

教育部中国职业技术教育IT课程改革项目专用教材

软件测试专业资格认证考试用书

软件测试员 培训教材

王健 苗勇 刘郢 编著



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

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

本书可作为高等院校计算机专业及相关专业的教材，也可供从事软件测试工作的工程技术人员参考。

软件测试员 培训教材

王 勇 王 强 王 磊 编

清华大学出版社

www.tup.tsinghua.edu.cn

- 教育部中国职业技术教育 IT 课程改革项目专用教材
- 软件测试专业资格认证考试用书

软件测试员培训教材

王健 苗勇 刘郢 编著

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书主要介绍软件测试理论及相关测试技术。全书共分为三大部分：第一部分为基础概念部分，包括第1章～第5章，综合介绍了软件测试的意义、软件开发过程和开发模式、软件测试的基本原则、软件测试的相关理论，以及软件测试用例的设计等方面的内容；第二部分为软件自动化测试部分，包括第6章～第11章，主要介绍软件自动化测试的基础知识、软件自动化测试生存周期方法学，WAS, DUNIT, Rational Robot, DataFactory等自动化测试工具的使用；第三部分包括第12章～第19章，从软件工程的角度出发，着重介绍了如何规划测试工作，如何报告软件缺陷，如何评价成效，并对软件质量标准、软件质量管理和软件质量评判进行了系统的阐述。

本书可作为高职、高专、中专软件测试专业及计算机软件开发专业课程的教材，也可作为软件测试技术应用培训的基础教材，并供从事软件测试和开发行业的技术人员学习参考。

本书对于高职使用的参考课时为80课时，本科为60课时。完成本书学习后，若通过统一组织的职业技术教育IT认证课程软件测试专业资格认证考试，可获得由中国软件评测中心颁发的初级软件测试员资格证书。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有，侵权必究。

图书在版编目(CIP)数据

软件测试员培训教材/王健等编著. —北京：电子工业出版社，2003.9

ISBN 7-5053-7269-6

I. 软… II. 王… III. 软件—测试—技术培训—教材 IV. TP311.5

中国版本图书馆CIP数据核字(2003)第080564号

责任编辑：闫德明 特约编辑：龚兰方

印 刷：北京市增富印刷有限责任公司

出版发行：电子工业出版社 <http://www.phei.com.cn>

北京市海淀区万寿路173信箱 邮编 100036

经 销：各地新华书店

开 本：787×1092 1/16 印张：18.75 字数：480千字

版 次：2003年9月第1版 2003年9月第1次印刷

印 数：8000册 定价：38.00元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：(010) 68279077。质量投诉请发邮件至 zltts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

前 言

为了适应信息技术日新月异的发展变化,探索一条既满足我国社会经济发展需要,又符合职业技术教育特点的高等职业技术教育发展道路,教育部于2002年4月正式批准《中国职业技术教育IT课程改革项目》立项。该项目对于我国现有IT专业课程体系改革、实用型“IT蓝领”人才培养,以及实现人才培养层次化等方面均具有重要意义。

本书的编写就是在《中国职业技术教育IT课程改革项目》软件测试专业课程模块建设的背景下进行的,相应地还将统一组织职业技术教育IT认证课程软件测试专业资格认证考试。

软件测试、软件工程和软件质量保证的概念逐渐为人所知,在当前国家重点扶持软件产业发展的形势下,其重要性愈来愈受到重视;但另一方面,软件工程和软件质量保证中存在的问题也愈来愈成为制约软件产业快速发展的重要因素。显然,如果软件测试业不能很快地发展、成熟起来,软件产业的发展仍然会遇到很大的障碍。

软件必须经过测试。换言之,测试是验证软件是否能达到期望功能的惟一有效的方法。就像一双鞋,做出来后是否合脚,只有购买者伸脚进去试试才知道。从商品的意义上说,软件和鞋子没什么区别,但从测试的角度而言,软件测试却绝不仅仅是使用者自己使用一下就能够完成的。

目前,国内的软件测试一般有下列几种形式:一是软件公司内部进行的功能性测试,就是测试设计的功能能否完成;二是用户进行的测试,即大量的用户一起寻找使用中遇到的bug;三是第三方测试,即专业软件测试人员运用一定的测试工具对软件的质量进行检测。在软件业较发达国家,绝大多数软件产品的认定,需要第三方测试的介入,而且软件测试行业的产值几乎占了软件行业总产值的1/4。对于国内的软件业,业内人士普遍认为,软件公司的自测是很不完善的,而市场化的第三方测试则如同没长大的孩子,目前软件测试的市场几乎可以忽略不计。

