21世纪电子信息与自动化系列规划教材

软件工程

（第二版）

主　编　卢　潇
副主编　孙　璐　刘　娟
　　张科英　蒋　华

中国水利水电出版社
www.waterpub.com.cn

21世纪电子信息与自动化系列规划教材

软件工程

（第二版）

主　编　卢　潇

副主编　孙　璐　刘　娟　张科英　蒋　华

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

本书是作者总结多年软件工程教学和科研实践经验，并吸取国内外大量同类书刊的精华，在第一版成功应用的基础上，结合近年来软件工程技术的发展，对原书内容做了调整和增删而成的。

全书正文共 13 章，内容可分四部分：第一部分主要介绍软件工程的基本概念，并概要地介绍软件生存周期、开发模型及软件开发的各种方法；第二部分按生命周期模型详细介绍软件计划、需求分析、设计、编码、测试和维护各个阶段的有关概念、工作内容，重点介绍结构化方法和 Jackson 方法的实施，并介绍软件体系结构的相关内容；第三部分介绍面向对象的方法及 UML 建模语言；第四部分介绍软件标准、文档、质量评价和质量保证技术、软件工程的管理和认证等内容。

本书可作为高等院校“软件工程”课程的教材或教学参考书，也可作为软件项目管理者和软件开发人员的参考书。

本书配有电子教案和素材文件，读者可以从中国水利水电出版社网站和万水书苑免费下载，网址为：http://www.waterpub.com.cn/softdown/和 http://www.wsbookshow.com。

图书在版编目（CIP）数据

软件工程 / 卢潇主编. -- 2版. -- 北京 : 中国水利水电出版社, 2011.1
21世纪电子信息与自动化系列规划教材
ISBN 978-7-5084-8069-5

Ⅰ. ①软… Ⅱ. ①卢… Ⅲ. ①软件工程－高等学校－教材 Ⅳ. ①TP311.5

中国版本图书馆CIP数据核字(2010)第220428号

策划编辑：周益丹　责任编辑：张玉玲　加工编辑：周益丹　封面设计：李　佳

书　名	21世纪电子信息与自动化系列规划教材 **软件工程（第二版）**
作　者	主　编　卢　潇 副主编　孙　璐　刘　娟　张科英　蒋　华
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路 1 号 D 座　100038） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（营销中心）、82562819（万水）
经　售	全国各地新华书店和相关出版物销售网点
排　版	北京万水电子信息有限公司
印　刷	北京市天竺颖华印刷厂
规　格	184mm×260mm　16 开本　18.75 印张　490 千字
版　次	2005 年 1 月第 1 版 2011 年 2 月第 2 版　2011 年 2 月第 4 次印刷
印　数	12001—15000 册
定　价	29.80 元

前　言

软件工程学（通常简称软件工程）是一门迅速发展的新兴学科。所谓软件工程，即用“工程化”的思想来指导并解决软件研制中的各种问题。其研究的范围非常广泛，包括技术方法、工具和管理等许多方面。软件工程的目标在于研究一套科学的工程方法，并建立与此相适应的、方便实用的工具系统，力求以最少的成本获得高的软件质量。近年来，软件工程发展迅速，新的技术方法和工具不断涌现。本书力求系统地给出软件工程的框架，在保证全书内容全面、系统的基础上，着重从实用角度讲述软件工程的基本原理、概念和技术方法。希望本书能为读者今后深入研究这门学科奠定良好的基础，并能对实际的软件开发工作有所帮助。

本书共13章，从内容上可分四部分：

第一部分（第1章）：概括介绍软件工程学产生的历史背景以及相关的基本原理、概念和方法。主要介绍软件工程的基本概念，并概要介绍软件生存周期、开发模型及软件开发的各种方法。

第二部分（第2～8章）：按生命周期模型，详细介绍软件计划、需求分析、设计、编码、测试和维护各个阶段的有关概念、工作内容，重点介绍结构化方法和Jackson方法的实施，并介绍了软件体系结构的相关内容。

第2章介绍软件生命周期的第一个阶段——软件计划的任务和相关技术，主要内容有可行性研究、软件计划和成本/效益分析。

第3章介绍需求分析的发现、求精、建模、规格说明和复审的过程，以及需求管理的内容，讨论获取需求的方法及需求分析的原则，重点介绍结构化分析技术和原型技术，并给出了软件需求规格说明书的要求。

