21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

软件测试

Software Testing

朱少民 编著

- 内容与软件测试行业实际需求相吻合
- 理论和实践有机结合
- 简单易懂、循序渐进、案例丰富

精品系列

人民邮电出版社
POSTS & TELECOM PRESS

图书在版编目（CIP）数据

软件测试 / 朱少民编著. —北京：人民邮电出版社，2009.8
21世纪高等学校计算机规划教材
ISBN 978-7-115-20609-1

Ⅰ. 软… Ⅱ. 朱… Ⅲ. 软件—测试—高等学校—教材
Ⅳ. TP311.5

中国版本图书馆CIP数据核字（2009）第053290号

内 容 简 介

软件测试是一门新兴的学科，同时，又是一门越来越重要的学科。本书首先从软件测试的产生和定义出发，描述了一个完整的软件测试知识体系轮廓，让读者从全局来把握软件测试；然后，针对软件测试不同的知识点展开讨论。

本书在内容组织上力求创新，尽量使软件测试知识具有很好的衔接性和系统性，使需求和设计评审、软件测试用例设计、自动化测试和各个阶段的实际测试活动有机地结合起来，使读者更容易领会如何将测试的方法和技术应用到单元测试、功能测试、系统测试和本地化测试中去。本书提供了丰富的实例和实践要点，更好地体现软件测试学科的特点，使读者掌握测试方法的应用之道和品味测试的极佳实践。

本书条理清晰、语言流畅、通俗易懂，内容丰富、实用，理论和实践能有效地结合。本书可作为高等学校的软件工程专业、计算机软件专业和相关专业的教材，也可作为软件测试工程师以及其他各类软件工程技术人员的参考书。

21 世纪高等学校计算机规划教材

软 件 测 试

◆ 编　　著　朱少民
责任编辑　刘　博

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
中国铁道出版社印刷厂印刷

◆ 开本：787×1092　1/16
印张：15.75
字数：410 千字　　2009 年 8 月第 1 版
印数：1－3 000 册　　2009 年 8 月北京第 1 次印刷

ISBN 978-7-115-20609-1/TP

定价：28.00 元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

出版者的话

计算机应用能力已经成为社会各行业最重要的工作要求之一，而计算机教材质量的好坏会直接影响人才素质的培养。目前，计算机教材出版市场百花争艳，品种急剧增多，要从林林总总的教材中挑选一本适合课程设置要求、满足教学实际需要的教材，难度越来越大。

人民邮电出版社作为一家以计算机、通信、电子信息类图书与教材出版为主的科技教育类出版社，在计算机教材领域已经出版了多套计算机系列教材。在各套系列教材中涌现出了一批被广大一线授课教师选用、深受广大师生好评的优秀教材。老师们希望我社能有更多的优秀教材集中地呈现在老师和读者面前，为此我社组织了这套“21世纪高等学校计算机规划教材——精品系列”。

“21世纪高等学校计算机规划教材——精品系列”具有下列特点。

（1）前期调研充分，适合实际教学需要。本套教材主要面向普通本科院校的学生编写，在内容深度、系统结构、案例选择、编写方法等方面进行了深入细致的调研，目的是在教材编写之前充分了解实际教学的需要。

（2）编写目标明确，读者对象针对性强。每一本教材在编写之前都明确了该教材的读者对象和适用范围，即明确面向的读者是计算机专业、非计算机理工类专业还是文科类专业的学生，尽量符合目前普通高等教学计算机课程的教学计划、教学大纲以及发展趋势。

（3）精选作者，保证质量。本套教材的作者，既有来自院校的一线授课老师，也有来自IT企业、科研机构等单位的资深技术人员。通过他们的合作使老师丰富的实际教学经验与技术人员丰富的实践工程经验相融合，为广大师生编写出适合目前教学实际需求、满足学校新时期人才培养模式的高质量教材。

（4）一纲多本，适应面宽。在本套教材中，我们根据目前教学的实际情况，做到“一纲多本”，即根据院校已学课程和后续课程的不同开设情况，为同一科目提供不同类型的教材。

（5）突出能力培养，适应人才市场要求。本套教材贴近市场对于计算机人才的能力要求，注重理论技术与实际应用的结合，注重实际操作和实践动手能力的培养，为学生快速适应企业实际需求做好准备。

（6）配套服务完善，共促提高。对于每一本教材，我们在教材出版的同时，都将提供完备的PPT课件，并根据需要提供书中的源程序代码、习题答案、教学大纲等内容，部分教材还将在作者的配合下，提供疑难解答、教学交流等服务。

