



# 软件工程及 软件开发应用

施一萍 主编

---

# 软件工程及 软件开发应用

---

主编 施一萍  
编者 施一萍 苏前敏 赵敏媛  
王 红 夏永祥

清华大学出版社  
北京

## 内 容 简 介

软件工程是计算机科学技术的一门新兴学科,其软件开发新技术、新方法正在不断地出现。本书以软件生命周期为主线,系统地介绍了软件工程基本知识、结构化分析、结构化设计的软件开发技术,同时介绍了面向对象开发技术,包括面向对象分析、面向对象设计、UML 建模语言和 Rational Rose 工具,最后介绍了通过软件测试、维护和工程管理来保证软件工程质量等内容。

本书内容新颖,实用性强,每章后有小结和习题。可作为高等院校“软件工程”课程的教材或参考书,也适用于软件工作人员和计算机用户阅读参考。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

## 图书在版编目(CIP)数据

软件工程及软件开发应用/施一萍主编. —北京:清华大学出版社,2009.9  
ISBN 978-7-302-20667-5

I. 软… II. 施… III. ①软件工程 ②软件开发 IV. TP311.5

中国版本图书馆 CIP 数据核字(2009)第 124611 号

责任编辑:张占奎

责任校对:刘玉霞

责任印制:王秀菊

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

<http://www.tup.com.cn>

邮 编:100084

社 总 机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者:北京鑫海金澳胶印有限公司

经 销:全国新华书店

开 本:185×260

印 张:13.75

字 数:332千字

版 次:2009年9月第1版

印 次:2009年9月第1次印刷

印 数:1~3000

定 价:25.00元

---

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。  
联系电话:010-62770177 转 3103 产品编号:032427-01

计算机学科一直处于高速发展的过程中,伴随知识更新速度的加快,计算机教学知识体系以及专业设置也在发生着迅速的变化。计算机从 20 世纪 80 年代前以硬件为主,发展到 20 世纪 90 年代以软件为主,再到现在软件引领发展的时代。软件工程是 20 世纪 60 年代末,为了克服“软件危机”而出现的一门研究软件开发与维护的工程学科。随着计算机技术的飞速发展,对软件开发和维护提出了更高的要求,软件工程和软件产业面临更大的挑战。软件工程是计算机科学技术的一门新兴的学科,其软件开发新技术、新方法不断涌现,因此,相应的软件工程的教材也要进行教材改革,以适应计算机学科发展的需要。

本书以软件生命周期为主线,主要介绍了结构化软件开发技术、面向对象的软件开发技术以及软件工程质量保证和管理的内容。本书有两个特点:一是内容新颖,反映了当前软件开发和管理的最新技术;二是实用性强,通过应用实例分析,使读者学习和掌握软件工程的原理、技术和方法,并对实际的软件开发工作起指导作用。

全书分 3 篇,共 12 章。

第 1 篇为软件开发方法,共有 5 章内容。第 1 章,软件工程概述,主要介绍软件工程起源和概念、软件生命周期及软件开发模型。第 2 章,可行性研究,主要介绍可行性研究的任务和过程、系统流程图、成本/效益分析和相应的文档编写。第 3 章,需求分析,主要介绍需求分析的任务和步骤、数据流图、数据字典等图形工具以及结构化分析方法。第 4 章,总体设计,主要介绍总体设计任务、软件设计概念和原理、层次图和结构图的图形工具以及结构化设计。第 5 章,详细设计,主要介绍结构化程序设计、过程设计的工具(流程图、N-S 图和 PAD 图等)、面向数据结构的设计方法、程序复杂程度的度量以及人机界面设计。

第 2 篇为面向对象开发方法,共有 4 章内容。第 6 章,面向对象方法学引论,主要介绍面向对象概念、面向对象方法学优点、面向对象建模(对象模型、动态模型和功能模型)。第 7 章,面向对象分析与设计,主要介绍面向对象分析过程、面向对象设计建立的子系统。第 8 章,UML 统一建模语言,主要介绍 UML 统一建模语言、静态模型图、动态模型图。第 9 章,Rational Rose 工具,主要介绍 Rose 工具的基本操作、模型视图和双向工程。

