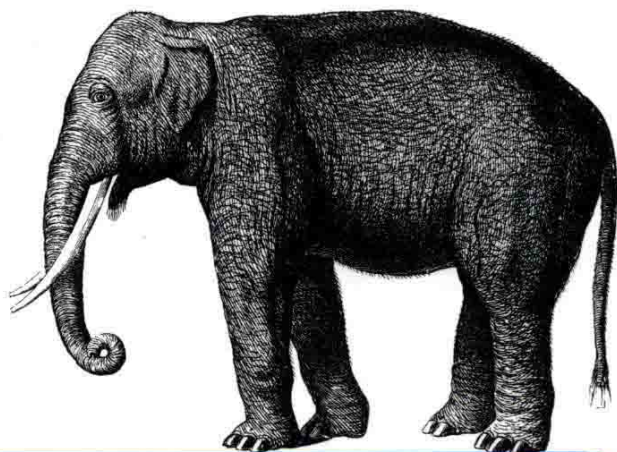




工业和信息化“十三五”
人才培养规划教材



软件工程

第3版

Software Engineering

陆惠恩 ◎ 主编

褚秋砚 ◎ 副主编



历经市场考验，第三次改版更新，深受读者喜爱

根据技术的发展增补全新内容，与时俱进

深入浅出、详略适当、理论和实际相结合、易于理解



中国工信出版集团

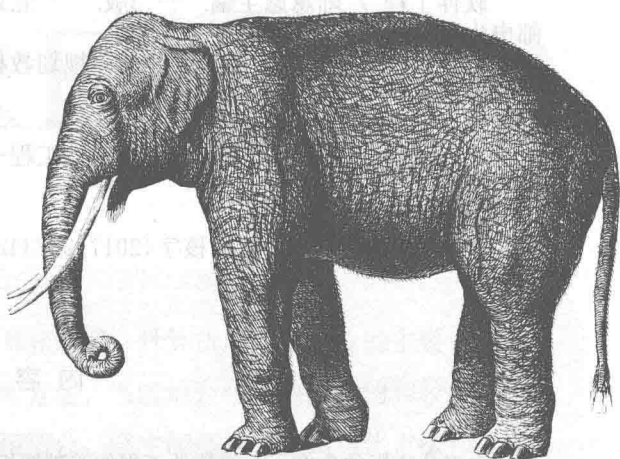


人民邮电出版社
POSTS & TELECOM PRESS



工业和信息化“十三五”
人才培养规划教材

第3版



软件工程

第3版

Software Engineering

陆惠恩 ◎ 主编

褚秋砚 ◎ 副主编

人民邮电出版社

北京

图书在版编目 (C I P) 数据

软件工程 / 陆惠恩主编. — 3版. — 北京: 人民邮电出版社, 2017.12
工业和信息化“十三五”人才培养规划教材
ISBN 978-7-115-46968-7

I. ①软… II. ①陆… 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(2017)第234266号

内 容 提 要

本书从实用的角度, 介绍软件工程的基础知识和软件工程技术方法。本书的编写力求做到结合实际、注重应用、便于教学, 体现内容的新颖性和系统性。

本书内容包括软件工程概述, 可行性研究与软件开发计划, 需求分析, 概要设计, 详细设计, 软件实现, 软件维护, 面向对象方法学与 UML, 面向对象技术与 UML 应用, WebApp 软件工程, 软件重用和再工程, 软件工程管理等。每章都有小结, 并配有适量的例题和习题, 有些例题贯穿于各章, 按软件工程的各个阶段分别就这些例题的实际处理方法、步骤等进行介绍, 可作为实践环节的样例, 有助于读者学习和掌握有关知识。

本书可作为高等院校“软件工程”课程的教材, 也可供软件工程师、软件项目管理人员和软件开发人员阅读参考。

-
- ◆ 主 编 陆惠恩
 - 副 主 编 褚秋砚
 - 责任编辑 左仲海
 - 责任印制 马振武
 - ◆ 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>
 - 北京市艺辉印刷有限公司印刷
 - ◆ 开本: 787×1092 1/16
 - 印张: 17.75 2017 年 12 月第 3 版
 - 字数: 408 千字 2017 年 12 月北京第 1 次印刷
-

