

软件工程与 项目管理案例教程

主 编 刘新航

副主编 王振铎



北京大学出版社
PEKING UNIVERSITY PRESS

21 世纪全国高职高专计算机案例型规划教材

软件工程与项目管理案例教程

主 编 刘新航
副主编 王振铎
参 编 刘 平 崔 岩 王振辉
刘 宁 薛 茹 李 平



北京大学出版社
PEKING UNIVERSITY PRESS

内 容 简 介

本书根据高职教学的特点和要求编写。本书共分 13 章,介绍了软件危机与软件工程、计算机系统工程等基本概念,软件需求分析管理、概要设计、详细设计、软件编程、软件测试技术、软件维护等软件开发过程,面向对象系统分析与设计、软件项目评审、软件质量保证与软件配置管理、CMM 软件成熟度模型、软件工程标准与软件知识产权。

本书采用案例教学和启发式教学,激发学生学习的兴趣,提高学生动手能力。本书内容翔实、结构合理、实用性强、适用面广。每章后附有习题,以利于知识点的巩固。

本书可作为职业技术学院教材,也可作为其他高等职业学校、高等专科学校、中等职业学校、在职人员、本科学院、独立学院及各种社会培训机构的参考书。

图书在版编目(CIP)数据

软件工程与项目管理案例教程/刘新航主编. —北京:北京大学出版社, 2009.8

(21 世纪全国高职高专计算机案例型规划教材)

ISBN 978-7-301-15519-6

I. 软… II. 刘… III. ①软件工程—高等学校:技术学校—教材②软件开发—项目管理—高等学校:技术学校—教材 IV. TP311.5

中国版本图书馆 CIP 数据核字(2009)第 121165 号

书 名: 软件工程与项目管理案例教程

著作责任者: 刘新航 主编

策 划 编 辑: 李彦红

责 任 编 辑: 魏红梅

标 准 书 号: ISBN 978-7-301-15519-6/TP · 1038

出 版 者: 北京大学出版社

地 址: 北京市海淀区成府路 205 号 100871

网 址: <http://www.pup.cn> <http://www.pup6.com>

电 话: 邮购部 62752015 发行部 62750672 编辑部 62750667 出版部 62754962

电 子 邮 箱: pup_6@163.com

印 刷 者:

发 行 者: 北京大学出版社

经 销 者: 新华书店

787mm×1092mm 16 开本 18.25 印张 417 千字

2009 年 8 月第 1 版 2009 年 8 月第 1 次印刷

定 价: 28.00 元

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

举报电话: 010-62752024

电子邮箱: fd@pup.pku.edu.cn

21 世纪全国高职高专计算机案例型规划教材 专家编写指导委员会

主 任	刘瑞挺	南开大学
副主任	安志远	北华航天工业学院
	丁桂芝	天津职业大学
委 员	(按拼音顺序排名)	
	陈 平	马鞍山师范高等专科学校
	褚建立	邢台职业技术学院
	付忠勇	北京政法职业技术学院
	高爱国	淄博职业学院
	黄金波	辽宁工程技术大学职业技术学院
	李 纓	中华女子学院山东分院
	李文华	湖北仙桃职业技术学院
	李英兰	西北大学软件职业技术学院
	田启明	温州职业技术学院
	王成端	潍坊学院
	王凤华	唐山工业职业技术学院
	薛铁鹰	北京农业职业技术学院
	张怀中	湖北职业技术学院
	张秀玉	福建信息职业技术学院
	赵俊生	甘肃省合作民族师范高等专科学校
顾 问	马 力	微软(中国)公司 Office 软件资深教师
	王立军	教育部教育管理信息中心

信息技术的案例型教材建设

(代丛书序)

刘瑞挺/文

北京大学出版社第六事业部在 2005 年组织编写了两套计算机教材，一套是《21 世纪全国高职高专计算机系列实用规划教材》，截至 2008 年 6 月已经出版了 80 多种；另一套是《21 世纪全国应用型本科计算机系列实用规划教材》，至今已出版了 50 多种。这些教材出版后，在全国高校引起热烈反响，可谓初战告捷。这使北京大学出版社的计算机教材市场规模迅速扩大，编辑队伍茁壮成长，经济效益明显增强，与各类高校师生的关系更加密切。

2007 年 10 月北京大学出版社第六事业部在北京召开了“21 世纪全国高职高专计算机案例型教材建设和教学研讨会”，2008 年 1 月又在北京召开了“21 世纪全国应用型本科计算机案例型教材建设和教学研讨会”。这两次会议为编写案例型教材做了深入的探讨和具体的部署，制定了详细的编写目的、丛书特色、内容要求和风格规范。在内容上强调面向应用、能力驱动、精选案例、严把质量；在风格上力求文字精练、脉络清晰、图表明快、版式新颖。这两次会议吹响了提高教材质量第二战役的进军号。