第4章阐述软件设计中用到的基本概念及软件总体结构、数据结构与软件过程等概念，重点介绍软件总体设计的方法和设计表达工具，详细阐述与结构化需求分析方法衔接的、面向数据流的设计方法——结构化软件设计方法（SD 方法）和面向数据结构的设计方法——Jackson 设计方法。

第5章介绍详细设计的任务、方法和工具，并阐述了近年来软件工程领域的新的研究热点和关键技术之一———软件体系结构的有关内容。

第6章围绕着编码质量谈论了程序设计语言的选择及编码风格等内容。

第7章讨论软件测试的相关概念、步骤，介绍常用的测试技术和工具。

第8章介绍软件维护的任务、特点、组织以及软件维护可能带来的问题，如何提高软件的可维护性等。

第三部分（第9章）：介绍了面向对象技术的基本概念，以实例的形式阐述用UML建模语言进行系统分析、系统设计和系统实施的基本过程。

第四部分（第10～12章）：介绍软件标准、文档、质量评价和质量保证技术、软件工程的管理和认证等内容。

第10章介绍软件工程标准化的相关概念、国内外的现状以及软件文档的相关内容。

第 11 章介绍软件质量的特性，并给出软件质量的度量方法及如何进行质量评价，最后还简单介绍了软件的质量保证和质量管理体系。

第 12 章讨论软件工程管理技术，并介绍目前流行的管理工具、IPMP 与 PMP 认证体系及我国目前的项目管理认证体系的发展状况等内容。

第 13 章比较完整地介绍一个实际软件的开发。着重阐述从问题定义到实现的过程，将这个具体例子与书中前几章的内容结合起来学习，有助于加深对一些基本概念和方法的理解。

本书第 1～5 章和第 13 章由卢潇和卢靓妮编写，第 6～9 章由孙璐和蒋华编写，第 10～12 章由刘娟和张科英编写。韩毅娜和车从领、张强、孙路等对书中的实例及图表做了大量的工作。

在本书编写过程中作者参考了大量书籍、资料和网站，同时融入了作者多年教学和科研工作的体会和经验。鉴于作者的学识水平有限，书中谬误和不足之处在所难免，敬请广大读者批评指正。

编　者

2010 年 12 月

目　　录

第1章　概述

1.1　软件的概念

一个可用的计算机系统离不开软件，20世纪40年代，随着世界上第一台计算机的诞生，产生了软件的概念，计算机硬件的高速发展和计算机的应用领域的不断拓展，促进了软件技术不断发展，出现了与软件相关的专业和领域。随着信息社会的到来，软件在人类社会中越来越重要。

1.1.1　软件的发展阶段

从20世纪40年代第一台计算机问世到现在，软件的发展过程可分为以下三个阶段：

（1）程序设计阶段（20世纪40年代至60年代初）。

在此阶段，计算机的应用仅限于一些专门的领域。计算机的使用者，也是一些经过专门训练的专业人员，如数学家和电子工程师。使用计算机的方式，就是根据需要，编写出相应的程序、运行程序、获得结果。此时，一般程序的规模都较小，程序的开发者、使用者和维护者往往是同一人（或同组人）。由于当时的硬件成本很高，所以，此时所谓软件就是程序，开发软件就是编写程序。程序设计追求的目标是采用编程技巧，提高程序的效率。

（2）程序系统阶段（20世纪60年代初至70年代初）。

随着计算机的应用领域的扩展，软件的规模越来越大，用户已无力承担软件的开发工作，出现了专门的软件开发人员和专门进行软件生产的“软件作坊”。所开发的软件已不是为开发者个人使用，而成为面向某一领域广泛用户的软件产品，主机和微机上的程序能够有数百甚至上千的用户使用。此时软件开发的方式是多人分工合作的一种作坊式的“个体化”开发方法。

（3）软件工程阶段（20世纪70年代之后）。

进入20世纪70年代后，由于硬件的飞速发展，对软件提出了更多的需求，所处理的对象种类越来越多，任务也越来越复杂，而作坊式的软件开发方式开发效率低，开发出来的软件质量差，无法满足日益增大的软件需求，出现了“软件危机”。为解决“软件危机”，整个产业界开始采用了软件工程实践，软件开发成为一门新兴的工程学科。

