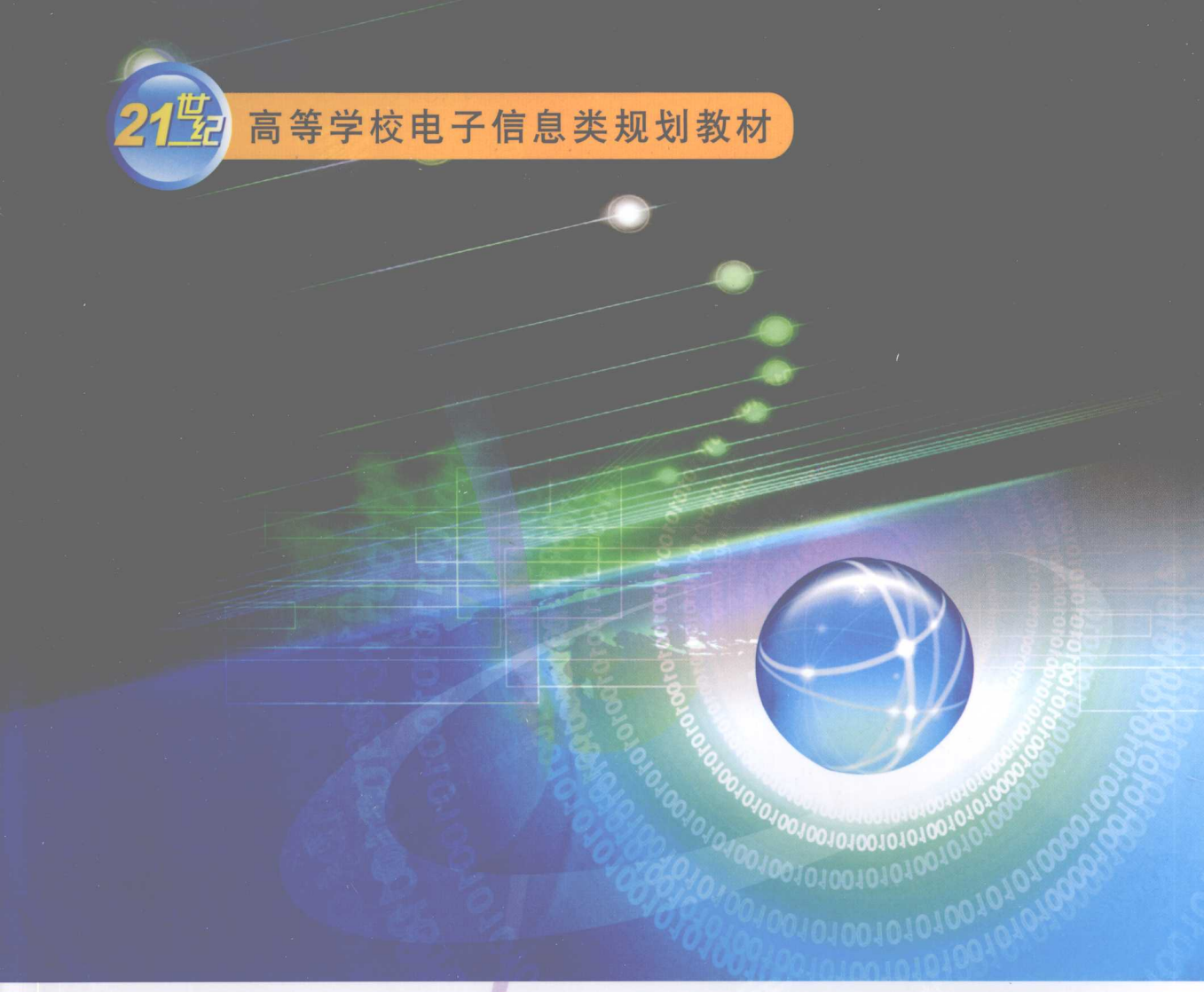
