信息技术的广泛应用，软件规模和复杂度不断增大，开发时间相应地持续增长，开发人员的增加致使软件工程开发和管理的难度不断增强。在软件开发的实践过程中，软件企业和开发者逐渐意识到，提高软件生产效率并保证软件质量的关键在于对“软件过程”的有效控制和管理，从而提出了对软件项目管理的计划、组织、成本估算、质量保证、软件配置管理等技术与策略，逐步形成了软件过程工程。

4. 构件工程

构件工程也称为第四代软件工程。从 20 世纪 90 起年代，基于**构件**(Component)的开发方法取得重要进展，软件系统的开发可利用已有的可复用构件组装完成，而无须从头开始构建，从而达到了提高效率和质量、降低成本的目的。

面对复杂的操作系统控制的桌面系统，连接各种网络、数字通信与先进的应用软件综合需求。计算机体系结构从主机环境转变为分布式的客户机/服务器等环境。**计算机辅助软件工程**(Computer Aided Software Engineering，CASE)将工具和代码生成器进行集成，为很多软件系统提供了可靠的解决方案；专家系统和人工智能软件的应用更加广泛；人工神经网络软件开阔了信息处理的新途径；并行计算、网络技术、虚拟技术、多媒体技术和现代通信技术等新技术新方法改变了人们原有的工作方式。

智能计算机、量子计算机、光计算机、生物计算机和化学计算机等新一代计算机硬件的开发，也极大地促进了软件工程技术的变革和发展。21 世纪将极大地促进软件开发工业化的大规模发展，以达到产品化质量要求的工业标准，满足软件开发自动化、智能化等需求。其突出特征是：计算机真正成为人们的一种工具，用户即分析员，软件工程即软

件，模型驱动及面向服务开发等新方法将成为软件工程的最新发展趋势。

课堂讨论：

(1) 上述案例项目的开发过程是否存在软件危机问题？

(2) 从案例项目的组织和管理等方面分析导致该项目失败的主要原因有哪些。

(3) 你认为应当吸取哪些教训？应该采取的主要措施有哪些？

1.2 软件及软件工程的概念

为了更好地学习有关软件工程的技术方法，先介绍一下有关软件及软件工程的概念、软件工程学相关内容和软件工程的基本原理等。

1.2.1 软件的概念及特点

1. 软件的概念

软件(Software)是计算机系统运行的指令、数据、资料和服务的集合，包括指令程序、数据、相关文档和完善的售后服务的完整集合，即：

软件＝程序＋数据＋文档＋服务

其中，**程序**是事先按照预定功能性能等要求设计和编写的指令序列；**数据**是使程序正常处理信息的数据结构及信息表示；**文档**(Document)是与程序开发、维护和使用有关的技术数据和图文资料。

【注意】 程序与软件不同，程序只是软件的组成部分。“软件就是程序”的观点为误解，也严重影响了软件工程的正常进行和发展。文档必不可少，只有程序不能称为软件。

国内外还有一些专家认为：软件是程序以及开发、使用和维护程序所需的所有文档，它是由应用程序、系统程序、面向用户的文档及面向开发者的文档构成的，即软件＝程序＋文档。其实这种观点也不够准确。

信息系统(Information System)有时也称为软件，是指由一系列相互联系的部件(程序模块)组成的，为实现某个目标对信息进行输入、处理、存储、输出、反馈和控制的集合。分为操作系统、应用系统等。通常实例提到的信息系统主要是指应用系统，即应用软件。

2. 软件的特点

在软件的实际研发、运行、维护、管理和使用过程中，需要掌握其**特点**：

(1) 智能性。软件是人类智能劳动的产物、代替和延伸，其中的程序、流程、算法、数据结构等需要通过人的思维进行设计、编排和组织。

(2) 抽象性。软件属于逻辑实体，而非物理实体，无形性和智能性致使软件难以认识和理解。在研发过程中，需要进行逻辑设计和组织，运用抽象思维和方法。软件只能通过用户界面与软件交互，其丰富的内涵被蕴涵在计算机内部，使软件具有高度的抽象性。

(3) 人工方式。软件的开发、维护及设置管理等方面目前尚未完全脱离手工方式。

(4) 复杂性和系统性。软件开发和运行经常受到计算机系统的限制，软件的开发和运行必须依赖于软件环境。软件是由多种要素组成的有机整体，具有显著的系统特性。

软件具有确定的目标、功能、性能、结构和要素。

(5) 泛域性。软件应用很广泛,在信息化中可服务于各种领域、行业和层面。

(6) 复制性。软件成本相对比较昂贵,计算机软件是人类创造性的特殊产品。而复制和推广的费用一般较低,并可以借助复用技术进行软件开发再利用。

(7) 非损及更新性。软件不存在物理性磨损和老化问题,但会退化,需要更新升级。

计算机软硬件的失效率不同。硬件的失效率曲线是一个 U 形"浴盆"曲线,如图 1-1 所示,表明硬件随着使用时间的增加,其失效率急剧上升。而如图 1-2 所示的软件失效率曲线,因为软件不存在磨损和老化问题,然而存在退化问题,所以,没有 U 形曲线的右半翼,表明计算机软件随着使用时间的增加,其失效率降低。

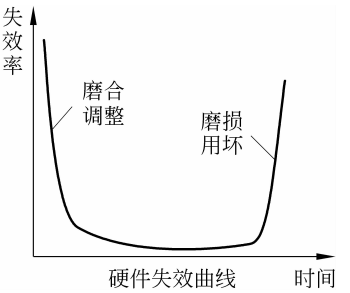

图 1-1　硬件失效率曲线

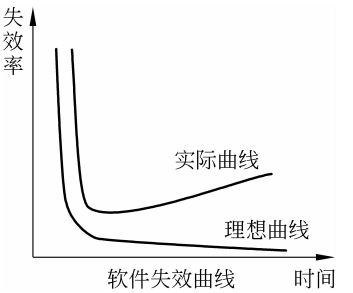

图 1-2　软件失效率曲线

3. 软件的分类

1) 按照软件功能划分

按照软件功能划分,可以将软件分为 3 类:

