



国家示范性软件学院系列教材

本书为教师
配有
电子教案

现代 软件工程

张家浩 编著
东南大学

以现代软件企业开发管理需要
为出发点的
软件工程教学新视界

- 理论：突出现代软件工程的新观点
- 完整：全面覆盖IEEE SWEBOK2004
- 综合：注重最佳实践的导向与解决方案的平衡
- 案例：真实、实际，有参考借鉴价值

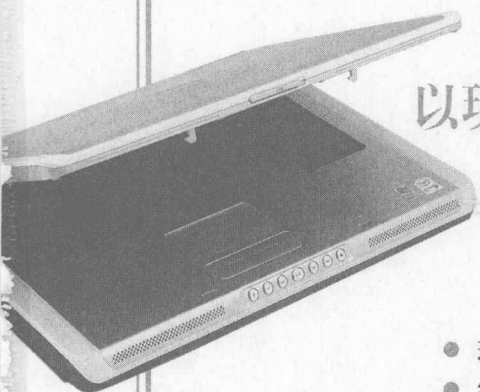


机械工业出版社
China Machine Press

本书为计算机专业、软件工程等专业必修课程教材，也可供从事计算机工作的工程技术人员参考。
国家示范性软件学院系列教材
本书为计算机专业、软件工程等专业必修课程教材，也可供从事计算机工作的工程技术人员参考。
本书为计算机专业、软件工程等专业必修课程教材，也可供从事计算机工作的工程技术人员参考。

现代 软件工程

张家浩 编著
东南大学



以现代软件企业开发管理需要
为出发点的
软件工程教学新视界

- 理论：突出现代软件工程的新观点
- 完整：全面覆盖IEEE SWEBOK2004
- 综合：注重最佳实践的导向与解决方案的平衡
- 案例：真实、实际，有参考借鉴价值



机械工业出版社
China Machine Press

本书按照软件开发生命周期,讲述软件工程的各个知识领域内容,系统地介绍了软件工程的概念、原理、方法和技术。

全书共9章,内容为现代软件工程导论、从市场和产品的角度理解软件开发、软件项目的规划管理、现代软件工程的需求工程、软件体系结构与系统概要设计、软件系统的构造与实现、软件质量管理、软件实施过程与管理、软件工程改进。其中,需求工程、体系结构与系统概要设计为开发过程的重点,项目管理、质量管理是支撑过程的重点。

本书内容新颖,讲述力求理论联系实际、深入浅出、循序渐进。全书共收录了20多个实际的案例分析,涉及金融、电信等领域。每章还配有课堂作业与项目实践及点评,方便老师和学生使用。

本书主要用作国家示范性软件学院软件工程专业的教材,也可作为计算机及相关专业软件工程课程的教学用书,或作为从事软件开发的科技人员的参考书、培训教材等。

版权所有,侵权必究。

本书法律顾问 北京市展达律师事务所

图书在版编目(CIP)数据

现代软件工程/张家浩编著. —北京:机械工业出版社, 2008.11

(国家示范性软件学院系列教材)

ISBN 978-7-111-25352-5

I. 现… II. 张… 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(2008)第158366号

机械工业出版社(北京市西城区百万庄大街22号 邮政编码 100037)

责任编辑:姚 蕾

北京慧美印刷有限公司印刷·新华书店北京发行所发行

2009年1月第1版第1次印刷

184mm×260mm·24印张

标准书号:ISBN 978-7-11-25352-5

定价:45.00元

凡购本书,如有倒页、脱页、缺页,由本社发行部调换

本社购书热线:(010) 68326294

前言

软件工程作为一门学科,从20世纪70年代以来,始终处在发展变化和探索前进之中。随着我国国民经济的起飞,软件产业逐步成为支撑国民经济发展的重要产业部门。国家成立示范性软件学院,也正是意识到传统计算机科学与技术专业学生培养的模式并不能适应现代企业发展的需要,这就要求软件学院从教学体制、教学内容、教学方式等方面做更深入的改革。但是,实事求是地讲,虽然很多软件学院为此进行了艰苦的努力,也取得了一定的进步,但大的格局没有发生根本的改变,很多基本要素没有发生质的变化。软件学院的教学和实践内容、教学组织形式和学科体系建设,都还处在“大拼盘”阶段。而从大的方面讲,我们现在的教育体系如何与国际接轨、与社会发展需求接轨,并不是引进几个人、请一些外教就能解决的。当然,这已经超出了我们这门软件工程课程要讨论的范围。

我在国内的IT企业待了20多年,也曾出国进修深造,过去多年的绝大部分时间是在软件公司从事开发和技术管理工作。因为一个偶然的机会,我受聘为东南大学软件学院讲授《软件项目管理》、《现代软件工程》等课程,期望利用自己的专业特长和企业背景,教书育人,使我所教授的学生在学习和掌握理论知识的同时,更多地了解现代软件企业的实践,毕业后更好地适应新的岗位,发挥出自己的潜力。

我在跟朋友聊天的时候,他们都建议我把这些年的经历写出来,特别是这20多年从事软件开发和管理的体会。同时,我也看到上海林锐博士的《软工思想》以及《我的大学十年》大受学生们的欢迎,很受启发和鼓舞,所以就萌发了编著本书的想法。在机械工业出版社华章公司的促成下,几经努力,终于成书。

本书的内容和想法如下。

一、教学目标和指导思想

软件学院的软件工程专业培养目标是培养为软件企业培养“软件生产”所需要的工程技术和专业管理人才。因此,“软件工程”作为软件工程专业核心课程,应在理论—技术—工程—管理的链条上,明确以工程为核心,以技术和管理为半径,建立教学和实践体系。这是本课程的基本指导思想。

