高等学校计算机应用规划教材

软件工程实用案例教程

- 软件工程综述
- 软件过程
- 可行性研究
- 结构化需求分析
- 结构化软件设计
- 面向对象的需求分析
- 面向对象的设计
- 基于构件的开发
- 软件项目的测试
- 软件实施、维护与进化
- 软件工程标准与文档

梁 洁 金 兰 主 编
张 硕 宋亚岚 孔德华 副主编

清华大学出版社

高等学校计算机应用规划教材

软件工程实用案例教程

梁 洁 金 兰 主 编

张 硕 宋亚岚 孔德华 副主编

清华大学出版社

北 京

内 容 简 介

本书结合软件工程的发展与教学需要，系统地阐述了软件工程这一领域的基本概念、原理与方法。本书共包括 11 章，主要内容有：软件工程综述，软件过程，可行性研究，结构化需求分析，结构化软件设计，面向对象的需求分析，面向对象的设计，基于构件的开发，软件项目的测试，软件实施、维护与进化，软件工程标准与文档等。

全书内容丰富、组织严谨，原理和方法结合密切，结构化方法和面向对象的方法均有一个实例贯穿始终，丰富的图表和应用实例有助于培养读者的实际分析设计能力和文档写作能力，书中含有丰富的例题与习题便于教学及读者自学。

本书可以作为高等院校软件工程专业、计算机科学与技术专业、计算机应用专业，以及其他相关专业高年级本科生的教材，同时可作为从事软件分析、设计与开发人员的参考书。

图书在版编目(CIP)数据

软件工程实用案例教程 /梁洁，金兰 主编. —北京：清华大学出版社，2019

(高等学校计算机应用规划教材)

ISBN 978-7-302-52277-5

Ⅰ. ①软… Ⅱ. ①梁… ②金… Ⅲ. ①软件工程－案例－高等学校－教材 Ⅳ. ①TP311.5

中国版本图书馆 CIP 数据核字(2019)第 024842 号

责任编辑：刘金喜
封面设计：孔祥峰
版式设计：思创景点
责任校对：牛艳敏
责任印制：杨　艳

出版发行：清华大学出版社

网　　址：http://www.tup.com.cn，http://www.wqbook.com

地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084

社 总 机：010-62770175　　邮　　购：010-62786544

投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn

质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn

印 刷 者：北京鑫丰华彩印有限公司
装 订 者：三河市溧源装订厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**19　　**字　　数：**439 千字
版　　次：2019 年 7 月第 1 版　　**印　　次：**2019 年 7 月第 1 次印刷
定　　价：58.00 元

产品编号：079700-01

前　言

软件工程学是一门综合型应用科学，它将计算机科学理论与现代工程方法论相结合，着重研究软件过程模型、设计方法及工程开发技术和工具，以指导软件的生产和管理。随着计算机科学和软件产业的迅猛发展，软件工程学已经成为一个重要的计算机分支学科，也是一个异常活跃的研究领域，新方法、新技术不断涌现。

“软件工程”是计算机专业学生必修的一门专业课程，也是工科各专业学生在计算机应用方面的一门重要选修课程。在多年的软件工程教学过程中，我们的教研团队参考或使用过许多软件工程教材，但很多教材大都侧重理论的讲解，教材中的案例较少，尤其没有一个完整、系统的软件工程案例贯穿其中，由于本科生普遍缺乏软件工程项目开发的实践经验，因此学习软件工程课程感觉非常抽象、空泛与枯燥。为改变目前软件工程教学中这种抽象、空泛的学习现状，我们决定编写本书。

本书的特色可以归纳为以下五点。

(1) 从软件危机、软件过程模型，再到软件可行性分析、需求分析、系统设计，都引入大量实际案例，解决软件工程理论教学过程中过于抽象和晦涩的问题；在第 4 章结构化的分析方法中引入了“电梯控制系统”案例的完整分析，主要是考虑到结构化分析与设计的优势在嵌入式系统中会更加凸显，而在第 6、7 章面向对象的需求分析与设计方法中引入了“网上计算机销售系统”案例，在电子商务如此发达的今天，让学生对自己熟悉的“网上销售系统”进行分析设计，有利于收集需求，同时能激发学生的学习兴趣。

(2) 第 4 章结构化需求分析详细介绍了业务需求、用户需求和系统需求 3 个层次需求各自的特点。系统讲解了需求工程活动，包括需求获取、需求分析、需求规格说明、需求验证和需求管理。其中需求分析包括过程建模和数据建模。过程建模引入“食品订货系统”案例。数据建模引入“学生研讨班”案例和“EMS 表单项目”案例。最后引入了“电梯控制系统”完整案例，按照创建上下文、建立 0 层图、产生 *N* 层图、定义逻辑说明、定义数据存储和数据流的步骤进行了系统完整的需求分析。

(3) 第 7 章面向对象的设计，遵循“分析类+设计模式=设计类”原则，逻辑体系架构的介绍从分层结构到三层架构再到熟知 MVC 模式演化，并对软件的 MVC 设计模式进行了详细的介绍，这种软件分层模式的理解与掌握对于从事软件开发的读者尤为重要。目前市面上的绝大多数的软件都是采用多层框架结构来实现的，依据构件的划分，对构件内部分析阶段得到的实体类，结合三层的设计模式，补充边界类、控制类、模型类得到可以用来指导开发的详细设计类图。

(4) 第 8 章基于构件的开发，该章节的主要内容是基于一个构件详细设计类图进行编码开发的，构件详细设计类图是第 7 章设计阶段的工作成果，将分析设计的模型直接指导编码，帮助读者领会软件工程的真正意义所在。很多从事软件开发的程序员容易“重编程轻设计”，

往往问题还没想清楚就开始编码，这章的内容告诉读者只要分析设计做得详尽，编码就会水到渠成。

(5) 本书提供第 8 章的构件开发代码，以及全套软件工程文档，供读者阅读及下载使用。

本书由梁洁、金兰主编，张硕、宋亚岚、孔德华任副主编。其中，梁洁编写第 1 章、第 2 章、第 6～8 章、第 11 章，金兰编写第 4 章和第 5 章，张硕编写第 9 章和第 10 章，宋亚岚编写第 3 章。全书由宋亚岚、孔德华统稿。

本书的宗旨是为了提高软件工程课程的教学质量，让学生真正“学有所用”。本书具有内容组织科学、合理、系统，注重理论与实践并重的特点，同时课后都配有和教学内容完全一致的练习题供读者思考与巩固知识。

本书可以作为高等院校软件工程专业、计算机科学与技术专业、计算机应用专业，以及其他相关专业高年级本科生的教材，同时可供从事软件工程、计算机应用、计算机软件专业，以及其他相关专业的科研人员、软件开发人员及有关大专院校的师生参考。

