

高等院校教材

软件工程学 教程

周 苏 王 文 编著

Software Engineering



科学出版社
www.sciencep.com

高等院校教材

软件工程学教程

周 苏 王 文 编著

科学出版社

北 京

内 容 简 介

软件工程学是一门理论性和实践性都很强的学科,它采用工程化的概念、理论、技术和方法来指导编程人员开发与维护计算机软件。本书系统、全面地介绍了传统软件工程的技术,在软件工程的观念、原理、方法及其应用等方面有较详尽的论述,具有较强的系统性和可读性,并力图反映软件工程领域的最新发展。按照软件工程的国家标准来表达和描述软件工程的知识和使软件工程技术具有很强的可操作性,是本书的特色之一。

本书可作为高等院校“软件工程”课程的教材和教学参考书,也可供有一定实践经验的软件开发人员、管理人员参考和作为继续教育的教材,它还可以作为各个级别的计算机软件专业技术资格和水平考试中相关内容的学习辅导用书。

图书在版编目(CIP)数据

软件工程学教程/周苏,王文编著. —北京:科学出版社,2002
(高等院校教材)

ISBN 7-03-010797-7

I. 软... II. ①周... ②王... III. 软件工程—教材 IV. TP311.5

中国版本图书馆 CIP 数据核字(2002)第 071807 号

责任编辑:陈晓萍 韩洁

责任印制:吕春珉/封面设计:王浩

科 学 出 版 社 出 版

北京东黄城根北街16号

邮政编码:100717

<http://www.sciencep.com>

新 蕾 印 刷 厂 印 刷

科学出版社发行 各地新华书店经销

*

2002年9月第 一 版 开本: B5(720×1000)

2003年1月第二次印刷 印张: 23

印数: 4 001-6 000 字数: 500 000

定价:30.00 元

(如有印装质量问题,我社负责调换(环伟))

编写说明

软件工程学是用来指导计算机软件开发和维护的工程性学科，它主要研究软件结构、软件设计方法、软件工具、软件工程标准和规范以及软件工程的有关理论。采用工程化的概念、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开发方法结合起来，这就是软件工程。

本书系统、全面地介绍了传统软件工程的技术，在软件工程的概念、原理、方法及其应用等方面有较详尽的论述，具有较强的系统性和可读性，并力图反映软件工程领域的最新发展。

按照软件工程的国家标准来表达和描述软件工程的知识，使软件工程技术具有很强的可操作性，是本书的特色之一。计算机软件开发规范（GB8567-88）和计算机软件产品开发文件编制指南（GB8567-88）等国家标准的有关要求贯穿于本书的始终。

本书选编了大量的思考与习题。本书可作为高等院校“软件工程”课程的教材和教学参考书，也可供有一定实践经验的软件开发人员、管理人员参考和作为继续教育的教材。

本书涉及面广，内容翔实，可作为程序员、高级程序员和系统分析员等各个级别的软件人员水平考试中的有关软件设计能力、软件工程知识等内容的学习辅导用书。

衷心感谢科学出版社的同志们，也感谢在本书编写过程中，直接、间接地给予我们很大帮助的朋友们。

编 者

2002年8月于北京

目 录

第 1 章 传统软件工程概述	1
1.1 计算机系统的发展	2
1.1.1 计算机系统及其软件的发展	2
1.1.2 计算机系统工程	4
1.2 软件和软件生存周期	4
1.3 软件生存周期模型	7
1.3.1 瀑布模型	7
1.3.2 渐增模型	9
1.3.3 演化模型	10
1.3.4 螺旋模型	11
1.3.5 喷泉模型	12
1.3.6 智能模型	13
1.4 软件工程定义	14
1.4.1 软件工程学的内容	15
1.4.2 软件工程过程	16
1.4.3 软件工程学的基本目标和原则	16
1.4.4 软件工程与一般工程的差异	18
1.5 小结	19
思考与习题	20
第 2 章 系统定义与软件计划	21
2.1 系统定义	21
2.1.1 系统分析	22
2.1.2 可行性研究	28
2.1.3 成本-收益分析	29
2.1.4 功能分配	31
2.1.5 系统规格说明	31
2.1.6 系统定义复审	32