美国电气与电子工程师协会等编制的《软件工程知识体系指南》2004版(简称:SWEBOK 2004)概括了现代软件工程知识体系的11个领域,特别是后6个领域,成为软件工程的重要组成部分,是现代软件工程的核心内容。本书将明确以SWEBOK 2004为基本蓝本,学生通过学习本书,能在已经获得的软件基础知识和语言编程技能的基础上,逐步开始建立将软件开发作为一个生产过程的概念,并初步学习和掌握软件工程的需求分析、系统设计和实现、软件测试等专门的工程技术方法,学习和了解软件产品管理、项目管理、质量管理、过程控制等基本过程管理知识。在同步的项目实践的配合下,亲身完成一个项目开发过程,感受作为一个开发者同时也是过程控制和管理者所必须注意的各个环节要点。这是本课程的基本教学目标。

因此,根据这个目标,我们应该在讲授现代软件工程相关知识点的同时,花更多的精力和时间,教会学生如何理解和把握软件开发过程。本课程内容和项目实践安排全面以IEEE SWEBOK 2004为参照,强调11个知识领域的重要性、完整性和协同性。同时,教师应根据学生的学习阶段,学生对相关管理知识的感知能力及程度,科学地安排相应领域知识点的出现顺序,使学生在课堂教学和项

目实践的适当环节上,恰到好处地体会和获得相应的知识,并取得自身真实的感受和经验。

为了达到这样的目标要求,本课程在注意相关知识的衔接,考虑学生的接受能力,适应学校的师资、教学管理和项目实践条件限制等因素的情况下,对教学内容、形式和考核方法,进行综合设计和改进。本书略去了大量理论和概念论述(这些部分已经被很多相应教材不断重复),而以案例分析代之,并重点突出软件过程的产品和项目管理。

二、先修课程与核心课程

在软件学院本科低年级(1、2年级)阶段,应基本完成相关的基础课程。这部分课程主要包括:计算机系统导论、数据结构与算法、数据库与网络开发技术、面向对象的程序设计(C++与Java)以及以学习系统设计基本思想方法为目的的操作系统、编译理论等课程。在基础课程中,建议教师压缩偏向具体物理层(计算机硬件结构等)、算法层(模糊数学、计算方法)的部分,也不过多地展开计算机网络原理、设计、安全等内容。相关部分可安排在项目活动中,由学生根据需要自行补充所需知识。

从本科高年级(3年级)开始,应进行软件工程的核心课程教学,核心课程是:软件工程概论、软件需求工程与UML、软件体系结构与系统设计,软件测试技术、软件过程与管理(配置、质量、项目、过程管理)、软件职业素质和实践教育。

在研究生阶段,只讲授核心课程部分,但更多地强调软件工程管理。基础部分作为研究生入学考试的考察点,应基本具备,否则,将无法继续进行深入的学习。

三、各章内容

本教材分为9章,各章配有课堂作业和项目实践及点评、参考书目等。

各章内容安排如下:

第1章:现代软件工程导论;

第2章:从市场和产品的角度理解软件开发;

第3章:软件项目的规划管理;

第4章:现代软件工程的需求工程;

第5章:软件体系结构与系统概要设计;

第6章:软件系统的构造与实现;

第7章:软件质量管理;

第8章:软件实施过程与管理;

第9章:软件过程改进。

四、教材使用建议

由于软件工程是软件学院的专业方向,因此,在教学时间安排上,本科生的核心课程时间至少应安排1个学年(2个学期)共128学时、研究生的核心课程时间可安排1个学期共64学时。

(1) 两种课程组织模式

考虑师资、实践条件等因素,第一种组织模式是:只开《现代软件工程》一门主课程,时间为1个学年,按软件开发生命周期,完整讲授软件工程的11个知识领域内容。其中,需求工程、体系结构设计为开发过程的重点,项目管理、质量管理是支撑过程的重点,配合一个大型的项目实践。同时可根据学生的情况和课时需要,适当增加面向对象设计、需求工程与UML、软件体系结构、软件测试技术等辅助课程或相关的讲座、选修课程。讲座或选修课程在时间、内容上应与主课程的内容衔接,与实践项目过程紧密配合。这种模式的好处是课程紧凑、内容衔接连贯性好,控制比较容易;缺点是缺乏相关技术和工具的深入了解和应用,还需要利用其他时间进行弥补或学生自学进行补充。这对于有学习兴趣和积极性的学生,具有更大的挑战性,而对投入相对不足的学生,则可能成为逃避的借口。

另一种组织模式是：分别开设软件工程概论、软件需求工程与UML、软件体系结构与系统设计、软件测试技术、软件过程与管理（配置、质量、项目、过程管理）等课程。这种模式的优点是知识面安排全面、周到；但缺点是需要在教学内容、进度和实践项目组织上，进行总体协调，否则容易搞成分散、重复甚至相互冲突的课程。在目前学校的管理模式下，要避免前面提到的问题，估计是很困难的，由此，学生的负担也会增大而产生负面效果。

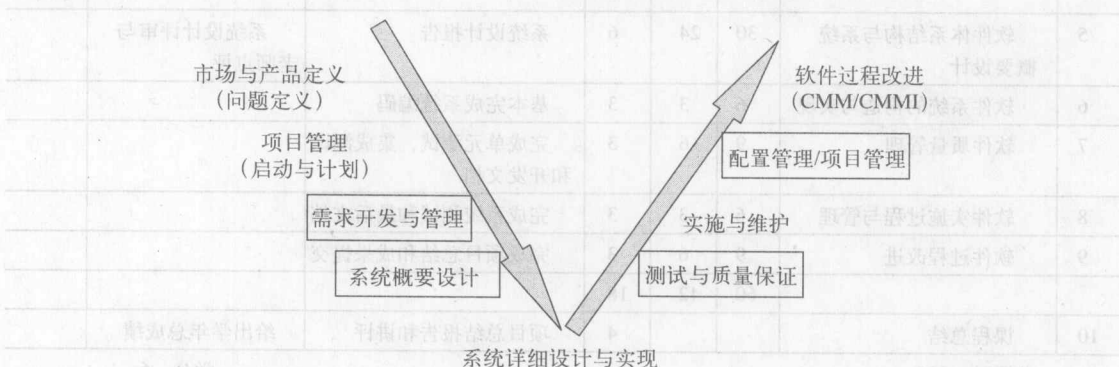
(2) 如何使用本教材

本教材是为第一种模式设计的，因为有相当一部分院校可能因为师资、项目实践等方面的原因，没有条件采用第二种模式。

在第一种模式下，本书作为主要教材。

在第二种模式下，本书可作为软件工程概论（开启）和软件过程管理（总结）两门课程的主要教材。

下图中加框的部分是本教材的重点，可根据实际情况需要采用并进行裁减。



五、教学形式和考核方式

在教学形式上，本课程力争实现以项目实践为中心的校内和企业实践教学，实际项目实践与知识传授相结合，课程与项目同步；逐步减少单纯以课堂、书本教学为主，以抽象概念的死记硬背为主的教学模式。

