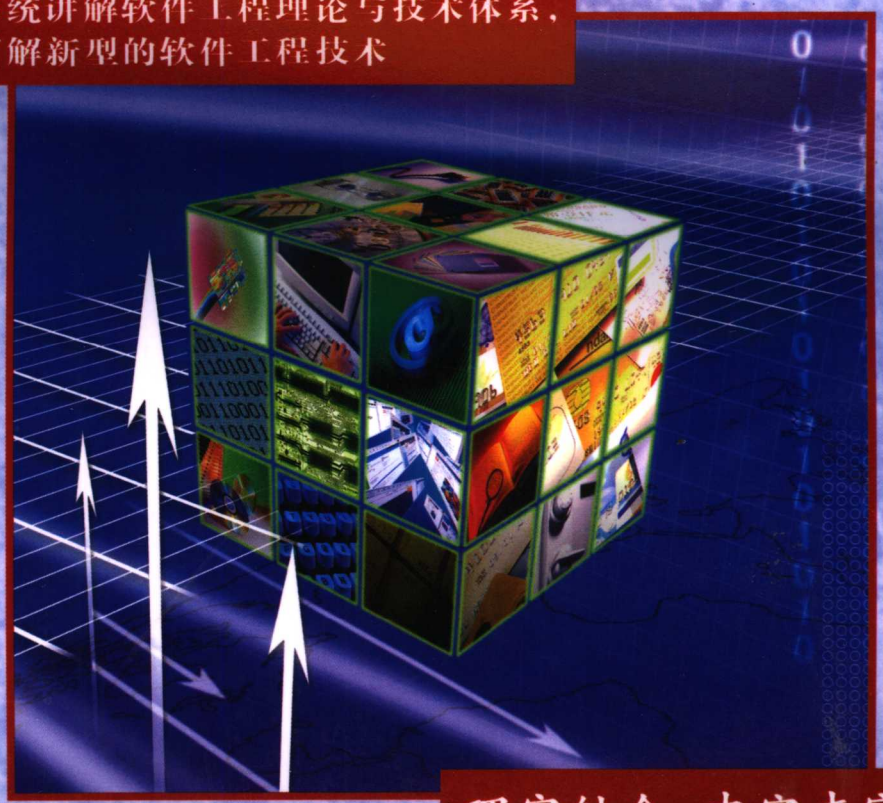
21世纪高等院校计算机教材

软件工程

臧铁钢 冷晟 钱晓明 朱健江 王晓勇 编著

系统讲解软件工程理论与技术体系，
了解新型的软件工程技术



理实结合 内容丰富

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

21 世纪高等院校计算机教材

软 件 工 程

臧铁钢　冷晟　钱晓明　朱健江　王晓勇　编著

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书主要面向广大非计算机专业的工程技术人员及大专院校师生，以满足其对工程软件开发的需求。本书介绍了工程软件的基本概念、软件系统可行性研究与需求分析、软件测试、软件维护和软件工程学等方面的内容，并以实例贯穿全书介绍了工程软件开发过程所涉及的相关知识。

本书内容丰富、全面，实用性强，既可作为高校非计算机专业工程类学生的教材，也可作为工程软件开发的参考资料。

图书在版编目（CIP）数据

软件工程 / 臧铁钢等编著. —北京：中国铁道出版社，
2007.7
21 世纪高等院校计算机教材
ISBN 978-7-113-07576-7

Ⅰ.软… Ⅱ.臧… Ⅲ.软件工程－高等学校－教材
Ⅳ.TP311.5

中国版本图书馆 CIP 数据核字（2007）第 103443 号

书　　名：软件工程
作　　者：臧铁钢　冷　晟　钱晓明 等
出版发行：中国铁道出版社（100054，北京市宣武区右安门西街 8 号）
策划编辑：严晓舟　秦绪好
责任编辑：杨　勇　黄园园
封面制作：白　雪
印　　刷：北京鑫正大印刷有限公司
开　　本：787×1092　1/16　**印张**：15　**字数**：350 千
版　　本：2007 年 7 月第 1 版　2007 年 7 月第 1 次印刷
印　　数：1～5 000 册
书　　号：ISBN 978-7-113-07576-7/TP・2250
定　　价：22.00 元