2.2	软件计划	33
2.2.1	软件的范围	34
2.2.2	资源	35
2.2.3	软件成本估算	37
2.2.4	软件生产率数据	37
2.3	进度安排	39
2.4	软件计划文件与复审	42
2.5	小结	43
	思考与习题	44
第3章	软件需求分析	46
3.1	需求分析阶段的任务	46
3.2	结构化分析方法	48
3.3	数据流程图	50
3.4	数据字典	54
3.4.1	数据流条目	55
3.4.2	文件条目	56
3.4.3	数据项条目	57
3.4.4	加工条目	57
3.5	加工的分析与表达	59
3.5.1	结构化语言	60
3.5.2	判定表	61
3.5.3	判定树	63
3.5.4	三种表达工具的比较	63
3.6	需求分析工具	64
3.7	软件需求分析文件与复审	65
3.7.1	国标 GB8567-88 规定的文件	65
3.7.2	国标 GB9385-88 规定的软件文件	68
3.7.3	软件需求分析的复审	70
3.8	小结	71
	思考与习题	71
第4章	软件概要设计	73
4.1	模块的划分	73
4.1.1	系统性能的衡量标准	74
4.1.2	软件结构	75
4.1.3	模块划分的基本原则	76

4.1.4 内聚度	76
4.1.5 耦合度	78
4.1.6 高内聚和低耦合	79
4.1.7 模块划分的方法	79
4.2 结构化设计方法	82
4.2.1 变换与事务型数据流分析	82
4.2.2 模块化设计	85
4.2.3 模块结构图	86
4.2.4 从数据流程图导出模块结构图	88
4.3 PARNAS 方法	90
4.3.1 信息隐蔽原则	90
4.3.2 加强系统各成分间的检查	91
4.4 JACKSON 方法	91
4.4.1 JACKSON 方法的基本思想	92
4.4.2 JACKSON 方法的描述方式	92
4.4.3 JACKSON 方法的基本步骤	93
4.5 程序的逻辑构造方法	94
4.5.1 WARNIER 图	94
4.5.2 LCP 设计方法	94
4.6 概要设计文件与复审	95
4.6.1 概要设计说明书	95
4.6.2 概要设计的复审	96
4.7 小结	96
思考与习题	97
第 5 章 软件详细设计	100
5.1 概述	100
5.2 结构化构造	101
5.3 图形设计工具	102
5.3.1 程序流程图	102
5.3.2 方块图	106
5.3.3 HIPO 图	108
5.3.4 PAD 图	109
5.4 伪码与程序设计语言	112
5.5 各种详细设计工具的比较	115
5.6 详细设计文件与复审	116

5.6.1 详细设计说明书	117
5.6.2 详细设计的复审	117
5.7 小结	118
思考与习题	118
第6章 软件界面设计	120
6.1 代码设计	120
6.1.1 代码基础	120
6.1.2 代码的种类	122
6.1.3 各种代码形态的比较	125
6.1.4 代码设计步骤	126
6.1.5 代码设计文件	126
6.2 输出设计	128
6.2.1 输出设计的研究范围	128
6.2.2 输出设计的规则	129
6.2.3 输出种类	130
6.2.4 输出设计方法	131
6.3 输入设计	135
6.3.1 输入设计的研究范围	135
6.3.2 输入设计的规则	136
6.3.3 输入种类	137
6.3.4 输入设计方法	138
6.4 用户界面设计	143
6.4.1 用户界面开发的基本原则	144
6.4.2 出错控制与处理	147
6.5 小结	148
思考与习题	148
第7章 数据结构和数据库设计	150
7.1 数据存储文件设计	150
7.1.1 文件设计内容	151
7.1.2 文件类别	151
7.1.3 文件媒体的选择	152
7.1.4 文件组织方式	153
7.1.5 文件记录格式设计	157
7.2 数据库设计文件	158
7.3 小结	158

思考与习题	158
第 8 章 软件安全性设计	160
8.1 系统安全的基本概念	160
8.1.1 系统安全控制的目的	160
8.1.2 系统中不安全的因素	161
8.2 软件安全控制的基本方法	163
8.2.1 数据的证实	163
8.2.2 用户的同一性检查	165
8.2.3 用户的使用权限检查	165
8.2.4 运行日志	166
8.2.5 违规行为的监查	167
8.2.6 加密	168
8.3 软件的安全控制设计	169
8.4 初级阶段的安全控制计划	169
8.4.1 分析软件安全运行的可能性	170
8.4.2 不安全因素分析	170
8.5 开发阶段的安全控制设计	172
8.5.1 定义安全控制的主要功能	172
8.5.2 软件安全控制的设计	173
8.5.3 程序设计的安全控制	175
8.6 系统运行中的安全控制	176
8.6.1 数据的控制	176
8.6.2 系统工作人员的管理	177
8.6.3 安全教育	177
8.6.4 异常现象的处理	177
8.6.5 硬、软件设备的增加和维护	178
8.6.6 事故处理计划	178
8.6.7 安全管理	179
8.7 小结	179
第 9 章 软件编码	180
9.1 结构化程序设计方法	180
9.2 程序设计语言	182
9.3 程序设计风格	183
9.3.1 源程序	183
9.3.2 数据说明	185