在教学成果考核方式上，可针对具体目标和成果考察要求进行。软件工程专业的本科生教育围绕大型软件开发过程中的工程方法、关键技术和相关工具展开，重点在学习软件开发技术的实际应用和理解、体会软件开发过程上。考察的方法可以一个大型项目实践过程（阶段）和项目成果的评审、答辩为依据。作为研究生教育，则重点应放在学习和掌握软件工程的工程方法和控制、支撑和辅助技术上，学习具体软件开发工程的管理。考察的方法是针对项目（可以是已有的项目，也可以是正在进行中的项目）开发过程的过程控制来进行各环节评审、控制和管理工作程序并控制成果。

六、各章建议课时安排

依据本教材，建议授课及项目实践的课时安排如下：

课程要求总学时数为128学时，包括项目实践与阶段评审、老师点评指导。

第一学期安排讲授前4章，其中课堂讲授时间为48学时，学生项目报告与评审/点评时间为12学时，机动4学时，总学时数为64学时，3个学分。学期成绩主要以学生项目实践活动——完成需求分析和评审为主，结合平时成绩，作为学期期末成绩。

第二学期安排讲授后5章，其中课堂讲授时间为42学时，学生项目实践与评审/点评时间为18学时，最后课程总结4学时，总学时数为64学时，3个学分。学期成绩主要以学生项目实践活动——完成项目成果（测试结果）和项目评审为主，结合平时成绩，作为学期期末成绩。

各章的核心内容、教学要求、教学形式、课时安排和结果检查，将在各章开始时给出。下表

内容为各章内容安排、课时数及配套实践活动内容及考核形式，仅作参考。

章节	课程内容	总学时	授课学时	实践学时	实践内容	考核形式	备注
1	现代软件工程导论	6	6		企业参观考察		
2	从市场和产品的角度理解软件开发	12	9	3	市场与产品分析	分析报告与老师点评	
3	软件项目的规划管理	12	9	3	项目选题与可行性分析、项目计划报告	课题论证报告与老师点评	
4	现代软件工程的需求过程	30	24	6	项目需求分析报告	需求评审与老师点评	
		60	48	12		给出学期总成绩	机动4学时
5	软件体系结构与系统概要设计	30	24	6	系统设计报告	系统设计评审与老师点评	
6	软件系统的构造与实现	6	3	3	基本完成系统编码		
7	软件质量管理	9	6	3	完成单元测试、集成测试和开发文档		
8	软件实施过程与管理	6	3	3	完成系统测试和最后文档		
9	软件过程改进	9	6	3	完成项目总结和成果提交		
		60	42	18			
10	课程总结			4	项目总结报告和讲评	给出学年总成绩	
总课时：128		学分：6					

七、致谢

真诚感谢东南大学软件学院的方宁生副院长及学院的各位老师，给我机会将自己20多年软件工程开发和软件项目管理积累的知识及心得体会，传授给新一代软件事业的后来人（包括我自己的儿子，他也是我的学生，并修完了我的这门课程，提出了很多非常不留情面的意见和建议）。

特别感谢机械工业出版社华章分社的各位编辑，由于他们的不断督促和鞭策，使我能按时完成这部书稿。

最后，我也要感谢我的家人对我的支持和帮助。

希望本书能对软件后来人，有所裨益。希望能对我的同行，有所帮助。

张家浩

2008年改于南京百合果园

目 录

前言	1
第1章 现代软件工程导论	1
1.1 软件、软件过程与现代软件工程	2
1.1.1 软件工程：对软件的再认识	2
1.1.2 软件危机：留给软件人长久的困惑	3
1.1.3 生命周期：30年前的初衷与设想	5
1.1.4 四大过程：走出危机的希望与前景	7
1.2 软件工程与软件工程知识体系	9
1.2.1 软件工程生命周期模型的特点	9
1.2.2 过程模型的典范：统一软件过程RUP	12
1.2.3 现代软件工程发展的主要技术特点	13
1.2.4 现代软件工程知识体系SWEBOK	17
1.3 软件企业的现代软件工程实践	18
1.3.1 软件生产过程的流程与工艺特性	18
1.3.2 软件生产过程的支撑与支持特性	20
1.3.3 软件生产过程的综合与协同特性	21
1.3.4 现代软件工程的最佳实践	22
1.4 市场需求与软件工程教学	22
1.4.1 IT职场素质需求	22
1.4.2 市场需求对软件人培养的要求	23
1.5 课堂作业与项目实践点评	23
1.5.1 课堂作业	23
1.5.2 项目实践	24
1.5.3 项目实践点评	24
1.6 本章参考文献	24
第2章 从市场和产品的角度理解软件开发	25
2.1 企业战略与市场分析	25
2.1.1 企业战略决定市场战略	25
2.1.2 市场环境的生物链与5种竞争因素分析	26
2.1.3 案例分析：某标案的竞争对手分析与策略选择	29
2.2 行业软件产品的市场营销特点	30
2.2.1 市场营销为产品研发和项目实施提供导向	30
2.2.2 方案销售是软件市场与产品营销的典型特点	31
2.2.3 案例分析：行业软件销售的过程、目标与责任	33
2.3 现代软件工程下的产品开发与管理	35
2.3.1 软件产品管理与产品经理的作用	35
2.3.2 产品经理面向市场的软件产品管理	37
2.3.3 产品经理面向开发的软件产品管理	40
2.3.4 现代软件工程概念下的软件研发管理	42
2.3.5 案例分析：采用产品线模式开发的案例	47
2.4 课堂作业与项目实践及点评	48
2.4.1 课堂作业	48
2.4.2 项目实践1：市场分析报告	48
2.4.3 市场分析报告点评	51
2.4.4 项目实践2：商业计划书	53
2.4.5 商业计划书点评	53
2.5 本章参考文献	55
第3章 软件项目的规划管理	56
3.1 项目与项目管理的基本概念	56
3.1.1 项目的概念与属性	56
3.1.2 项目的范畴要素	57
3.1.3 项目管理的概念	59
3.1.4 项目的生命周期与任务	60
3.1.5 项目的组织结构与项目经理	62
3.1.6 项目管理知识体系PMBOK	65
3.1.7 案例分析：项目管理成熟度水平的判断	67
3.2 软件项目的时间管理	70
3.2.1 项目的工作分解结构WBS	70
3.2.2 PMBOK的时间管理概念	73