随着我国软件产业的不断发展和规范化,对软件测试人员的需求必将迅速地增长。一般地,开发和测试人员之比大约为3:5,这才是一个成熟的软件产品制造过程中技术人员的合理比例。而国内公司中的软件开发和测试人员结构明显失调,目前仍然停留在开发人员自行测试的阶段,存在着很难发现软件潜在错误的弊端。因为沿用软件开发人员的思路去进行测试,是无论如何也挑不出大毛病的。

未雨绸缪,我们此时编写这样一本软件测试专业培训教材,就是为了能够在我国职业技术教育改革过程中尽绵薄之力,为未来软件产业的快速发展储备大量的具备相应软件测试理论和应用基础的技术人才。本书共分19章,包括三大部分,第一部分介绍有关软件测试的基本理论和基础概念,第二部分重点介绍软件自动化测试理论和测试工具的使用,第三部分则讲述软件质量标准、保证和管理方面的内容。编写思路力求紧紧围绕理论联系实际这一中心,

特别重视实际应用能力的培养。

在本书的编写过程中,得到了中国软件评测中心大量的技术支撑和资源支持,在此表示衷心的感谢。清华大学的侯炳辉教授和吕文超教授审阅了本书的主要内容,此外,叶宜强、杨丽娟等在书稿的校核、整理和资料分析、管理方面做了大量的工作,在此向以上诸位一并致以特别的谢意。

本书虽经反复推敲,但由于时间仓促,水平所限,书中的不足之处敬请广大读者谅解,并欢迎提出批评指正意见。

编 者

目 录

第 1 章 软件测试的意义	(1)
1.1 软件缺陷带来影响的例子	(2)
1.1.1 游戏软件 bug	(2)
1.1.2 火星登陆事故	(2)
1.1.3 英特尔奔腾芯片缺陷	(3)
1.1.4 千年虫问题	(3)
1.1.5 Windows 2000 中文版输入法漏洞	(4)
1.1.6 金山词霸 bug	(4)
1.2 软件缺陷	(5)
1.3 软件缺陷产生的原因	(6)
1.4 修复软件缺陷的代价	(7)
1.5 软件测试员的工作目标	(7)
1.6 软件测试员的必备素质	(8)
1.7 软件测试职业	(9)
1.8 小结	(11)
第 2 章 软件开发过程简介	(13)
2.1 软件产品的组成	(14)
2.1.1 各种各样的开发投入	(14)
2.1.2 客户需求	(14)
2.1.3 产品说明	(15)
2.1.4 软件设计文档	(15)
2.1.5 软件测试文档	(16)
2.1.6 开发进度表	(16)
2.1.7 软件产品的组成部分	(16)
2.2 软件开发项目组成员	(17)
2.3 软件开发模式	(18)
2.3.1 大棒模式	(18)
2.3.2 边写边改模式	(18)
2.3.3 瀑布模式	(19)
2.3.4 快速原型模式	(21)
2.3.5 螺旋模式	(21)
2.4 小结	(22)
第 3 章 软件测试的原则	(23)

3.1 软件测试的基本原则	(24)
3.1.1 不可能对程序进行完全测试	(24)
3.1.2 无法显示潜伏的软件缺陷	(25)
3.1.3 软件测试的风险	(25)
3.1.4 软件缺陷越找越多	(26)
3.1.5 难以描述的软件缺陷	(26)
3.1.6 产品说明书不断变化	(27)
3.1.7 杀虫剂现象	(27)
3.1.8 软件缺陷不都能得到修复	(27)
3.1.9 软件测试员的作用	(28)
3.1.10 软件测试的条理性	(28)
3.2 有关软件测试的专用术语	(29)
3.2.1 精确与准确	(29)
3.2.2 验证检查与合法性检查	(29)
3.2.3 质量与可靠性	(30)
3.2.4 质量评判 (QA) 与测试	(30)
3.3 小结	(31)
第4章 软件测试的相关理论	(33)
4.1 软件测试的基本概念	(34)
4.1.1 有关软件测试的基础知识	(34)
4.1.2 静态测试方法和动态测试方法	(36)
4.1.3 黑盒测试方法和白盒测试方法	(37)
4.1.4 软件测试过程	(39)
4.2 单元测试	(40)
4.2.1 单元测试的主要任务	(40)
4.2.2 单元测试的执行过程	(42)
4.3 集成测试	(42)
4.3.1 非增式测试方法	(42)
4.3.2 增式测试方法	(43)
4.3.3 不同集成测试方法的比较	(45)
4.4 确认测试	(45)
4.4.1 确认测试的准则	(45)
4.4.2 配置审查的内容	(46)
4.5 系统测试	(46)
4.5.1 恢复测试方法	(46)
4.5.2 安全测试方法	(46)
4.5.3 强度测试方法	(46)
4.5.4 性能测试方法	(47)
4.6 验收测试	(47)

