

可下载教学资料

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



高等学校教材
软件工程

软件体系结构

(第2版)

张友生 等 编著

清华大学出版社



高等学校教材
软件工程

软件体系结构

(第2版)

张友生 等 编著

清华大学出版社
北京

内 容 简 介

本书系统地介绍软件体系结构的基本原理、方法和实践,全面反映软件体系结构研究和应用的最新进展。既讨论软件体系结构的基本理论知识,又介绍软件体系结构的设计和工业界应用实例,强调理论与实践相结合。

全书共10章,第1章简单介绍软件体系结构的概念、发展和应用现状;第2章讨论软件体系结构建模,包括视图模型、核心模型、生命周期模型和抽象模型;第3章介绍软件体系结构的风格和特定领域软件体系结构;第4章讨论软件体系结构的描述方法,重点介绍软件体系结构描述语言;第5章介绍动态软件体系结构及其描述方法;第6章介绍WEB服务体系结构相关知识,以及面向服务的体系结构的基本概念和设计原则;第7章讨论基于体系结构的软件开发方法,介绍基于体系结构的软件过程;第8章讨论软件体系结构的分析与测试问题,重点介绍软件体系结构的可靠性风险分析;第9章讨论软件体系评估方法,重点介绍ATAM和SAAM方法;第10章介绍软件产品线的原理和方法、框架技术,重点讨论产品线体系结构的设计和演化。

本书可作为计算机软件专业高年级本科生、研究生和软件工程硕士的软件体系结构教材,作为软件工程高级培训、系统分析师和系统架构设计师培训教材,也可作为软件开发人员的参考书。

版权所有,翻印必究。举报电话:010-62782989 13501256678 13801310933

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

本书防伪标签采用特殊防伪技术,用户可通过在图案表面涂抹清水,图案消失,水干后图案复现;或将表面膜揭下,放在白纸上用彩笔涂抹,图案在白纸上再现的方法识别真伪。

图书在版编目(CIP)数据

软件体系结构/张友生等编著. —2版. —北京:清华大学出版社,2006.11

(高等学校教材·软件工程)

ISBN 7-302-13316-6

I. 软… II. 张… III. 软件—系统结构 IV. TP311.5

中国版本图书馆CIP数据核字(2006)第073245号

出 版 者:清华大学出版社 地 址:北京清华大学学研大厦

<http://www.tup.com.cn> 邮 编:100084

社 总 机:010-62770175 客户服务:010-62776969

责任编辑:丁 岭

印 刷 者:北京市昌平环球印刷厂

装 订 者:三河市金元印装有限公司

发 行 者:新华书店总店北京发行所

开 本:185×260 印张:20.25 字数:501千字

版 次:2006年11月第2版 2006年11月第1次印刷

书 号:ISBN 7-302-13316-6/TP·8295

印 数:1~4000

定 价:30.00元

改革开放以来,特别是党的十五大以来,我国教育事业取得了举世瞩目的辉煌成就,高等教育实现了历史性的跨越,已由精英教育阶段进入国际公认的大众化教育阶段。在质量不断提高的基础上,高等教育规模取得如此快速的发展,创造了世界教育发展史上的奇迹。当前,教育工作既面临着千载难逢的良好机遇,同时也面临着前所未有的严峻挑战。社会不断增长的高等教育需求同教育供给特别是优质教育供给不足的矛盾,是现阶段教育发展面临的基本矛盾。

教育部一直十分重视高等教育质量工作。2001年8月,教育部下发了《关于加强高等学校本科教学工作,提高教学质量的若干意见》,提出了十二条加强本科教学工作提高教学质量的措施和意见。2003年6月和2004年2月,教育部分别下发了《关于启动高等学校教学质量与教学改革工程精品课程建设工作的通知》和《教育部实施精品课程建设提高高校教学质量和人才培养质量》文件,指出“高等学校教学质量和教学改革工程”是教育部正在制定的《2003—2007年教育振兴行动计划》的重要组成部分,精品课程建设是“质量工程”的重要内容之一。教育部计划用五年时间(2003—2007年)建设1500门国家级精品课程,利用现代化的教育信息技术手段将精品课程的相关内容上网并免费开放,以实现优质教学资源共享,提高高等学校教学质量和人才培养质量。