3.2.3 PMBOK的时间管理过程	74	4.4.3 需求分析与UML的动态模型	136
3.2.4 软件生命周期模型与时间管理	80	4.4.4 需求分析与UML的体系结构模型	140
3.2.5 软件项目的规模、工作量和 进度估算	82	4.4.5 需求分析阶段的成果与需求管理 重点	141
3.2.6 案例分析: 软件项目规划 管理的关键与难点	91	4.4.6 案例分析: 平衡与细化需求定义	141
3.3 软件项目的成本管理	91	4.5 需求处理和验证过程与软件需求管理	145
3.3.1 PMEBOK的成本管理概念	92	4.5.1 需求处理过程与需求管理重点	145
3.3.2 PMEBOK的成本管理过程	93	4.5.2 需求验证过程与需求管理重点	149
3.3.3 案例分析: 软件项目成本 管理的实例操作	94	4.5.3 案例分析: 需求规格说明书的 实例参考	150
3.4 课堂作业与项目实践及点评	101	4.6 需求实现过程与软件需求管理	152
3.4.1 课堂作业	101	4.6.1 需求的实现过程与管理	153
3.4.2 项目实践	102	4.6.2 需求的变更控制与管理	157
3.4.3 项目实践点评	102	4.6.3 案例分析: 使用RequisitePro 需求管理工具	161
3.5 本章参考文献	103	4.7 课堂作业与项目实践及点评	163
第4章 现代软件工程的需求工程	104	4.7.1 课堂作业	163
4.1 从传统需求分析到现代需求过程	104	4.7.2 项目实践1: 目标和范围定义	163
4.1.1 需求与需求管理的概念	104	4.7.3 目标和范围定义点评	164
4.1.2 传统需求分析的局限	105	4.7.4 项目实践2: 需求分析报告	166
4.1.3 需求管理的问题与现代需求过程	107	4.7.5 需求分析报告点评	167
4.1.4 CMM的需求管理要求	110	4.7.6 期末考试方法建议	168
4.1.5 PMBOK的范围管理过程	110	4.8 本章参考文献	169
4.2 项目范围与软件需求管理	111	第5章 软件体系结构与系统概要设计	170
4.2.1 需求开发过程的四个阶段	111	5.1 体系结构的基本问题	170
4.2.2 需求获取与软件项目的范围定义	112	5.1.1 体系结构的基本概念	170
4.2.3 传统问题定义过程的方法	114	5.1.2 传统体系结构的思路方法	174
4.2.4 案例分析: 确定系统的目标与 假定约束	117	5.1.3 案例分析: RUP基于体系 结构的迭代	175
4.3 需求获取过程与软件需求管理	120	5.2 软件系统的框架与体系结构	176
4.3.1 需求获取阶段的目标与要求	120	5.2.1 现代体系结构模型的基本概念	176
4.3.2 需求获取与UML的业务模型	120	5.2.2 两种典型的体系结构模型分析	179
4.3.3 需求获取阶段的成果与需求 管理重点	124	5.2.3 应用系统体系结构的构成要素	183
4.3.4 需求用户确认与需求评审的误区	125	5.2.4 案例分析: 《电信综合营业 管理系统》的设计思路	184
4.3.5 案例分析: 分析执行者与用例, 确定系统范围	126	5.3 面向对象的设计模式	189
4.4 需求分析过程与软件需求管理	130	5.3.1 从体系结构到设计模式	189
4.4.1 需求分析阶段的目标与要求	130	5.3.2 创建型设计模式	190
4.4.2 需求分析与UML的静态模型	132	5.3.3 其他设计模式	195