前言

FOREWORD >>>

软件开发是一项需要科学理论和相应技术支持的复杂的系统工程。软件工程就是这样一门能指导并管理软件开发过程的学问，它的产生和发展是人们对软件开发的规律深入认识的结果。为了解决“软件危机”，从20世纪60年代末开始，人们在软件开发方法方面进行了大量深入的研究，逐渐形成了一门新的学科——软件工程。直到现在，软件工程的目标一直未变，即研究科学地开发软件的方法，并配套发展相应的实现工具，直至构造良好的开发环境，力求达到投资省、开发品质高的目的。

软件工程是计算机科学的一个重要的、涉及面相当广泛的分支，是计算机科学与技术专业教学计划中培养合格的软件人才的必修课程。考虑到软件工程的实践性较强，发展迅速等特点，本书在对基本概念、基本原理、基本技术和基本方法进行深入说明的基础上，突出了理论与实践相结合的思想，并引进了一些目前较新的开发技术，如面向对象的软件开发方法等，使读者不仅能系统地了解软件工程的理论与技术体系，还能把握其发展脉络，了解到新型的软件工程技术。各章后都附有难度适宜的习题，以强化知识点的学习。本书可作为高等院校本、专科计算机专业教材，也可供计算机软件开发人员和计算机用户自学阅读。

本书共分为10章。第1章介绍了软件工程的概论；第2章介绍了软件系统可行性研究与需求分析技术；第3章介绍了软件设计技术；第4章介绍了编码与程序设计语言；第5章介绍了软件的技术度量及质量保证技术；第6章介绍了软件测试技术；第7章介绍了软件维护技术；第8章介绍了软件项目管理；第9章介绍了新型软件工程技术；第10章介绍了软件工程文件。以上各章基本上囊括了软件工程技术的各个方面的内容，使读者能全面了解软件工程学科，并初步具备进行软件开发的理论基础。

臧铁钢负责编写了第1、2章并统编全书，冷晟负责编写了第3、4、9章，钱晓明负责编写了第5、6、10章，朱健江负责编写了第7、8章，王晓勇负责本书的习题验证。

由于时间仓促，书中难免会有疏漏和不妥之处，恳请广大读者批评指正。

编　者

2007年4月

目 录

CONTENTS >>>

第1章 软件工程概论

首先，本章对软件技术进行了概括性的说明，在此基础上，对软件工程产生的原因、发展历程、研究内容等进行了阐述。之后，本章进一步对软件工程的基本原理和原则、软件生命周期模型、软件工程工具和开发集成环境进行了较详细的讲解。通过本章的学习，读者能对软件工程学科的全貌有较深入的了解。

1.1 软件技术概况

1.1.1 软件简介

随着人类科学技术的发展，以及实际应用的需求，20 世纪出现了计算机系统。计算机系统所拥有的强大计算能力使得人类的脑力得到了极大的延伸。目前，无论在复杂计算、工业控制、工程设计、信息处理和交换等领域，还是在人们的日常生活中，计算机都发挥着极为重要的作用。计算机是一项伟大的技术发明，对人类社会已产生了深刻的积极影响，并在可预见的未来，这一影响还将延续。

计算机系统可分为硬件系统和软件系统，两部分系统互为依赖，缺一不可。

计算机的硬件是计算机系统中各种设备的总称。硬件系统直观可触，是计算机运行的基础物理平台。从计算机发明以来，作为计算机硬件核心的处理器芯片已经历了电子管、晶体管、集成电路和大规模集成电路 4 个时代，现正在向集成度更高和运行速度更快的方向发展，计算机的功能也越来越强。根据统计，总结出了所谓的“摩尔定律”。该定理指出：芯片上可容纳的晶体管数目，约每隔 18 个月便会增加一倍，性能也将提升一倍。硬件的发展速度极其迅猛，从目前的情况来看，摩尔定律周期正在呈逐渐递减的趋势。

从 1946 年第一台计算机 ENIAC 问世以后，用于计算机运作的程序就开始从硬件中分离了出来，并形成了软件的概念。计算机硬件技术的每次技术突破，都为软件技术的发展提供了更为广阔的发展空间。与此同时，软件的概念也在不断地变化和扩展。目前，对于软件概念的一种公认的解释是：软件是计算机系统中与硬件相互对应的另一部分，是程序、数据及相关文档的完整集合。其中，程序是按事先设计的功能和性能要求执行的指令序列；数据是使程序正常操纵信息的数据结构；文档是与程序开发、维护和使用有关的图文材料。也就是说，程序和软件有着不同的概念，软件的范畴要比程序大。

