



普通高等教育
软件工程

“十二五”规划教材



工业和信息化普通高等教育
“十二五”规划教材

12th Five-Year Plan Textbooks
of Software Engineering

软件工程

李爱萍 崔冬华 李东生 © 主编

*Software
Engineering*



人民邮电出版社
POSTS & TELECOM PRESS



普通高等教育
软件工程

“十二五”规划教材



12th Five-Year Plan Textbooks
of Software Engineering

软件工程

李爱萍 崔冬华 李东生 © 主编

*Software
Engineering*

人民邮电出版社

北京

图书在版编目(CIP)数据

软件工程 / 李爱萍, 崔冬华, 李东生主编. -- 北京:
人民邮电出版社, 2014.3
普通高等教育软件工程“十二五”规划教材
ISBN 978-7-115-34079-5

I. ①软… II. ①李… ②崔… ③李… 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(2014)第018875号

内 容 提 要

本书系统地介绍了软件工程的观念、原理、方法和案例, 比较全面地反映了软件工程技术的全貌。全书共3篇, 分16章, 分别以“面向过程的软件工程”、“面向对象的软件工程”和“软件工程实验”为主线阐述软件工程原理和方法。本书第2篇还介绍了软件体系结构、设计模式、软件工程新技术等内容, 体现了软件工程的最新理论和技术。本书的主要特色是强调以面向过程或面向对象的思想指导软件开发过程, 并配以实例分析和说明, 便于教学和应用。

本书适合作为高等院校计算机、软件工程专业或信息类相关专业本科生或研究生软件工程课程的教材, 也可作为软件开发技术人员的参考书。

◆ 主 编 李爱萍 崔冬华 李东生

责任编辑 邹文波

责任印制 彭志环 焦志炜

◆ 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>

北京铭成印刷有限公司印刷

◆ 开本: 787×1092 1/16

印张: 22.75

字数: 598千字

2014年3月第1版

2014年3月北京第1次印刷



定价: 48.00 元

读者服务热线: (010)81055256 印装质量热线: (010)81055316

反盗版热线: (010)81055315

自 1968 年第一届 NATO 会议上首次提出“软件工程”的概念至今,已经过去了四十多年,软件工程得到了很大发展,不断出现新方法、新技术和新模型,形成了软件工程领域的基础理论、工程方法和技术体系,具备了学科的完整性和教育学特色。2011 年 2 月我国国务院学位委员会在新修订的学科目录中,将软件工程(学科代码为 0835)增设为一级学科,进一步促进了软件工程的发展。

软件工程课程是高等学校计算机科学与技术、软件工程等专业的一门重要的专业核心课程,是信息类专业的推荐课程,也是每一个从事软件分析、设计、开发、测试、管理和维护人员的必备知识。

本书按照面向过程和面向对象两种开发方法,对软件生存周期各个阶段的基础理论和相关技术进行了介绍,同时给出两种不同方法的实验指导,比较全面地反映了软件工程的全貌。本书可用于高等学校相关专业的教学。

全书由 3 篇共计 16 章组成,按专题安排,便于组织教学。

第一篇为面向过程的软件工程,以传统的软件生存周期为主线,介绍软件工程的原理和方法。第 1 章介绍软件工程相关的基本概念。第 2 章至第 8 章按照软件生存周期的各个阶段顺序介绍可行性研究、需求分析、软件总体设计、详细设计、软件编码、软件测试和软件维护,详细介绍有关概念和软件工程方法,重点介绍结构化的分析和设计方法。

第二篇内容为面向对象的软件工程,以 UML 为主线,结合当前流行的开源工具 StarUML,介绍面向对象软件工程的原理和方法。第 9 章为面向对象方法学以及面向对象的基本概念介绍;第 10 章介绍面向对象分析的过程,并给出一个面向对象软件开发过程案例的分析;第 11 章介绍面向对象设计与实现;第 12 章介绍软件开发工具 StarUML 的整体功能及其应用,并以一个教学管理系统实例介绍使用 StarUML 完成整个系统的分析、设计的全过程;第 13 章介绍几种典型的软件体系结构;第 14 章是设计模式简介;第 15 章介绍软件工程领域的一些最新技术。

第三篇给出了软件工程的实验环节要求和示例,包括面向过程的软件工程和面向对象的软件工程两种不同开发方法的实验要求,并通过两个典型的案例具体说明两类软件工程实验的具体实施步骤。

