

全国计算机等级 考试四级教程



教育部考试中心

——软件工程
(2017年版)

高等教育出版社

全国计算机等级考试四级教程

——软件工程

(2017年版)

Quanguo Jisuanji Dengji Kaoshi Siji Jiaocheng
——Ruanjian Gongcheng

教育部考试中心

主编 殷人昆

高等教育出版社·北京

内容提要

本书根据教育部考试中心制订的《全国计算机等级考试四级软件工程考试大纲(2013年版)》编写而成。主要内容包括软件工程概念、面向对象的基本概念与UML、软件需求分析、软件设计、程序实现、软件测试、软件维护、软件过程、软件项目管理、软件质量管理、软件工程标准化与软件文档等。本书每章前面有考核目标,后面有小结,有助于读者掌握该章的主要精髓。书后包含考试样题及参考答案,以便考生考试复习并通过考试。

本书可供报考全国计算机等级考试四级软件工程的考生使用,也可用作普通高等学校软件工程课程教材或参考书,还可作为社会读者学习用书。

图书在版编目(CIP)数据

全国计算机等级考试四级教程:2017年版. 软件工程 / 教育部考试中心编. --北京:高等教育出版社, 2016.10

ISBN 978-7-04-046579-2

I. ①全… II. ①教… III. ①电子计算机-水平考试-教材②软件工程-水平考试-教材 IV. ①TP3

中国版本图书馆CIP数据核字(2016)第235131号

策划编辑 何新权
责任校对 殷然

责任编辑 何新权
责任印制 耿轩

封面设计 王琰

版式设计 范晓红

出版发行	高等教育出版社	网 址	http://www.hep.edu.cn
社 址	北京市西城区德外大街4号		http://www.hep.com.cn
邮政编码	100120	网上订购	http://www.hepmall.com.cn
印 刷	大厂益利印刷有限公司		http://www.hepmall.com
开 本	787mm×1092mm 1/16		http://www.hepmall.cn
印 张	16.5		
字 数	400千字	版 次	2016年10月第1版
购书热线	010-58581118	印 次	2016年10月第1次印刷
咨询电话	400-810-0598	定 价	35.00元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换

版权所有 侵权必究

物料号 46579-00

积极发展全国计算机等级考试 为培养计算机应用专门人才、促进信息 产业发展作出贡献 (序)

中国科协副主席 中国系统仿真学会理事长
第五届全国计算机等级考试委员会主任委员
赵沁平

当今,人类正在步入一个以智力资源的占有和配置,知识生产、分配和使用为最重要因素的知识经济时代,也就是小平同志提出的“科学技术是第一生产力”的时代。世界各国的竞争已成为以经济为基础、以科技(特别是高科技)为先导的综合国力的竞争。在高科技中,信息科学技术是知识高度密集、学科高度综合、具有科学与技术融合特征的学科。它直接渗透到经济、文化和社会的各个领域,迅速改变着人们的工作、生活和社会的结构,是当代发展知识经济的支柱之一。

在信息科学技术中,计算机硬件及通信设施是载体,计算机软件是核心。软件是人类知识的固化,是知识经济的基本表征,软件已成为信息时代的新型“物理设施”。人类抽象的经验、知识正逐步由软件予以精确地体现。在信息时代,软件是信息化的核心,国民经济和国防建设、社会发展、人民生活都离不开软件,软件无处不在。软件产业是增长快速的朝阳产业,是具有高附加值、高投入高产出、无污染、低能耗的绿色产业。软件产业的发展将推动知识经济的进程,促进从注重量的增长向注重质的提高方向发展。软件产业是关系到国家经济安全和文化安全,体现国家综合实力,决定 21 世纪国际竞争地位的战略产业。

为了适应知识经济发展的需要,大力促进信息产业的发展,需要在全民中普及计算机的基本知识,培养一批又一批能熟练运用计算机和软件技术的各行各业的应用型人才。

1994 年,国家教委(现教育部)推出了全国计算机等级考试,这是一种专门评价应试人员对计算机软硬件实际掌握能力的考试。它不限制报考人员的学历和年龄,从而为培养各行业计算机应用人才开辟了一条广阔的道路。

1994 年是推出全国计算机等级考试的第一年,当年参加考试的有 1 万余人,2012 年报考人数已达 549 万人。截至 2013 年年底,全国计算机等级考试共开考 38 次,考生人数累计达 5 422 万人,有 2 067 万人获得了各级计算机等级证书。

事实说明,鼓励社会各阶层人士通过各种途径掌握计算机应用技术,并通过等级考试对他们的能力予以科学、公正、权威性的认证,是一种比较好的、有效的计算机应用人才培养途径,符合我国的具体国情。等级考试同时也为用人单位录用和考核人员提供了一种测评手段。从有关公司对等级考试所作的社会抽样调查结果看,不论是管理人员还是应试人员,对该项考试的内容和

形式都给予了充分肯定。

计算机技术日新月异。全国计算机等级考试大纲顺应技术发展和社会需求的变化,从2010年开始对新版考试大纲进行调研和修订,在考试体系、考试内容、考试形式等方面都做了较大调整,希望等级考试更能反映当前计算机技术的应用实际,使培养计算机应用人才的工作更健康地向前发展。

全国计算机等级考试取得了良好的效果,这有赖于各有关单位专家在等级考试的大纲编写、试题设计、阅卷评分及效果分析等多项工作中付出的大量心血和辛勤劳动,他们为这项工作的开展作出了重要的贡献。我们在此向他们表示衷心的感谢!