为了深入贯彻落实教育部《关于加强高等学校本科教学工作,提高教学质量的若干意见》精神,紧密配合教育部已经启动的“高等学校教学质量与教学改革工程精品课程建设工作”,在有关专家、教授的倡议和有关部门的大力支持下,我们组织并成立了“清华大学出版社教材编审委员会”(以下简称“编委会”),旨在配合教育部制定精品课程教材的出版规划,讨论并实施精品课程教材的编写与出版工作。“编委会”成员皆来自全国各类高等学校教学与科研第一线的骨干教师,其中许多教师为各校相关院、系主管教学的院长或系主任。

按照教育部的要求,“编委会”一致认为,精品课程的建设工作从开始就要坚持高标准、严要求,处于一个比较高的起点上;精品课程教材应该能够反映各高校教学改革与课程建设的需要,要有特色风格、有创新性(新体系、新内容、新手段、新思路,教材的内容体系有较高的科学创新、技术创新和理念创新的含量)、先进性(对原有的学科体系有实质性的改革和发展、顺应并符合新世纪教学发展的规律、代表并引领课程发展的趋势和方向)、示范性(教材所体现的课程体系具有较广泛的辐射性和示范性)和一定的前瞻

性。教材由个人申报或各校推荐(通过所在高校的“编委会”成员推荐),经“编委会”认真评审,最后由清华大学出版社审定出版。

目前,针对计算机类和电子信息类相关专业成立了两个“编委会”,即“清华大学出版社计算机教材编审委员会”和“清华大学出版社电子信息教材编审委员会”。首批推出的特色精品教材包括:

(1) 高等学校教材·计算机应用——高等学校各类专业,特别是非计算机专业的计算机应用类教材。

(2) 高等学校教材·计算机科学与技术——高等学校计算机相关专业的教材。

(3) 高等学校教材·电子信息——高等学校电子信息相关专业的教材。

(4) 高等学校教材·软件工程——高等学校软件工程相关专业的教材。

(5) 高等学校教材·信息管理与信息系统。

(6) 高等学校教材·财经管理与计算机应用。

清华大学出版社经过 20 多年的努力,在教材尤其是计算机和电子信息类专业教材出版方面树立了权威品牌,为我国的高等教育事业做出了重要贡献。清华版教材形成了技术准确、内容严谨的独特风格,这种风格将延续并反映在特色精品教材的建设中。

清华大学出版社教材编审委员会

E-mail: dingl@tup.tsinghua.edu.cn

体系结构一词在英文里就是“建筑”的意思。把软件系统比做一座楼房,从整体上讲,是因为它有基础、主体和装饰,即操作系统之上的基础设施软件,实现计算逻辑的主体应用程序,方便使用的用户界面程序。从细节上看,每一个程序也是有结构的。早期的结构化程序就是以语句组成模块,模块的聚集和嵌套形成层层调用的程序结构,也就是体系结构。结构化程序的程序(表达)结构和(计算的)逻辑结构的一致性及自顶向下的开发方法自然而然地形成了体系结构。由于结构化程序设计时代程序规模不大,通过强调结构化程序设计方法学,自顶向下、逐步求精,并注意模块的耦合性就可以得到相对良好的结构,所以,并未特别深入研究软件体系结构。

随着软件系统规模越来越大、越来越复杂,整个系统的结构和规格说明就显得越来越重要。对于大规模的复杂软件系统来说,总体的系统结构设计和规格说明比起对计算的算法和数据结构的选择变得明显重要。在此种背景下,人们认识到了软件体系结构的重要性,并认为对软件体系结构的系统进行深入的研究将会成为提高软件生产率和解决软件维护问题的新的最有希望的途径。