由软件技术的发展过程可以看出，软件发展的动力是软件的需求。第一阶段软件开发只是为了满足开发者自己的需要，进入软件工程阶段以后，软件开发的成果具有社会属性，它要成为产品，在市场中流通以满足广大用户的需要。随着软件技术的发展软件工作的范围，从只考虑编写程序扩展到涉及软件的计划、分析、设计、测试以及运行维护的各个方面。同时，软件的概念也在不断地充实和完善。计算机软件发展的三个时期及特点如表1-1所示。

表 1-1　计算机发展的三个时期及特点

阶段 / 特点	程序设计	程序系统	软件工程
软件的范畴	程序	程序及说明书	产品（项目）软件
主要程序设计语言	汇编及机器语言	高级语言	高级语言系统 程序设计语言
软件工作范围	编写程序	编写程序、设计、测试	软件生存期各阶段
需求者	程序设计者	少数用户	市场用户
维护者	程序设计者	开发小组	专职维护人员
硬件特征	价格高、存储容量小、可靠性差	价格下降，运算速度、存储容量、可靠性明显提高	向超高速、大容量、微型化发展

1.1.2　软件的定义

随着软件技术的发展，人们对软件的认识也在不断地加深，软件的定义也随之不断地完善。早期，人们认为软件就是源程序，开发软件就是编写程序。那些被认为是优秀的程序通常充满了程序技巧，很难被别人看懂。随着对软件及其特性的更深层的研究，人们认识到，优秀的程序除了功能正确、性能优良之外，还应该容易看懂、容易使用、容易修改和扩充，并且软件也不仅仅只有程序。1983 年 IEEE 为软件下的定义是：计算机程序、方法、规则和相关的文档资料以及在计算机上运行时所必需的数据。目前对软件通俗的解释为：

软件=程序+数据+文档资料

其中，程序是完成特定功能和满足性能要求的指令序列；数据是程序运行的基础和操作的对象；文档是与程序开发、维护和使用有关的图文材料。

1.2　软件危机

1.2.1　什么是软件危机

在软件技术发展的第二阶段，随着计算机硬件技术的进步，计算机的应用领域越来越广泛，软件的数量急剧膨胀，软件需求日趋复杂。然而软件开发技术的进步严重滞后于形势发展提出的要求，造成供求关系失调，形成了日益尖锐的矛盾。主要问题有：

（1）软件的开发费用和进度难以控制。由于缺乏软件开发的经验和有关软件开发数据的积累，使得开发工作的计划很难制定，致使经费预算常常被突破，进度计划无法实施。

（2）开发出来的软件不能满足用户的要求。由于软件需求在开发的初期阶段提得不够明确，或者表达不确切、理解存在偏差，在开发过程中，软件人员和用户又未能及时交换意见，造成开发后期矛盾的集中暴露。

（3）软件的可维护性差。由于开发过程没有统一的、公认的方法论和规范指导，参加的人员各行其是，设计和实现过程的资料很不完整，或忽视了每个人与其他人的接口，使得软件维护在源

代码级上进行，增加了软件维护的难度。

（4）软件质量差。由于未能在测试阶段充分做好检测工作，致使提交用户的软件隐藏大量错误，软件运行的可靠性差。

这些问题导致一些软件开发项目耗费了大量的人力、物力、财力，而失败的软件开发项目却屡见不鲜，例如，美国公司开发的 IBM 360 机的操作系统，耗资几千万美元，花费 5000 多人力，拖延了几年才完成，但交付使用后仍不断发现错误。

失败的教训引起了人们对这些问题的重视，1968 年北大西洋公约组织的计算机科学家在联邦德国召开的国际学术会议上第一次提出了“软件危机”（Software Crisis）一词。

软件危机指的是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，概括来说，软件危机包含两方面问题：一是如何开发软件，以满足不断增长、日趋复杂的需求；二是如何维护数量不断膨胀的软件产品。

