



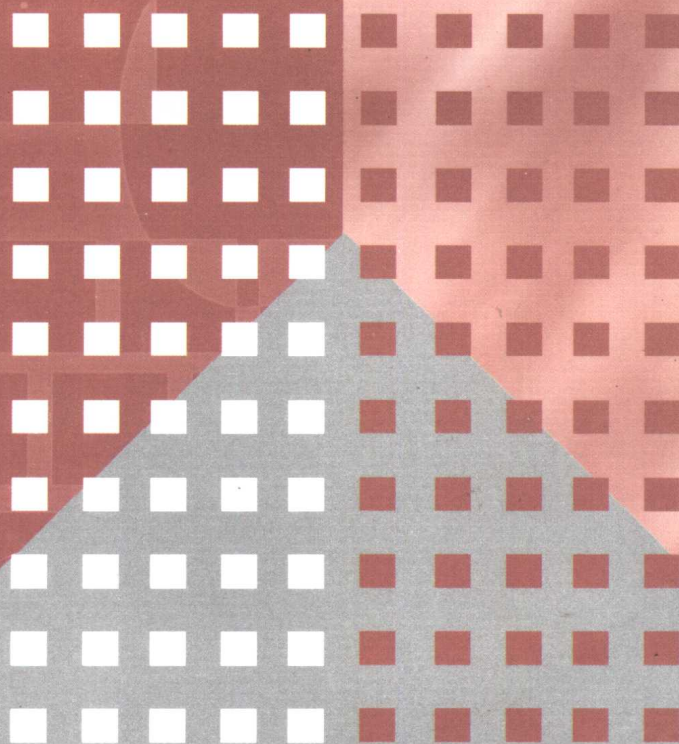
中国计算机软件专业技术水平考试指定用书

中国计算机软件专业技术资格和水平考试中心组织编写

软件工程

郑人杰 主编

(高级)



清华大学出版社

<http://www.tup.tsinghua.edu.cn>



中国计算机软件专业技术水平考试指定用书

软 件 工 程

(高 级)

中国计算机软件专业技术资格和水平考试中心 组织编写

郑人杰 主编

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书是中国计算机软件专业技术水平考试指定用书之一。本书是软件工程的高级读物,全书分为三个部分,即软件工程技术、软件质量管理与质量保证及软件工程管理。主要内容包括:软件生存期过程和软件工程,软件需求分析,软件复用技术,软件测试,软件维护与软件再工程,软件工具与软件开发环境,软件质量保证,软件工程标准化和软件文档,软件过程能力评估,软件工程项目管理,软件度量,软件配置管理,软件人员组织与管理,软件知识产权保护等。

本书是“中国计算机软件专业技术软件(高级)”水平考试的必读教材,也可作为相应培训班的教材,通过该级考试的考生具有软件工程高级工程师的相应水平。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

书 名: 软件工程(高级)

主 编: 郑人杰

出版者: 清华大学出版社(北京清华大学校内,邮编 100084)

<http://www.tup.tsinghua.edu.cn>

印刷者: 北京市清华园胶印厂

发行者: 新华书店总店北京发行所

开 本: 787×1092 1/16 **印张:** 25.25 **字数:** 596 千字

版 次: 1999 年 8 月 第 1 版 1999 年 8 月 第 1 次印刷

书 号: ISBN 7-302-03534-2/TP·1937

印 数: 00001~31000

定 价: 37.00 元

序

科学技术的日新月异,信息化时代的来临,使以计算机技术为基础的信息科学在经济和社会生活各个领域得到了极为广泛的应用,其发展水平成为衡量国家经济与科技实力的重要标志已是不争之事实。各国都把培养大量高水平计算机专业人才作为 21 世纪经济和科技发展的重要战略目标之一。一些经济发达国家通过开展对计算机专业人才的教育培训,尤其是开展不同层次、不同规模的计算机水平测试,吸引、储备了大量计算机高级人才,为迎接日趋激烈的科技竞争奠定了坚实基础。这些国家的成功经验值得我们学习、借鉴。

中国计算机软件专业技术资格和水平考试自 1991 年开始实施至今已走过了 8 年的历程,共有近 40 万人参加考试,在国内外已产生较大影响。特别是 1999 年度中国计算机软件专业技术水平考试将由原来的一个专业扩展为程序设计、软件工程、数据库技术、计算机网络、多媒体技术 5 个专业,这无疑是中国计算机软件专业技术水平考试发展的一个质的飞跃,必将把中国计算机软件专业技术水平考试推向新的阶段。