4.6.1	验收测试的内容	(47)
4.6.2	软件配置和文档资料	(48)
4.7	调试过程	(48)
4.8	面向对象的软件测试	(48)
4.8.1	面向对象的单元测试	(49)
4.8.2	面向对象的集成测试	(49)
4.8.3	面向对象的确认测试	(49)
4.9	小结	(49)
第5章	软件测试用例设计	(51)
5.1	黑盒测试方法	(52)
5.1.1	边界值分析法	(52)
5.1.2	等价类划分方法	(53)
5.1.3	因果图方法	(54)
5.1.4	比较测试法	(54)
5.1.5	测试方法的选择	(54)
5.2	白盒测试方法	(55)
5.2.1	基本概念	(55)
5.2.2	逻辑覆盖测试方法	(58)
5.2.3	路径测试方法	(64)
5.3	特定环境及应用的测试	(67)
5.3.1	客户/服务器体系结构的测试	(67)
5.3.2	GUI 的测试	(67)
5.3.3	实时系统的测试	(68)
5.4	小结	(69)
第6章	软件自动化测试	(71)
6.1	软件自动化测试基础	(72)
6.1.1	自动化测试的产生	(72)
6.1.2	自动化测试的定义	(72)
6.1.3	自动化测试的引入	(72)
6.1.4	自动化工具的作用及优势	(73)
6.1.5	测试自动化的几个例子	(73)
6.2	软件自动化测试生存周期方法学	(75)
6.2.1	决定采用自动化测试	(77)
6.2.2	测试工具的采购	(78)
6.2.3	自动化测试的引入阶段	(78)
6.2.4	测试计划、设计和开发	(78)
6.2.5	测试执行与管理	(79)
6.2.6	测试活动评审与评估	(79)

6.3 自动化测试生存周期方法学的应用	(79)
6.4 自动化测试决定	(81)
6.4.1 建立正确的自动化测试目标	(82)
6.4.2 应用自动化测试的优点	(85)
6.5 小结	(91)
第7章 WEB 服务器负载测试软件简介	(93)
7.1 Web ApplicationStress Tool 概述	(94)
7.2 Web Application Stress Tool 的安装	(94)
7.2.1 系统要求	(94)
7.2.2 如何获取	(94)
7.2.3 如何安装	(95)
7.3 Web Application Stress Tool 的设置及基本操作	(95)
7.3.1 主界面窗口	(95)
7.3.2 制作脚本	(96)
7.4 小结	(102)
第8章 单元测试工具软件简介	(105)
8.1 Dunit 概述	(106)
8.2 Dunit 的配置	(106)
8.2.1 系统要求	(106)
8.2.2 如何获取	(106)
8.2.3 文件内容	(106)
8.2.4 配置方法	(107)
8.3 Dunit 的使用介绍	(108)
8.3.1 创建一个被测试的 Project	(108)
8.3.2 建立一个测试方案	(108)
8.3.3 查看测试结果	(111)
8.3.4 测试组 (TestSuite) 的使用	(112)
8.3.5 在控制台模式下使用测试个案	(113)
8.4 Dunit 附加功能的使用	(114)
8.4.1 重复运行测试案例	(114)
8.4.2 使用 TTestSetup 类	(115)
8.4.3 测试 Exception 的说明	(116)
8.5 Dunit 小结	(116)
第9章 RATIONAL ROBOT 软件的使用	(117)
9.1 Rational Robot 概述	(118)
9.1.1 概述	(118)
9.1.2 基本概念	(119)
9.2 Rational Robot 的安装	(119)