对于软件项目的开发来说,具有清晰的软件体系结构是首要的。传统的软件开发过程可以划分为从概念直到具体实现的若干个阶段,包括问题定义、需求分析、软件设计、软件实现及软件测试等。软件体系结构的建立应在需求分析之后,软件设计之前。但在传统的软件工程方法中,需求和设计之间存在一条很难逾越的鸿沟,从而难以有效地将需求转换为相应的设计。软件体系结构就是试图在软件需求与软件设计之间架起一座桥梁,着重解决软件系统的结构和需求向实现平坦地过渡的问题。

体系结构在软件开发中为不同的人员提供了共同交流的语言,体现并尝试了系统早期的设计决策,并作为系统设计的抽象,为实现框架和构件的共享和重用、基于体系结构的软件开发提供了有力的支持。鉴于体系结构的重要性,Perry 将软件体系结构视为软件开发中第一类重要的设计对象,Barry Boehm 也明确指出:“在没有设计出体系结构及其规则时,整个项目不能继续下去,而且体系结构应该看作软件开发中可交付的中间产品”。

软件体系结构是根植于软件工程发展起来的一门新兴学科,目前已经成为软件工程研究和实践的主要领域。专门和广泛的研究软件体系结构是从 20 世纪 90 年代才开始的,1993 年到 1995 年,卡耐基梅隆大学的 Mary Shaw 与 David Garlan,贝尔实验室

的 Perry,南加州大学的 Barry Boehm,斯坦福大学的 David Luckham 等人开始将注意力投向软件体系结构的研究和学科建设。

目前,软件体系结构的领域研究非常活跃,如南加州大学专门成立了软件体系结构研究组,曼彻斯特大学专门成立了软件体系结构研究所。同时,业界许多著名企业的研究中心也将软件体系结构作为重要的研究内容。如由 IBM、Nokia 和 ABB 等企业联合一些大学研究嵌入式体系的体系结构项目。国内也有不少机构在从事软件体系结构方面的研究,如北京大学软件工程研究所一直从事基于体系结构软件组装的工业化生产方法与平台的研究,北京邮电大学则研究了电信软件的体系结构,国防科学技术大学推出的 CORBA 规范实现平台为体系结构研究提供了基础设施所需的中间件技术。许多大学为计算机软件专业硕士和软件工程硕士都开设了软件体系结构课程。

本书共分 10 章,第 1 章简单介绍软件体系结构的概念、发展和应用现状;第 2 章讨论软件体系结构建模,包括视图模型、核心模型、生命周期模型和抽象模型;第 3 章介绍软件体系结构的风格和特定领域软件体系结构;第 4 章讨论软件体系结构的描述方法,重点介绍软件体系结构描述语言;第 5 章介绍动态软件体系结构及其描述方法;第 6 章介绍 Web 服务体系结构相关知识,以及面向服务的体系结构的基本概念和设计原则;第 7 章讨论基于体系结构的软件开发方法,介绍基于体系结构的软件过程;第 8 章讨论软件体系结构的分析与测试问题,重点介绍软件体系结构的可靠性风险分析;第 9 章讨论软件体系评估方法,重点介绍 ATAM 和 SAAM 方法;第 10 章介绍软件产品线的原理和方法、框架技术,重点讨论产品线体系结构的设计和演化。第 4.6、4.7、5、6、7.7 节和第 8 章由李雄编写,其他章节由张友生编写。

在本书出版之际,我们要特别感谢国内外软件工程和软件体系结构专著、教材和许多高水平论文、报告的作者们(恕不一一列举,名单详见各章中的主要参考文献),他们的作品为本书提供了丰富的营养,使我们受益匪浅。在本书中引用了他们的部分材料,使本书能够尽量反映软件体系结构研究和实践领域的最新进展。