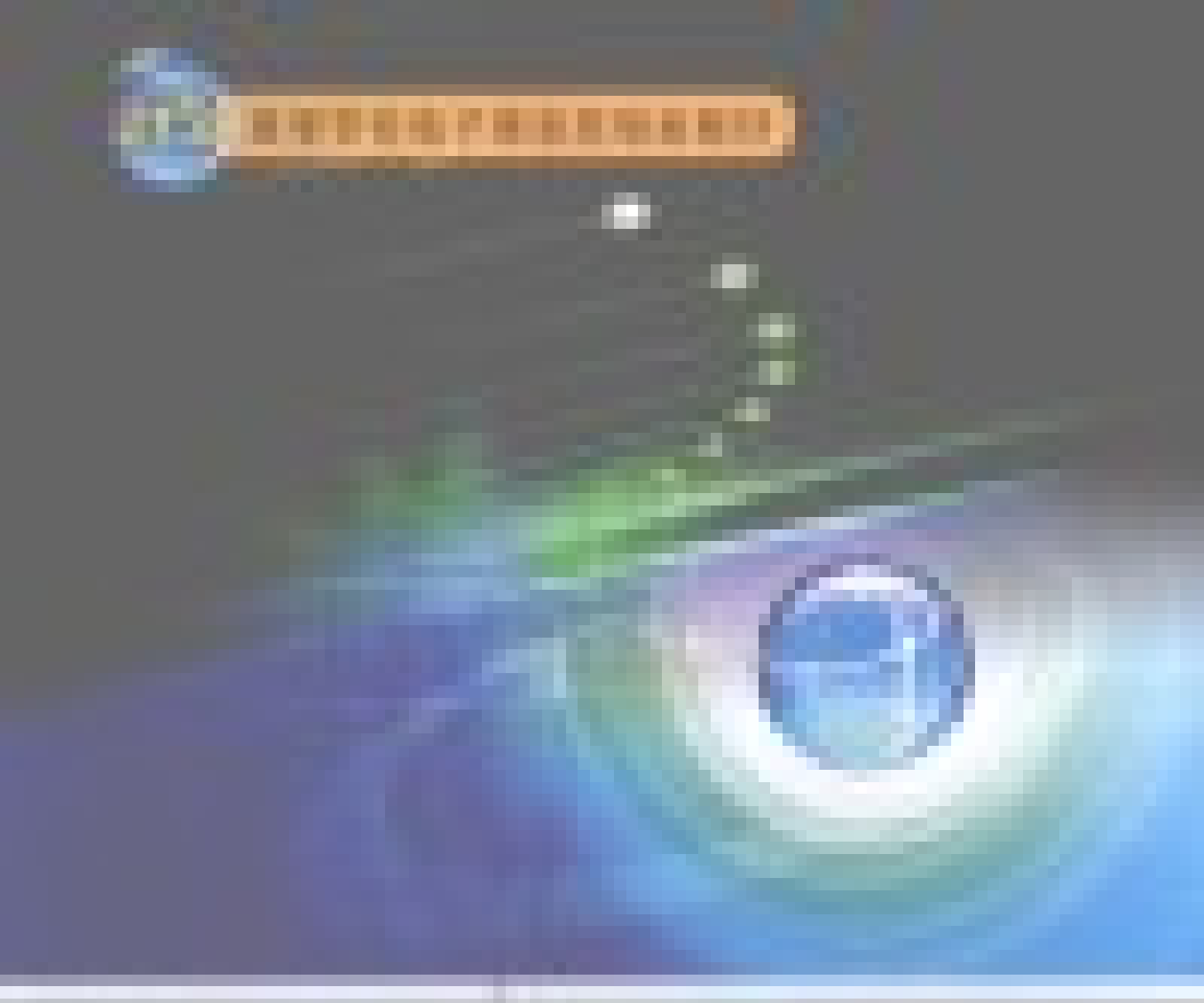


