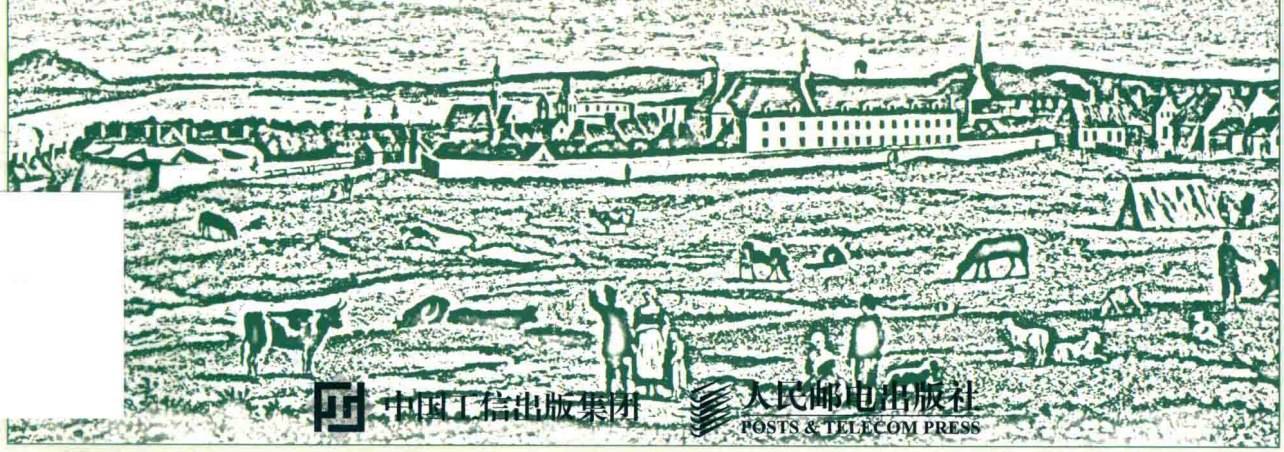
普通高等教育
软件工程 “十二五”规划教材

12th Five-Year Plan Textbooks
of Software Engineering

软件工程
——理论与实践
（附微课视频 第2版）

吕云翔 ◎ 编著

Theory and Practice of Software Engineering



中国工信出版集团 人民邮电出版社
POSTS & TELECOM PRESS

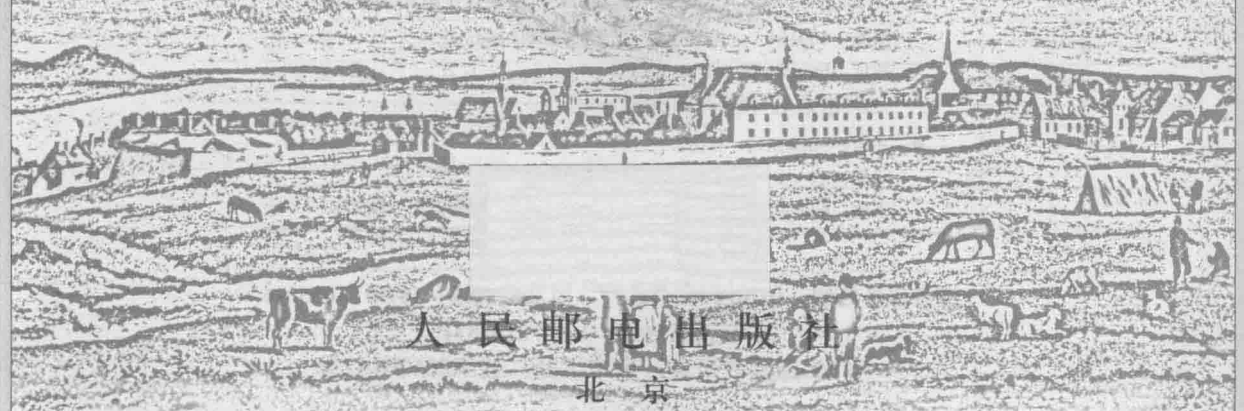
普通高等教育
软件工程“十二五”规划教材

12th Five-Year Plan Textbooks
of Software Engineering

软件工程
——理论与实践
（附微课视频 第2版）

吕云翔 ◎ 编著

Theory and Practice of
Software Engineering



人民邮电出版社
北京

图书在版编目（CIP）数据

软件工程 : 理论与实践 / 吕云翔编著. -- 2版. -- 北京 : 人民邮电出版社, 2018.5
普通高等教育软件工程“十二五”规划教材 : 附微课视频
ISBN 978-7-115-48019-4