5.3.4 案例分析: 创建游戏迷宫的设计模式应用	196	第7章 软件质量管理	252
5.4 MVC设计模式应用	201	7.1 软件质量的要素与度量	252
5.4.1 界面设计的可变需求	201	7.1.1 软件质量的概念	252
5.4.2 MVC的结构	202	7.1.2 软件质量的度量要素	255
5.4.3 MVC的实现	204	7.1.3 软件质量要素的度量	258
5.4.4 MVC的更进一步发展	205	7.1.4 软件质量度量的实施	261
5.4.5 MVC的变化与评价	206	7.1.5 案例分析: 软件测试活动的绩效度量	262
5.4.6 案例分析: Struts应用	207	7.2 软件测试活动的组织与管理	263
5.5 基于构件的现代软件工程	215	7.2.1 软件测试的概述	263
5.5.1 构件概念与构件的特性	215	7.2.2 软件测试的计划、组织与管理	267
5.5.2 三种主要构件模型与应用特点	216	7.2.3 案例分析: 测试错误分类及测试报告	271
5.5.3 基于构件的框架与体系结构设计	219	7.3 软件审查活动的组织与管理	273
5.5.4 案例分析: 软件无线电系统的体系结构	222	7.3.1 审查准备与审查过程	274
5.6 从需求分析到系统概要设计过程	227	7.3.2 审查内容与审查的关键点	276
5.6.1 系统概要设计的任务与要求	227	7.3.3 案例分析: 测试评审与分析报告	279
5.6.2 面向结构的系统设计方法	228	7.4 软件配置管理的方法与实践	280
5.6.3 案例分析: 产生“菜单”结构的系统结构分析	230	7.4.1 软件配置及其管理的概念	280
5.6.4 面向对象的系统设计方法	233	7.4.2 主要配置管理活动和流程	283
5.6.5 案例分析: 电梯控制系统的设计过程	236	7.4.3 配置管理下的版本管理	286
5.7 课堂作业与项目实践及点评	238	7.4.4 配置管理下的变更管理	293
5.7.1 课堂作业	238	7.4.5 配置状态监测、报告与评审	297
5.7.2 项目实践	239	7.4.6 基于配置管理的软件项目管理	298
5.7.3 项目实践点评	239	7.4.7 案例分析: 配置管理的实施	302
5.8 本章参考文献	240	7.5 课堂作业与项目实践及点评	307
第6章 软件系统的构造与实现	241	7.5.1 课堂作业	307
6.1 系统详细设计的目标与实现的任务	241	7.5.2 项目实践	308
6.1.1 概要设计与详细设计目标的区别	241	7.5.3 项目实践点评	308
6.1.2 面向对象的详细设计任务和原则	242	7.6 本章参考文献	308
6.1.3 面向对象的详细设计与实现	244	第8章 软件实施过程与管理	309
6.1.4 案例分析: Java程序编写格式的内部参考规范	247	8.1 软件的实施过程与控制	309
6.2 课堂作业与项目实践及点评	251	8.1.1 软件实施过程的概念	309
6.2.1 课堂作业	251	8.1.2 软件系统的现场实施过程	310
6.2.2 项目实践	251	8.1.3 案例分析: 一个现场项目的实施过程介绍	311
6.2.3 项目实践点评	251	8.2 软件的维护过程与控制	315
6.3 本章参考文献	251	8.2.1 软件维护的概念	315
		8.2.2 软件维护的过程	318

8.2.3 软件的可维护性管理	319	9.2.1 PSP的产生	356
8.3 软件项目的实施阶段管理	324	9.2.2 PSP与CMM	356
8.3.1 项目团队管理与激励	324	9.2.3 PSP的成熟度过程	357
8.3.2 项目沟通与协调	332	9.2.4 PSP的主要内容	358
8.3.3 软件项目风险管理	337	9.2.5 PSP与TSP过程	359
8.4 课堂作业与项目实践及点评	350	9.2.6 PSP、TSP的应用及效果	359
8.4.1 课堂作业	350	9.3 其他软件过程探讨	360
8.4.2 项目实践	350	9.3.1 极限编程 (XP) 与敏捷软件过程	360
8.4.3 项目实践点评	350	9.3.2 开源软件 (OSS) 的得与失	367
8.5 本章参考文献	350	9.4 课堂作业与项目实践及点评	370
第9章 软件过程改进	351	9.4.1 课堂作业	370
9.1 软件过程概念与过程改进的基本问题	351	9.4.2 项目实践	371
9.1.1 过程与软件过程的概念	351	9.4.3 项目实践点评	371
9.1.2 实施软件过程改进	353	9.4.4 期末考试	371
9.2 个人软件过程PSP	355	9.5 本章参考文献	372

第1章 现代软件工程导论

本章核心内容：通过软件工程发展历程的简单回顾，自然地导出高级软件经理与一般编码工程师在认知层次上的不同，即把软件作为一个“过程和工程”来看待和管理的基本思想方法以及现代软件工程在继承和发展传统软件工程的基础上所产生的新思想和新方法。

本章教学要求：通过本章的学习，使学生对软件工程的基本内容，建立一个初步的、整体性的认识，并为以后的学习（特别是本课程重点强调的：高级软件经理所处的认知层次、理论教学和项目实践相结合的学习方式、本课程项目实践与其他课程作业的区别、复杂程度和要求）做好必要的思想准备。

本章教学形式：（1）以课堂教学为主，以软件企业的参观、考察为辅；（2）根据学生实际人数、能力情况指定，或根据自愿组合或随机组合的方式，产生项目小组的划分。每组以5~8人为宜，并确定项目经理、技术经理、质量经理等角色；（3）为每组学生配备一名必要的辅导老师。辅导老师可以是有企业实践经验并具有实际动手能力的研究生，也可以直接是企业导师。

本章课时安排：本课程要求的总学时数为128学时，本章课堂授课时间应不少于6学时。在授课时间之外，另行安排企业参观、考察时间。

本章成果检查：（1）完成阶段课堂作业（附后），计入平时成绩；（2）针对项目小组组成情况与角色定位，任课老师或辅导老师应检查成果，必要时可进行调整。

有不少软件专业的同学，在没有学习软件工程这门课程之前，常常会有这样的认识，以为所谓学习软件，就是重点掌握好一二门计算机语言（如C++、Java），然后编一些程序，做过几个项目就可以了。他们认为只要编程能力强，成为所谓的“编程高手”，在软件公司里是所谓的“牛人”，薪金待遇、职位前途都不在话下。社会上的很多人，包括一些软件公司的老总，也都是这么看的。其实，这种观点是不正确，至少是不全面的。

作为软件学院的学生，与其他高校的计算机相关专业学生相比，在面对软件企业招聘和市场选择的时候，我们的优势在哪里？与他们之间的根本差别在哪里？什么是我们的核心竞争力？如果没有差别，或者没有太大差别的话，我们自身的价值在哪里？我们比其他院系学生所投入的多得多的学习成本的价值在哪里？可能在你填写高考志愿的时候，对此并不了解。