我们相信,在21世纪知识经济和加快发展信息产业的形势下,在教育部考试中心的精心组织领导下,在全国各有关专家的大力配合下,全国计算机等级考试一定会以“激励引导成才,科学评价用才,服务社会选材”为目标,服务考生和社会,为我国培养计算机应用专门人才的事业作出更大的贡献。

前 言

随着计算机和网络深入到国民经济的各个部门,应用到人们日常生活的各个角落,人们对于各种软件的需求呈现爆炸式的增长趋势,如何更快、更好、更多、更方便地开发软件,已成为软件开发人员面临的主要任务。软件工程越来越受到人们的关注。现在,在大学里,几乎所有学科或多或少与软件工程发生关系。

从历史来看,软件工程是计算机科学与技术发展的一个重要分支,所涉及的范围非常广泛,包括软件开发技术、软件工程环境、软件工程经济学、软件心理学、软件项目管理等许多方面的知识。为了推进高质量软件工程人才的培养,教育部考试中心决定在计算机等级考试中实施“软件工程”(四级)的考试,它的作用有两个,一是为促进软件工程技术和方法的普及和应用;二是为取得后续软件工程师类考试资格。

教育部考试中心在设计实施这项考试中,得到了有关计算机专家的热情支持和大力帮助。专家们对考试大纲、内容及考试的组织实施都进行了认真研究和精心设计。为便于考生学习和考试,考试中心组织编写了本书。

本书的内容分别与考试大纲的各个知识点对应。主要包括软件工程概论、面向对象的基本概念与UML、软件需求分析、软件设计、软件实现、软件测试、软件维护、软件过程、软件项目管理、软件质量管理、软件工程标准化与软件文档等。

本书前第1、2两章注重软件工程基本概念的介绍;第3~6章基于软件生存周期分别说明软件开发和维护时期所用到的原则、方法、过程和文档;第7~11章侧重软件过程、管理和质量有关的思想、活动、标准、文档以及某些最佳实践。全书在内容组织方面注重科学性和系统性,在行文叙述方面则注重简洁性和可读性。

本书每章前面有考核目标,后面有小结,有助于读者掌握该章的精髓。最后有复习习题,但没有采用真题(选择题)形式,而是列表给出各章节要掌握的关键概念,提示读者学习的重点,帮助读者带着问题有目的地学习和复习。