人们对软件定义的认识经历了一个较长的过程。20 世纪 50 年代，程序设计还处于个体手工生产阶段，程序是由设计者本人开发和维护的结构简单、功能单一、可靠性差的小型源程序。所以，人们认为软件就是程序，软件系统就是程序系统；20 世纪 60 年代，软件的功能、规模日益增大，软件的开发只能由群体来承担，软件设计进入了作坊式生产阶段。为了提高程序的可维护性，人们认为软件等于程序加文档，所谓文档就是指软件开发过程中与分析、设计、实现、测试、维护相关的文档，但不包括与软件开发过程相关的管理文档；对管

理文档的全面认识是从20世纪70年代开始的。在这一时期，随着软件的规模、数量剧增和交付的紧迫，要求大型程序的开发活动要按照工程项目的模式运作。此时，软件工程方法被引入到软件开发过程中，对软件定义的认识也得到了提高，一般认为软件构成公式如下：

软件＝程序+文档+数据

上式中的文档包括了管理文档。数据则不仅包括初始化数据、测试数据、研发数据、运行数据、维护数据，还包括软件企业积累的项目工程数据和项目管理数据中的大量决策原始记录数据；20世纪80年代，人们开始认识到软件管理是一个过程管理。因此，到20世纪90年代，出现了软件过程能力成熟度模型，人们开始研究软件过程管理的具体内容与方法。1997年出现的统一建模语言（Unified Modeling Language，UML）就是突出的研究成果之一。UML的目标之一就是为开发团队提供标准通用的设计语言来开发和构建计算机应用。UML的出现标志着软件开发进入到了一个以规范化、过程化、工厂化、大型化、自动化为特征的较成熟的阶段。

一方面，软件与一般的产品有共同点；另一方面，软件具有与一般产品不同的特点，具体体现如下：

（1）软件体现的是某种逻辑的流程，可以存在于存储介质上，但其本身却不可见，需通过分析才能了解其功能。

（2）软件的生产过程就是大量复制，它没有明显的制造过程，故生产成本相当低廉。

（3）软件不存在磨损、老化的问题，但存在退化的现象。即软件会为了适应新的运行环境和新的需求而进行修改，当修改成本最终变得不可接受时，软件就会出现退化，从而被抛弃。

（4）软件对硬件和运行环境有着不同程度的依赖性，软件多少存在可移植性问题。

（5）软件的开发涉及的影响因素众多，如管理方式、开发模式，以至于理念等都直接影响到软件开发项目。

在计算机诞生之初，软件主要用于科学和工程数值计算，如今软件已大量用于非数值计算领域，如数据管理和传送。软件已从最初作为硬件的附属，发展成为计算机系统不可缺少的组成部分。软件的应用领域在不断扩张，使得软件的种类不断丰富。从软件使用和开发的角度来看，要给出一个统一、严格的分类标准是不可能的，只能根据软件的用途和软件的适用对象对其进行分类。软件大致可以分为如表1-1所示的几类。

表1-1 软件的分类

分类准则	软件类别
软件功能	系统软件、支持软件、应用软件
工作方式	实时软件、分时软件、交互式软件、批处理软件
规模	微型软件、小型软件、中型软件、大型软件、超大型软件、极大型软件
服务对象	定制软件、产品软件
应用领域	操作系统、数据库管理系统、软件开发系统、工程软件、办公软件、财会软件、网络工具软件、图形图像处理软件、多媒体软件、游戏软件、教育软件等
收费方式	商品软件、共享软件、自由软件

可以预计，随着计算机系统的不断发展，软件的概念还会因其自身的发展而不断变化，软件的种类也会不断地丰富，各类软件的功能和复杂程度也将随着硬件水平的提高而不断提升。软件已成为人们工作、生活中不可缺少的部分。