软件技术基础

周大为　钟　桦
朱虎明　潘晓珠
姚若玉　编著
李伯成　主审

西安电子科技大学出版社
http://www.xduph.com



软件技术基础

21 世纪高等学校电子信息类规划教材

软件技术基础

周大为　钟　桦
朱虎明　潘晓珠
姚若玉　编著

李伯成　主审

西安电子科技大学出版社

2008

内 容 简 介

本书是高等学校非计算机专业继计算机文化与计算机语言类课程之后的第二层次的教材，旨在学生掌握了上述课程的知识后，继续以更具体、更深层次的课程教授使学生掌握最新、最实用的计算机软件基础知识。全书介绍了软件工程、数据结构、操作系统和数据库系统及其应用等四部分内容，在介绍各部分内容的同时，给出了相关知识的应用实例，具有较高的实用价值。

本书可供高等学校本、专科非计算机专业开设计算机软件技术基础课程之用，也可供自学相关知识的读者参考。

★本书配有电子教案，需要的老师可与出版社联系，免费提供。

图书在版编目(CIP)数据

软件技术基础 / 周大为等编著. —西安：西安电子科技大学出版社，2008.8

21 世纪高等学校电子信息类规划教材

ISBN 978-7-5606-2115-9

Ⅰ. 软… Ⅱ. 周… Ⅲ. 软件—高等学校—教材 Ⅳ. TP31

中国版本图书馆 CIP 数据核字(2008)第 122130 号

策　　划　毛红兵

责任编辑　邵汉平　毛红兵

出版发行　西安电子科技大学出版社(西安市太白南路 2 号)

电　　话　(029)88242885　88201467　邮　编　710071

http: //www.xduph.com　　E-mail: xdupfxb@pub.xaonline.com

经　　销　新华书店

印刷单位　陕西光大印务有限责任公司

版　　次　2008 年 8 月第 1 版　2008 年 8 月第 1 次印刷

开　　本　787 毫米×1092 毫米　1/16　印 张 21.375

字　　数　506 千字

印　　数　1～4000 册

定　　价　30.00 元

ISBN 978 - 7 - 5606 - 2115 - 9/TP • 1084

XDUP 2407001-1

前　言

为了实现教育部关于“加强非计算机专业计算机基础教学工作的几点意见”所提出的目标，切实将非计算机专业的计算机课程体系的改革落到实处，我们组织部分长期在教学一线、具有丰富教学经验的教师编写了本书。

计算机软件技术基础是计算机文化基础和程序设计语言的后续课程。对于计算机软件技术基础课程而言，各学校、各专业的课程内容设置和教学目标都不尽相同，因而教材的组成部分也有所不同。我们结合计算机应用领域的发展情况，本着“加强基础，注重应用”的原则，选择软件工程、数据结构、操作系统和数据库系统及应用等四个部分作为本书的内容。我们期待学生在学完本书之后，能够掌握更具体、更深层次的计算机软件基础技术，进而开发出一些小型实用的软件系统。

软件工程部分从软件开发的方法学角度出发，介绍了软件工程的过程和软件生存周期的各种模型，对软件系统的开发起到指导性的作用。本部分以软件生命周期的各个阶段和结构化的软件开发方法为主线，介绍了软件系统的分析与定义、软件设计、软件编码、软件测试和软件维护的有关概念和软件工程方法，还介绍了面向对象的软件开发方法的基本思想、基本概念和基本原理，以及面向对象的分析、设计方法。

数据结构部分主要讲述线性表、栈、队列、串、数组、树和图等数据的逻辑结构、存储结构及有关的算法，还讲述了数据的查找和排序方法。采用类 C 语言作为数据的存储结构和算法的描述语言，并且尽可能做到与 C 语言接近，以便于将算法转换为能够上机执行的 C 程序。数据结构的学习过程也是较复杂的程序设计的训练过程，学生在通过编写程序来解决实际问题时，应当采用规范的算法，并且按照软件开发方法所要求的模块独立性高的原则，设计出高质量的程序。

操作系统部分以阐述操作系统原理为主，分别介绍了批处理操作系统、分时操作系统和实时操作系统的概念和特点，围绕资源管理的观点分别讲述了处理机管理、存储管理、文件管理、设备管理和作业管理。同时还简要讲述了目前广泛使用的 Windows、UNIX 和 Linux 几种典型的操作系统，尽可能反映出当代操作系统的新技术和新特点。操作系统是计算机系统中最主要的系统软件，学习操作系统不仅有助于使用计算机，更有助于开发一些软件系统。

数据库系统及应用部分包括六个方面的内容：(1) 数据库系统的基础知识；(2) 与关系数据库有关的实体关系模型、关系表、关系代数，以及关系数据库规范化理论的知识；(3) 关系数据库应用系统设计、开发方法，以及数据库系统开发的各个阶段所要完成的工作；(4) 结合 Access 讲述了数据库建立的方法；(5) 数据库的各种查询方式以及结构化查

询语言(SQL 语言)的查询功能；(6) 以网络教学管理信息系统为例，讲述了网络管理信息系统的基本知识和开发方法。

本书第 1 章由姚若玉编写，第 2、3、4、5 章由周大为编写，第 6、7、8、9 章由钟桦编写，第 10 章由潘晓珠编写，第 11 章由朱虎明编写。周大为对全书进行了校对和统稿。

本书在编写过程中得到了西安电子科技大学出版社的支持与合作；西安电子科技大学石光明教授对本书的出版给予了大力支持，并提出了许多建设性的意见；李伯成教授审阅了全稿，并提出了许多宝贵意见。在此，一并表示衷心的感谢。