1.2.2　产生软件危机的原因

分析软件危机的原因，一方面与软件本身的特点有关；另一方面是由软件开发和维护的方法不正确有关。

1.2.2.1　软件的特点

软件与硬件产品相比，有以下特性：

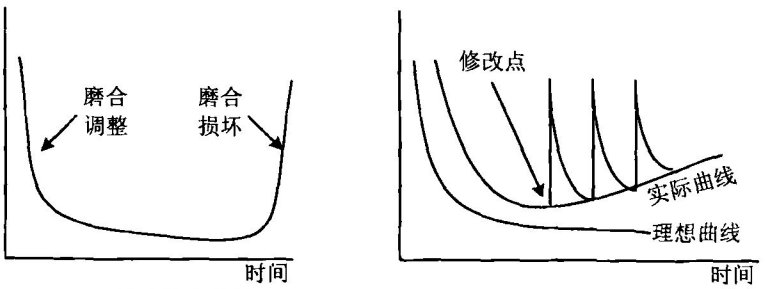
（1）软件是一种逻辑实体，而不是具体的物理实体。可以把软件记录在各种介质上，但无法看到软件本身的形态，必须通过观察、分析、思考、判断，才能了解它的功能和性能。因而它具有很强的抽象性。

（2）软件的生产与硬件不同，软件是由开发或工程化而形成的，而不是传统意义上的制造产生的，软件没有明显的制造过程。其生产过程主要表现为人脑的思维过程，不可见性，并且带有个人色彩。在 20 世纪 80 年代中期，“软件工厂”的概念被正式引入。但这个术语并没有把硬件制造和软件开发认为是等价的，而是通过软件工厂这个概念提出了软件开发中应该使用自动化工具。

（3）在软件的运行和使用期间，没有硬件那样的机械磨损，老化问题。任何机械、电子设备在运行和使用中，其失效率大都遵循如图 1-1（a）所示的 U 形曲线（即浴盆曲线）。而软件的情况与此不同，因为它不存在磨损和老化问题，而存在退化问题。为了适应硬件、系统环境以及需求的变化，必须要多次修改（维护）软件，如图 1-1（b）所示。然而软件的修改不可避免地会引入新的错误，导致软件的失效率升高，从而使得软件可靠性下降。当修改的成本变得难以接受时，软件就被抛弃。

（4）软件对硬件和环境有着不同程度的依赖性，这导致了软件升级和移植的问题。随着计算机技术的发展，计算机硬件和支撑环境不断升级，为了适应运行环境的变化，软件也需要不断维护，并且维护成本通常比开发成本高许多。

（5）软件的复杂性越来越高。随着软件需求的增长，软件所处理的对象类型由单纯的数值型发展到字符、图形、声音等，软件处理问题的规模日趋庞大，IBM 360 操作系统第 16 版有 100 万条指令；1973 年美国阿波罗计划达 1000 万条指令。这些庞大软件的功能非常复杂，其程序逻辑结构、调用关系、接口信息以及数据结构都很复杂，随着软件规模的增大，对软件人员的要求越来越高，其复杂程度超出了人能接受的程度，出现了软件复杂性与软件技术发展的不适应现象，如图 1-2 所示。



（a）硬件失效率曲线　　（b）软件失效率曲线

图 1-1　失效率曲线

（6）软件成本相当昂贵。软件的研制工作需要投入大量的、复杂的、高强度的脑力劳动，它的成本自 20 世纪 80 年代以来，已大大超过硬件成本，硬/软件成本的变化趋势如图 1-3 所示。

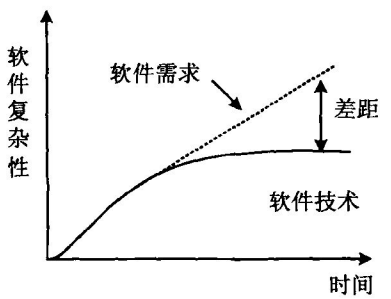


图 1-2　软件技术进步落后于需求增长

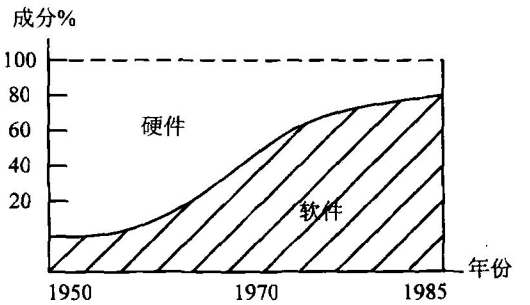


图 1-3　硬件/软件成本变化趋势

（7）硬件的设计和建造可通过画电路图，做一些基本的分析以保证可以实现预定的功能，根据功能和接口要求选定并购买零件。而软件设计中几乎没有软件构件。大多数软件是新开发的，而不是通过已有的构件组装而来的。有可能在货架上买到的软件，它本身就是一个完整的软件，而不能作为构件再组装成新的程序。