我相信,新近编写出版的《中国计算机软件专业技术水平考试指定用书》能对广大应试者起到很好的指导作用。

我更加希望,在世纪之交,中国计算机软件专业技术水平考试能够抓住机遇,迎接竞争与挑战,为促进我国科教兴国战略的贯彻实施做出应有的贡献。

赵东亮

编者的话

软件工程是个年轻的学科,但它具有鲜明的实践性。近年来在计算机领域中,它的地位显得越来越重要,不仅大型软件项目离不开它,就是一般的,甚至小型软件项目也必须运用它的概念、原则和方法。软件分析人员、软件设计、编程、测试、维护人员,以及软件管理人员都必须很好地掌握软件工程知识,才能适应岗位工作的要求。特别是在当前,软件产业已经被中央、各地方政府和有关部门当作国民经济中新的增长点的形势下,软件工程更加受到重视。同时,事实也一再表明,不掌握软件工程知识,不按照软件工程的要求去做软件项目,提供给用户的产品只能是低质量的、难于维护的。

这套软件工程教材是为适应软件人员水平考试要求编写的。分为初级、中级和高级三册,分别面向软件的编程人员、以软件设计与开发为主的软件工程师及软件分析人员或软件管理人员。

本书是软件工程的高级读物,全书分为三个部分,即软件工程技术、软件质量管理与质量保证及软件工程管理。其中第3章“软件复用技术”由中科院软件所研究员卢慧琼编写,第6章“软件工具与软件开发环境”由上海复旦大学计算机系教授钱乐秋、任杰编写,第11章“软件过程能力评估”由中国航天工业总公司第二研究院计算机应用和仿真技术研究所研究员王伟编写,第15章15.2“软件人员能力成熟度模型”一节由华北电力大学信息系副教授徐磊编写,15.3“软件工程师道德和职业活动规范”一节由高等教育出版社副编审刘建元编写,第16章“软件知识产权保护”由中国软件登记中心主任李维编写。其余各章由郑人杰编写。

限于时间过于紧促,书中不当之处欢迎读者和同行批评指正,以利改进和提高。

郑人杰

1999年3月于清华园

目 录

第 1 篇 软件工程技术

第 1 章 软件生存期过程和软件工程	1	开发过程	29
1.1 软件生存期及其模型	1	3.2.3 领域工程和应用系统工程	31
1.1.1 软件生存期	1	3.3 构件技术	33
1.1.2 软件生存期模型	2	3.3.1 应用系统和应用系统族	33
1.2 软件生存期过程	6	3.3.2 应用系统与构件	33
1.3 软件工程的基本目标	9	3.3.3 构件系统	34
1.3.1 软件工程的定义	9	3.3.4 构件系统的门面	35
1.3.2 软件工程项目的目标	10	3.3.5 可变性和专门化	36
第 2 章 软件需求分析	12	3.3.6 打包和编写文档	37
2.1 软件需求分析的任务	12	3.4 分层式体系结构	37
2.2 需求分析的过程	13	3.4.1 软件体系结构	37
2.3 软件需求分析的原则	17	3.4.2 良好的软件体系结构的 重要作用	37
2.4 分析员和用户责任	19	3.4.3 分层式的体系结构	38
2.5 软件需求分析方法	19	3.5 渐进地实施复用和复用单位的 组织结构	40
2.6 软件需求分析工具	21	3.5.1 软件复用需要改变开发 单位的组织结构	40
2.6.1 SADT	22	3.5.2 渐进地系统地采用 复用技术	41
2.6.2 PSL/PSA	23	3.5.3 充分利用可共享复 用成果	46
2.7 面对确定需求的困难应 采取的对策	24	3.5.4 实施系统复用需要 遵循的原则	46
2.7.1 需求工程和确定 需求的困难	24		
2.7.2 软件开发人员面对确定需求 的困难应采取的对策	25		
第 3 章 软件复用技术	26	第 4 章 软件测试	48
3.1 软件复用的概述	26	4.1 软件测试基础	48
3.1.1 软件复用的宗旨	26	4.1.1 什么是软件测试	48
3.1.2 软件复用的实际效益	27	4.1.2 软件测试的目的和原则	48
3.1.3 软件复用技术的发展概况	27	4.1.3 软件测试的对象	50
3.2 软件开发过程	28	4.1.4 测试信息流	51
3.2.1 以往的软件开发技术不能 满足复用的需要	28	4.1.5 测试与软件开发各 阶段的关系	52
3.2.2 软件复用需要改变软件		4.2 两种类型的测试	53