在本套教材的策划组织过程中，我们获得了来自清华大学、北京大学、人民大学、浙江大学、吉林大学、武汉大学、哈尔滨工业大学、东南大学、四川大学、上海交通大学、西安交通大学、电子科技大学、西安电子科技大学、北京邮电大学、北京林业大学等院校老师的大力支持和帮助，同时获得了来自信息产业部电信研究院、联想、华为、中兴、同方、爱立信、摩托罗拉等企业和科研单位的领导和技术人员的积极配合。在此，人民邮电出版社向他们表示衷心的感谢。

我们相信，“21世纪高等学校计算机规划教材——精品系列”一定能够为我国高等院校计算机课程教学做出应有的贡献。同时，对于工作欠缺和不妥之处，欢迎老师和读者提出宝贵的意见和建议。

前　言

在过去半个世纪，软件获得了空前的发展，逐渐渗透到各个领域，从最早的科学计算、文字处理、数据库管理、银行业务处理，到工业自动控制和生产、办公自动化、新闻媒体、通信、汽车、消费电子、娱乐等，软件无处不在，改变了人类的生活与生产方式。随着计算机软件在各行各业的普及应用，人们对软件质量的要求也越来越高，软件的专业化和多样化特点越来越显著。但同时，我们看到软件产业目前还不够成熟，软件质量的现状不容乐观，软件在运行和使用过程中出现的问题还比较多。例如，2008 年互联网 Web 发展十大失败事件中，90%都是由质量问题造成的，与“宕机”、“停机”、“崩溃”等一系列严重的质量问题联系在一起。

软件质量一直是软件工程中的一个焦点，成为人们几十年来不断研究、探索的领域。为了改善软件质量，人们不仅从企业文化、软件过程模型、需求工程、设计模式等不同方面来获取有效的方法和实践，而且开始重视软件测试，在软件测试上有更多的考虑和投入。虽然质量是内建的，但软件测试依旧承担着非常重要的作用。软件测试自身也在发生变化，已经不再只充当门卫——仅在软件发布之前进行检验，而是正在形成一个持续的反馈机制，贯穿软件开发的整个过程，以便尽早地发现问题，降低开发成本，提高软件开发生产力。软件测试人员不再是软件开发的辅助人员，而是软件开发团队的主体之一、积极的参与者。从项目开始的第一天，测试人员就参与项目需求和设计的讨论、评审等各种活动，尽早发现软件需求定义和设计实现上的问题，及时发现软件项目中存在的质量风险。软件开发团队必须尽可能地在交付产品之前控制未来的质量风险，这就必然需要依赖于卓有成效的软件测试。将传统的程序测试的狭义概念扩展到今日业界逐渐认可的、广义的软件测试概念，测试涵盖了需求验证（评审）、设计验证（评审）等活动。软件测试贯穿整个软件生命周期，从需求评审、设计评审开始，就介入到软件产品的开发活动或软件项目的实施中，和其他开发团队相互协作、相互补充，共同构成软件生命周期中的有机整体。

软件测试不是一项简单的工作，它远比人们所直观想象的要复杂。高效、高质量地完成一个软件系统的测试，涉及的因素很多，也会碰到各种各样的问题，并且要在测试效率和测试风险之间找到最佳平衡点和有效的测试策略，这些都需要测试人员一一克服。要做好软件测试，测试人员不仅需要站在客户的角度思考问题，真正理解客户的需求，具备良好的分析能力和创造性的思维能力，完成功能测试和用户界面的测试，而且要能理解软件系统的实现机理和各种使用场景，具备扎实的技术功底，能通过测试工具完成相应的性能测试、安全性测试、兼容性测试和可靠性测试等更具挑战性的任务。软件测试的主要目的是发现软件中的缺陷；坚持“质量第一”的原则，就会在实际操作中遇到一些阻力，这都需要测试人员去克服。从这些角度看，相对设计、编程人员，对一个优秀的测试工程师的要求要高得多，不仅要体现高超的技术能力，如系统平台设置、架构设计分析、编程等方面的能力，而且要展示自己的业务分析能力、对客户需求的理解能力和团队沟通协作的能力。

本书在内容上进行了精心组织，从概念到方法，再从方法到实践，满足知识层次和逻辑关系的要求，使课程教学和学习更加自然和有效。例如，先介绍软件测试用例的基本内容和基本方法，然后结合单元测试、功能测试和系统测试来介绍测试用例的具体设计方法，包括白盒方法和黑盒方法。再比如，先介绍自动化测试的特点、原理和方法，然后在后面各章结合实际测试需要，引入所需要的测试工具，使方法、工具和测试实践有机地结合起来，使读者更容易理解和掌握工具的使用。

