



“十二五”普通高等教育本科国家级规划教材



北京高等教育精品教材
BEIJING GAODENG JIAOYU JINGPIN JIAOCAI

全国优秀畅销书
全国高校出版社优秀畅销书

21世纪软件工程专业规划教材

软件工程导论 (第6版)

张海藩 牟永敏 编著

10010101000101

111010010010

10010101000101

111010010010

1000101

11101001

10010101000101

111010010010

清华大学出版社





“十二五”普通高等教育本科国家级规划教材



北京高等教育精品教材

BEIJING GAODENG JIAOYU JINGPIN JIAOCAI

全国优秀畅销书

全国高校出版社优秀畅销书

21世纪软件工程专业规划教材

软件工程导论 (第6版)

张海藩 牟永敏 编著

清华大学出版社

北京

内 容 简 介

本书的前5个版本累计销售达130万册,已成为软件工程领域的经典教材,先后荣获全国普通高等学校工科电子类专业优秀教材二等奖、一等奖,并被评为全国优秀畅销书(前10名)、全国高校出版社优秀畅销书、北京高等教育精品教材和“十二五”普通高等教育本科国家级规划教材。为了反映最近4年来软件工程的发展状况,作者对第5版作了精心修改,编写了第6版。

本书全面系统地讲述了软件工程的原理和典型的方法学,并介绍了软件项目的管理技术。本书正文共13章,第1章是概述,第2~8章顺序讲述软件生命周期各阶段的任务、过程、结构化方法和工具,第9~12章分别讲述面向对象方法学引论、面向对象分析、面向对象设计和面向对象实现,第13章介绍软件项目管理。附录讲述了用面向对象方法开发软件的过程,对读者深入理解软件工程学很有帮助,也是上机实习的好材料。

本书可作为高等院校“软件工程”课程的教材或教学参考书,也可供有一定实际经验的软件工作人员和需要开发应用软件的广大计算机用户阅读参考。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

软件工程导论 / 张海藩,牟永敏编著. --6版. --北京:清华大学出版社,2013

21世纪软件工程专业规划教材

ISBN 978-7-302-33098-1

I. ①软… II. ①张… ②牟… III. ①软件工程—高等学校—教材 IV. ①TP311

中国版本图书馆CIP数据核字(2013)第150343号

责任编辑:袁勤勇

封面设计:常雪影

责任校对:白蕾

责任印制:何芊

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦A座 邮 编:100084

社 总 机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 刷 者:清华大学印刷厂

装 订 者:三河市新茂装订有限公司

经 销:全国新华书店

开 本:185mm×260mm 印 张:23 字 数:514千字

版 次:1996年7月第1版 2013年8月第6版 印 次:2013年8月第1次印刷

印 数:1~5000

定 价:39.50元

产品编号:050164-01

第6版前言

INTRODUCTION

《软件工程导论》已经出版了5个版本，累计发行量达到130万册，颇受读者欢迎，先后被评为全国优秀畅销书（前10名）、全国高校出版社优秀畅销书和北京高等教育精品教材、“十二五”普通高等教育本科国家级规划教材。经过4年多的时间，这一学科有了不少新的发展，为了跟踪学科的发展方向，更好地为广大读者服务，作者根据几年来的教学实践和软件开发经验对第5版进行了认真系统的修订，编写出了第6版。

鉴于先进、适用的软件过程对提高软件生产率和确保软件产品质量有相当大的作用，第6版在保持原书结构及篇幅基本不变的前提下，主要考虑知识的更新换代，由牟永敏负责对书中面向过程部分的内容进行了适量删减，同时，为了加强软件工程的实践教学，增加了面向对象设计部分的内容，此外还对书中的一些具体内容作了适当修改。全书由张海藩统一定稿。