在本书的编写过程中得到了武昌首义学院的领导和同事们的支持与帮助，在此一并表示感谢。

由于编者水平有限，书中难免存在不妥与疏漏之处，敬请广大读者批评指正。

本书 PPT 课件等相关教学资源可通过 http://www.tupwk.com.cn/downpage 下载。

服务邮箱：wkservice@163.com。

编者

2019 年 3 月

目　　录

第1章　软件工程综述

本章的目标是介绍软件工程的概念，并为理解本书其他部分内容提供一个框架。读完本章，你将了解以下内容。

- 什么是软件？软件有什么特点？
- 什么是软件危机？为什么会出现软件危机？
- 什么是软件工程？为什么它很重要？
- 软件工程的目标是什么？软件工程三要素是什么？
- 什么是软件工程的多样性？软件工程通用原则有哪些？
- 了解道德和职业问题对于软件工程的重要性。

1.1　软件工程的背景

现代社会离不开软件，国家基础设施和公共建设都是基于计算机的系统控制，大多数电子产品都有控制软件。工业制造和分销已经完全计算机化，金融系统也是如此。娱乐业，包括音乐产业、计算机游戏产业、电影和电视产业，同样也是一个软件密集型的产业。

1.1.1　软件及其特性

1. 软件

计算机软件是由计算机程序的发展而形成的一个概念。它是与计算机系统操作有关的程序、规程、规则及其文档和数据的统称。软件由两部分组成：一是机器可执行的程序和有关的数据；二是与软件开发、运行、维护、使用和培训有关的文档。

程序是按事先设计的功能和性能要求执行的语句序列。数据是程序所处理信息的数据结构。文档则是与程序开发、维护和使用相关的各种图文资料，如各种规格说明书、设计说明书、用户手册等。在文档中记录着软件开发的活动和阶段成果。

软件产品主要有以下两类。

(1) 通用软件产品，由软件开发机构制作，在市场上公开销售，可以独立使用。这类软件产品有数据库软件、文字处理软件、绘图软件及工程管理工具等，还包括用于特定目的的所谓“垂直”应用产品，如图书馆信息系统、财务系统等。

(2) 定制软件产品，这些产品受特定的客户委托，由软件承包商专门为某类客户开发。

这类软件有电子设备的控制系统、特定的业务处理系统和空中交通管制系统等。

这两类产品的一个重要区别在于：在通用软件产品中，软件描述由开发人员自己完成，而在定制软件产品中，软件描述通常是由客户给出，开发人员必须按客户要求进行开发。

然而这两类产品之间的界限正在变得越来越模糊。现在更多的公司从一个通用软件产品开始进行定制处理，以满足特别客户的具体要求。例如，企业资源规划(ERP)这类系统，需要通过嵌入一系列信息，如业务和操作规则及各种报表等，以适应一个新的企业。

2. 软件的特点

软件是一种逻辑产品而不是实物产品，软件功能的发挥依赖于硬件和软件的运行环境，没有计算机硬件的支持，软件将毫无实用价值。若要对软件有一个全面而正确的理解，我们应从软件的本质来剖析软件的特征。

1) 软件的固有特性

(1) 复杂性。软件是一个庞大的逻辑系统。一方面在软件中要客观地体现人类社会的事物，反映业务流程的自然规律；另一方面在软件中还要集成多种多样的功能，以满足用户在激烈的竞争中对大量信息及时处理、传输、存储等方面的需求，这使软件变得非常复杂。如表 1-1 所示为某购物广场的系统总体结构表，系统中包括大量的业务活动和主题数据库之间的交互，表中“C”代表数据的创建与修改，“U”代表数据的使用。

表 1-1　某购物广场的系统总体结构表

主题数据库 / 业务过程	客户数据库	客户服务主题库	赠品信息库	费用信息库	表基本信息库	计费标准信息库	物业工作标准信息库	物业派工信息库	维修信息库	设备信息库	对外宣传信息库	商铺信息库	租约主题库	财务信息库	员工信息库	员工培训信息库
宣传促销											C					
广告制作											C					
广告招租											C					
商铺管理												C				
租户管理	C											U				
租约管理	U											U	C			
经营分析	U	U	U	U		U		U	U	U	U		U	U	U	U
客户服务	U	C	C	C	U	U			U			U	U	C		
抄表处理					C	C										
保洁管理							C	C							U	
安保管理							C	C							U	
维修管理		U							C	C					U	
往来账目管理			U	U	U				U	U		U	U	C	U	U
租户费用结算												U	U	C		
凭证结转管理			U	U					U	U		U	U	C		

(续表)

主题数据库 / 业务过程	客户数据库	客户服务主题库	赠品信息库	费用信息库	表基本信息库	计费标准信息库	物业工作标准信息库	物业派工信息库	维修信息库	设备信息库	对外宣传信息库	商铺信息库	租约主题库	财务信息库	员工信息库	员工培训信息库
财务分析	U		U	U					U	U	U		U	U		
员工档案															C	
人员招聘															C	
员工薪酬管理															C	
员工培训管理															U	C

(2) 抽象性。软件是人们经过大脑思维后加工出来的产品，一般存储在内存、磁盘、光盘等载体上，我们无法观察到它的具体形态，这就导致了软件开发不仅工作量难以估计，进度难以控制，而质量也难以把握。

(3) 依赖性。软件必须和运行软件的机器保持一致，软件的开发和运行往往受计算机硬件的限制，两者对计算机系统有着不同程度的依赖性。软件与计算机硬件的这种密切相关性与依赖性，是一般产品没有的特性。为了减少这种依赖性，在软件开发中提出了软件的可移植性问题。

(4) 软件使用特性。软件的价值在于应用。软件产品不会因多次反复使用而磨损老化，一个久经考验的优质软件，可以长期使用。软件投入运行后，总会存在缺陷甚至暴露出潜伏的错误，需要进行长期的维护及再开发。

2) 软件生产特性

(1) 软件开发特性。由于软件固有的特性，使得软件的开发不仅具有技术复杂性，还有管理复杂性。技术复杂性体现在软件提供的功能比一般硬件产品提供的功能多，而且功能的实现具有多样性，需要在各种实现中做出选择，更有实现算法上的优化带来的不同，而实现上的差异会带来使用上的差别，例如，仅用户注册登录功能就有多种实现。管理上的复杂性表现在：第一，软件产品的能见度低(包括如何使用文档表示的概念能见度)，要看到软件开发进度比看到有形产品的进度困难得多；第二，软件结构的合理性差，结构不合理使软件管理复杂性随软件规模增大而呈指数增长。因此，领导一个庞大人员的项目组进行规模化生产并非易事，软件开发比硬件开发更依赖于开发人员的团队精神、智力和对开发人员的管理与组织。