由于编写时间仓促,书中恐有疏漏、不妥之处,敬请读者提出宝贵意见,以便修订时改进。

编 者

目 录

第1章 软件工程概论 1

1.1 软件和软件工程的概 念 1

1.1.1 软件的概念 1

1.1.2 软件危机 2

1.1.3 软件工程的概 念 3

1.2 软件工程方法 4

1.2.1 面向过程方法 4

1.2.2 面向对象方法 5

1.2.3 形式化方法 6

1.3 软件过程与软件生存周期 6

1.3.1 软件生存周期 6

1.3.2 软件过程 7

1.4 软件过程模型 8

1.5 软件工具概述 15

小结 18

复习要点 18

第2章 面向对象的基本概念与 UML 20

2.1 面向对象系统的基本概 念 20

2.1.1 面向对象系统的概 念 20

2.1.2 对象 21

2.1.3 类与封装 21

2.1.4 继承 22

2.1.5 多态与动态绑定 23

2.1.6 消息通信 23

2.2 统一建模语言 UML 概 述 23

2.2.1 UML 的产生和发展 24

2.2.2 UML 的特点 24

2.3 UML 的模型元素 24

2.3.1 UML 的事物 24

2.3.2 UML 中的关系 26

2.4 UML 中的图 29

2.4.1 外部视图 29

2.4.2 内部视图 31

小结 34

复习要点 34

第3章 软件需求分析 36

3.1 系统的概 念 36

3.1.1 基于计算机的系统 36

3.1.2 计算机系统工程 37

3.1.3 可行性研究 37

3.2 软件需求分析的任务和原 则 38

3.2.1 软件需求的定义和层 次 39

3.2.2 软件需求分析的任务 40

3.2.3 需求分析的原则 40

3.3 软件需求获取 41

3.3.1 需求获取的任务和原 则 41

3.3.2 需求获取的过程 42

3.3.3 需求的表达 43

3.4 结构化分析方法 43

3.4.1 数据建模 44

3.4.2 功能建模 45

3.4.3 行为建模 48

3.4.4 数据字典 48

3.4.5 基本加工逻辑说明 49

3.5 面向对象的分析方法 51

3.5.1 面向对象分析概 述 51

3.5.2 识别类或对象 52

3.5.3 识别关系(结构) 53

3.5.4 标识类的属性和服 务 54

3.6 需求规格说明和需求评 审 55

3.6.1 软件需求规格说明的 目标 55

3.6.2 软件需求规格说明编 制

原则 55

3.6.3 软件需求规格说明模 板 56

3.6.4 软件需求评审	57	4.7.1 软件设计规格说明	88
小结	58	4.7.2 软件概要设计评审	89
复习要点	58	4.7.3 软件详细设计评审	90
第4章 软件设计	60	小结	90
4.1 软件设计的任务和原则	60	复习要点	91
4.1.1 软件设计的概念	60	第5章 程序实现	93
4.1.2 软件设计的任务	60	5.1 程序实现的任务	93
4.1.3 软件设计的过程	61	5.2 结构化程序设计方法	94
4.1.4 软件设计的原则	61	5.2.1 自顶向下和逐步求精	94
4.2 结构化设计方法	65	5.2.2 使用基本控制结构构造程序	94
4.2.1 结构化设计与结构化分析的 关系	65	5.3 面向对象的程序设计方法	94
4.2.2 软件结构及表示工具	65	5.4 程序设计风格与编码规范	96
4.2.3 结构化设计的过程	67	5.5 编程语言的选择	99
4.2.4 变换流映射	67	5.5.1 编程语言特性的比较	100
4.2.5 事务流映射	68	5.5.2 编程语言的分类	101
4.2.6 软件模块结构改进	69	5.5.3 编程语言的选择	103
4.2.7 接口设计	70	5.6 程序复杂性	104
4.3 软件的过程设计	70	5.6.1 代码行度量法	104
4.3.1 部署设计	70	5.6.2 McCabe 度量法	105
4.3.2 算法设计	71	5.7 程序调试	106
4.3.3 处理过程的描述	71	5.7.1 程序调试的步骤	106
4.4 面向对象的系统设计	73	5.7.2 几种主要的调试方法	107
4.4.1 系统设计的目标和准则	73	5.7.3 程序调试的原则	109
4.4.2 子系统分解	73	小结	110
4.4.3 问题域部分的设计	74	复习要点	110
4.4.4 人机交互部分的设计	75	第6章 软件测试	112
4.4.5 任务管理部分的设计	76	6.1 软件测试的任务	112
4.4.6 数据管理部分的设计	77	6.2 软件测试方法	116
4.5 体系结构设计	79	6.2.1 白盒测试方法	116
4.5.1 体系结构的概念	79	6.2.2 黑盒测试方法	120
4.5.2 体系结构的4+1视图	80	6.2.3 其他测试方法	122
4.5.3 常用的体系结构风格	81	6.3 软件测试的策略	123
4.5.4 分布式系统体系结构	84	6.3.1 软件测试活动	123
4.6 对象设计	86	6.3.2 单元测试	123
4.6.1 使用模式设计对象	86	6.3.3 集成测试	124
4.6.2 接口规格说明设计	87	6.3.4 系统测试	127
4.6.3 重构对象设计模型	87	6.3.5 验收测试	128
4.6.4 优化对象设计模型	87	6.4 人工测试	129
4.7 软件设计规格说明与评审	88		