4.2.1 黑盒测试	53	5.3 软件维护的实施	112
4.2.2 白盒测试	54	5.3.1 分析和理解程序	112
4.3 白盒测试的测试用例设计	55	5.3.2 修改程序	113
4.3.1 逻辑覆盖	55	5.3.3 重新验证程序	115
4.3.2 基本路径测试	59	5.4 软件可维护性	117
4.4 黑盒测试的测试用例设计	65	5.4.1 软件可维护性的定义	117
4.4.1 等价类划分	65	5.4.2 可维护性的度量	118
4.4.2 边界值分析	68	5.5 提高可维护性的方法	119
4.4.3 错误推测法	71	5.5.1 建立明确的软件质量 目标和优先级	120
4.4.4 因果图	72	5.5.2 使用提高软件质量的 技术和工具	120
4.4.5 功能图	75	5.5.3 进行明确的质量保证审查	121
4.5 软件测试的策略	79	5.5.4 选择可维护的程序 设计语言	124
4.5.1 单元测试	80	5.5.5 改进程序的文档	124
4.5.2 组装测试	82	5.5.6 开发软件时考虑到维护	125
4.5.3 确认测试	87	5.6 软件再工程	126
4.5.4 系统测试	89	5.6.1 什么是软件再工程	126
4.5.5 测试的步骤及相应的 测试种类	89	5.6.2 为什么要实施软件再工程	129
4.6 程序的静态分析方法	93	5.6.3 软件再工程技术	131
4.6.1 对程序的静态分析	93	5.6.4 软件再工程的风险	134
4.6.2 人工测试	95	第6章 软件工具与软件开发环境	136
4.7 软件测试工具	97	6.1 软件工具	136
4.7.1 静态分析工具	97	6.1.1 概述	136
4.7.2 动态测试工具	98	6.1.2 软件开发工具	138
4.7.3 测试数据自动生成工具	100	6.1.3 软件维护工具	143
4.7.4 模块测试台	101	6.1.4 软件管理和软件支持 工具	144
4.7.5 测试合成环境	102	6.1.5 软件开发工具的评价 和选择	145
第5章 软件维护与软件再工程	105	6.2 软件开发环境	146
5.1 软件维护的概念	105	6.2.1 概述	146
5.1.1 软件维护的定义	105	6.2.2 集成型软件开发环境	147
5.1.2 影响维护工作量的因素	107	6.2.3 ECMA/NIST 集成型软件 开发环境参考模型	150
5.1.3 软件维护的策略	108	6.2.4 PCTE: 可移植公共 工具环境	153
5.1.4 维护成本	108	6.2.5 青鸟系统	153
5.2 软件维护活动	109		
5.2.1 维护机构	109		
5.2.2 软件维护申请报告	110		
5.2.3 软件维护工作流程	110		
5.2.4 维护档案记录	111		
5.2.5 维护评价	112		