丁媛、刘梦婷、刘昂、李慧丽、张亚楠等同学对第6版增加的内容进行了测试，并提出了有益的建议，谨在此表示感谢。

编者

2013年5月

第5版前言

INTRODUCTION

本书第4版出版后，受到广大读者的热烈欢迎，先后被评为全国优秀畅销书（前10名）、全国高校出版社优秀畅销书和北京高等教育精品教材。为了反映最近4年来软件工程的发展状况，作者对原书内容作了认真修改，写出了第5版。

鉴于先进适用的软件过程对提高软件生产率和确保软件产品质量有相当大的作用，第5版在保持原书结构和篇幅基本不变的前提下，主要增加了目前比较流行的 Rational 统一过程、以极限编程为杰出代表的敏捷过程以及微软过程的介绍，此外还对书中的一些具体内容作了适当的增删或修改。

倪宁对第5版应增加的内容提出了有益的建议，谨在此向他表示感谢。

编者

2008年1月

第4版前言

PREFACE

光阴荏苒，本书第3版已经出版5年多了。在此期间软件工程又有了很大发展，为了跟踪学科发展方向，更好地为广大读者服务，编者对原书内容作了认真修改，写出了第4版。

在保持原书的结构和篇幅基本不变的前提下，第四版主要对原书内容作了下述修改：

(1) 删掉了一些较陈旧的或较次要的内容。删掉的内容主要有：Warnier 程序设计方法，程序设计语言概述，程序设计途径，日立预测法，自动测试工具，COCOMO 模型，估算成本的标准值法，软件管理工具。

(2) 增加了一些较新颖的或较重要的内容。增加的内容主要有：软件过程，与用户沟通获取需求的方法，形式化说明技术，逐步求精，人机界面设计，回归测试，控制结构测试，预防性维护与软件再工程，面向对象测试策略及设计测试用例的方法，COCOMO2 模型，能力成熟度模型 (CMM)。

(3) 用统一建模语言 (UML) 的概念与符号重新改写了讲述面向对象方法学的第9、10、11、12章和附录A。

此外，还对书中许多具体内容作了修改或更新，对文字叙述作了进一步的加工和润色。

与第4版配套出版的还有《软件工程导论学习辅导》，该书共分10章，涵盖了教材的主要内容。每章均由三部分组成：第一部分系统扼要地复习本知识单元的重点内容；第二部分给出了与本单元内容密切配合的习题；第三部分是习题解答，对典型题目还详细分析了解题思路。附录给出了三套模拟试题以及参考答案，可供读者在课程学习之后检验学习效果。

为便于教学，本书制作了电子教案。采用本书作为教材的教师，可以从清华大学出版社免费获取电子教案。联系方法请参阅本书后面的

“读者意见反馈卡”。

我的学生张劲松和张展新参与了附录 A 所述的 C++ 类库管理系统的设计和实现工作，张雯和张杰为本书出版做了许多具体工作。谨在此向他们表示感谢！

编者

2003年8月

目 录

CONTENTS

第 1 章 软件工程学概述	1
1.1 软件危机	1
1.1.1 软件危机的介绍	1
1.1.2 产生软件危机的原因	3
1.1.3 消除软件危机的途径	4
1.2 软件工程	5
1.2.1 软件工程的介绍	5
1.2.2 软件工程的基本原理	7
1.2.3 软件工程方法学	9
1.3 软件生命周期	11
1.4 软件过程	14
1.4.1 瀑布模型	15
1.4.2 快速原型模型	16
1.4.3 增量模型	17
1.4.4 螺旋模型	19
1.4.5 喷泉模型	21
1.4.6 Rational 统一过程	22
1.4.7 敏捷过程与极限编程	25
1.4.8 微软过程	29
1.5 小结	31
习题 1	32
第 2 章 可行性研究	35
2.1 可行性研究的任务	35
2.2 可行性研究过程	36
2.3 系统流程图	38

