

21世纪

高等院校计算机系列教材

实用软件 工程教程

阎 菲 主 编
潘正清 吴年志 副主编



中国水利水电出版社
www.waterpub.com.cn

21 世纪高等院校计算机系列教材

实用软件工程教程

阎 菲 主编

潘正清 吴年志 副主编

中国水利水电出版社

内 容 提 要

本书强调软件工程的理论与实践相结合、技术与管理相结合、方法与设计工具相结合的理念,并以应用性极强的系统为例,全面地阐述了软件工程的内涵;同时自始至终贯穿反映学科最新发展方向的新内容,加强了计算机网络、面向对象软件技术实用技术的编写。

全书内容主要包括:软件工程的基础知识,软件过程,软件工程项目管理,计算机系统工程,需求分析,总体设计,详细设计,编码与测试,软件维护,面向对象技术,快速原型技术,软件重用技术等。本书努力做到内容精练,重点突出,概念清楚,针对性和实际操作性强。

本书可作为本科计算机及相关专业学生的教材,也可作为软件开发人员的参考书。

本书所配电子教案可以从中国水利水电出版社网站上免费下载,网址为:
<http://www.waterpub.com.cn/softdown/>。

图书在版编目(CIP)数据

实用软件工程教程 / 阎菲主编. —北京:中国水利水电出版社, 2006

(21世纪高等院校计算机系列教材)

ISBN 7-5084-3930-9

I. 实... II. 阎... III. 软件工程—高等学校—教材 IV. TP311.5

中国版本图书馆CIP数据核字(2006)第078591号

书 名	实用软件工程教程
作 者	阎 菲 主 编 潘正清 吴年志 副主编
出版 发行	中国水利水电出版社(北京市三里河路6号 100044) 网址: www.waterpub.com.cn E-mail: mchannel@263.net (万水) sales@waterpub.com.cn 电话: (010) 63202266 (总机)、68331835 (营销中心)、82562819 (万水)
经 售	全国各地新华书店和相关出版物销售网点
排 版	北京万水电子信息有限公司
印 刷	北京市天竺颖华印刷厂
规 格	787mm×1092mm 16开本 19.5印张 471千字
版 次	2006年8月第1版 2006年8月第1次印刷
印 数	0001—4000册
定 价	28.00元

凡购买我社图书,如有缺页、倒页、脱页的,本社营销中心负责调换

版权所有·侵权必究

前 言

社会经济和计算机技术迅速发展,国际国内信息化持续向前推进,使得软件建设的人才十分紧缺,软件工程的理论与方法不断遇到新的挑战。作为高校大学生的教材,本书力求做到既反映这个领域新的发展,又能使学生掌握软件工程的基本方法与技能,培养学生分析解决本专业实际问题的能力。

本书的特点是反映学科最新发展方向的新内容,加强计算机网络、面向对象软件实用技术,对统一建模语言(UML)和统一过程(RUP)作了深入浅出的介绍,进一步强调软件工程的理论与实践相结合、技术与管理相结合、方法与设计工具相结合的理念。

本书努力做到:内容安排适当,呈现方式清晰,让学生很快找出需要的章节;文笔流畅,用语准确,加强学生对重点、难点的理解。

全书主要内容如下:

第1~2章是软件工程基础,主要讨论软件工程的基础理论、软件工程的过程与软件工程项目管理。

第3~4章论述计算机工程及需求分析。

第5~10章讨论软件系统分析、设计与系统实施的方法,软件工程开发的项目管理与运行维护的基本知识。

第11~14章讨论面向对象方法。

第15~16章构成本书的最后一部分,专门介绍比较新颖、颇具发展潜力的软件开发技术。

本书每章都附有习题,其中有的是复习思考题,有的可作为课外作业或课程设计的内容。

成功的软件工作人员无不经过大量工程实践的锻炼,只有在学习书本内容的同时辅以相应的实际练习和实验环节,才能真正掌握书中介绍的知识和技能。为指导读者进行软件工程实践,本教材提供了相应的软件工程素材。