- 系统软件。如操作系统、设备驱动程序等。
- 支撑软件。协助用户开发的工具软件,如编译程序、程序库、图形软件包等。
- 应用软件。如企业业务管理软件、CAD/CAM 软件、CAI 软件、图书管理信息系统、库存管理信息系统等。

2) 按照软件规模划分

按照软件规模的大小(源代码行)、参与研发人数、研制时间,可以将软件分为微型、小型、中型、大型、超大型 5 种,如表 1-2 所示。但是,随着软件产品规模的不断扩大,类别指标也会发生变化。

表 1-2　软件规模分类

类　　别	研发人数	研制期限	产品规模(源代码行)
微型	1	1～4 周	小于 500 行
小型	1～3	1～6 月	500～1 万行
中型	4～10	1～2 年	1 万～5 万行
大型	11～20	2～3 年	5 万～10 万行
超大型	20 人以上	3 年以上	10 万行以上

3）按照软件工作方式划分

按照软件工作方式划分，可以将软件分为实时处理软件、分时软件、交互式软件、批处理软件 4 种。

4）按照软件服务对象的范围划分

按照软件服务对象的范围划分，可以将软件分为如下类型。

- 项目软件：由客户委托开发的软件。
- 产品软件：由软件开发机构开发，提供给市场的软件。

此外，还可以按照软件使用的频度或按照软件失效的影响等方面进行划分。

1.2.2 软件工程的概念及特点

1. 软件工程的定义

随着软件技术的快速发展，软件工程定义也在不断发展和完善，但是其基本思想都是强调在软件开发过程中利用工程化准则、技术和方法的重要性。

Boehm 曾为软件工程下过定义：运用现代科学技术知识，设计并构造计算机程序及开发、运行和维护这些程序所必需的相关文件资料。

1983 年，IEEE 在《IEEE 软件工程标准术语》中给出的定义是：软件工程是开发、运行、维护和修复软件的系统方法，其中“软件”的定义为：计算机程序、方法、规则、相关的文档资料以及在计算机上运行时所必需的数据。1990 年，IEEE 又将定义修改为：对软件开发、运作、维护的系统化的、有规范的、可定量的方法之应用，即是对软件的工程化应用。

中国国家标准 GB/T 11457—1995《软件工程术语》的**定义**是：软件工程（Software Engineering）是软件开发、运行、维护和引退的系统方法。

《计算机科学技术百科全书》中对软件工程的定义是：应用计算机科学、数学及管理科学等原理开发软件的工程。软件工程借鉴传统工程的原则、方法，以提高质量，降低成本。其中，计算机科学、数学用于构建模型与算法，工程科学用于制定规范、设计范型（paradigm）、评估成本及确定权衡，管理科学用于计划、资源、质量、成本等管理。

实际上，对软件工程可以理解为：采用工程的概念、原理、技术和方法，在计划、开发、运行、维护与管理软件的过程中，将科学的管理和最佳的技术方法紧密结合，以比较经济的手段获得满足用户需求的可靠软件的一系列方法，即：

软件工程＝工程原理＋技术方法＋管理技术

2. 软件工程的特点

软件工程学是软件工程化的思想、规范、过程、技术、环境和工具的集成，是将具体的技术和方法结合形成的一个完整体系。软件工程是在软件开发中采用工程化的原理和方法，采用一系列科学的、现代化的方法技术来开发软件。这种工程化的思想贯穿于软件开发和维护的全过程。

软件工程是计算机科学中的一个重要分支，是一门指导计算机软件系统开发、运行、维护和管理技术的工程学科，不仅具有一般工程学科的共性，以及智能性、抽象性、复杂性

和更新性等特性，还具有软件工程学科的系统性、工程化、综合性和学科交叉性的基本特点。软件工程学科的**主要特点**是实践性和发展性，软件工程的问题来源并应用于实践，最终目的是有效地生产软件产品。其**特点**体现为“三多”：一是多学科，不仅包含有关课题，还涉及计算机科学、工程科学、管理科学、数学等多个学科；二是多目标，不仅关心项目产品及其功能，还注重质量、成本、进度、性能、可靠性、安全性、通用性、可维护性、有效性和界面等；三是多阶段，软件开发不只是编程，而是由可行性研究、计划立项、需求分析、总体设计、详细设计、编程、测试、运行、维护等阶段构成的完整过程。软件工程的目的是在规定的时间和开发经费内，开发出满足用户需求的、高质量的软件产品。其目标是实现软件研发与维护的优质高效和自动化。

1.2.3 软件工程学的主要内容

1. 软件工程方法

通常将在软件研发计划、开发、运行和维护过程中所使用的一整套技术方法称为方法学(methodology)或范型(paradigm)。软件工程学的**主要内容**包括软件开发技术和软件工程管理两个方面。软件开发技术包括软件工程方法、软件工具和软件开发环境；软件工程管理学包含软件工程经济学和软件管理学。

软件工程的**研究范围**很广，不仅涵盖软件系统的开发方法和技术、维护与管理技术，还包括软件工具、环境及软件开发的规范。目前，对软件工程学科的构成和内容尚未达成共识，根据软件工程研究的对象和任务，软件工程学科主要包括软件工程原理、软件工程过程、软件工程方法、软件工程技术、软件工程模型、软件工程管理、软件工程度量、软件工程环境、软件工程应用等基本内容。软件工程学科的主要内容如表 1-3 所示。

表 1-3 软件工程学科的主要内容

软件工程原理	软件目标、原则、学科基础
软件工程过程	开发过程、运作过程、维护过程，如获取、供应、管理、开发、运作、维护、支持、裁减
软件工程技术	开发技术、管理技术、度量技术、维护技术、应用技术
软件工程方法	开发方法、管理方法、度量方法、维护方法、应用方法、环境方法
软件工程模型	领域模型、需求模型、设计模型、实现模型、测试模型
软件工程管理	项目管理、质量管理、文档管理
软件工程度量	规模、复杂度、进度、费用、工作量
软件工程环境	硬件、网络、支撑软件
软件工程应用	应用软件工程基本原理、方法、技术解决特定领域的问题