第2篇 软件质量管理与质量保证

第7章 软件质量	153	8.7.2 估算软件中错误总数 E_T 的方法	200
7.1 软件危机尚未过去	153	8.7.3 测试精确度和测试覆盖度 的评价	201
7.1.1 软件发展远远落后于硬件	153	8.7.4 测试开始时的预测模型	203
7.1.2 软件质量问题提出的挑战	156	8.8 软件容错技术	205
7.1.3 有银弹吗?	157	8.8.1 什么是容错软件	206
7.2 软件质量问题的根源	158	8.8.2 容错的一般方法	206
7.2.1 软件不同于硬件或 其他产品	158	8.8.3 容错软件的设计过程	209
7.2.2 影响软件质量的因素	159	8.8.4 软件的容错系统结构	209
7.3 什么是软件质量	161	8.9 软件过程改进	212
7.3.1 通常的理解	161	8.9.1 软件过程改进的含意和 现有的方案	212
7.3.2 McCall 的质量特性	162	8.9.2 软件过程改进模式	213
7.3.3 国际标准和国家标准规定的 质量特性	163		
7.4 产品质量与过程质量	166		
第8章 软件质量保证	169	第9章 软件工程标准化和软件文档	215
8.1 软件质量保证概述	169	9.1 什么是软件工程标准	215
8.1.1 质量保证的概念	169	9.2 软件工程标准化的意义	217
8.1.2 软件质量保证的主要任务	170	9.3 软件工程标准的制订与推行	218
8.1.3 质量保证与检验	171	9.4 软件工程标准的层次	219
8.2 软件质量保证体系	172	9.5 软件工程国家标准	220
8.3 质量保证的实施	176	9.6 在开发机构中推行软件工程 标准	225
8.3.1 质量目标与度量	176	9.7 软件文档	225
8.3.2 质量度量方法	176	9.7.1 软件文档的作用和分类	225
8.3.3 软件质量管理小组	179	9.7.2 对文档编制的质量要求	228
8.4 软件的质量设计	179	9.7.3 文档的管理和维护	232
8.4.1 质量特性转换为软件的 内部结构	180	第10章 在软件开发机构中贯彻 ISO 9000 国际标准	234
8.4.2 软件的质量展开	183	10.1 质量管理、质量认证与质量审核	234
8.5 技术评审	186	10.1.1 质量管理	234
8.5.1 设计质量的评审	186	10.1.2 质量认证与审核	236
8.5.2 程序质量的评审	191	10.2 ISO 9000 国际标准简介	237
8.6 软件可靠性	195	10.2.1 ISO 9000 标准概述	237
8.6.1 软件生存期与软件 寿命的关系	195	10.2.2 ISO 9000 标准的特点	238
8.6.2 软件可靠性的定义	197	10.2.3 ISO 9000 标准的 科学依据	239
8.6.3 软件可靠性的主要指标	198	10.3 ISO 9000 族标准的构成	241
8.7 测试中的可靠性分析	198	10.4 质量体系	243
8.7.1 推测错误的产生频度	198		

10.5	ISO 9001 标准的主要内容	245	11.1.2	降低软件风险的需要	265
10.6	ISO 9000-3 标准简介	249	11.2	软件过程评估方法的产生	265
10.6.1	理解标准与指南的关系	249	11.3	软件能力成熟度模型 CMM	
10.6.2	理解 ISO 9000-3 对 20 个质量 体系要素的解释	250		简介	266
10.6.3	理解 ISO 9000-3 与 ISO/ IEC 12207 两个标准之间 的关系	253	11.3.1	模型概要	266
10.7	软件开发机构为什么要按 ISO 9000 标准建立并实施质量保证体系	255	11.3.2	模型的产生和原理	268
10.7.1	软件质量缺陷不可能 完全避免	255	11.3.3	不成熟和成熟软件 组织的比较	268
10.7.2	技术上解决软件质量问题的 局限性	257	11.3.4	软件过程成熟度的 5 个等级	269
10.7.3	为什么软件开发机构要加强 质量管理	258	11.3.5	跳越成熟度等级	272
10.7.4	为什么软件开发机构要开展 ISO 9000 质量体系认证工作	259	11.3.6	关键过程域	273
10.8	软件开发机构实施 ISO 9000 标准 应做的工作	260	11.3.7	关键实践	275
10.9	若干认识问题	262	11.3.8	CMM 的应用	277
第 11 章	软件过程能力评估	264	11.3.9	软件过程成熟度提问单	279
11.1	软件过程评估的意义	264	11.3.10	对 CMM 1.1 的 几点考虑	280
11.1.1	软件过程改进的需要	264	11.4	软件过程评估的国际标准概述	284
			11.4.1	软件过程评估国际 标准的制定	284
			11.4.2	软件过程评估标准的 组成	285
			11.4.3	参考模型	286
			11.4.4	评估框架	290
			11.4.5	软件过程评估标准的特点	292

第 3 篇 软件工程管理

第 12 章	软件工程项目管理	296	12.3.4	自动估算工具	314
12.1	软件工程项目管理的任务	296	12.4	风险分析	316
12.2	软件项目估算	298	12.4.1	风险识别	316
12.2.1	针对估算的考虑	298	12.4.2	风险估计	317
12.2.2	软件项目计划的目标	299	12.4.3	风险评价	318
12.2.3	软件的范围	299	12.4.4	风险驾驭和监控	318
12.2.4	软件开发中的资源	300	12.5	进度安排	320
12.2.5	软件项目估算	303	12.5.1	软件开发小组人数与软件 生产率	321
12.2.6	分解技术	303	12.5.2	任务的确定与并行性	322
12.3	软件开发成本估算	307	12.5.3	制定开发进度计划	322
12.3.1	软件开发成本估算方法	307	12.5.4	进度安排的图形方法	324
12.3.2	专家判定技术	308	12.5.5	项目的追踪和控制	326
12.3.3	软件开发成本估算的 经验模型	309	12.6	软件项目的组织与计划	327