本书由阎菲任主编,潘正清、吴年志任副主编。各章主要编写人员分工如下:第1、3、4、11、12、13章由阎菲编写,第2、5、6、7、8、9、10、14、16章和附录由潘正清编写,第15章由吴年志编写,阎菲、潘正清统稿并校对。参加本书编写的还有:向郑涛、陈宇峰、裘子煦、薛礼、胡志慧、刘振章、李发海、周应仙、曹霞、史建芳、雷平、张海涛、王丽、吴涛、赵峰林等。

由于时间仓促,水平有限,书中难免有错误和不妥之处,敬请读者与专家指正。如有意见和建议,请与作者联系。作者的E-mail: yanfei131@163.com。

编 者

2006年6月

目 录

前言

第 1 章 软件工程概述	1
1.1 软件工程中的常见问题	1
1.1.1 软件	2
1.1.2 软件危机	2
1.2 软件工程	4
1.2.1 软件工程的定义与原理	4
1.2.2 软件工程与计算机科学	5
1.2.3 软件工程目标	6
1.2.4 软件工程面临的挑战	7
1.3 软件过程和软件生命周期	7
1.3.1 软件过程	7
1.3.2 软件生命周期	8
1.4 软件开发模型	10
1.4.1 瀑布模型	10
1.4.2 演化模型	11
1.4.3 原型模型	12
1.4.4 螺旋模型	13
1.4.5 喷泉模型及面向对象的 开发过程	14
1.4.6 基于四代技术的模型	14
1.4.7 智能模型	15
1.5 职业道德	15
1.6 本章小结	16
习题	16
第 2 章 软件项目管理	17
2.1 软件项目管理概述	17
2.1.1 软件项目管理的概念	17
2.1.2 软件项目管理的特点	18
2.1.3 软件项目管理的内容	19
2.1.4 项目的组织模式	20
2.1.5 项目的人员管理	21
2.1.6 软件配置管理	22
2.2 软件项目规划	23

2.2.1	项目规划的基本概念	23
2.2.2	项目规划的步骤	24
2.2.3	项目规划的原则	24
2.2.4	项目规划的形成文件	25
2.2.5	项目规划过程之间的相互联系	27
2.3	软件项目风险管理	27
2.3.1	风险管理概述	27
2.3.2	软件项目的风险策略	28
2.3.3	软件风险	28
2.3.4	风险管理的过程	28
2.3.5	软件风险的总结	35
2.4	软件项目进度管理	35
2.4.1	项目进度的基本概念	35
2.4.2	制定进度计划	36
2.4.3	面向网络化管理	36
2.4.4	建立里程碑事件	37
2.4.5	实施保证	38
2.5	软件项目质量管理	38
2.5.1	基本定义	38
2.5.2	质量管理主要内容	39
2.5.3	技术与方法	41
2.5.4	质量管理实施	43
2.6	本章小结	46
	习题	46
第3章	软件系统及可行性研究	47
3.1	用户中心设计——UCD 方法	47
3.2	计算机系统	48
3.2.1	硬件和硬件工程	50
3.2.2	软件和软件工程	51
3.2.3	人机工程	54
3.2.4	数据库和数据库工程	54
3.3	可行性研究	55
3.3.1	可行性研究的任务	55
3.3.2	技术可行性	56
3.3.3	操作可行性	56
3.3.4	经济可行性	56
3.3.5	可行性研究的步骤	58
3.3.6	方案选择	59
3.4	系统开发过程的新体系结构——快速原型与面向对象	60