本书由长期从事软件工程教学和科研的教师编写。李爱萍、崔冬华同志负责全书架构的设计和统稿。本书共 16 章,其中第 1 章、第 2 章、第 3 章、第 4 章由崔冬华编写,第 5 章、第 6 章、第 7 章由武昭宇编写,第 9 章、第 14 章和附录由李新编写,第 10.6 节由李东生编写,第 3.7 节、4.4 节、第 10 章、第 11 章、第 12 章由段利国编写,第 8 章、第 13 章、第 15 章、第 16 章由李爱萍编写,研究生马俊伟参与了 StarUML 工具的应用作图部分的相关工作。

本书编写过程中参考了国内外有关软件工程的专著、教材和论文，详见书后的主要参考文献。在此，向所有作者表示谢意。

由于软件工程近年来的快速发展以及作者水平所限，书中若存在错误和不足之处，敬请读者提出宝贵意见和建议。

作 者

2014 年 1 月

目 录

第 1 篇 面向过程的软件工程

第 1 章 概述	2	第 2 章 可行性研究	17
1.1 软件和软件危机	2	2.1 可行性研究的目的和任务	17
1.1.1 软件的定义、特点及分类	2	2.2 可行性研究的步骤	18
1.1.2 软件危机的定义及表现形式	3	2.3 系统流程图	19
1.1.3 软件危机的产生原因及解决 途径	4	2.3.1 系统流程图的符号	19
1.2 软件工程	5	2.3.2 系统流程图示例	20
1.2.1 软件工程的定义及目标	5	2.4 成本—效益分析	20
1.2.2 软件工程的研究内容和基本 原理	6	2.4.1 货币的时间价值	20
1.2.3 软件工程的发展史	7	2.4.2 投资回收期	21
1.3 软件过程和软件的生存周期	8	2.4.3 纯收入	21
1.3.1 软件过程	8	2.5 可行性研究报告的主要内容	21
1.3.2 软件生存周期	8	本章小结	22
1.4 软件开发模型	10	习题 2	22
1.4.1 瀑布模型	10	第 3 章 软件需求分析	23
1.4.2 快速原型模型	11	3.1 需求分析的任务和步骤	23
1.4.3 增量模型	12	3.1.1 需求分析的任务	23
1.4.4 螺旋模型	13	3.1.2 需求分析的步骤	24
1.4.5 喷泉模型	14	3.2 需求获取的常用方法	25
1.4.6 基于构件的开发模型	14	3.2.1 需求获取的常用方法	25
1.4.7 统一过程 (RUP) 模型	15	3.2.2 快速建立软件原型模型来获取 需求	26
1.4.8 基于形式化的开发模型	15	3.3 需求分析的常用方法	27
1.5 传统的软件工程和面向对象的软件 工程	15	3.3.1 功能分解方法	27
1.5.1 传统软件工程方法	15	3.3.2 结构化分析方法	27
1.5.2 面向对象软件工程方法	15	3.3.3 信息建模方法	27
1.5.3 传统软件工程和面向对象软件 工程的分析方法对比	16	3.3.4 面向对象方法	28
本章小结	16	3.4 结构化分析方法	28
习题 1	16	3.4.1 自顶向下逐层分解的分析策略	28
		3.4.2 结构化分析描述工具	29
		3.4.3 数据流程图	29