Ⅰ. ①软… Ⅱ. ①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(2018)第042379号

内 容 提 要

本书按照典型的软件开发过程来组织内容，旨在培养读者具备软件工程思想及实际软件开发的能力。全书共 12 章，内容涉及软件与软件工程、软件过程、可行性研究与项目开发计划、结构化分析、结构化设计、面向对象方法与 UML、面向对象分析、软件体系结构与设计模式、面向对象设计、软件实现、软件测试、软件维护与软件工程管理。本书理论与实践相结合，内容翔实，可操作性强。

本书可作为高等院校计算机科学、软件工程及相关专业“软件工程”课程的教材。

◆ 编　　著　吕云翔
责任编辑　刘　尉
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
三河市君旺印务有限公司印刷

◆ 开本：787×1092　1/16
印张：19　　2018 年 5 月第 2 版
字数：472 千字　　2018 年 5 月河北第 1 次印刷

定价：59.80 元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字20170147号

前　言

《软件工程——理论与实践》自 2012 年 8 月出版以来，经过了几次印刷。许多高校将其作为“软件工程”课程的教材，深受这些学校师生的喜爱，获得了良好的社会效益。但从另外一个角度来看，作者有责任和义务维护好这本书的质量，及时更新本书的内容，做到与时俱进。

本书对原教材进行了全面的修订、再组织和更新。本书改动内容如下。

（1）将书的组织结构分为 6 个部分：软件工程概述、可行性研究与项目开发计划、结构化分析与设计、面向对象分析与设计、软件实现与测试、软件维护与软件工程管理。

（2）丰富了“结构化分析”与“结构化设计”的内容。

（3）增加了“面向对象分析”与“面向对象设计”的许多内容。

（4）丰富了“软件测试”的内容。

（5）将案例“网上书店系统”改为了“小型网上书店系统”，并用最新的开发工具进行了重新开发。读者可采用扫描二维码的形式，获取案例的电子文档和源代码。

（6）去掉了对一些工具（如 Visio、Rose、Visual Studio、Project）的具体介绍。

本书共 12 章，内容涉及软件与软件工程、软件过程、可行性研究与项目开发计划、结构化分析、结构化设计、面向对象方法与 UML、面向对象分析、软件体系结构与设计模式、面向对象设计、软件实现、软件测试、软件维护与软件工程管理。并且介绍了如何使用各种自动化工具（以实验的形式）来辅助软件开发的过程，以及课程设计的指导和开发文档的撰写。

本书的理论知识的教学安排建议如下。

章节	内　容	学时数
第 1 章	软件与软件工程	2～4
第 2 章	软件过程	2
第 3 章	可行性研究与项目开发计划	2
第 4 章	结构化分析	2～4
第 5 章	结构化设计	2～4
第 6 章	面向对象方法与 UML	4～6
第 7 章	面向对象分析	4～6
第 8 章	软件体系结构与设计模式	2
第 9 章	面向对象设计	4～6
第 10 章	软件实现	2
第 11 章	软件测试	4～6
第 12 章	软件维护与软件工程管理	2～4