案例型教材真能提高教学的质量吗？

是的。著名法国哲学家、数学家勒内·笛卡儿(Rene Descartes, 1596~1650)说得好：“由一个例子的考察，我们可以抽出一条规律。(From the consideration of an example we can form a rule.)”事实上，他发明的直角坐标系，正是通过生活实例得到的灵感。据说是在 1619 年夏天，笛卡儿因病住进医院。中午他躺在病床上苦苦思索一个数学问题时，忽然看到天花板上有一只苍蝇飞来飞去。当时天花板是用木条做成正方形的格子。笛卡儿发现，要说出这只苍蝇在天花板上的位置，只需说出苍蝇在天花板上的第几行和第几列。当苍蝇落在第四行、第五列的那个正方形时，可以用(4, 5)来表示这个位置……由此他联想到可用类似的办法来描述一个点在平面上的位置。他高兴地跳下床，喊着“我找到了，找到了”，然而不小心把国际象棋撒了一地。当他的目光落到棋盘上时，又兴奋地一拍大腿：“对，对，就是这个图”。笛卡儿锲而不舍的毅力，苦思冥想的钻研，使他开创了解析几何的新纪元。千百年来，代数与几何井水不犯河水。17 世纪后，数学突飞猛进的发展，在很大程度上归功于笛卡儿坐标系和解析几何学的创立。

这个故事，听起来与阿基米德在浴池洗澡而发现浮力原理，牛顿在苹果树下遇到苹果落到头上而发现万有引力定律，确有异曲同工之妙。这就证明，一个好的例子往往能激发灵感，由特殊到一般，联想出普遍的规律，即所谓的“一叶知秋”、“见微知著”的意思。

回顾计算机发明的历史，每一台机器、每一颗芯片、每一种操作系统、每一类编程语言、每一个算法、每一套软件、每一款外部设备，无不像闪光的珍珠串在一起。每个案例都闪烁着智慧的火花，是创新思想不竭的源泉。在计算机科学技术领域，这样的案例就像大海边上的贝壳，俯拾皆是。

事实上，案例研究(Case Study)是现代科学广泛使用的一种方法。Case 包含的意义很广，包括 Example 例子，Instance 事例、示例，Actual State 实际状况，Circumstance 情况、事件、境遇，甚至 Project 项目、工程等。

大家知道在计算机的科学术语中，很多是直接来自日常生活的。例如 Computer 一词早在 1646 年就出现于古代英文字典中，但当时它的意义不是“计算机”而是“计算工人”，即专门从事简单计算的工人。同样的，Printer 的意义当时也是“印刷工人”而不是“打印机”。正是由于这些“计算工人”和“印刷工人”常出现计算错误和印刷错误，才激发查尔斯·巴贝奇(Charles Babbage, 1791—1871)设计了差分机和分析机，这是最早的专用计算机和通用计算机。这位英国剑桥大学数学教授、机械设计专家、经济学家和哲学家是国际公认的“计算机之父”。

20 世纪 40 年代，人们还用 Calculator 表示计算机。到电子计算机出现后，才用 Computer 表示计算机。此外，硬件(Hardware)和软件(Software)来自销售人员，总线(Bus)就是公共汽车或大巴，故障和排除故障源自格瑞斯·霍普(Grace Hopper, 1906—1992)发现的“飞蛾子”(Bug)和“抓蛾子”或“抓虫子”(Debug)。其他如鼠标、菜单……不胜枚举。至于哲学家进餐问题、理发师睡觉问题更是操作系统文化中脍炙人口的经典。

以计算机为核心的信息技术，从一开始就与应用紧密结合。例如，ENIAC 用于弹道曲线的计算，ARPANET 用于资源共享以及核战争时的可靠通信。即使是非常抽象的图灵机模型，也受到“二战”时图灵博士破译纳粹密码工作的影响。

在信息技术中，既有许多成功的案例，也有不少失败的案例；既有先成功而后失败的案例，也有先失败而后成功的案例。好好研究它们的成功经验和失败教训，对于编写案例型教材有重要的意义。

我国正在实现中华民族的伟大复兴，教育是民族振兴的基石。改革开放 30 年来，我国高等教育在数量上、规模上已有相当大的发展。当前的重要任务是提高培养人才的质量，为此，培养模式必须从学科知识的灌输转变为素质与能力的培养。应当指出，大学课堂在高新技术的武装下，利用 PPT 进行的“高速灌输”、“翻页宣科”有愈演愈烈的趋势，我们不能容忍用“技术”绑架教学，而是让教学工作乘信息技术的东风自由地飞翔。

本系列教材的编写，以学生就业所需的专业知识和操作技能为着眼点，在适度的基础知识与理论体系覆盖下，突出应用型、技能型教学的实用性和可操作性，强化案例教学。本套教材将会融入大量最新的示例、实例以及操作性较强的案例，力求提高教材的趣味性和实用性，打破传统教材自身知识框架的封闭性，强化实际操作的训练，使本系列教材做到“教师易教，学生乐学，技能实用”。有了广阔的应用背景，再造计算机案例型教材就有了基础。