定价: 49.80 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



第3版 前言

FOREWORD

“软件工程”是计算机科学与技术、计算机软件、计算机应用等专业的主要专业课程。本书讲述软件工程的基本概念、原理和方法，系统地介绍目前流行的和较成熟的软件工程技术，通过理论教学与实践教学相结合，学生能基本掌握结构化方法和面向对象方法等软件开发技术，对软件工程管理等内容有总体了解；能基本掌握如何系统、规范地开发和维护软件，如何规范地编写软件工程的文档资料，如何合理地安排软件开发与维护；能提高软件开发与维护的能力，以及软件开发过程的效率和质量。

针对应用型人才培养的要求，本书适当削减了理论叙述方面的内容，增加了一些易于理解的例题。本书第1版于2007年出版，得到了广大师生的认可。2012年出版的第2版在第1版的基础上增加了一些较新颖、较重要的内容以及应用实例，删减了一些较陈旧的内容。第2版出版至今已经累计重印了9次，深受学校师生的喜爱。本版为第3版，在第2版的基础上增加基于Web的系统和应用（WebApp）开发的相关内容等，同时删减较陈旧的内容。

本书的特点如下。

（1）深入浅出，详略得当，实用性强，易于理解。

（2）关于软件工程的基本概念、技术、方法等尽可能采用软件工程国家标准所介绍的术语和规范。

（3）引入软件工程的最新技术。例如，较详细地介绍统一建模语言（UML）及其在面向对象技术中的应用，介绍了WebApp的设计模式、需求分析、设计过程和测试方法等。

（4）每章给出小结，并配有适量精选的例题和习题。有些例题贯穿于各章，可作为实践环节的样例，有利于学生对内容的理解和掌握。

（5）书中介绍了软件工程各阶段所需的文档的规范，使学生在编写文档时有参考依据。

（6）附录中有部分习题解答和试题类型举例进行自测。

本课程适合在程序设计语言、数据库原理与应用、数据结构、计算机网络技术等专业课之后，毕业实习、毕业设计之前开设。

学习“软件工程”课程，建议理论学习为45~54学时，并适当安排实践环节。通过软件开发的实际训练来培养和提高学生开发、维护软件的能力；实践环节可要求学生完成一个难度适当的软件设计课题，可集中2~4周进行课程设计；也可在每章结束时安排实践环节，分阶段逐步完成课题。

本书由陆惠恩主编、褚秋砚副主编，参加本书编写工作的还有张成姝、曹辉、胡维荣、陆培恩、胡瑞明等。

书中难免存在不足之处，敬请读者批评指正。

编者

2017年5月

目录

CONTENTS

第 1 章 概述	1	2.2.5 软件工程开发计划实例	26
1.1 软件工程的产生	1	本章小结	31
1.1.1 软件生产的发展	1	习题 2	31
1.1.2 软件危机	2	第 3 章 需求分析	32
1.2 软件工程	4	3.1 需求分析的任务	32
1.2.1 软件工程定义	4	3.1.1 确定目标系统的具体要求	33
1.2.2 软件工程学的内容	4	3.1.2 建立目标系统的逻辑模型	35
1.2.3 软件工程的基本原理	7	3.2 结构化分析步骤	35
1.3 软件生命周期	7	3.2.1 进行调查研究	35
1.4 软件过程模型	9	3.2.2 分析和描述系统的逻辑模型	36
1.4.1 瀑布模型	9	3.2.3 需求分析的复审	37
1.4.2 快速原型模型	10	3.3 需求分析图形工具	38
1.4.3 增量模型	11	3.3.1 实体—关系图	38
1.4.4 喷泉模型	12	3.3.2 数据流图	40
1.4.5 统一过程	13	3.3.3 状态转换图	42
本章小结	14	3.3.4 IPO 图	43
习题 1	15	3.4 数据字典	44
第 2 章 可行性研究与软件开发计划	16	3.4.1 数据字典的内容	44
2.1 软件定义与可行性研究	16	3.4.2 数据字典使用的符号	45
2.1.1 软件定义	16	3.4.3 数据字典与图形工具	47
2.1.2 可行性研究	18	3.5 软件需求分析举例	48
2.2 软件工程开发计划的制定	19	3.5.1 系统管理	48
2.2.1 软件工程项目概述和 实施计划	20	3.5.2 商品信息管理	48
2.2.2 Gantt 图	20	3.5.3 商品供销存管理	49
2.2.3 工程网络图	21	3.5.4 商品销售数据流图	50
2.2.4 软件工程开发计划的 复审	25	3.5.5 数据字典	50
		3.6 需求分析文档	51