1.1.2　软件的发展历程

自从 20 世纪 40 年代中期出现了世界上第一台计算机以来，软件技术就开始了发展的历程。当时，人们将大部分关注放在了昂贵的硬件上，对软件的开发不是很重视。软件的性能和可靠性往往取决于开发者的能力，开发的随意性很大。随着计算机的性能和应用水平的提高，人们深刻地认识到了软件开发技术的重要性，从而开始了相应的研究。到目前为止，软件技术已经历了以下 3 个阶段。

1．程序设计阶段（20 世纪 50 年代至 20 世纪 60 年代）

在这一阶段，通用硬件已较为普遍，但关于软件的相关概念还没有正式出现。软件一般就是指为具体应用而编写的程序指令，结构功能简单、规模小。软件生产方式是个体手工劳动，使用的工具大多是机器语言、汇编语言和一些出现很早的高级语言，如 FORTRAN 语言；片面追求编程技巧，追求程序运行效率，使得程序很难维护，且运行可靠性差。这一阶段的软件开发严重缺乏管理，也没有系统化的软件开发方法可以遵循，几乎没有文档资料保存下来，但软件学科已出现了萌芽。到 20 世纪 60 年代末期，由于软件开发的个体化与不断提升的软件规模和复杂度间的矛盾，使得软件的可靠性相当低，出现了所谓的“软件危机”。

2．程序系统阶段（20 世纪 60 年代至 20 世纪 70 年代）

这一阶段计算机硬件采用了集成电路，这为软件技术的发展提供了坚实的物质基础。此时，操作系统引入了多道程序、多用户分时和实时处理的概念，程序开始系统化，出现了软件产品，用户开始使用购买的软件，软件规模和数量也因此急剧增加。程序系统时代的生产方式是作坊式的小集团合作化生产，采用了高级语言；尽管开发方法仍依靠个人技巧，但在很短的时间内就提出了模块化、结构化、自顶向下逐步求精、程序变换、程序的推理与综合、数据类型抽象、程序正确性证明、符号测试等各种程序设计和验证的方法。这时人们已意识到软件开发的重要性，提出了以正确性、结构清晰、便于阅读、便于测试和维护为软件质量的标准。这一时期，相对落后的开发技术已不适应规模大、功能结构复杂的软件开发，“软件危机”进一步加剧。在这一阶段，人们认为软件就是程序及其规格说明书的总和。

3．软件工程阶段（20 世纪 70 年代至今）

该时期的计算机硬件正向超高速、大容量、微型化及网络化的方向发展，计算机的应用已经深入到社会生活中的各个领域。“软件作坊”式的生产方式受到了批判，软件业开始打破软件开发的个体化特征，软件开发有了软件工程化的设计原则、方法和标准可以遵循，以软件产品化、系列化、工程化、标准化为标志的软件产业逐渐兴起。这一时期的生产方式是工程化的生产，强调软件的全生命周期，网络、分布式技术、面向对象技术成为了软件开发的基础。目前，计算机应用正在逐渐向企业计算机化、社会信息化的计算机系统工程阶段过渡，以满足现代社会各种计算机应用对计算机软件的巨大需求。

以上3个阶段描述了软件技术发展的脉络，即从无序到有序，从不规范到规范，从简单到复杂，从不可靠到可靠。随着计算机系统的发展，新的软件技术还将不断出现，软件工程的理论和方法还将进一步丰富。

1.2 软件工程简介

1.2.1 软件工程的产生

软件工程产生的诱因是20世纪60年代的软件危机。通过解决软件危机，软件工程的理论和方法逐渐形成，并最终成为了现代软件技术的核心之一。

在软件技术发展的程序系统阶段，随着计算机硬件技术的发展，硬件成本急剧下降，计算机得以在各个领域普及；应用领域的增加使得对软件的需求急剧增加，软件的种类增多，规模膨胀，并强烈地产生了对软件维护的需求。但是，这一时期仍然沿用早期个体化的软件开发方法，落后的软件开发和维护技术使得软件根本无法维护。这样，在用户的需求和软件开发的实际状况之间形成了尖锐的矛盾，这就形成了所谓的软件危机。此时，软件业面临的问题可概括为：如何开发软件以满足用户日益增长的需求和如何维护已有的数量庞大的软件两个方面。