本书涉及的内容较广，书中难免存在不足之处，敬请广大读者提出宝贵意见和建议。

编　者

2008 年 5 月

目　录

第1章 软件工程

随着计算机技术的发展和日趋成熟，软件技术已在计算机领域占有了相当重要的地位。目前，软件工程已发展到第四代，在整个产业界，“软件工程师”已替代“程序员”而成为更受欢迎的工作头衔。在应用软件中，已广泛而成功地采用了软件过程模型、软件工程方法以及软件工具。管理者和实践者均认为，学习和掌握软件技术是非常必要的。

20世纪末最受瞩目的无疑是席卷全球的信息技术(IT)革命，人们将这场革命视为21世纪——知识经济时代的前奏曲。在这场IT革命中，软件无疑扮演了极其重要的角色。软件产业作为一个独立形态的产业，正在全球经济中占据着越来越重要的地位，而软件工程正是软件产业健康发展的关键技术之一。从1968年软件工程概念的正式提出到现在，软件工程已有30多年的发展历程，产生了大量的研究成果，也进行了大量的技术实践。正是学术界和产业界的共同努力，使软件工程逐步发展成为一门成熟的专业学科。该学科以解决软件生产的质量和效率问题为宗旨，在软件产业的发展中起到了重要的技术保障和促进作用。

1.1 软件的基本概念

关于软件是这样定义的：软件是能够完成预定功能和性能的可执行的指令(计算机程序)，使得程序能够适当地操作信息的数据结构，描述程序操作和使用的文档的集合。

1.1.1 软件的特征

要理解软件的含义，首先要了解软件的特征。软件是逻辑的而不是物理的产品，因此，软件具有与硬件完全不同的特征。

(1) 软件的开发不是传统意义上的生产制造。

软件开发与硬件制造之间有一些相似之处，但却有着本质的区别。软件产品的生产主要是脑力劳动、手工开发方式，大部分产品是“定做”的。

(2) 软件不会“磨损”。

硬件在其生命初期有较高的故障率(这些故障主要是设计或制造的缺陷)，这些缺陷修正之后，故障率在一段时间内会降到一个很低的水平；随着时间的推移，故障率又将提升。硬件故障率的变化曲线如图1.1所示。

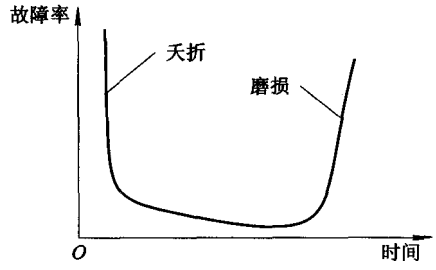


图1.1 硬件故障率的变化曲线

软件并不受引起硬件磨损的环境因素的影响。因此，理论上讲，软件的故障率曲线呈现出如图 1.2 所示的形式。软件在其生命初期具有较高的故障率，但在这些错误改正之后(我们假设理想情况下改正过程中并不引入其他错误)，曲线就会趋于平稳。软件不会被磨损，不过它会退化，这一点可以通过图 1.2 来解释清楚。在其生命期中，软件会经历修改(维护)，随着这些修改有可能会引入新的错误，使得故障率曲线呈现为图 1.2 所示的锯齿形。

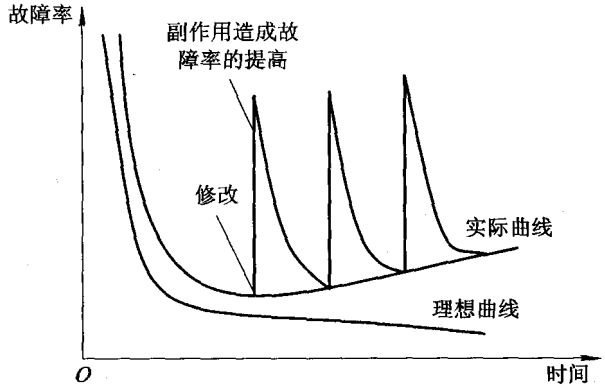


图 1.2　软件故障率的理想曲线与实际曲线

硬件和软件之间的不同还表现在当一个硬件构件磨损时，可以用另外一个备用零件替换它，但对于软件就没有备用零件可以替换了。每一个软件故障都表明了设计或是将设计转换成机器可执行代码的过程中存在错误。因此，软件维护要比硬件维护复杂得多。

