

信息与电子学科百本精品教材工程

新编计算机类本科规划教材

软件工程方法与实践

李 芷 窦万峰 任满杰 编著

<http://www.phei.com>

.5



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

新编计算机类本科规划教材

软件工程方法与实践

李芷 窦万峰 任满杰 编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

软件工程学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、工程开发技术和工具,是指导软件生产和管理的一门新兴的、综合性的应用科学。本书以传统的软件工程和面向对象的软件工程为主线,根据软件开发“工程化”思想,结合大量的应用示例,系统地介绍软件工程学的基本原理、软件过程、开发方法、应用技术和实用工具。

本书适合作为高等院校计算机科学与技术专业学生的教科书,也可以作为通用软件工程课程,或者需求工程学、软件设计、高级程序设计和软件管理等软件工程学专题的教科书,还可以作为从事计算机软件系统研究和项目开发等应用型技术/管理人员的参考书。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有,侵权必究。

图书在版编目(CIP)数据

软件工程方法与实践/李芷等编著. —北京:电子工业出版社, 2004.11

新编计算机类本科规划教材

ISBN 7-121-00455-0

I. 软… II. 李… 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(2004)第105296号

责任编辑:李岩 特约编辑:陈新中

印 刷:北京牛山世兴印刷厂

出版发行:电子工业出版社

北京市海淀区万寿路173信箱 邮编:100036

经 销:各地新华书店

开 本:787×1092 1/16 印张:16.25 字数:415千字

印 次:2004年11月第1次印刷

印 数:5000册 定价:22.00元

凡购买电子工业出版社的图书,如有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系。
联系电话:(010)68279077。质量投诉请发邮件至 zltz@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

前 言

软件工程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、工程开发技术和工具，是指导软件生产和管理的一门新兴的、综合性的应用科学。随着计算机科学和软件产业的迅猛发展，软件工程学已成为一个重要的计算机分支学科，一个异常活跃的研究领域，正在不断涌现新方法、新技术，蓬蓬勃勃地发展着。

软件工程是计算机科学与技术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。

本书编著者结合长期教学经验和工程项目实践，参考国内外众多最新（版本）教材和论文精选内容，注重基础性、系统性、实用性和新颖性，并结合大量软件项目的实例分析，深入浅出地阐述软件工程方法、应用技术和实用工具。

本书共分 10 章。第 1~2 章作为软件工程导论，分专题介绍了软件工程的一些重要概念、软件工程学研究范畴、软件工程方法学基本原理、软件开发过程和软件过程模型等。第 3~9 章以传统的和面向对象的软件工程为主线，系统介绍软件开发过程主要阶段的实现过程、采用的方法和技术、应用实例剖析，以及软件项目管理等。这 7 章为软件工程方法学和应用实践的结合提供了一个蓝本，是本书学习（参考）的重点和应用实践的指南。第 10 章简述软件工程学的高级研究课题。

第 1 章“软件工程引论”从软件的特点、软件危机的发生，引入软件工程化概念、软件工程定义和软件工程学研究范畴，从而诞生了软件工程学这一门新兴学科。随着计算机科学和软件产业的迅猛发展，软件工程学已成为一个异常活跃的研究领域，正在不断涌现新方法、新技术。

第 2 章“软件过程”介绍软件工程方法学的分解与抽象、逐步求精、软件生存周期、软件过程等基本概念，概述软件过程从问题定义到退役的阶段划分和各个阶段的任务，讨论典型的软件过程模型（瀑布、原型、增量、螺旋等模型）和面向对象软件过程模型（RUP、构件集成模型）的特征。本章是读者学习以后各章内容的重要指南。

第 3 章“软件需求分析”介绍需求分析过程、需求分析方法、需求规格说明文档。典型而实用的需求分析方法有非形式化需求分析、快速原型技术、结构化分析（SA）等。本章采用结构化分析技术，给出一个“图书馆系统”实例分析。

第 4 章“软件设计”介绍软件设计过程、设计说明文档、软件系统模型、软件设计方法。传统的软件设计方法有面向数据流分析（DFA）的设计、面向数据的设计（JSD 和 LCP）等。本章采用 DFA 设计技术，给出一个“汽车数字化仪表盘控制”实例设计。