3.4.4 数据字典.....	32	本章小结.....	65
3.4.5 加工逻辑的描述.....	34	习题 4.....	66
3.5 需求分析图形工具.....	36	第 5 章 软件详细设计	67
3.5.1 层次方框图.....	36	5.1 详细设计的目的与任务.....	67
3.5.2 维纳图.....	36	5.2 结构化程序设计.....	67
3.5.3 IPO 图.....	37	5.3 详细设计工具.....	68
3.6 SA 方法的应用.....	37	5.3.1 程序流程图.....	68
3.7 数据库内容的需求分析和描述.....	42	5.3.2 盒图 (N-S 图).....	69
3.7.1 数据库内容的需求分析.....	42	5.3.3 问题分析图 (PAD 图).....	69
3.7.2 数据库内容的需求分析描述.....	44	5.3.4 过程设计语言 (PDL).....	70
3.8 需求规格说明书.....	45	5.4 人机界面设计.....	73
本章小结.....	46	5.4.1 人机界面设计问题.....	73
习题 3.....	46	5.4.2 人机界面设计原则.....	75
第 4 章 软件总体设计	48	5.4.3 人机界面设计过程.....	75
4.1 软件总体设计的目标和任务.....	48	5.5 详细设计说明书.....	76
4.2 软件结构设计准则.....	48	本章小结.....	76
4.2.1 软件体系结构设计准则.....	49	习题 5.....	77
4.2.2 软件模块设计准则.....	49	第 6 章 软件编码	78
4.3 软件设计的概念和原理.....	50	6.1 程序设计语言的分类.....	78
4.3.1 模块和模块化.....	50	6.1.1 基础语言.....	78
4.3.2 抽象.....	51	6.1.2 结构化语言.....	79
4.3.3 信息隐蔽和局部化.....	51	6.1.3 面向对象语言.....	80
4.3.4 模块独立性及其度量.....	52	6.2 程序设计语言的选择.....	81
4.4 数据库设计.....	54	6.3 程序设计风格.....	82
4.4.1 数据库系统的三级模式概念.....	54	6.3.1 程序内部文档.....	82
4.4.2 数据库逻辑设计原则.....	55	6.3.2 数据说明.....	83
4.4.3 关系数据库规范化.....	56	6.3.3 语句构造.....	83
4.5 软件结构设计的图形工具.....	58	6.3.4 输入/输出.....	83
4.5.1 软件结构图.....	58	6.3.5 效率.....	83
4.5.2 层次图.....	58	本章小结.....	85
4.5.3 HIPO 图.....	59	习题 6.....	85
4.6 结构化设计方法.....	59	第 7 章 软件测试	86
4.6.1 数据流图及其类型.....	59	7.1 软件测试的目标.....	86
4.6.2 结构化设计方法的步骤.....	60	7.2 软件测试的原则.....	86
4.6.3 变换型数据流图的分析设计.....	61	7.3 软件测试方法及分类.....	87
4.6.4 事务型数据流图的分析设计.....	62	7.3.1 静态测试与动态测试.....	87
4.6.5 综合分析设计.....	64		
4.7 总体设计说明书.....	64		

7.3.2 黑盒测试与白盒测试	88	8.1.3 软件维护的特点	109
7.4 软件测试用例的设计	89	8.2 软件维护的过程	109
7.4.1 白盒技术	89	8.2.1 结构化维护与非结构化维护	109
7.4.2 黑盒技术	93	8.2.2 维护组织	110
7.5 软件测试过程	97	8.2.3 维护工作的流程	111
7.5.1 单元测试	98	8.2.4 维护工作的组织管理	113
7.5.2 集成测试	99	8.3 软件可维护性	113
7.5.3 确认测试	101	8.3.1 软件可维护性的定义	114
7.5.4 系统测试	102	8.3.2 软件可维护性的度量	115
7.5.5 α 测试和 β 测试	103	8.3.3 提高软件可维护性的方法	115
7.6 调试	103	8.4 软件维护中存在的问题	117
7.6.1 调试的目的和任务	103	8.4.1 软件维护困难的表现	117
7.6.2 常用调试技术	103	8.4.2 软件维护的副作用	118
7.7 软件测试阶段终止的条件	104	8.4.3 软件维护的代价	118
本章小结	105	8.5 再工程和逆向工程	119
习题 7	105	8.5.1 再工程与逆向工程的概念	119
第 8 章 软件维护	107	8.5.2 为什么要实施软件再工程	120
8.1 软件维护概述	107	8.5.3 软件再工程技术	121
8.1.1 软件维护的定义	107	本章小结	122
8.1.2 软件维护的分类	107	习题 8	123

第 2 篇 面向对象的软件工程

第 9 章 面向对象方法学	125	9.4.1 RUP 的历史	149
9.1 面向对象方法概述	125	9.4.2 RUP 的特色	149
9.1.1 面向对象方法学的发展	125	9.4.3 RUP 软件开发的生存周期	150
9.1.2 面向对象方法学的优点和不足	126	9.4.4 RUP 过程的建模	152
9.2 面向对象的软件工程	127	本章小结	153
9.2.1 面向对象基本概念	128	习题 9	153
9.2.2 对象、类及类之间关系的分析	129	第 10 章 面向对象的分析	155
9.2.3 典型的面向对象开发方法	133	10.1 面向对象的分析过程	155
9.3 统一建模语言 UML	138	10.1.1 用例模型	156
9.3.1 UML 的发展	138	10.1.2 对象模型	156
9.3.2 UML 的定义及主要内容	139	10.1.3 动态模型	156
9.3.3 UML 的特点和用途	141	10.1.4 物理（实现）模型	157
9.3.4 UML 的模型视图简介	142	10.1.5 4 种模型之间的关系	157
9.3.5 UML 常用软件开发工具	144	10.2 建立用例模型	158
9.4 统一软件开发过程 RUP 概述	148	10.2.1 需求分析与用例建模	158