当时，软件危机的具体表现就是软件可靠性非常差。例如，IBM公司开发的OS/360系统，耗资几千万美元，拖延了几年才交付使用，交付使用后每年都会发现相当多的错误，导致软件退化加剧。其他大型软件也同样面临着花费了大量的人力、财力后，不得不半途而废的结局。一些由软件控制的高价值的系统（如导弹、卫星、航天器等）也频频出现故障。1979年，美国财政部对政府投资开发的软件项目进行了调查，结果令人震惊：47%的资金花费在从未使用过的软件上，另外29%的资金花费在那些半途而废和交付后还需进一步完善的软件上，可以称为成功的项目仅占3%～4%。

对于具体的软件开发和维护过程而言，软件危机表现为：

（1）由于缺乏软件开发经验和科学的理论指导，开发者不能准确地估计软件开发的成本和进度。所制定的成本规划可能大大低于实际的成本，实际的开发进度也可能要比计划的慢，为了赶进度往往就会牺牲软件的质量。

（2）由于缺乏使开发人员与用户进行交流的有效机制，开发人员常会犯闭门造车的错误。从而会造成用户对已完成的软件系统不满意。

（3）由于测试工作不够充分，又没有好的软件质量保证技术，导致提交给用户的软件质量不高。

（4）软件的可维护性差，程序中的错误难以改正。程序的可移植性差，很难适应不同的运行环境。软件的可重用性差，大量的软件人员在重复开发。

（5）开发过程缺乏标准和规范的指导，各个开发组织都有自己的开发方法，开发组织之间进行工作交接的流程也不规范。

（6）软件开发生产率的提高速度难以满足对软件需求的增长速度，软件产品供不应求。

（7）由于软件开发和维护方法不当，使得软件成本居高不下，过高的成本已严重制约了软件的开发。

总之，造成软件危机的原因很多，归纳起来主要有两个方面：一是与软件本身的特点有关，二是与软件开发与维护方法不当有关。所以，为了解决软件危机，人们认识到有以下 3 种途径：

（1）采用工程化方法来开发和维护软件。软件开发不应只是个体化的劳动，而应该是由组织良好、管理严密、各类人员共同配合完成的一个工程项目，因此应该注意吸收和借鉴从事其他工程项目的行之有效的科学原理和方法。

（2）采用先进的技术、方法、工具开发和设计软件。即采用先进的管理技术、规范的开发方法和模型、各种提高开发效率的软件工具等。

（3）采用必要的组织管理措施。软件工程正是在解决软件危机问题的过程中形成的一门综合技术与管理两个方面的新兴学科，并逐渐成为指导计算机开发、维护、管理的理论依据。

在寻求以上 3 种途径的具体实现方法的过程中，逐渐形成了软件工程学科。软件危机使软件科学工作者和软件开发人员意识到：作为一种特殊的产品，软件开发工作也应跟机械产品、电子产品、化工产品一样，遵循系统工程的思想和方法。1968 年，北大西洋公约组织在德国举行的一次学术会议中首先提出了软件工程的初步概念，但软件工程作为一门工程学科，到 20 世纪 70 年代末至 80 年代初才正式形成。

对于软件工程的定义有很多的版本。1983 年，IEEE（Institute of Electrical and Electronics Engineers）为软件工程下的定义是：软件工程是开发、运行、维护和修复软件的系统方法；1993 年，IEEE 进一步给出了一个更全面的定义：

（1）把系统化的、规范的、可度量的信息途径应用于软件开发、运行和维护的过程，也就是把工程化应用于软件中。

（2）研究（1）中提到的途径。

概括起来，所谓软件工程就是采用工程化的概念、原理和技术、方法来开发和维护软件，把经过时间考验证明是正确的管理技术和当前好的技术方法结合起来指导计算机软件开发和维护的工程学科。

软件工程已经历了传统软件工程时代、对象软件工程时代、过程软件工程时代、构件软件工程时代。目前的软件工程发展趋势正在这 4 个时代的基础上朝着流水线装配软件工程的方向发展。