感谢阅读本书第一版的读者,特别要感谢使用本书第一版作为教材的老师,他们为本书的修订和第 2 版的出版提出了宝贵的意见。

感谢希赛网(<http://www.csai.cn>)为本书的意见反馈提供了空间和程序,感谢清华大学出版社的帮助。

由于作者水平有限,时间紧迫,加上软件体系结构是一门新兴的学科,本身发展很快,对有些新领域作者尚不熟悉。因此,书中难免有不妥和错误之处,我们诚恳地期望各位专家和读者不吝指教和帮助。对此,我们将深为感激。



2006年9月

第 1 章 软件体系结构概论	1
1.1 从软件危机谈起	1
1.1.1 软件危机的表现	1
1.1.2 软件危机的成因	2
1.1.3 如何克服软件危机	3
1.2 构件与软件重用	4
1.2.1 构件模型及实现	4
1.2.2 构件获取	5
1.2.3 构件管理	6
1.2.4 构件重用	10
1.2.5 软件重用实例	15
1.3 软件体系结构的兴起和发展	18
1.3.1 软件体系结构的定义	19
1.3.2 软件体系结构的意义	20
1.3.3 软件体系结构的发展史	23
1.4 软件体系结构的应用现状	23
主要参考文献	29
第 2 章 软件体系结构建模	30
2.1 软件体系结构建模概述	30
2.2 “4+1”视图模型	30
2.2.1 逻辑视图	31
2.2.2 开发视图	32
2.2.3 进程视图	33
2.2.4 物理视图	35

2.2.5	场景	36
2.3	软件体系结构的核心模型	37
2.4	软件体系结构的生命周期模型	38
2.5	软件体系结构抽象模型	41
2.5.1	构件	41
2.5.2	连接件	43
2.5.3	软件体系结构	44
2.5.4	软件体系结构关系	44
2.5.5	软件体系结构范式	46
	主要参考文献	48
第3章	软件体系结构风格	49
3.1	软件体系结构风格概述	49
3.2	经典软件体系结构风格	50
3.2.1	管道和过滤器	50
3.2.2	数据抽象和面向对象组织	51
3.2.3	基于事件的隐式调用	52
3.2.4	分层系统	53
3.2.5	仓库系统及知识库	53
3.2.6	C2 风格	54
3.3	客户/服务器风格	55
3.4	三层 C/S 结构风格	57
3.4.1	三层 C/S 结构的概念	57
3.4.2	三层 C/S 结构应用实例	60
3.4.3	三层 C/S 结构的优点	64
3.5	浏览器/服务器风格	65
3.6	公共对象请求代理体系结构	66
3.7	正交软件体系结构	69
3.7.1	正交软件体系结构的概念	69
3.7.2	正交软件体系结构的实例	70
3.7.3	正交软件体系结构的优点	73
3.8	基于层次消息总线的体系结构风格	73
3.8.1	构件模型	75
3.8.2	构件接口	75
3.8.3	消息总线	76

3.8.4	构件静态结构	77
3.8.5	构件动态行为	78
3.8.6	运行时刻的系统演化	78
3.9	异构结构风格	79
3.9.1	为什么要使用异构结构	79
3.9.2	异构结构的实例	80
3.9.3	异构组合匹配问题	83
3.10	互联系统构成的系统及其体系结构	84
3.10.1	互联系统构成的系统	84
3.10.2	基于 SASIS 的软件过程	85
3.10.3	应用范围	87
3.11	特定领域软件体系结构	89
3.11.1	DSSA 的定义	89
3.11.2	DSSA 的基本活动	90
3.11.3	参与 DSSA 的人员	91
3.11.4	DSSA 的建立过程	92
3.11.5	DSSA 实例	94
3.11.6	DSSA 与体系结构风格的比较	97
	主要参考文献	98
第 4 章	软件体系结构描述	99
4.1	软件体系结构描述方法	99
4.2	软件体系结构描述框架标准	101
4.3	体系结构描述语言	102
4.3.1	ADL 与其他语言的比较	102
4.3.2	ADL 的构成要素	104
4.4	典型的软件体系结构描述语言	106
4.4.1	UniCon	106
4.4.2	Wright	108
4.4.3	C2	109
4.4.4	Rapide	113
4.4.5	SADL	114
4.4.6	Aesop	115
4.4.7	ACME	116
4.5	软件体系结构与 UML	123