2.3.1	符号	38
2.3.2	例子	38
2.3.3	分层	40
2.4	数据流图	40
2.4.1	符号	40
2.4.2	例子	42
2.4.3	命名	44
2.4.4	用途	45
2.5	数据字典	47
2.5.1	数据字典的内容	47
2.5.2	定义数据的方法	47
2.5.3	数据字典的用途	48
2.5.4	数据字典的实现	49
2.6	成本/效益分析	49
2.6.1	成本估计	50
2.6.2	成本/效益分析的方法	51
2.7	小结	53
	习题2	53
第3章	需求分析	55
3.1	需求分析的任务	56
3.1.1	确定对系统的综合要求	56
3.1.2	分析系统的数据要求	57
3.1.3	导出系统的逻辑模型	58
3.1.4	修正系统开发计划	58
3.2	与用户沟通获取需求的方法	58
3.2.1	访谈	58
3.2.2	面向数据流自顶向下求精	59
3.2.3	简易的应用规格说明技术	59
3.2.4	快速建立软件原型	61
3.3	分析建模与规格说明	62
3.3.1	分析建模	62
3.3.2	软件需求规格说明	62
3.4	实体-联系图	62
3.4.1	数据对象	63
3.4.2	属性	63
3.4.3	联系	63
3.4.4	实体-联系图的符号	64

3.5	数据规范化	64
3.6	状态转换图	65
3.6.1	状态	65
3.6.2	事件	65
3.6.3	符号	66
3.6.4	例子	66
3.7	其他图形工具	67
3.7.1	层次方框图	68
3.7.2	Warnier图	68
3.7.3	IPO图	69
3.8	验证软件需求	70
3.8.1	从哪些方面验证软件需求的正确性	70
3.8.2	验证软件需求的方法	70
3.8.3	用于需求分析的软件工具	71
3.9	小结	72
	习题3	73
第4章	形式化说明技术	75
4.1	概述	75
4.1.1	非形式化方法的缺点	75
4.1.2	形式化方法的优点	76
4.1.3	应用形式化方法的准则	76
4.2	有穷状态机	77
4.2.1	概念	77
4.2.2	例子	79
4.2.3	评价	82
4.3	Petri网	82
4.3.1	概念	82
4.3.2	例子	84
4.4	Z语言	85
4.4.1	简介	85
4.4.2	评价	88
4.5	小结	88
	习题4	89
第5章	总体设计	91
5.1	设计过程	91
5.2	设计原理	94

5.2.1	模块化	94
5.2.2	抽象	95
5.2.3	逐步求精	95
5.2.4	信息隐藏和局部化	96
5.2.5	模块独立	97
5.3	启发规则	99
5.4	描绘软件结构的图形工具	102
5.4.1	层次图和 HIPO 图	102
5.4.2	结构图	103
5.5	面向数据流的设计方法	104
5.5.1	概念	104
5.5.2	变换分析	105
5.5.3	事务分析	111
5.5.4	设计优化	112
5.6	小结	113
	习题 5	114
第 6 章	详细设计	117
6.1	结构程序设计	117
6.2	人机界面设计	119
6.2.1	设计问题	119
6.2.2	设计过程	121
6.2.3	人机界面设计指南	122
6.3	过程设计的工具	124
6.3.1	程序流程图	124
6.3.2	盒图	125
6.3.3	PAD 图	126
6.3.4	判定表	127
6.3.5	判定树	128
6.3.6	过程设计语言	128
6.4	面向数据结构的设计方法	129
6.4.1	Jackson 图	130
6.4.2	改进的 Jackson 图	131
6.4.3	Jackson 方法	132
6.5	程序复杂程度的定量度量	136
6.5.1	McCabe 方法	137
6.5.2	Halstead 方法	139
6.6	小结	140