软件工程方法学是研发软件的系统方法，确定软件开发阶段，规定每一阶段的目标、任务、技术、方法、产品、验收等步骤和完成准则。具有方法、工具和过程三个要素，也称**软件工程三要素**。

- 软件工程方法：包括软件开发“如何做”的技术和管理准则及文档等技术方法。

- 软件工具：为方法的运用提供自动或半自动的软件支撑工具的集成环境。
- 软件工程过程：主要包括任务的工作阶段、工作内容、产品、验收的步骤和完成准则。也有人将这一要素确定为“组织管理”，实际上称为“过程与管理”更合适。

目前，**常用的软件工程方法**主要分为以下 7 种类型。

1）面向功能方法

面向功能的软件开发方法也称为**结构化方法**，主要采用结构化技术，包括结构化分析、结构化设计和结构化实现，按照软件的开发过程、结构和顺序完成开发任务。此方法是在 1978 年由 E. Yourdon 和 L. L. Constantine 提出的，即结构化分析与设计(Structured Analysis and Structured Design，SASD)方法。

结构化方法是 20 世纪 80 年代使用最广泛的软件开发方法。先用结构化分析(SA)对软件进行需求分析，再用结构化设计(SD)方法进行软件设计，最后是结构化编程(SP)。此方法不仅开发步骤明确，SA、SD、SP 任务连贯、相辅相成，而且给出了规范和变换型及事务型两类典型的软件结构，便于参照，使软件开发效率得到极大提高，从而深受开发人员的欢迎，现在仍然用于自动化及过程控制等方面。

2）面向数据方法

1975 年，M. A. Jackson 提出了**面向数据(结构)的软件开发方法**，也称 Jackson 方法。从目标系统输入、输出数据的结构，导出程序框架结构，再补充其他细节，得到完整的程序结构图。也可与其他方法结合，用于模块的详细设计和数据处理等。此方法对输入输出数据结构明确的中小型系统很有效，如商用文件表格处理等。

3）面向对象方法

面向对象方法(Object-Oriented Method，OOM)是一种将面向对象的思想应用于软件开发过程中，指导开发活动的系统方法。将对象作为数据和对数据的操作相结合的软件构件，用对象分解取代了传统方法的功能分解。该方法将所有对象都划分为类，将若干个相关的类组织成具有层次结构的系统，下层的子类继承上层的父类所具有的数据和操作，而对象之间通过发送消息相互联系。

OOM 是 20 世纪 80 年代推出的一种全新的软件开发方法。非常直观实用高效，被誉为 20 世纪 90 年代软件的核心技术之一。其基本思想是：对问题领域进行自然的分割，以更接近人类通常思维的方式建立问题领域的模型，以便对客观的信息实体进行结构和行为的模拟，从而使设计的软件更直接地表现问题的求解过程。面向对象的开发方法，以对象作为最基本的元素，是分析和解决问题的核心。OOM 的**要素**是对象、类、继承以及消息通信。可概括为：

面向对象＝对象＋类＋继承＋消息通信

实际上，所有按照这样四个概念设计和实现的软件系统，都可以认为是面向对象的。OOM 由 OOA(面向对象的分析)、OOD(面向对象的设计)和 OOP(面向对象的程序设计)三部分**组成**。OOM 是多次反复、迭代开发的过程。面向对象方法在分析和设计时使用相同的概念和表示方法，两者之间没有明显的界限。最终产品是由许多基本独立的对象组成的，这些对象具有简单、易于理解、易于开发、易于维护的特点，并且具有可重用性。

4) 面向问题方法

面向问题方法也称**问题分析法**(Problem Analysis Method, PAM),是在 20 世纪 80 年代末由日立公司提出的,是在 Yourdon 方法、Jackson 方法和自底向上的软件开发方法的基础上扬长避短改进而成的。其**基本思想**是:以输入、输出数据结构指导系统的问题分解,经过系统分析逐步综合。其**步骤**是:从输入、输出数据结构导出基本处理框;分析这些处理框之间的先后关系;按先后关系逐步综合处理框,直到画出整个系统的问题分析图。此方法成功率较高,曾在日本较为流行,但在输入、输出数据结构与整个系统之间仍然存在着难以解决的问题,因此只适用于中小型系统问题。

5) 面向方面的开发方法

面向方面的程序设计(Aspect-Oriented Programming, AOP)是继面向对象技术之后的新的软件开发的研究方向。随着软件规模和复杂性的不断增加,各组件之间的相互影响越来越复杂,限制了软件的重用性和适应性,并导致验证系统的逻辑正确性更加困难。软件开发的传统方法,已无法从根本上解决由于提高系统复杂度带来的代码混乱及纠缠问题。面向方面的系统是面向对象系统的扩展,在现有的 AOP 实现技术中,可通过创建 Aspect 库或专用 Aspect 语言实现面向方面的编程。

6) 基于构件的开发方法

基于构件的开发(Component-Based Development, CBD)或基于构件的软件工程(Component-Based Software Engineering, CBSE)方法是软件开发新范型。它是在一定构件模型的支持下,利用复用技术又好又快地构造应用软件的过程。由于以分布式对象为基础的构件实现技术日趋成熟,CBD 已成为现今软件复用实践的研究热点,被认为是最具潜力的软件工程发展方向之一。

软件复用(Software Reuse)或软件重用是指将已有的软件构件用于构造新的软件系统的过程。该技术是有效提高软件生产率和质量,降低成本的好方法。软件复用方法采用的复用方式包括以下几种。

(1) 复用分析:利用原有的需求分析结果,进一步深入分析比对查找异同及特性等。

(2) 复用结构:主要复用系统模块的功能结构或数据结构等,并进行改进提高。

(3) 复用设计:由于复用受环境影响小,设计结果比源程序的抽象级别高,因此可通过从现有系统中提取全部或不同粒度的设计构件,或独立于具体应用开发设计构件。

(4) 复用程序:包括目标代码和源代码的复用,可通过连接(Link)、绑定(Binding)、包含(Include)等功能,支持对象链接及嵌入(OLE)技术实现。