4.5.1	UML 简介	123
4.5.2	UML 的主要内容	125
4.5.3	直接使用 UML 建模	130
4.5.4	使用 UML 扩展机制	134
4.6	可扩展标记语言	138
4.6.1	XML 语言简介	138
4.6.2	XML 相关技术简介	140
4.7	基于 XML 的软件体系结构描述语言	142
4.7.1	XADL 2.0	142
4.7.2	XBA	147
	主要参考文献	150
第 5 章	动态软件体系结构	151
5.1	动态软件体系结构概述	151
5.2	软件体系结构动态模型	153
5.2.1	基于构件的动态系统结构模型	153
5.2.2	π ADL 动态体系结构	157
5.3	动态体系结构的描述	162
5.3.1	动态体系结构描述语言	162
5.3.2	动态软件体系结构的形式化描述	163
5.4	动态体系结构特征	165
5.5	化学抽象机	166
	主要参考文献	169
第 6 章	Web 服务体系结构	170
6.1	Web 服务概述	170
6.1.1	什么是 Web 服务	170
6.1.2	Web 服务的不同描述	171
6.1.3	Web 服务的特点	172
6.2	Web 服务体系结构模型	173
6.3	Web 服务的核心技术	176
6.3.1	作为 Web 服务基础的 XML	176
6.3.2	简单对象访问协议	177
6.3.3	Web 服务描述语言	179
6.3.4	统一描述、发现和集成协议	180

6.4	面向服务的软件体系结构	182
6.4.1	面向服务体系结构概念	182
6.4.2	面向服务体系结构的设计原则	184
6.5	Web 服务的应用实例	186
	主要参考文献	189
第 7 章	基于体系结构的软件开发	191
7.1	设计模式	191
7.1.1	设计模式概述	191
7.1.2	设计模式的组成	194
7.1.3	模式和软件体系结构	196
7.1.4	设计模式方法分类	197
7.2	基于体系结构的设计方法	201
7.2.1	有关术语	201
7.2.2	ABSD 方法与生命周期	203
7.2.3	ABSD 方法的步骤	205
7.3	体系结构的设计与演化	211
7.3.1	设计和演化过程	211
7.3.2	实验原型阶段	212
7.3.3	演化开发阶段	214
7.4	基于体系结构的软件开发模型	215
7.4.1	体系结构需求	215
7.4.2	体系结构设计	216
7.4.3	体系结构文档化	217
7.4.4	体系结构复审	218
7.4.5	体系结构实现	218
7.4.6	体系结构演化	219
7.5	应用开发实例	220
7.5.1	系统简介	220
7.5.2	系统设计与实现	224
7.5.3	系统演化	225
7.6	基于体系结构的软件过程	226
7.6.1	有关概念	226
7.6.2	软件过程网	228
7.6.3	基本结构的表示	230

7.6.4 基于体系结构的软件过程 Petri 网	231
7.7 软件体系结构演化模型	235
7.7.1 SA 静态演化模型	235
7.7.2 SA 的动态演化模型	238
主要参考文献	240
第 8 章 软件体系结构的分析与测试	242
8.1 体系结构的可靠性建模	242
8.2 软件体系结构的可靠性风险分析	246
8.2.1 软件体系结构风险分析背景	246
8.2.2 软件体系结构风险分析方法	247
8.3 基于体系结构描述的软件测试	252
8.3.1 测试方法	252
8.3.2 实例与实现	254
主要参考文献	255
第 9 章 软件体系结构评估	256
9.1 体系结构评估概述	256
9.2 软件体系结构评估的主要方式	260
9.3 ATAM 评估方法	261
9.3.1 ATAM 评估的步骤	262
9.3.2 ATAM 评估的阶段	268
9.4 SAAM 评估方法	271
9.4.1 SAAM 评估的步骤	271
9.4.2 SAAM 评估实例	275
主要参考文献	279
第 10 章 软件产品线体系结构	280
10.1 软件产品线的出现和发展	280
10.1.1 软件体系结构的发展	281
10.1.2 软件重用的发展	282
10.2 软件产品线概述	282
10.2.1 软件产品线的基本概念	282
10.2.2 软件产品线的过程模型	283
10.2.3 软件产品线的组织结构	285

