21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

实用软件工程

Practical Software Engineering

张海藩 吕云翔 编著

- 传统与创新共存，集成前沿知识
- 理论实践并重，提高实践能力
- 选取典型应用，案例贯穿始终

名家系列

人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

实用软件工程

Practical Software Engineering

张海藩 吕云翔 编著

名家系列

人 民 邮 电 出 版 社

北 京

图书在版编目（CIP）数据

实用软件工程 / 张海藩，吕云翔编著. -- 北京 : 人民邮电出版社，2015.5
21世纪高等学校计算机规划教材. 名家系列
ISBN 978-7-115-37404-2

Ⅰ. ①实… Ⅱ. ①张… ②吕… 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(2014)第267540号

内 容 提 要

本书按照典型的软件开发过程来组织内容，旨在培养学生具备软件工程思想及实际软件开发的能力。全书共 9 章，主要内容包括软件工程概述、可行性研究及需求分析、软件设计、编码及实现、软件测试与维护、面向对象方法学与 UML、面向对象软件设计与实现、软件工程管理、课程设计。软件工程中涉及的管理方面的内容，如软件规模估算、进度计划、人员组织、软件开发风险管理等内容。

本书可以作为普通高校计算机相关专业“软件工程”课程的教材，也可以供学习软件工程的读者（包括参加计算机等级考试或相关专业自学考试）参考。

◆ 编　　著　张海藩　吕云翔
责任编辑　武恩玉
责任印制　沈　蓉　彭志环
◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
固安县铭成印刷有限公司印刷
◆ 开本：787×1092　1/16
印张：21.25　　2015 年 5 月第 1 版
字数：561 千字　　2015 年 5 月河北第 1 次印刷

定价：49.00 元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315

前 言

软件工程是应用计算机科学技术、数学、管理学的原理，运用工程科学的理论、方法和技术，研究和指导软件开发和演化的一门交叉学科。随着科技的发展，软件工程已成为计算机科学及其相关专业的一门重要的必修课。其教学目的在于使学生掌握软件工程的基本概念和原则，培养学生使用工程化的方法高效地开发高质量软件的能力，以及进行项目管理的能力。

软件工程是一门理论与实践并重的课程。本书在讲述软件工程的基本概念、原理和方法的基础上，详细而全面地介绍了可以实际用于软件开发实践的各种技能，旨在使学生通过有限课时的学习后，不仅能对软件工程的原理有所认识，而且能具备实际开发软件的各种技能，如熟练使用各种软件工程工具、按照标准和规范编写文档等。

本书共分 9 章，内容涉及软件工程的基本原理和概念、软件开发生命周期的各个阶段、软件工程管理的相关内容、如何使用各种自动化工具来辅助软件开发的过程，以及课程设计。每章分为理论部分和实践部分。理论部分从理论学习的角度，阐述软件工程的基本概念、原理和方法。在内容的安排上详略得当，使读者在有限的时间内能领会软件工程的精髓。实践部分配合理论部分的学习内容，指导读者利用相关的工具对所学内容进行运用。实践与理论的紧密结合，不仅有利于巩固和掌握知识，还能提高读者的实践能力。此外，本书使用了一个案例（“小型二手货交易平台”）贯穿于各章的实践部分，使读者能通过实例对软件开发过程有一个系统的了解。

本书的理论教学安排建议如下。

章节	内容	学时数
第 1 章	软件工程概述	2～4
第 2 章	可行性研究及需求分析	4～6
第 3 章	软件设计	4～6
第 4 章	编码及实现	2
第 5 章	软件测试与维护	6～8
第 6 章	面向对象方法学与 UML	4～6
第 7 章	面向对象软件设计与实现	6～8
第 8 章	软件工程管理	2～6
第 9 章	课程设计	2

建议先修课程：计算机导论、面向对象程序设计、数据结构、数据库原理等。

建议理论教学时数：32～48 学时。

建议实践教学时数：16～32 学时。

教师可以按照自己对软件工程的理解适当地删除一些章节；也可以根据教学目标，灵活地调整章节的顺序，增减各章的学时数。

另外，与本书配套的各章节的部分习题的参考答案，以及教学 PPT 可在人民邮电出版社教学服务与资源网（www.ptpedu.com.cn）上的本书网页中免费注册下载。