3.5 本章小结	61
习题	62
第4章 需求分析	63
4.1 需求分析基础	63
4.1.1 分析的任务与原则	63
4.1.2 需求建模	67
4.1.3 开发原型系统	68
4.2 结构化分析方法	68
4.2.1 结构化分析的策略	69
4.2.2 数据流图	70
4.2.3 数据字典	73
4.2.4 小说明	75
4.2.5 结构化分析实施步骤	75
4.3 其他分析方法与图形工具	77
4.4 实体—关系图	80
4.4.1 数据对象、属性与关系	80
4.4.2 E-R 方法和实体模型	81
4.4.3 数据结构的规范化	82
4.5 需求规格说明与评审	83
4.5.1 需求规格说明书的目标与内容	84
4.5.2 需求评审	86
4.6 本章小结	87
习题	87
第5章 总体设计	89
5.1 总体设计的任务及过程	89
5.1.1 设计供选择的方案	89
5.1.2 功能解剖和设计软件结构	90
5.1.3 数据库设计	90
5.1.4 制定测试计划	91
5.1.5 文档的编写	91
5.1.6 进行审查和复查	91
5.2 软件设计基本原理	91
5.2.1 模块化	91
5.2.2 抽象	93
5.2.3 信息隐蔽	94
5.2.4 模块独立性	94
5.3 设计准则	97
5.4 总体设计的图形描述工具	100
5.4.1 层次图	100

5.4.2	HIPO 图	101
5.4.3	结构图	101
5.5	结构化设计方法	102
5.5.1	数据流图的类型	103
5.5.2	设计步骤	104
5.5.3	变换设计	105
5.5.4	事务设计	110
5.5.5	设计的后处理	111
5.6	本章小结	112
	习题	113
第 6 章	详细设计	114
6.1	结构化程序设计	114
6.2	详细设计工具	116
6.2.1	程序流程图	116
6.2.2	盒图 (N-S 图)	117
6.2.3	PAD 图	118
6.2.4	过程设计语言 (PDL)	120
6.2.5	判定表	120
6.2.6	判定树	121
6.3	Jackson 程序设计方法	121
6.3.1	Jackson 方法	122
6.3.2	Jackson 图	122
6.3.3	Jackson 程序设计方法	123
6.4	Warnier 程序设计方法简介	127
6.4.1	Warnier 图	127
6.4.2	Warnier 程序设计方法的步骤	127
6.5	人机界面设计	128
6.5.1	人机界面设计概述	128
6.5.2	人机界面风格	130
6.5.3	人机界面实现的原则	130
6.5.4	人机界面的设计过程	132
6.6	本章小结	134
	习题	134
第 7 章	软件编码	136
7.1	程序设计语言	136
7.1.1	程序设计语言概述	136
7.1.2	程序设计语言的分类	137
7.1.3	程序设计语言的特点	140
7.1.4	程序设计语言的选择	143

7.2	程序设计风格	144
7.3	编码的质量要求	147
7.4	编码工具	148
7.5	本章小结	149
	习题	149
第 8 章	软件测试	150
8.1	软件测试的概述	150
8.1.1	软件测试的定义	150
8.1.2	软件测试的基本原则	151
8.1.3	软件测试的步骤	152
8.1.4	软件测试信息流	152
8.1.5	测试人员组织	153
8.2	软件测试方法	154
8.2.1	黑盒测试	154
8.2.2	白盒测试	155
8.3	测试用例设计	156
8.3.1	逻辑覆盖	156
8.3.2	等价类划分	158
8.3.3	边界值分析	159
8.3.4	因果图	160
8.3.5	错误推测法	160
8.4	软件测试的步骤	161
8.4.1	单元测试	161
8.4.2	集成测试	163
8.4.3	有效性测试	166
8.4.4	系统测试	167
8.5	调试	168
8.5.1	调试的步骤	168
8.5.2	调试的策略	168
8.6	软件可靠性	170
8.6.1	软件可靠性的定义	171
8.6.2	软件正确性证明	171
8.7	验收测试	171
8.7.1	验收测试的步骤	172
8.7.2	Alpha 测试和 Beta 测试	173
8.8	测试工具	173
8.8.1	人工测试	173
8.8.2	自动测试工具	175
8.9	本章小结	176