3.6.1 软件需求规格说明	51	5.2 用户界面设计	88
3.6.2 用户手册编写提示	52	5.2.1 用户界面设计问题	88
3.6.3 编写需求分析文档的步骤	52	5.2.2 用户界面设计过程	90
本章小结	53	5.2.3 用户界面设计的基本原则	90
习题3	53	5.2.4 用户界面设计指南	90
第4章 概要设计	55	5.3 数据代码设计	92
4.1 概要设计步骤	55	5.3.1 数据代码设计原则	93
4.1.1 软件结构设计	55	5.3.2 代码种类	93
4.1.2 数据结构设计及数据库设计	56	5.3.3 数据代码设计方法	95
4.1.3 系统接口设计	57	5.4 数据输入/输出设计	96
4.1.4 设计测试方案	57	5.4.1 输入设计	96
4.2 软件结构设计的基本原理	57	5.4.2 输出设计	96
4.2.1 模块与模块化	57	5.5 数据安全设计	97
4.2.2 模块的耦合和内聚	60	5.6 详细设计文档与复审	98
4.2.3 软件结构设计优化准则	63	5.6.1 详细设计说明书	98
4.3 软件结构设计的图形工具	64	5.6.2 操作手册编写提示	99
4.3.1 层次图	64	5.6.3 详细设计的复审	100
4.3.2 结构图	66	本章小结	100
4.4 概要设计方法	67	习题5	100
4.4.1 结构化方法	67	第6章 软件实现	102
4.4.2 面向数据结构设计方法	69	6.1 结构化程序设计	102
4.5 概要设计文档与复审	73	6.2 选择程序设计语言	103
4.5.1 概要设计说明书	73	6.3 程序设计风格	105
4.5.2 概要设计复审	74	6.4 程序设计质量评价	106
4.5.3 数据库设计说明书	74	6.5 程序设计文档	107
本章小结	75	6.6 软件测试目标和原则	107
习题4	75	6.6.1 软件测试目标	107
第5章 详细设计	77	6.6.2 软件测试原则	108
5.1 过程设计	77	6.7 软件测试方法	108
5.1.1 流程图	78	6.7.1 静态分析与动态测试	109
5.1.2 盒图	82	6.7.2 黑盒法与白盒法	109
5.1.3 PAD	84	6.8 软件测试步骤	110
5.1.4 判定表	85	6.8.1 模块测试	110
5.1.5 判定树	86	6.8.2 集成测试	110
5.1.6 过程设计语言	86	6.8.3 程序审查会和人工运行	111
		6.8.4 确认测试	112
		6.8.5 平行运行	113

6.9 设计测试方案	113	8.3.1 用例图	144
6.9.1 等价类划分法	113	8.3.2 类图和包	145
6.9.2 边界值分析法	114	8.3.3 对象图	150
6.9.3 错误推测法	115	8.3.4 状态图	150
6.9.4 逻辑覆盖法	115	8.3.5 顺序图	152
6.9.5 实用测试策略	118	8.3.6 活动图	152
6.10 软件调试、验证与确认	119	8.3.7 协作图	153
6.10.1 软件调试	119	8.3.8 构件图	154
6.10.2 软件验证	120	8.3.9 部署图	154
6.10.3 软件确认	121	本章小结	155
6.11 软件测试计划和分析报告	121	习题 8	156
本章小结	122	第 9 章 面向对象技术与 UML	
习题 6	123	应用	157
第 7 章 软件维护	127	9.1 面向对象分析	157
7.1 软件维护过程	127	9.1.1 面向对象分析过程	157
7.1.1 软件维护的种类	127	9.1.2 面向对象分析原则	158
7.1.2 软件维护的困难	128	9.2 建立对象模型	158
7.1.3 软件维护的实施	129	9.2.1 确定对象和类	159
7.1.4 软件维护的副作用	131	9.2.2 确定类的相互关系	159
7.2 软件的可维护性	131	9.2.3 划分主题	161
7.2.1 决定可维护性的因素	132	9.3 建立动态模型	163
7.2.2 可维护性的度量	132	9.3.1 编写脚本	164
7.2.3 提高软件的可维护性	133	9.3.2 设计用户界面	164
本章小结	134	9.3.3 画 UML 顺序图或活动图	165
习题 7	134	9.3.4 画状态转换图	165
第 8 章 面向对象方法学与 UML	136	9.4 建立功能模型	166
8.1 面向对象方法概述	136	9.5 面向对象设计	167
8.1.1 面向对象方法学的主要优点	137	9.5.1 系统设计	168
8.1.2 面向对象的概念	138	9.5.2 对象设计	171
8.2 UML 概述	140	9.5.3 面向对象设计的准则和启发式规则	172
8.2.1 UML 的发展	140	9.6 面向对象系统的实现	173
8.2.2 UML 设计目标和内容	141	9.6.1 选择程序设计语言	173
8.2.3 UML 的语义	142	9.6.2 面向对象程序设计	174
8.2.4 UML 的扩展机制	143	9.7 面向对象的测试	174
8.3 UML 图	144	9.7.1 面向对象测试策略	175