第 5 章“面向对象的分析和设计”介绍面向对象分析（OOA）和面向对象设计（OOD）技术。实际上，根据面向对象方法学理论，OOA 和 OOD 之间没有严格划分，可以认为 OOD 是 OOA 的逐步求精过程，即 OOA 和 OOD 应该是“一气呵成”的。本章采用面向对象技术，给出“图书馆系统”和“电梯控制系统”实例的 OOA 和 OOD。其中，“图书馆系统”实例的 OOA 与第 3 章的“图书馆系统”实例分析，便于读者对这两种分析方法做类比性参考。

第 6 章“软件实现”介绍软件根据其设计“翻译”成程序源代码的实现过程和系统集成

过程。软件产品的实现，主要依赖于合理选择编程语言和采用良好的程序设计技术。软件系统的集成，主要介绍集成方法、系统集成环境等相关技术。软件集成的测试将在第7章中讨论。

第7章“软件测试”介绍软件测试的测试任务、测试原理，常用的测试策略、测试方法和测试技术，以及一些特定应用系统的软件（面向对象软件、人机界面软件、分布式软件、实时系统软件）的测试要点。

本书有一个重要的观点，认为软件测试不是一个独立的阶段。它不是仅仅在产品交付前才进行，也不是在软件开发过程的各个阶段结束时才进行，而是和所有软件生产过程的各个环节并行。因此，本章介绍的软件测试方法和技术贯穿在软件开发全过程，当然也包括贯穿在软件过程各个独立阶段的测试/验证环节中。

第8章“软件维护”介绍软件产品从“发布”到“退役”为止，整个运行期间软件维护的全过程，着重讨论维护工作的特点、维护任务的类型、提高可维护性的技术等。

第9章“软件项目管理”介绍软件项目的人员、过程、计划、度量等全方位的协调管理，着重讨论了软件项目管理计划（SPMP）、软件成本估计、软件质量保证、项目过程管理等内容。

和软件测试一样，软件项目管理工作贯穿在软件开发全过程。

第10章“软件工程高级课题”介绍软件工程学的一些新的高级研究课题：软件复用、新型软件过程模型（形式化方法、再工程、客户/服务器软件工程等）和CASE等。

本书内容适宜，编写有特色，应用指导性强。每章开头有前言，结束有小结和练习思考题。对开发过程中较为实用的方法和技术给出系列化应用示例。建议教学60学时，其中40学时授课，20学时做一个适当规模的软件项目开发实践训练。

本书适用面广，可以作为高等院校计算机科学与技术专业学生的教科书，也可以作为通用软件工程课程，或者是需求工程学、软件设计、高级程序设计和软件管理等软件工程专题的教科书，也可以作为从事计算机软件系统研究和项目开发等应用型技术/管理人员的参考书。

本书由李芷、窦万峰、任满杰编著，李芷主编。窦万峰执笔第3、4、5章和9.3节，任满杰执笔第8章和6.3节，其余由李芷执笔。江苏大学查杰民教授审阅了全书，并对编写提出了宝贵意见。此外，靳松、黄永来、陈晔、齐宁超、吴奕斐、李春萍、李桂香、周兴旺、吕培祥、宗雷等参加了资料整理工作。谨在此一并向他们致以衷心的感谢。

编著者水平有限，难免有疏漏和不当之处，敬请广大读者不吝赐教。

编 著 者

2004年5月

目 录

第 1 章 软件工程引论	1
1.1 软件工程诞生背景	1
1.1.1 软件的特性	1
1.1.2 软件危机	2
1.1.3 软件危机解决途径	4
1.2 软件工程学	5
1.2.1 软件工程化思想	5
1.2.2 软件工程学研究范畴	6
1.2.3 软件工程学的发展	9
1.3 软件工程方法学基本原理	9
1.3.1 推迟实现	9
1.3.2 逐步求精	10
1.3.3 抽象建模	11
1.3.4 质量保证	12
1.4 小结	14
习题 1	15
第 2 章 软件过程	16
2.1 软件过程概述	16
2.1.1 过程、方法和工具	16
2.1.2 软件生存周期	17
2.1.3 软件过程模型	18
2.1.4 软件过程各阶段任务	18
2.2 典型的软件过程模型	21
2.2.1 瀑布模型	21
2.2.2 快速原型模型	22
2.2.3 增量模型	24
2.2.4 螺旋模型	25
2.3 面向对象的软件过程模型	26
2.3.1 面向对象的基本概念	26
2.3.2 软件开发统一过程	30
2.3.3 构件集成模型	31
2.4 小结	32
习题 2	32