本书全面介绍了软件测试的先进理念和知识体系，演绎了使用软件测试的方法、技术和工具，帮助大家尽快成为优秀的测试工程师。在第1章回答了一系列关于软件测试的基本问题，如：为什么要进行软件测试？软件测试是什么？软件测试学科是如何发展而来的？软件测试带给我们什么？

第2～10章逐步地深入到软件测试领域的各个知识点，包括需求和设计的评审、测试用例设计、自动化测试、单元测试、功能测试、系统测试、缺陷报告、测试计划和管理等。

第2章，不仅介绍了各种评审方法，而且从测试人员角度出发，讨论了如何理解需求和设计实现、如何对需求和设计进行评审，包括清晰地描述了需求评审和各类设计评审的标准。

第3章，从一个简单的实例引入测试用例，阐述了为什么需要测试用例以及如何从书写、基本设计方法和评审等各个方面保证测试用例的质量，并说明如何组织、使用和维护测试用例。

第4章，也是从一些有趣的、简单的实验和一个实实在在的自动化测试的例子引入软件测试自动化，然后介绍自动化测试的概念、原理、特点和优势，包括代码分析、对象识别和脚本技术。最后，讨论了如何引入自动化测试以及选择测试工具。

第5章，重点介绍了单元测试中的白盒测试方法和代码评审，包括分支覆盖、条件覆盖、基本路径、组合覆盖等方法和代码缺陷检查表等。而且，还讨论了集成测试的策略和方法，以及单元测试工具，包括JUnit和微软VSTS等。

第6章，重点介绍了功能测试用例的各种设计方法，包括等价类划分、边界值分析、因果图、决策表、功能图和正交试验等方法，并讨论了可用性测试、功能测试执行策略和实践、功能测试工具及其使用。

第7章，介绍了国际化和本地化的概念及其关系，详细展示了国际化测试和本地化测试的要求、方法、工具和具体的技巧，包括功能、数据格式、UI、配置、兼容性和翻译等方面的验证。

第8章，内容丰富，详细讨论了负载测试、压力测试和性能测试等概念及其联系，重点讨论了负载测试和性能测试的方法、技术和工具，包括输入/输出参数、场景设置、结果分析等。然后，介绍了兼容性测试、安全性测试、容错性测试和可靠性测试等。

第9章，描述了缺陷报告的内容、格式和各种属性，如类型、来源、严重性和优先级等，重点阐述了缺陷生命周期、如何有效地报告和处理缺陷、各种缺陷分析方法及其作用。

第10章，强调了测试原则，详细介绍了软件测试计划的过程，包括如何制定测试的目标和策略、如何分析测试范围和预估测试的工作量、如何完成资源安排和进度管理等。最后，讨论了测试风险、测试报告和测试管理工具等。

本书特别重视理论与实践相结合，使读者既能领会软件测试的思想和方法，又能将这些方法和技术应用到实际工作中去。因此，本书既适合作为高等院校计算机应用、计算机软件、软件工程、软件测试等专业的教材；也适合从事软件开发和维护的工程技术人员阅读，包括软件测试人员、开发人员、项目经理和产品经理。

由于作者水平有限，书中难免会存在一些不当和错误之处，恳请读者批评指正。

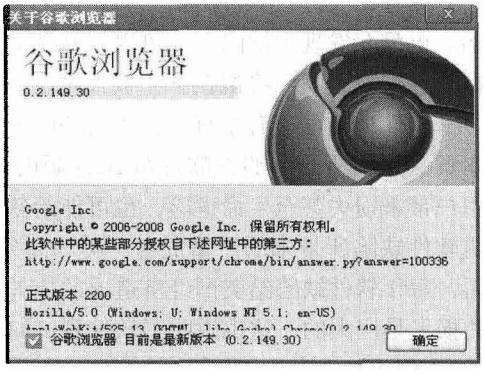
作者

2009年1月

目　录

第 1 章 软件测试概述

每当人们使用一个刚发布的新软件时，总是比较容易发现问题，例如，谷歌浏览器 Chrome 在 2008 年 9 月 3 日发布后，被发现了不少问题，除了“缓冲区溢出”这样严重的缺陷之外，还有其他一些问题，涉及功能、硬件兼容、网络传输、插件等多个方面问题，例如：