（8）软件工作涉及许多社会因素，如机构、体制、管理方式等，包括人的观念及心理，都直接影响软件工作的成败。

以上特点使得软件开发进展情况较难衡量；软件开发质量难以评价；使得产品的生产管理、过程控制及质量保证都相当困难。

1.2.2.2　软件开发的方法和技术

导致“软件危机”的原因除了软件固有特性的这个客观因素外，还有一个重要的主观因素，就是在软件开发和维护过程中，使用不正确的开发方法和落后的开发技术，表现在：

（1）对软件开发与维护存在许多错误认识。

①软件与程序的概念不清，认为“所谓软件开发就是编写程序并设法使它运行”。

②忽视软件开发前期的调研和需求分析工作，认为“有一个对目标的概括描述就足以着手编写程序了，许多细节可以在以后再补充”。

③开发过程没有统一的、规范的方法论的指导，文件资料不齐全，忽视人与人的交流。

④忽视测试阶段的工作，提交用户的软件品质差。

⑤轻视软件的维护，认为“软件投入生产性运行以后需要的维护工作并不多，而且维护是一件

很容易做的简单工作”。

（2）软件开发与维护的方法不正确。

软件生产至今尚未摆脱手工方式，长期以来始终没有一种高效的开发方法，导致了软件生产效率非常低。而软件开发的手工行为还产生了一个致命的问题，就是为应用“量身订做”软件。软件产品大多是为客户“定做”的，通用性差，所有软件都要全部从头开发，使得效率问题更加恶化。

由于软件涉及人类社会的各行各业，常常涉及其他领域的专门知识，而软件开发实际上应该有比较明确的分工，例如，业务（咨询）专家负责规范、描述用户的业务流程；系统分析员负责将用户需求转换为系统功能和性能需求；系统设计人员负责规划系统框架，构建系统模型，设计系统蓝图；软件工程师负责实施设计方案；测试工程师负责软件测试，发现软件缺陷；质量工程师负责制定软件质量标准，监督软件开发过程，评估软件质量等。也就是说，软件开发过程中不同的工作应该有不同的人员专门负责实施，然而，实际工作中，却经常是一个项目中只有几个软件程序员，既做系统分析又做系统设计，最后还要编写代码。这样不仅对软件工程师提出了过高的、难以胜任的要求，同时也造成软件对开发者的依赖性非常强，开发队伍中一旦走掉一个主力，有可能使整个产品的开发工作都处于停止状态。

1.2.3 解决软件危机的途径

为解决软件危机，许多计算机和软件科学家尝试着将其他工程领域中行之有效的工程学知识运用到软件开发工作中来。人们认识到为了解决软件危机，既要有技术措施（方法和工具），还要有必要的组织管理措施。一方面先进的开发方法和工具，不仅可以提高软件开发及维护的效率，也保证了软件的质量。另外由于软件开发活动不是简单的个体行为，要保证软件开发活动顺利地、有效地、高质量地完成，严密地组织、严格地管理和各类人员协调一致地工作，也是必不可少的因素。实践表明，有经验的组织管理人员，行之有效的原理、概念、技术和方法，都将对软件的有效开发起到重要作用。最后得出一个结论：按工程化的原则和方法组织软件开发工作是有效的，是摆脱软件危机的一个主要出路。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学科。

1.3 软件工程

1968年秋季，NATO（北约）的科技委员会召集了近50名一流的程序工程人员、计算机科学家和工业界巨头，讨论和制定摆脱“软件危机”的对策。提出了软件工程（Software Engineering）这个概念。

1.3.1 软件工程的定义

软件工程是一门研究如何用系统化、规范化、数量化等工程原则和方法去进行软件的开发和维护的学科。实质上，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正确的管理方法和最先进的软件开发技术结合起来，应用到软件开发和维护过程中，来解决软件危机问题，生产出无故障的、及时交付的、在预算之内的和满足用户需求的软件。

1993年，IEEE为软件工程下的定义是：软件工程是将系统化的、规范的、可度量的方法应用

于软件的开发、运行和维护过程，即将工程化应用于软件中的方法的研究。

1.3.2 软件工程的内容

软件工程是一种层次化的技术，如图 1-4 所示。

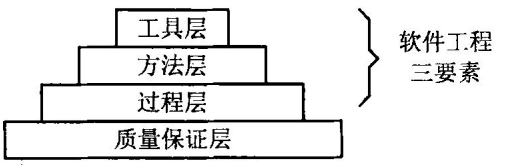


图 1-4 软件工程层次图

和任何工程方法一样，软件工程以质量为关注焦点。以全面质量管理及相关的现代管理理念，为软件工程奠定强有力的根基。全面的质量管理和质量需求是推动软件过程不断改进的动力，正是这种改进的动力导致了更加成熟的软件工程方法不断涌现。