10.2.4	软件产品线的建立方式	287
10.2.5	软件产品线的演化	288
10.3	框架和应用框架技术	289
10.4	软件产品线基本活动	291
10.5	软件产品线体系结构的设计	294
10.5.1	产品线体系结构简介	294
10.5.2	产品线体系结构的标准化和定制	296
10.6	软件产品线体系结构的演化	297
10.6.1	背景介绍	298
10.6.2	两代产品的各种发行版本	300
10.6.3	需求和演化的分类	303
	主要参考文献	306

软件体系结构概论

1.1 从软件危机谈起

软件危机 (software crisis) 是指在计算机软件的开发 (development) 和维护 (maintenance) 过程中所遇到的一系列严重问题。20 世纪 60 年代末至 70 年代初,“软件危机”一词在计算机界广为流传。事实上,几乎从计算机诞生的那一天起,就出现了软件危机,只不过到了 1968 年在原西德加施召开的国际软件工程会议上才被人们普遍认识到。

1.1.1 软件危机的表现

1. 软件成本日益增长

在计算机发展的早期,大型计算机系统主要是设计 (design) 应用于非常狭窄的军事领域。在这个时期,研制计算机的费用主要由国家财政提供,研制者很少考虑到研制代价问题。随着计算机市场化和民用化的发展,代价和成本就成为投资者考虑的最重要的问题之一。20 世纪 50 年代,软件成本 (cost) 在整个计算机系统成本中所占的比例为 10%~20%。但随着软件产业的发展,软件成本日益增长。相反,计算机硬件随着技术的进步、生产规模的扩大,价格却在不断下降。这样一来,软件成本在计算机系统中所占的比例越来越大。到 20 世纪 60 年代中期,软件成本在计算机系统中所占的比例已经增长到 50% 左右。而且,该数字还在不断地递增,下面是一组来自美国空军计算机系统的数字: 1955 年,软件费用约占总费用的 18%,1970 年达到 60%,1975 年达到 72%,1980 年达到 80%,1985 年达到 85% 左右。

2. 开发进度难以控制

由于软件是逻辑、智力产品,软件的开发需建立庞大的逻辑体系,这与其他产品的生产不一样。例如,工厂要生产某种机器,在时间紧的情况下可以要工人加班或者实行“三班倒”,而这些方法都不能用在软件开发上。

在软件开发过程中,用户需求 (requirement) 变化等各种意想不到的情况层出不穷,常常令软件开发过程很难保证按预定的计划实现,给项目计划和论证工作带来很大的困难。

BROOK 曾经提出：“在已拖延的软件项目上，增加人力只会使其更难按期完成”。事实上，软件系统的结构很复杂，各部分附加联系极大，盲目增加软件开发人员并不能成比例地提高软件开发能力。相反，随着人员数量的增加，人员的组织、协调、通信、培训和管理等方面的问题将更为严重。

许多重要的大型软件开发项目，如 IBM OS/360 和世界范围的军事命令控制系统（WWMCCS），在耗费了大量的人力和财力之后，由于离预定目标相差甚远而不得不宣布失败。

3. 软件质量差

软件项目即使能按预定日期完成，结果却不尽如人意。1965 年至 1970 年间，美国范登堡基地发射火箭多次失败，绝大部分故障是由应用程序错误造成的。程序的一些微小错误可以造成灾难性的后果，例如，有一次在美国肯尼迪发射一枚阿脱拉斯火箭，火箭飞离地面几十英里高空开始翻转，地面控制中心被迫下令炸毁。后经检查发现是飞行计划程序里漏掉了一个连字符，就是这样一个小小的疏漏造成了这枚价值 1850 万美元的火箭试验失败。