1.2.2　软件工程的基本内容和目标

软件工程的内容也就是软件工程所包含的要素，主要包括软件开发模型、软件开发方法、软件工程工具、软件过程管理 4 个方面，如表 1-2 所示。

（1）软件开发模型就是软件开发流程的模型，它确定了一个软件开发过程的基本模式。常见的软件开发模型有瀑布模型、增量模型、原型模型及迭代模型等。

（2）软件开发方法为软件开发提供了具体的指导，明确了软件开发的核心对象。目前常用的软件开发方法有面向过程的方法、面向数据的方法和面向对象的方法。

（3）软件工程工具为软件开发方法提供了自动的或半自动的软件支持环境。能有效提高软件开发的效率，并减少人为造成的错误。

（4）软件过程管理是对软件开发的过程进行全程监控，以保证开发和维护任务的顺利进行。软件过程管理包含了多方面的任务，如项目的计划与成本估算，软件系统的需求分析，数据结构和系统总体结构的设计、算法过程的设计、编码、测试及维护等。软件工程方法常采用相应的标准（如 ISO9000）来保证软件的质量。

表 1-2 软件工程的研究内容

研究内容	内容举例
软件开发模型	瀑布模型、增量模型、原型模型、迭代模型
软件开发方法	面向过程的方法、面向数据的方法、面向对象的方法
软件工程工具	CASE 工具类
软件过程管理	ISO9000、CMM

软件工程的初衷就是要消除软件危机，这也是它的根本目标。对一个具体的软件工程项目而言，采用软件工程的技术和管理方法组织软件的开发，最终目标是希望获得软件项目的成功，使软件产品能正确、可靠和高效率地运行。具体而言，软件工程的目标主要包括：有效控制开发成本、达到要求的软件功能、取得较好的软件性能、开发软件易于移植、需要较低的维护费用、有较高的开发效率和能按时交付使用等。在实际的开发过程中，要想使以上的几个目标都能达到理想程度往往是很困难的，有些目标之间可能存在着冲突。因此，实施软件开发项目还要努力协调以上目标，突出重点，取得平衡。

1.2.3 软件工程的基本原理

自从软件工程诞生以来，研究软件工程的专家学者们陆续提出了很多关于软件工程的原理或准则。由软件工程专家 B. W. Boehm 提出的软件工程的 7 条基本原理被普遍采用。这 7 条原理是确保软件产品质量和开发效率原理的最小集合。这 7 条原理是互相独立而完备的，其中任意 6 条原理的组合都不能代替另一条原理。可以证明其他软件工程原理都可以由这 7 条基本原理的任意组合蕴含或派生。这 7 条原理如下。

1. 用分阶段的生命周期计划严格管理

该原理是在吸取前人的教训的基础上而提出来的。经统计发现，由于计划不周而失败的软件项目占全部失败软件项目的一半左右。该原理把软件生命周期划分成若干个阶段，并相应地制定出切实可行的计划，然后严格按照计划对软件的开发与维护工作进行管理。软件的整个生命周期中应该制定 6 类计划，即项目概要计划、阶段计划、项目控制计划、产品控制计划、测试计划和维护计划。

2. 坚持进行阶段评审

在软件开发过程中，错误的发现和改正越晚，所付出的代价也越高。因此，在每个开发阶段都进行严格的评审，尽早发现在软件开发过程中所犯的错误是一条必须遵循的重要原则。

3. 实行严格的产品控制

一方面在软件开发过程不应随意改变需求，因为改变需求意味着计划的修正，这往往需要付出较大的代价，另一方面客户提出改变需求的要求也应是允许的，但这就需要进行严格

的约束。也就是说，当改变需求时，为了保持软件各个配置成分的一致性，必须实行严格的产品控制，其中主要是实行基准配置管理。基准配置是经过阶段评审后的软件配置成分。一切有关修改软件的建议，特别是涉及到对基准配置的修改建议，都必须按照严格的规程进行评审，获得批准以后才能实施。

4．采用现代程序设计技术

采用先进的程序设计技术不仅可以提高软件开发和维护的效率，而且可以提高软件产品的质量。自从提出软件工程的概念开始，人们一直致力于研究各种新的程序设计技术，已被广泛采用的程序设计技术有结构设计技术、面向对象技术等。

5．结果应能清楚地审查

由于软件本身的特点，使得软件产品的开发过程比一般产品的开发过程更难于评价和管理。为了提高软件开发过程的精细管理水平，应该根据软件开发项目的总目标及完成期限，在给定成本和目标进度的前提下，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．开发小组的成员应该少而精