本书作者一直在北京信息科技大学和北京航空航天大学软件学院担任软件工程课程的教

学工作，积累了软件工程教学与实践的丰富经验。在本书编写的过程中，我们得到了张星宇、温鑫、朱锋、王浣洲、李智勇、宋任飞和金晶的支持，在此对他们表示感谢，同时也感谢其他对本书有贡献的同仁们。

由于软件工程是一门新兴学科，软件工程的教学方法本身还在探索之中，加之我们的水平和能力有限，本书难免有疏漏之处。恳请广大读者给予批评指正，也希望各位能将实践过程中的经验和心得与我们交流（yunxianglu@hotmail.com）。

编　者

2014 年 10 月

目　录

第1章 软件工程概述

1.1 软　　件

1.1.1 软件的概念及特点

软件是计算机系统中不可或缺的一部分，它与硬件合为一体，从而完成特定的系统功能。人们对软件的认识也是在不断发展的。在计算机发展的初期，计算机的功能主要是由计算机的各个硬件部件通过有机地协调工作来完成的。当时所谓的软件就是程序，它的作用并没有得到人们足够的重视。

随着计算机技术的发展，人们越来越充分地认识到高质量的软件会使计算机系统的功能和效率大大地提高。于是，程序在计算机系统中的作用也日益重要。人们通常把各种不同功能的程序，包括系统程序、应用程序、用户自己编写的程序等称为软件。然而，当计算机的应用日益普及，软件日益复杂，规模日益增大，人们意识到软件并不仅仅等于程序。

程序是人们为了完成特定的功能而编制的一组指令集，它由计算机的语言描述，并且能在计算机系统上执行。而软件不仅包括程序，还包括程序的处理对象——数据，以及与程序开发、维护和使用有关的图文资料，即文档。

计算机系统由软件和硬件组成。当制造硬件时，人的创造性过程最终被转换成有形的形式。任何事物都有自己的特点，这是区别于其他事物的根本。理解事物的特点有利于人们更加深刻、更加准确地认识事物的本质。作为计算机系统的重要组成部分，计算机软件功能的发挥依赖于计算机硬件的支持，它与硬件相比，具有以下一些特点。

- 软件是一种逻辑实体，具有抽象性。硬件是有形的设备，软件不像硬件那样具有明显的可见性。人们可以把软件记录在介质上，但是却无法直观地观察到它的形态，而必须通过在计算机上实际地运行才能了解它的功能、性能及其他特性。
- 软件的生产与硬件的制造不同。它更多地渗透了人类的智能活动，是人类智力劳动的产物。软件是被开发或设计的，不是传统意义上被制造的。软件成本集中于开发上，这意味着软件项目不能像制造项目那样管理。
- 软件在运行使用过程中，不会磨损。在软件的运行和使用期间，它不会产生像硬件那样的磨损和老化现象，然而却存在着缺陷维护和技术更新的问题。软件不会磨损，但是它会退化，而软件的退化是由于修改。因此，软件维护比硬件维护要复杂得多。图 1-1 和图 1-2 分别展示了硬

件的失效率和使用时间的关系以及软件的失效率和时间的关系。

- 软件的开发至今尚未完全摆脱手工艺的开发方式。在硬件世界，构件复用是工程过程的自然的一部分，而在软件世界，它是刚刚开始起步的事物。虽然软件产业正在向基于构件的组装前进，但大多数软件仍是定制的。
- 软件的开发和运行必须依附于特定的计算机系统环境。它不像有些设备一样，能够独立地工作，而是受到了物理硬件、网络配置、支撑软件等因素的制约。由此引发了软件的可移植性问题。