(2) 软件产品形式的特性。软件产品的设计成本高昂而生产成本极低。硬件产品试制成功之后，批量生产需要建设生产线，投入大量的人力、物力和资金，生产过程中还要对产品进行质量控制，对每件产品进行严格的检验。然而，软件是把人的知识与技术转化为信息的逻辑产品，开发成功之后，只需对原版软件进行复制即可，而大量人力、物力、资金的投入和质量控制、软件产品检验都是在软件开发中进行的。由于软件的复制非常容易，软件的知识产权保护就显得极为重要。

(3) 软件维护特性。软件在运行过程中的维护工作比硬件复杂得多。首先，软件投入运

行后，总会存在缺陷甚至暴露出潜伏的错误，需要进行纠错性维护。其次，用户可能要求完善软件性能，对软件产品进行修改，进行完善性维护。当支撑软件产品运行的硬件或软件环境改变时，也需要对软件产品进行修改，进行适应性维护。软件的缺陷或错误属于逻辑性的，因此不需要更换某种配件，而是修改程序，纠正逻辑缺陷，改正错误，提高性能，增加适应性。当软件产品规模庞大、内部的逻辑关系复杂时，经常会发生纠正一个错误而产生新的错误的情况，因此，软件产品的维护要比硬件产品的维护工作量大且复杂。

1.1.2　软件危机

20 世纪 60—70 年代，由于软件规模的扩大、功能的增强和复杂性的增加，使得在一定时间内依靠少数人开发一个软件变得越来越困难。在软件开发中经常会出现时间延迟、预算超支、质量得不到保证、移植性差等问题，甚至有的项目在耗费了大量人力、财力后，由于离目标相差甚远而宣布失败。这种情况使人们认识到“软件危机”的存在。

最典型“软件危机”的项目失败案例

1963—1966 年，IBM 公司开发 OS 360 系统，共有 4000 多个模块，约 1 000 000 条指令，投入 5000 人/年，耗资数亿美元，结果还是延期交付。在交付使用后的系统中仍发现大量(2000 个以上)错误。

该项目负责人 Brooks F.D.事后总结了他在开发过程中的沉痛教训时说：“正像逃亡的野兽落在泥潭中垂死挣扎一样，越是挣扎，陷得越深。最后无法逃脱灭顶的灾难，……程序设计工作正像这样一个泥潭，……一批程序员被迫在泥潭中拼命挣扎，……谁也没料到我们竟会陷入这样的困境……”

1. 软件危机的突出表现

(1) 软件生产率低。软件生产率提高的速度远远跟不上计算机应用的迅速普及和深入的趋势。落后的生产方式与开发人员的匮乏，使得软件产品的供需差距不断扩大。由于缺乏系统有效的方法，现有的开发知识、经验和相关数据难以积累与复用，另外，低水平的重复开发过程浪费了大量的人力、物力、财力和时间。人们为不能充分发挥计算机硬件提供的巨大潜力而苦恼。

(2) 软件产品常常与用户要求不一致。开发人员与用户之间的信息交流往往存在障碍，除了知识背景的差异，缺少合适的交流方法及需求描述工具也是一个重要原因。这使得获取的需求经常存在二义性、遗漏，甚至是错误。由于开发人员对用户需求的理解与用户的本意有所差异，以致造成开发中后期需求与现实之间的矛盾集中暴露。

(3) 软件规模的增长，带来了复杂度的增加。由于缺乏有效的软件开发方法和工具的支持，过分依靠程序设计人员在软件开发过程中的技巧和创造性，所以，软件的可靠性往往随着软件规模的增长而下降，质量保障越来越困难。

(4) 不可维护性突出。软件的局限性和欠灵活性，不仅使错误非常难以改正，而且不能适应新的硬件环境，也很难根据需要增加一些新的功能。整个软件维护过程除程序外，没有

适当的文档资料可供参考。

(5) 软件文档不完全、不一致。软件文档是计算机软件的重要组成部分，在开发过程中，管理人员需要使用这些文档资料来管理软件项目；技术人员则需要利用文档资料进行信息交流；用户也需要通过文档来认识软件，对软件进行验收。但是，由于软件项目管理工作的不规范，软件文档往往不完整、不一致，这给软件的开发、交流、管理、维护等都带来了困难。

2. 产生软件危机的原因

软件危机是指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。这些问题不仅局限于那些“不能正确完成功能”的软件，还包括我们如何开发软件、如何维护大量已有软件、如何使软件开发速度与对软件需求增长相适应等问题。产生软件危机的主要原因有以下几个。

(1) 软件独有的特点给开发和维护带来困难。由于软件的抽象性、复杂性与不可预见性，使得软件在运行之前，对开发过程的进展情况较难衡量，软件的错误发现较晚，软件的质量也较难评价，因此，管理和控制软件开发过程相当困难。此外，软件错误具有隐蔽性，往往在很长时间里软件仍可能需要改错，这在客观上使得软件维护较为困难。

(2) 软件人员的错误认识。相当多的软件专业人员对软件开发和维护还有不少的错误观念。例如，软件开发就是编写程序，忽视软件需求分析的重要性，轻视文档的作用，轻视软件维护等。这些错误认识加重了软件危机的影响。

(3) 软件开发工具自动化程度低。尽管软件开发工具比 30 年前已经有了很大的进步，但直到今天，软件开发仍然离不开工程人员的个人创造与手工操作，软件生产仍不可能像硬件设备生产一样，达到高度制度自动化。这样不仅浪费了大量的财力、物力和宝贵的人力资源，还无法避免低水平的重复性劳动，而且软件的质量也难以保证。此外，软件生产工程化管理程度低，致使软件项目管理混乱，难以保障软件项目成本、开发进度按计划执行。

1.2　软件工程概述

为了克服“软件危机”，1968 年在北大西洋公约组织(NATO)召开的计算机科学会议上，Fritz Bauer 首先提出“软件工程”的概念，试图用工程的方法和管理手段，将软件开发纳入工程化的轨道，以便开发出成本低、功能强、可靠性高的软件产品。几十年来，人们一直在努力探索克服软件危机的途径。

1.2.1　软件工程的基本概念

许多计算机和软件科学家尝试把其他工程领域中行之有效的工程学知识应用到软件开发工作中。经过不断实践和总结，最后得出这样的结论：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。

虽然软件工程概念的提出已有 50 余年，但直到目前为止，软件工程概念的定义并没有统一。

在 NATO 会议上，Fritz Bauer 对于软件工程的定义是：“为了经济地获得可靠的、能在实际机器上高效运行的软件，而建立和使用的健全的工程原则。”

对于软件工程，美国电气与电子工程师学会(IEEE)给出了如下定义。

软件工程是：①将系统化的、规范化的、可量化的方法应用于软件的开发、运行和维护中，即将工程化方法应用于软件；②对于①中所述方法的研究。

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当前能够得到的最好的技术方法结合起来，以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。