10.2.2 确定系统范围和系统边界	158	11.3.1 服务设计	197
10.2.3 确定参与者	158	11.3.2 关联设计	198
10.2.4 确定用例	159	11.3.3 聚合关系设计	202
10.2.5 确定用例之间的关系	161	11.4 面向对象设计优化	203
10.3 建立对象模型	162	11.5 面向对象系统的实现	205
10.3.1 确定类和对象	163	11.5.1 面向对象程序设计语言的 选择	205
10.3.2 确定关联	163	11.5.2 面向对象程序设计风格	205
10.3.3 确定属性	164	本章小结	206
10.3.4 建立对象类图	164	习题 11	206
10.3.5 划分主题	165		
10.3.6 优化对象模型	165		
10.4 建立动态行为模型	166	第 12 章 软件开发工具 StarUML 及其应用	208
10.4.1 建立顺序图	166	12.1 软件开发工具 StarUML 概述	209
10.4.2 建立通信图	167	12.1.1 StarUML 的安装及使用	209
10.4.3 建立状态图	168	12.1.2 StarUML 的主要功能	209
10.4.4 建立活动图	171	12.2 StarUML 环境下的 UML 图形建模	211
10.5 建立物理实现模型	175	12.2.1 StarUML 的用例图、类图和 包图	213
10.5.1 建立构件图	175	12.2.2 StarUML 的交互图和状态机图	217
10.5.2 建立部署图	178	12.2.3 StarUML 的构件图与部署图	222
10.6 面向对象软件开发过程的案例 分析	181	12.2.4 StarUML 的正向工程和逆向 工程	223
10.6.1 系统需求	181	12.3 一个简易教学管理系统的分析和 设计	225
10.6.2 系统用例模型	182	12.3.1 系统需求描述及分析	226
10.6.3 系统对象模型	182	12.3.2 系统问题领域分析	226
10.6.4 系统动态行为模型	183	12.3.3 静态结构模型的建立	230
10.6.5 系统物理实现模型	185	12.3.4 动态行为模型的建立	232
本章小结	186	12.3.5 系统物理模型的建立	236
习题 10	186	12.3.6 代码框架自动生成的过程	237
		本章小结	238
		习题 12	238
第 11 章 面向对象的设计与实现	188	第 13 章 软件体系结构	240
11.1 面向对象的设计准则和启发式规则	188	13.1 软件体系结构概述	240
11.1.1 面向对象设计准则	188	13.2 经典的软件体系结构风格	241
11.1.2 启发式规则	189	13.2.1 管道-过滤器（流程处理）体系 结构	241
11.2 系统设计	190		
11.2.1 问题域子系统设计	191		
11.2.2 人机交互子系统设计	193		
11.2.3 任务管理子系统设计	194		
11.2.4 数据管理子系统设计	195		
11.3 服务、关联与聚合关系设计	197		

13.2.2 分层体系结构.....	242	15.1 软件复用技术.....	271
13.2.3 客户机/服务器体系结构.....	243	15.1.1 软件复用概念及分类.....	271
13.2.4 浏览器/服务器体系结构 (B/S).....	245	15.1.2 软件复用的关键技术和复用 粒度.....	272
13.3 现代的软件体系结构风格.....	246	15.2 基于构件的软件工程技术.....	272
13.3.1 公共对象请求代理体系结构.....	246	15.2.1 中间件技术.....	273
13.3.2 基于层次消息总线的体系结构 风格.....	248	15.2.2 构件与构件化.....	275
13.3.3 异构结构风格.....	248	15.2.3 构件模型及描述语言.....	276
本章小结.....	251	15.2.4 构件的检索与组装.....	278
习题 13.....	251	15.2.5 基于构件的软件工程方法.....	281
第 14 章 设计模式	252	15.2.6 SOA 与 SaaS.....	283
14.1 设计模式概述.....	252	15.3 软件过程与标准化.....	285
14.1.1 设计模式的出现和发展.....	252	15.3.1 软件过程及其改进.....	285
14.1.2 设计模式和软件体系结构的 关系.....	253	15.3.2 ISO 9000 标准.....	286
14.1.3 设计模式的优点和分类.....	253	15.3.3 软件能力成熟度模型(CMM).....	287
14.2 GRASP 设计模式及其应用.....	254	15.3.4 PSP、TSP 和 CMMI.....	289
14.2.1 GRASP 设计模式概念及其 分类.....	254	15.4 敏捷软件开发过程.....	291
14.2.2 几种典型的 GRASP 设计模式 应用简介.....	256	15.4.1 敏捷及敏捷过程相关概念.....	292
14.3 GoF 设计模式及其应用.....	258	15.4.2 典型的敏捷过程模型.....	294
14.3.1 GoF 设计模式的组成及其分类.....	260	15.5 Web 软件工程.....	298
14.3.2 创建型 GoF 设计模式应用实例.....	261	15.5.1 Web 软件工程概述.....	298
14.3.3 结构型 GoF 设计模式应用实例.....	265	15.5.2 Web 软件的需求分析.....	300
14.3.4 行为型 GoF 设计模式应用实例.....	268	15.5.3 Web 软件的设计.....	302
本章小结.....	269	15.5.4 Web 软件的测试.....	305
习题 14.....	270	15.6 软件产品线技术.....	306
第 15 章 软件工程新技术	271	15.6.1 软件产品线基本概念.....	307
		15.6.2 软件产品线方法.....	309
		15.6.3 北大青鸟工程.....	310
		本章小结.....	311
		习题 15.....	312