6.4.1 桌上检查	129	8.2.1 软件生存周期过程模型	150
6.4.2 代码检查	130	8.2.2 生存周期的基本过程	151
6.4.3 走查	131	8.2.3 生存周期的支持过程	151
小结	132	8.2.4 生存周期的组织过程	152
复习要点	132	8.3 软件过程成熟度模型	153
第7章 软件维护	134	8.3.1 软件过程成熟度	153
7.1 软件维护的任务	134	8.3.2 CMM 与 CMMI	154
7.1.1 软件维护的定义	134	8.3.3 CMMI 的分级表示	155
7.1.2 软件维护的类型	134	8.3.4 CMMI 的连续表示	158
7.2 软件维护的活动	135	8.3.5 CMMI 的模型构件	159
7.2.1 维护机制	135	8.3.6 CMMI 评估	161
7.2.2 软件维护申请报告	136	8.4 软件过程改进	162
7.2.3 软件维护过程模型	136	8.4.1 软件过程改进的 IDEAL	
7.2.4 GB/T 20157—2006 软件		模型	162
维护过程	137	8.4.2 软件过程改进框架	164
7.2.5 维护记录文档	138	8.4.3 有效的软件过程	165
7.3 程序修改的步骤及修改的		小结	165
副作用	138	复习要点	166
7.3.1 分析和理解程序	139	第9章 软件项目管理	167
7.3.2 评估修改范围	139	9.1 软件项目与项目管理概述	167
7.3.3 修改程序	140	9.1.1 项目和项目管理的概念	167
7.3.4 修改程序的副作用及其		9.1.2 项目管理的定义	168
控制	140	9.1.3 过程与项目管理	168
7.3.5 重新验证程序	141	9.2 软件项目计划与项目集成	
7.4 软件可维护性	141	管理	169
7.4.1 可维护性的定义	142	9.2.1 项目集成管理的概念	169
7.4.2 软件可维护性度量	142	9.2.2 项目计划制订的过程	169
7.5 软件演进与再工程	143	9.2.3 项目计划的执行和控制	172
7.5.1 遗留系统的演化	143	9.3 软件项目度量与工作量	
7.5.2 软件再工程	144	估算	174
7.5.3 遗留系统的现代化改造		9.3.1 软件度量的概念	174
过程	145	9.3.2 软件范围管理	176
7.5.4 重构与逆向工程	146	9.3.3 软件项目中的资源	177
小结	147	9.3.4 软件项目的工作量估算	178
复习要点	147	9.4 项目的成本管理	178
第8章 软件过程	149	9.4.1 项目成本的概念	178
8.1 软件过程的概念	149	9.4.2 项目成本管理的过程	179
8.2 软件过程的建模	150	9.5 项目的进度管理	180
		9.5.1 项目进度管理的概念	180

9.5.2 项目进度管理的过程	181	标准	220
9.6 项目人员与沟通管理	182	10.4.5 国际标准 ISO 90003	221
9.6.1 项目人员管理的概念	182	10.5 验证与确认	222
9.6.2 项目的组织规划	182	10.5.1 软件验证和确认的概念	222
9.6.3 项目的人员组织	185	10.5.2 生存周期中的验证和确认 工作	222
9.6.4 项目团队的组织与建设	186	10.6 软件评审	224
9.6.5 项目冲突及管理	188	10.6.1 软件评审的概念	224
9.6.6 项目沟通管理	189	10.6.2 软件评审的作用	225
9.7 项目风险管理	190	10.6.3 软件评审的实施	225
9.7.1 风险与风险管理的概念	190	10.6.4 评审的方法和技术	227
9.7.2 项目风险管理的过程	192	10.7 审核	229
9.8 软件配置管理	196	小结	231
9.8.1 软件配置管理的概念	196	复习要点	232
9.8.2 软件配置管理的过程	198	第 11 章 软件工程标准化与软件 文档	233
9.9 需求管理	200	11.1 标准和标准化	233
9.9.1 需求管理的概念	200	11.1.1 标准与标准化的概念	233
9.9.2 需求管理的任务	200	11.1.2 软件工程标准的制定与 实施	234
9.9.3 需求变更请求的管理	203	11.2 软件工程标准的分类和 分级	235
小结	205	11.3 软件文档的作用和分类	238
复习要点	206	11.4 软件工程文档的概要	240
第 10 章 软件质量管理	208	11.5 对文档编制的质量要求	243
10.1 软件质量与质量模型	208	小结	244
10.1.1 软件质量的概念	208	复习要点	245
10.1.2 软件质量特性	209	附录 1 全国计算机等级考试四级 软件工程考试大纲 (2013 年版)	246
10.1.3 软件质量模型	209	附录 2 全国计算机等级考试四级 软件工程样题及参考答案	249
10.2 软件质量度量和度量模型	212	参考文献	252
10.2.1 软件质量的度量	212		
10.2.2 软件质量度量模型	213		
10.2.3 软件质量度量方法	214		
10.2.4 软件质量评价	215		
10.3 软件质量计划	217		
10.3.1 软件质量计划编制的目的	217		
10.3.2 软件质量计划的内容	218		
10.4 软件质量保证	218		
10.4.1 软件质量保证的概念	218		
10.4.2 软件质量保证的过程	219		
10.4.3 软件质量保证的任务	220		
10.4.4 质量保证体系与 ISO 9000			

第1章 软件工程概论

“软件工程”的概念自从1968年在NATO的一次会议上被正式提出以来,软件工程学科得到了迅速的发展,在指导人们科学地开发软件,制作软件产品,集成计算机系统,保证软件产品的质量,按期低成本地完成软件产品的生产等方面起了巨大的作用。

考核目标:

1. 了解软件及软件工程的定义,包括软件的定义、软件分类、软件危机的表现、软件工程的定义、软件工程的框架内容、软件工程的层次。
2. 了解两种软件工程方法的基本概念,包括结构化方法和面向对象方法。
3. 了解软件生存周期过程的概念,包括软件生存周期的定义、阶段划分,各个阶段的主要任务和应交付的文档。
4. 了解软件过程的概念,包括软件过程的定义,软件过程的主要活动。
5. 了解主要软件(生存周期)过程模型的活动、流程和优缺点。
6. 了解主要软件工具的概念。