软件工程的基础是过程层。软件工程过程是为获得软件产品，在软件工具支持下由软件人员完成的一系列软件工程活动。过程层将方法和工具结合在一起，它定义了一组关键过程区域的框架，定义了方法使用的顺序、要求交付的文档资料、为保证质量和协调变化所需要的管理，以及软件开发各个阶段完成的里程碑。其目的是保证软件工程技术被有效地应用，使得软件能够被及时地、高质量和合理地开发出来。

方法层提供了软件开发的各种方法。方法覆盖了一系列的任务，包括项目计划与估算方法；需求分析和设计的方法；编程、测试方法及维护方法等。软件工程方法依赖于一组基本原则，这些原则控制了每一个技术区域，包括建模活动和其他描述技术。

软件工具为软件工程方法提供了自动的或半自动的软件支撑环境。目前，已经推出了许多软件工具。将这些软件工具集成起来，即构成称之为计算机辅助软件工程（CASE）的软件开发支撑系统。CASE 将各种软件工具、开发机器和一个存放开发过程信息的工程数据库组合起来形成一个软件工程环境。使用软件工程工具可以有效地改善软件开发过程，提高软件开发的效率，降低开发成本。

一般将方法、工具和过程称为软件工程的三要素。由此可见，软件工程是一门涉及内容广泛的学科，所依据的基础理论极为丰富，包括数学、计算机科学、经济学、工程学、管理学和心理学等其他学科，如图 1-5 所示。其研究的内容包括软件开发技术和软件项目管理。

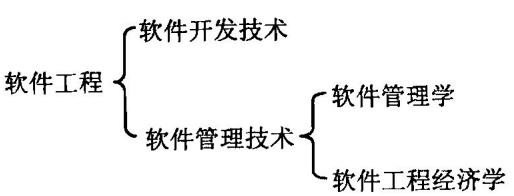


图 1-5 软件工程主要研究内容

软件开发技术包括软件开发方法学、软件工具和软件工程环境。

软件项目管理包括软件度量、项目估算、进度控制、人员组织、配置管理、项目计划等。

1.3.3　软件工程的基本原理

自从“软件工程”这一概念提出以来，不少研究软件工程的学者和专家们从系统工程和工程管理学的观点出发，提出了许多有关软件工程的准则，其中著名软件工程专家 B.W.Boehm 根据自己多年开发软件的经验，并综合了这些专家的意见，在 1983 年提出的软件工程的 7 条基本原理，并认为这些原理是确保软件产品质量和提高软件开发效率的最小原理集合。7 条基本原理的内容如下：

原理 1：严格按照软件生命周期各阶段的计划进行管理。

有统计资料表明，在不成功的软件项目中有 50%左右是由于计划不周，或没有严格按计划进行软件工程管理造成的，把建立完善的计划作为第一条基本原理是通过总结经验教训得来的。所以为保证软件开发质量，应将软件开发过程划分为若干阶段，并相应地制定出切实可行的计划。软件的管理人员须严格按照计划管理软件开发与维护工作，不应受客户或上级人员的影响而擅自背离预定计划，如实践中确实发现计划中存在问题需要修改时，应进行验证，并重新讨论修改意见，通过后方可执行。

原理 2：坚持进行阶段评审。

有统计数据表明，软件的大部分错误是在编码之前造成的。软件设计错误约占软件错误的 63%，而编码错误仅占软件错误的 37%。因此，软件的质量保证工作不能等到编码阶段结束之后再进行。错误发现得越晚，改正越麻烦，相应地，付出的代价也越高。因此，在每个阶段严格地进行评审，尽早发现在软件开发过程中所犯的错误，这也是一条很重要的原则。