9.7.2	面向对象的测试步骤	177	10.5.4	模型—视图—控制器模式	210
9.8	UML 的应用	180	10.5.5	装饰者模式	211
9.8.1	UML 模型	181	10.6	WebApp 的设计	212
9.8.2	UML 视图	182	10.6.1	WebApp 的特点及应用类型	212
9.8.3	UML 使用准则	183	10.6.2	WebApp 的需求分析	213
9.8.4	UML 的应用领域	184	10.6.3	WebApp 需求分析过程	217
9.9	统一过程	184	10.6.4	WebApp 设计过程	219
9.9.1	RUP 的开发模式	185	10.6.5	WebApp 体系结构设计	223
9.9.2	RUP 的特点	186	10.6.6	WebApp 导航设计	224
9.9.3	RUP 的要素	186	10.7	WebApp 测试	226
9.10	Rational Rose	187	10.7.1	WebApp 测试过程概述	226
9.10.1	Rational Rose 简介	187	10.7.2	WebApp 内容测试	226
9.10.2	Rational Rose 的界面	189	10.7.3	WebApp 界面测试	227
9.10.3	Rational Rose 模型的4个视图	189	10.7.4	WebApp 导航测试	228
9.10.4	Rational Rose 的使用	191	10.7.5	WebApp 配置测试	228
本章小结		191	10.7.6	WebApp 安全性测试	229
习题 9		192	10.7.7	WebApp 性能测试	229
第 10 章	WebApp 软件工程	194	本章小结		230
10.1	网页软件的特性	194	习题 10		231
10.2	网络系统的层次结构	196	第 11 章	软件重用和再工程	232
10.2.1	二层 C/S 结构	196	11.1	可重用的软件成分	232
10.2.2	三层 C/S 结构	197	11.2	软件重用过程	233
10.2.3	四层 C/S 结构	199	11.2.1	软件重用过程模型	233
10.2.4	B/S 结构	199	11.2.2	开发可重用的软件构件	235
10.3	客户端层可用的技术	201	11.2.3	分类和检索软件构件	236
10.3.1	基本 HTML	201	11.2.4	软件重用环境	236
10.3.2	脚本语言	202	11.3	软件逆向工程	237
10.3.3	Applet	203	11.4	软件再工程	238
10.3.4	AJAX	203	本章小结		239
10.4	网络服务器层使用的技术	204	习题 11		240
10.4.1	Servlet	204			
10.4.2	JSP	205			
10.5	WebApp 的设计模式	206			
10.5.1	观察者模式	206			
10.5.2	组合模式	208			
10.5.3	工厂方法模式和策略模式	209			

第 12 章 软件工程管理	241	12.6.1 软件开发风险的分类	254
12.1 软件工程管理概述	241	12.6.2 软件开发风险的识别	255
12.2 软件规模估算	242	12.6.3 软件开发风险的预测	256
12.2.1 软件开发成本估算		12.6.4 处理软件开发风险的	
方法	242	策略	257
12.2.2 代码行技术和任务估算		12.7 软件工程标准与软件工程	
技术	243	文档	258
12.2.3 COCOMO2 模型	244	12.7.1 软件工程标准	258
12.2.4 程序环行复杂程度的		12.7.2 软件工程文档的编写	260
度量	246	12.8 CASE 技术	262
12.3 软件工程人员组织	247	本章小结	263
12.4 软件配置管理	249	习题 12	264
12.5 软件质量保证	252	附录 A 部分习题参考答案	265
12.5.1 软件质量的特性	252	附录 B 试题类型举例	271
12.5.2 软件质量保证措施	253	参考文献	273
12.6 软件开发风险管理	254		



第1章 概述

随着计算机应用的日益普遍，计算机软件的开发、维护工作越来越重要。如何以较低的成本开发出高质量的软件？如何开发出用户满意的软件？怎样使所开发的软件易于维护，以延长软件的使用时间？这些就是软件工程学研究的问题。软件工程学是指导计算机软件开发和维护工作的工程学科。