我相信北京大学出版社在全国各地高校教师的积极支持下，精心设计，严格把关，一定能够建设出一批符合计算机应用型人才培养模式的、以案例型为创新点和兴奋点的精品教材，并且通过一体化设计实现多种媒体有机结合的立体化教材，为各门计算机课程配齐电子教案、学习指导、习题解答、课程设计等辅导资料。让我们用锲而不舍的毅力，勤奋好学的钻研，向着共同的目标努力吧！

刘瑞挺教授 本系列教材编写指导委员会主任、全国高等院校计算机基础教育研究会副会长、中国计算机学会普及工作委员会顾问、教育部考试中心全国计算机应用技术证书考试委员会副主任、全国计算机等级考试顾问。曾任教育部理科计算机科学教学指导委员会委员、中国计算机学会教育培训委员会副主任。PC Magazine《个人电脑》总编辑、CHIP《新电脑》总顾问、清华大学《计算机教育》总策划。

前 言

软件工程是研究如何用工程化的思想方式有效地管理软件开发，以较低成本开发出高质量的软件的一门学科。软件工程已经成为异常活跃的研究领域，在软件开发实践中发挥着重要作用。人们已经意识到，在软件项目开发中若不遵守软件工程的原则、思想、方法，必然会导致软件项目的失败。所以软件工程技术对软件专业人员来说是必须掌握的技术。

本书主要有以下特点。

(1) 在保证学科体系完整的基础上，不过度强调基础理论的深度和难度，坚持“够用为度”的原则。

(2) 采用“任务驱动”的编写方式，引入案例式教学，在相关章节中引入案例，把软件工程过程、工具、方法讲懂讲透，强调理论和实践结合，注重技能培养。

(3) 教材内容生动活泼，力求改变一般软件工程教材学生学习起来枯燥的情况。

(4) 加入了一些软件项目管理、CMM、标准化和知识产权等内容，拓展学生管理技能。

本书共 13 章，建议理论课时 48 课时。第 1 章软件危机与软件工程(4 课时)，第 2 章计算机系统工程(2 课时)，第 3 章软件需求分析管理(4 课时)，第 4 章概要设计(4 课时)，第 5 章详细设计(4 课时)，第 6 章软件编程(2 课时)，第 7 章软件测试技术(6 课时)，第 8 章软件维护(4 课时)，第 9 章面向对象系统分析与设计(6 课时)，第 10 章软件项目评审(2 课时)，第 11 章软件质量保证与软件配置管理(4 课时)，第 12 章 CMM 软件成熟度模型(4 课时)，第 13 章软件工程标准与软件知识产权(2 课时)。

除了理论教学外，建议安排实验课。让学生掌握相关 CASE 工具及文档写作，例如 Project、Visio、PowerDesigner、Rose 等 CASE 工具，项目开发计划(GB 8567—1988)、软件需求说明书(GB 8567—1988)、概要设计说明书(GB 8567—1988)、详细设计说明书(GB 8567—1988)、数据库设计说明书(GB 8567—1988)、测试计划(GB 8567—1988)等软件工程文档。CASE 工具及文档实验内容教师可根据实际情况进行筛选。

本书由刘新航任主编，王振铎任副主编。刘新航负责全书的策划、修改、补充、统稿工作。各章编写分工如下：刘平编写第 1 章和第 12 章，崔岩编写第 2 章和第 11 章，王振辉编写第 3 章和第 6 章，刘宁编写第 4 章和第 9 章，王振铎编写第 5 章和第 7 章，薛茹编写第 8 章，李平编写第 10 章，刘新航编写第 13 章。