值得注意的是，对于某个软件开发队伍来说可能是“系统化的、规范的、可量化的”方法，而对于另外一个团队却可能是负担。因此，我们需要规范，也需要可适应性和灵活性。

1.2.2　软件工程的目标

软件工程的目标是基于软件项目目标的成功实现而提出的，主要体现在以下几个方面。

- 软件开发成本较低。
- 软件功能能够满足用户的需求。
- 软件性能较好。
- 软件可靠性高。
- 软件易于使用、维护和移植。
- 能够按时完成开发任务，并及时交付使用。

在实际开发中，企图让以上几个质量目标同时达到理想的程度往往是不现实的。软件工程目标之间存在的相互关系如图 1-1 所示。从图中可以看出：有些目标之间是相互补充的，如易于维护和高可靠性之间、低开发成本与按时交付之间；有些目标是彼此相互冲突的，若只考虑降低开发成本，很可能同时也降低了软件的可靠性，如果一味追求提高软件的性能，可能造成开发出的软件对硬件的依赖性较强，从而影响软件的可移植性。不同的应用对软件质量的要求不同，如对实时系统来说，其可靠性和效率比较重要；对生命周期较长的软件来说，其可移植性和可维护性比较重要。

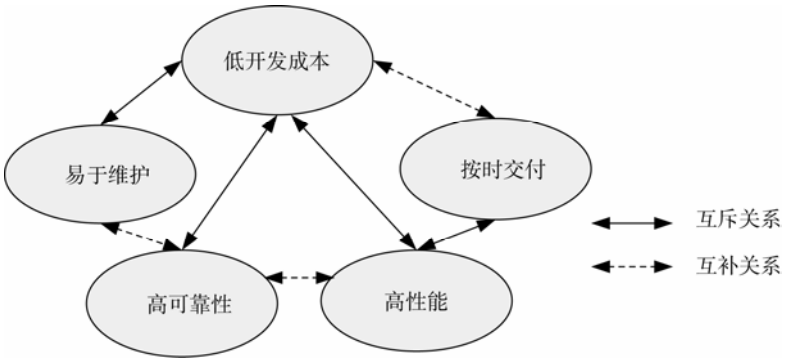

图 1-1　软件工程目标之间的关系

1.2.3　软件工程三要素

软件工程是一种层次的技术，如图 1-2 所示。任何工程方法包括软件工程必须构建在质量承诺的基础之上。全面质量管理的理念促进了持续不断的过程改进文化，正是这种文化最终引导人们开发出更有效的软件工程方法。支持软件工程的根基在于质量关注点。

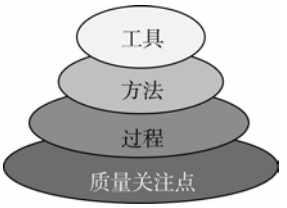

图 1-2　软件工程层次图

软件工程的首要问题是质量问题。软件工程的目的就是在多种目标的冲突之间取得一定程度的平衡。因此，在涉及平衡软件工程目标这个问题时，软件的质量应该摆在最重要的位置上加以考虑。

软件质量可用功能性、可靠性、可用性、效率、可维护性和可移植性等特性来评价。

- 功能性是指软件所实现的功能能达到它的设计规范和满足用户需求的程度。
- 可靠性是指在规定的时间和条件下，软件能够正常维持其工作的能力。
- 可用性是指为了使用该软件所需要的能力。
- 效率是指在规定的条件下用软件实现某种功能所需要的计算机资源的有效性。
- 可维护性是指当环境改变和软件运行发生故障时，为了使其恢复正常运行所做的努力的程度。
- 可移植性是指软件从某一环境转移到另一个环境时所有努力的程度。

在不同类型的应用系统中对软件的质量要求是不同的。将软件的质量作为根基，软件工程研究的内容主要包括过程、方法、工具这三个方面，也被称为“软件工程三要素”。

1) 软件工程的基础是过程(Process)

无论是什么软件系统，一个通用的软件工程过程框架通常包含以下 6 个活动。

(1) 沟通。在技术工作开始之前，和客户及其他利益相关者的沟通与协作是极其重要的，其目的是了解利益相关者的项目目标，并收集需求及定义软件特性和功能。

(2) 策划。如果有地图，任何复杂的旅程都可以变得简单。软件项目好比一个复杂的旅程，策划活动就像是创建一个地图，以指导团队的项目旅程，这个地图称为软件项目计划，它定义和描述了软件工程工作，包括需要执行的技术任务、可能的风险、资源需求、工作产品和工作进度计划。

(3) 建模。无论你是庭院设计师、桥梁建造师、航空工程师、工匠还是建筑师，每天的工作都离不开模型。你会画一张草图来辅助理解整个项目大的构想——体系结构、不同的构件如何结合，以及其他一些特性。如果需要，可以把草图不断细化，以便更好地理解问题并找到解决方案。软件工程师也是如此，需要利用模型来更好地理解软件需求，并完成符合这些需求的软件设计。

(4) 构建。必须要对所做的设计进行构建，包括编码(手写的或者自动生成的)和测试，后者用于发现编码中的错误。

(5) 部署。软件(全部或者部分增量)交付给用户，用户对其进行评测并给出反馈意见。

(6) 进化。软件随不同的客户和变化的市场需求而进行修改。

上述 6 个通用框架活动既适用于简单小程序的开发，也可用于 Web 应用程序的构建，以及基于计算机的大型复杂系统工程。不同的应用案例中，软件过程的细节可能差别很大，但是框架活动都是一致的。

对于软件项目来说，随着项目的开展，框架活动可以迭代应用。也就是说，在项目的多次迭代过程中，沟通、策划、建模、构建、部署等活动不断重复。每次项目迭代都会产生一个软件增量，每个软件增量都实现了部分的软件特性和功能。随着每一次增量的产生，软件将逐渐完善。

软件工程框架活动由很多支持性活动补充实现。通常，这些支持性活动贯穿软件项目始终，以帮助软件团队管理和控制项目进度、质量、变更和风险。典型的支持性活动有以下几个。

- 软件项目跟踪和控制：项目组根据计划来评估项目进度，并且采取必要的措施保证项目按进度计划进行。
- 风险管理：对可能影响项目成果或者产品质量的风险进行评估。
- 软件质量保证：确定和执行保证软件质量的活动。
- 技术评审：评估软件工程产品，尽量在错误传播到下一个活动之前发现并清除。
- 测量：定义和收集过程、项目及产品的度量，以帮助团队在发布软件时满足利益相关者的要求。同时，测量还可与其他框架活动和支持性活动配合使用。
- 软件配置管理：在整个软件过程中管理变更所带来的影响。
- 可复用管理：定义工作产品复用的标准(包括软件构件)，并且建立构件复用机制。
- 工作产品的准备和生产：包括生产产品(如建模、文档、日志、表格和列表等)所必需的活动。

2) 软件工程方法(Method)