第 3 篇,软件工程质量保证与管理,共有 3 章内容。第 10 章,软件测试,主要介绍软件测试概念、分类和基本步骤、黑盒测试、白盒测试、测试用例设计、调试和面向对象测试。第 11 章,软件维护,主要介绍软件维护的概念、过程、可维护性以及软件再工程。第 12 章,软件工程管理,主要介绍软件项目管理、配置管理、质量、风险管理以及人力资源管理。

本书由施一萍主编。施一萍编写第 1~4 章;苏前敏编写第 5 章及第 10 章;赵敏媛编写第 6 章及第 7 章;王红编写第 8 章及第 9 章;夏永祥编写第 11 章及第 12 章。全书由施一萍统稿。

由于专业水平有限,书中错误或欠妥之处,望读者批评指正。

编 者

2009 年 5 月

## 第 1 篇 软件开发方法

第 1 章 软件工程概述	3
1.1 软件与软件危机	3
1.1.1 计算机软件概念	3
1.1.2 计算机软件的发展	4
1.1.3 软件危机	5
1.2 软件工程	6
1.2.1 软件工程定义	6
1.2.2 软件工程的内容	6
1.2.3 软件工程基本原理	7
1.2.4 软件开发方法	8
1.3 软件生命周期	9
1.3.1 软件生命周期的划分	9
1.3.2 各阶段的基本任务	9
1.4 软件开发模型	11
1.4.1 瀑布模型	11
1.4.2 快速原型模型	12
1.4.3 增量模型	13
1.4.4 螺旋模型	14
1.4.5 喷泉模型	16
1.5 应用实例	16
1.6 小结	17
习题	17
第 2 章 可行性研究	19
2.1 可行性研究的任务	19
2.2 可行性研究的过程	20
2.3 系统流程图	21

2.4	成本/效益分析	23
2.4.1	成本估算	23
2.4.2	成本/效益分析方法	24
2.5	可行性研究报告	25
2.6	软件立项和合同	26
2.6.1	立项方法和立项文档	26
2.6.2	签订合同方法和合同文档	27
2.7	项目开发计划	29
2.8	小结	31
	习题	31
<b>第3章</b>	<b>需求分析</b>	<b>32</b>
3.1	需求分析概述	32
3.1.1	需求分析的任务	32
3.1.2	需求分析的步骤	33
3.2	图形工具	34
3.2.1	数据流图	34
3.2.2	数据字典	37
3.2.3	实体-联系图	38
3.2.4	其他图形工具	39
3.3	结构化分析	41
3.3.1	结构化分析的步骤	41
3.3.2	应用实例	42
3.3.3	软件需求规格说明书	43
3.4	验证软件需求	44
3.5	小结	45
	习题	46
<b>第4章</b>	<b>总体设计</b>	<b>47</b>
4.1	总体设计的过程和任务	47
4.2	软件结构设计概念和原理	48
4.2.1	模块和模块化	48
4.2.2	抽象	49
4.2.3	信息隐藏	50
4.2.4	模块独立	50
4.3	软件结构设计优化	53
4.4	图形工具	55
4.4.1	层次图和 HIPO 图	55
4.4.2	结构图	56

4.5	结构化设计	57
4.5.1	结构化设计概述	57
4.5.2	结构化设计的过程	58
4.5.3	变换分析和事务分析	60
4.6	总体设计说明书	61
4.7	小结	62
	习题	62
<b>第5章</b>	<b>详细设计</b>	<b>64</b>
5.1	任务和原则	64
5.2	结构化程序设计	64
5.3	过程设计的工具	65
5.3.1	程序流程图	65
5.3.2	N-S图	66
5.3.3	PAD图	68
5.3.4	PDL语言	69
5.3.5	判定树	70
5.3.6	判定表	71
5.4	面向数据结构的设计方法	72
5.4.1	逻辑数据结构	72
5.4.2	Jackson图和图解逻辑	73
5.4.3	应用实例	76
5.5	程序复杂程度的度量	80
5.6	人机界面设计	81
5.7	小结	85
	习题	85

## 第2篇 面向对象开发方法

<b>第6章</b>	<b>面向对象方法学引论</b>	<b>91</b>
6.1	面向对象方法学概述	91
6.1.1	面向对象的基本概念	92
6.1.2	面向对象软件过程模型	95
6.2	面向对象方法学的优点	96
6.3	面向对象建模	98
6.3.1	对象模型	98
6.3.2	动态模型	99
6.3.3	功能模型	99