习题	176
第 9 章 软件维护	177
9.1 软件维护的概念	177
9.1.1 软件维护的定义	177
9.1.2 软件维护的种类	177
9.1.3 影响维护工作量的因素	178
9.1.4 软件维护的策略	179
9.2 软件维护的特点	180
9.2.1 软件工程与软件维护的关系	180
9.2.2 维护成本	181
9.2.3 维护的问题	181
9.3 软件维护过程	182
9.3.1 维护机构	182
9.3.2 维护申请报告	183
9.3.3 维护的工作流程	183
9.3.4 维护记录	184
9.3.5 维护评价	185
9.4 软件可维护性	185
9.4.1 软件可维护性的度量	185
9.4.2 提高软件可维护性的方法	186
9.5 软件维护的副作用	187
9.6 本章小结	189
习题	189
第 10 章 软件过程改进	190
10.1 基本概念	190
10.2 软件过程的改进方法	191
10.2.1 CMM 介绍	191
10.2.2 SPICE 介绍	195
10.2.3 AGILE 介绍	197
10.3 本章小结	198
习题	199
第 11 章 面向对象概念	200
11.1 传统方法学与面向对象方法比较	200
11.1.1 传统方法学	200
11.1.2 面向对象方法学	201
11.2 面向对象基础概念	203
11.2.1 对象	203
11.2.2 其他概念	205
11.3 面向对象的特性	207

11.3.1 抽象性	207
11.3.2 封装性	209
11.3.3 继承性	209
11.3.4 多态性	210
11.4 本章小结	211
习题	212
第 12 章 面向对象的分析	213
12.1 面向对象建模	213
12.2 对象模型	214
12.2.1 表示类—&—对象的图形符号	214
12.2.2 表示结构的图形符号	214
12.2.3 实例	217
12.3 动态模型	218
12.3.1 术语	218
12.3.2 表示方法	219
12.4 功能模型	220
12.5 UML 概述	220
12.5.1 UML 的主要内容	220
12.5.2 UML 的主要元素	221
12.6 UML 的图形	222
12.6.1 用例图	223
12.6.2 类图	224
12.6.3 序列图	224
12.6.4 状态图与活动图	225
12.7 UML 的示例	226
12.8 本章小结	227
习题	228
第 13 章 面向对象设计	229
13.1 从面向对象分析到面向对象设计	230
13.2 问题域部分 (PDC) 设计之一——基础理论	231
13.2.1 为什么需要问题域部分设计	232
13.2.2 如何进行问题域部分设计	233
13.3 人机交互部分 (HIC) 设计	235
13.3.1 为什么需要人机交互部分	235
13.3.2 如何设计人机交互部分	236
13.4 任务管理部分 (TMC) 设计	238
13.4.1 为什么需要有任务管理部分	238
13.4.2 怎样设计任务管理部分	238
13.5 数据管理部件 (DMC) 的设计	239

13.6 面向对象设计的实例	240
13.6.1 系统总体结构和使用模型	241
13.6.2 体系结构的设计	242
13.6.3 识别对象的类	243
13.6.4 设计模型	245
13.6.5 对象接口描述	247
13.7 本章小结	248
习题	249
第 14 章 面向对象测试	250
14.1 面向对象测试的基本概念	250
14.1.1 概述	250
14.1.2 测试模型简介	251
14.2 面向对象测试用例设计	254
14.2.1 测试用例设计的复杂性	254
14.2.2 类测试用例设计	254
14.3 面向对象的测试分析工具	255
14.3.1 面向对象的测试分析工具 LOGISCOPE 简介	255
14.3.2 面向对象的测试分析工具 WinRunner 简介	257
14.3.3 面向对象的测试分析工具 LoadRunner 简介	258
14.4 本章小结	260
习题	260
第 15 章 软件重用技术	261
15.1 软件重用	261
15.1.1 软件重用的概念和意义	261
15.1.2 软件重用技术的理解	261
15.1.3 软件重用的过程	263
15.1.4 可重用软件的设计过程	264
15.2 软部件库的构造	265
15.2.1 基于域分析的软部件选取	265
15.2.2 软部件的开发	265
15.2.3 软部件库的组织与检索	266
15.3 面向对象的软件重用技术	268
15.4 本章小结	269
习题	269
第 16 章 软件工程的深入研究	270
16.1 软件的再生工程	270
16.1.1 重构	270
16.1.2 逆向工程	270
16.1.3 前向工程	271