第 3 篇 软件工程实验

第 16 章 软件工程实验	314	计划.....	316
16.1 面向过程的软件工程实验.....	315	实验 3 图书管理系统的需求分析.....	317
实验 1 图书管理系统的可行性分析.....	316	实验 4 图书管理系统的总体设计.....	317
实验 2 图书管理系统的项目开发		实验 5 图书管理系统的详细设计及编码 实现.....	318

实验 6 图书管理系统的测试.....	319	A.3 通信图.....	336
实验 7 图书管理系统用户手册的 撰写.....	319	A.4 构件图.....	337
实验 8 图书管理系统项目开发总结报告 的撰写.....	320	A.5 组合结构图.....	337
16.2 面向对象的软件工程实验.....	320	A.6 部署图.....	337
实验 9 浏览器系统的可行性分析.....	320	A.7 对象图.....	338
实验 10 浏览器系统的项目开发计划.....	321	A.8 包图.....	338
实验 11 浏览器系统的需求分析.....	322	A.9 参数化通信图.....	338
实验 12 浏览器系统的体系结构设计.....	327	A.10 顺序图.....	338
实验 13 采用面向对象方法进行浏览 器系统的详细设计.....	329	A.11 状态图.....	338
实验 14 利用 StarUML 正向工程功能完成 代码框架的自动生成实验.....	331	A.12 计时图.....	339
实验 15 利用 StarUML 逆向工程功能 完成修改设计的实验.....	332	A.13 用例图.....	339
实验 16 浏览器系统的测试.....	332	附录 B UML 中定义的常用版型 (stereotype)、约束 (constraint) 和标记 (tag)	341
实验 17 浏览器系统用户手册的撰写.....	334	附录 C GOF 给出的软件设计 模式	344
实验 18 浏览器系统项目开发总结报告 的撰写.....	334	C.1 创建型模式 (Creational)	344
附录 A UML 图总结	335	C.2 结构型 (Structural) 模式.....	346
A.1 活动图.....	335	C.3 行为 (Behavioral) 模式.....	349
A.2 类图.....	336	参考文献.....	354

第 1 篇

面向过程的软件工程

- 第 1 章 概述
- 第 2 章 可行性研究
- 第 3 章 软件需求分析
- 第 4 章 软件总体设计
- 第 5 章 软件详细设计
- 第 6 章 软件编码
- 第 7 章 软件测试
- 第 8 章 软件维护

第 1 章 概 述

“软件工程”一词是由北大西洋公约组织（North Atlantic Treaty Organization, NATO）的计算机科学家在联邦德国召开的国际会议上首次提出来的。产生软件工程这门学科的时代背景是“软件危机”。软件工程的发展和应用不仅缓和了软件危机，而且促使一门新兴的工程学科诞生了。

软件工程是应用计算机科学、工程学、管理学及数学的原则、方法来创建软件的学科，它对指导软件开发、质量控制以及开发过程的管理起着非常重要的作用。本章介绍软件和软件工程的基本概念，包括软件、软件危机、软件工程方法学、软件过程和软件生存周期模型，传统软件工程与面向对象软件工程等，从而使读者对软件工程与软件开发技术有所认识。