软件工程方法为构建软件提供技术上的解决方法(如何做)，主要有以下两种方法。

(1) 结构化方法。结构化方法是传统的基于软件生命周期的软件工程方法，自 20 世纪 70 年代产生以来，获得了极有成效的软件项目应用。结构化方法是以软件功能为目标来进行软件构建的，包括结构化分析、结构化设计、结构化实现、结构化维护等内容。这种方法主要是通过数据流模型来描述软件的数据加工过程，并通过数据流模型，由对软件的分析过渡到对软件的结构设计。

(2) 面向对象方法。面向对象方法是从现实世界中客观存在的事物出发来构造软件，包括面向对象分析、面向对象设计、面向对象实现、面向对象维护等内容。一个软件是为了解决某些问题，这些问题所涉及的业务范围被称作该软件的问题域。面向对象强调以问题域中的事物为中心来思考问题、认识问题，并根据这些事物的本质特征，把它抽象地表示为系统中的对象，作为系统的基本构成单位。确定问题域中的对象成分及其关系，建立软件系统对象模型，是面向对象分析与设计过程中的核心内容。自 20 世纪 80 年代以来，人们提出了许多有关面向对象的方法，其中由 Booch、Rumbaugh、Jacobson 等人提出的一系列面向对象方法成为主流方法，并被结合为统一建模语言(UML)，成为面向对象方法中的公认标准。

3) 软件工程工具(Tool)

软件工程工具是辅助软件开发、维护和管理的软件。

计算机辅助软件工程(Computer Aided Software Engineering，CASE)是通过一组集成化的工具，辅助软件开发人员实现各项活动的全部自动化，使软件产品在整个生命周期中的开发和维护生产率得到提高，质量得到保证。

下面简要介绍一些常用的软件工程工具。

(1) Microsoft Visio。Visio 提供了日常使用中的绝大多数框图的绘画功能，包括信息领域的各种原理图、设计图等，同时提供了部分信息领域的实物图。Microsoft Visio 的优点在于使用方便，安装后既可以单独运行，也可以在 Word 中作为对象插入，与 Microsoft Office 2003 集成良好，其图生成后在没有安装 Microsoft Visio 的 Office 工具中仍然能够查看。Visio 支持 UML 静态建模和动态建模，对于 UML 建模提供了单独的组织管理。作为 Office 大家庭的一员，它从 Microsoft Visio 2000 版本后在各种器件模板上有了多处增进。它是最通用的图表设计软件，易用性高，尤其对于不善于自己构造图的软件人员。

(2) Rational Rose。Rose 是一个完全的、具有能满足所有建模环境(如 Web 开发、数据建模、Visual Studio 和 C++)需求能力和灵活性的一套解决方案。Rose 2007 功能上可以完成 UML 的 9 种标准建模，即静态建模(包括用例图、类图、对象图、组件图、配置图)和动态建模(包括协作图、序列图、状态图、活动图)。作为一款优秀的分析和设计工具，Rose 具有强大的正向工程和逆向工程能力。正向工程指由设计产生代码，逆向工程指代码归纳出设计。通过逆向工程，Rose 可以对历史系统分析，然后进行改进，再通过正向工程产生新系统的代码，这样的设计方式称为“再工程”。

(3) Microsoft VSS(Visual Source Safe)。其解决了软件开发小组长期所面临的版本管理问题，可以有效地帮助项目开发组的负责人对项目程序进行管理，将所有的项目源文件(包括各种文件类型)以特有的方式存入数据库。开发组的成员不能对数据库中的文件进行直接修改，而是由版本管理器将项目或子项目的源程序复制到各个成员自己的工作目录下进行调试和修改，然后将修改后的项目文件提交给 VSS，由它进行综合更新。VSS 可以很便捷地与 Microsoft Access、Visual Basic、Visual C++、Visual FoxPro 及其他的开发工具集成在一起，一旦集成到开发环境中，就可以像控件一样使用，能很好地体现出 VSS 的易用性和强大功能。

(4) WinRunner。WinRunner 是一种企业级的功能测试工具，用于测试应用程序是否能够达到预期的功能及是否能够正常运行。通过自动录制、检测和回放用户的应用操作，WinRunner 能够有效地帮助测试人员对复杂的企业级应用的不同发布版本进行测试，提高测试人员的工作效率和质量，确保跨平台的、复杂的企业级应用无故障发布及长期稳定运行。

(5) LoadRunner。LoadRunner 是一种性能的负载测试工具，通过模拟成千上万的用户进行并发负载及实时的性能测试来查找问题，并能对整个企业架构进行测试。通过使用自动化性能测试工具，能够最大限度地缩短测试时间，优化性能和加速应用系统的发布周期。

1.2.4　软件工程的多样性

软件工程是生产软件的系统化的方法，它考虑到实际成本、进度、可靠性等问题，以及

软件生产者和消费者的需要。怎样实施这个系统化的方法取决于软件开发机构、软件类型和开发过程中的人员。没有一个通用的软件工程方法和技术适合所有的系统和公司。多样的软件工程方法和工具已经进化了 50 多年。决定使用哪一种软件工程方法和技术主要取决于要开发的应用的类型。这里有许多不同的类型的应用，具体如下。

(1) 系统软件：整套服务于其他程序的程序。某些系统软件(如编译器、编辑器、文件管理软件)处理复杂但确定的系统结构，另一些系统应用程序(如操作系统构建、驱动程序、网络软件、远程通信处理器)主要处理的是不确定的数据。

(2) 独立的应用软件：解决特定业务需要的独立应用程序。这类应用程序运行在本地计算机上，它们拥有所有必要的功能但不需要连接到网络上。这类应用有 PC 上的办公软件、CAD、图片处理软件等。

(3) 嵌入式控制系统：这类应用由一个软件控制系统和管理硬件设备。嵌入式控制系统在数量上远远多过其他类型的系统，包括汽车上仪表盘显示和控制防抱死的软件，以及微波炉上控制烹饪过程的软件。

(4) 以网络为中心的交互式应用：这类应用在远程计算机上执行，用户通过自己的 PC 或移动终端进行访问，包括基于浏览器的 Web 应用(如电子商务网站)和安装在移动设备上的 APP，以及基于云计算的广泛计算机资源的共享。

(5) 娱乐系统：这类系统主要是个人用户用于娱乐。大多数这类系统是各种各样的游戏。这类系统所提供的交互质量是娱乐系统和其他系统的一个重要区别。

(6) 建模和仿真系统：科学家和工程师用这类系统模拟物理过程或环境，这里有许多独立且相互交互的对象，通常是计算机密集型的，需要高性能的并行系统才能运行。

(7) 数据采集系统：这类系统用一些传感器从环境中采集数据并发送这些数据给其他系统进行处理。软件必须能同传感器进行交互且通常是安装在恶劣环境中，如安装在发动机内部或者荒郊野外。

(8) 人工智能软件：利用非数值算法解决计算和直接分析无法解决的复杂问题。这个领域的应用包括机器人、专家系统、模式识别(图像和语音)、人工神经网络、定理证明和博弈等。