9.2.1	系统要求	(121)
9.2.2	如何获取	(121)
9.3	Rational Robot 产生用于功能测试的 GUI 脚本	(121)
9.3.1	设置以及预定义	(121)
9.3.2	记录 GUI 脚本	(126)
9.3.3	在 GUI Script 中加入特写	(132)
9.3.4	使用查证点	(137)
9.3.5	使用 Datapool	(137)
9.3.6	编辑 GUI 脚本	(140)
9.3.7	编译 GUI 脚本	(142)
9.3.8	调试 GUI 脚本	(143)
9.3.9	回放 GUI 脚本	(144)
9.3.10	工具条操作	(145)
9.4	Rational Robot 产生用于性能测试的 VU	(146)
9.4.1	设置以及预定义	(146)
9.4.2	记录 VU 脚本	(147)
9.4.3	回放 VU 脚本	(149)
9.4.4	重录 VU 脚本	(149)
9.4.5	复制 VU 脚本	(150)
9.4.6	删除 VU 脚本	(150)
9.4.7	编译 VU 脚本	(150)
9.4.8	查询会话中的脚本列表	(150)
9.4.9	用会话生成脚本	(151)
9.4.10	将 VU 脚本融入会话	(151)
9.4.11	手工 VU 脚本编码	(151)
9.5	删除 VB 脚本	(155)
9.6	定制 SQABASIC 脚本	(156)
9.6.1	定制 SQABasic 脚本	(156)
9.6.2	库文件 (Library Source Files)	(156)
9.6.3	创建、编辑.sbl 库文件	(156)
9.6.4	创建、编辑.rec 库文件	(157)
9.6.5	在 Global.sbl 中加入子程序	(158)
9.6.6	使用库文件	(158)
9.6.7	SQABasic 头文件	(158)
9.6.8	创建、编辑仓库 (Repository) 范围的头文件	(158)
9.6.9	创建、编辑项目头文件	(159)
9.6.10	使用头文件	(159)
9.6.11	使用模板文件	(159)
9.7	测试应用程序举例	(160)

9.7.1 测试 Delphi 应用程序	(160)
9.7.2 测试 Visual Basic 应用程序	(161)
9.8 小结	(162)
第 10 章 DataFactory 软件的使用	(165)
10.1 DataFactory 概述	(166)
10.2 DataFactory 的安装	(166)
10.2.1 系统要求	(166)
10.2.2 如何获取	(166)
10.2.3 如何安装	(167)
10.3 DataFactory 的基本操作	(167)
10.3.1 主界面窗口	(167)
10.3.2 新建项目	(167)
10.3.3 数据库的链接	(167)
10.3.4 数据的产生方式	(170)
10.4 小结	(176)
第 11 章 其他测试工具简介	(177)
11.1 QACenter	(178)
11.2 WinRunner	(179)
11.3 LoadRunner	(181)
11.4 PureLoad	(181)
11.5 Open System Testing Architecture (OpenSTA)	(182)
第 12 章 规划测试工作	(183)
12.1 建立测试计划的目标	(184)
12.2 测试计划的主要内容	(184)
12.2.1 测试计划的高级期望	(185)
12.2.2 产品基本情况调研	(185)
12.2.3 测试策略	(186)
12.2.4 资源需求	(186)
12.2.5 问题跟踪报告	(187)
12.2.6 测试计划中的人员、地点和事件	(187)
12.2.7 术语和定义的统一	(187)
12.2.8 明确团队责任	(188)
12.2.9 确定测试内容	(189)
12.2.10 测试阶段	(190)
12.2.11 测试员的任务分配	(190)
12.2.12 测试进度控制	(191)
12.2.13 频度和统计手段	(192)
12.2.14 风险和问题	(192)

12.3 小结	(192)
第 13 章 测试案例的编写和跟踪	(195)
13.1 计划测试案例的原因	(196)
13.2 测试案例计划一览	(196)
13.2.1 有关测试设计说明	(197)
13.2.2 有关测试案例说明	(198)
13.2.3 有关测试过程说明	(200)
13.2.4 测试计划的细节	(201)
13.3 组织和跟踪测试案例	(201)
13.4 小结	(202)
第 14 章 报告软件缺陷	(203)
14.1 如何面对软件缺陷	(204)
14.2 软件缺陷的分离和再现	(207)
14.3 软件缺陷的严重性和优先级	(208)
14.4 软件缺陷的生命周期	(209)
14.5 软件缺陷跟踪系统	(211)
14.5.1 测试事件报告	(211)
14.5.2 手工软件缺陷报告和跟踪	(211)
14.6 小结	(213)
第 15 章 评价成效	(215)
15.1 软件缺陷跟踪数据库	(216)
15.2 测试使用频度	(217)
15.3 常用项目级频度	(220)
15.4 小结	(223)
第 16 章 软件质量标准简介	(225)
16.1 能力成熟度模型简介	(226)
16.1.1 CMM 基本的过程概念	(226)
16.1.2 CMM 的 5 级体系结构	(227)
16.2 ISO 9000 标准简介	(228)
16.2.1 ISO 9000 标准的构成	(229)
16.2.2 ISO 9000 标准的要求	(229)
16.2.3 ISO 9000 质量管理的 8 项原则	(230)
16.3 小结	(231)
第 17 章 软件质量管理	(233)
17.1 质量与质量管理的概念	(234)
17.1.1 什么是质量	(234)
17.1.2 质量管理的基本概念	(234)