- 当 Chrome 运行时，笔记本电脑有时无法进入休眠状态；
- Chrome 不接受 Windows 互联网选项中“对这些网址绕过代理服务器”的列表；
- Chrome 不支持 SSL 身份验证；
- 在插件（如视频或 flash）比较多的网页上，滚动条响应迟钝；
- 当打开 RSS feed 或 XML 文件时，Chrome 只显示 XML 原始数据；
- 在某些网站上 Chrome 的固定宽度字体显得太小；
- 图片有时不能显示；
- 鼠标滚轮只能向下滚不能向上滚；
- Chrome 几乎打开每一个网页都需刷新一遍。

从这里可以看出，软件产品越来越复杂，会存在各种各样的问题，而这需要通过软件测试来发现这些问题，并促使这些问题得到修正，从而使发布的软件能够满足质量要求，受到用户的喜欢。软件测试是软件生命周期中非常重要的活动之一，它要完成对软件设计和构建的验证，并与软件开发相互呼应，构成对立而统一的软件工程的有机整体。

1.1 一个真实的故事

这是一个真实的故事，故事发生在 1945 年 9 月 9 日，一个炎热的下午。当时的机房是一间第一次世界大战时建造的老建筑，没有空调，所有窗户都敞开着。Grace Hopper（第一个程序语言编译器的开发人员，后来成为海军少将，正领导着一个研究小组夜以继日地工作，研制一台称为“MARK II”的计算机，它使用了大量的继电器（电子机械装置，那时还没有使用晶体管），一台并不纯粹的电子计算机。突然，MARK II 死机了。研究人员试了很多次还是启动不了，然后就开

始用各种方法找问题，看问题究竟出现在哪里，最后定位到板子 F 第 70 号继电器出错。Hopper 观察这个出错的继电器，惊奇地发现一只飞蛾躺在中间，已经被继电器打死。她小心地用摄子将蛾子夹出来，用透明胶布贴到“事件记录本”中，并注明“第一个发现虫子的实例”，然后计算机又恢复了正常。从此以后，人们将计算机中出现的任何错误戏称为臭虫（Bug），而把找寻错误的工作称为“找臭虫”（Debug）。

这个故事告诉我们，在软件运行之前，要将计算机系统中可能存在的问题找出来，否则计算机系统可能会在某个时刻不能工作，造成更大的危害。从这个故事，我们也知道软件缺陷为什么被称为“bug”的原因，而且知道在什么时候，第一个“Bug”被我们发现了。

1.2 为什么要进行软件测试

为什么要进行软件测试？就是因为软件缺陷的存在。因为只有通过测试，才可以发现软件缺陷。也只有发现了缺陷，才可以将软件缺陷从软件产品或软件系统中清理出去。至于为什么会存在软件缺陷，我们将留在下一节来讨论。

软件缺陷危害有小有大，小的缺陷可能使软件看起来不美观、使用起来不流畅或不方便，如本章一开始所介绍的谷歌发布 Beta 版的浏览器中存在各种各样的问题。而严重的缺陷则可能给用户带来损失甚至生命危险，也可能给软件企业自身带来巨大损失。下面就有好几个例子可以说明软件缺陷所带来的严重危害。美国商务部国家标准和技术研究所（NIST）进行的一项研究表明，每年软件缺陷给美国经济造成的损失高达 595 亿美元；这说明软件中存在的缺陷给我们带来的损失是巨大的，这也说明了软件测试的必要性和重要性。

【例 1】

随着苹果公司推出它万众期待的 iPhone 3G，同时也推出了一个同步服务器 MobileMe。MobileMe 允许 Mac 和 PC 用户通过一个 Web 界面去同步他们的联系人、日历、电子邮件、照片等内容。但在它推出的第一天便充满了大量的问题——性能缓慢、宕机、用户随机注销等，还有一个致命的问题——整整一天，同步服务无法同步日历和全部联系人。就像苹果公司 CEO Steve Jobs 在一封内部邮件里所写的一样——这不是苹果公司的“光荣时刻”。后来，苹果公司修复了那些漏洞，并且承诺所有的 MobileMe 用户可以免费使用 90 天。2008 年，宕机的现象非常严重，包括 Twitter 网站频繁出现宕机、Gmail 服务宕机 30 小时等，Twitter 宕机标志 Fail Whale 甚至拥有了其狂热者（Fans）的俱乐部、商店等，如 http://www.zazzle.com/failwhale。

【例 2】

Facebook 推出他们的创新广告平台 Beacon 的时候，受到了极其严厉的批评。事实证明，Facebook 的用户不喜欢让 Web 上的每个人知道他们的交易记录。例如一个小伙子在某个电子商务站点上买了订婚戒指，他的 Facebook 资料里立刻显示了这个交易信息，从而暴露了不该暴露的信息，毁坏了这个小伙子刻意营造的订婚惊喜。之后，Facebook 在 Beacon 里增加了选项，允许用户设置不显示相关的信息。但是很多不良的影响已经造成了，例如就有一对夫妇一起诉讼 Facebook 及其合作伙伴的这类服务。