7) 可视化方法

20 世纪 90 年代随着可视化技术的兴起及其元素生成的不便,使用户界面在软件系统中所占的比例逐渐扩大,为此 Windows 提供了应用程序设计接口 API(Application Programming Interface),包含 600 多个函数,极大地方便了图形用户界面的开发。但是,大量的函数参数和使用数量更多的有关常量,使基于 Windows API 的开发变得很难,因此,Borland C++ 推出了 Object Windows 编程,将 API 的各部分用对象类进行封装,提供了大量预定义的类,并为这些定义了许多成员函数。利用子类对父类的继承性,以及实例对类的函数的引用,应用程序的开发可省略大量类的定义、大量成员函数的定义或只需作

少量修改即可定义子类。Object Windows 还提供了许多标准的缺省处理，极大地减少了应用程序开发工作量。由于非专业人员较难掌握，所以利用 Windows API 或 Borland C ++ 的 Object Windows 开发了一批可视开发工具。使**可视化开发**在可视开发工具提供的图形用户界面上，通过操作界面元素，如菜单、按钮、对话框、编辑框、单选框、复选框、列表框和滚动条等，由可视开发工具即可自动生成应用软件，这类应用软件工作方式是事件驱动。对每一事件，由系统产生相应的消息，再传递给相应的消息响应函数。这些消息响应函数是由可视开发工具在生成软件时自动装入的。

【注意】 由于软件与程序是不同的概念，软件开发方法与程序设计方法也是两个不同的概念。软件开发方法可以是针对局部的，也可以是针对全局的。软件工程方法更加强调和重点研究的是需求分析与软件设计的开发方法。

本书既介绍传统方法，以便掌握软件开发的基本步骤、方法和文档书写规范，也介绍面向对象等流行方法。在实际工作中，各种软件工程方法各有其特点，应当根据具体情况进行选择，也可将不同方法结合起来，取长补短合理利用，在提高软件开发效率的同时，提高软件的质量。

2. 软件工具

“工欲善其事，必先利其器。”软件工具和软件开发方法密切相关，它们是软件开发的两大支柱。**软件工具**(Software Tools)是指为支持软件的开发、维护、管理而专门研发的计算机程序系统。其目的是提高软件开发的质量和效率，降低软件开发、维护和管理的成本，支持特定的软件工程方法，减少手工方式管理的负担。软件工具通常由工具、工具接口和工具用户接口三部分构成。工具通过工具接口与其他工具、操作系统以及通信接口、环境信息库接口等进行相连交互。

软件工具种类繁多、涉及面广，可组成“工具箱”或“集成工具”，如编辑、编译、正文格式处理，以及静态分析、动态跟踪、需求分析、设计分析、测试、模拟和图形交互等。按照应用阶段分为计划工具、分析工具、设计工具、测试工具等，按照功能分为分析设计、Web 开发、界面开发、项目管理、软件配置、质量保证、软件维护等。

在 1.1.2 节中提到的计算机辅助软件工程 CASE，实际上是为软件开发提供的一组优化集成高效的软件开发工具。目前，软件开发环境进入了第三代 ICASE(Integrated Computer-Aided Software Engineering)。系统集成方式从数据交换到公共用户界面，再到信息中心库方式，不仅提供数据集成和控制集成，还提供了一组用户界面管理设施和一大批工具，如垂直工具集(支持软件生存期各阶段，保证生成信息的完备性和一致性)、水平工具集(用于不同的软件开发方法)以及开放工具。ICASE 的进一步发展则是与其他软件开发方法的结合，如与面向对象技术、软件重用技术结合，以及智能化的 ICASE。近几年已出现了能实现全自动软件开发的 ICASE。最终目标是实现应用软件的全自动开发，即开发人员只要写好软件的需求规格说明书，软件开发环境就自动完成从需求分析开始的所有的软件开发工作，自动生成供用户直接使用的软件及有关文档。

在应用最成熟的数据库领域，目前已有能实现全部自动生成的应用软件，如 MSE 公司的 Magic 系统。它只要求软件开发人员填写一系列表格(相当于要求软件实现的各种功能)，系统就会自动生成应用软件，可节省 90％以上的软件开发和维护的工作量，还能

将应用软件的开发工作转交给熟练的用户。

软件开发人员在软件开发的各个阶段还可根据不同的需要，选择不同合适的工具。目前，软件工具发展迅速，目标是实现软件研发各阶段的自动化、智能化和集成化。

3. 软件开发环境

1985 年在第八届国际软件工程会议上，对**软件开发环境**（Software Development Environment）下的定义为："软件开发环境是相关的一组软件工具集合，它支持一定的软件开发方法或按照一定的软件开发模型组织而成。"软件开发环境也称为**软件工程环境**（Software Engineering Environment），是包括方法、工具和管理等多种技术的综合系统。其设计目标是简化软件开发过程，提高软件开发的质量和效率。

软件开发环境应具备以下**特点**：

（1）适应性。适应用户要求，环境中的工具可修改、增加、减少和更新。

（2）坚定性。环境可自我保护，不受用户和系统影响，可进行非预见性的环境恢复。

（3）紧密性。各种软件工具可以密切配合工作，提高效率。

（4）可移植性。指软件工具可以根据需要进行移植。

常用的软件工程环境具有**三级结构**，如图 1-3 所示。

（1）核心级。主要包括核心工具组、数据库、通信工具、运行支持、功能和与硬件无关的移植接口等。

（2）基本级。一般包括环境的用户工具、编译、编辑程序和作业控制语言的解释程序等。

（3）应用级。通常指应用软件的开发工具。

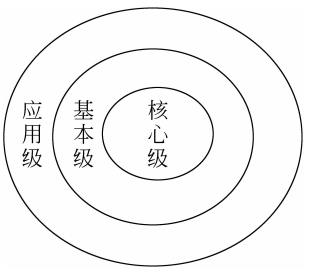

图 1-3　典型的软件工程环境

CASE 是一组工具和方法的集成，是多年来在软件工程管理、开发方法、开发环境和软件工具等方面研究与发展的成果，吸取了 CAD（计算机辅助设计）、软件工程、操作系统、数据库、网络和其他计算机领域的原理和技术，是对软件方法的辅助。