(3) 软件的可复用性差，不能通过已有的构件组装而成。

我们先来看一下一个基于微处理器的控制硬件是如何设计和建造出来的。设计工程师画一个简单的数字电路图，做一些基本的分析以保证可以实现预定的功能，然后查阅所需的元器件的目录，每一个集成电路(通常称为“IC”或“芯片”)都有一个零件编号、固定的功能、定义好的接口和一组标准的集成指南，每一个选定的零件都可以买到，由此可以很方便地组装起一个基于微处理器的控制硬件。

在软件开发中，采用一些已有的构件组建一个软件的方法，仅仅在小范围内得到应用。多数软件的设计开发还必须完全从零开始。

1.1.2　软件的分类

软件种类繁多，概括起来可分为两类：系统软件和应用软件。

1．系统软件

系统软件是指操作系统及与之相关的各种软件的总称。系统软件是一组为其他程序服务的程序。一类系统软件(如编译器、编辑器和文件管理程序)所处理的信息结构是复杂的，但又是确定的；还有一类系统软件(如操作系统、驱动程序和通信进程等)则处理大量的非确定的信息。系统软件具有以下特点：

- 与计算机硬件频繁交互；
- 支持多用户；
- 需要精细调度、资源共享及灵活的进程管理的并发操作；

- 复杂的数据结构；
- 多外部接口；
- 具有可移植性，例如嵌入式系统中的实时操作系统。

常见的操作系统有 DOS、UNIX、Linux 以及 Windows。

2. 应用软件

应用软件是指为用户的特殊应用目的而开发的软件。例如财务管理软件、人力资源管理软件。

1.1.3 软件的发展

今天，软件担任着双重角色，它是一种产品，同时又是开发和运行产品的载体。作为一种产品，它扩充了计算机硬件的功能；作为开发和运行产品的载体，它是计算机控制(操作系统)的基础、信息通信(网络)的基础，也是创建和控制其他程序(软件工具和环境)的基础。

计算机硬件的发展经历了四个时代，同样，计算机软件的发展也大致经历了四个阶段。

(1) 20 世纪 60 年代中期以前，是计算机系统发展的早期阶段。

在计算机发展的早期阶段，大多数人把软件开发看成是不需预先计划的事情。计算机编程很简单，没有什么系统化的方法。软件的开发没有任何管理，一旦进度拖后了或者成本提高了，程序员才开始手忙脚乱地弥补。由于那个时期的软件很简单，因而他们的努力在一般情况下往往也会见效。

当通用的硬件已经非常普遍的时候，软件却相反，对每一类应用均需自行设计，应用范围很有限。此时，软件产品还处在婴儿阶段，大多数软件均是由使用它们的人员或组织自行开发的，软件在使用过程中出现了问题，编写软件的人员必须负责修改。在这种个人化的软件开发环境中，设计往往仅是人们头脑中的一种模糊想法，而软件文档根本就不存在。在早期，我们了解了很多关于计算机系统的实现，但对于计算机系统工程却几乎一无所知。

(2) 从 20 世纪 60 年代中期到 70 年代中期，是计算机系统发展的第二阶段。

计算机系统发展的第二阶段跨越了从 20 世纪 60 年代中期到 70 年代中期的十余年。其主要特征是：多道程序设计、多用户系统引入了人机交互的新概念；交互技术打开了计算机应用的新世界及硬件和软件配合的新层次；实时系统能够从多个源收集、分析和转换数据，使得进程的控制和输出的产生以毫秒而不是分钟来进行；在线存储的发展导致了第一代数据库管理系统的出现。

第二阶段的一个特点就是软件产品的使用和“软件作坊”的出现。随着计算机应用领域的扩大，来自工业界、政府和学术界的人们纷纷开始开发各类软件包，并取得了巨大的经济利益。

随着计算机系统的增多，新的问题出现了，当发现软件存在错误或缺陷时，需要纠正部分代码甚至全部代码；当用户需求发生变化时，需要对软件进行修改；当硬件环境更新时，需要修改软件以适应硬件的变化。这些活动统称为软件维护。在软件维护上所花费的精力开始以惊人的速度消耗资源。更糟糕的是，许多程序的个人化特性使得它们根本不能

维护。这就是所谓的“软件危机”。

(3) 计算机系统发展的第三个阶段从20世纪70年代中期开始，并且跨越了整整10年。

在分布式系统中，多台计算机之间的分布式处理和通信极大地提高了计算机系统的复杂性。广域网和局域网、高带宽数字通信以及对“即时”数据访问需求的增加都对软件开发者提出了更高的要求。然而，软件仍然继续应用于工业界和学术界，个人应用很少。第三阶段的另外一个特点是微处理器的出现和广泛应用，微处理器的出现使得一系列智能产品纷纷面世，大到汽车，小到微型医疗设备等，使计算机的应用真正成为大众化的应用。