如果拿一个机械加工车间作为比喻的话，其他计算机专业学生学的是机械加具体体的车、钳、洗、刨，而软件工程专业学生学的是如何管理一个工件从第一个加工工序（通常把软件过程划分为：问题定义、需求分析、系统设计、编码调试、软件测试、运行维护6道工序）开始，直到最后一个加工工序，最终完成这个工件，成为合格的产品的全过程。软件工程专业培养的是，懂得软件生产各工位、各工序、各项工艺要求的软件生产过程的系统设计师、架构师、需求分析师、技术经理和软件项目经理。其他专业的学生可能成为一名优秀的八年级钳工，软件学院学生应该把自己定位为车间主任。在人才市场、在软件企业，我们的学生与其他计算机专业的学生的不同就在这里，这就是我们的核心竞争力。

所以，从现在开始，我们要换一个角度、换一种眼光，提升一下自己的思维层面。我们需要进入一个更高、更抽象、更复杂的领域——软件工程领域。我们面对的问题，将不再是如何用语言实现一个具体的算法，不再是如何设计好一个数据结构或数据库结构，也不再是仔细推敲网络协议或接口的细节。我们研究的内容，是把软件看成一个生产过程的完整的工程性问题。这就是实

现从软件到软件工程、从编程向系统设计、软件过程管理的转换。实现从编码工程师到高级软件经理的转换。

因此,从某种意义上讲,软件工程这门课是培养“高级软件经理”的一门课程。学生必须做好必要的心理、思想和知识准备,通过这门课程,打下坚实的基础,以准备有一天,最终实现从编码工程师到高级软件经理的提升和转换。

1.1 软件、软件过程与现代软件工程

管理理论把组织中的人分为三个层次:高层管理者、中层组织者、低层执行者。在任何组织结构中,这样的划分都具有一定的普遍适用性。处在不同层次的人的知识结构的构成是不相同的。在高层,管理者需要的是综合概括和判断能力;在中层,组织者需要的是沟通协同组织能力;而在低层,执行者需要的是专业技术执行能力。软件的系统设计师、架构师、需求分析师、技术经理和软件项目经理(简称为高级软件经理或软件经理)是处在中间层次的人,而不是低层的、只关注设计实现代码的编码工程师。因此,站在这个层面上,我们将首先重新审视我们已经熟悉的对象——软件。

1.1.1 软件工程:对软件的再认识

1.1.1.1 软件

什么是软件?当我还是一个编码工程师的时候,会回答:就是那些由程序设计语句(面向过程的或者面向对象的程序设计语句)构成的程序。如同置身于大森林之中,看到的除了树,还是树。今天,我们换一个视角,站在软件经理——软件工程师的角度来看,软件还可以是另一样东西。软件是下面三个方面的组合:

- 一个或多个计算机程序,其执行时能提供所期望的功能和性能。

- 一个或多个数据结构,这些结构使得程序能够完全操纵信息。

- 一个或多个文档,这些文档描述了程序分析、设计、实现、使用和维护过程的细节。

在这里,我们已经开始不是单纯地只看到计算机程序,还加进了数据结构和文档。为什么要考虑数据结构?为什么还需要有文档?因为从现在开始,我们已经不仅仅是一个编码工程师。作为软件经理,要站在系统(作为开发对象的系统和作为开发组织的系统)一级来考虑问题。

从软件工程的发展过程来看,实际上,仅就什么是“程序”本身而言,也经历了面向过程与数据、面向对象和面向构件三个不同的认识阶段:

- 面向过程的程序=算法+数据结构。

- 面向对象的程序=对象+消息。

- 面向构件的程序=构件+架构。

不同阶段对“程序”的不同认识,从根本上反映了软件生产者和组织者,在不同的生产能力水平和生产组织方式下,采用不同的生产技术和方法生产目标产品时,所呈现的对目标本身的认识差异。

20世纪50年代,人们认为软件就是程序。20世纪60年代,人们认为软件是程序+文档(分析、设计、测试、维护,但不包括管理文档)。到了20世纪70年代,人们认为软件是程序+文档+数据(初始化数据、测试数据、研发数据、运行数据、维护数据、工程数据、项目管理数据等)。而到了1984年,美国开始认识到软件管理是一个过程管理,在1991年出现了CMM1.0,1996年出现了UML。

从这个变化中可以看到,在软件开发的初始阶段,软件就是程序(最终结果),没有过程。或者说,过程在开发者的脑子里。随后,软件开发过程本身慢慢显现出来,因此产生了两种不同的软件:

①“软件制品”——开发过程中产生的各种中间软件(中间制品而非成品);②“软件产品”——最

后交付的软件成品。从这里, 我们开始感到, 软件——这个我们追逐的对象, 已经不再是软件经理的唯一目标, 我们开始更多地关注产生这个目标的过程。

1.1.1.2 软件过程

从软件程序, 到软件过程, 是一个认识的进步。什么是过程? 过程就是人们使用相应的方法、规程、技术、工具等将原材料(输入)转化为用户需要产品的活动。因此, 过程是为了达到给定目标而执行的一系列活动的有序集合, 包括: 工具、方法和人。

在不同的领域, 过程的概念可能有所差异。在软件过程的定义中, 比较经典的是SEI CMM的定义: 过程是用于生产以及软件进化的一系列的、活动、方法及实践。

过程的一般定义可以归纳为3个基本要素: 人、方法与规程、技术与工具。过程与产品存在因果关系。即: 好的过程有可能(并不一定)得到好的产品, 而不好的过程一定得不到好的产品。

因此, 软件过程是将用户的需求转化为有效的软件解决方案的一系列活动的集合, 包括定义的一系列软件开发中采用的规则和方法、人员的组织和培训以及开发的工具和设备。软件过程把这些要素科学地定义并合理地组织起来, 把软件技术、人员技能、管理水平和组织机制整合在一起, 以完成给定的项目目标。成功且成熟的软件过程被文档化后, 成为软件过程规范而沉淀下来, 成为软件组织宝贵的知识资产。许多软件组织无法正确定义和控制这一过程, 这恰恰是软件开发失败的原因。