第 3 章 软件需求分析	33
3.1 需求分析概述	33
3.1.1 需求分析的任务	33
3.1.2 需求分析过程	35
3.1.3 需求分析的原型技术	37
3.2 非形式化需求分析	37
3.2.1 会谈	38
3.2.2 调查表	39
3.2.3 场景分析	40
3.3 结构化需求分析	40
3.3.1 结构化分析方法	41
3.3.2 结构化分析模型	41
3.3.3 结构化分析步骤	47
3.3.4 结构化分析实例：图书馆系统	49
3.4 需求分析的描述	52
3.4.1 需求描述的分类	52
3.4.2 需求规格说明文档	53
3.5 小结	55
习题 3	55
第 4 章 软件设计	57
4.1 软件设计概述	57
4.1.1 软件设计过程	57
4.1.2 软件模块化设计	59
4.1.3 软件系统结构模型	62
4.1.4 软件设计说明文档	65
4.2 面向数据流分析（DFA）的设计	66
4.2.1 数据流类型	66
4.2.2 数据流分析的设计步骤	67
4.2.3 数据流分析设计技术	68
4.2.4 DFA 设计实例：汽车数字化仪表盘控制	69
4.3 面向数据的设计	72
4.3.1 Jackson（JSD）方法	73
4.3.2 Warnier（LCP）方法	76
4.4 软件详细设计	77
4.4.1 结构化程序设计	77
4.4.2 详细设计工具	77
4.5 特定应用软件的设计	79
4.5.1 实时系统设计	79

4.5.2	人机界面设计	83
4.6	小结	85
	习题 4	86
第 5 章	面向对象的分析与设计	88
5.1	面向对象建模	88
5.1.1	面向对象模型	89
5.1.2	统一建模语言 (UML)	90
5.2	面向对象分析 (OOA)	93
5.2.1	面向对象分析模型	93
5.2.2	面向对象分析过程	96
5.2.3	OOA 实例: 图书馆系统	97
5.2.4	OOA 实例: 电梯控制系统	101
5.3	面向对象设计 (OOD)	104
5.3.1	面向对象设计模型	104
5.3.2	面向对象设计过程	105
5.3.3	构件类设计	108
5.3.4	OOD 实例: 图书馆系统	109
5.3.5	OOD 实例: 电梯控制系统	112
5.4	小结	114
	习题 5	115
第 6 章	软件实现	116
6.1	编程语言选择	116
6.1.1	程序设计语言的发展	117
6.1.2	程序设计语言的特点	118
6.1.3	编程语言的选择	121
6.2	程序设计技术	122
6.2.1	编码风格	123
6.2.2	面向对象程序设计	126
6.2.3	程序设计自动化	127
6.3	软件系统集成	129
6.3.1	软件系统集成方法	130
6.3.2	系统集成过程	132
6.3.3	面向对象的系统集成	133
6.4	小结	133
	习题 6	134
第 7 章	软件测试	136
7.1	软件测试概述	136
7.1.1	软件测试任务	136

7.1.2	软件测试原理	137
7.1.3	测试方法分类	139
7.1.4	测试终止标准	141
7.2	软件测试技术	141
7.2.1	测试用例设计	142
7.2.2	黑盒测试方法	142
7.2.3	白盒测试方法	145
7.2.4	自动测试工具	150
7.3	软件调试技术	151
7.3.1	软件调试过程	151
7.3.2	软件调试策略	152
7.4	多模块的软件测试	154
7.4.1	多模块的测试层次	154
7.4.2	单元测试	155
7.4.3	集成测试	155
7.4.4	确认测试	156
7.4.5	系统测试	156
7.5	特定应用软件的测试	156
7.5.1	面向对象软件测试	157
7.5.2	人机界面软件测试	159
7.5.3	分布式软件测试	162
7.5.4	实时软件测试	163
7.6	小结	165
	习题 7	166
第 8 章	软件维护	167
8.1	软件维护概述	167
8.1.1	维护阶段的任务	167
8.1.2	软件维护的特点	168
8.1.3	软件可维护性	168
8.2	软件维护类型	169
8.2.1	改正性维护	169
8.2.2	完善性维护	170
8.2.3	适应性维护	170
8.2.4	预防性维护	170
8.3	软件维护技术	171
8.3.1	软件维护过程	171
8.3.2	提高软件可维护性	174
8.3.3	面向对象的软件维护	175