17.2 软件质量管理的内容	(238)
17.2.1 软件质量策划的内容	(238)
17.2.2 软件质量控制与质量保证	(240)
17.2.3 对软件质量进行的度量和验证	(241)
17.2.4 软件质量改进的活动内容	(242)
17.3 小结	(243)
第 18 章 实践软件质量管理	(245)
18.1 ISO 9000 和 CMM 对软件组织的适应性	(246)
18.1.1 对 ISO 9000 的误解	(246)
18.1.2 ISO 9000 和 CMM 的联系	(247)
18.2 建立质量管理体系的方法	(247)
18.2.1 对顾客需求和期望进行确定	(248)
18.2.2 质量方针和质量目标的建立	(248)
18.2.3 实现目标的过程和职责	(249)
18.2.4 质量管理体系文件的形成	(250)
18.3 采用 ISO 9000 和 CMM 的软件质量管理	(251)
18.3.1 软件项目策划	(251)
18.3.2 软件项目跟踪与监督	(253)
18.3.3 软件需求的管理	(256)
18.3.4 软件配置管理	(260)
18.3.5 软件质量保证	(264)
18.3.6 软件子合同管理	(267)
18.3.7 同行评审	(270)
18.3.8 缺陷预防	(274)
18.3.9 内部质量审核	(276)
18.3.10 纠正措施	(278)
18.4 小结	(279)
第 19 章 软件质量的评判	(281)
19.1 质量是免费的	(282)
19.2 软件测试和质量评判	(282)
19.2.1 软件测试员	(283)
19.2.2 软件质量评判人员	(283)
19.2.3 软件测试团队的其他名称	(284)
19.2.4 软件测试团队的管理和组织结构	(284)
19.3 小结	(286)
参考文献	(287)

第 1 章 软件测试的意义

本章要点

- 充分认识软件缺陷对人类生活的影响
- 软件缺陷的定义
- 导致软件缺陷出现的原因
- 软件测试的意义和目标

软件测试是由于软件存在缺陷而产生的，那么，最早的软件缺陷是什么时候出现的呢？可以说早在 20 世纪 40 年代，当 Mark II 型等电子管计算机诞生的时候，软件缺陷就随之出现了。此后随着计算机的不断更新换代，硬件性能的飞速提高，个人计算机应用的迅速普及，软件缺陷也越来越成为一个令人关注的问题。

1.1 软件缺陷带来影响的例子

通常情况下，人们并没有真正意识到软件已经渗透到了自己的日常生活之中，比如电子产品包装盒中附带的软件光盘、年轻人所热衷的视频游戏软件等，比比皆是。以前称为新奇的小玩意儿，如寻呼机和手机，现在都已经变得非常普通。而对于一个在现代化企业工作的职员来说，如果一天不登录因特网查看电子邮件，已经是一件不可想像的事情。事实上，我们已经离不开各种各样的信息服务，为了提供这些快捷服务，除了需要基础硬件之外，还需要各种各样、功能各异的计算机软件。

可以这样讲，软件无处不在。然而，软件是由人编写的，因此软件不可能十全十美，难免存在各种缺陷。

1.1.1 游戏软件 bug

1994 年秋天，迪斯尼公司发布了首张面向儿童的多媒体光盘游戏“狮子王动画故事书”（Lion King Animated Storybook）。这是迪斯尼公司首次进军游戏市场，他们进行了大力的宣传促销。结果，销售额非常可观。该游戏成为全美儿童当年夏季的必买游戏。但是，后来却产生了迪斯尼公司意想不到的后果。12 月 26 日，刚刚过完圣诞节，迪斯尼公司客户支持部的电话就开始响个不停。很快，售后服务支持部门就淹没在愤怒的家长和哭诉玩不成游戏的孩子们的电话狂潮之中。

后来得到的证实表明，迪斯尼公司没有对投入市场上使用的各种 PC 机型进行正确的测试。软件在少数系统中工作正常——例如迪斯尼的程序员用于开发游戏的系统，但在大众使用的常见系统中却存在问题。