(4) 计算机系统发展的第四个阶段已经不再着重于单台计算机和计算机程序，而是面向计算机和软件的综合影响。

20世纪70年代后由复杂的操作系统控制的功能强大的计算机、广域网和局域网，配以先进的应用软件已成为标准。计算机体系结构迅速从集中的主机环境转变为分布的客户机/服务器环境。世界范围的信息网提供了一个基本结构，使得“信息高速公路”和“网际空间连通”成为可能。事实上，Internet可以看做是能够被单个用户访问的“软件”。

软件产业在世界经济中不再是无足轻重的，由产业巨子(如微软)所做的一个决定可能会带来成百上千亿美元的风险。随着第四阶段的进展，一些新技术开始涌现。面向对象技术在许多领域中迅速取代了传统软件开发方法。虽然关于“第五代”计算机的预言仍是一个未知数，但是软件开发的“第四代技术”确实改变了软件界开发计算机程序的方式。专家系统和人工智能软件终于从实验室里走了出来，进入了实际应用，解决了现实世界中的大量问题。结合模糊逻辑应用的人工神经网络软件揭示了模式识别和类似人的信息处理能力的可能性。虚拟现实和多媒体系统使得与最终用户的通信可以采用完全不同的方法。“遗传算法”则提供了可以驻留于大型并行生物计算机上的软件的潜在可能性。

现将计算机软件的发展历程归纳于表1.1。

表1.1　计算机软件的发展历程

早　期	第二阶段	第三阶段	第四阶段
面向批处理	多用户	分布式系统	强大的桌面系统
有限的分布	实时	嵌入“智能”	面向对象技术
自定义软件	数据库	低成本硬件	专家系统
	软件产品	消费者的影响	人工神经网络
			并行计算
			网络计算机

1.1.4　软件危机

1. 软件危机的概念

软件危机是指在计算机软件的开发和维护过程中所遇到的一系列严重问题。这些问题不仅仅是不能正常运行的软件才具有的，实际上，几乎所有软件都不同程度地存在这些问题，如开发周期延长，成本增加，可靠性降低等。

具体来说，软件危机主要有以下一些典型表现：

(1) 对软件开发成本和进度的估计常常很不准确。

(2) 用户对“已完成的”软件系统不满意。

(3) 软件产品的质量不可靠。

(4) 软件常常是不可维护的。

(5) 软件通常没有适当的文档资料。

(6) 软件成本在计算机系统总成本中所占的比例逐年上升。

(7) 软件开发生产率提高的速度，既跟不上硬件的发展速度，也远远跟不上计算机应用迅速普及、深入的趋势。

以上列举的仅仅是软件危机的一些明显表现，与软件开发和维护有关的问题远远不止这些。

在软件产业中，“危机”已经伴随我们走过了30多年的历程。面向对象方法(OO)就是在这种背景下诞生的，它使业界看到了成功的希望，同时也促使OO方法和技术的研究得以迅速发展。

2. 产生软件危机的原因

在软件开发和维护的过程中之所以存在这么多问题，一方面与软件本身的特点有关，另一方面也和软件开发与维护的方法不正确有关。

与软件开发和维护有关的许多错误认识和做法的形成，可以归因于在计算机系统发展的早期阶段软件开发的个体化特点。错误的认识和做法主要表现为忽视软件需求分析的重要性，认为软件开发就是写程序并设法使之运行，轻视软件维护等。

了解产生软件危机的原因，澄清错误认识，建立起关于软件开发和维护的正确概念，仅仅是解决软件危机的开始，全面解决软件危机需要一系列综合措施。

3. 消除软件危机的途径

为了消除软件危机，首先应该对计算机软件有一个正确的认识，应该推广使用在实践中总结出来的开发软件的成功的技术和方法，并且研究探索更好、更有效的技术和方法，尽快消除在计算机系统早期发展阶段形成的一些错误概念和做法，应该开发和使用更好的软件工具。

总之，为了消除软件危机，既要有技术措施(方法和工具)，又要有必要的组织管理措施。软件工程正是从管理和技术两方面研究如何更好地开发和维护计算机软件的一门新兴学科。

1.2 软件工程

1.2.1 软件工程的基本概念

概括地说，软件工程是指导计算机软件开发和维护的工程学科。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正确的管理技术和方法与当前能够得到的最好的技术和方法相结合，经济地开发出高质量的软件并有效地维护它，这就是软件工程。