12.6.1 软件项目管理的特点	327	15.2.5 成熟度级别和关键 过程域	361
12.6.2 制定计划	328	15.2.6 主题	365
第 13 章 软件度量	331	15.2.7 关键过程域的目标	366
13.1 软件度量的概念	331	15.2.8 软件人员能力成熟度 模型的应用	368
13.2 功能点方法计算软件的大小	332	15.3 软件工程师道德和职业 活动规范	370
13.3 程序环路复杂度计算	335	15.3.1 引言	370
13.4 霍尔斯特德(Halstead)程序 工作量计算	336	15.3.2 软件开发项目的特点、影响 及其与人的关系	370
13.5 程序风格度量	337	15.3.3 规范的内容	372
第 14 章 软件配置管理	339	15.3.4 职业道德教育的作用 及其重要性	376
14.1 什么是软件配置管理	339	第 16 章 软件知识产权保护	377
14.1.1 软件配置管理的几种 定义	339	16.1 软件知识产权的保护必须 依法实施	377
14.1.2 什么是软件配置项	340	16.1.1 知识产权的法律框架	377
14.1.3 软件配置管理的任务	340	16.1.2 我国保护计算机软件的 法律制度	378
14.2 软件配置标识	341	16.2 计算机软件著作权	379
14.3 变更管理	343	16.2.1 计算机软件著作权的 主体	379
14.4 版本控制	347	16.2.2 计算机软件著作权的 客体	379
14.4.1 版本管理和发行管理	347	16.2.3 计算机软件著作权的 权利内容	380
14.4.2 版本标识	347	16.2.4 计算机软件著作权归属	381
14.4.3 发行管理	348	16.2.5 软件著作权的行使	381
14.5 系统建立	349	16.2.6 软件专有权利的限制	382
14.6 配置审核	350	16.3 计算机软件著作权登记管理	383
14.7 配置状态报告	350	16.3.1 计算机软件著作权登记制度 的法律功能和作用	383
第 15 章 软件人员组织与管理	352	16.3.2 计算机软件著作权登记 的几种主要形式	384
15.1 软件项目的人员组织与管理	352	16.4 计算机软件著作权侵权与法律 保护	384
15.1.1 项目组的组织结构	352	16.4.1 软件著作权侵权行为 类型	384
15.1.2 人员配备	355	16.4.2 侵犯软件著作权的 法律责任	385
15.1.3 指导与检验	357		
15.2 软件人员能力成熟度模型	358		
15.2.1 引入软件人员能力成熟 度模型的必要性	358		
15.2.2 软件人员能力成熟度 模型的发展溯源	359		
15.2.3 软件人员能力成熟度 模型简介	359		
15.2.4 软件人员能力成熟度 模型的结构	360		

16.4.3 法定的例外情况	385	16.5.3 侵害计算机软件商业 秘密的不正当行为与 法律责任	388
16.5 计算机软件的商业秘密与 反不正当竞争	386		
16.5.1 商业秘密的法律特征	386		
16.5.2 计算机软件与商业秘密	387	参考文献	391

第 1 篇 软件工程技术

第 1 章 软件生存期过程和软件工程

在计算机软件的发展历史上曾经出现过若干个不同的时期,各个发展的历史时期反映了人们对软件的认识在逐步扩展。

在 60 年代以前甚至还没有软件开发的说法,那时只有程序设计(programming)的概念,最多在写出程序时配有程序结构说明和使用说明。

70 年代以来人们认识到软件的工作不能只是编制程序,于是出现了软件生存期的概念。认为软件开发工作在程序编制之前和在其之后还有许多重要的工作不能忽视,例如需求分析、测试等等。与此同时,在总结“软件危机”的教训时,认识到必须建立软件工程的观念,摒弃了那种只有充满编程技巧的程序才能高水平地发挥个人才能的观念,强调程序的可读性、可理解性、可测试性和可修改性等工程原则。

90 年代开始人们更加强调软件开发的效率和软件的质量以及与其相关的软件管理。软件开发固然是重要的、主要的软件工作,但一些支持性、管理性的工作也不可忽视,于是建立了软件过程的概念。软件工程过程或软件生存期过程使我们更全面、系统和深刻地理解了软件的有关工作。

本章将在软件生存期及其模型的基础上,围绕国际标准阐述软件生存期过程的有关内容。最后简要讨论软件工程的目标和软件工程的定义。

1.1 软件生存期及其模型

1.1.1 软件生存期(life cycle)