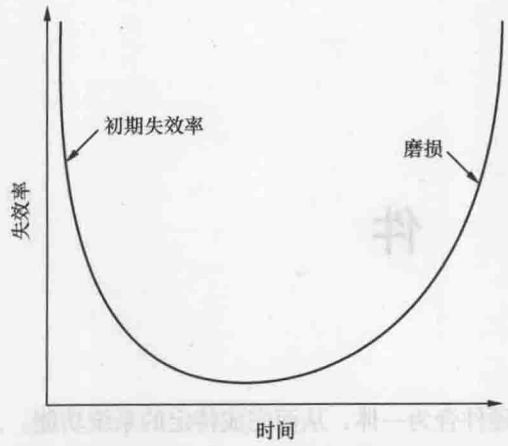

图 1-1　硬件失效曲线图

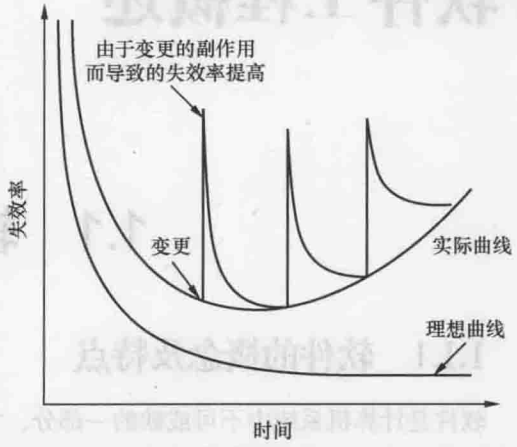

图 1-2　软件失效曲线图

1.1.2　软件的分类

随着计算机软件复杂性的增加，在某种程度上人们很难对软件给出一个通用的分类，但是人们可以按照不同的角度对软件进行分类。按照功能的不同，软件可以分为系统软件、支撑软件和应用软件三类。系统软件是居于计算机系统中最靠近硬件的一层，为其他程序提供最底层系统服务，它与具体的应用领域无关，如编译程序和操作系统等。支撑软件以系统软件为基础，以提高系统性能为主要目标，支撑应用软件的开发与运行，主要包括环境数据库、各种接口软件和工具组。应用软件是提供特定应用服务的软件，如字处理程序等。系统软件、支撑软件和应用软件之间既有分工又有合作，是不可以截然分开的。

基于规模的不同，软件可以划分为微型、小型、中型、大型和超大型软件。一般情况下，微型软件只需要一名开发人员，在 4 周以内完成开发，并且代码量不超过 500 行。这类软件一般仅供个人专用，没有严格的分析、设计和测试资料。小型软件开发周期可以持续到半年，代码量一般控制在 5 000 行以内。这类软件通常没有预留与其他软件的接口，但是需要遵循一定的标准，附有正规的文档资料。中型软件的开发人员控制在 10 人以内，要求在 2 年以内开发 5 000 到 50 000 行代码。这种软件的开发不仅需要完整的计划、文档及审查，还需要开发人员之间，开发人员和用户之间的交流与合作。大型软件是 10 到 100 名开发人员在 1 到 3 年的时间内开发的，具有 50 000 到 100 000 行代码的软件产品。在这种规模的软件开发中，统一的标准、严格的审查制度及有效的项目管理都是必须的。超大型软件往往涉及上百名甚至上千名成员以上的开发团队，开发周期可以持续到 3 年以上，甚至 5 年。这种大规模的软件项目通常被划分为若干个小的子项目，由不同的团队开发。

根据软件服务对象的不同，软件还可以分为通用软件和定制软件。通用软件是由特定的软件

开发机构开发，面向市场公开销售的独立运行的软件系统，如操作系统、文档处理系统和图片处理系统等。定制软件通常是面向特定的用户需求，由软件开发机构在合同的约束下开发的软件，如为企业定制的办公系统、交通管理系统和飞机导航系统等。

按照工作方式的不同，计算机软件还可以划分为实时软件、分时软件、交互式软件和批处理软件。

软件的分类示意图如图 1-3 所示。

图 1-3　软件的分类

1.2 软 件 危 机

1.2.1 软件危机的表现与原因

软件危机就是指人们在开发软件和维护软件过程中所遇到的一系列的问题。在 20 世纪 60 年代中期，随着软件规模的扩大，复杂性的增加，功能的增强，使得高质量的软件开发变得越来越困难。在软件开发的过程中，会经常出现不能按时完成任务、产品质量得不到保证、工作效率低下和开发经费严重超支等现象。这些情况逐渐使人们意识到软件危机的存在及其重要性。计算机软件的开发、维护和应用过程中普遍出现的这些严重的问题，主要表现如下。