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：32～48 学时。

建议实验（实践）教学时：16～32 学时。

教师可以按照自己对软件工程的理解和教学需要，灵活调整教学内容、章节顺序，增减各章的学时数。

本书的编者为吕云翔，曾洪立、吕彼佳、姜彦华进行了素材整理及配套资源制作等。

本书配套的视频二维码位置如下。

序　号	视频内容标题	视频二维码位置	所在页码
1	模块设计启发规则	5.1.2　软件设计的原则	69
2	模块分割方法	5.1.2　软件设计的原则	70
3	结构化软件设计的任务	5.4　结构化软件设计概述	78
4	用 C++理解类和对象	6.1.1　面向对象的基本概念	103
5	用 C++理解继承和组合	6.1.1　面向对象的基本概念	104
6	用 C++理解虚函数和多态	6.1.1　面向对象的基本概念	104
7	使用 UML 的规则	6.2.2　UML 的特点	106
8	在统一软件开发过程中使用 UML	6.2.3　UML 的应用范围	107
9	用例的特征	6.3.1　用例图	109
10	绘制机票预订系统的用例图	6.3.1　用例图	109
11	绘制机票预订系统的类图	6.3.2　类图和对象图	111
12	绘制机票预订系统的顺序图	6.4.1　顺序图	116
13	绘制机票预订系统的协作图	6.4.2　协作图	117
14	绘制机票预订系统的状态图	6.4.3　状态图	118
15	绘制机票预订系统的活动图	6.4.4　活动图	118
16	绘制机票预订系统的构件图	6.5.1　构件图	120
17	绘制机票预订系统的部署图	6.5.2　部署图	120
18	面向对象的实现——提高可重用性	10.3　面向对象实现	207
19	面向对象的实现——提高可扩充性	10.3　面向对象实现	207
20	面向对象的实现——提高健壮性	10.3　面向对象实现	207
21	手工测试与自动测试	11.2　软件测试的分类	219
22	黑盒测试的优缺点	11.5.7　黑盒测试方法选择	231
23	白盒测试的优缺点	11.6.6　白盒测试方法选择	236
24	导致维护困难的一些因素	12.1　软件维护	261
25	维护工作的过程	12.1.1　软件维护的过程	261
26	维护的代价及其主要因素	12.1.3　软件的可维护性	264

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。恳请各位同仁和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

2018 年 1 月

目 录

第5部分　软件实现与测试

第6部分　软件维护与软件工程管理

第 1 部分　软件工程概述

第 1 章　软件与软件工程

1.1　软　　件

1.1.1　软件的概念及特点

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（文档）。例如，用户购买的 Windows 10 操作系统这个软件，它不仅包含可执行的程序，还有一些支持的数据（都放在光盘中），以及纸质的用户手册等文档。Roger S. Pressman 对软件给出了这样的定义：计算机软件是由专业人员开发并长期维护的软件产品。完整的软件产品包括在各种不同容量和体系结构计算机上的可执行程序、运行过程中产生的各种结果，以及以硬复制和电子表格等多种方式存在的软件文档。

软件具有以下几个特点。

（1）软件是一种逻辑实体，而不是具体的物理实体，因而它具有抽象性。

（2）软件的生产与硬件不同，它没有明显的制造过程。要提高软件的质量，必须在软件开发方面下功夫。

（3）在软件的运行和使用期间，不会出现硬件中出现的机械磨损、老化问题，然而它存在退化问题，必须对其进行多次修改与维护，直至其退役。如早期的 DOS 操作系统，就是进行了多次修改与维护，实在难以与 Windows 操作系统匹敌而退役了。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。

（4）计算机的开发与运行常常受到计算机系统的制约，它对计算机系统有着不同程度的依赖性。如有专门针对 PC 的游戏，也有针对苹果计算机的游戏。为了解除这种依赖性，在软件开发中提出了软件移植的问题。