1.1 软件和软件工程的定义

1.1.1 软件的概念

1. 软件的定义

“软件(Software)”一词是在20世纪60年代出现的。一般认为,软件是计算机系统中与计算机硬件相互依存的另一部分。从系统工程角度来看,它作为系统元素,与计算机硬件、人、数据库等共同构成计算机系统。

GB/T 11457—2006《信息技术软件术语》中给出的定义是,软件是与计算机系统的操作有关的计算机程序、规程及可能的相关文档的完整集合。其中,计算机程序(Program)是计算机指令和数据定义的组合,使得计算机硬件能够执行计算或控制功能;规程(Procedure)是为执行给定的任务而应采取的一系列动作的描述;文档(Document)是与程序开发、维护和使用有关的图文材料。

2. 软件分类

R. S. Pressman根据计算机软件的应用,将计算机软件分为以下7个大类。

(1) 系统软件。系统软件是一套服务于其他程序的程序。一些系统软件(如编译器、编辑器、文件管理实用程序)处理的是复杂但确定的信息结构;另一些系统软件(如操作系统构件、驱动程序、网络软件和远程通信处理器)处理的主要是不确定的数据。

(2) 应用软件。应用软件是一些可以满足特定业务需要的独立应用软件。应用软件用于处

理商务或技术数据,以协助业务操作和管理或技术决策。除了传统的数据处理应用外,应用软件也被用于业务领域的实时控制(如销售点的交易处理、实时制造过程控制等)。

(3) 工程/科学软件。工程和科学软件涵盖了广泛的应用领域。当今科学工程领域的应用软件已经不仅仅局限于传统的数值算法。计算机辅助设计、系统仿真和其他的交互性应用程序已经呈现出实时和系统软件的特性。

(4) 嵌入式软件。嵌入式软件存在于某个产品或者系统中,可实现和控制面向最终使用者和系统本身的特性和功能。嵌入式软件可以执行有限的功能(如微波炉的按键控制)或者提供重要的功能和控制能力(如汽车中的燃油控制、仪表盘显示、刹车系统等)。

(5) 产品线软件。产品的设计方向是为多个不同的用户提供特定功能,关注有限的特定市场(如库存控制产品)或大众消费品市场(如文字处理、电子制表、电脑绘图、多媒体、娱乐、数据库管理、个人及公司财务应用等)。

(6) Web 应用软件。最简单的 Web 应用可以是一组超文本链接文件,仅仅用文本和有限的图形表达信息。但随着电子商务和 B2B 应用的发展,网络应用演变成成为复杂的计算环境与企业数据库和商务应用程序相结合,为最终用户提供了更强大的计算功能。

(7) 人工智能软件。人工智能软件利用非数值算法来解决计算和直接分析无法解决的复杂问题。这个领域的应用程序包括机器人、专家系统、模式识别(图像和语音)、人工神经网络、定理证明和博弈等。

为了经济地、顺利地开发软件产品,从事软件开发的人员还需要了解以下方面。

① 遗留软件:以往的软件开发人员开发并成功使用的软件。为了使它们能够在新的环境下适应新的需要,必须对它们加以改造。这样做,多少可以减轻未来软件开发的负担。

② 网络平台:当前,大多数软件开发都是基于网络的,软件开发人员采用适当的开发模式,建立灵活的系统架构,已成为网络软件开发的关键。

③ 开源软件:开源软件就是网络上可以下载并使用的免费软件,是允许用户自行修改和利用的各种软件程序。利用开源软件,软件开发人员可以少编或不编程序,将从网络上下载的开源软件直接集成到目标系统中,完成软件开发的任务。

1.1.2 软件危机

20 世纪 60 年代中期,大容量、高速度计算机的出现,使计算机的应用范围迅速扩大,软件开发急剧增长,软件系统的规模越来越大,复杂程度越来越高,软件可靠性问题也越来越突出。原来的个人设计、个人使用的方式不再能满足要求,迫切需要改变软件生产方式,提高软件生产率,软件危机开始爆发。

1. 软件危机的主要表现

(1) 软件开发成本和进度严重失控,软件开发的实际成本在计算机系统总成本中所占的比例居高不下,投资一再追加,实际成本远远超出预算成本。

(2) 软件开发生产率提高的速度远远跟不上计算机应用的需要,拖延工期几个月甚至几年的现象也并不罕见,这种现象降低了软件开发组织的信誉。

(3) 软件的可靠性差,尽管耗费了大量的人力和物力,而系统的正确性却越来越难以保证,出错率大大提高,由于软件错误而造成的损失十分惊人。

(4) 生产出来的软件难以维护,很多程序缺乏相应的文档,程序的错误难以定位和改正,有时改正了已有的错误又会引入新的错误。

2. 软件危机的解决途径

解决软件危机的途径在于工程化和标准化,也就是说,使用工程化的原则来指导软件产品的开发和维护,通过标准化的手段来规范软件产品的开发过程和管理过程,这就是软件工程的由来。作为一个工程学科,软件工程从硬件工程和其他人类工程中吸收了许多成功的经验,明确提出了软件生存周期的模型,发展了许多软件开发与维护阶段适用的技术和方法,并应用于软件工程实践,取得了良好的效果。