本章将介绍软件工程的产生、软件工程的定义及软件工程学的内容、软件工程的基本原理、软件生命周期、软件过程模型。

1.1 软件工程的产生

在计算机软件生产的发展过程中，产生了软件危机，为了解决软件危机，产生了软件工程。软件工程形成和发展的过程，实际上是软件生产的发展过程。本节介绍软件生产的发展过程，软件危机的表现形式、形成原因和解决途径。

1.1.1 软件生产的发展

自从 20 世纪 40 年代电子计算机问世以来，计算机软件即随着计算机硬件的发展而逐步发展。软件和硬件一起构成计算机系统，最初只有程序的概念，后来才出现软件的概念。软件生产的发展，大体经历了程序设计、软件、软件工程及第 4 代技术等阶段。

1. 程序设计时期

程序设计期指 20 世纪 40 年代中期到 60 年代中期，此期间，电子计算机价格昂贵，运算速度低，存储量小；计算机程序主要是描述计算任务的处理对象、处理规则和处理过程。早期的程序规模小，程序往往是个人设计、自己使用。在进行程序设计时，通常要注意如何节省存储单元，提高运算速度，除了程序清单之外，没有其他任何文档资料。

2. 软件=程序+文档时期

20 世纪 60 年代中期到 70 年代中期为软件时期，采用集成电路制造的电子计算机的运算速度和内存容量大大提高。随着程序数量的增加，人们把程序区分为系统程序和应用程序，并把它们称为软件。随着计算机技术的发展，计算机软件的应用范围也越来越广泛，当软件需求量大大增加后，许多用户去“软件作坊”购买软件。人们把软件视为产品，确定了软件生产的各个阶段必须完成的，为描述计算机程序的功能、设计和使用而编制的文字或图形资料，并把这些资料称为“文档”。软件是程序以及描述程序的功能、设计和使用的文档的总称，没有文档的软件，用户是无法使用的。

软件产品交付给用户使用之后，为了纠正错误或适应用户需求的改变，对软件进行的修改称为软件维护（Software Maintenance）。以前，由于软件开发过程中很少考虑到将来

的维护问题，软件维护费用以惊人的速度增长，不能及时满足用户要求，质量得不到保证。所谓的“软件危机”就是由此开始的。人们由此开始重视软件的“可维护性”问题，软件开发采用结构化程序设计技术，规定软件开发时必须编写各种需求规格说明书、设计说明书、用户手册等文档。

1968年，北大西洋公约组织（NATO）的计算机科学家在前联邦德国召开国际会议，讨论软件危机问题，正式提出了“软件工程（Software Engineering）”这一术语，从此一门新兴的工程学科诞生了。

3. 软件工程阶段

20世纪70年代中期到90年代为软件工程阶段，采用大规模集成电路制作的计算机，功能和性能不断提高，个人计算机已经成为大众化商品，计算机应用空前普及。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的趋势，软件产品供不应求，软件危机日益严重，为了维护软件要耗费大量的资金。美国当时的统计数据表明，对计算机软件的投资占计算机硬件总投资的70%，到1985年，软件成本大约占总成本的90%。为了对付不断增长的“软件危机”，软件工程学把软件作为一种产品进行批量生产，运用工程学的基本原理和方法来组织和管理软件生产，以保证软件产品的质量和提高软件生产率。软件生产使用数据库、软件开发工具等，软件开发技术有了很大的进步，开始采用工程化开发方法、标准和规范以及面向对象技术。

4. 软件发展第四阶段

计算机系统发展的第四阶段不再是针对单台计算机和计算机程序，而是面向计算机和软件的综合影响，复杂的操作系统控制的强大的桌面系统，连接局域网和互联网，高带宽的数字通信与先进的应用软件相互配合，产生了综合的效果。计算机体系结构从主机环境转变为分布式的客户机/服务器环境。

随着移动通信技术的快速发展和智能终端的普及，人们进入了移动互联网时代。移动通信是指利用无线通信技术，完成移动终端与移动终端之间或移动终端与固定终端之间的信息传送，即通信双方至少有一方处于运动中。移动互联网通过智能移动终端，采用移动通信的方式获取移动通信网络服务和互联网服务，包含终端、软件和应用三个层面。终端层包括智能手机、平板电脑、电子书、MID（Mobile Internet Device）等；软件包括操作系统、中间件、数据库和安全软件等；应用层包括休闲娱乐类、工具媒体类、商务财经类等不同应用与服务。移动互联网阶段的软件开发和维护工作有新的特点，光计算机、化学计算机、生物计算机和量子计算机等新一代计算机的研制发展，必将给软件工程技术带来一场革命。