1.1.1.3 软件工程

在20世纪60年代, 软件生产的基本特征是软件作坊——个体化生产。为了跟上硬件的发展速度、解决进度, 并实现成本可控制、质量可保证、系统可维护、过程可管理, 产生了把软件开发作为一个工程来管理的思想, 即软件工程的观念。

为摆脱软件危机, 北约(NATO)的科学委员会于1968年在前联邦德国召开的研讨会上, 首次提出了软件工程(Software Engineering)的概念, 其主要想法是把人类长期从事的其他领域各种工程所积累的经验行之有效的原理、概念、技术和方法(包括对硬件研发积累的经验)导入到软件开发中。

IEEE将“软件工程”定义为: “①应用系统化的、学科化的、量化的方法, 来开发、运行和维护软件, 即, 将工程应用到软件; ②对①中各种方法的研究”。

把这个定义分解开来理解, 就是:

1) 应用计算机科学、数学及管理科学等原理, 借鉴传统工程的原则和方法, 来创建软件, 从而达到提高质量、降低成本的目的。

2) 其中, 采用的方法包括:

- 计算机科学和数学用于构造模型、分析算法。
- 工程科学用于制定规范、明确风险、评估成本和确定权衡。
- 管理科学用于进度、资源、质量、成本的管理。

因此, 软件工程是计算机科学、工程和管理三个学科的综合。

从某种意义上说, 软件过程是软件组织长期改进活动的对象和成果(沉淀后, 成为组织宝贵的知识资产)。软件工程是活动, 软件过程是结果。软件工程包含软件过程: 软件工程除指明软件过程外, 还研究软件工程中应用的技术——技术方法和自动化工具、质量控制等。

1.1.2 软件危机: 留给软件人长久的困惑

1.1.2.1 软件的复杂度

20世纪80年代以后, 虽然软件的生产组织过程还处在很不令人满意的阶段, 但是, 现实需求对我们的要求却越来越高, 计算机软件所承担的作用也越来越大。这最终表现为开发出的软件的

复杂度越来越高。

图1-1是针对软件技术和软件开发过程管理两个维度的复杂度描述。我们以技术复杂度（纵坐标）和管理复杂度（横坐标）做一个四维分布。有关技术和管理复杂度的标志因素，图1-1上给出了简单的描述。作为度量的参照，图1-1的中心位置代表一个中等规模的软件项目的复杂度。中等规模的含义定义为：5~10个人、10~15个月的开发周期、3~5个外部界面。其他系统如图1-1所示。

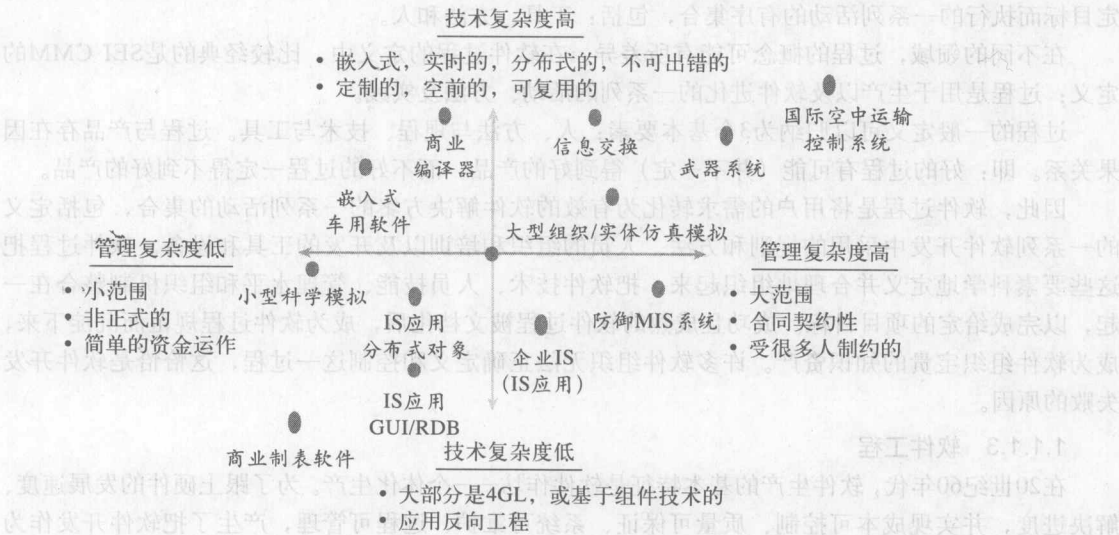


图1-1 软件系统的复杂度

从图1-1中我们可以看到，国际空中运输控制系统可能具有技术和管理的最高复杂度，而商业制表软件在技术和管理上均只有较低的复杂度。

1.1.2.2 “软件作坊”的软件危机

虽然软件的作用、重要性以及复杂度越来越大，但是软件生产手段自身的改进，始终没有大的提升，巨大的反差就构成了所谓“软件危机”。可以说，软件是始终伴随着“危机”发展的。在20世纪60年代，就开始提出“软件危机”的概念。所谓“软件危机”是指：软件的可靠性没有保障、维护费用不断上升、进度无法预测、成本增长无法控制、程序员无限度增加等，形成软件开发局面失控的状态。

在不同的时代，对软件危机的认识也不相同。

20世纪70年代（软件工程史前）的软件危机，表现为以下特征：

- 1) 对软件开发的进度和成本无法估计。
- 2) 用户对已经开发完成的软件的满意度非常低。
- 3) 软件质量无法保证。
- 4) 软件开发后的维护工作很难进行。
- 5) 软件通常没有合适的文档资料。
- 6) 软件成本在系统总成本中所占的比例越来越高。
- 7) 软件开发的生产率跟不上需求。

在那个时代，软件危机的例子很多，经常被拿出来作为典型的有：1962年美国水手I号因导航软件一个语句的语义错误，导致偏离航线，任务失败；阿波罗8号因计算机软件错误，造成存储器信息丢失；阿波罗14号在飞行的10天中，出现了18个软件错误；美国IBM公司的OS/360系统，花了几千人很多年的努力而失败等。