4. 软件工程管理

软件工程管理学包括软件管理学、软件经济学和软件度量学。其**目的**是按项目申报时确定的时间、费用和其他指标，通过有效管理提高软件研发的质量和效率。软件工程管理的**任务**是有效地组织人员，按照适当的技术、方法，利用好的软件工具"又好又快"地完成预定的软件项目。相关内容有专门书籍资料进行详细介绍，本书仅概述一些有关软件工程管理的内容，在第 8 章单独介绍软件项目管理等。美国软件项目管理学会从八个知识方面描述了项目管理知识体系的应用，分别是范围管理、时间管理、成本管理、质量管理、人力资源管理、交流管理、风险管理、采购与分包管理。

软件工程管理的**主要内容**包括软件人员组织、计划管理、费用管理、软件配置管理等。

（1）人员组织。软件开发需要团队合作，应有良好的组织、周密的管理，各类人员优化组合，协同配合，共同完成工程任务，这是软件开发项目成功的重要保证。

（2）计划管理。在软件开发前进行可行性研究立项后，还需要确定软件工程计划并

进行落实。在计划实施过程中，必要时可根据需要对工程进度进行适当调整。开发结束后完成软件开发总结报告，以总结经验并备用，为以后制定出更合适的软件开发计划打好基础。

(3) 费用管理。开发软件项目是一种投资，往往期望获得较大的经济、社会和应用效益，并降低成本。应从软件开发成本、运行费用、经济效益等多方面预算整个软件开发的投资和效益预测情况。

(4) 软件配置管理。指在系统整个开发、运行和维护时控制配置的状态和变动，验证配置项的完整性和正确性等。在软件工程各阶段所产生的文档和软件本身构成软件配置，每完成一个工程步骤应及时进行对应的软件工程配置，使其始终能够保持精确性。

1.2.4 软件过程及开发过程

ISO 9000 将**软件过程**(Software Process)定义为："将输入转化为输出的一组彼此相关的资源和活动。"软件过程是软件开发过程的简称，是为了获得高质高效软件所需要完成的一系列任务的框架，规定了完成各项任务的具体步骤。定义了运用方法的顺序、交付的文档、开发软件的管理措施和各阶段任务完成的标志。软件过程是软件工程方法学的三个要素中方法和工具的重要基础，必须科学合理才能获得高质量的软件产品。而软件工程过程则包括软件的开发过程、运作过程、维护过程。

根据合同、管理、工程和支持这四种观点可将软件过程分为获取过程、供应过程、管理过程、开发过程、运作过程、维护过程和支持过程。为了具体实现软件过程，需要根据软件项目需求等实际情况，在软件生存周期内，通过确定软件开发模型，将方法和技术相结合，在软件工具的支持下依次进行开发进程并循序渐进。

软件过程通常包括 4 类**基本过程**。

(1) 软件规格说明：规定软件的功能、性能、可靠性及其运行环境等。

(2) 软件开发：研发满足规格说明的具体软件。

(3) 软件确认：确认软件能够完成客户提出的需求。

(4) 软件演进：为满足用户的变更要求，软件必须在使用过程中引进新技术、新方法并根据新业务及时升级更新。

软件过程具有可理解性、可见性(过程的进展和结果可见)、可靠性、可支持性(易使用 CASE 工具支持)、可维护性、可接受性(为软件工程师所接受)、开发效率和健壮性(抵御外部意外错误的能力)等特性。

为了有效运用软件工程技术，软件过程定义了一个关键区域(阶段)的划分，如分析、设计、编程、测试等阶段，软件过程的阶段构成了软件项目开发控制和管理的基础，确立了整个过程各阶段之间的关系，包括技术方法的应用、工程产品(模型、数据、文档等)的形成、质量保障和开发进程管理。

软件工程最注重软件过程中的开发过程，主要包括项目启动、需求调研分析、设计(概要设计及详细设计)、编码(实现)、测试、部署、测试和结束等过程，如图 1-4 所示。

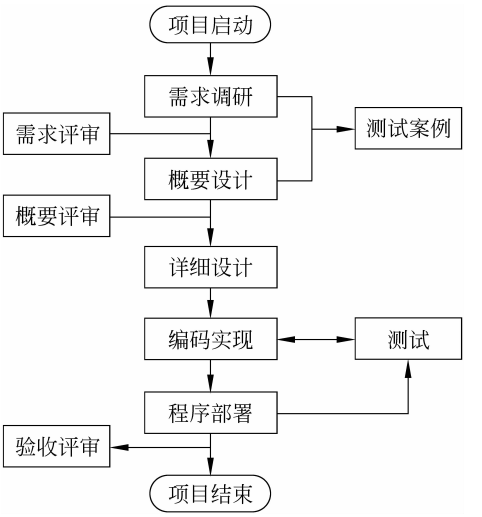

图 1-4　系统开发过程

【案例 1-4】“企业人事管理信息系统”总体功能需求和目标要求。其主要功能是用于支持企事业单位完成劳动人事管理工作，实现的主要目标包括：

(1) 支持企业高效率完成劳动人事管理的日常业务，包括新职员调入时人事的管理，以及职员调出、辞职、退休等的管理。

(2) 支持企业进行劳动人事管理及其相关方面的科学决策，如企事业单位领导根据现有的岗位员工需求情况决定招聘的岗位及人数等。

根据新系统总体功能需求等要求，通过调研、论证可以基本确定系统开发过程的总体框架。

对于软件开发过程的具体工作任务、参与人员及生成文档或程序，可以通过一个表具体详细地列出来，以便于清楚各阶段具体做什么工作，如表 1-4 所示。

表 1-4　软件开发工作任务、人员及输出

步　骤	任务及说明	参与者	生成文档或程序
可行性分析	对项目的技术、功能需求和市场进行调研和初步分析，确定是否需要启动项目	部门主管 核心技术人员	可行性分析报告 技术调研报告
启动项目	正式启动项目，由部门主管指定项目经理。项目经理制定初步计划，初步计划包括设计和开发时间的初步估计	部门主管 核心技术人员	项目计划书 项目合同
需求分析	对项目进行详细需求分析，编写需求文档，对于 B/S 结构的系统应制作静态演示页面。需求分析文档和静态演示页面需要通过部门主管审批才能进行到下一步骤	项目经理 项目小组核心成员	需求分析说明书 静态演示页面 项目计划修订版本