6.3.4	三种模型之间的关系	100
6.4	小结	100
	习题	101
<b>第7章</b>	<b>面向对象分析与设计</b>	<b>102</b>
7.1	面向对象分析	102
7.1.1	面向对象分析过程	102
7.1.2	需求陈述	103
7.1.3	建立对象模型	104
7.1.4	建立动态模型	107
7.1.5	建立功能模型	110
7.1.6	应用实例	111
7.2	面向对象设计	115
7.2.1	面向对象设计的准则	116
7.2.2	问题域子系统设计	120
7.2.3	人机交互子系统设计	120
7.2.4	任务管理子系统设计	120
7.2.5	数据管理子系统设计	121
7.2.6	应用实例	122
7.3	小结	126
	习题	127
<b>第8章</b>	<b>UML 统一建模语言</b>	<b>128</b>
8.1	UML 概述	128
8.1.1	UML 的产生及发展目标	128
8.1.2	UML 的组成及主要内容	129
8.1.3	UML 的应用领域	131
8.2	UML 静态建模	131
8.2.1	用例图	131
8.2.2	类图	133
8.2.3	对象图	136
8.2.4	构件图	137
8.2.5	实施图	137
8.3	UML 动态建模	138
8.3.1	顺序图	138
8.3.2	协作图	139
8.3.3	状态图	140
8.3.4	活动图	140
8.4	基于 UML 的软件开发过程	141



8.5 小结 .....	143
习题 .....	143
<b>第9章 Rational Rose 工具 .....</b>	<b>144</b>
9.1 Rose 建模功能及特点 .....	144
9.2 Rose 操作界面 .....	144
9.2.1 Rose 的主界面 .....	144
9.2.2 Rose 的基本操作 .....	146
9.3 Rose 的模型视图 .....	148
9.3.1 用例图的创建、删除和设计 .....	148
9.3.2 模型元素的添加 .....	149
9.3.3 工具栏按钮的添加 .....	152
9.4 Rose 的双向工程 .....	154
9.5 小结 .....	158
习题 .....	158

### 第3篇 软件工程质量保证与管理

<b>第10章 软件测试 .....</b>	<b>161</b>
10.1 软件测试概述 .....	161
10.1.1 软件测试的概念 .....	162
10.1.2 软件测试的分类 .....	163
10.1.3 软件测试的基本步骤 .....	163
10.2 白盒测试 .....	166
10.2.1 逻辑覆盖 .....	167
10.2.2 路径测试 .....	167
10.3 黑盒测试 .....	169
10.3.1 等价分类法 .....	169
10.3.2 边界值分析法 .....	170
10.3.3 错误推测法 .....	170
10.4 测试用例 .....	170
10.4.1 测试用例概念 .....	170
10.4.2 应用实例 .....	171
10.5 调试 .....	172
10.6 面向对象的测试 .....	173
10.7 软件测试的发展趋势 .....	175
10.8 小结 .....	176
习题 .....	176

<b>第 11 章 软件维护</b>	179
11.1 软件维护的概念	179
11.2 软件维护的工作量	181
11.3 软件维护过程	181
11.4 软件可维护性	184
11.5 软件再工程	185
11.5.1 软件再工程的概念	186
11.5.2 软件重构	187
11.6 应用实例	188
11.7 小结	188
习题	189
<b>第 12 章 软件工程管理</b>	190
12.1 软件工程管理的目标	190
12.2 软件项目管理	190
12.2.1 软件项目管理的目标和过程	191
12.2.2 软件项目失控的原因	192
12.3 软件配置管理	192
12.3.1 软件配置管理的定义	192
12.3.2 软件配置管理的任务	193
12.4 软件质量管理	195
12.4.1 软件质量的概述	195
12.4.2 软件质量管理的主要活动	196
12.4.3 软件质量保证	198
12.4.4 软件质量保证体系及认证	199
12.4.5 软件过程能力评估的模型	200
12.5 软件风险管理	202
12.5.1 软件项目风险的概念	202
12.5.2 软件风险管理过程及风险驾驭	203
12.6 软件项目人力资源管理	205
12.7 应用实例	206
12.8 小结	206
习题	207
<b>缩略语</b>	208
<b>参考文献</b>	209