8.4 小结	176
习题 8	176
第 9 章 软件项目管理	177
9.1 软件项目管理范围	177
9.1.1 人员	177
9.1.2 项目和过程	179
9.1.3 软件度量	179
9.1.4 软件项目管理文档	181
9.2 人员组织范式	183
9.2.1 民主分权式	184
9.2.2 控制集权式	185
9.2.3 控制分权式	186
9.3 软件成本估算	187
9.3.1 软件规模度量	187
9.3.2 成本估算分解技术	189
9.3.3 经验估算模型	192
9.3.4 成本估算管理	195
9.4 软件质量保证 (SQA)	195
9.4.1 SQA 活动	196
9.4.2 软件质量度量	197
9.4.3 软件复审	201
9.4.4 软件认证标准	202
9.5 软件过程管理	206
9.5.1 过程度量和过程改进	206
9.5.2 项目进度安排	208
9.5.3 风险分析	211
9.5.4 软件配置管理 (SCM)	213
9.6 小结	215
习题 9	216
第 10 章 软件工程高级课题	218
10.1 软件复用	218
10.1.1 软件复用的基本概念	218
10.1.2 软件复用过程	220
10.1.3 领域工程	221
10.1.4 基于构件的软件开发	224
10.2 新型软件过程模型	225
10.2.1 形式化方法	226
10.2.2 软件再工程	228

10.2.3 客户-服务器软件工程	231
10.3 计算机辅助软件工程 (CASE)	234
10.3.1 软件工程环境	234
10.3.2 CASE 环境	236
10.3.3 CASE 实例: SUITE 企业开发环境	239
10.4 小结	243
习题 10	244
附录 A 缩略词中英文对照表	245
参考文献	247

第 1 章 软件工程引论



教学重点

本章主要介绍软件的基本概念、软件工程学科诞生的背景、软件工程学的研究范畴，以及软件工程的发展和应用。随着计算机科学和软件产业的迅猛发展，软件工程学已成为一个异常活跃的研究领域，正在不断涌现新方法和新技术。

重点掌握：软件的特性、软件工程学的研究范畴，以及学习软件工程的意义。

计算机使社会生产力得到迅速解放、社会高度自动化和信息化。计算机软件，在这场席卷全球的信息技术（IT）革命中无疑是最重要的；计算机软件产业，作为一个具有独立形态的产业，在全球经济中占据着越来越重要的地位；而软件工程，在软件产业发展中起到关键的技术保障和促进作用。

三十多年来，以解决软件生产的质量和效益问题为宗旨的软件工程，在软件学术界和软件产业界共同努力下，理论、方法和技术等诸多方面出现了大量的研究成果和技术实践，逐步形成了一门成熟的计算机分支学科，并且还在不断地探索和发展着。

1.1 软件工程诞生背景

随着计算机科学与技术的迅猛发展，计算机应用更加普及，使得科学和技术各个领域、工业和社会各个部门对高性能计算机系统的需求量迅速增加，依赖性日益增强。

和计算机硬件一样，从 20 世纪 60 年代以来，计算机软件也从品种、规模、功能等方面得到了很大发展。然而，由于对软件质量的要求越来越高，软件的规模越来越大，也越来越复杂，人们的软件开发能力显得力不从心。于是，引起了软件界人们的思考：软件的问题究竟出在哪里？什么是软件？软件有哪些特性？