1.1 软件和软件危机

1.1.1 软件的定义、特点及分类

1. 软件的定义

软件不是程序，而是程序、数据以及开发、使用和维护程序需要的所有文档的完整集合。1983 年 IEEE 为软件下的定义是：计算机程序、方法、规则、相关的文档资料以及在计算机上运行程序时所必需的数据。其中的方法和规则通常是在文档中说明并在程序中实现的。特别是当软件成为商品时，相关的文档资料是必不可少的。没有相关文档，仅有程序是不能称为软件产品的。

程序是为了解决某个特定问题而用程序设计语言描述的适合计算机处理的语句序列。它是由软件开发人员设计和编码的。程序执行时一般要输入一定的数据，也会输出运行的结果。而文档则是软件开发活动的记录，主要供人们阅读，既可用于专业人员和用户之间的通信和交流，也可以用于软件开发过程的管理和运行阶段的维护。我国国家标准局已参照国际标准，陆续颁布了《计算机软件开发规范》、《计算机软件需求说明编制指南》、《计算机软件测试文件编制规范》、《计算机软件配置管理计划规范》等文档规范。为了提高软件开发的效率和方便软件产品的维护，现在的软件人员越来越重视文档的作用及其标准化工作。

2. 软件的特点

为了能全面、正确地理解计算机和软件，必须了解软件的特点。软件是一种特殊的产品，与传统的工业产品相比，它具有以下一些独特的特点。

（1）软件是一种逻辑产品，而不是具体的物理实体，具有抽象性，人们可以把它记录在纸上，保存在计算机内存、磁盘和光盘等存储介质上，但却无法看到软件本身的形态，必须

通过观察、分析、思考、判断以及通过计算机的执行才能了解到它的功能和作用。

(2) 软件产品的生产主要是开发研制,没有明显的制造过程。软件开发研制完成后,通过复制可以产生大量软件产品,所以对软件的质量控制,必须着重在软件开发方面下功夫。

(3) 软件产品在使用过程中,不存在磨损、消耗、老化等问题。但软件在运行时,为了适应软件硬件、环境以及需求的变化而进行修改、完善时,会引入一些新的错误,从而使软件退化,在修改的成本变得让人们难以接受时,软件就被抛弃,生存期停止。

(4) 软件产品的开发主要是脑力劳动,还未完全摆脱手工开发方式,大部分产品是“定做的”,生产效率低。

(5) 软件产品的成本相当昂贵,软件费用不断增加,软件的研制需要投入大量的人力、物力和资金,生产过程中还需对产品进行质量控制,对每件产品进行严格的检验。

(6) 软件对硬件和环境有不同程度的依赖性,为了减少这种依赖性,在软件开发中提出了软件的可移植性问题。

(7) 软件是复杂的。软件是人类有史以来生产的复杂度最高的工业产品,软件是一个庞大的逻辑系统。软件开发,尤其是应用软件的开发常常涉及其他领域的专门知识,这就对软件开发人员提出了很高的要求。

3. 软件的分类

计算机软件发展非常迅速,其内容十分丰富,要给计算机软件做出科学的分类是比较困难的。传统意义上从计算机系统角度看,软件分为两大类:系统软件和应用软件。系统软件是指管理、控制和维护计算机及外设,以及提供计算机与用户界面等的软件,如操作系统、各种语言的编译系统、数据库管理系统及网络软件等。应用软件是指能解决某一应用领域问题的软件,如财会软件、通信软件、计算机辅助教学(CAI)软件等。

若从计算机软件用途来划分,大致分为服务类、维护类和操作管理类。

(1) 服务类软件。此类软件是面向用户的,为用户提供各种服务,包括多种软件开发工具和常用的库函数及多种语言的集成化软件,如 Windows 下的 Visual C++ 软件等。

(2) 维护类软件。此类软件是面向计算机维护的,包括错误诊断和检测软件、测试软件、多种调试所用软件如 Debug 等。

(3) 操作管理软件。此类软件是面向计算机操作和管理的,包括各种操作系统、网络通信系统、计算机管理软件等。

1.1.2 软件危机的定义及表现形式

1. 软件危机

“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。软件危机主要包含了两方面的问题:一是如何开发软件以满足软件日益增长的需求;二是如何维护数量不断增长的已有软件。

2. 软件危机表现形式

(1) 对软件开发成本和研制进度的估计常常很不精确。经费预算经常突破,完成时间一拖再拖。这种现象降低了软件开发组织的信誉,而且有时为了赶进度和节约成本所采取的一些权宜之计又往往影响了软件产品的质量,从而不可避免地会引起用户的不满。