在“软件作坊”里，由于缺乏工程化思想的指导，程序员几乎总是习惯性地以自己的想法去代替用户对软件的需求，软件设计带有随意性，很多功能只是程序员的“一厢情愿”而已，这是造成软件不能令人满意的重要因素。

4. 软件维护困难

正式投入使用的软件，总是存在一些错误，在不同的运行条件下，软件就会出现故障，因此需要维护。但是，由于在软件设计和开发过程中，没有严格遵循软件开发标准，随意性很大，没有完整的真实反映系统状况的记录文档，给软件维护造成了巨大的困难。特别是在软件使用过程中，开发人员可能因各种原因已经离开了原来的开发组织，使得软件几乎不可维护。

另外，软件修改是一项很“危险”的工作，对一个复杂的逻辑过程，哪怕作一项微小的改动，都可能引入潜在的错误，常常会发生“纠正一个错误带来更多新错误”的问题，从而产生副作用。

有资料表明，工业界为维护软件支付的费用占全部硬件和软件费用的 40%~75%。

1.1.2 软件危机的成因

从软件危机的种种表现和软件作为逻辑产品的特殊性可以发现造成软件危机的成因。

1. 用户需求不明确

在软件开发过程中，用户需求不明确。问题主要体现在四个方面：

- (1) 在软件开发出来之前，用户自己也不清楚软件的具体需求。
- (2) 用户对软件需求的描述不精确，可能有遗漏、有二义性甚至有错误。
- (3) 在软件开发过程中，用户还提出修改软件功能(function)、界面(interface)、支撑环境(environment)等方面的要求。

(4) 软件开发人员对用户需求的理解与用户本来愿望有差异。

2. 缺乏正确的理论指导

缺乏有力的方法学和工具方面的支持。由于软件不同于大多数其他工业产品,其开发过程是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。由于过分依靠程序设计人员在软件开发过程中的技巧和创造性,加剧了软件产品的个性化,也是导致软件危机的一个重要原因。

3. 软件规模越来越大

随着软件应用范围的增广,软件规模愈来愈大。大型软件项目需要组织一定的人力共同完成,而多数管理人员缺乏开发大型软件系统的经验,且多数软件开发人员又缺乏管理方面的经验。各类人员的信息交流不及时、不准确,有时还会产生误解。软件项目开发人员不能有效地、独立自主地处理大型软件的全部关系和各个分支,因此容易产生疏漏和错误。

4. 软件复杂度越来越高

软件不仅仅是在规模上快速地发展扩大,而且其复杂性(complexity)也急剧增加。软件产品的特殊性和人类智力的局限性,导致人们无力处理“复杂问题”。所谓“复杂问题”的概念是相对的,一旦人们采用先进的组织形式、开发方法和工具提高了软件开发效率和能力,新的、更大的、更复杂的问题又摆在人们的面前。

1.1.3 如何克服软件危机

人们在认真地研究和分析了软件危机背后的真正原因之后,得出了“人们面临的不光是技术问题,更重要的是管理问题。管理不善必然导致失败”的结论,便开始探索用工程的方法进行软件生产的可能性,即应用现代工程的概念、原理、技术和方法进行计算机软件的开发、管理和维护。于是,计算机科学技术的一个新领域——软件工程(software engineering)诞生了。

软件工程是用工程、科学和数学的原则与方法研制、维护计算机软件的有关技术及管理方法。软件工程包括三个要素:方法、工具和过程。

软件工程方法为软件开发提供了“如何做”的技术,是完成软件工程项目的手段;

软件工具是人类在开发软件的活动中智力和体力的扩展和延伸,为软件工程方法提供自动的或半自动的软件支撑环境;

软件工程的过程则是将软件工程的方法和工具综合起来以达到合理、及时地进行计算机软件开发的目的。

迄今为止,软件工程的研究与应用已经取得很大成就,它在软件开发方法、工具、管理等方面的应用大大缓解了软件危机造成的被动局面。