Part 1

第 1 篇

# 软件开发方法

第 1 章 软件工程概述

第 2 章 可行性研究

第 3 章 需求分析

第 4 章 总体设计

第 5 章 详细设计



# 软件工程概述

软件工程是 20 世纪 60 年代末,为了克服“软件危机”而出现的一门研究软件开发与维护的工程学科。随着计算机技术的飞速发展,对软件开发和维护提出了更高的要求,软件工程和软件产业面临更大的挑战。本章主要介绍软件工程的基本知识,包括软件的概念、软件危机、软件工程的内容、基本原理以及软件开发的方法和模型。

## 1.1 软件与软件危机

### 1.1.1 计算机软件概念

计算机系统是由硬件和软件组成。计算机硬件是计算机系统的物理部件,如中央处理器、存储器、键盘、显示器和打印机。计算机软件是计算机系统逻辑部件。计算机硬件和软件有明显的不同,具体表现如下:

(1) 硬件是物理实体,能够看得见,而软件是逻辑实体。

(2) 硬件会“磨损”或“老化”,但软件存在“退化”问题,必须要多次修改和维护软件。

(3) 硬件与软件的制造过程不同。硬件需要相应的设计、工艺和制造过程,软件是通过分析、设计和编写程序完成的。

(4) 硬件与软件的质量控制过程不同。由于制造过程不同,所以它们的质量控制也在各个阶段完成。

计算机硬件和软件又是相互联系、相互依赖的,二者缺一不可。硬件如果没有相应软件的支持,只能称为“裸机”,它是不能正常工作的。计算机系统只有配备了硬件以及控制和管理硬件的软件,才能发挥作用。

计算机系统的发展离不开软件的发展。计算机软件通常定义如下:软件是程序、数据和相关文档的完整集合。其中程序是按照设计文档的功能和性能要求所编写的指令序列。数据是能使程序正常操纵信息的数据结构。文档是与程序开发、管理、维护和使用有关的图文材料。软件又与程序不同,它的规模比较庞大。

计算机软件的主要特点如下:

(1) 软件是一种逻辑实体,具有抽象性。人们可以把它记录在介质上,但却无法看到软件的形态,只有通过分析、判断才能很好地理解它的功能和性能。

(2) 软件的生产与硬件不同,它没有明显的制造过程,它是通过人的智力劳动,将知识和技术转化为信息的一种产品。它一旦研制成功,就可以重复复制。

(3) 软件规模庞大,维护比较困难。如果软件运行中发现了错误,必须改正,这将花费很大的工作量。

(4) 软件成本昂贵。

计算机软件又分为系统软件和应用软件。系统软件是控制计算机系统各部件协调、高效工作的软件,如操作系统、数据库管理系统和编译系统等。应用软件是为特定的应用目的而开发的软件,如企业信息管理系统和计算机辅助设计系统等。

### 1.1.2 计算机软件的发展

自从1946年第一台计算机ENIAC问世以来,计算机系统经历了四个不同的发展阶段,其中计算机软件系统也经历了不同的变化。

20世纪50到60年代是计算机软件发展的早期时代,通用硬件相当普遍,用汇编语言编程逐步取代用二进制的机器语言,程序是为每个具体应用而专门编写的。这时的软件通常是规模较小的程序,编写者和使用者往往是同一个(或同一组)人。这种个体化的软件环境,使得软件设计通常是在人们头脑中进行的一个隐含的过程,除了程序清单之外,没有其他文档资料保存下来。

20世纪60年代中期到70年代中期是计算机软件系统发展的第二个阶段,这个阶段的一个重要特征是出现了“软件作坊”,广泛使用产品软件。这个时期的软件开发语言发展很快,不仅使用汇编语言,而且使用面向过程的程序设计语言。这一阶段出现了多用户系统、数据库、实时监控系统等软件产品。但是,“软件作坊”基本上仍然沿用早期形成的个体化软件开发方法。因此,开发出来的软件质量低下,一旦软件交付给用户使用后,会出现这样那样的错误,而要使程序正常运行,必须设法改正这些错误;当用户有了新的需求时,必须相应地修改程序;当硬件或操作系统更新时,通常需要修改程序以适应新的环境。上述种种的软件维护工作,实在是耗时耗力。更严重的是,许多程序的个体化特性使得它们最终不可维护。“软件危机”就这样开始出现了!1968年北大西洋公约组织的计算机科学家在联邦德国召开国际会议,讨论软件危机问题,在这次会议上正式提出并使用了“软件工程”这个名词。