习题 6	140
第 7 章 实现	145
7.1 编码	146
7.1.1 选择程序设计语言	146
7.1.2 编码风格	147
7.2 软件测试基础	149
7.2.1 软件测试的目标	150
7.2.2 软件测试准则	150
7.2.3 测试方法	151
7.2.4 测试步骤	151
7.2.5 测试阶段的信息流	152
7.3 单元测试	153
7.3.1 测试重点	153
7.3.2 代码审查	154
7.3.3 计算机测试	155
7.4 集成测试	156
7.4.1 自顶向下集成	157
7.4.2 自底向上集成	158
7.4.3 不同集成测试策略的比较	159
7.4.4 回归测试	160
7.5 确认测试	160
7.5.1 确认测试的范围	160
7.5.2 软件配置复查	161
7.5.3 Alpha 和 Beta 测试	161
7.6 白盒测试技术	162
7.6.1 逻辑覆盖	162
7.6.2 控制结构测试	165
7.7 黑盒测试技术	171
7.7.1 等价划分	172
7.7.2 边界值分析	175
7.7.3 错误推测	175
7.8 调试	176
7.8.1 调试过程	176
7.8.2 调试途径	178
7.9 软件可靠性	179
7.9.1 基本概念	179
7.9.2 估算平均无故障时间的方法	180

7.10 小结	182
习题7	183
第8章 维护	189
8.1 软件维护的定义	189
8.2 软件维护的特点	190
8.2.1 结构化维护与非结构化维护差别巨大	190
8.2.2 维护的代价高昂	190
8.2.3 维护的问题很多	191
8.3 软件维护过程	192
8.4 软件的可维护性	194
8.4.1 决定软件可维护性的因素	194
8.4.2 文档	195
8.4.3 可维护性复审	196
8.5 预防性维护	197
8.6 软件再工程过程	198
8.7 小结	200
习题8	201
第9章 面向对象方法学引论	203
9.1 面向对象方法学概述	203
9.1.1 面向对象方法学的要点	203
9.1.2 面向对象方法学的优点	205
9.2 面向对象的概念	209
9.2.1 对象	209
9.2.2 其他概念	211
9.3 面向对象建模	215
9.4 对象模型	216
9.4.1 类图的基本符号	217
9.4.2 表示关系的符号	218
9.5 动态模型	223
9.6 功能模型	224
9.6.1 用例图	224
9.6.2 用例建模	227
9.7 3种模型之间的关系	228
9.8 小结	229
习题9	229

第 10 章 面向对象分析	231
10.1 面向对象分析的基本过程	231
10.1.1 概述	231
10.1.2 3 个子模型与 5 个层次	232
10.2 需求陈述	233
10.2.1 书写要点	233
10.2.2 例子	234
10.3 建立对象模型	235
10.3.1 确定类与对象	236
10.3.2 确定关联	238
10.3.3 划分主题	241
10.3.4 确定属性	241
10.3.5 识别继承关系	244
10.3.6 反复修改	244
10.4 建立动态模型	247
10.4.1 编写脚本	247
10.4.2 设想用户界面	248
10.4.3 画事件跟踪图	249
10.4.4 画状态图	250
10.4.5 审查动态模型	251
10.5 建立功能模型	253
10.5.1 画出基本系统模型图	253
10.5.2 画出功能级数据流图	254
10.5.3 描述处理框功能	254
10.6 定义服务	255
10.7 小结	256
习题 10	256
第 11 章 面向对象设计	259
11.1 面向对象设计的准则	259
11.2 启发规则	261
11.3 软件重用	263
11.3.1 概述	263
11.3.2 类构件	265
11.3.3 软件重用的效益	266
11.4 系统分解	267
11.5 设计问题域子系统	270
11.6 设计人机交互子系统	273