【例 3】

2008 年 8 月诺基亚承认该公司 Series40 手机平台存在严重缺陷，Series40 手机所使用的旧版 J2ME 中的缺陷使黑客能够远程访问本应受到限制的手机功能，从而使黑客能够在他人的手机上

秘密地安装和激活应用软件。

【例 4】

2007 年 10 月 30 日上午 9 点，北京奥运会门票面向境内公众的第二阶段预售正式启动。由于瞬间访问数量过大造成网络堵塞，技术系统应对不畅，造成很多申购者无法及时提交申请，为此，票务中心向广大公众表示歉意，并宣布暂停第二阶段的门票销售。

【例 5】

2007 年美国 12 架 F-16 战机执行从夏威夷飞往日本的任务中，因电脑系统编码中犯了一个小错误，导致飞机上的全球定位系统纷纷失灵，有一架战机折戟沉沙。

【例 6】

2003 年 8 月 14 日发生的美国及加拿大部分地区史上最大停电事故就是软件错误导致的。SecurityFocus 的数据表明，位于美国俄亥俄州的第一能源（FirstEnergy）公司下属的电力监测与控制管理系统“XA/21”出现软件错误，是北美大停电的罪魁祸首。根据第一能源公司发言人提供的数据，由于系统中重要的预警部分出现严重故障，负责预警服务的主服务器与备份服务器接连失控，使得错误没有得到及时通报和处理，最终多个重要设备出现故障导致大规模停电。

【例 7】

2003 年 8 月 11 日，“冲击波”计算机病毒首先在美国发作，使美国的政府机关、企业及个人用户的成千上万台计算机受到攻击。随后，“冲击波”蠕虫很快在 Internet 上广泛传播，结果使十几万台邮件服务器瘫痪，给整个世界范围内的 Internet 通信带来惨重损失。“冲击波”计算机病毒仅仅是利用微软 Messenger Service 中的一个缺陷，就攻破了计算机的安全屏障，使基于 Windows 操作系统的计算机崩溃。

【例 8】

导航软件 bug 使俄罗斯飞船偏离降落地。2003 年 5 月 4 日，搭乘俄罗斯“联盟—TMA1”载人飞船的国际空间站第七期考察团的宇航员们返回了地球，但在返回途中，飞船偏离了降落目标地点约 460km。据来自美国国家航空航天局的消息称，这是由飞船的导航计算机软件设计中的错误引起的。

【例 9】

仅仅由于没有进行集成测试，导致 1999 年美国宇航局的火星探测器在试图登陆火星表面时坠毁。原因是当探测器的脚迅速摆开准备着陆时，触发了着地开关，设置了错误的数据位，导致探测器在着陆之前反推器被关闭。

【例 10】

爱国者导弹防御系统存在一个致命的软件缺陷，当时钟累计运行超过 14h 后，防御系统的跟踪系统就不准确了。在 1999 年多哈袭击战中，由于爱国者导弹防御系统连续运行 100 多个小时，所以问题就发生了，它未能防住飞毛腿导弹，从而造成 28 名美国士兵死亡。

【例 11】

1996 年欧洲航天局阿丽亚娜 5 型火箭发射后 40s 火箭爆炸，发射基地 2 名法国士兵当场死亡，历时 9 年的航天计划严重受挫，震惊了国际宇航界。爆炸是由惯性导航系统软件技术和设计中的一个小失误引起的。

【例 12】

1994 年，已大批卖出的 Intel 奔腾 CPU 芯片存在浮点运算的缺陷，导致 Intel 公司为此付出了

4.5 亿美元的代价。

1.3 软件缺陷的由来

软件缺陷是指软件中所存在各种各样的问题，包含了一些偏差、谬误或错误，其主要表现形式是结果出错、功能失效、与用户需求不一致（偏差）等。软件产品或系统中存在的任何一种影响正常运行能力的问题或错误、隐藏的功能缺陷或瑕疵，都被认为是软件缺陷。说到底，软件缺陷会导致软件产品在某种程度上不能满足用户的需要。IEEE 国际标准 729 给出了软件缺陷的定义：软件缺陷就是软件产品中所存在的问题，最终表现为用户所需要的功能没有完全实现，不能满足或不能全部满足用户的需求。