1.1.1 软件的特性

对于计算机软件，绝大多数初学者，甚至包括早期的软件界人士都普遍认为软件就是程序，围绕“软件”的一切就是编写程序（编程）。显然，对软件的这种理解是不完全的。若要对软件有一个全面而正确的理解，首先要从软件的本质、软件的生产等方面剖析软件的特性。

1. 软件（产品）固有特性

如果是建造/生产硬件产品，从设计图纸、生产零/部件到装配/组装成产品，每一步生产结果都可转化成物理形式，最终演化成所需要的物理产品。而软件产品，却与硬件产品有着本质的区别。从外部特征来看，软件是一种非实物型的信息产品；从内部特性来说，软件是

一个具有高度抽象性和严密逻辑性的逻辑系统（逻辑元素的集合）。

软件（产品）是逻辑的而不是物理的，本质上存在着复杂性、一致性、易变性和不可见性等固有特性。

① 复杂性。软件是一个庞大的逻辑系统，比任何人类构造的其他产品更复杂，甚至硬件的复杂性和软件比起来也是微不足道的。此外，软件主要是靠人脑“智力”构造出来的，多种人为因素使得软件难以统一化，更增加了其复杂性。软件的复杂性使得软件产品难以理解、难以生产、难以维护，更难以对生产过程进行管理。

② 一致性。软件必须和运行软件的机器（硬件）保持一致，这是计算机的基本原理。虽然可以用机器顺应软件，或者软件顺应机器来保持一致，但一般都采用软件顺应硬件接口，而不是硬件顺应软件的方案。如果硬件系统是“现存”的，软件必须和现有硬件系统接口。还有一种情况，由于计算机的软件和硬件是具有功能互换性的，所以也可能出现用软件来替代硬件接口的功能。无论使软件适应硬件接口，还是用软件替代硬件功能，软件设计人员都必须将软件结构做得复杂，从而迫使软件复杂性达到一定程度。

③ 易变性。软件在生产过程中，甚至在投入运行之后，也还可以再改变。软件必须经历变化并容易改变，这是软件产品的特有属性。软件易变性的好处是：改变软件往往可以收到改变或者完善系统功能的功效；修改软件毕竟比更换硬件容易，使得软件易维护、易移植、易复用。修改软件的压力总是存在的，所以软件产品始终在“变”。这种动态的变化不仅难以预测、难以控制，而且可能对软件的质量产生负面影响。

④ 不可见性，或称为不可可视化。软件是一个逻辑元素的集合，显然没有物理产品直观。软件无法形式化表示的结果，不仅使软件难以理解、难以维护，而且严重妨碍了软件专业人员之间的交流。

软件（产品）的这些本质特性是软件生产中可能无法改变的方面，而非本质问题总是有可能获得研究上的突破和实际上的解决。

2. 软件（产品）生产特性

由于软件（产品）是逻辑的而不是物理的，在开发、生产、维护和使用等方面，都具有与硬件完全不同的特征。

① 软件开发不同于硬件设计。与硬件设计相比，软件开发更依赖于开发人员的素质、智力，以及对人员的组织和管理。软件开发成本很难估计，不能像硬件设计那样来管理。

② 软件生产形式不同于硬件制造。硬件设计完成之后就投入批量制造，每个硬件产品的制造都是一个复杂过程。而软件成为产品之后，其制造仅仅是简单的复制而已。

③ 软件维护不同于硬件维修。硬件产品在运行初期有较高的故障率（这是源于存在设计或制造缺陷），故障排除之后的一段时间故障率会降到一个较低的稳定水平，随着使用时间的增加，产品会因硬件磨损等损害，故障率将再次升高，故障率达到一定程度之后硬件产品报废。软件产品虽然不会磨损和老化，但是，在使用过程中的维护比硬件产品复杂得多，而且维护过程难以控制，甚至还可能产生新的错误。

1.1.2 软件危机

当代计算机硬件更新换代迅速，成本急速下降，性能迅速提高。遗憾的是，计算机软件却没有得到类似的巨大进步，软件生产方式落后，软件生产率远远跟不上计算机应用的需求。

20 世纪 60 年代，计算机软件界普遍感觉到，随着软件规模和复杂性的急剧增加，几乎所有软件都不同程度地出现了一系列严重问题，于是发出了“软件危机（Software Crisis）”警告。