由于时间仓促，作者水平所限，书中难免有疏漏和不足之处，恳请各位读者批评指正。

编 者

2009 年 5 月

目 录

第 1 章 软件危机与软件工程.....1	
1.1 软件危机2	
1.1.1 软件危机的表现.....2	
1.1.2 软件危机的原因.....4	
1.1.3 解决软件危机的途径.....5	
1.2 软件工程概述6	
1.2.1 软件工程的定义.....6	
1.2.2 软件工程的范畴.....6	
1.2.3 软件开发的几个模型.....8	
1.2.4 软件生命周期.....14	
1.2.5 软件工程的目标.....15	
1.3 软件项目管理概述16	
1.3.1 软件项目管理的特点.....16	
1.3.2 软件项目管理的目标.....18	
1.3.3 影响软件项目成功的因素.....19	
1.4 软件项目失败与成功案例.....23	
1.4.1 软件项目失败的案例.....23	
1.4.2 软件项目成功的案例.....24	
本章小结25	
习题25	
第 2 章 计算机系统工程.....27	
2.1 基于计算机系统28	
2.2 计算机系统工程概述.....29	
2.2.1 人机工程31	
2.2.2 软件工程32	
2.2.3 硬件工程35	
2.2.4 数据库工程37	
本章小结38	
习题38	
第 3 章 软件需求分析管理.....39	
3.1 软件需求分析概述40	
3.1.1 软件需求分析的目标40	
3.1.2 软件需求分析的原则40	
3.1.3 软件需求分析的过程41	
3.2 结构化分析方法.....42	
3.2.1 结构化分析方法概述42	
3.2.2 数据流图.....42	
3.2.3 数据字典.....44	
3.2.4 处理说明.....45	
3.3 如何做好需求分析.....46	
3.3.1 需求分析的 5 个阶段46	
3.3.2 需求分析的重要性.....48	
3.4 图书管理系统需求分析.....49	
本章小结54	
习题55	
第 4 章 概要设计56	
4.1 概要设计的任务与过程.....57	
4.1.1 概要设计的任务.....57	
4.1.2 概要设计的过程.....57	
4.2 概要设计原则.....58	
4.2.1 模块化.....58	
4.2.2 自顶向下, 逐步求精59	
4.2.3 模块的低耦合性和高内聚性 ...59	
4.2.4 其他原则.....63	
4.3 软件结构图.....63	
4.3.1 软件结构图符号.....63	
4.3.2 软件结构图绘制步骤64	
4.3.3 数据流图的类型.....64	
4.3.4 变换分析设计.....65	
4.3.5 事务分析设计.....66	
4.3.6 软件结构图绘制实例67	
4.4 案例: 图书管理系统的概要设计69	
4.4.1 图书管理系统的结构设计69	

4.4.2 图书管理系统的接口设计	71	6.4.1 冗余程序设计	110
4.4.3 图书管理系统的 数据结构设计	71	6.4.2 防错程序设计	110
4.4.4 图书管理系统的 出错处理设计	72	6.5 软件编码管理	111
本章小结	72	6.5.1 软件编码过程的管理	112
习题	73	6.5.2 编码后的管理	112
第 5 章 详细设计	76	本章小结	113
5.1 详细设计的目标和原则	77	习题	113
5.1.1 详细设计的任务和目标	77	第 7 章 软件测试技术	114
5.1.2 详细设计的原则	78	7.1 软件测试的基本概念	115
5.2 详细设计的内容	79	7.1.1 软件测试的重要性	115
5.2.1 代码设计原则	79	7.1.2 软件测试的定义	115
5.2.2 数据库设计	79	7.1.3 软件测试与软件调试的 区别	116
5.2.3 界面设计	81	7.1.4 软件测试的基本原则	116
5.2.4 网络结构设计	84	7.1.5 软件测试的目标	117
5.3 程序流程图设计	85	7.1.6 软件测试过程	117
5.3.1 N-S 图	85	7.2 测试团队的组织	118
5.3.2 PAD 图	87	7.2.1 测试组织者	118
5.3.3 PDL(结构化语言)	89	7.2.2 专业测试人员	119
5.4 案例分析	91	7.2.3 测试配置管理人员	119
本章小结	95	7.2.4 测试相关人员	120
习题	95	7.3 软件测试及测试用例设计	121
第 6 章 软件编程	97	7.3.1 软件测试的分类	121
6.1 结构化程序设计方法	98	7.3.2 白盒测试及其用例设计	122
6.1.1 结构化程序设计的原则	98	7.3.3 黑盒测试及其用例设计	124
6.1.2 逐步求精设计法	99	7.3.4 静态测试	127
6.1.3 结构化程序设计风格	101	7.3.5 动态测试	128
6.2 程序设计的算法与效率	103	7.4 软件测试策略	128
6.2.1 程序设计的算法	103	7.5 软件测试工具	130
6.2.2 程序的运行效率	106	7.6 案例分析	132
6.3 程序设计语言的特点与应用范围	107	7.6.1 图书管理系统黑盒 测试用例	132
6.3.1 程序设计语言的 发展与分类	108	7.6.2 图书管理系统白盒 测试用例	134
6.3.2 程序设计语言的选择	109	7.6.3 图书管理系统界面测试	138
6.4 冗余程序设计与防错程序设计	110	7.6.4 安装/卸载测试	140
		本章小结	141