（1）从产品内部看，软件缺陷是软件产品开发或维护过程中所存在的错误、误差等各种问题。

（2）从外部看，软件缺陷是系统所需要实现的某种功能的失效或违背。

软件缺陷反映了软件开发过程中需求分析、功能设计、用户界面设计、编程等环节所隐含的问题。软件缺陷表现的形式有多种，不仅体现在功能的失效方面，而且体现在下列其他方面。

- 设计不合理，不是用户所期望的风格、格式。
- 部分实现了软件某项功能。
- 实际结果和预期结果不一致。
- 系统崩溃、界面混乱。
- 数据结果不正确、精度不够。
- 存取时间过长、界面不美观。

由于软件开发人员思维上的主观局限性，且目前开发的软件系统都具有相当的复杂性，所以在开发过程中出现软件错误是不可避免的。引起软件缺陷的原因比较复杂，来源于方方面面，有软件自身的，也有沟通问题，还有技术问题等。主要有下列因素。

- 在需求定义时，用户或产品经理在产品功能上还没有想清楚。
- 当一个产品在做出来之前的设想过程中，很难想得很完美。
- 需求分析、系统设计时，相关方不能准确表达自己的意见，沟通时也会存在误解，特别是技术人员和用户、市场人员的沟通上存在较大的困难。
- 沟通不充分，或在多个环节沟通以后使信息失真，误差就会不断放大，真可谓“失之毫厘，谬以千里”。
- 软件设计规格说明书（Functional Specification，Spec）中有些功能不合理导致其无法实现。
- 系统设计存在的不合理性，难以面面俱到，容易忽视某些质量特性要求。
- 复杂的程序逻辑处理不当，数据范围的边界考虑不够周全，容易引起程序出错。
- 算法错误或没有进行算法优化，从而造成精度不够或性能低下甚至程序错误。
- 软件模块或组件多，接口参数多，配合不好，容易出现不匹配的问题。
- 异常情况一时难以想到；某些特别的应用场合，如时间同步、大数据量和用户的特别操作等难以想到。
- 其他人为错误，如文字写错、数据输入出错、程序敲错等。

1.4　软件测试学科的发展历程

上面的故事告诉我们，任何一个系统都可能存在故障，计算机系统也不例外。有了计算机软件，就可能存在问题；有了问题，就需要测试，一切看起来都是自然而然地发展起来的。实际上，软件测试的发展历程并不一帆风顺，而是经历了一些波折，特别是在早期阶段，软件测试得不到重视，测试技术发展比较慢；只是在最近一二十年软件测试技术的发展比较快，成为软件业的热门领域之一，也达到了较高的水准。

对于软件测试的发展历史，没有统一的、明确的阶段划分。有一种划分，是从测试的基本思想或导向来划分的，分为 4 个阶段：

（1）1957—1978 年，以功能验证为导向，测试是证明软件是正确的（正向思维）。

（2）1978—1983 年，以破坏性为导向，测试是为了找到软件中的错误（逆向思维）。

（3）1983—1987 年，以质量评估为导向，测试是提供产品的评估和质量度量。

（4）自 1988 年起，以缺陷预防为导向，测试是为了展示软件符合设计要求，发现缺陷、预防缺陷。

这里将软件测试历史分为 3 个阶段——初级阶段、发展阶段和成熟阶段。

1. 初级阶段（1957 年—1971 年）

在早期，软件规模都很小、复杂程度比较低，而且软件开发的过程混乱无序、相当随意，所以测试的含义比较狭窄，测试被视为“调试”，目的是纠正软件中已经知道的故障。早期对测试的投入极少，而且测试介入也晚，不到代码全部写完，测试不会开始。这时，还没有形成真正意义上的软件测试，也还没有出现专业的测试人员。

直到 1957 年，软件测试才开始与调试区别开来，作为一种发现软件缺陷的活动。但测试通常被认为是对产品进行事后检验，检查软件产品是否能正常工作，所以测试活动始终落后于开发活动，往往被安排在软件生命周期中的最后阶段。

这一阶段，缺乏有效的测试方法，主要依靠“错误推测（Error Guessing）”来寻找软件中的缺陷。因此，软件交付后，一般会存在比较多的问题，软件产品的质量无法得到保证。

2. 发展阶段（1972—1982 年）

这个阶段的标志性事件是 1972 年在美国北卡罗来纳大学（The University of North Carolina）召开了历史上第一次关于软件测试的正式会议。而此前，软件危机愈演愈重，迫使人们开始思考如何用系统的、工程化的方法来克服软件危机。在 1968 年北大西洋公约组织（NATO）的计算机科学家在原联邦德国召开国际会议，讨论软件危机问题，正式提出了“软件工程”思想。软件开发的方式逐渐由混乱无序的开发过程过渡到结构化的开发过程，在需求分析、设计和测试等活动中普遍采用了结构化方法。在软件工程的思想影响下，软件测试得到发展，软件测试的地位也得到了确认。