（5）软件开发至今尚未完全摆脱人工的开发方式。

（6）软件本身是复杂的。软件的复杂性可能来自它反映的实际问题的复杂性，也可能来自程序逻辑结构的复杂性。

（7）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，它的成本是比较高的。

（8）相当多的软件工作涉及社会因素。许多软件的开发和运行涉及机构、体制及管理方式等问题，它们直接决定项目的成败。

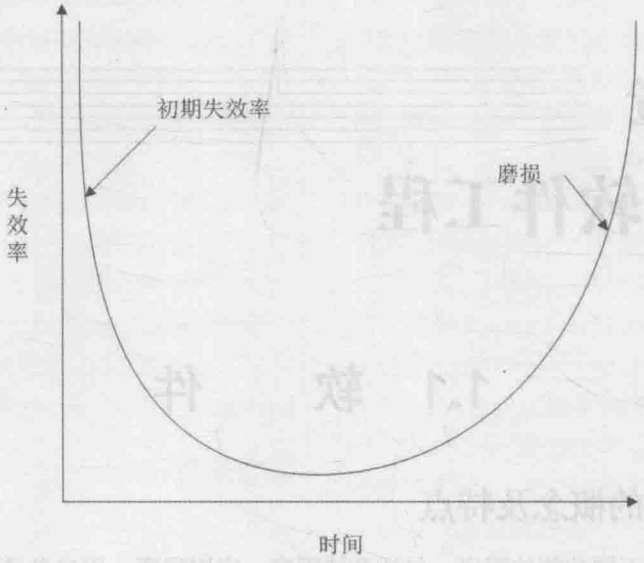


图 1-1　硬件失效曲线图

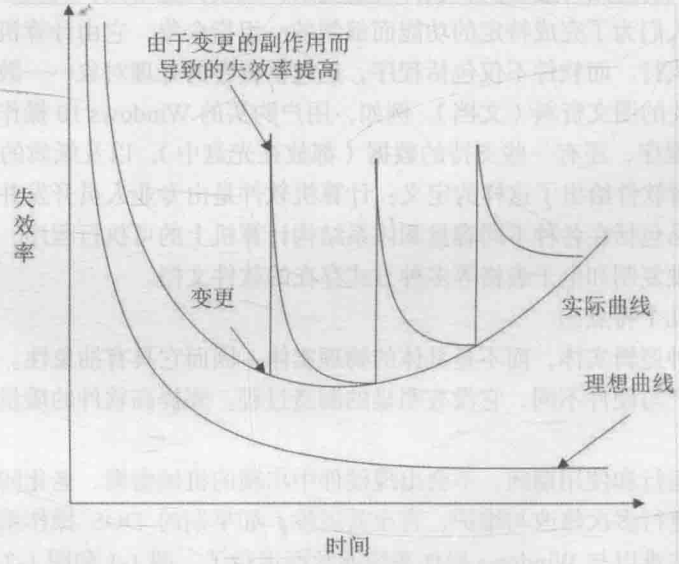


图 1-2　软件失效曲线图

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 3 类。系统软件是计算机系统中最靠近硬件的一层，为其他程序提供最底层的系统服务，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，支撑软件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。例如，某个学生为完成软件工程课程的作业而编制的程序，就属于微型软件。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。例如，某个学生团队为完成软件工程课程的大作业（学期项目）而编制的程序，就属于小型软件。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 5 000～50 000 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。例如，某个软件公司为某个客户开发的办公自动化系统（OA）而编制的程序，就属于中型软件。大型软件是指 10～100 名开发人员在 1～3 年的时间内开发的，具有 50 000～100 000 行（甚至上百万行）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需的。例如，某个软件公司开发的某款多人在线的网络游戏，就属于大型软件。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小的子项目，由不同的团队开发。例如微软公司开发的 Windows 10 操作系统，就属于超大型软件。

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。

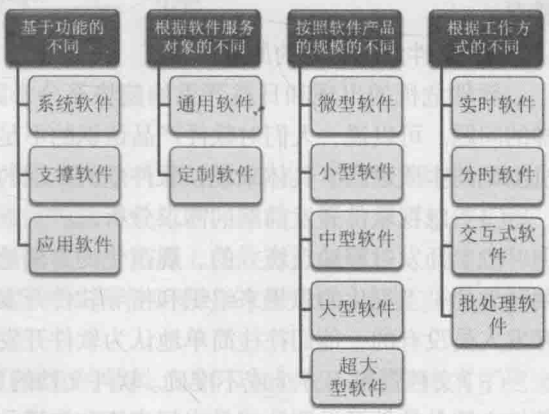


图 1-3　软件的分类

1.2　软 件 危 机

1.2.1　软件危机的表现与原因

1. 软件危机的表现

软件危机就是指人们在开发软件和维护软件过程中遇到的一系列问题。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，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问题。这些情况逐渐使人们意识到软件危机的存在及解决危机的重要性。