习题	142	9.4 UML 建模工具—— Rational Rose	182
第 8 章 软件维护	144	9.4.1 Rational Rose 简介	182
8.1 软件维护的基本概念	145	9.4.2 Rational Rose 安装	182
8.1.1 软件维护类型	145	9.4.3 Rational Rose 使用介绍	187
8.1.2 软件维护策略	146	9.5 案例: 图书管理系统面向对象	
8.1.3 软件维护的副作用	147	分析与设计	192
8.1.4 软件维护的困难	148	9.5.1 图书管理系统需求定义	192
8.2 软件维护过程	148	9.5.2 图书管理系统分析	193
8.2.1 维护组织	148	9.5.3 图书管理系统设计	199
8.2.2 维护工作流程	149	9.5.4 图书管理系统实现	201
8.3 软件可维护性	151	本章小结	202
8.3.1 影响软件维护的软件属性	151	习题	203
8.3.2 软件可维护性定量度量	152	第 10 章 软件项目评审	206
8.4 软件再生工程技术	153	10.1 项目评审概论	207
本章小结	154	10.1.1 评审的目的	207
习题	155	10.1.2 评审人员及其职责	208
第 9 章 面向对象系统分析与设计	156	10.1.3 评审的基本要求	
9.1 面向对象概述	157	和评审依据	208
9.1.1 传统开发方法存在的问题	157	10.1.4 评审内容	209
9.1.2 面向对象的概念	158	10.1.5 评审方式	209
9.1.3 面向对象的特征	160	10.1.6 评审工作程序	211
9.2 面向对象开发方法	161	10.2 各阶段的评审内容和要点	214
9.2.1 面向对象 Coad 方法	162	10.2.1 需求分析的评审	215
9.2.2 面向对象 OMT 方法	163	10.2.2 概要设计评审	220
9.2.3 统一软件开发过程		10.2.3 详细设计评审	223
—— RUP	164	本章小结	224
9.3 UML 建模	168	习题	224
9.3.1 UML 概论	168	第 11 章 软件质量保证	
9.3.2 用例图	169	与软件配置管理	225
9.3.3 类图	172	11.1 软件质量与 SQA	226
9.3.4 包图	175	11.1.1 SQA 的目标	227
9.3.5 时序图	176	11.1.2 软件质量计划	227
9.3.6 协作图	177	11.2 软件配置管理	231
9.3.7 状态图	178	11.2.1 软件配置的重要性	231
9.3.8 活动图	180	11.2.2 软件配置项	233
9.3.9 组件图	181	11.2.3 基线管理	233
9.3.10 部署图	181		

11.2.4 软件配置方法	234	12.5.2 CMM 实施之路	256
11.3 图书管理系统质量保证 和配置管理	235	12.5.3 CMM 实施总结	260
11.3.1 图书管理系统质量 保证计划示例	235	12.6 中国软件企业 CMM 的 应用现状与趋势	260
11.3.2 图书管理系统软件 配置管理计划示例	239	本章小结	263
本章小结	242	习题	263
习题	242		
第 12 章 CMM 软件成熟度模型	244	第 13 章 软件工程标准 与软件知识产权	265
12.1 CMM 简介	245	13.1 软件工程标准	266
12.2 CMM 的 5 个等级	246	13.1.1 软件工程标准的 必要性及内容	266
12.3 CMM 的模型框架	249	13.1.2 中国软件工程标准	267
12.4 CMM 的关键过程域	250	13.1.3 国际软件工程标准	267
12.4.1 等级 2 中的关键过程域	251	13.2 计算机软件知识产权	269
12.4.2 等级 3 中的关键过程域	253	13.3 计算机软件知识产权案例分析	274
12.4.3 等级 4 中的关键过程域	254	本章小结	276
12.4.4 等级 5 中的关键过程域	255	习题	276
12.5 CMM 应用案例	255	参考文献	277
12.5.1 公司概况	256		

第 1 章

软件危机与软件工程

教学目标

理解软件危机与软件工程的基本概念，软件工程的基本范畴，软件开发的模型以及软件生命周期。在理解软件以及软件工程的基础上，掌握软件项目管理的特点和目标，了解影响软件项目成功的一系列因素。

教学要求

知 识 要 点	能 力 要 求	关 联 知 识
软件危机	了解软件危机的产生和发展，以及解决软件危机的途径	软件、软件成本、软件质量
软件工程	掌握软件工程的概念及其目标	软件、软件生产率
软件生命周期	掌握软件生命周期各个阶段的定义及其任务	软件定义、软件开发、软件维护
软件生命周期模型	掌握几种典型的软件生命周期模型	软件生命周期、软件生命周期模型
软件项目管理	理解项目管理与软件项目管理的概念，软件项目管理的目标及特点	项目管理、软件过程



引例

提到软件开发,很多人认为就是坐在计算机前面编写代码。其实,编写代码仅仅是软件开发过程中的一个很小的部分。举个例子来说,盖一座大楼时所涉及的工作不仅仅是砌砖和垒墙,还必须对建筑进行主体设计、绘制图纸、估算建筑成本、安排任务、验收建筑质量等。相同的道理,软件开发过程所涉及的环节和活动也不仅仅是编写代码这样简单。比如:在编码之前,要了解编写的程序是为了解决一个什么样的问题,即实现什么样的功能;对于规模较大的软件,应该对软件的系统架构进行整体的规划。编码完成之后,还要考虑采用什么样的方法和途径来减少软件系统中存在的错误。软件产品交付给用户后,要考虑怎样对软件系统进行维护,以及在这个过程中如何保证软件产品的质量等一系列的问题。