下面我们给出软件工程的基本原理，以期对软件工程的概念有更深刻的理解。

(1) 用分阶段的生命周期计划严格管理。

(2) 坚持进行阶段评审。

(3) 实行严格的产品控制。

(4) 采用现代程序设计技术。

(5) 结果应能清楚地审查。

(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。

1.2.2 软件工程方法学

通常把在软件生命周期全过程中使用的一整套技术的集合称为软件工程方法学。

软件工程方法学包括三个要素：方法、工具和过程。其中，方法是完成软件开发的各项任务的技术方法，回答“如何做”的问题；工具是为方法的运用提供自动的或半自动的软件支撑环境；过程是为了获得高质量的软件所需要完成的一系列任务的框架，它规定了完成各项任务的工作步骤。

软件工程方法学分为结构化方法、Jackson 方法、维也纳方法（VDM）、面向对象方法。目前使用最广泛的软件工程方法学是传统方法学和面向对象方法学。

1.2.3 软件工程的目标

软件工程的目标是提高软件的质量与生产率，最终实现软件的工业化生产。质量是软件需求方最关心的问题。生产率是软件供应方最关心的问题，老板和员工都想用更少的时间挣更多的钱。质量与生产率之间有着内在的联系，高生产率必须以质量合格为前提。从短期效益看，追求高质量会延长软件开发时间并且增大费用，似乎降低了生产率。从长期效益看，高质量将保证软件开发的全过程更加规范流畅，大大降低了软件的维护代价，提高了生产率，同时可获得很好的信誉。质量与生产率之间不存在根本的对立，好的软件工程方法可以同时提高质量与生产率。

质量与生产率的提高就是开发人员与项目经理共同努力的结果。对开发人员而言，如果非得在质量与生产率之间分个主次不可，那么应该是质量第一，生产率第二。这是因为：

(1) 质量直接体现在软件的每段程序中，高质量自然是开发人员的技术追求，也是职业道德的要求。

(2) 高质量对所有的用户都有价值，而高生产率只对开发方有意义。

(3) 如果一开始就追求高生产率，则容易留下隐患。宁可进度慢些，也要保证每个环节的质量，以图长远利益。

软件的质量因素很多，如正确性、性能、可靠性、容错性、易用性、灵活性、可扩充性、可理解性、可维护性等。有些因素相互重叠，有些因素则相互抵触，要提高质量是非常不容易的。

软件工程的主要环节有人员管理、项目管理，其中包括可行性与需求分析、系统设计、程序设计、测试、维护等，如图 1.3 所示。

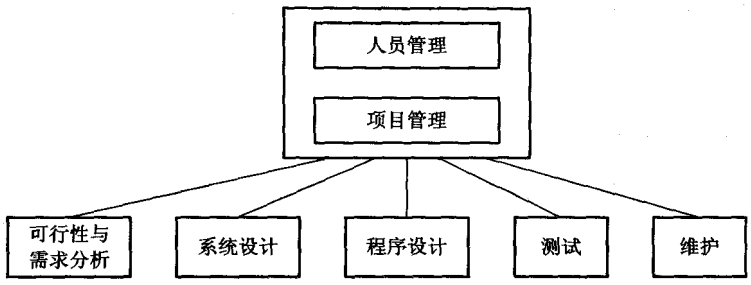


图 1.3 软件工程的主要环节

1.3 软件生存周期

1.3.1 生存周期的划分及各阶段的主要任务

概括地说，软件生存周期由软件定义、软件开发和运行维护三个时期组成，每个时期又可进一步划分成若干个阶段。下面简要介绍上述各个阶段应该完成的基本任务。

(1) 问题定义：问题定义阶段必须回答的关键问题是“要解决的问题是什么”。

(2) 可行性研究：这个阶段要回答的关键问题是“上一个阶段所确定的问题是否有行得通的解决办法”。

(3) 需求分析：这个阶段的任务仍然不是具体地解决客户的问题，而是准确地回答“目标系统必须做什么”这个问题。这个阶段的另外一项重要任务是用正式文档准确地记录对目标系统的需求，这份文档通常称为需求说明(specification)。

(4) 概要设计：这个阶段的基本任务是概括地回答“怎样实现目标系统”这个问题。概要设计又称为初步设计、逻辑设计、高层设计或总体设计。首先，应该设计出实现目标系统的几种可能的方案。概要设计的另一项主要任务就是设计程序的体系结构，也就是确定程序由哪些模块组成以及模块间的关系。