计算机软件开发、维护和应用过程中普遍出现的严重问题主要表现如下。

（1）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差异。

（2）相比越来越廉价的硬件，软件代价过高。

（3）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诸多缺陷。

（4）难以准确估计软件开发、维护的费用以及开发周期。往往软件产品不能在预算范围之内，按照计划完成开发。在很多情况下，软件产品的开发周期或经费会大大超出预算。

（5）难以控制开发风险，开发速度赶不上市场变化。

（6）软件产品修改维护困难，集成遗留系统更困难。

（7）软件文档不完备，并且存在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。软件文档不完整和不一致会给软件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。20 世纪 60 年代初期，IBM 公司组织了 OS/360 操作系统的开发，这是一个超大型的软件项目，它使用了 1 000 名左右的程序员。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。

2. 软件危机出现的原因

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。

（1）忽视软件开发前期的需求分析。

（2）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没有的，他们往往简单地认为软件开发就是程序设计。

（3）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，就势必会给软件开发带来很多不便。

（4）忽视与用户之间、开发组成员之间的交流。

（5）忽视测试的重要性。

（6）不重视维护或由于上述原因造成维护工作困难。软件的抽象性和复杂性使得软件在运行之前，很难估计开发过程的进展情况。再加上软件错误的隐蔽性和改正的复杂性，使得软件开发和维护在客观上比较困难。

（7）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，人们对它的认识缺乏深刻性。

（8）没有完善的质量保证体系。建立完善的质量保证体系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，从而导致软件质量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。

（1）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。

（2）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。

（3）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，并逐步向工程一段漂移，但很难发展到完全的工程。

（4）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。

（5）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，适当调整设计和实现手段是不可避免的。

（6）“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。

随着 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（Institute of Electrical and Electronics Engineers，电气和电子工程师协会）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。

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地开发。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。而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。

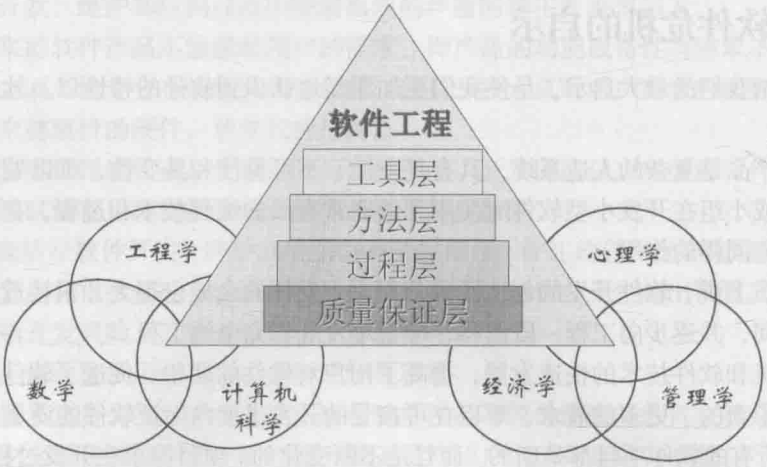


图 1-4　软件工程层次

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 3 个要素。

1.3.2　软件工程研究的内容

软件工程研究的内容主要包括以下两个部分。

（1）软件开发技术。主要研究软件开发方法、软件开发过程、软件开发工具和环境。

（2）软件开发过程管理。主要研究软件工程经济学和软件管理学。

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

1.3.3　软件工程的目标和原则

软件工程要达到的基本目标如下。

（1）达到要求的软件功能。

（2）取得较好的软件性能。

（3）开发出高质量的软件。

（4）付出较低的开发成本。

（5）需要较低的维护费用。

（6）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 综合了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. 用分阶段的生命周期计划严格管理

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，人们可以根据软件的特点或目标，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，这样有益于管理整个软件开发过程。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

严格贯彻与实施阶段评审制度可以帮助软件开发人员及时发现错误并将其改正。在软件开发过程中，错误发现得越晚，修复错误要付出的代价就会越大。实施阶段评审，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取到全部正确的需求。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。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，如面向对象，可以使开发出来的软件产品更易维护和修改，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准确地审查和度量，这样才有利于有效管理项目。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。

1.3.4　软件工程的知识体系

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

软件需求（Software Requirements）涉及软件需求的获取、分析、规格说明和确认。

2. 软件设计

软件设计（Software Design）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

软件构建（Software Construction）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

软件测试（Software Testing）是为评价、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

软件维护（Software Maintenance）是指由于一个问题或改进的需要而修改代码和相关文档，