同任何事物一样,软件也有一个孕育、诞生、成长、成熟、衰亡的许多阶段,一般称其为计算机软件的生存期。根据这一思想,把上述基本的过程活动进一步展开,可以得到软件生存期的 6 个阶段工作,即制定计划、需求分析、设计、程序编制、测试及运行维护。以下对这 6 个阶段的任务作一概括的描述。

1. 制定计划 (planning)

确定要开发软件系统的总目标,给出它的功能、性能、可靠性以及接口等方面的要求;由系统分析员和用户合作,研究完成该项软件任务的可行性,探讨解决问题的可能方案,

并对可利用的资源(计算机硬件、软件、人力等)、成本、可取得的效益、开发的进度做出估计,制定出完成开发任务的实施计划,连同可行性研究报告,提交管理部门审查。

2. 需求分析和定义 (requirement analysis and definition)

对待开发软件提出的需求进行分析并给出详细的定义。软件人员和用户共同讨论决定:哪些需求是可以满足的,并对其加以确切地描述。然后编写出软件需求说明书或系统功能说明书,及初步的系统用户手册,提交管理机构评审。

3. 软件设计 (software design)

设计是软件工程的技术核心。在设计阶段中,设计人员把已确定的各项需求转换成一个相应的体系结构。结构中的每一组成部分都是意义明确的模块,每个模块都和某些需求相对应,即概要设计。进而对每个模块要完成的工作进行具体的描述,为源程序编写打下基础,即详细设计。所有设计中的考虑都应以设计说明书的形式加以描述,以供后继工作使用并提交评审。

4. 程序编写 (coding/programming)

把软件设计转换成计算机可以接受的程序代码,即写成以某一种特定程序设计语言表示的“源程序清单”。这一步工作也称为编码。自然,写出的程序应当是结构良好、清晰易读的,且与设计相一致的。

5. 软件测试 (testing)

测试是保证软件质量的重要手段,其主要方式是在设计测试用例的基础上检验软件的各个组成部分。首先是进行单元测试,查找各模块在功能和结构上存在的问题并加以纠正;其次是进行组装测试,将已测试过的模块按一定顺序组装起来;最后按规定的各项需求,逐项进行有效性测试,决定已开发的软件是否合格,能否交付用户使用。

6. 运行/维护 (running/maintenance)

已交付的软件投入正式使用,便进入运行阶段。这一阶段可能持续若干年甚至几十年。软件在运行中可能由于多方面的原因,需要对其进行修改。其原因可能有:运行中发现了软件中的错误需要修正;为了适应变化了的软件工作环境,需做适当变更;为了增强软件的功能需做变更。

1.1.2 软件生存期模型 (Software Life Cycle Model)

类似于其他工程项目中安排各道工序那样,为反映软件生存期内各种活动应如何组织,上节所给出的 6 个步骤应如何衔接,需要用软件生存期模型做出直观的图示表达。

软件生存期模型是从软件项目需求定义直至软件经使用后废弃为止,跨越整个生存期的系统开发、运行和维护所实施的全部过程、活动和任务的结构框架。

到现在为止,已经提出了多种软件生存期模型。例如,瀑布模型、演化模型、螺旋模型、喷泉模型和智能模型等。以下对这些模型作一简要描述。

1. 瀑布模型 (Waterfall Model)

瀑布模型规定了各项软件工程活动,包括:制定开发计划、进行需求分析和说明、软件设计、程序编码、测试及运行维护。参看图 1.1。并且规定了它们自上而下、相互衔接的固定次序,如同瀑布流水,逐级下落。

然而软件开发的实践表明,上述各项活动之间并非完全是自上而下,呈线性图式。实际情况是,每项开发活动均应具有以下特征:

- (1) 从上一项活动接受该项活动的工作对象,作为输入;
- (2) 利用这一输入实施该项活动应完成的内容;
- (3) 给出该项活动的工作成果,作为输出传给下一项活动;
- (4) 对该项活动实施的工作进行评审。若其工作得到确认,则继续进行下一项活动,在图 1.1 中用向下指的箭头表示;否则返回前项,甚至更前项的活动进行返工,在图 1.1 中由向上指的箭头表示。