续表

步　骤	任务及说明	参与者	生成文档或程序
概要设计	根据需求分析进行概要设计。编写目的是说明对系统的设计考虑，包括程序系统流程、组织结构、模块划分、功能分配、接口设计、运行设计、数据结构设计和出错处理设计等，为详细设计提供基础。概要设计经过评审后，项目经理会同部门主管一起指定项目小组成员	项目经理 项目小组核心成员	概要设计说明书
详细设计	详细设计编制目的是说明一个软件各个层次中的每一个程序（每个模块或子程序）的设计考虑，如果一个软件系统比较简单，层次很少，可以不单独编写，有关内容合并入概要设计说明书	项目经理 项目小组成员	详细设计文档项目计划确定版本
编码实现	根据设计开发项目，同时由美工对操作界面进行美化	项目经理、程序设计员、美工	项目计划修订版本
测试	项目经理提交测试申请，由测试部门对项目进行测试，项目小组配合测试部门修改软件中的错误	项目经理 程序开发人员 测试部门	测试申请 测试计划 测试报告
项目验收	项目验收归档	部门主管 项目经理	项目的所有文档和程序

1.2.5　软件工程基本原理及原则

1. 软件工程的基本原理

著名软件工程专家 B. Boehm 综合有关专家和学者的意见并总结了多年来开发软件的经验，于 1983 年提出了互相独立、缺一不可的软件工程 7 条**基本原理**。

（1）用分阶段的生存周期计划进行严格的管理。不成功的软件项目中有大约一半是由于计划不周造成的。

（2）坚持进行阶段评审。软件的质量保证工作不能等到编码阶段结束之后再进行。

（3）实行严格的产品控制。在软件开发过程中不应随意改变需求，以免带来其他变更和为此付出较高的代价。

（4）采用现代程序设计技术。采用先进的技术既可提高软件开发的效率，又可提高软件维护的效率。

（5）软件工程结果应能清楚地审查。根据软件开发项目的总目标及完成期限，规定开发组织的责任和产品标准，从而使得所得到的结果能够清楚地审查。

（6）开发小组的人员应该少而精。

（7）承认不断改进软件工程实践的必要性。不仅要积极主动地采纳新的软件技术，而且要注意不断总结经验。

B. Boehm 指出，遵循前六条基本原理，能够实现软件的工程化生产；按照第七条原理，不仅要积极主动地采纳新的软件技术，而且要注意不断总结经验。

2. 软件工程的基本原则

根据软件工程的基本原理并总结软件研发实际经验，围绕工程设计、工程支持和工程管理，需要注意以下 4 条**基本原则**：

(1) 选取适宜的开发模型。软件设计应权衡软硬件需求及其他因素间相互制约和影响，必须认识需求定义的易变性，采用适当的开发模型，保证软件产品满足用户需求。

(2) 采用合适的设计方法。软件设计中应考虑软件的模块化、抽象与信息隐蔽、局部化、一致性和适应性等特征。优选设计方法有助于实现这些特征，并达到软件工程的目标。

(3) 提供高质量的工程支撑。在软件工程中，软件工具与环境对软件过程的支持颇为重要。软件工程项目的质量与开销直接取决于对软件工程所提供的支撑质量和效用。

(4) 重视软件工程的管理。软件工程管理直接影响可用资源的有效利用，只有对软件过程进行有效管理，才能研发出满足目标的软件产品并提高软件生产效能。

近年来，印度的软件产业迅速发展，其成功关键是严格按照国际规范进行科学管理。一般教材主要讨论软件开发技术，较少讨论软件管理技术，而软件管理仍然是软件开发成功的关键因素之一。在实际开发过程中，同时还应兼顾具体开发原则：抽象(abstraction)、信息隐藏(information hiding)、模块化(modularity)、局部化(localization)、一致性(consistency)、完整性(completeness)和可预测性(verifiability)。

课堂讨论：

(1) 软件和软件工程的概念是什么？软件工程方法学的定义是什么？软件工程三要素是什么？

(2) 软件工程开发的方法主要有哪些？

(3) 结合“人事管理信息系统”案例讨论软件工程目标。

1.3 软件生存周期

1.3.1 软件生存周期的概念

软件生存周期(Software Life Cycle)是指从开始研发软件到软件停止使用的整个过程，即软件产品从用户提出开发需求开始，经过开发、使用和维护，直到最后淘汰的整个周期，因此也称为软件生命周期或软件生存期，是软件工程的一个重要概念。

软件工程中的过程对应软件生存周期中的**阶段**(Phase)，也是实现软件生产工程化的重要步骤，并赋予各阶段相对独立的任务。可以将一个软件的生存周期划分为市场调研、立项、需求分析、规划、概要设计、详细设计、编程、单元测试、集成测试、运行、维护这几个过程，前一过程的终点就是后一过程的起点。完成阶段性工作的标志称为**里程碑**(Milestone)，某些重要的里程碑又称为**基线**(Baseline)。

软件开发过程中每一阶段的工作都应以前一阶段的结果为依据，并作为下一阶段的前提。每个阶段结束时都要有技术审查和管理复审，从技术和管理两方面对这个阶段的

开发成果进行检查，及时决定工作是否继续、停工或返工，主要检查是否有高质量的文档资料。前一个阶段复审通过了，后一个阶段才能开始。应防止开发到最后，才发现前期工作存在严重问题，造成难以挽回的损失或失败。