这一阶段产生了专业的测试人员，软件测试开始得到重视。软件测试的先驱者建议在软件生命周期的开始阶段就应该根据需求制订测试计划，并进行了大量的研究、探索和实践。人们开始进行主动的测试，努力搜寻软件缺陷，但软件测试的工作主要集中在基本的软件功能验证之上。

3. 成熟阶段（1983 年至今）

从 20 世纪 80 年代以来，软件业进入了高速发展时期，渗透到工业的各个领域和人们日常生活中的各个方面，软件产业逐渐走向成熟，软件质量越来越重要。同时，软件系统规模越来越大、

复杂程度越来越高，不断对软件的开发和测试提出新的挑战。

这个阶段的测试不再单纯是一个发现错误的过程，而且包含软件质量评价的内容。软件开发人员和测试人员开始坐在一起探讨软件工程和测试的问题，制定软件测试的专业标准。1983 年，国际电气和电子工程师协会（IEEE）发布了国际标准 Std 829-1983《软件测试文档 IEEE 标准》IEEE Standard For Software Test Documentation，其最新版本是 Std 829-2008 IEEE Standard for Software and System Test Documentation），可以看作是一个标志性的事件。软件测试终于有了自己的国际标准，形成一门独立的学科和专业，成为软件工程学科中的一个重要组成部分。这也预示着软件测试走向成熟，从此，软件测试的内涵发生了变化，测试不再停留在发现问题上面，软件测试被看作是软件质量保证（SQA）的重要手段，软件测试已完全溶于整个软件生命周期中。正如 Bill Hetzel 在《软件测试完全指南》（*The Complete Guide to Software Testing*）一书中指出："测试是以评价一个程序或者系统属性为目标的任何一种活动。测试是对软件质量的度量。"

2002 年，Rick.D.Craig 和 Stefan.P.Jaskiel 在《系统的软件测试》（*Systematic Software Testing*）中对软件测试重新进行了定义："测试是为了度量和提高被测软件的质量，对测试软件进行工程设计、实施和维护的整个生命周期过程"。这进一步推动了软件测试的研究和发展，软件测试的理论、方法和技术也逐渐走向成熟。例如，面向对象的测试方法、面向构件的测试方法和测试驱动开发的思想等相继诞生。而且，软件测试工具也得到了快速发展，无论是商业化的测试工具还是开源的软件测试工具，可以说，应有尽有。

1.5 软件测试的定义

在传统的制造业产品生产过程中的每一道工序结束时，都由质量人员进行检验，或由仪器自动进行检验。软件测试，简单地理解，就是对软件产品进行检验和验证，包括对软件阶段性产品和最终产品进行检验。如果要对软件测试有更全面的理解，这样的定义还不够，我们还需要从不同的角度对软件测试进行更为科学的、全面的定义。

1.5.1 基本定义的正反两面性

最早给软件测试下定义的是 Bill Hetzel 博士，他在 1973 年给出了软件测试的第一个定义：软件测试就是为程序能够按预期设想那样运行而建立足够的信心。在 1983 年，他又将软件测试的定义修改为：软件测试是用以评价一个程序或系统的特性或能力并确定是否达到预期的结果的一系列活动，即测试是对软件质量的度量。上述两个定义中的"设想"和"预期的结果"可以被认为是用户的需求或者是产品的功能设计。从 Bill Hetzel 的定义可以看出，测试是为了验证软件是否符合用户需求，即验证软件产品是否能正常工作，是正向思维，测试是针对软件系统的所有功能点来逐个验证其正确性。

1983 年，IEEE 给软件测试下的定义是：使用人工或自动的手段来运行或测定某个软件系统的过程，其目的在于检验它是否满足规定的需求或弄清预期结果与实际结果之间的差别。这个定义和 Bill Hetzel 给出的定义比较接近，强调软件测试的目的是为了检验软件系统是否满足（客户的）需求。随后，1990 年 IEEE 再次给出了软件测试的定义（IEEE/ANSI，1990 [Std 610.12- 1990]）：（1）在特定的条件下运行系统或构件，观察或记录结果，对系统的某个方面做出评价；（2）分析某个软件项以发现现存的和要求的条件之差别（即错误）并评价此软件项的特性。