在软件开发过程中人们可以使用软件工具,以辅助进行软件项目管理与产品的生产,人们还将软件生存周期各阶段使用的软件工具有机地集成为一个整体,形成集成化软件支援环境,以期从管理和技术两方面解决软件危机问题。

1.1.3 软件工程的观念

1. 软件工程的定义

1968年,在NATO(北大西洋公约组织)于德国Garmish举行的学术会议上,德国人Fritz Bauer第一次为软件工程下了定义:“软件工程是为了经济地获得能够在实际机器上有效运行的可靠软件而建立和使用的一系列完善的工程化原则。”

此后,IEEE多次给软件工程下定义。我国2006年的国家标准GB/T 11457—2006《信息技术软件工程术语》中给出的软件工程定义为“应用计算机科学理论和技术以及工程管理原则和方法,按预算和进度,实现满足用户要求的软件产品的定义、开发、发布和维护的工程或进行研究的学科”。

2. 软件工程框架

北京大学杨芙清院士基于软件工程的不同侧面,提出了一个软件工程框架模型,如图1.1所示,见于《计算机科学与技术大百科全书》。

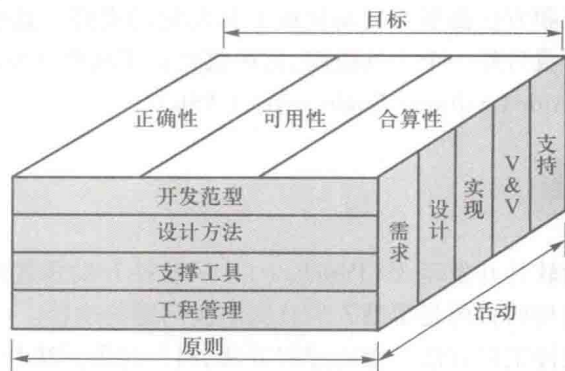


图 1.1 软件工程框架

软件工程的目的是“生产具有正确性、可用性及开销适宜的产品”。其中,正确性表明软件产品达到预期功能的程度;可用性表明软件基本结构、实现和文档为用户所接受的程度;合算性表明软件开发、运行的整个开销满足用户要求的程度。这些目标形成了对过程、过程模型及工程方法选取的约束。

软件工程活动是“生产一个最终满足需求并达到工程目标要求的软件产品所需要的一系列步骤”,主要包括需求分析、设计、实现、V&V(验证与确认)以及支持等活动。支持活动又包括修改和完善。伴随以上活动的还有管理过程、支持过程和培训过程等。

围绕工程设计、工程支持和工程管理,应注意遵守以下4条基本原则。

(1) 选取适宜的开发范型(也称为开发方法学)。软件需求、硬件需求及其他因素间是相互制约、相互影响的,必须认清需求定义的易变性,采用适当的开发范型予以控制。

(2) 采用合适的设计方法。在软件设计中,通常需要考虑软件的模块化、抽象和信息隐蔽、局部化、一致性、适应性等特征。选取合适的设计方法有助于这些特征的实现。

(3) 提供高质量的工程支持。在软件工程中,软件工具与环境对软件过程的支持非常重要。软件工程项目的质量与开销直接取决于支撑软件工程的工具的质量与效用。

(4) 重视开发过程的管理。软件工程的管理工作直接影响到能否有效利用可用资源,生产满足要求的软件产品以及提高软件组织的生产能力等问题。因此,只有当软件过程得到有效管理时,才能实现软件工程的有效运用。

从技术角度来看,R. S. Pressman认为软件工程是一种层次化的技术,如图1.2所示。

(1) 任何工程(包括软件工程)最基本的就是质量。任何产品(包括软件产品)的质量关注点就是要让客户满意,这意味着合格的产品要在规定的时间和成本内交付。

(2) 软件产品的质量需要在软件生产过程中层层把关。因此,软件过程的作用就是把各个技术层次结合在一起,合理地组织所有相关活动、人员、规程、方法,高效地开发计算机软件。

(3) 软件工程方法为构建软件产品提供了技术上的解决方法。方法覆盖面很广,包括沟通、需求分析、设计建模、编程、测试和技术支持。

(4) 软件工具为过程和方法提供了自动化或半自动化的支持。这些工具可以集成起来,使得一个工具产生的信息可被另外一个工具使用,这样就建立了软件开发的支撑系统,称为计算机辅助软件工程(Computer-Aided Software Engineering, CASE)。