(2) “已完成”的软件不能满足用户要求。软件开发人员常常在对用户需求只有模糊的了解,甚至对所要解决的问题还没有确切认识的情况下,就匆忙着手编写程序了。软件开发人

员和用户又未能及时交换意见,使得一些问题不能得到及时解决,导致开发的软件不能满足用户要求,使得开发失败。

(3) 软件产品质量差,可靠性得不到保证。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,提交给用户的软件质量差,在运行中暴露大量问题。

(4) 软件产品可维护性差,软件开发人员在开发过程中按各自的风格工作,各行其是,没有统一、公认的规范和完整规范的文档,发现问题后进行杂乱无章的修改。程序结构不好,运行时发现错误也很难修改,导致维护性差。

(5) 软件成本在计算机系统总成本中所占的比例逐年上升。软件的发展跟不上硬件的发展。由于微电子技术的进步和生产自动化程度的不断提高,硬件成本逐年下降,然而软件开发需要大量人力,软件成本也随着通货膨胀以及软件规模和数量的不断扩大而持续上升。

(6) 软件开发生产率提高的速度远远跟不上计算机应用速度普及深入的趋势。软件的发展跟不上用户的要求。软件产品“供不应求”的现象使人类不能充分利用现代计算机硬件提供的巨大潜力。

以上列举的仅仅是软件危机的典型表现,与软件开发和维护有关的问题远远不止这些。

1.1.3 软件危机的产生原因及解决途径

1. 产生软件危机的原因

造成上述软件危机的原因与软件自身特点有关,也与软件开发人员在开发和维护时所采用的生产方式、方法、技术有关,可概括为以下几个方面。

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

(2) 软件开发的管理困难。软件规模大、结构复杂,又具有无形性,这导致管理困难,进度控制困难,质量控制困难,可靠性无法保证。

(3) 软件开发费用不断增加,维护费用急剧上升,直接威胁计算机应用的扩大。

(4) 软件开发技术落后。在 20 世纪 60 年代,人们注重如编译原理、操作系统原理、数据库原理等一些计算机理论问题的研究,而不注重软件开发技术的研究,用户要求的软件复杂性与软件技术解决复杂性的能力不相适应。

(5) 生产方式落后。有人统计,硬件的性能价格比在过去 30 年中增长了 10^6 。一种新器件的出现,其性能较旧器件提高,价格反而有所下降,而软件则相形见绌。软件规模与复杂性增长了几个数量级,但生产方式仍然采用个体手工方式开发,根据个人习惯爱好工作,无章可循,无规范可依靠,带有很强的“个性化”特征的程序,因缺乏文档而根本不能维护,加剧了供需之间的矛盾。

(6) 开发工具落后,生产效率提高缓慢。软件开发工具趋于原始,没有出现高效率的开发工具,因而软件生产效率低下。还有软件开发人员忽视需求分析的重要性,轻视软件维护也是造成软件危机的原因。

2. 解决软件危机途径

目前,计算机的应用日益广泛,世界上发达国家的许多企业将全部投资的 10% 以上用于计算机领域。但到目前为止,计算机的体系结构在硬件上仍然是冯·诺依曼计算机。实际中复杂、庞大的问题,只能由专门人员编制软件来解决。假设计算机能实现智能化,能进行推理和运算,正确解决用户所提出的问题,那么软件危机就会有根本性的缓解。然而新

一代计算机体系结构的研制可能还需要一段时间。那么在目前计算机硬件条件下,要想解决软件危机必须解决以下问题。

(1) 首先应该对计算机软件有一个正确的认识,彻底清除“软件就是程序”的错误观念。

(2) 要使用好的开发技术和方法,并且要不断研究探索更好更有效的技术和方法。尽快消除在计算机系统早期发展阶段形成的一些错误观念和做法。

(3) 要有良好的组织、严密的管理,各类人员要相互配合,共同完成任务。充分认识软件开发不是某种个体劳动的神秘技巧。

(4) 应该开发和使用好的软件工具。正如机械工具可以“放大”人类的体力一样,软件工具也可以“放大”人类的智力,从而可以有效提高软件生产率。

软件系统开发与制造一台机器或建造一栋大厦有许多相同之处,所以要采用“工程化”的思想做指导来解决软件研究中面临的困难和混乱,从而走出软件危机的困境。

1.2 软件工程

1.2.1 软件工程的定义及目标

1. 软件工程

软件工程是指导计算机软件开发和维护的一门工程学科。软件工程采用工程的概念、原理、技术和方法来开发和维护软件。

人们曾从不同的角度给软件工程下过各种定义,下面给出两种比较典型的定义。

1968年,在第一届NATO会议上曾经给出了软件工程的一种早期定义:“软件工程就是为了经济地获得可靠的且能在实际机器上有效地运行的软件,而建立和使用完善的工程原理。”这个定义不仅指出了软件工程的目标是经济地开发出高质量的软件,而且强调了软件工程是一门工程学科,它应该建立并使用完善的工程原理。