9.3.3 语句结构	186
9.4 源代码文件	186
9.4.1 综合文件	186
9.4.2 程序组织文件	187
9.4.3 指令级注释	189
9.5 冗余程序设计	189
9.6 防错性程序设计	190
9.7 程序设计质量的评价	191
9.8 软件编码工具	192
9.9 软件编码文件与复审	192
9.10 小结	193
思考与习题	193
第 10 章 软件测试	196
10.1 测试的基本概念	196
10.2 测试方法	199
10.2.1 静态分析技术	199
10.2.2 动态测试技术	201
10.3 单元测试	202
10.4 组装测试	203
10.5 确认测试	207
10.6 测试用例设计	209
10.6.1 白盒法	209
10.6.2 黑盒法	212
10.7 测试工具	215
10.7.1 静态分析工具	215
10.7.2 动态分析工具	216
10.7.3 测试数据生成工具	216
10.8 测试文件	216
10.8.1 测试文件综述	216
10.8.2 测试文件的内容要求	218
10.8.3 测试文件编制指南	222
10.9 测试的复审	222
10.10 排错技术	224
10.10.1 蛮干法	225
10.10.2 原因消除法	225

10.11 系统转换	228
10.12 小结	229
思考与习题	230
第 11 章 软件文件	236
11.1 目的和作用	236
11.2 软件生存周期与各种文件的编制	237
11.3 文件编制中考虑的因素	239
11.3.1 文件的读者	239
11.3.2 文件内容的重复性	240
11.3.3 文件内容的灵活性	241
11.4 各种文件的内容要求	243
11.4.1 用户手册	243
11.4.2 操作手册	244
11.4.3 模块开发卷宗	244
11.4.4 开发进度月报	245
11.4.5 项目开发总结报告	246
11.5 文件编制实施规定的实例	247
11.6 文件编制的质量要求	249
11.7 文件的管理和维护	251
11.7.1 文件的形成	251
11.7.2 文件的分类与标识	251
11.7.3 文件控制	251
11.7.4 文件的修改管理	252
11.8 文件编制工具	253
11.9 小结	253
思考与习题	254
第 12 章 软件维护	256
12.1 概述	256
12.1.1 软件维护工作的必要性	256
12.1.2 软件维护的内容	258
12.1.3 维护工作的过程	259
12.1.4 影响维护工作的因素	260
12.1.5 维护策略	261
12.2 软件的可维护性	267
12.2.1 可维护性的度量	267

12.2.2 提高软件的可维护性	273
12.3 软件维护的副作用	278
12.3.1 修改代码的副作用	279
12.3.2 修改数据的副作用	279
12.3.3 文件的副作用	280
12.4 软件维护工具	280
12.4.1 维护技术方面的工具	280
12.4.2 维护管理方面的工具	281
12.5 软件维护的管理	281
12.6 软件维护的文件	283
12.6.1 软件问题报告	283
12.6.2 软件修改报告	285
12.7 小结	287
思考与习题	288
第 13 章 软件工具和软件开发环境	290
13.1 软件开发工具	290
13.1.1 软件工具的基础	291
13.1.2 软件工具的作用	291
13.1.3 软件工具的通用性	291
13.1.4 软件工具的分类	292
13.2 软件开发环境	292
13.2.1 软件开发环境的类型	293
13.2.2 软件开发环境的构成	293
13.2.3 对软件开发环境的要求	294
13.3 计算机辅助软件工程	295
13.4 小结	297
思考与习题	297
第 14 章 软件的质量与评价	298
14.1 各类人员对质量的不同需求	298
14.2 软件质量度量模型	299
14.3 软件质量评价过程模型	301
14.3.1 质量需求的定义	301
14.3.2 评价的准备	301
14.3.3 评价过程	302
14.4 软件复杂性度量	302