1.1.2 软件危机

软件危机是指在计算机软件开发和维护时所遇到的一系列问题。软件危机主要包含如下两方面的问题。

- 如何开发软件以满足社会对软件日益增长的需求。
- 如何维护数量不断增长的已有软件。

1. 软件危机主要表现形式

(1) 软件的发展速度跟不上硬件的发展和用户的需求, 软件成本高。

硬件成本逐年下降, 软件应用日趋广泛, 软件产品“供不应求”, 与硬件相比, 软件成本越来越高。

(2) 软件的成本和开发进度不能预先估计, 用户不满意。

由于软件应用范围越来越广, 很多应用领域往往是软件开发人员所不熟悉的, 加之开发人员与用户之间信息交流不够, 导致软件产品不符合要求, 不能如期交付。因而, 软件开发成本和进度都与原计划相差太大, 引起用户不满。

(3) 软件产品质量差, 可靠性不能保证。

软件质量保证技术没有应用到软件开发的全过程, 导致软件产品质量问题频频发生。

(4) 软件产品可维护性差。

软件设计时不注意程序的可读性, 不重视可维护性, 程序中存在的错误很难改正。软件需求发生变化时, 维护相当困难。

(5) 软件没有合适的文档资料。

软件开发时文档资料不全或文档与软件不一致, 使用户不满意, 同时也会造成软件难以维护。

2. 软件危机产生的原因

(1) 软件危机产生的原因与软件的特点有关, 也与软件开发的方式、方法、技术和软件人员本身的素质有关。

(2) 软件是计算机系统逻辑部件, 软件产品往往规模庞大, 给软件的开发和维护带来客观的困难。

(3) 软件一般要使用 5~10 年, 在这段时间里, 很可能出现开发时没有预料到的问题。例如, 当系统运行的硬件、软件环境发生变化时, 或者系统需求发生变化时, 需要及时地维护软件, 使软件可以继续使用。

(4) 软件开发技术落后, 生产方式和开发工具落后。

(5) 软件人员忽视软件需求分析的重要性, 对软件的可维护性不重视, 也是造成软件危机的原因之一。

3. 解决软件危机的途径

目前, 计算机硬件的基本功能只是做简单的运算与逻辑判断, 主要还是适用于数值计算。随着计算机应用的日益广泛, 许多企事业单位的计算机 80% 以上用于管理方面。对于这样的非数值计算问题, 要设计计算机软件来进行处理, 因而使软件复杂、庞大。

要解决软件危机问题, 需要采用以下措施。

- 使用好的软件开发技术和方法。
- 使用好的软件开发工具, 提高软件生产率。
- 有良好的组织、严密的管理, 各方面人员相互配合共同完成任务。

为了解决软件危机, 既要有技术措施(好的方法和工具), 也要有组织管理措施。软件工程正是从技术和管理两方面来研究如何更好地开发和维护计算机软件的。

1.2 软件工程

本节介绍软件工程的定义、软件工程学的主要内容、软件工程过程及软件工程的基本原理。

1.2.1 软件工程定义

要知道什么是软件工程，首先要知道什么是软件。

1. 软件

软件是计算机程序及其有关的数据和文档。

计算机程序是能够完成预定功能和性能的可执行的指令序列；数据是程序能适当处理的信息，具有适当的数据结构；软件文档（Software Documentation）是开发、使用和维护程序所需要的图文资料。

软件文档是以人们可读的形式出现的技术数据和信息。文档用来描述或规定软件设计的细节，说明软件所具备的能力，介绍使用软件的操作过程。

著名软件工程专家 B.Boehm 指出：“软件是程序以及开发、使用和维护程序所需要的所有文档。”特别当软件成为商品时，文档更是必不可少的。没有文档仅有程序，是不能称为软件产品的。

2. 软件工程

软件工程是计算机科学的一个重要分支。国家标准《GB/T 11457—1995 软件工程技术语》对软件工程的定义是：“软件工程是软件开发、运行、维护和引退的系统方法。”

软件工程是指导计算机软件开发和维护的学科。软件工程采用工程的概念、原理、技术和方法来开发与维护软件。软件工程的目标是实现软件的优质高产，目的是在经费的预算范围内按期交付出用户满意的、质量合格的软件产品。