1. 软件危机的突出表现

概括地说，软件危机主要表现在软件开发和维护过程中的高成本、低质量和难以维护。具体地说，软件危机主要有以下一些突出表现。

① 软件开发生产率提高的速度，远远跟不上计算机应用迅速普及深入的趋势。软件产品的供需差不断扩大，使得人们为不能充分利用计算机硬件提供的巨大潜力而苦恼。

② 由于微电子技术的进步和硬件生产自动化程度的提高，硬件成本逐年下降。然而，软件由于规模和数量的扩大，软件开发人力需求的激增，软件成本逐年上升。软、硬件成本在计算机系统总投资中的比例迅速逆转。例如，1985 年，美国空军的软件成本大约占计算机系统总成本的 90%。

③ 对软件开发成本和进度的估计常常很不准确。实际成本比估计的可能高出一个数量级，实际进度比预期的可能拖延几个月，甚至拖延几年的现象也不罕见。为了赶进度和节约成本所采取的一些权宜之计，往往会损害软件的质量。这些现象都大大降低了软件开发组织的信誉。

④ 软件开发人员和用户之间的信息交流往往很不充分。软件开发人员可能只模糊地理解了用户要求，甚至对所要解决的问题还没有确切认识的情况下，就仓促上阵匆忙着手编写程序。这样“闭门造车”的产品，必然不符合用户的实际需要，所以，用户对已完成的软件产品不满意的现象经常发生。

⑤ 软件规模的增长，带来了它的复杂度的增加。如果说编写一个数百行的程序连初学者也不难完成，而开发一个数万行以至数百万行的软件，其复杂度则大大上升，即使是富有经验的程序员，也难免顾此失彼。所以，软件的可靠性往往随着软件规模的增长而下降，质量保证越来越困难。

⑥ 软件生产形式是自由化的“个体作坊”式，软件开发方法是优劣共存的“合成/捏合”技术。这种软件开发的“无政府”状态，使得软件开发人力即使严重不足，也仍然在大量重复地开发类似的，或者基本类似的软件。

⑦ 软件往往需要不断地“修修补补”，不可维护性突出。软件的局限性和欠灵活性，不仅使错误非常难改正，而且不能适应新的硬件环境，也不能根据需要增加一些新的功能。更头痛的是，整个软件维护过程除了程序之外，没有适当的文档资料可供参考。

2. 产生软件危机的原因

追究产生软件危机的原因，除了软件自身固有的特点之外，还与软件开发和维护的方法不正确有关。

（1）软件独有的特点给开发和维护带来困难

软件不同于硬件，它是计算机系统中的逻辑部件而不是物理部件。在程序运行之前，软件开发过程的进展情况较难衡量，软件开发的质量也较难评价，因此，管理和控制软件开发过程相当困难。此外，在软件运行过程中如果发现错误，很可能就是一个在开发早期就引入的、未被检测出来的故障。软件维护通常意味着要修改原来的开发，或者是重新开发，在客观上使得软件维护较为困难。

软件不同于一般程序，它的一个显著特点是规模庞大。例如，美国宇宙飞船的软件呈指数增长，到了 20 世纪 70 年代末，第四代宇宙飞船的软件有 4000 万行目标代码。若一个人一年可开发出 1 万行程序，开发 4000 万行的软件，是否集中 4000 人的力量一年就可以完成呢？绝对做不到！因为代码长度增加了 4000 倍，程序的复杂程度远远超过了 4000 倍。而且，如何保证每个人完成的工作合在一起确实能构成一个高质量的大型软件系统，更是一个极端复杂而困难的问题。这不仅涉及许多技术问题，诸如分析方法、设计方法、形式说明方法、版本控制等，更重要的是必须有严格而科学的管理。

(2) 软件人员的错误认识

人们在开发和和使用计算机系统的长期实践中，也积累和总结了许多成功的经验。如果所有的人，都坚持不懈地使用经过实践考证明是正确的方法，许多困难是可以克服的。但是，相当多的软件专业人员对软件开发和维护还有不少糊涂观念，在各自的实践中或多或少采用了一些错误的方法和技术。例如，认为软件开发就是编写程序，忽视软件需求分析的重要性，轻视软件维护等。