1.3.2 软件生存周期的阶段划分

软件生存周期划分阶段的方法有多种，可按软件规模、种类、开发方式、开发环境等来划分。划分阶段的原则是相同的，目的主要是便于确立系统开发计划，明确各类开发人员的分工与职责范围，以便选用不同的开发模型、技术方法，加强管理、分工协作、保证质量、提高效率。开发单位的技术人员可根据所开发软件的性质、用途及规模等因素决定在软件生存周期中增加或减少相应的阶段。

软件生存周期阶段划分的原则主要包括：

(1) 各阶段的任务相对独立，便于分阶段进行计划，逐步完成。

(2) 同一阶段的工作任务性质尽量相同，有利于软件开发和组织管理，明确开发人员的分工与职责，以便协同工作，保证质量。

1.3.3 软件生存周期各阶段的任务

软件生存周期一般由软件策划、软件开发和运行维护三个时期组成。软件策划时期分为问题定义、可行性研究和需求分析三个阶段。软件开发时期可分为需求分析、软件设计、软件实现和综合测试阶段。其中，软件设计阶段可分为软件概要设计和详细设计阶段，软件实现阶段进行程序设计和软件单元测试，最后进行综合测试等。软件交付使用后，在运行过程中需要不断地进行维护才能使软件持久地满足用户的需要。

下面简要介绍软件生存周期各阶段的**主要任务**。

在 GB 8567—2006 中将软件生存周期分为 7 个阶段，如图 1-5 所示。

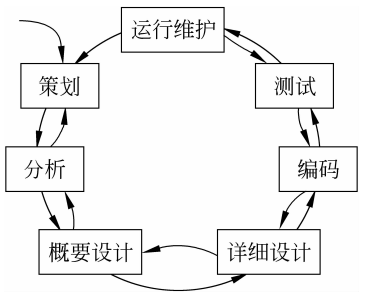

图 1-5 软件生存周期各阶段的关系

(1) 开发策划。主要完成问题定义、可行性论证、制定开发计划和项目申报工作，明确“要解决的问题是什么”。

(2) 需求分析。需求分析和定义阶段的任务不是具体地解决问题，而是确定软件须具备的具体功能、性能等，即明确“必须做什么”及其他指标要求。

(3) 概要设计。主要设计软件的结构，结构的组成模块，模块的层次结构、调用关系及功能，并设计总体数据结构等。

(4) 详细设计。对模块功能、性能、可靠性等进行具体技术描述，并转化为过程描述。

(5) 编写程序。又称编码，将模块的控制结构转换成程序代码。

(6) 测试。为了保证软件需求和质量，在设计测试用例的基础上对软件进行检测。

(7) 运行维护。对交付并投入使用的软件进行各种维护，并记录保存文档。

【案例 1-5】 “网上商品销售管理信息系统”由项目问题定义(调研论证计划申报)、软件开发和软件维护三个时期组成,每个时期又可进一步划分成若干个阶段。

1. 软件定义时期

(1) 问题定义。这是软件生存周期的第一个阶段,主要任务是弄清用户要计算机解决的问题是什么。

(2) 可行性研究。任务是针对前一阶段提出的问题,寻找技术上可行,且在经济上有较高效益的解决方案。

2. 软件开发时期

(1) 需求分析。通过调研搞清用户对软件系统的具体需求,主要是确定目标系统必须具备哪些具体的功能、性能、可靠性、接口等。

(2) 总体设计。设计软件结构,即确定程序由哪些模块组成以及模块间的关系。

(3) 详细设计。即针对单个模块的设计,如查询、统计等。

(4) 编码。按照选定的语言,把模块的过程性描述翻译为源程序。

(5) 测试。通过各种类型的测试及相应的调试使软件达到预定的要求。

3. 软件运行时期

此时期的主要工作是做好软件维护与管理工作。维护与管理的目的是使软件在整个生存周期内保证满足用户的正常使用和延长软件的使用寿命。

课堂讨论:

(1) 什么叫软件生存周期?软件生存周期各阶段如何划分?

(2) 软件生存周期各阶段的主要任务有哪些?

(3) 结合“企业人事管理信息系统”案例进行阶段划分,并指出各阶段的主要任务。

1.4 软件开发模型

根据软件开发工程化及实际需要,软件生存周期的划分有所不同,形成了不同的**软件开发模型**,或称软件生存周期模型(Software Life Cycle Model)。模型通常是对现实系统本质特征的一种抽象、模拟、简化和描述,用于表示事物的重要方面和主要特征,包括描述模型、图表模型、数学模型和实物模型。软件开发模型可分为瀑布模型、快速原型模型、增量模型、喷泉模型、螺旋模型、变换模型、基于知识的模型和统一过程等。

1.4.1 瀑布模型

瀑布模型(Waterfall Model)是 1970 年 W. Royce 最早提出的软件开发模型。将软件开发过程划分为几个互相区别且彼此相联的阶段,各阶段的工作都以上一个阶段工作的结果为依据,并作为下一阶段的工作基础,形如瀑布流水承前启后。

瀑布模型将生存期的计划时期、开发时期和运行时期又细分为若干个阶段:计划时期可分为问题定义、可行性研究、需求分析 3 个阶段,开发时期分为概要设计、详细设计、

软件实现、软件测试等阶段，运行时期则需要不断进行运行维护，需要不断修改错误、排除故障，或根据用户需求、运行环境改变进行更改调整。图 1-6 中的实线箭头表示开发流程，每个阶段顺序进行，有时会返工；虚线箭头表示维护工作的流程，根据不同情况返回到不同的阶段进行维护。