1.2.2 软件工程学的内容

软件工程学的主要内容是软件开发技术和软件工程管理。

软件开发技术包含软件工程方法学、软件工具和软件工程环境；软件工程管理包含软件工程经济学和软件管理学。

1. 软件工程方法学

最初，程序设计是个人进行的，只注意如何节省存储单元和提高运算速度。以后，兴起了结构化程序设计，人们采用结构化的方法来编写程序。结构化程序设计只采用顺序结构、条件分支结构和循环结构 3 种基本结构，用并且仅用这 3 种结构可以组成任何一个复杂的程序，软件工程的设计过程就是用这 3 种基本结构的有限次组合或嵌套来描述软件功能的实现算法。这样不仅改善了程序的清晰度，而且能提高软件的可靠性和软件生产率。

后来，人们逐步认识到编写程序仅是软件开发过程中的一个环节。典型的软件开发工作中，编写程序的工程量只占软件开发全部工作量的 10%~20%。软件开发工作应包括需求分析、软件设计、编写程序等几个阶段，于是形成了结构化方法、面向数据结构的 Jackson 方法、Warnier 方法等传统软件工程方法，20 世纪 80 年代得以广泛应用的是面向对象设计

方法。

软件工程方法学是编制软件的系统方法，它确定软件开发的各个阶段，规定每一阶段的活动、产品、验收的步骤和完成准则。

软件工程方法学有 3 个要素，包括方法、工具和过程。

方法：完成软件开发任务的技术方法。

工具：为方法的运用提供自动或半自动的软件支撑环境。

过程：规定了完成任务的工作阶段、工作内容、产品、验收的步骤和完成准则。

各种软件工程方法的适用范围不尽相同，目前使用得最广泛的软件工程方法学是传统方法学和面向对象方法学。

(1) 传统方法学。

软件工程传统方法学也称结构化方法，采用结构化技术，包括结构化分析、结构化设计和结构化程序设计，来完成软件开发任务。软件工程传统方法学把软件开发工作划分成若干个阶段，顺序完成各阶段的任务；每个阶段的开始和结束都有严格的标准；每个阶段结束时要进行严格的技术审查和管理复审。传统方法学先确定软件功能，再对功能进行分解，确定怎样开发软件，然后再实现软件功能。

(2) 面向对象方法学。

面向对象方法学把对象作为数据和在数据上的操作（服务）相结合的软件构件，用对象分解取代了传统方法的功能分解。把所有对象都划分成类，把若干个相关的类组织成具有层次结构的系统，下层的类继承上层的类所定义的服务。对象之间通过发送消息相互联系。使用面向对象方法开发软件时，可以重复使用对象和类等软件构件，从而降低了软件开发成本。

2. 软件工具

软件工具（Software Tools）是指为了支持计算机软件的开发和维护而研制的程序系统。使用软件工具的目的是提高软件设计的质量和生产效率，降低软件开发和维护的成本。

软件工具可用于软件开发的整个过程。软件开发人员在软件生产的各个阶段可根据不同的需要选用合适的工具。例如，需求分析工具使用类生成需求说明；设计阶段需要使用编辑程序、编译程序、连接程序，有的软件能自动生成程序等；在测试阶段可使用排错程序、跟踪程序、静态分析工具和监视工具等；软件维护阶段有版本管理、文档分析工具等；软件管理方面也有许多软件工具。目前，软件工具发展迅速，其目标是实现软件生产各阶段的自动化。

3. 软件工程过程

国际标准化组织（International Standards Organization, ISO）是世界性的标准化专门机构。ISO 9000 把软件工程过程（Software Engineering Process）定义为：“把输入转化为输出的一组彼此相关的资源和活动。”

软件工程过程是为了获得高质量软件所需要完成的一系列任务的框架，它规定了完成各项任务的工作步骤。

软件工程过程简称软件过程，是把用户要求转化为软件需求，把软件需求转化为设计，

用代码来实现设计，对代码进行测试，完成文档编制，并确认软件可以投入运行使用的全部过程。

软件过程定义了运用方法的顺序、应该交付的文档、开发软件的管理措施和各阶段任务完成的标志。

软件过程是软件工程方法学的3个要素（方法、工具和过程）之一。软件过程必须科学、合理，才能获得高质量的软件产品。

4. 软件工程环境

软件工程方法和软件工具是软件开发的两大支柱，它们之间密切相关。软件工程方法提出了明确的工作步骤和标准的文档格式，这是设计软件工具的基础，而软件工具的实现又将促进软件工程方法的推广和发展。