图 1.2 软件工程层次图

1.2 软件工程方法

软件工程方法又称为软件开发范型(Paradigm)。从软件开发角度来看,范型与问题解决技术有关。研究软件开发范型的目的是帮助人们找到解决问题的途径。

目前使用最广泛的软件工程方法是面向过程方法、面向对象方法和形式化方法。

1.2.1 面向过程方法

面向过程方法是使用最广泛、历史最长的软件开发范型。该方法从功能角度出发,把软件视为处理流,并定义成由一系列步骤构成的程序过程。每一个步骤都是带有预定输入和特定输出的一个过程,把这些步骤串联在一起可产生合理的、稳定的、贯通于整个程序的控制流,最终产生一个简单的具有静态结构的体系结构,如图1.3所示。



图 1.3 面向过程方法所构造系统的基本架构

面向过程方法侧重于建立解决问题的处理流,数据结构是根据程序算法步骤的要求开发的,它贯穿于整个过程中,提供过程所要求操作的信息。系统在运行中处于什么状态,可用一组全局变量来描述,并把系统当前状态用一组值保存在这组全局变量中,可从一个过程传送到另一个过程。

用面向过程方法建立起来的软件结构是由一系列程序模块组成的,目前已经有许多支持开发的工具,例如,用 C 语言或 Pascal 语言编写的程序都属于面向过程的方法,解决问题的每一步都必须用一条条程序语句写在程序里。

1.2.2 面向对象方法

面向对象方法是问题分解方法的演化结果。这种方法在考虑问题解决方法时优先考虑的不是按照功能建立系统,而是问题域中的实体。该方法把标识和模型化问题领域中的主要实体作为系统开发的起点,主要考虑实体对象的行为而不是必须执行的一系列动作。

面向对象软件中的基本单位是对象,它是数据抽象与过程抽象的综合体。数据抽象就是对象中的属性,过程抽象就是对象中操作属性值的一系列方法。软件系统在运行中的状态用各个对象的属性值来表述。完成系统功能所涉及的控制流是由包含在各个对象中的方法,以及各个方法之间传送的消息组成的,可以在将来修改软件系统时使得整个系统的架构保持稳定。

图 1.4 给出一个 draw 程序的面向对象体系结构。图中的箭头描述了从一个对象传送到另一个对象的消息。为了简化这个图,没有把所有的消息都画出来。这个体系结构中的对象有外部的物理实体,如鼠标、键盘和显示器等;还有数据实体,如事件、图形、队列等;以及控制实体,如事件处理器等。在系统中还存在继承关系,描述了一般和特殊的关系,如图形可具体化为矩形、椭圆形和三角形等。

总之,面向对象方法的出发点和基本原则是尽量模拟人类习惯的思维方式,使开发软件的方法和过程尽可能接近人类认识问题和解决问题的方法与过程,从而使描述的问题空间与其解决空间在结构上尽可能一致。

用面向对象方法开发软件的过程是多次反复迭代的演化过程。面向对象方法在概念和表示方法上的一致性,保证了各项开发活动之间的平滑过渡。对于大型、复杂及交互性比较强的系统,使用面向对象方法更有优势。

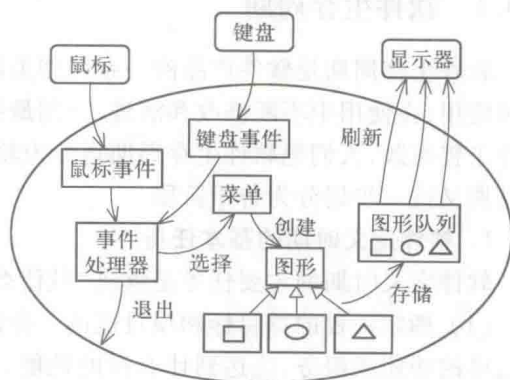


图 1.4 一个 draw 程序的面向对象体系结构

1.2.3 形式化方法

形式化方法是一种基于形式化数学变换的软件开发方法,它可将系统的规格说明转换为可执行的程序。过程的描述如图 1.5 所示。为了简化模型,过程的迭代在图中没有画出。

形式化方法的主要特点如下。

- (1) 软件需求规格说明被细化为用数学记号表达的详细的形式化规格说明。
- (2) 设计、实现和单元测试等开发过程由一个变换开发过程代替。通过一系列变换将形式的规格说明细化成为程序。

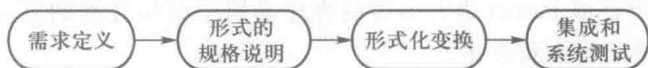


图 1.5 形式化方法

1.3 软件过程与软件生存周期

软件系统的开发与其他工业产品的开发一样,也需要有设计、制作、检验等环节。作为一种产品,它的开发也应划分阶段。基于这种认识,1976 年之后,人们提出了“软件生存周期”这一概念,开创了软件生产工程化与规范化的先河。