我们仔细看看软件产品发展的历史，再与人类社会其他产品发展的历史进行一个简单的比较，就可以看出，软件开发的历史具有一些独特的发展特性：

1) 与建筑技术、制造技术、计算机硬件技术不同，软件生产具有非常特殊的个体化、个性化和生产过程的不可复制性。

2) 软件开发的工具、技术手段获得了很大的进步，但开发方法、管理方法和水平并没有获得自动和同步的进步。

在这个年代，软件开发团队通常被视为“软件作坊”。软件开发已经出现了近半个世纪，但软件的“手工作坊”，到现在依然普遍存在。

什么是手工作坊，以下的描述非常经典：

1) 个人对所负责的“局部”负责，在这个局部内，是完全个性化和自由的，系统就是由几个这样的“局部”叠加（例如：通过菜单的形式）构成的。

2) 没有任何设计文档和可用于维护的资料，即生产过程是没有记录的。

3) 没有评审和独立的系统测试，即结果不依靠第三方确认和验证。

4) 进度、成本、质量是不可预测的。

而更要命的是，同处于计算机生物圈的计算机硬件，却有一个可怕的摩尔定律。根据摩尔定律：硬件成本每隔18个月就降低一半，例如：存储器每年降低40%、主机硬件的性价比每十年提高一个数量级。而这个定律直到现在也没有被打破。

可想而知，软件人应该是怎样的郁闷和压抑。谈到软件危机，我们不能不提到美国人弗雷德里克·布鲁克斯（Frederick P. Brooks JR.）和他的《人月神话》（The Mythical Man-Month）。

《人月神话》是一本畅销30年经久不衰、在业界具有深远影响的书。作者是美国IBM公司被认为是IBM System/360和OS/360之父、曾担任360系统项目经理的布鲁克斯博士。作为未来的软件人，你一定要找来这本书读一遍。

那么，现代软件工程是否找到了克服软件危机的方向，进展情况如何呢？

以面向对象技术为手段，以可重用软件构件化和体系架构为基础，以工业化生产方式和管理支撑体系为核心的软件新变革，即推进软件开发的构架平台化、组件业务化、编码自动化和管理工厂化为代表的“软件四化”，正发起攻克软件危机的一轮新的冲锋。软件进入工业化生产的时代即将到来，而这也正是我们现代软件工程所要讨论的核心内容。

1.1.3 生命周期：30年前的初衷与设想

1.1.3.1 早期软件工程管理的基本原则

1968年正式提出“软件工程”这一术语之后，软件工程的专家和学者围绕计算机科学、工程和管理三个方面做了很多研究，建立了早期关于软件工程管理的一些基本准则。简单回顾一下这个过程，从中可以看出早期软件工程的一些思路与出发点。其中影响比较大的是著名软件工程专家B.W.Boehm在1983年的一篇论文中提出的软件工程7条基本原理，其反映了软件工程应该关注和考虑的若干本质问题：

1) 用分阶段的生命周期计划严格管理。

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) 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。

1.1.3.2 软件工程的生命周期阶段

到20世纪70年代末，按照软件工程的思路，人们已经取得了大量的研究成果，形成了基本的方法，这一代的软件工程，称为“传统软件工程”。传统软件工程最主要的成果，就是所谓的软件生命周期的“瀑布模型”和相应生命周期的工作规范。从软件工程的生命周期划分及任务定义中，可以明显地看到上述软件工程基本原则的实际应用。

传统软件工程把软件的生命周期定义为6个阶段，即：问题定义与可行性研究、需求分析、软件设计、编码、测试、运行与维护。

(1) 问题定义与可行性研究阶段

在这个阶段，系统分析员通过对系统实际用户、使用管理部门、相关部门及人员进行的实际调查，搞清楚“问题”的背景、目的是什么。然后，据此提出关于“问题”的性质、工程目标、规模、相关联系等项目的基本情况，进行可行性分析，编制开发计划。

(2) 需求分析阶段

在这个阶段，通过问题识别、分析与综合、制订规格说明和评审等活动，达到以下目标，它们都是对“用户需求”进行更计算机专业化的“描述”与“转换”。

需求分析阶段的任务和目标包括：

- 确定对系统的综合要求。
- 分析系统的数据要求。
- 抽象并确立目标系统的逻辑模型。
- 编写需求规格说明书。

(3) 软件设计阶段又可细分为概要设计和详细设计两个阶段

在概要设计（总体设计）阶段，开发人员要回答需求分析中获得的系统目标如何去实现的问题。因此，概要设计阶段的目标是：

- 体现对需求的完整实现。
- 保证与需求的一致性。
- 能够达到向需求的反向可追踪。
- 关注对系统结构设计的逻辑性、合理性和可扩展性。

传统软件工程提出了很多概要设计方法，最主要的面向结构的概要设计方法包括：结构分析（SA）和结构设计（SD）等。

在详细设计阶段，是对概要设计进行细化，回答如何具体实现系统目标的问题。详细设计是面向具体程序编码，重点是编码规范。因此，详细设计阶段的要求是设计的规范性和为编码实现的可性。为此，开发了HIPO（层次图加输入/输出处理）、PDL（过程设计语言）等详细设计工具。

(4) 编码阶段

这个阶段的主要任务是写出符合规范的代码，并完成相应的调试。

(5) 测试阶段

测试包括：单元测试、集成测试、系统测试和验收测试等。

(6) 运行与维护阶段

运行是把开发完成的系统交付并达到正式使用的过程，通常也称为实施。维护包括：改正性维护、适应性维护、完善性维护和预防性维护等。

表1-1 简要地说明了生命周期各阶段的主要工作和交付成果。

表1-1 传统软件工程生命周期阶段的任务与成果

阶 段	关 键 任 务	交 付 成 果
问题定义	问题是什么	关于规模和目标的报告书
可行性研究	有可行的解吗	系统的高层逻辑模型：数据流图、成本/效益分析