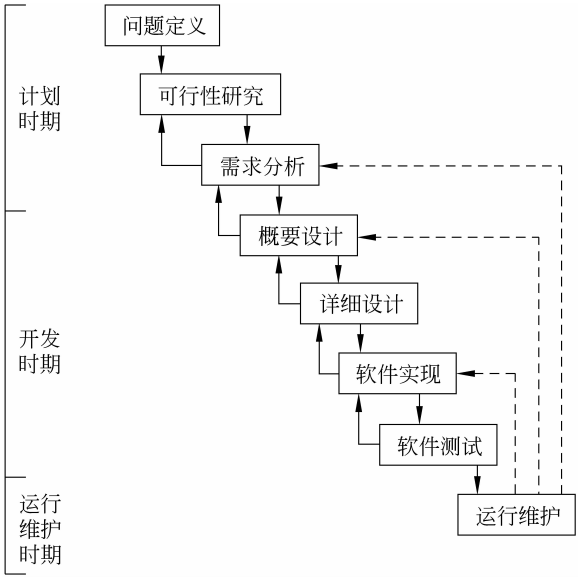

图 1-6　瀑布模型

利用瀑布模型开发软件有 3 个特点：

(1) 开发过程的顺序性。此模型特点是文档驱动。只有当前一阶段任务完成后，下一阶段的工作才能开始；前一阶段的输出文档作为后一阶段的输入文档。前面的正确输出决定后面结果的正确性，若在某一阶段出现错误，则要向前追溯返工。

瀑布模型开发适合于应用软件需求明确、开发技术成熟、工程管理较为严格的场合下使用。

(2) 统筹兼顾不过早编程。在编程前安排的需求分析、概要设计、详细设计等阶段，将逻辑设计和编码清楚地划分开来，以便协同工作效果更好。实践表明，大、中型软件编程开始得越早，完成所需的时间反而越长。

(3) 严格要求保证质量。为确保质量，应坚持做到：

① 必须各阶段都按照要求认真完成规定的文档。

② 各阶段须对完成文档复审，及时发现隐患并排除。

瀑布模型的缺陷是将充满回溯且相互重叠的软件开发过程硬性地分为多个阶段，随着开发软件规模的增加，造成的危害大增。如图 1-7 的循环模

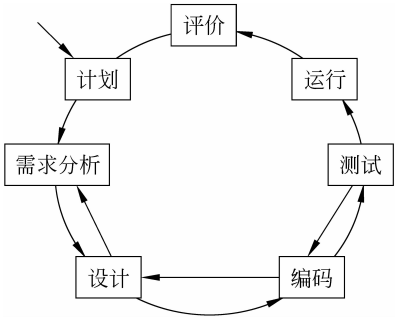

图 1-7　循环模型

型，是为了描述软件开发过程中可能的回溯，对瀑布模型进行了改进，开发各阶段可能循环重复。

1.4.2 快速原型模型

快速原型模型需要先建造一个快速原型，如操作窗口及界面等，进行客户或潜在用户与系统间的交流，用户/客户可以通过对原型的评价及改进意见，进一步细化待开发软件的需求，通过逐步调整原型达到客户要求，从中确定客户的具体需求；然后按照需求开发软件，如图 1-8 所示。此模型最适合于可以先尽快构建成一个原型的应用系统。

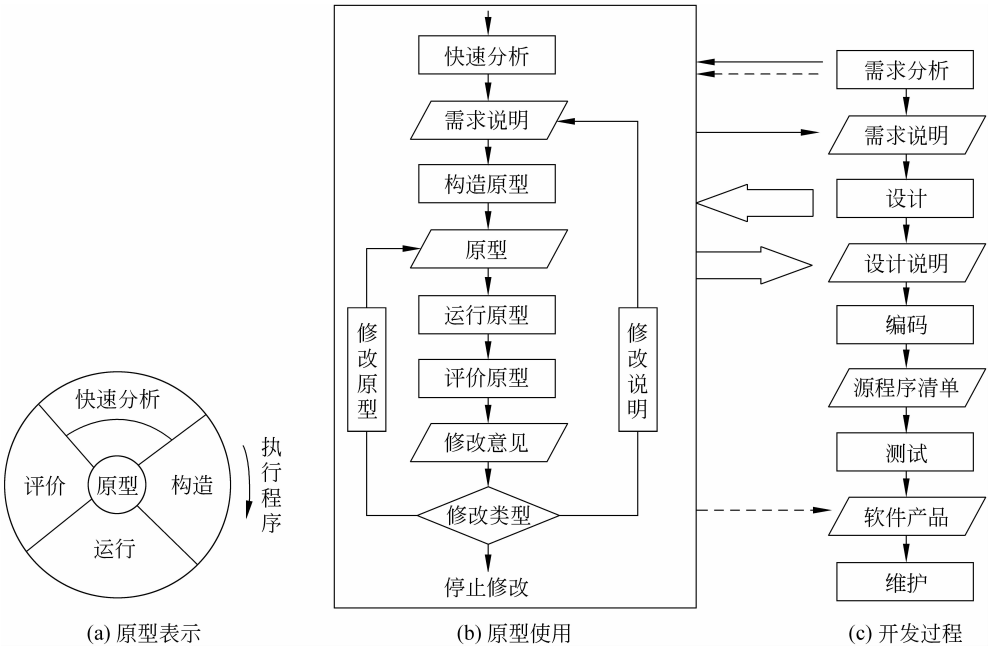

图 1-8 快速原型模型

1.4.3 增量模型

将利用**增量模型**开发的软件作为一系列增量构件来进行设计、实现、集成和测试，每个构件具有一定功能，并最终能组合成一个具有完整功能软件的模块，如图 1-9 所示。

增量模型灵活性很强，适用于软件需求不明确、设计方案有一定风险的软件项目。它与瀑布模型之间的本质区别是：瀑布模型属于整体开发模型，规定在开始下一个阶段的工作之前，必须完成前一阶段的所有细节。而增量模型属于非整体开发模型，可推迟某些阶段或所有阶段中的细节，从而较早地研发出软件。

增量模型的**缺陷**有两个方面：

(1) 需要软件具备开放式的体系结构。主要是因为各构件是逐渐并入已有的软件体系结构中的，所以加入构件不能破坏已构造好的系统部分。

(2) 软件过程的控制易失去整体性。软件在开发中难免需求的变化，增量模型的灵