软件工程就是一门将所有与软件开发相关的活动归纳在一起,并形成系统的方法和理论的学科。

在第1章中,将学习什么是软件,什么是软件工程,以及如何运用软件工程的理论和方法来开发软件。

1.1 软件危机

软件危机产生于20世纪60年代,给当时的软件行业造成了极大的损失。但同时也促使人们对如何开发软件进行了更加深入的研究和探讨,与程序设计方法学密切相关的软件工程也应运而生。在学习软件工程之前,首先来了解软件危机的产生与发展,以及人们是如何通过各种途径来解决软件危机的。



引例

1963年,美国用于控制火星探测器的计算机软件中的一个“,”号被误写为“.”,最终致使飞往火星的探测器发生爆炸,造成高达数亿美元的损失。

美国IBM公司于1963—1966年开发了IBM360系列机的操作系统,该操作系统花了大约5000人一年的工作量,最多时,有1000人投入开发工作,写出近100万行的源程序。尽管投入了这么多的人力和物力,得到的结果却极其糟糕的。据统计,这个操作系统每次发行的新版本都是从前一版本中找出1000个程序错误而修正的结果。该项目的负责人在项目结束后的总结中写道:“……正像一只逃亡的野兽落到泥潭中做垂死挣扎,越是挣扎,陷得越深,最后无法逃脱灭顶的灾难……程序设计工作正像这样一个泥潭……一批批程序员被迫在泥潭中拼命挣扎……谁也没有料到问题竟会陷入这样的困境……”

上面提到的两个事件,在软件开发的历史进程上可谓赫赫有名。直到今天,软件开发仍然常常提起它们。遗憾的是,这并非是因为它们给软件开发带来荣耀或灵感,而是因为人们要借助它们的失败,给后世的软件开发以警醒和教训。

在20世纪70年代之前,像这样的事件并不罕见。并且随着软件规模的扩大、软件复杂度的增强,类似事件越来越多。后来,人们知道,这一切都是由游荡在软件世界里的幽灵——“软件危机”造成的。

1.1.1 软件危机的表现

在计算机技术应用的初期阶段,也就是20世纪60年代以前,软件的设计和运行只是为了在特定的计算机硬件上完成一个特定的任务。编写软件所采用的语言工具是机器语言或汇编语言;软件的规模比较小;软件的编写者和使用者往往是同一个人;软件产品除了源代码清单外,几乎没有其他文档资料;软件开发方式随意,很少使用系统化的开发方法;此时的软件开发所采用的是“手工作坊”式的开发方式。

20 世纪 60 年代中后期, 计算机硬件的存储能力、计算速度以及可靠性都得到了大幅度的提高, 其成本也不断降低, 计算机设备得到了广泛的应用。在这种形势下, 软件开发需求急剧增长, 人们需要大量的能够完成多种应用任务的计算机软件。与此同时, 高级语言开始出现, 软件系统的规模越来越大, 复杂程度越来越高, 软件数量急剧膨胀, 但是软件可靠性越来越差, 软件维护工作很难进行, 开发成本惊人的高。“手工作坊”式的单打独斗的软件生产方式已经不能够适应日益增长的软件需求, 供求关系严重失调, 最终形成了不可调和的尖锐矛盾, 这就是软件危机的爆发。

“手工作坊”式的生产方式无法满足迅速增长的计算机软件需求, 从而导致了软件开发和维护过程中的一系列严重问题, 这一现象叫做软件危机(Software Crisis)。软件危机主要有以下几个方面的表现。

1. 软件开发费用和进度难以控制

软件开发过程所耗费的实际成本往往高于预算成本, 而实际进度却远远落后于预期进度。用户一边不断地投入大量资金, 一边无限期地等待软件产品的投入使用。这势必会大大降低用户对产品以及软件公司的满意度。而软件开发公司为了压缩成本或加快进度所采取的一些权宜之计, 又常常会严重影响软件产品的质量, 导致软件公司的信誉大大降低。

2. 软件不能满足用户的需求

在软件开发的初期阶段, 一些软件开发人员常常急于求成, 轻视与用户的沟通 and 交流, 在对用户需求只有粗略的了解, 甚至没有进行认真分析的前提下, 就开始编写代码。有些软件开发人员和用户之间虽然有交流, 但开发人员不懂应用系统的专业知识, 用户不懂计算机的专业知识, 两者之间的知识体系不同, 所表达的思想和意图未必能够被对方正确地理解, 再加上未能进行及时有效地沟通, 导致对用户的需求理解不准确, 或是对用户需求定义有错误。以上这些情况都导致了对用户的软件需求定义不准确, 偏离用户需求, 最终生产出来的软件产品不符合用户的实际需要。