16.2 基于 C/S 与 B/S 的软件工程	271
16.2.1 基于 C/S 的软件开发	271
16.2.2 基于 B/S 的 Web 系统开发	272
16.3 计算机辅助软件工程 (CASE)	274
16.3.1 定义	274
16.3.2 CASE 工具在软件开发中的作用	274
16.3.3 CASE 工具的分类	274
16.3.4 CASE 工具的定位	275
16.4 基于构件的软件工程	276
16.4.1 构件技术	276
16.4.2 基于构件的开发方法	277
16.4.3 构件模型	277
16.4.4 构件标准规范模型简介	278
16.5 本章小结	281
习题	281
附录 1 可行性研究报告规范	282
附录 2 项目开发计划规范	285
附录 3 需求规格说明书规范	287
附录 4 概要设计说明书规范	290
附录 5 详细设计说明规范	292
附录 6 项目开发总结报告规范	294
附录 7 用 Word 撰写文档规范	296
参考文献	299

第 1 章 软件工程概述

计算机学科是 20 世纪发展最快的新兴学科，在短暂的 50 年里，计算机已经渗透到社会的各个领域，有力地推动了整个社会信息化的发展。进入 21 世纪以来，经济全球化的趋势加速，现代科学技术突飞猛进，市场竞争更加剧烈，人类面临着巨大的社会经济变革。在这个复杂多变的时代，每个人、每个社会组织对信息资源的开发利用能力已成为其竞争力的主要标志之一。

随着技术进步，计算机性能在不断地提高，计算机的体积、功耗、价格却不断下降。今天的计算机在科学计算、数据处理、过程控制、计算机辅助系统、人工智能等领域得到广泛的应用。尤其计算机与全球 Internet 相连接，使今天的社会进入了以计算机为核心的信息社会。在信息社会中，信息的获取、处理、交流和决策都需要大量高质量的计算机软件，软件系统无所不在。这样就促使人们对计算机软件的品种、数量、功能、质量、成本和开发时间等提出越来越高的要求。

为了使世界上丰富的软件资源为人类共享，人们越来越重视软件、软件开发及运行环境的标准化。计算机的各类程序设计语言 and 多媒体人机交互工具已被越来越多的人所掌握，成为世界性的文化现象。

软件工程是在 20 世纪 60 年代末期提出的。这一概念的提出，其目的是倡导以工程的原理、原则和方法进行软件开发，以期解决当时出现的“软件危机”。

1.1 软件工程中的常见问题

本节讨论软件工程的常见问题，并希望根据国内外专家的观点，按照“FAQ(常见问题表)”给出简明的答案，然后逐次加以简述，深入的研究在各章节进行。软件工程中常见问题如表 1.1 所示。

表 1.1 软件工程中常见问题（一）

问题	答案
什么是软件	计算机程序和相关文档。软件产品可为特定客户或通用市场开发
什么是软件危机	“软件危机”一词在 IT 界广为流传，主要针对软件代价高和软件错误多现象
什么是软件工程	软件工程是关于软件生产的各个方面的工程学科
软件工程和计算机科学有何区别	计算机科学侧重理论和基础，而软件工程侧重于软件开发和交付的实际活动
软件工程和系统工程有何区别	系统工程侧重基于计算机系统开发的所有方面，包括硬件、软件和处理工程。软件工程只是它的一部分
什么是软件过程	以软件开发和进化为目的的一系列活动
什么是软件生命周期	软件生命全过程。分为可行性研究、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 8 个阶段
什么是软件过程模型	从特定角度提出的软件过程的简化表示形式

续表

问题	答案
什么是软件工程成本	软件开发成本约占 60%，测试成本占 40%。对于定制软件而言，进化成本常常高于开发成本
什么是软件工程方法	软件开发的结构化研究方法，包括系统模型、标记法、规则、设计忠告和过程指南
什么是优良软件特点	软件应具有用户所需的功能与性能，而且应该可维护、可靠、可用
软件工程面临的主要挑战是什么	要面临正在使用的旧系统、不断增长的多样性以及减少递交次数等问题的挑战