首先，根据所谓的“二八定律”所述，20%的人，解决了软件中 80%的问题；其次，随着开发小组人数的增加，因交流讨论而造成的通信开销也急剧增加，同时，通信路径交叉过多，更易产生错误。也就是说，软件开发小组的组成人员应具备相应的素质，而人数不宜多。

7．承认不断改进软件工程实践的必要性

为了保证软件开发与维护的过程能跟上时代前进的步伐，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，以作为今后的软件开发的借鉴。该原理保证了软件工程技术的先进性。

1.2.4　软件工程的原则

为了开发出低成本、高质量的软件产品，软件工程学应遵循以下基本原则。

1．抽象原则

抽取事物最基本的特性和行为，忽略非基本细节。采用层次抽象，自顶向下、细化的方法控制软件开发过程的复杂性。

2．信息隐藏原则

将模块中对外的部分设计成清晰的接口，而实现细节则隐藏在模块内部，不让模块的用户直接访问，也就是所谓信息封装。用户只能通过模块接口访问模块中封装的数据。

3．模块化原则

模块是程序中在逻辑上相对独立的功能集成体，它应有良好的接口定义，模块与模块或模块与操作者间通过设定的接口进行联系。模块化有助于信息隐藏和抽象，有助于表示功能复杂的系统。

4. 局部化原则

在一个模块内集中逻辑上相互关联的计算机资源，并保证模块之间松散的耦合，而模块内部有较强的内聚，这就是计算机资源的局部化，局部化有助于控制软件结构的复杂化趋势。

5. 确定性原则

软件开发过程中所有概念的表达应是确定的，无歧义的，规范的。这样有助于人们在交流时不会产生误解、遗漏，保证整个开发工作的协调一致。

6. 一致性原则

整个软件系统的各个模块应使用一致的概念、符号和术语；程序内、外部接口应保持一致；软件同硬件、操作系统的接口应保持一致；系统规格说明与系统行为应保持一致，用于形式化规格说明的公理系统应保持一致。

7. 完备性原则

软件系统不丢失任何重要成分，可以完全实现系统所要求的功能。为了保证系统的完备性，在软件开发和运行过程中需要严格的技术评审。

8. 可验证性原则

开发大型的软件系统时需要对系统自顶向下、逐层分解。系统分解应遵循使系统易于检查、测试、评审的原则，以确保系统的正确性。

实际的软件开发过程所面临的问题是多样的、复杂的。只有在以上原则的指导下，软件工程的方法才能得以实施。

1.2.5 软件生命周期模型

软件生命周期是指一个软件系统从目标提出到最后淘汰的整个过程。软件工程基本原理强调软件生命周期的阶段性，其基本思想是各阶段任务相对独立，具有明确的完成标志。阶段的划分使人员分工职责清楚，项目进度控制和软件质量得到确认。原则上，前一阶段任务的完成是后一阶段工作的前提和基础；而后一阶段的任务则是对前一阶段问题求解方法的具体化。整个软件生命周期可以划分为 3 个阶段，即软件定义、软件实现和运行维护。各阶段的划分如图 1-1 所示。

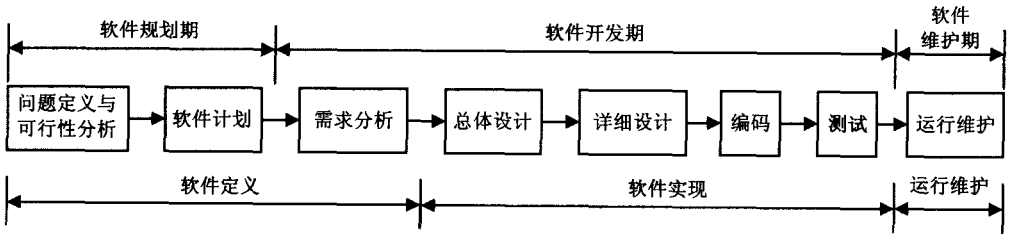


图 1-1 软件生命周期的划分

软件生命周期模型是描述软件开发过程中各种活动如何执行的模型。它为软件开发提供了强有力的支持，为软件开发过程中所有活动提供了统一的纲领，为参与软件开发的所有成员提供了帮助与指导。它展示了软件的演绎过程，是软件生命周期模型化技术的基础，也是建立软件开发环境的核心。