3. 软件可靠性差

软件产品的正确性难以保证, 出错率高、质量问题频繁发生, 难以满足用户的实际应用需求, 甚至会因为软件的错误给用户带来难以弥补的重大损失, 令用户丧失对软件产品的信任。

4. 软件产品缺乏相应的文档资料

以往人们认为, 软件就是程序。但是, 软件不仅仅是程序, 还应当包括一整套相应的文档资料。文档资料在软件开发过程中产生, 同时也准确真实地记录软件开发的全过程。每个软件开发过程的环节都离不开文档资料。如: 在软件的初期阶段, 要编写用户需求文档(用户需求说明书), 该文档既是与用户进行交流的书面工具, 也是软件开发人员生产软件产品的依据。缺乏文档资料, 必然会给软件开发和软件维护带来极大的不便和严重的问题。

5. 软件可维护性差

由于软件需求定义不准确、缺乏充分的软件测试、缺少相应的文档资料等原因, 软件

产品中的错误常常难以定位和改正。同时这些程序不能很好地适应新的硬件环境，或是很难根据用户的需求在原有的程序上增加新的功能。“软件的可重用性”还是一个正在努力追求的目标，各软件公司仍然在重复开发着类似或者基本类似的软件，浪费了大量的人力与时间。

6. 软件开发的速度与计算机应用的普及速度不相适应

随着计算机硬件技术的飞速发展，其成本逐年下降。与之相对应的是，软件生产需要耗费大量的人力物力，其成本随着软件规模和数量的不断扩大而增长，致使软件成本在计算机系统总成本中所占的比例居高不下。据美国在 1985 年所统计的数据显示，软件成本大约占计算机系统总成本的 90%。软件开发的生产率远远不能适应计算机应用迅速普及的需要，软件产品的高成本和供不应求使得人们不能充分利用现代计算机硬件所提供的巨大潜力。

1.1.2 软件危机的原因

软件危机所面临的问题有两方面：一是如何开发软件才能满足对软件快速增长的需求；二是如何维护数量和规模不断增长和扩大的已有软件。

软件危机的出现和由它所带来的危害，促使人们去探究其产生的根本原因。最终，人们发现软件危机的产生有两方面的因素：一方面与软件本身的复杂性有关，这是内在因素；另一方面与软件开发和维护所采用的技术和方法有关，这是外在因素。

1. 引发软件危机的内因

引发软件危机的内因主要是软件的复杂性。

软件是一种特殊的逻辑产品，它用代码来体现人的主观思维活动。在编写代码时，其他人很难进行控制和管理。在整个软件系统完成之前，软件开发过程的进度难以衡量和控制，软件质量也较难评价。因此，管理和控制软件开发过程极为困难。

开发过程中任何一个环节的错误，都会导致软件在运行过程中出错，这需要软件维护人员花费大量的时间和精力来进行错误定位和修改。当缺少必要的文档资料时，这一过程将变得异常艰难，软件维护的费用也十分惊人，最终导致软件的维护性越来越差。

随着计算机软件技术的发展，软件的规模越来越大，结构也越来越复杂，控制、管理和维护过程中所遇到的问题越多，开发难度也就越大。

2. 引发软件危机的外因

1) 软件的开发方法不恰当，开发技术落后

在软件开发过程中，人们往往忽略了开发人员和用户之间的矛盾。通常情况下，参与项目的人分为两类：用户和开发人员。用户提出需求；开发人员根据需求进行软件设计和开发。实际存在的问题是，用户因为缺乏计算机知识，不能准确、完整地表达对软件项目的需求；开发人员由于不熟悉用户的专业知识，也同样不能很好地理解和定义用户的需求。实践证明，在没有准确完整地了解用户需求和定义问题的前提下就急于编程，是导致软件开发工程失败的主要原因之一。

在开发过程中, 缺乏有力的开发和管理方法, 缺乏合理有序的工作流程, 各个环节之间不能严密地衔接和关联。例如, 常常有设计方案没有确定, 编码人员就已经着手编码的现象, 一旦设计方案有所改动, 编码人员就不得不进行返工, 造成人力和时间的极大浪费, 甚至会严重影响软件产品的质量。

软件开发是一个复杂的逻辑思维过程, 其产品极大程度地依赖于开发人员高度的智力投入, 但是过分地依赖编程人员在开发过程中的技巧和创造性, 会加剧软件开发产品中的人为因素, 这也是导致软件危机的一个重要原因。

软件产品是人的思维结果, 因此软件生产水平的提高离不开软件人员的教育、训练和经验的积累。当软件从业人员的技术水平停滞不前时, 就很难适应更复杂、更庞大的软件产品的开发了。