20世纪70年代中期到90年代是软件发展的第三个阶段,这一阶段,计算机硬件成本大幅度下降,用户对软件需求越来越高。软件开发技术不断改进,人们在开发软件时开始遵循软件工程的思想,并采用结构化技术进行软件的开发。

20世纪90年代以来,软件产业迅猛发展,计算机成了人们工作、学习和获取信息时不可缺少的工具。新的软件开发技术不断出现,出现了面向对象的技术和快速原型法技术,并在软件开发中得到了广泛应用,一些先进的软件系统(如专家系统、神经网络、通信和网络软件等)不断出现并得到应用。

### 1.1.3 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。这些问题绝不仅仅是不能正常运行的软件才具有的,实际上,几乎所有软件都不同程度地存在这些问题。

概括地说,软件危机包含下述两方面的问题:如何开发软件,以满足对软件日益增长的需求;如何维护数量不断膨胀的已有软件。

具体地说,软件危机主要有以下一些表现:

(1) 对软件开发成本和进度的估计常常很不准确。由于缺乏软件开发经验,很难制定软件开发计划和进度,估计的成本有可能比实际成本少一个数量级,预期进度比实际进度快几个月甚至几年的现象并不罕见。

(2) 用户对“已完成的”软件系统不满意的现象经常发生。软件开发人员在对用户需求未做深入了解,甚至对所要解决的问题还没有确切认识的情况下,就匆忙设计、编写程序。软件开发人员和用户之间的信息交流往往很不充分,这样必然会导致最终的产品不符合用户的实际需要。

(3) 软件产品的质量往往靠不住。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,这些都导致软件产品发生质量问题。

(4) 软件常常是不可维护的。很多程序中的错误很难改正,为了适应新的硬件环境,要想根据用户的需要在原有程序中增加一些新的功能也很难。人们仍然在重复开发类似的或基本类似的软件,开发“可重用的软件”成为人们努力追求的目标。

(5) 软件通常没有适当的文档资料。计算机软件不仅仅是程序,还应该有一整套文档资料。这些文档资料应该是在软件开发过程中产生的。软件开发组织的管理人员可以使用这些文档资料作为“里程碑”,来管理和评价软件开发工程的进展状况;软件开发人员可以利用它们作为通信工具,在软件开发过程中准确地交流信息;对于软件维护人员而言,这些文档资料可以给软件开发和维护带来许多的便利。

(6) 软件成本在计算机系统总成本中所占的比例逐年上升。由于微电子学技术的进步和生产自动化程度不断提高,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。

(7) 软件开发生产率提高的速度远远跟不上计算机应用迅速普及及深入的趋势。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。

产生软件危机的原因是复杂的,但基本可以分为两方面:一方面是与软件本身的特点有关,另一方面也和软件开发与维护的方法不正确有关。软件本身与硬件不同,它是计算机系统逻辑部件而不是物理部件。同时,软件又与一般程序不同,它规模庞大,而且程序复杂性将随着程序规模的增加而呈指数上升。在软件开发和维护过程中采用的错误方法包括:忽视软件需求分析的重要性,轻视软件维护,简单地认为软件开发就是编写程序并设法使之运行。

为了解决软件危机,既要有技术措施,包括软件开发的方法和工具,又要有必要的组织管理措施。软件工程正是从管理和技术两方面研究如何更好地开发和维护计算机软件。

## 1.2 软件工程

### 1.2.1 软件工程定义

软件工程是指导计算机软件开发和维护的一门工程学科。其定义如下:采用工程的概念、原理、技术和方法来开发与维护软件,把经过时间考验而证明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,经济地开发出高质量的软件并有效地维护。

人们曾经给软件工程下过许多定义,下面给出几个典型的定义。

(1) 1968年在第一届NATO会议上曾经给出了软件工程的一个早期定义:“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的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。

(2) 1983年IEEE的软件工程定义:“软件工程是开发、运行、维护和修复软件的系统方法。”