(3) 软件生产技术进步缓慢

早期软件开发方法植根于计算机硬件性能比较低下、计算机应用尚不广泛的年代，因而无法摆脱人工方式范畴，无法避免重复性劳动，这大量浪费了宝贵的人力、财力和物力资源。

随着软件规模和复杂度急剧增加，软件生产仍然长期未突破手工作坊方式，软件开发技术进步也十分缓慢，这严重阻碍了软件产业的发展，制约了软件生产率的提高。

软件质量的度量标准、软件的质量保证等，从概念、理论、机制，到实现技术都很不完善，更没有应用到软件开发全过程。这些都导致软件产品发生质量问题，而且难以解决。

(4) 软件维护成本和维护难度急剧增加

根据一些大公司的统计，软件维护成本大约占到软件总成本的 2/3，比开发成本高出了一倍。不言而喻，软件的规模愈大，软件维护成本必然愈高。这是因为，软件维护工作有“纠错性维护”、“完善性维护”、“适应性维护”等多种类型的维护活动，与软件开发工作相比，是周期更长、更繁杂的。此外，软件维护既耗费财力，也耗费人力。软件维护需要投入大量的软件人员，这样更加剧了软件开发人力资源的缺乏。所以，难怪有些文献把维护工作中出现的问题比做“冰海中横在前进航道上的冰山”，或称为维护墙 (Maintenance Wall)，视之为软件维护难以逾越的障碍。

1.1.3 软件危机解决途径

解决软件危机主要是探求两方面的问题：如何高效地开发软件，满足对软件日益增长的需求；如何对数量不断扩大的已有软件，实施有效的维护。

① 软件开发不再是个体劳动的神秘技巧，而应该是一种组织良好、管理严密、各类人员协同配合、共同完成的生产项目；必须充分吸取和借鉴人类长期以来从事各种工程项目所积累的行之有效的概念、原理、技术和方法；特别要吸取几十年来人类从事计算机硬件研究和软件开发的经验教训。

② 应该推广和使用在软件开发实践中总结出来的成功的技术和方法，并且研究探索更好、更有效的技术和方法，尽快纠正正在计算机早期发展阶段所形成的一些错误概念和做法。

③ 应该制定软件开发过程的规范和标准。因为人类的一切生产活动只有纳入科学的、规范的轨道，执行有约束力的标准化，才是最高效的。

④ 应该开发和使用更多、更好的软件工具 (Software Tools)。借鉴“利用机械工具可以放大人的体力”原理,软件工具可以“放大”人的智力。软件开发过程许多繁琐而重复的工作,可以在适当的软件工具辅助下做得又快又好。

总之,为了解决软件危机,使软件生产和软件维护从根本上得到改进,既要有技术措施(方法和工具),又要有必要的组织管理措施。软件工程 (Software Engineering) 正是从技术和管理两个方面研究如何更好地开发和维护计算机软件的一门新兴学科。软件工程的诞生反映了软件界人士应用已有工程规则的理论 and 模式,谋求解决软件危机所做的积极努力。

1.2 软件工程学

由于认识到软件的设计、实现、维护和传统的工程规则有相同的基础,北大西洋公约组织 (NATO, North Atlantic Treaty Organization) 于 1968 年在德国 Garmisch 召开的计算机国际学术会议上提出了“软件工程”这一术语。软件工程名称本身就反映出将软件生产作为一种类似于工程行动的思想。

软件工程是指导计算机软件(产品)开发、维护和管理的工程学科。它强调软件产品的生产特性,采用工程的概念、原理、技术和方法来开发与维护软件,并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。

1.2.1 软件工程化思想

软件工程化的基本思想是,把软件当做一种需要计划、分析、设计、实现、测试、维护和管理的产品,要求在一系列仔细地控制和系统化的执行过程中研制(开发)出来。软件(产品)生产如果这样做的话,不仅能按预期的进度和成本完成软件生产计划,更能提高软件的生产率和保证软件的质量。

1. 软件工程化

软件专家们借鉴传统工程领域的生产和管理技术,把软件(产品)生产过程演变成一个工程项目过程,从而提出了软件工程的观念。软件工程在许多方面与传统的工程相似,但也有独特的属性和问题,所以要在借鉴的前提下,善于总结、研究和创新。