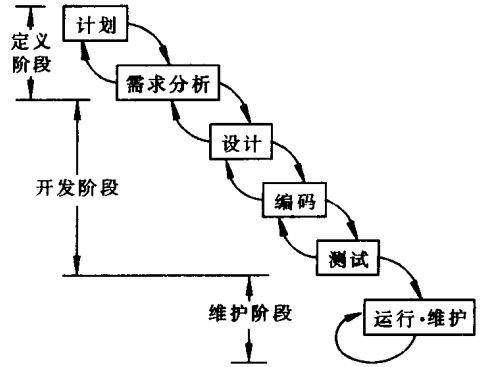


图 1.1 软件生存期的瀑布模型

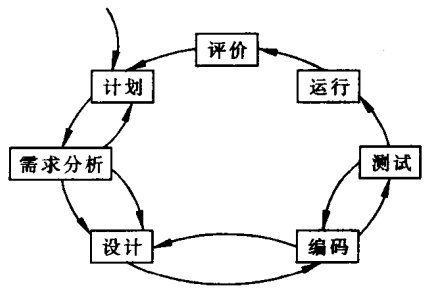


图 1.2 软件生存期循环

需要注意:软件维护在软件生存期中有它的特点。一方面,维护的具体要求是在软件投入运行以后提出来的,经过“评价”,确定变更的必要性,才进入维护工作。另一方面,维护中对软件的变更仍然要经历上述软件生存期在开发中已经历过的各项活动。如果把这些活动一并表达,就构成了生存期循环,如图 1.2 所示。事实上,有人把维护称为软件的二次开发,正是出于这种考虑。由于软件在投入使用以后可能经历多次变更,为把开发活动和维护活动区别开来,便有了 b 形的软件生存期表示,如图 1.3 所示。它与前述的软件生存期循环一样,都是软件生存期瀑布模型的变种。

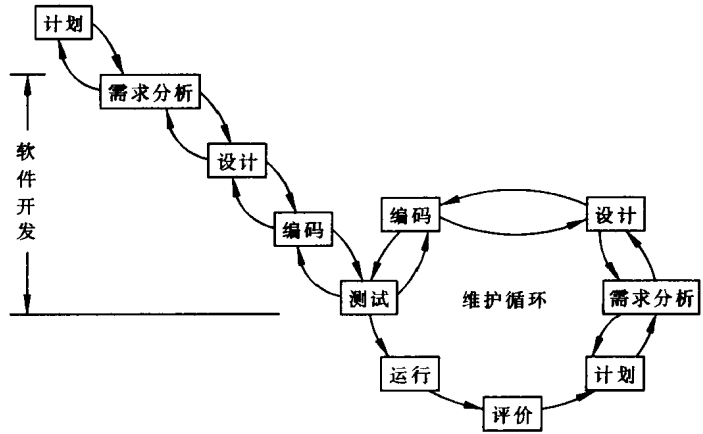


图 1.3 具有维护循环的软件生存期

瀑布模型为软件开发和软件维护提供了一种有效的管理图式。根据这一图式制定开发计划、进行成本预算、组织开发力量,以项目的阶段评审和文档控制为手段有效地对整个开发过程进行指导,从而保证了软件产品及时交付,并达到预期的质量要求。瀑布模型20多年来之所以广为流行,是因为它在消除非结构化软件、降低软件的复杂度、促进软件开发工程化方面起着显著作用。与此同时,瀑布模型在大量的软件开发实践中也逐渐暴露出它的严重缺点。其中最为突出的缺点是该模型缺乏灵活性,特别是无法解决软件需求不明确或不准确的问题。这些问题的存在对软件开发会带来严重影响,最终可能导致开发出的软件并不是用户真正需要的软件,并且这一点在开发过程完成后才有所察觉。面对这些情况,无疑需要进行返工或是不得不在维护中纠正需求的偏差。但无论上述哪一种情况都必须付出高额的代价,并将为软件开发带来不必要的损失。另一方面,随着软件开发项目规模的日益庞大,由于瀑布模型不够灵活等缺点引发出的上述问题显得更为严重。

为弥补瀑布模型的不足,近年来已经提出了多种其他模型。

2. 演化模型 (Evolutional Model)

由于在项目开发的初始阶段人们对软件的需求认识常常不够清晰,因而使得开发项目难于做到一次开发成功,出现返工再开发在所难免。有人说,往往要“干两次”后开发出的软件才能较好地令用户满意。第一次只是试验开发,其目标只是在于探索可行性,弄清软件需求;第二次则在此基础上获得较为满意的软件产品。通常把第一次得到的试验性产品称为“原型”。显然,演化模型在克服瀑布模型缺点、减少由于软件需求不明确而给开发工作带来风险方面,确有显著的效果。

3. 螺旋模型 (Spiral Model)