软件工程环境（Software Engineering Environment, SEE）是方法和工具的结合。软件工程环境的设计目标是提高软件生产率和改善软件质量。本书将在后续章节中介绍一些常用的软件工程方法、软件工具及软件工程环境。

计算机辅助软件工程（Computer Aided Software Engineering, CASE）是一组工具和方法的集合，可以辅助软件工程生命周期各阶段进行软件开发活动。CASE是多年来在软件工程管理、软件工程方法、软件工程环境和软件工具等方面研究和发展的产物。CASE吸收了CAD（计算机辅助设计）、软件工程、操作系统、数据库、网络和许多其他计算机领域的原理和技术。因此，CASE领域是一个应用、集成和综合的领域。其中，软件工具不是对任何软件工程方法的取代，而是对方法的辅助，它旨在提高软件工程的效率和软件产品的质量。

5. 软件工程管理

软件工程管理就是对软件工程各阶段的活动进行管理。软件工程管理的目的是能按预定的时间和费用，成功地生产出软件产品。软件工程管理的任务是有效地组织人员，按照适当的技术、方法，利用好的工具来完成预定的软件项目。

软件工程管理的内容包括软件费用管理、人员组织、工程计划管理、软件配置管理、软件开发风险管理等方面内容。

（1）费用管理。

一般来讲，开发一个软件是一种投资，人们总是期望将来获得较大的经济效益。从经济角度分析，开发一个软件系统是否划算，是软件使用方决定是否开发这个项目的依据。需要从软件开发成本、运行费用、经济效益等方面来估算整个系统的投资和回报情况。

软件开发成本主要包括开发人员的工资报酬、开发阶段的各项支出。软件运行费用取决于系统的操作费用和维护费用，其中操作费用包括操作人员的人数、工作时间、消耗的各类物资等开支。系统的经济效益是指因使用新系统而节省的费用和增加的收入。

由于运行费用和经济效益两者在软件的整个使用期内都存在，总的效益和软件使用时间的长短有关，所以，应合理地估算软件的寿命。在进行成本/效益分析时，一般假设软件使用期为5年。

(2) 人员组织。

软件开发不是个体劳动,需要各类人员协同配合,共同完成工程任务,因而应该有良好的组织和周密的管理。

(3) 工程计划管理。

软件工程计划是在软件开发的早期确定的。在软件工程计划实施过程中,需要时应对工程进度作适当的调整。在软件开发结束后应写出软件开发总结,以便今后能制订出更切实际的软件工程计划。

(4) 软件配置管理。

软件工程各阶段所产生的全部文档和软件本身构成软件配置。每完成一个软件工程步骤,都涉及软件配置,必须使软件配置始终保持其精确性。软件配置管理就是在系统的整个开发、运行和维护阶段内控制软件配置的状态和变动,验证配置项的完全性和正确性。

(5) 软件开发风险管理。

软件开发总会存在某些风险,对付风险应该采取主动的策略。早在技术工作开始之前就应该启动风险管理活动,标识出潜在的风险,评估它们出现的概率和影响,并且按重要性把风险排序;然后制订计划来管理风险。风险管理的主要目标是预防风险,但并非所有风险都能预防。因此,软件开发人员还必须制定处理意外事件的计划,以便一旦风险变成现实,能以可控的和有效的方式做出反应。

1.2.3 软件工程的基本原理

著名软件工程专家 Boehm 结合有关专家和学者的意见,根据自己多年来开发软件的经验,提出了如下所述 7 条软件工程的基本原理。

(1) 用分阶段的生命周期计划进行严格的管理。

(2) 坚持进行阶段评审。

(3) 实行严格的产品控制。

(4) 采用现代程序设计技术。

(5) 软件工程结果应能清楚地审查。

(6) 开发小组的人员应该少而精。

(7) 承认不断改进软件工程实践的必要性。

Boehm 指出,遵循前 6 条基本原理,能够实现软件的工程化生产;按照第 7 条原理,不仅要积极主动地采纳新的软件技术,还要注意不断总结经验。通过学习本课程,读者将体会到软件工程基本原理的含义和作用。

1.3 软件生命周期

软件生命周期是软件工程学的一个重要概念。本节介绍什么是软件生命周期,在传统软件工程方法中如何将软件生命周期划分为若干个阶段以及各阶段的基本任务是什么。

1. 软件生命周期简介

软件生命周期 (Software Life Cycle) 是从设计软件产品开始,到产品不能使用为止的时间周期。软件生命周期通常包括软件计划阶段、需求分析阶段、设计阶段、实现阶段、