1993年,IEEE进一步给出了一个更全面更具体的定义:“软件工程是:①把系统的、规范的、可度量的途径应用于软件开发、运行和维护过程,也就是把工程应用于软件开发;②研究①中提到的途径。”

虽然软件工程的定义使用了不同的词句,强调的重点也有所差异,但是它的中心思想是把软件当作一种工业产品,要求“采用工程化的原理和方法对软件进行计划、开发和维护。”宗旨是为了提高软件生产率、降低生产成本,以较小的代价获得高质量的软件产品。

2. 软件工程基本目标

软件工程是一门工程性学科,目的是成功地建造一个大型软件系统。所谓成功是指要达到以下几个目标:

- (1) 降低软件开发成本;
- (2) 满足用户要求的全部软件功能;
- (3) 符合用户要求,令用户满意的软件性能;
- (4) 具有较好的易用性、可重用性和可移植性;
- (5) 较低的维护成本,较高的可靠性;
- (6) 按合同要求完成开发任务,及时交付用户使用。

1.2.2 软件工程的研究内容和基本原理

1. 软件工程的研究内容

软件工程的主要研究内容是软件开发技术和软件开发过程管理两个方面。在软件开发技术方面,主要研究软件开发方法、软件开发过程、软件开发工具和技术。在软件开发过程管理方面,主要研究软件工程经济学和软件管理学。技术与管理是软件开发中缺一不可的两个方面。没有科学的管理,再先进的技术也不能充分发挥作用。

2. 软件工程的基本原理

自从1968年“软件工程”正式提出并使用以来,从事软件工程研究的世界各国专家学者先后提出100余条关于软件工程的准则。著名的软件工程师B.W.Boehm综合有关专家和学者的意见,总结了多年来开发软件的经验,于1983年在一篇论文中提出了软件工程7条基本原理。他认为这7条基本原理是确保软件产品质量和开发效率原理的最小集合。它们之间相互独立,其中任意6条原理的组合都不能代替另一条原理,因此,它们是缺一不可的最小集合。下面简单介绍软件工程的7条基本原理。

(1) 用分阶段的生存周期计划严格管理

阶段划分为计划、分析、设计、编程、测试和运行维护。B.W.Boehm认为在软件的整个生存周期中应该制定并严格执行6类计划:项目概要计划、里程碑计划、项目控制计划、产品控制计划、验证计划和运行维护计划。

(2) 坚持进行阶段评审

上一阶段评审没有通过,就不能进入下一阶段工作。在软件生存周期的每个阶段都要进行严格的评审,以便尽早发现在软件开发中犯下的错误并及时改正,对于保证软件质量、提高开发效率是非常重要的。

(3) 实行严格的产品控制

尽管面向对象软件开发支持用户需求变化,但是在开发过程中,改变一项要求是要付出较高代价的。在实际开发过程中,由于外部环境等的变化,相应地改变用户需求是一种客观需要,是难免的,显然不能硬性禁止用户提出改变需求的要求,而只是依据科学的产品控制技术来适应这种要求。

(4) 采用现代程序设计技术

实践表明,采用先进的程序设计技术既可以提高软件的开发效率,又可以提高软件的维护效率。面向对象技术在许多领域已经取代了传统的结构化开发方法。软件的开发效率、可维护性、可重用性都有一定程度的提高。

(5) 结果应能清楚地审查

软件产品是一种逻辑产品,软件开发工作的进度情况可见性差,很难准确度量,使得软件产品的开发过程比一般产品的开发过程更难以评价和管理。为了提高软件开发过程的可见性,更好地进行组织和管理,应该根据软件产品的总目标及完成期限,规定开发人员的责任和各阶段产品标准,从而使得软件开发每一阶段所得到的结果能够清楚地审查。

(6) 开发小组人员应该少而精

B.W.Boehm总结多年从事软件开发的实践经验得出这条基本原理,其含义是,开发小组的组成人员应该具有高素质、高水平,而人数不宜太多。软件开发组人员的素质和数量是影响软件产品质量和开发效率的关键因素。