(5) 详细设计：概要设计阶段以抽象概括的方式提出了解决问题的办法。详细设计阶段的任务就是把解法具体化，也就是回答“应该怎样具体地实现这个系统”这个关键问题。这个阶段的任务还不是编写程序，而是设计出程序的详细规格说明。

(6) 编码：这个阶段的关键任务是写出正确的容易理解、容易维护的程序模块。

(7) 测试：这个阶段的关键任务是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。

(8) 软件维护：维护阶段的关键任务是，通过各种必要的维护活动使系统持久地满足用户的需要。通常有四类维护活动：改正性维护，也就是诊断和改正在使用过程中发现的软件错误；适应性维护，即修改软件以适应环境的变化；完善性维护，即根据用户的要求改进或扩充软件，使它更完善；预防性维护，即修改软件，为将来的维护活动做准备。

在实际从事软件开发工作时，软件规模、种类、开发环境及开发时使用的技术方法等因素会影响阶段的划分。

生命周期模型规定了把生命周期划分成哪些阶段及各个阶段的执行顺序，因此，也称为过程模型。

1.3.2　软件生存周期模型

软件开发方法是指在规定的投资规模和时间限制内，为实现符合用户需求的高质量软件所制定的开发策略。人们提出了多种软件开发策略，常见的软件设计模型有瀑布模型(Waterfall Model)、增量模型(也叫渐进模型)(Increamental Model)、快速原型模型(Rapid Prototype Model)、演化模型(Evolutionary Model)、螺旋模型(Spiral Model)、喷泉模型(Fountain Model)、智能模型(Intelligent Model)等。这里介绍其中主要的几种模型。

1．瀑布模型

瀑布模型于 1970 年由 W.Royce 提出，其开发过程依照固定顺序进行，各阶段的任务与工作结果如图 1.4 所示。该模型严格规定各阶段的任务，上一阶段任务的输出作为下一阶段工作的输入。此模型适合于用户需求明确、开发技术比较成熟、工程管理严格的场合使用。其缺点是，由于任务顺序固定，软件研制周期长，前一阶段工作中造成的差错越到后期越大，而且纠正前期错误的代价高。

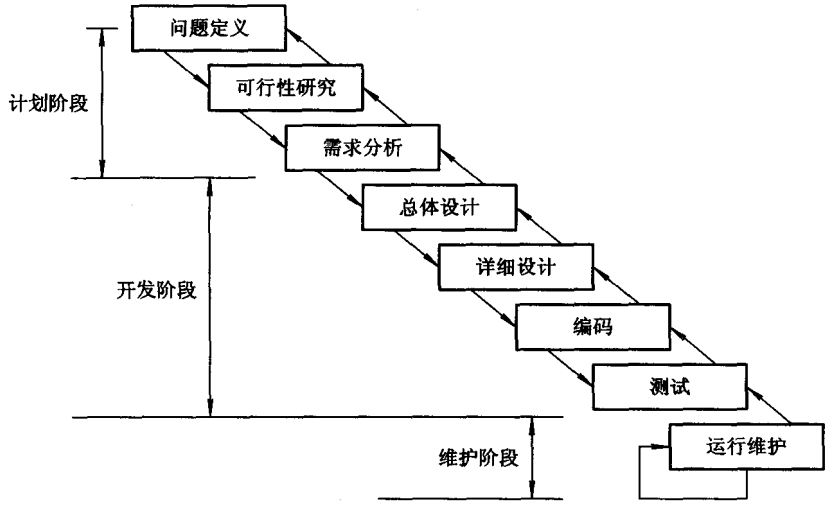


图 1.4　传统的瀑布模型

在 20 世纪 80 年代之前，瀑布模型一直是唯一被广泛采用的生命周期模型，现在它仍是软件工程中应用最广泛的过程模型。

按照传统的瀑布模型来开发软件，有如下几个特点：

(1) 阶段间具有顺序性和依赖性。

(2) 推迟实现的观点：清楚地区分逻辑设计与物理设计，尽可能推迟程序的物理实现，是按照瀑布模型开发软件的一条重要的指导思想。

(3) 质量保证的观点：每个阶段都必须完成规定的文档，没有交出合格的文档就是没有完成该阶段的任务；每个阶段结束前都要对本阶段所完成的文档进行评审，以便尽早发现问题，改正错误。