1.3.1 软件生存周期

软件生存周期是软件产品的一系列相关活动的整个生命期,即从形成概念开始,经过开发、交付使用,在使用中不断修改和演进,直到最终被废弃,被新的软件产品代替的整个时期。根据软件工程实践,人们把软件生存周期划分为软件定义、软件开发和交付后运行维护 3 个时期,每个时期又进一步划分为若干阶段。

1. 软件定义时期的基本任务

软件定义时期的主要任务是解决“做什么”的问题。

(1) 确定工程的总目标和项目范围。弄清到底要软件干什么,初步确立软件应提供给客户什么样的功能或服务,应达到什么样的性能,有哪些质量要求(目标),为完成这样的目标,需要进行哪些活动(范围)。

(2) 开展可行性研究。进行成本-效益分析,探讨在成本和时间的限制条件下项目是否值得做,进行技术解决方案分析和开发组织(或团队)技术实力评估,探讨能否解决客户要求解决的问题,还要进行法律、用户操作方面的可行性研究,最后撰写可行性研究报告,提交给用户和上级管理部门供其决策用。

(3) 需求获取。继续开展调查研究,明确待开发软件的用户需求、功能需求和非功能需求,并通过需求描述或快速原型与客户沟通。

(4) 制订项目计划。对项目活动和产品需求做进一步分解,细化工作分解结构(WBS),同时,对可利用的计算机硬件、软件、人力、成本、开发的进度做出估算,初步制订出完成开发任务的

实施计划。

2. 软件开发时期的基本任务

软件开发时期的主要任务是解决“如何做”的问题。给出待开发软件的设计方案并构造出待开发软件。这一时期又可分为概念设计、概要设计、详细设计、编码和测试等阶段。

(1) 概念设计。这个阶段包括两大任务,一是建立分析模型,从功能、数据、行为等方面细致地描述系统的静态特性和动态特性,引入软件的各种需求细节。二是基于分析结果,编写软件需求规格说明,确认测试计划和初步的用户手册。

(2) 概要设计。从系统的输入/输出入手,搭建系统的体系结构和全局数据结构,同时对软件的人机交互界面和运行控制机制建模。

(3) 详细设计。对软件体系结构中的每一成分(构件)进行算法和局部数据结构设计,同时设计人机交互界面的细节、构件内部的处理细节和构件在实际运行环境下的部署。

(4) 编码。使用某种编程语言把设计结果转换为程序代码,同时进行单元测试,发现差错需进行调试(排错),还可能要执行回归测试。

(5) 测试。通过集成测试组装系统;通过系统测试检查系统的实现是否与软件需求规格说明保持一致,通过接受测试检验软件产品的实用性。

3. 交付后维护时期的基本任务

交付后维护时期的主要任务是使软件持久地满足用户的需要。通常有4类维护活动。

(1) 改正性维护。诊断和改正在使用过程中发现的软件错误所做的维护。

(2) 适应性维护。修改软件以适应环境的变化所做的维护。

(3) 完善性维护。根据用户的要求改进或扩充软件使其更完善所做的维护。

(4) 预防性维护。修改软件为将来的维护活动预先做准备。

1.3.2 软件过程

在从事软件开发工作时,由于项目的类型、规模不同,因而使用的开发方法不同,开发时需执行的活动也会有差异。但不论是何种情况,最基本的活动都可归为以下4种。

(1) 软件规格说明。定义软件产品的功能和操作约束。

(2) 软件设计与实现。生产满足规格说明的软件产品。

(3) 软件确认。确认软件产品的有效性,确保该软件产品所做的是用户所需要的。

(4) 软件演进。改进软件产品,满足用户新的需要。

这些活动的集合就构成软件过程。通常定义软件过程为软件生存周期中一系列相关软件工程活动的集合,它是由一组工作任务、软件阶段里程碑、工作制品和交付物以及质量保证检查点组成的。一个软件开发组织应有一个过程框架,如图1.6所示。

这个过程框架定义了少量可用于所有软件项目的活动,针对各个项目的具体特点和项目组的需求,可以选取和构建适合于该项目的过程活动。

保护伞活动,如软件质量保证、软件配置管理等,它

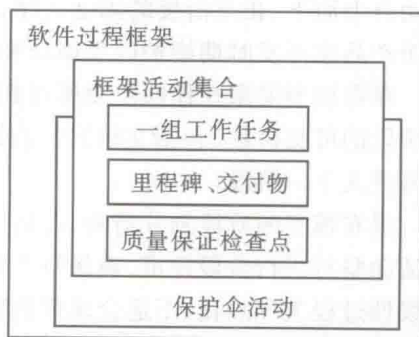


图 1.6 软件过程的框架