2) 开发过程缺乏统一的规范

规模日益增大的软件往往需要许多人合作开发, 这就需要在用户和软件开发人员之间, 以及软件开发人员之间进行有效及时的相互通信, 以消除开发过程中对问题理解的差异, 从而防止后续错误的发生。然而多年的“手工作坊”式的工作环境, 使得软件开发人员更习惯独自工作, 甚至养成了“独行侠”的工作方式。开发过程中所采用的技术没有一个统一的规范和标准, 每个人按照自己的喜好来决定要做什么。这种情况势必会妨碍整个项目开发的团队合作, 这也是最终导致软件危机的一个重要原因。

软件可靠性和质量保证的确切定量概念刚刚出现, 软件质量保证技术(审查、复审和测试)还没有坚持不懈地应用到软件开发的全过程中, 这些也都是导致软件产品发生质量问题的原因。

3) 软件开发管理困难而复杂

大型软件项目需要规模较大的团队来共同完成, 多数管理人员缺乏大型软件系统的管理经验, 而多数软件开发人员又缺乏管理方面的经验。两方面不能进行及时准确的信息交流, 甚至还会产生误解。软件开发人员不能有效、独立地处理软件开发过程中的各种工作流程和工作关系, 因此容易产生疏漏和错误。另外, 开发过程中忽视撰写和保存相关文档的工作, 使文档缺乏一致性和完整性, 甚至没有文档, 从而导致开发者失去工作的基础、管理者失去管理的依据。

4) 开发工具落后, 生产率提高缓慢

软件开发工具过于原始, 没有出现高效率的开发工具, 因而软件生产率低。1960~1980年期间, 计算机硬件的生产由于采用计算机辅助设计、自动生产线等先进工具, 生产率提高了近 100 倍, 而同时期的软件生产率只提高了 2 倍。

1.1.3 解决软件危机的途径

1968 年, 北大西洋公约组织成员国(NATO)在联邦德国召开会议, 近 50 名一流的编程人员、计算机科学家和工业界巨头共同讨论软件危机问题, 并商讨和制定缓解或解决“软件危机”的对策。在这次会议上, 第一次提出了“软件工程”(Software Engineering)的概念。从此, 软件工程作为一门新兴的工程学科诞生了。

软件工程体现了采用工程化的方法从事软件系统的研究和维护的必要性。软件工程的

主要思想是,运用工程学的基本原理和方法来组织和管理软件生产。为了解决软件危机,既要有技术措施(包括方法和工具),又要有必要的组织管理措施。先进的方法和工具可以提高软件开发和维护的效率,更为软件质量提供保证。有效的组织管理措施可以评价、控制、管理整个开发流程,从而保证软件开发能够顺利有效地完成。软件工程正是从技术和管理两个方面,研究如何更好地发展计算机软件技术。

尽管“软件危机”至今尚未被彻底解决,但在 40 年的发展中,经过不断地实践和总结,人们得到一个结论:按照工程化的原则和方法组织软件开发,是摆脱软件危机的一个主要出路。

1.2 软件工程概述

软件工程是一门指导计算机软件开发和维护的工程学科。它运用工程学中的概念、原理、方法和技术来指导软件开发和维护工作。软件是一种逻辑产品,它也有产生和消亡的过程。为了采用工程化的方式对软件生产过程进行有效的管理,人们将软件存在的过程划分成多个阶段,形成“软件生存周期”的概念,它涵盖了软件生产的一系列相关活动。

1.2.1 软件工程的定义

对于软件工程(Software Engineering),不同的学者和组织机构都给出了不同定义。

美国著名的软件工程专家 Barry.W.Boehm 对软件工程的定义是:运用现代科学技术知识来设计并构造计算机程序及为开发、运行和维护这些程序所必需的相关文件资料。

1983 年,IEEE(电气和电子工程师协会)给出的定义是:软件工程是开发、运行、维护和修复软件的系统方法。

归纳起来,目前比较认可的一种定义是:软件工程是指导软件开发和维护的工程学科。其核心思想是采用工程的概念、原理、技术和方法来开发和维护软件,把经过实践考验而证明是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,从而大大提高软件开发的成功率和生产率。

1.2.2 软件工程的范畴

作为一门新兴的学科,软件工程学包括软件开发技术和软件工程管理。而软件开发技术又分为软件开发方法学、软件工具、软件工程环境。软件工程管理又分为软件管理学、软件经济学、软件度量学。

1. 软件开发方法学

软件开发方法学是指导软件开发的某种标准规程,它告诉开发人员“什么时候做以及怎样做”,具体来说,一个软件开发方法规定了明确的工作步骤和具体的描述方式。软件开发方法覆盖了软件开发过程中的一系列活动和任务,包括软件定义、软件开发和软件维护等。在软件开发方法的指导和约束下,面对每个环节的问题,所有开发人员都遵循统一的标准,按照统一的步骤和方式共同完成软件产品。有软件开发方法作为软件生产的行为