- 开发出来的软件产品不能满足用户的需求，即产品的功能或特性与需求不符。这主要是由于开发人员与用户之间不能充分、有效地交流造成的，使得开发人员对用户需求的理解存在着差异。
- 相比越来越廉价的硬件，软件代价过高。
- 软件质量难以得到保证，且难以发挥硬件潜能。开发团队缺少完善的软件质量评审体系以及科学的软件测试规程，使得最终的软件产品存在着诸多缺陷。
- 难以准确估计软件开发、维护的费用以及开发周期。软件产品往往不能在预算范围之内、按照计划完成开发。很多情况下，软件产品的开发周期或经费会大大超出预算。
- 难于控制开发风险，开发速度赶不上市场变化。
- 软件产品维护困难，集成遗留系统更困难。
- 软件文档不完备，并且存在着文档内容与软件产品不符的情况。软件文档是计算机软件的重要组成部分，它为在软件开发人员之间以及开发人员与用户之间信息的共享提供了重要的平台。软件文档的不完整和不一致的问题会给软件的开发和维护等工作带来很多麻烦。

这些问题严重影响了软件产业的发展，制约着计算机的应用。为了形象地描述软件危机，

OS/360 经常被作为一个典型的案例。这是一个超大型的软件项目，使用了 1 000 个左右的程序员。在经历了数十年的开发之后，极度复杂的软件项目甚至产生了一套不包括在原始设计方案之中的工作系统。Fred Brooks 是这个项目的管理者，他在自己的著作《人月神话》中曾经承认，自己犯了一个价值数百万美元的错误。

软件危机的出现和日益严重的趋势充分暴露了软件产业在早期的发展过程中存在的各种各样的问题。可以说，人们对软件产品认识的不足以及对软件开发的内在规律理解的偏差是软件危机出现的本质原因。具体来说，软件危机出现的原因可以概括为以下几点。

- 忽视软件开发前期的需求分析。
- 开发过程缺乏统一的、规范化的方法论的指导。软件开发是一项复杂的工程，人们需要用科学的工程化的思想来组织和指导软件开发的各个阶段。而这种工程学的视角正是很多软件开发人员所没有的，他们往往简单地认为软件开发就是程序设计。
- 文档资料不齐全或不准确。软件文档的重要性没有得到软件开发人员和用户的足够重视。软件文档是软件开发团队成员之间交流和沟通的重要平台，还是软件开发项目管理的重要工具。如果人们不能充分重视软件文档的价值，势必会给软件开发带来很多不便。
- 忽视与用户之间、开发组成员之间的交流。
- 忽视测试的重要性。
- 不重视维护或由于上述原因造成维护工作的困难。由于软件的抽象性和复杂性使得软件在运行之前，对开发过程的进展情况很难估计。再加上软件错误的隐蔽性和改正的复杂性，这些都使得软件开发和维护在客观上比较困难。
- 从事软件开发的专业人员对这个产业认识不充分，缺乏经验。软件产业相对于其他工业产业而言，是一个比较年轻、发展不成熟的产业，人们在对它的认识上缺乏深刻性。
- 没有完善的质量保证体系。完善的质量保证体系的建立需要有严格的评审制度，同时还需要有科学的软件测试技术及质量维护技术。软件的质量得不到保证，使得开发出来的软件产品往往不能满足人们的需求，同时人们还可能需要花费大量的时间、资金和精力去修复软件的缺陷，从而导致了软件质量的下降和开发预算超支等后果。

1.2.2 软件危机的启示

软件危机给我们的最大启示，是使我们更加深刻地认识到软件的特性以及软件产品开发的内在规律。

- 软件产品是复杂的人造系统，具有复杂性、不可见性和易变性，难以处理。
- 个人或小组在开发小型软件时使用到的非常有效的编程技术和过程，在开发大型、复杂系统时难以发挥同样的作用。
- 从本质上讲，软件开发的创造性成分很大、发挥的余地也很大，很接近于艺术。它介于艺术与工程之间的某一点，并逐步向工程一段漂移，但很难发展到完全的工程。
- 计算机和软件技术的快速发展，提高了用户对软件的期望，促进了软件产品的演化，为软件产品提出了新的、更多的需求，难以在可接受的开发进度内保证软件的质量。
- 几乎所有的软件项目都是新的，而且是不断变化的。项目需求在开发过程中会发生变化，而且很多原来预想不到的问题会出现，对设计和实现手段进行适当的调整是不可避免的。
- “人月神化”现象——生产力与人数并不成正比。