14.5 软件可靠性评价	303
14.6 软件性能评价	304
14.7 软件运行评价	305
14.8 小结	305
思考与习题	307
第 15 章 软件管理	310
15.1 软件项目的特点与软件管理职能	310
15.1.1 软件项目的特点	310
15.1.2 软件管理的特殊困难	311
15.1.3 软件管理的主要职能	311
15.2 软件开发组织	312
15.2.1 软件设计小组的形式	312
15.2.2 对项目经理人员的要求	313
15.2.3 评价软件人员的条件	314
15.3 标准化管理	314
15.3.1 软件工程标准化	314
15.3.2 软件工程标准化的意义	315
15.3.3 软件工程国家标准	316
15.4 软件配置管理	318
15.4.1 软件配置	318
15.4.2 配置标识	319
15.4.3 里程碑与变更控制	320
15.4.4 配置状态登录	321
15.4.5 配置审查	321
15.5 软件的知识产权	322
15.5.1 软件知识产权的法律保护	322
15.5.2 计算机软件是著作权法保护的客体	324
15.5.3 软件著作权人享有的专有权利	326
15.5.4 软件著作权的登记	328
15.6 小结	328
思考与习题	329
第 16 章 软件工程的发展	331
16.1 软件工程的四个发展阶段	331
16.2 原型化方法	332
16.3 面向对象的分析与设计	333

16.3.1	面向对象的基本概念	334
16.3.2	面向对象分析	336
16.3.3	面向对象设计	337
16.4	统一的建模语言	337
16.4.1	UML 概述	338
16.4.2	使用 UML 的过程	339
16.5	软件工程过程	341
16.5.1	软件过程的定义	342
16.5.2	软件过程的基本成分	343
16.5.3	软件过程标准	343
16.5.4	软件过程工程的定义	345
16.5.5	软件过程工程和软件项目工程	345
16.5.6	软件过程周期	346
16.6	软件复用和构件技术	346
16.6.1	软件复用的概念	347
16.6.2	软件复用的过程	348
16.6.3	构件和体系结构	348
16.6.4	构件和构件系统	349
16.7	小结	349
	思考与习题	350
	参考文献	351



第 1 章

传统软件工程概述

随着计算机系统的迅速发展和应用范围的日益广泛，计算机软件的规模越来越大，其复杂程度也不断增加。进入 20 世纪 60 年代以来，人们开始逐渐认识到了确实存在着“软件危机”这样一个事实。例如：

- (1) 软件生产不能满足日益增长的需要。
- (2) 软件开发成本和开发进度的估计往往不准确。实际成本有时高出预计成本好几倍，预计完工的时间往往推迟几个月，甚至更长时间。
- (3) 软件开发人员和用户之间信息交流不充分，用户对完成的软件满意度很低。
- (4) 软件价格昂贵，软件成本在整个计算机系统中所占的比例急剧上升，软件已成为许多计算机系统中花钱最多的项目。
- (5) 软件质量难以保证，软件质量保证技术还没有真正应用到软件开发的全过程。
- (6) 软件可维护性差，程序中的错误很难改正，或者当硬件环境发生变化时，想要进行的适应性或完善性维护都是极其困难的。

导致这一系列问题的一个重要原因，就是由于软件的研制和维护本身是工程性的任务，但软件人员所采取的方式却未能工程化。

为了克服软件危机，促使人们开始考虑采用工程化方法和工程途径来研制和维护软件。20 世纪 60 年代末至 70 年代初开始，逐渐发展起一组总称为“软件工程”的技术。这些技术把软件作为一个工程产品来处理，它需要计划、分析、设计、实现、测试以及维护。软件工程学主要研究软件结构、软件设计方法论、软件工具、软件工程标准和规范，以及与软件工程有关的理论等。软件产业的逐步建立，也带动了软件工程这门学科的发展。同时，软件生产的系列化、产品化、工程化和标准化形成了软件产业的标志。

1.1 计算机系统的发展

软件工程和硬件工程都可以看成是一门更广义的学科——计算机系统工程的一部分。

用于计算机硬件的工程技术是由电子设计技术发展起来的，而且，在几十年的时间里已经达到了比较成熟的水平，硬件设计技术已经很好地确立。虽然制造方法仍在不断地改进，可靠性则已是一种可以期待的现实而不再是一种愿望了。