对于复杂的大型软件,开发一个原型往往达不到要求。螺旋模型将瀑布模型与演化模型结合起来,并且加入被两种模型都忽略了的风险分析,弥补了两者的不足。

在此先简要说明什么是风险分析。“软件风险”是普遍存在于任何软件开发项目中的实际问题。对于不同的项目,其差别只是风险有大有小而已。在制定软件开发计划时,系统分析员必须回答:项目的需求是什么,需要投入多少资源以及如何安排开发进度等一系列问题。然而,若要他们当即给出准确无误的回答是不容易的,甚至是不可能的。但系统分析员又不可能完全回避这一问题。凭借经验的估计给出初步的设想便难免带来一定风险。实践表明,项目规模越大,问题越复杂,资源、成本、进度等因素的不确定性越大,承担项目所冒的风险也越大。总之,风险是软件开发不可忽视的潜在不利因素,它可能在不同程度上损害到软件开发过程或软件产品的质量。软件风险驾驭的目标是在造成危害之前,及时对风险进行识别、分析,采取对策,进而消除或减少风险的损害。

螺旋模型沿着螺旋线旋转,如图 1.4 所示,在笛卡尔坐标的 4 个象限上分别表达了 4 个方面的活动,即:

- (1) 制定计划: 确定软件目标,选定实施方案,弄清项目开发的限制条件;
- (2) 风险分析: 分析所选方案,考虑如何识别和消除风险;
- (3) 实施工程: 实施软件开发;
- (4) 客户评估: 评价开发工作,提出修正建议。

沿螺旋线自内向外每旋转一圈便开发出更为完善的一个新的软件版本。例如,在第一

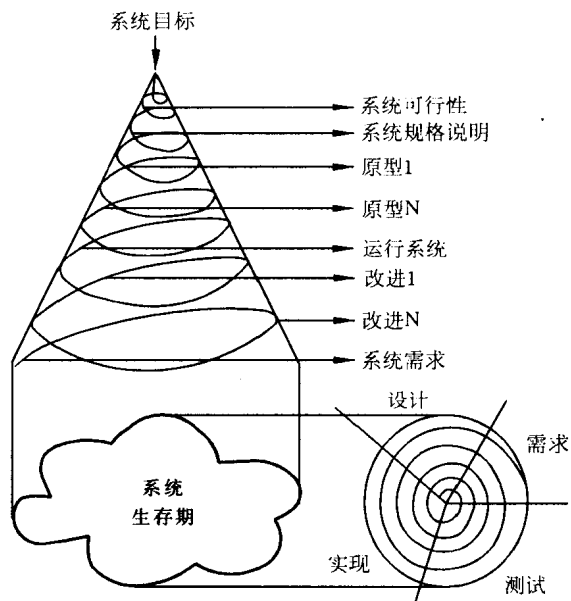


图 1.5 螺旋模型的另一图式

瀑布模型的另一个不足之处在于,它对软件复用和生存期中多项开发活动的集成并未提供支持,因而难于支持面向对象的开发方法。“喷泉”一词体现了迭代和无间隙特性。喷泉模型在系统某个部分常常重复工作多次,相关功能在每次迭代中随之加入演进的系统。无间隙是指在开发活动,即分析、设计和编码之间不存在明显的边界。

5. 智能模型 (Intelligence Model)

智能模型也称为基于知识的软件开发模型,它综合了上述若干模型,并把专家系统结合在一起。该模型应用基于规则的系统,采用归约和推理机制,帮助软件人员完成开发工作,并使维护在系统规格说明一级进行。为此,建立了知识库,将模型、软件工程知识与特定领域的知识分别存入数据库。以软件工程知识为基础的生成规则构成的专家系统与含有应用领域知识规则的其他专家系统相结合,构成了这一应用领域软件的开发系统。

1.2 软件生存期过程

软件生存期概念的出现曾经帮助我们较为全面地认识软件开发(包括软件的运行与维护)。这主要表现在1988年制订和公布的国家标准《GB 8566-88 计算机软件开发规范》中,该标准曾将软件生存期划分为8个阶段,即:可行性研究和计划、需求分析、概要设计、详细设计、实现、组装测试、确认测试、使用和维护。

该标准为每个阶段规定了任务、实施步骤、实施要求以及完成的标志。对软件生存期按此方式做8个阶段的划分大致符合也适应前述的瀑布模型,反映了80年代人们对软件工程的认识。

此后,于20世纪90年代初提出了软件工程过程的概念。认为软件工程过程是为了获