为了解决软件危机，人们开始尝试着用工程化的思想去指导软件开发，于是软件工程诞生了。

1.3 软 件 工 程

1.3.1 软件工程的概念

1968 年，在北大西洋公约组织举行的一次学术会议上，人们首次提出了软件工程这个概念。当时，该组织的科学委员们在开会讨论软件的可靠性与软件危机的问题时，提出了“软件工程”的概念，并将其定义为“为了经济地获得可靠的和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规则”。这个定义肯定了工程化的思想在软件工程中的重要性，但是并没有提到软件产品的特殊性。

经过 40 多年的发展，软件工程已经成为一门独立的学科，人们对软件工程也逐渐有了更全面、更科学的认识。

IEEE 对软件工程的定义为：（1）将系统化、严格约束的、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件。（2）对（1）中所述方法的研究。

具体说来，软件工程是以借鉴传统工程的原则、方法，以提高质量、降低成本为目的指导计算机软件开发和维护的工程学科。它是一种层次化的技术，如图 1-4 所示。

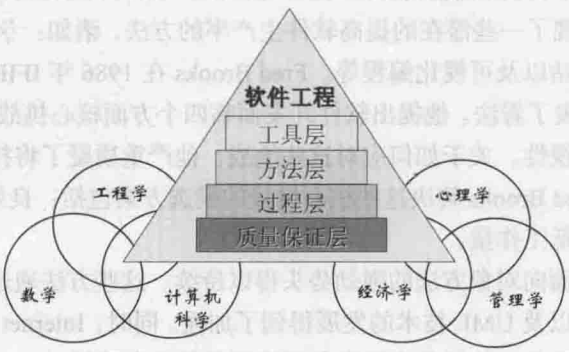

图 1-4　软件工程层次图

软件工程的根基就在于对质量的关注；软件工程的基础是过程层，它定义了一组关键过程区域的框架，使得软件能够被合理和及时的开发；软件工程的方法提供了建造软件在技术上需要“做什么”，它覆盖了一系列的任务，包括需求分析、设计、编程、测试和支持等；软件工程的工具对过程和方法提供了自动的或半自动的支持。软件工程本身是一个交叉学科，涉及多种学科领域的相关知识，包括工程学、数学、计算机科学、经济学、管理学、心理学等。

软件工程以关注质量为目标，其中过程、方法和工具是软件工程的三要素。

1.3.2 软件工程的发展

随着软件项目的规模和难度逐渐增大，以个人能力为基础的软件开发所具有的弊端逐渐体现，随之出现了著名的“软件危机”。在这种情况之下，NATO 科学委员会在 1968 和 1969 年召开了两次里程碑似的的“软件工程会议”，许多顶尖级的研究员和工程师参加了这次会议，真正意义上的“软件工程”就此诞生。

20 世纪 70 年代，人们开始采用与 60 年代的“编码和组装”相反的过程，先做系统需求分析，然后设计，最后再编码，并把 50 年代硬件工程技术最好的方面和改进的软件方向的技术加以总结，提出了“瀑布模型”。需要指出的是，瀑布模型本身在提出时，是一个支持迭代和反复的模型。然而为了更方便地对软件进行约束，瀑布模型总是被解释为一种纯顺序化的模型，另外对瀑布模型的固定过程标准的解释也加深了这种误解。

另一方面，C. Bohm 和 G. Jacoponi 提出了“go to 语句是有害的”论点，并提出所有程序都可以转换为三种逻辑（顺序、分支、循环）来实现，奠定了结构化编程的基础。随后，很多种结构化软件开发方法被提出，极大地改善了软件质量，提高了软件开发效率。数据结构和算法理论迅速发展，取得了很多重要成就；形式方法和程序证明技术也成为人们关注的发展焦点。