(3) 1993年IEEE进一步给出了一个更全面更具体的定义:“软件工程是研究如何以系统化的、规范的、可度量的方法应用于软件开发、运行和维护过程,也就是把工程应用于软件上。”

### 1.2.2 软件工程的内容

软件工程主要包括三方面的内容:软件开发方法、软件开发模型和软件工具。软件开发方法是在软件开发过程中所采用的技术,如结构化技术和面向对象技术。软件开发模型是描述软件开发过程中各种活动如何执行的模型,如瀑布模型、渐增模型和螺旋模型。软件工具是在软件开发过程中采用的工具,如建模工具UML和Rational Rose。

软件工程方法学包含3个要素:方法、工具和过程。方法是完成软件开发的各项任务的技术方法,回答“怎样做”的问题。工具是为运用方法而提供的自动的或半自动的软件工程支撑环境。过程是为了获得高质量的软件所需要完成的一系列任务的框架,它规定了完成各项任务的工作步骤。

2001年5月ISO/IEC JTC发布了《SWEBOK指南V0.95(试用版)》,即Guide to the Software Engineering Body of Knowledge。SWEBOK把软件工程学科的主体知识分为10个知识领域,这10个领域包括:软件需求、软件设计、软件构造、软件测试、软件维护、软件配置管理、软件工程管理、软件工程过程、软件工程工具和方法以及软件质量。



### 1.2.3 软件工程基本原理

自从 1968 年在联邦德国召开的国际会议上正式提出并使用了“软件工程”这个术语以来,研究软件工程的专家学者们陆续提出了 100 多条关于软件工程的准则或“信条”。著名的软件工程专家 B. W. Boehm 综合这些学者们的意见并总结了 TRW 公司多年开发软件的经验,于 1983 年提出了软件工程的 7 条基本原理。

软件工程的基本原理如下:

(1) 用分阶段的生命周期计划严格管理。把软件生命周期划分成若干个阶段,并相应地制定出切实可行的计划,然后严格按照计划对软件的开发与维护工作进行管理。不同层次的管理人员都必须严格按照计划各尽其职地管理软件开发与维护工作,绝不能受客户或上级人员的影响而擅自背离预定计划。其中计划包括 6 类: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。在每个阶段都进行严格的评审,若上一阶段评审通不过,就不能进入下一阶段的工作。错误发现与改正得越早,所付出的代价就越低。

(3) 实行严格的产品控制。在软件开发过程中,不应随意改变需求。如要改变需求,为了保持软件各个配置成分的一致性,必须按照严格的规程进行评审,获得批准以后才能实施修改。将影响软件质量的因素在整个过程中置于严格控制之下。

(4) 采用现代程序设计技术。现代程序设计技术就是结构化技术,包括结构化分析、结构化设计、结构化编码和结构化测试。采用先进的技术不仅可以提高软件开发和维护的效率,而且可以提高软件产品的质量。

(5) 结果应能清楚地审查。为了提高软件开发过程的可见性,更好地进行管理,应该根据软件开发项目的总目标及完成期限,规定开发组织的责任和产品标准,从而使得所得到的结果能够清楚地审查。

(6) 开发小组的人员应该少而精。开发小组人员的素质和数量是影响软件产品质量和开发效率的重要因素。开发组织不在人多,在于每个人的技能符合要求。同时用人少而精,可减少沟通路径,提高生产率。

(7) 承认不断改进软件工程实践的必要性。要积极主动地采纳新的软件技术,注意不断总结经验,改进开发的组织和过程,有效地通过过程质量的改进提高软件产品的质量。

上述 7 条原理是在面向过程的程序设计时代提出来的,但是在目前出现了面向对象程序设计的时代仍然有效。还有一条基本原理在软件的开发和管理中特别重要,需要补充进去,作为软件工程的第(8)条基本原理。

(8) 二八定律。对软件项目进度和工作量的估计:一般人主观上认为已经完成了 80%,但实际上只完成了 20%。对程序中存在问题的估计:80%的问题存在于 20%的程序之中。对模块功能的估计:20%的模块实现了 80%的功能。对人力资源的估计:20%的人解决了软件中 80%的问题。对投入资金的估计:企业信息系统中 80%的问题,可以用 20%的资金来解决。在软件开发和管理的历史上有无数的案例都验证了二八定律。所以软件工程发展到今天,可以认为它的基本原理共有 8 条。