每种类型的软件都有不同的特征，因此需要使用不同的软件工程技术。例如，汽车上的嵌入式控制系统是对安全性要求极高的，在车上安装时要烧制到 ROM 中。因此一旦发生改变代价就会非常昂贵。这样的一个系统应得到全面的核查和校验，以确保售后因为软件问题被召回的可能性最小。用户的交互在这里很少，因此没有必要使用依赖于用户接口原型开发过程。而基于 Web 的交互式系统涉及大量与用户交互的操作，所以更适合用迭代式开发和交付，而且系统可以包含很多可复用的组件。

1.2.5　软件工程与 Web

今天万维网的发展已经对我们的生活产生了深远的影响，基于 Web 的软件系统已经成了软件行业的主流，各种移动应用程序(APP)、基于浏览器的 Web 应用、基于云计算的数据存储、数据共享的使用已经渗透到了日常生活的每一个角落，最常见的 Web 应用有以下三种结构。

1) C/S(Client/Server，客户端/服务器)结构

以 C/S 结构为基础的系统，通常将数据库安装在服务器端，应用程序安装在客户端。常见的 C/S 结构的系统如银行 ATM 机取款系统。C/S 结构如图 1-3 所示。

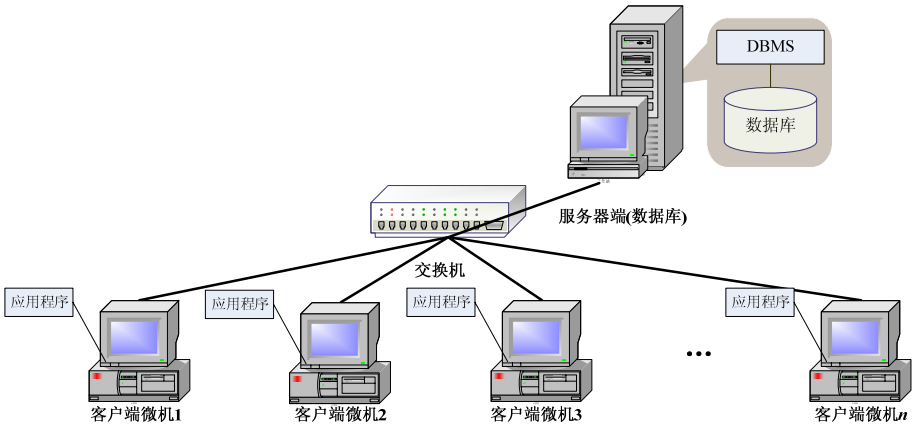

图 1-3　C/S 结构图

数据库被集中存放在服务器上，这个服务器通常称为数据库服务器，数据库集中存放的目的是方便实现数据共享。应用程序安装在客户端计算机上，只有当应用程序需要访问数据库中的数据时，才通过网络访问数据库服务器，因此 C/S 结构的网络资源比较少。客户端计算机和服务器通过交换机等其他网络设备连接在一起，所形成的计算机网络为内部局域网。

C/S 结构的优点是：系统安装在内部局域网中，其安全性比较高，受外界影响比较小；由于应用程序在客户端计算机上，系统即时性比较好，满足快速响应需求；数据库被集中存放，共享程度高。C/S 结构存在的问题是：当应用程序升级时需要对若干台计算机的应用程序同步升级操作，如果有计算机未进行升级操作，就会产生应用程序版本不同步的问题；另外，集中存放的数据库可以被多个用户同时访问，存在系统内部的安全性控制问题。

值得注意的是，当今 C/S 已不仅仅局限于局域网内，当用户在移动手机上下载一个应用程序时，手机就成为 C/S 结构的客户端，每当在应用程序上进行数据访问时，请求就会发给数据库服务器，因此，如支付宝、微信、微博、淘宝等移动端的应用程序都属于典型的 C/S 结构。

2) B/S(Browse/Server，浏览器/服务器)结构

以 B/S 结构为基础的系统，数据库和应用程序均被安装在服务器端，客户端计算机不安装应用程序，而是通过浏览器来访问服务器应用程序，再由服务器应用程序访问数据库。常见的 B/S 结构的系统如新闻浏览网站、电子商务网站等。B/S 结构如图 1-4 所示。

以浏览器方式访问应用程序，并对数据库中的数据进行操作，需要一直占用网络资源，因此访问速度与计算机网络的带宽、访问量等技术指标有关，但这种方式可以不考虑地域的限制，可以在广域网上运行，当然也可以在内部局域网上运行。在硬件配置上，需要考虑路由器等网络设备。

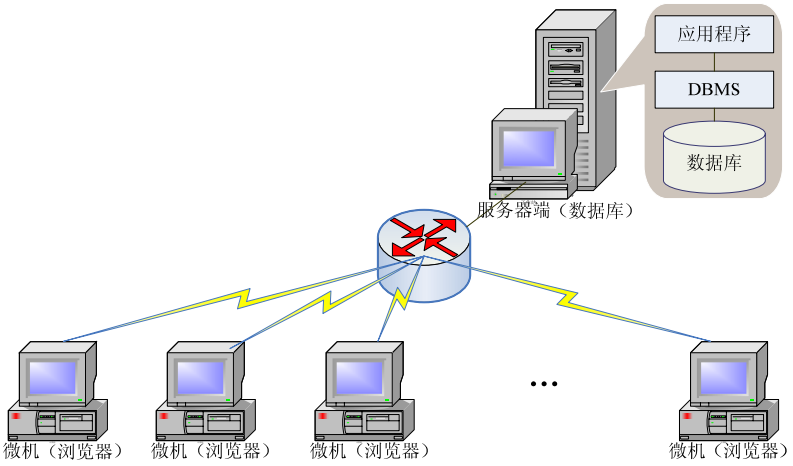

图 1-4　B/S 结构图

B/S 结构的优点是：应用程序和数据库均被安装在服务器端，当应用程序升级时，只需对服务器端的应用程序升级，客户端计算机通过刷新浏览器便可以运行最新版本的应用程序，因此不存在应用程序升级过程中版本不同的问题。B/S 结构存在的问题是：由于系统可以建立在广域网上，其安全性受到威胁，需要面对目前网络上的黑客攻击、病毒传播等问题，因此必须有良好的安全措施予以保障，如配置防火墙、安装防病毒软件等；另外，应用程序运行过程中的人机交互是靠不断刷新浏览器页面来实现的，对系统的运行效率有较大的影响，目前通过引用 Ajax 技术，可以在无须重新加载页面的情况下对网页的某部分进行更新。

3) 云计算

云计算是使计算分布在大量的分布式计算机上，而非本地计算机或远程服务器中，如图 1-5 所示，就像是从古老的单台发电机模式转向了电厂集中供电的模式。它意味着网络资源、计算能力也可以作为一种商品进行流通，就像煤气、水电一样，取用方便，费用低廉。最大的不同在于，它是通过互联网进行传输的。