然而随着形式化模型和连续化的瀑布模型所带来的问题大幅度增加，对于一个缺乏经验的团体来说，用形式化的方法，要使软件在可靠性和有用性上达到要求十分困难。瀑布模型在文档编写时消耗很大，而且速度慢，使用起来代价大。

伴随 20 世纪 70 年代开发的一些“最佳实践”，80 年代开始了一系列工作以处理 70 年代遗留的问题，并且开始改进软件工程的生产效率和可测量性。COCOMO 模型、CMM 模型等被提出，软件体系结构相关研究和技术日益成熟，随后关系数据库被提出。

在软件工具方面，除了 20 世纪 70 年代已经出现的软件需求和设计工具，其他领域一些重要的工具也得到了改进，例如，测试工具和配置管理工具。工具集和集成开发支持环境先后出现，最终人们将范围扩展到了计算机辅助软件工程（CASE），软件开发的效率进一步得到提高。

在其他方面，也出现了一些潜在的提高软件生产率的方法，诸如：专家系统、高级程序语言、面向对象、强大的工作站以及可视化编程等。Fred Brooks 在 1986 年 IFIP 发表的著名论文《没有银弹》中对上述内容发表了看法。他提出软件开发面临四个方面核心挑战：高等级的软件复杂度、一致性、可变性和不可视性。关于如何应对这些挑战，他严重质疑了将技术说成是软件解决方案的“银弹”的观点。Fred Brooks 解决这些核心挑战的候选方案包括：良好的设计者、快速原型、演化开发和通过复用降低工作量。

20 世纪 90 年代，面向对象方法的强劲势头得以持续。这些方法通过设计模式、软件体系结构和体系结构描述语言以及 UML 技术的发展得到了加强。同时，Internet 的继续扩展和 WWW 的出现同时增强了面向对象的方法以及市场竞争环境下软件的危险性。

软件作为竞争鉴别器，其重要性的增强以及缩减软件推向市场时间的需要，引发了从顺序的瀑布模型向其他模型的转变潮流，这类模型强调并行的工程性的需求、设计、编码、产品和过程以及软件和系统。软件复用成为软件开发中重要的内容，开源文化露出头角，可用性以及人机交互也成为软件开发中的重要指标。

20 世纪 90 年代末，出现了许多的敏捷方法，例如，自适应软件开发、水晶项目开发、动态系统开发、极限编程、特征驱动开发、Scrum 等。这些主要的敏捷方法的创始人在 2001 年聚集一堂并发表了《敏捷软件开发宣言》。

在 21 世纪，对快速应用开发追求的趋势仍在继续，在信息技术、组织、竞争对策以及环境等方面的变革步伐也正在加快。这种快速的变革步伐引发了软件开发领域越来越多的困难和挫折，更多的软件开发过程、方法和工具也相继出现，软件工程在持续的机遇与挑战中不断发展。“大规模计算”、“自治和生化计算机”、“模型驱动体系结构”、“构件化软件开发”等新领域都可能成为接下来软件工程发展的主要方向。

1.3.3　软件工程研究的内容

软件工程是一门新兴的边缘学科，涉及的学科多，研究的范围广。归结起来软件工程研究的主要内容有以下 4 个方面：方法与技术、工具与环境、管理技术、标准与规范。

- 软件开发与技术，主要讨论软件开发的各种方法及其工作模型，它包括多方面的任务，如软件系统需求分析、总体设计，以及如何构建良好的软件结构、数据结构及算法设计等，同时讨论具体实现的技术。
- 软件工具与环境为软件工程方法提供支持，研究计算机辅助软件工程，建立软件工程环境。
- 软件工程管理技术，是指对软件工程全过程的控制和管理，包括计划安排、成本估算、项目管理、软件质量管理。
- 软件工程标准与规范，使得各项工作有章可循，以保证软件生产效率和软件质量的提高。软件工程标准可分为 4 个层次：国际标准、行业标准、企业规范和项目规范。