11.7	设计任务管理子系统	275
11.8	设计数据管理子系统	277
11.8.1	选择数据存储管理模式	277
11.8.2	设计数据管理子系统	278
11.8.3	例子	280
11.9	设计类中的服务	280
11.9.1	确定类中应有的服务	280
11.9.2	设计实现服务的方法	281
11.10	设计关联	282
11.11	设计优化	283
11.11.1	确定优先级	283
11.11.2	提高效率的几项技术	284
11.11.3	调整继承关系	285
11.12	小结	287
	习题 11	288
第 12 章	面向对象实现	289
12.1	程序设计语言	289
12.1.1	面向对象语言的优点	289
12.1.2	面向对象语言的技术特点	290
12.1.3	选择面向对象语言	294
12.2	程序设计风格	294
12.2.1	提高可重用性	295
12.2.2	提高可扩充性	297
12.2.3	提高健壮性	297
12.3	测试策略	298
12.3.1	面向对象的单元测试	298
12.3.2	面向对象的集成测试	299
12.3.3	面向对象的确认测试	299
12.4	设计测试用例	299
12.4.1	测试类的方法	300
12.4.2	集成测试方法	301
12.5	小结	303
	习题 12	304
第 13 章	软件项目管理	305
13.1	估算软件规模	305
13.1.1	代码行技术	305

13.1.2	功能点技术	306
13.2	工作量估算	308
13.2.1	静态单变量模型	308
13.2.2	动态多变量模型	308
13.2.3	COCOMO2 模型	309
13.3	进度计划	312
13.3.1	估算开发时间	312
13.3.2	Gantt 图	314
13.3.3	工程网络	315
13.3.4	估算工程进度	316
13.3.5	关键路径	318
13.3.6	机动时间	318
13.4	人员组织	320
13.4.1	民主制程序员组	320
13.4.2	主程序员组	321
13.4.3	现代程序员组	322
13.5	质量保证	324
13.5.1	软件质量	324
13.5.2	软件质量保证措施	326
13.6	软件配置管理	328
13.6.1	软件配置	329
13.6.2	软件配置管理过程	329
13.7	能力成熟度模型	331
13.8	小结	334
	习题 13	335
附录 A C++ 类库管理系统的分析与设计		337
A.1	面向对象分析	337
A.1.1	需求	337
A.1.2	建立对象模型	338
A.2	面向对象设计	339
A.2.1	设计类库结构	339
A.2.2	设计问题域子系统	340
A.2.3	设计人机交互子系统	341
A.2.4	设计其他类	344
参考文献		347

软件工程概述

迄今为止,计算机系统已经经历了 4 个不同的发展阶段,但是,人们仍然没有彻底摆脱“软件危机”的困扰,软件已经成为限制计算机系统发展的瓶颈。

为了更有效地开发与维护软件,软件工作者在 20 世纪 60 年代后期开始认真研究消除软件危机的途径,从而逐渐形成了一门新兴的工程学科——计算机软件工程(通常简称为“软件工程”)。

1.1 软件危机

在计算机系统发展的早期时代(20 世纪 60 年代中期以前),通用硬件相当普遍,软件却是为每个具体应用而专门编写的。这时的软件通常是规模较小的程序,编写者和使用者往往是同一个(或同一组)人。这种个体化的软件环境,使得软件设计通常是在人们头脑中正在进行的一个隐含的过程,除了程序清单之外,没有其他文档资料保存下来。

从 20 世纪 60 年代中期到 70 年代中期是计算机系统发展的第二个时期,这个时期的一个重要特征是出现了“软件作坊”,广泛使用产品软件。但是,软件作坊基本上仍然沿用早期形成的个体化软件开发方法。随着计算机应用的日益普及,软件数量急剧膨胀。在程序运行时发现的错误必须设法改正;用户有了新的需求时必须相应地修改程序;硬件或操作系统更新时,通常需要修改程序以适应新的环境。上述种种软件维护工作,以令人吃惊的比例耗费资源。更严重的是,许多程序的个体化特性使得它们最终成为不可维护的。“软件危机”就这样开始出现了! 1968 年北大西洋公约组织的计算机科学家在西德召开国际会议,讨论软件危机问题,在这次会议上正式提出并使用了“软件工程”这个名词,一门新兴的工程学科就此诞生了。

1.1.1 软件危机的介绍

软件危机是指在计算机软件的开发和维护过程中所遇到的一系列严重问题。这些问题绝不仅仅是不能正常运行的软件才具有的,实际上,几