原理 3：实施严格的产品控制。

在软件开发和维护过程中，修改软件是经常发生的，一般情况下，应严格控制对需求的更改，因为需求的更改往往影响后面阶段的各项工作，改变一项需求往往需要付出较高的代价。但是，实际应用中，由于外部环境不断变化，用户的需求改变是一种客观需要，而且分析与设计工作也常常会出现考虑不周的情况。因此对软件的修改是不可避免的，只能依靠有效地控制管理顺应这种要求，即实施基准配置管理，保证软件的各个配置成分的一致性。基准配置也称基线配置，它的基本管理思想是：对软件的各项修改建议，严格按照规程进行评审和控制，获得批准以后才能实施修改，同时要保证其他有关的各阶段的文档和代码相应改动，以保证软件配置成分的一致性。

原理 4：采用先进的程序设计技术。

实践表明，采用先进的技术既可以提高软件质量，又可以提高软件开发和维护的效率。如结构化分析与设计技术、面向对象的分析和设计技术等。人们对先进的程序设计技术的研究仍在继续，选择一种好的程序设计技术就意味着能够有效地进行软件开发和维护工作。

原理 5：结果应能清楚地审查。

由于软件产品不同于任何一件硬件产品，是一种看不见、摸不着的逻辑产品，软件开发小组的工作进展情况可见性差，因此对产品的开发过程难于评价和管理。为提高软件开发过程的可见性，更好地进行管理，应明确各软件小组的责任，严格规定软件产品的标准，在软件的完成期限内，对所得的结果进行审查，以确定软件产品是否符合标准。

原理 6：开发小组的人员应该少而精。

目前，大多数的软件需要多人合作开发，这种开发方式可以减少每个成员的工作量，缩短开发时间。实践表明，对于一个复杂项目的开发工作，并非开发人员越多越好，因为当人数很多时，人员之间的信息交流量相应增加，出现问题的可能性也同样增大，这样反而会降低程序的质量和开发

效率。因此，软件开发小组成员的选择，应基于素质高、人数适宜的原则，少而精的开发小组有利于软件开发的进行，这也是软件工程的一条基本原理。

原理 7：承认不断改进软件工程实践的必要性。

这条原理的含义是，在采用上述 6 条基本原理进行软件开发的过程中，要时刻不忘吐故纳新的道理，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，这样才能保证软件工程思想有效地指导实践活动，最大限度发挥它的作用。

这 7 条原理是相互独立，缺一不可的，它们从大的方向概括了软件工程的方方面面。实践过程中，可以对这些原理进行细化和再生，灵活运用这些原理指导软件开发，必然会获得最后的成功。

1.3.4 软件工程项目的基本目标

软件工程是一门工程性学科，其目的是采用各种技术上和管理上的手段组织实施软件工程项目，成功地建造软件系统。项目成功的几个主要目标是：付出较低的开发成本，在用户规定时限内，获得功能、性能方面满足用户需求的软件；开发的软件移植性较好；易于维护且维护费用较低；软件系统的可靠性高。

在实际开发过程中，要同时满足上述几个目标是非常困难的。这些目标之间有些是互补关系，有些是互斥关系，因此，在解决问题时，要根据具体情况，必要时牺牲某个目标以满足其他优先级更高的目标，只要保证总体目标满足要求，软件开发就是成功的。

1.4 软件过程和软件生存期

1.4.1 软件过程

进入 20 世纪 90 年代，软件工程领域提出了软件过程的概念。所谓软件过程是为了获得高质量软件产品，在软件工具支持下由软件人员完成的一系列软件工程活动。它规定了开发软件所需完成的各项任务的步骤，它与软件生存期、生存期模型、软件开发工具和参与开发的人员等多方面因素有关。不同的开发机构可制定自己的软件过程，同一开发机构也可对不同的软件项目制定专门的软件过程，对软件过程没有统一的规定，只要求它的内容科学、有效。1995 年，国际标准化组织公布了新的国际标准《ISO/IEC 12207 信息技术——软件生存期过程》，该标准将软件开发需要完成的活动概括为主要过程、支持过程和组织过程三大活动，每个大的过程里又包含若干具体过程，共 17 个。过程的结构如图 1-6 所示。下面简要说明每个过程的主要任务。

1.4.1.1 主要过程

1. 获取

需方获取系统、软件产品或软件服务的活动。由需方定义需求，然后委托供方或双方一起进行需求分析，需求分析的结果要由需方确认。需方应该准备招标书、合同以及验收条款。

2. 供应

由供方向需方提供系统、软件产品或软件服务的活动。供方要对需求进行评审，然后准备投标，中标后签订合同。制定项目计划、实施计划、开展评审、交付产品。