必须要强调的是，随着人们对软件系统研究的逐渐深入，软件工程所研究的内容也在不断更新和发展。

1.3.4　软件工程目标和原则

软件工程要达到的基本目标如下。

- 达到要求的软件功能。
- 取得较好的软件性能。
- 开发出高质量的软件。
- 付出较低的开发成本。
- 需要较低的维护费用。
- 能按时完成开发工作，及时交付使用。

为了达到上述目标，软件工程设计、工程支持以及工程管理在软件开发过程中必须遵循一些基本原则。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，提出了软件工程的 7 条基本原则。

1. 用分阶段的生没周期计划进行严格的管理

这条原则指将软件的生命周期划分为多个阶段，对各个阶段实行严格的项目管理。软件开发是一个漫长的过程，人们可以根据工作的特点或目标，把整个软件的开发周期划分为多个阶段，并为每个阶段制定分阶段的计划及验收标准，这样有益于对整个软件的开发过程进行管理。在传统的软件工程中，软件开发的生命周期可以划分为可行性研究、需求分析、软件设计、软件实现、软件测试、产品验收和交付等阶段。

2. 坚持进行阶段评审

严格的贯彻与实施阶段评审制度可以帮助软件开发人员及时地发现错误并将其改正。在软件开发的过程中，错误发现的越晚，修复错误所要付出的代价就会越大。实施阶段评审，就是指只有在本阶段的工作通过评审后，才能进入下一阶段的工作。

3. 实行严格的产品控制

在软件开发的过程中，用户需求很可能在不断地发生变化。有些时候，即使用户需求没有改变，软件开发人员受到经验的限制以及与客户交流不充分的影响，也很难做到一次性获得全部的正确的需求。可见，需求分析的工作应该贯穿到整个软件开发的生命周期内。在软件开发的整个

过程中，需求的改变是不可避免的。当需求更新时，为了保证软件各个配置项的一致性，实施严格的版本控制是非常必要的。

4. 采用现代程序设计技术

现代的程序设计技术，例如，面向对象，可以使开发出来的软件产品更易维护和修改，同时还能缩短开发的时间，并且更符合人们的思维逻辑。

5. 软件工程结果应能被清楚地审查

虽然软件产品的可见性比较差，但是它的功能和质量应该能够被清楚地审查和度量，这样才能有利于有效的项目管理。一般软件产品包括可以执行的源代码、一系列相应的文档和资源数据等。

6. 开发小组的人员应该少而精

开发小组成员的人数少有利于组内成员充分的交流，这是高效团队管理的重要因素。而高素质的开发小组成员是影响软件产品的质量和开发效率的重要因素。

7. 承认不断改进软件工程实践的必要性

随着计算机科学技术的发展，软件从业人员应该不断地总结经验并且主动学习新的软件技术，只有这样才能不落后于时代。

B.Boehm 指出，遵循前 6 条基本原则，能够实现软件的工程化生产；按照第 7 条原则，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。

1.3.5 软件工程知识体系

IEEE 在 2014 年发布的《软件工程知识体系指南》中将软件工程知识体系划分为以下 15 个知识领域。

1. 软件需求（software requirements）

软件需求涉及软件需求的获取、分析、规格说明和确认。

2. 软件设计（software design）

软件设计定义了一个系统或组件的体系结构、组件、接口和其他特征的过程以及这个过程的结果。

3. 软件构建（software construction）

软件构建是指通过编码、验证、单元测试、集成测试和调试的组合，详细地创建可工作的和有意义的软件。

4. 软件测试（software testing）

软件测试是为评价和改进产品的质量、标识产品的缺陷和问题而进行的活动。

5. 软件维护（software maintenance）

软件维护是指由于一个问题或改进的需要而修改代码和相关文档，进而修正现有的软件产品并保留其完整性的过程。

6. 软件配置管理（software configuration management）

软件配置管理是一个支持性的软件生命周期过程，它是为了系统地控制配置变更，在软件系统的整个生命周期中维持配置的完整性和可追踪性，而标识系统在不同时间点上的配置的学科。

7. 软件工程管理（software engineering management）

软件工程的管理活动建立在组织和内部基础结构管理、项目管理、度量程序的计划制定和控制三个层次上。