图 1-5　云计算的逻辑结构

“云”具有相当的规模，Google 云计算已经拥有 100 多万台服务器，Amazon、IBM、微软、Yahoo 等的“云”均拥有几十万台服务器。企业私有云一般拥有数百上千台服务器。“云”

能赋予用户前所未有的计算能力。云计算支持用户在任意位置、使用各种终端获取应用服务，所请求的资源来自“云”，而不是固定的、有形的实体。应用在“云”中某处运行，但实际上用户无须了解，也不用担心应用运行的具体位置，只需要一台笔记本或者一个手机，就可以通过网络服务来实现需要的一切，甚至包括超级计算这样的任务。

目前云平台所能提供的服务有：①用户可以租用一台虚拟服务器来发布自己的网站；②可以为自己的计算机部署某种基础软件环境，如操作系统、Java 开发环境；③为企业客户提供办公平台及应用；④为企业或客户提供网络安全配置服务；⑤提供各种 API(Application Programming Interface，应用程序编程接口)服务；⑥提供数据分析、数据计算、数据存储服务等。当然这些服务都是需要收费才能提供的。

“云”使用了数据多副本容错、计算节点同构可互换等措施来保障服务的高可靠性，使用云计算比使用本地计算机可靠。云计算不针对特定的应用，在“云”的支撑下可以构造出千变万化的应用，同一个“云”可以同时支撑不同的应用运行。由于“云”的特殊容错措施可以采用极其廉价的节点来构成“云”，“云”的自动化集中式管理使大量企业无须负担日益高昂的数据中心管理成本，“云”的通用性使资源的利用率较之传统系统大幅提升，因此用户可以充分享受“云”的低成本优势，经常只要花费几百美元、几天时间就能完成以前需要数万美元、数月时间才能完成的任务。

云计算服务除提供计算服务外，还提供了存储服务。但是云计算服务当前垄断在私人机构(企业)手中，而他们仅仅能够提供商业信用。对于政府机构、商业机构(特别像银行这样持有敏感数据的商业机构)对于选择云计算服务应保持足够的警惕。一旦商业用户大规模使用私人机构提供的云计算服务，无论其技术优势有多强，都不可避免地让这些私人机构以“数据(信息)”的重要性来挟制整个社会。对于信息社会而言，“信息”是至关重要的。另外，云计算中的数据对于数据所有者以外的其他云计算用户是保密的，但是对于提供云计算的商业机构而言却是毫无秘密可言。所有这些潜在的危险是商业机构和政府机构选择云计算服务、特别是国外机构提供的云计算服务时，不得不考虑的一个重要的前提。

云计算的实现需要开发包含前端和后端服务的体系结构。前端包括客户设备和应用软件(如浏览器)，用于访问后端。后端包括服务器和相关的计算资源、数据存储系统(如数据库)、服务器驻留应用程序和管理服务器。通过建立对“云”及其驻留资源的一系列访问协议管理服务器，使用中间件对流量进行协调和监控。

1.2.6　软件工程的通用原则

虽然不同类型的软件采用的开发过程和开发方法不尽相同，但还是有一些软件工程的基本原则适用于所有类型的软件系统。

1. 第一原则：存在价值

一个文件系统应能为用户提供价值而具有存在价值，所有的决策都应该基于这个思想。在确定系统需求关注系统功能，以及决定硬件平台或者开发过程之前，问问自己：“这确实能为系统增加真正的价值吗？”如果答案是不，那就坚决不做。所有的其他原则都以这条原

则为基础。

2. 第二原则：有管理的软件过程

应使用有管理的和理解了的开发过程进行开发。软件开发机构应规划它们的开发过程，并清楚地知道产出什么及什么时候完工。当然，对于不同类型的软件使用不同的开发过程。

3. 第三原则：可用性和信息安全性

可用性和性能对所有类型的系统来说都很重要。软件应该如所期待的那样表现，没有失败且在用户需要时是可用的。它也应该是操作安全且信息安全的，能抵御来自外部的攻击。同时系统应是高效的，而且不会浪费资源。

4. 第四原则：需求工程

理解和管理系统需求及描述(系统应该做的是什么)是很重要的。你必须知道不同的客户和用户的期望是什么，然后管理这些期望以便在预算范围内按期交付一个有用的系统。对于任何一个软件产品，其工作产品都可能有很多用户。需求说明时时刻想到用户，设计中始终想到实现，编码时想着要维护和扩展系统的人，尽可能地使他们的工作简单化会大大提升系统的价值。

5. 第五原则：提前计划复用

应尽可能高效地使用当前存在的资源。这就意味着，应该在适当的地方复用已开发的软件，而不是重新写一个新软件。为达到面向对象(或是传统)程序设计技术所能提供的复用性，需要有前瞻性的设计和计划。提前做好复用计划将降低开发费用，并增加可复用构件及构件化系统的价值。

6. 第六原则：面向未来

生命周期持久的系统具有更高的价值。在现今的计算机环境中，需求规格说明随时会改变，硬件平台几个月后就会被淘汰，软件生命周期都是以月而不是以年来衡量的。然而，真正具有“产业实力”的软件系统必须持久耐用。为了成功地做到这一点，系统必须能适应各种变化，因此一开始就应以这种路线来设计系统。

1.2.7　软件工程人员的职业道德

用软件工程人员职业生涯中经常会面临的艰难抉择举例如下。

场景一：你觉得一个软件项目有问题，你会选择什么时机向管理层报告呢？如果只是在怀疑，这时向管理层报告未免有点敏感；如果时间拖得过长，则有可能延误了解决难题的时机。

场景二：当你的意见和团队内部的观点甚至是高层领导的决策发生冲突时，是据理力争，还是坚持原则毅然辞职，或是妥协，哪种方法是最好的呢？

场景三：公司负责开发对安全性要求极高的系统，由于时间紧张而篡改了安全的有效性验证记录，这时工程人员的职责是保守秘密，还是以一定的方式向客户披露交付的系统可能

不安全?