1.1.2 火星登陆事故

1999 年 12 月 3 日，美国航天局的火星极地登陆飞船在试图登陆火星表面时失踪了。错误修正委员会观测到故障，并确定出现误动作的原因极可能是由于某一个数据位被意外地更改。大家都对这一错误感到非常震惊，认为该问题应该在内部测试时予以解决。

简单来讲，火星登陆计划的过程是这样的：当飞船降落到火星表面时，它将打开降落伞减缓飞船的下落速度。降落伞打开后的几秒钟内，飞船的三条腿将迅速撑开，并在预定地点着陆。当飞船离火星表面 1800m 时，它将丢弃降落伞，点燃登陆推进器，在余下的高度缓缓降落到火星表面。

但是，美国航天局为了省钱，简化了确定何时关闭推进器的装置。为了替代其他太空船上使用的贵重雷达，他们在飞船的脚上装了一个廉价的触点开关，在计算机中设置一个数据位来关掉燃料。很简单，飞船的脚不着地，引擎就会着火。

遗憾的是, 错误修正委员会在测试中发现, 当飞船的脚迅速撑开准备着陆时, 机械震动很容易触发着地开关, 设置错误的数据位。设想飞船开始着陆时, 计算机极有可能关闭推进器, 而火星登陆飞船下坠 1800m 之后便冲向火星表面, 撞成碎片。

这一事故产生的后果非常严重, 然而其幕后的原因却如此简单。在登陆开始之前, 飞船经过了多个小组测试。其中一个小组测试飞船的脚落地过程, 另一个小组测试此后的着陆过程。前一个小组并不去注意着地数据位是否置位, 因为这不是他们负责的范围; 后一个小组总是在开始测试之前重置计算机、清除数据位。双方独立工作都很好, 但从未组合在一起进行过集成测试, 从而导致了这一严重事故的发生。

1.1.3 英特尔奔腾芯片缺陷

对于以下算式:

$$(4195835/3145727) \times 3145727 - 4195835$$

如果你的计算机的“计算器”程序的答案是 0, 就说明你的计算机没问题。如果得出别的结果, 就表示计算机使用的是带有浮点除法软件缺陷的老式英特尔奔腾 CPU——这个软件缺陷被刻录在一个计算机芯片中, 并在生产过程中反复制造。

1994 年 12 月 30 日, 美国弗吉利亚州 Lynchburg 大学的 Thomas R. Nicely 博士在他的奔腾 PC 机上做除法实验时记录了一个没想到的结果。他把发现的问题放到因特网上, 随后引发了一场风暴, 成千上万的人发现了同样的问题, 以及其他得出错误结果的情形。万幸的是, 这种情况很少见, 仅仅在进行精度要求很高的数学、科学和工程计算中才导致错误。大多数进行财会管理和商务应用的用户不会遇到此类问题。

事实上, 这个案例中重要的不是软件缺陷, 而是英特尔解决问题的方式:

- 1) 英特尔的软件测试工程师在芯片发布之前进行内部测试时已经发现了这个问题, 但是英特尔的管理层认为这没有严重到要保证修正, 甚至公开的地步。
- 2) 当软件缺陷被发现时, 英特尔通过新闻发布会和公开声明试图掩饰这个问题的严重性。
- 3) 当开始受到巨大的舆论压力时, 英特尔承诺更换有问题的芯片, 但要求用户必须证明自己受到软件缺陷的影响。

舆论因此大哗。因特网新闻组充斥着愤怒的客户要求英特尔解决问题的呼声。英特尔被新闻舆论称为毫无信誉的公司。最后, 英特尔公司为自己处理软件缺陷的行为道歉, 并拿出 4 亿多美元来支付更换坏芯片的费用。从此之后, 英特尔在 Web 站点上报告已发现的问题, 并认真查看客户在因特网新闻组上的反馈意见。

1.1.4 千年虫问题

千年虫问题是一个非常著名的计算机软件缺陷。

20 世纪 70 年代, 一位负责公司工资系统的程序员所使用的计算机存储空间很小, 迫使他尽量节省每一个字节。他用尽了一切手段将自己的程序压缩到最小, 其中之一就是把表示年份的 4 位数, 例如 1975, 缩减为 2 位数, 即 75。因为工资系统极度依赖数据处理, 因此他得以节省可观的存储空间。他认为只有在到达 2000 年时, 程序计算 00 或 01 这样的年份时才会出现问题, 但是在 25 年之内他使用的程序肯定会更改或升级, 而且眼前的任务比现在计划