1.1.1 软件

软件包括了使计算机运行所需要的各种程序及其有关的文档资料。其中，程序是计算机任务的处理对象和处理规则的描述；文档是为了解程序所需的阐述性资料。

计算机软件主要包括系统软件与应用软件两大类。

系统软件是生成、准备和执行其他程序所需要的一组文件和程序，如操作系统（包括 DOS、Windows、UNIX 等）……

应用软件是计算机用户为了解决某些具体问题而购买、开发或研制的各种程序或软件包，包括字处理软件 Word、WPS、各种电子设备的控制系统、特定的业务处理系统等……

这两类产品的一个重要区别在于：在系统软件中，软件设计由开发者自己完成，而应用软件通常是由用户提出要求，开发者按用户要求进行开发。

1.1.2 软件危机

进入 21 世纪，软件已经使我们比以往任何时候更快、更有效地完成任务。软件支撑了在医学、农业、交通和经济和其他许多领域的改革。软件使我们能做以前不可思议的事情，如显微外科手术、多媒体教育、机器人技术等。

然而，软件并非没有问题，事实上，软件的缺陷和开发无缺陷软件的困难是科学文献和日常工作中经常讨论的问题。从 20 世纪 60 至 70 年代，“软件危机”一词在计算机界广为流传，其主要针对普遍存在的软件代价高和软件错误多的现象。

1. 软件代价高

计算机产业已被我们普遍认为是国民经济的一个重要组成部分。但很少有人会意识到计算机系统的耗费的巨大。在美国政府 1980 年的财政年度中，计算机系统方面竟耗费了大约 570 亿美元，而其中的 320 亿美元（占总数的 56%）是用于计算机软件方面的。人们逐渐开始意识到在开发一个新型计算机系统或修改一个现有系统的过程中，最大部分的资金是用在系统软件开发方面。图 1.1 表明了计算机系统硬件/软件成本变化趋势，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%；许多重要的软件开发项目，在耗费了大量的人力与财力之后，由于离预定目标甚远，宣告失败。

2. 软件产品缺陷和故障

IEEE 已经提出了描述软件产品中“错误”的标准术语，解释了缺陷、错误和故障之间的关系。

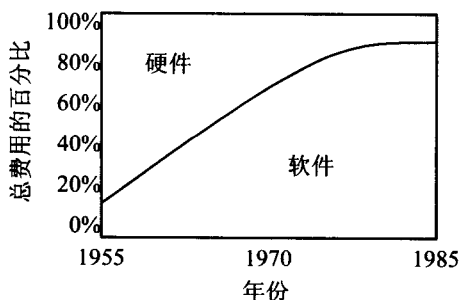


图 1.1 计算机系统硬件/软件成本变化趋势

通常，我们在软件中谈到的“错误”（Bug）有很多意思。错误可能是需求说明的错误、代码中的语法错误或引起系统崩溃的原因（该原因目前未知）。

缺陷（Fault）是人进行软件开发活动中人为出错（称为错误，Error）造成的。例如，一个设计者可能误解需求，并得出了与需求分析员和用户的实际需求不符的设计。这个设计缺陷是一种错误的代码，它能导致其他缺陷，如不正确的代码或用户手册中不正确的描述等。因此，单个错误可能产生多个缺陷，而且任何开发或维护过程中都可能存在缺陷。

故障（或称失效，Failure）是相对于系统指定行为的偏离。它可能会在系统交付前后、测试过程或在操作和维护的过程中被发现。既然需求文档可能包含错误，因此故障意味着系统即使按说明执行，也可能没有按照需要来执行。

因此，缺陷是系统的内部表现，而故障是外部表现：即用户所看到的问题。

下面来看几个故障的例子并了解出错原因。

20 世纪 80 年代早期，美国国内税收服务处委托 Sperry 公司建立一套联邦税收表格自动处理系统。根据华盛顿邮报的报道，“系统被证明不适合当前的工作量，花费几乎是预算的两倍，必须立即更换”（Sawyer1985）。1985 年，还需要再加 9 千万美元来改进 Sperry 公司最初价值 1.03 亿的设备。另外，因为出现的问题妨碍了 IRS 按时返还纳税者的税款，IRS 被迫偿付 4.02 千万美元的利息以及 2.23 千万美元的工资给加班职员。1996 年，情况并没有得到改善。洛杉矶时报在 3 月 29 日报道说，仍然没有更新 IRS 计算机系统的成熟计划，目前只有一个 6000 页的技术文档。美国国内舆论把这个项目称为“40 亿美元的彻底失败，原因是没有充分计划就错误行事”。