但是，计算机软件工程却还处于某种困境之中。在以计算机为基础的系统，软件已经取代硬件成为系统中设计起来最困难、最不容易成功（按时完成和不超过预计的成本），而且是最不易管理的部分。另一方面，随着以计算机为基础的系统在数量、复杂程度和应用范围上的不断增长，对软件的需求却仍然有增无减。

1.1.1 计算机系统及其软件的发展

软件发展的历史是与计算机系统发展紧密相关的。

在计算机系统发展的初期（20 世纪五六十年代），硬件经历了不断的变化，而软件则被多数人作为一种事后工作来看待，对它而言，几乎没有什么系统的方法可以遵循。在此期间，多数系统采用批处理工作方式。在这些系统中，硬件通常仅用来执行一个单一的程序，而这个程序又是为一个特定的应用目的而编制的。

在早期，当通用硬件已经成为平常的事情时，软件则仍然是为每一种用途分别设计的，它们的通用性是有限的。作为产品的软件（即为了销售给一个或多个用户而研制的程序）还处在它的幼年时期。大多数软件是由使用该软件的人或机构研制的，使软件带有较强烈的个人色彩：软件设计是在某个人的头脑中完成的一种隐含的过程。而且，往往没有软件说明书。

计算机系统发展的第二个时期跨越了从 20 世纪 60 年代中期到 70 年代中期这十年，这个时期以产品化软件的使用和“软件车间”的出现为特征。多道程序设计、多用户系统引入了人机对话的新概念。人机对话技术为计算机的应用开辟了新的天地，并使硬件和软件提高到更为精湛的新水平。实时系统能够在几毫秒内而不是几分钟内收集、分析和变换来自多个信息源的数据，进而控制处理过程并产生输出。联机辅助存储设备的发展导致了数据库管理系统的出现。人们开发软件是为了广泛销售。

随着以计算机为基础的系统日益增多，计算机软件库开始膨胀。本单位自己开发的软件和从外部购买来的软件产品堆集了几十万条源程序的语句。所有这些程序，即所有这些源语句，当检测出故障时就必须加以维护；当使用者的要求改变时，必须加以修改；或者要使软件在新购置的硬件上运行时，也必须修改。软件维护的开销越来越大，更坏的是，许多软件所带有的个人色彩使得它们实际上是不可维护的。“软件危机”开始了。

计算机系统发展的第三个时期从 20 世纪 70 年代初期开始。分布式系统（多个计算机、各机器并行执行和相互通信）极大地增加了以计算机为基础的系统的复杂性。由于微处理机和有关部件的功能越来越强而价格越来越低，因此，在最常用的计算机应用领域中，具有“嵌入智能”的产品取代了较大的计算机。此外，微处理机的出现也使得人们可能以极低的成本实现复杂的逻辑功能。

硬件的迅速发展已经超过了人们提供支持软件的能力。在第三个发展时期，软件危机日益严重。用于维护软件的费用占了数据处理预算的 50% 以上，而软件开发的生产率又跟不上新系统对软件需求的步伐。为了对付不断增长的软件危机，软件工程开始得到认真的对待，如表 1-1 所示。

表 1-1 计算机软件发展的三个时期

时 期 特 点	程序设计	程序系统	软件工程
软件所指	程序	程序及说明书	程序、文档及数据
主要程序设计语言	汇编及机器语言	高级语言	软件语言*
软件工作范围	程序编写	包括设计和测试	软件生存周期
需求者	程序设计者本人	少数用户	市场用户
开发软件的组织	个人	开发小组	开发小组及大、中型软件开发机构
软件规模	小型	中、小型	大、中、小型
决定质量的因素	个人程序设计技术	小组技术水平	管理水平
开发技术和手段	子程序、程序库	结构化程序设计	数据库、开发工具、开发环境、工程化开发方法、标准和规范、网络和分布式开发、面向对象技术、软件过程与过程改进
维护责任者	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、工作可靠性差	降价，速度、存储容量及工作可靠性有明显提高	向超高速、大容量、微型化及网络化方向发展
软件特征	完全不受重视	软件技术的发展不能满足需求，出现软件危机	开发技术有进步，但未获突破性进展，价格高，未完全摆脱软件危机

* 这里，软件语言包括需求定义语言、软件功能语言、软件设计语言、程序设计语言等。

如今，计算机系统的发展正在向第四个时期过渡，即从技术性应用向消费