软件工程化是将计算机科学和工程方法相结合,用完善的工程原理和技术研究软件生产的经济、可靠的规范方法。它既强调软件(一般指大型软件)开发的工程特点,又强调软件开发方法和技术的科学性和先进性。它重视对软件开发有指导作用的理论、概念的定义,减少软件开发的复杂性,加强开发过程控制和管理。

根据软件工程化思想,“软件工程师”替代了“程序员”,成为更为确切的工作头衔和备受青睐的职业。

2. 软件工程定义

计算机软件界曾从不同角度给软件工程下过各种定义。无论有多少种说法,软件工程的中心思想是,把软件当做一种工业产品,要求采用工程化的原理和方法对软件进行规范的计划、开发和维护。

软件专家们在软件工程领域总结、改进和完善基于经典开发范式涉及的理论、方法和技

术的同时，探索软件开发的新理论、新思想、新方法和新技术等各个方面，从而逐步形成了“软件工程学”这一计算机新兴学科。

软件工程学是将计算机科学理论与现代工程方法论相结合，围绕软件生产过程自动化和软件产品质量保证，展开对软件生产方式、生产管理、软件开发方法、生产工具系统和产品质量保证的系统研究。

随着计算机科学和软件产业的迅猛发展，软件工程学已成为一个重要的专业学科，一个异常活跃的研究领域。

3. 软件定义

软件的发展（包括人们对软件的认识）经历了程序、软件和软件产品三个阶段。早期的程序规模较小，就直接称为程序。随着程序种类的增多和规模的扩大，人们把程序称为软件，进而把系统程序和应用程序，分别称为系统软件和应用软件。但是，无论程序或软件，在开发过程中都很少考虑到它们的维护问题。在软件工程兴起之后，人们才把软件视为产品，并强调软件的维护特性。这是因为软件工程主要讨论软件的生产过程，而软件生产过程的最终结果是软件产品，所以在软件工程背景下，软件就是指表现其生产特性的软件产品。

从提高软件开发效率和方便软件产品维护角度，软件工程还引入了文档（Documents）的概念。一个缺乏文档资料仅提供程序清单的软件，不能够称之为软件（产品）。所以，软件（产品）被更为合理地、简明地定义为：

$$\text{软件（产品）} = \text{程序} + \text{文档}$$

这就是说，计算机软件不仅仅是程序，还应该有一整套记录软件开发过程的文档资料。

文档是软件开发活动的记录，用于软件开发、管理和维护过程中开发者、用户、管理者等各类人员之间的通信和交流。软件开发人员可以利用文档作为通信工具，在软件开发过程中准确地交流信息；软件开发组织管理人员可以使用文档作为“里程碑”，管理和评价软件开发工程的进展状况；对于软件维护人员，文档更是至关重要的；缺乏必要的文档资料，或者文档资料不合格，必然会给软件维护带来许多意想不到的困难。

现在，软件科学界和软件产业界越来越重视文档的作用及其标准化工作。我国国家标准局参照国际标准，陆续颁布了《计算机软件开发规范》、《计算机软件需求说明编制指南》、《计算机软件测试文件编制规范》、《计算机配置管理计划规范》等一系列文档规范。

1.2.2 软件工程学研究范畴

软件工程学是研究软件开发过程模型、设计方法、工程开发技术和工具，指导软件生产和管理的一门综合性的应用科学。因此，软件工程学涉及计算机科学、方法学、系统工程学、经济学、管理学、心理学和法律学等多个学科领域。

软件工程学的研究范畴如图 1.1 所示，分成软件开发技术和软件工程管理两个方面，重点是对软件开发方法和工程性技术的研究。软件开发技术和软件工程管理的复杂程度，均与软件的规模密切相关。规模越大的软件（产品），越要严格遵守软件工程的开发原则和方法。

下面简述软件工程学的软件开发方法、软件工具、软件工程环境和软件工程管理等主要研究领域的内容。它们包含了广大软件工作者长期实践活动的归纳总结，也包含了他们为实现更高的软件可靠性、更高的软件生产自动化程度所进行的一系列理论、方法和技术的探索研究。