尽管许多软件供应商致力于设计零缺陷的软件，事实上大多数软件产品都做不到。许多软件工程师认为反弹道导弹系统至少需要 1 千万行代码；有些人估计竟高达 1 亿行。事实是，支持美国航天飞机的软件只包含 300 万行代码，包括控制发射和飞行的地面控制计算机；1985 年，航天飞机上只有 10 万行代码。因此，一个反导弹软件系统将需要数量巨大的代码测试。实际上，可靠性约束问题是无法测试的。一般地，我们说某些事情是安全攸关则其可靠性应当至少是 10^{-9} 。这意味着系统运行 10^9 小时其失效不能超过一次。要观察这个数量级的可靠性，就不得不使这个系统运行至少 10^9 小时验证它没有失效，但是 10^9 小时超过 114000 年——作为一个测试周期来说它实在是太长了。

如果不能适当地设计软件或进行编码，原本有用的技术可能会致命。例如，当 Themc-25（一种射线疗法和 X 射线机器）发生故障并使几个病人死亡时，医学界变得惊恐万状。软件设计者没有估计到会有不按标准使用几个方向键的情况；结果，当需要低剂量的射线束时，软

件却保持高剂量的设置并发出了极为集中的射线束。

西方计算机科学家把软件开发和维护过程中遇到的一系列严重问题统称为“软件危机”，其表现为：

- (1) 不能正确地估计软件开发成本和进度，致使实际开发成本往往高出预算很多。
- (2) 软件产品不可靠，满足不了用户的需求，甚至无法使用。
- (3) 交付使用的软件不易演化，以致于人们不得不重复开发类似的软件。
- (4) 软件生产率低下，远远满足不了社会发展的需求。
- (5) 软件缺乏适当的文档资料。

以上列举的仅仅是软件危机的一些明显的表现，在实际应用中与软件产品缺陷和故障有关的问题远不止这些。

3. 克服危机的途径

(1) 应该加强软件开发过程的管理，做到组织有序、各类人员协同配合，共同保证工程项目完成，避免软件开发过程中个人单干的现象。

(2) 推广使用开发软件的成功技术与方法，并且不断探索更好的技术与方法；消除一些在计算机系统早期发展阶段形成的一些错误概念和做法。在软件设计活动的整个过程中必须考虑对系统意料之外的使用。拓展你的想象力去想象系统能如何被滥用（与想象如何正确使用一样），还可以假定系统将被滥用并设计软件处理这种情况。

(3) 开发和使用好的软件工具，支持软件开发的全过程，即建立软件工程支持环境。

总之，为了解决软件危机，要从技术、管理两个方面入手，引入“软件工程”的概念，从而解决软件开发过程中的技术和管理问题。

1.2 软件工程

1968 年和 1969 年北大西洋公约组织成员国软件工作者两次召开会议（NATO 会议），讨论摆脱软件危机的办法，提出了软件工程的定义，试图建立并使用正确的工程方法开发出成本低、可靠性好并能高效运转的软件，从而解决或缓解软件危机。

1.2.1 软件工程的定义与原理

根据国际电子与电气工程师协会（IEEE）给出的定义，软件工程是：

(1) 将系统化、严格约束、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件。

(2) 将工程化应用于软件方法的研究。

在软件工程的定义中有两个关键概念：

第一，工程学科。干什么事情都离不开工程人员，他们既能恰当地应用理论、方法和工具，又能有选择地利用它们，即使在没有可用的理论和方法的情况下，也力求找出解决问题的方法。同时他们也认识到必须在行政或财政状况所允许的限度内工作，即要在此限度内寻找解决办法。

第二，软件生产的各个方面。软件工程不仅涉及软件开发的技术过程，也涉及诸如软件项目管理、支持软件生产的工具、方法和理论的开发等活动。