上述情况中潜在的灾难、灾难的严重程度及灾难的受害者这些因素都将影响决定的做出。如果情况非常危险，就应该通过一定的方式披露出来，但这时还应该同时尊重雇主的权利。

软件工程人员必须坚持诚实正直的行为准则，他们不能用掌握的知识和技能做不诚实的事情，更不能给软件工程行业抹黑。然而，在有些方面，某些行为没有法律加以规范，只能靠职业道德来约束。在这一方面职业协会和机构肩负重任。ACM、IEEE(电气和电子工程师协会)和英国计算机协会等组织颁布了职业行为准则或职业道德准则，凡是加入这些组织的成员必须严格遵守。

软件工程职业道德和职业行为准则

(ACM/IEEE-CS 联合制定以规范软件工程行业的职业道德和职业行为)

序言

准则的简写版把对软件工程人员的要求做了高度抽象的概括，完整版中的条款把这些要求细化，并给出了实例，用以规范软件工程专业人员的工作方式。没有这些总体要求，所有的细节都是教条而又枯燥的；而没有这些细节，总体要求就会变成空洞的高调。只有把两者紧密结合才能形成有机的行为准则。

软件工程人员应当做出承诺，使软件的分析、描述、设计、开发、测试和维护等工作对社会有益且受人尊重。基于对公众健康、安全和福利的考虑，软件工程人员应当遵守以下八条原则。

(1) 公众感——软件工程人员应始终与公众利益保持一致。

(2) 客户和雇主——软件工程人员应当在与公众利益保持一致的前提下，保证客户和雇主的最大利益。

(3) 产品——软件工程人员应当保证他们的产品及其相关附件达到尽可能高的行业标准。

(4) 判断力——软件工程人员应当具备公正和独立的职业判断力。

(5) 管理——软件工程管理者和领导者应当维护并倡导合乎道德的有关软件开发和维护的管理方法。

(6) 职业感——软件工程人员应当弘扬职业正义感和荣誉感，尊重社会公众利益。

(7) 同事——软件工程人员应当公平地对待和协助每一位同事。

(8) 自己——软件工程人员应当毕生学习专业知识，倡导合乎职业道德的职业活动方式。

本 章 小 结

- 软件的基本特性及一些错误观念和方法导致了软件危机的产生。
- 软件工程是涉及软件生产的各个方面的一门工程学科。
- 软件工程的目标是在合理的成本范围内，开发出满足用户需求的高质量的软件产品。

- 软件工程过程包括开发软件产品过程中的所有活动。软件过程中的活动主要有沟通、策划、建模、构建、部署、进化。
- 软件工程方法为构建软件提供技术上的解决方法，软件工程的主要方法包括结构化方法和面向对象方法。
- 软件工程工具是辅助软件开发、维护和管理的软件，软件工程的常用工具包括 Microsoft Visio、Rational Rose 等。
- 世界上存在很多不同类型的软件，每一种类型的软件开发都需要一种与之相适应的软件工程工具和技术。
- 软件工程的基本原则适用于所有的软件系统，这些基本原则包括存在价值、有管理的软件过程、可用性和信息安全性、需求工程、提前做好复用计划、面向未来。
- 软件工程人员对软件工程行业和整个社会负有责任，不应该只关心技术。
- 职业协会颁布的行为准则规定了一系列协会成员应该遵守的行为标准。

本章练习题

一、选择题

1. 软件是(　　)。
   A. 可执行程序　　B. 数据　　C. 过程文档　　D. 包含以上三种
2. 在下列选项中，(　　)不是软件的固有特性。
   A. 复杂性　　B. 抽象性　　C. 依赖性　　D. 可靠性
3. 软件是一种(　　)产品。
   A. 有形　　B. 逻辑　　C. 物质　　D. 消耗
4. (　　)不是软件危机的突出表现。
   A. 对软件开发成本和进度的估计常常很不准确
   B. 无法完成功能复杂的软件
   C. 用户对“已完成的”软件系统不满意现象经常发生
   D. 软件产品的复杂性增加，可靠性、质量却在下降
5. 产生软件危机的原因不包括(　　)。
   A. 没有合适的软件开发人员
   B. 软件人员与用户的交流存在障碍
   C. 软件开发过程不规范，缺乏方法论和规范的指导
   D. 缺少有效的软件评测手段，提交用户软件质量差
6. 衡量软件质量的因素不包括(　　)。
   A. 功能性　　B. 可靠性　　C. 可移植性　　D. 互补性

7. 与计算机科学的理论研究不同，软件工程是一门(　　)学科。

A. 理论性　　B. 工程性　　C. 原理性　　D. 心理性

8. 软件工程三要素不包括(　　)。

A. 过程　　B. 方法　　C. 工具　　D. 对象

二、简答题

1. 通用软件产品开发和定制软件开发之间有什么不同？这在实际应用中对通用软件产品用户意味着什么？

2. 什么是软件危机？产生的原因有哪些？它和软件工程有什么关系？

3. 简述软件工程的定义和软件工程的目标。

4. 软件工程过程活动主要有哪些？解释每个活动的必要性。

5. Web 的普遍使用是如何改变软件系统的？

6. 为什么软件工程的基本原则适用于所有的软件系统？

7. 对 ACM/IEEE 职业道德准则中的某项条款，举出一个恰当的例子加以说明。

8. 分别列举一两个失败或成功的软件项目实例，试说明其失败或成功的原因。

第2章　软 件 过 程

同任何事物一样，软件也有一个孕育、诞生、成长、成熟和衰亡的生存过程，我们把这个过程称为软件的生命周期。为了使软件生命周期的各项任务能够有序地按规程进行，需要一定工作模式对各项任务给予约束，这样的工作模式称为软件过程模型。本章的目标是介绍软件过程的思想——软件生产的一组互相连贯的活动。读完本章，你将了解以下内容。

- 什么是软件过程。
- 软件过程中有哪些通用的框架活动。
- 了解各种类型的软件过程模型及其优点与缺点，以及适用的开发场景。

2.1　软件过程概述

在开发产品或构建系统时，遵循一系列可预测的步骤(即路线图)是非常重要的，它有助于及时交付高质量的产品。软件开发中所遵循的路线图就称为“软件过程”。软件工程师及其管理人员应根据需要调整开发过程，并遵循该过程。除此之外，软件的需求方也需要参与过程的定义、建立和测试。

软件过程提高了软件工程活动的稳定性、可控性和有组织性，如果不进行控制，软件活动将变得混乱。但是现代软件工程方法必须是灵活的，也就是要求软件工程活动、控制及工作产品适合于项目团队和将要开发的产品。具体来讲，采用的过程依赖于构造软件的特点。例如，航空系统的软件与网站的建设可能需要采用两种截然不同的软件过程。

虽然有许多不同的软件过程，但是以下 4 种基本活动对软件工程来说是必需的，它们也是软件生命周期中最主要的活动。

(1) 软件描述。必须定义软件的功能及软件操作上的约束。

(2) 软件设计和实现。必须生产符合描述的软件。

(3) 软件有效性验证。软件必须得到有效性验证，即确保软件是客户所想要的。

(4) 软件进化。软件必须进化以满足不断变化的客户需要。

2.1.1　软件描述

软件描述主要是解决目标系统要“做什么”的问题，如果不知道问题是什么就试图解决这个问题，只会白白浪费时间和金钱，最终得出的结果很可能是毫无意义的。软件描述阶段是软件项目的早期阶段，主要是由软件分析人员和用户合作，针对有待开发的软件系统进行分析规划和规格描述，确定软件做什么，为今后的软件开发做好准备。