与正向思维相反的是逆向思维，即人们无法证明软件是正确的，只能认定软件是有错误的，然后去发现尽可能多的错误，以提高软件产品的质量。这种观点的代表人物是 Glenford J. Myers（代表作《软件测试的艺术》），他还从人的心理学的角度论证，如果将“验证软件是工作的”作为测试的目的，非常不利于测试人员发现软件的错误。于是他于 1979 年给软件测试下了一个完全不同的定义：测试是为发现错误而针对某个程序或系统的执行过程。简单地说就是验证软件是“不工作的”，或者说是有问题的。从这个概念出发，一个成功的测试必须是发现缺陷的测试，不然就没有价值。这就如同一个病人（假定此人确实有病），到医院做一项医疗检查，结果各项指标都正常，那说明该项医疗检查对于诊断该病人的病情是没有价值的，是失败的。概括起来，Myers 给出了与测试相关的三个要点：

- 测试是为了证明程序有错，而不是证明程序无错误。
- 一个好的测试用例是在于它能发现至今未发现的错误。
- 一个成功的测试是发现了至今未发现的错误的测试。

基于“验证”的观点，主要是在设计规定的环境下，运行软件的各项功能，将其结果与用户需求或设计结果相比较，如果相符则测试通过，如果不相符则视为不通过，要进行修正，直至所有的功能通过验证。基于“找错”的观点，强调测试人员发挥主观能动性，用逆向思维方式，不断思考人们容易犯错误的地方（如误解、不良习惯造成的，数据边界）和系统的薄弱环节，试图破坏系统，从而发现系统中所存在的问题。图 1-1 对上述两种看似对立的观点进行了总结。

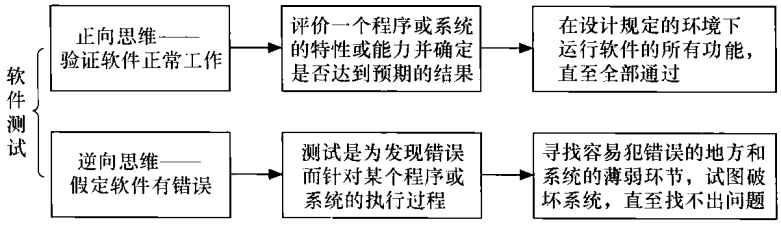


图 1-1　软件测试定义的对立性（两面性）

这两种观点都有一定的局限性，正向思维有利于界定测试工作的范畴、促进与开发人员协作，但可能降低测试工作的效率。而逆向思维有利于测试人员主观能动性的发挥、可以使测试人员发现更多的问题，但也容易使测试人员忽视用户的需求，使测试工作存在一定的随意性和盲目性。实际上，测试可以看作这两者的统一，既要尽可能地、快速地发现问题，加快测试的进程；又要对实现的各项功能进行验证，保证测试的完整性和全面性。

1.5.2　服从于用户需求——V&V

前面讨论的是软件测试是验证软件能正常工作、还是设法找出软件不能工作的地方，重点讨论的是从正向思维还是反向思维来考虑，但忽视了一个基本点，那就是基于什么来判断软件是能正常工作还是不能正常工作。在软件测试时，必须建立判断的基准，也就是判断软件是否存在缺陷的依据。判断软件是否存在缺陷的基本依据是软件的用户需求，软件功能特性就是为了满足用户需求，不能满足用户需求的功能是有缺陷的。从这一点来看，测试要服从于用户需求，以用户需求为依据，来对产品进行检验。

但是，软件测试不能靠用户来完成，还必须由软件开发组织的测试人员来完成，所以我们要为软件建立相应的质量标准和软件设计规格说明书（Spec）。软件规格说明书是用户需求的描述，

是对待实现的功能特性的说明，从而使软件设计、编程人员知道要完成哪些功能、要将软件做成什么样子。从这一点来说，软件测试就是检验开发人员是否是按照规格说明书来构造产品的，所构造的产品是否和规格说明书一致。

V&V 代表 Verification 和 Validation，即“验证”和“有效性确认”。软件测试被看作就是执行这两项任务，软件测试可以被定义为“验证”和“有效性确认”两项活动构成的整体。

（1）“验证”是检验软件是否已正确地实现了产品规格书所定义的系统功能和特性。验证过程提供证据表明软件相关产品与所有生命周期活动的要求（如正确性、完整性、一致性、准确性等）相一致。相当于，以软件规格说明书为标准进行软件测试的活动。

（2）“有效性确认”是确认所开发的软件是否满足用户真正需求的活动。因为软件规格说明书本身就可能存在问题，这些问题是无法通过“验证”活动来发现，而必须通过“有效性确认”活动来发现问题，即一切从客户的观点出发，正确理解客户的需求，敢于怀疑需求定义和设计中不合理的东西，发现需求定义和产品设计中的各种问题。“有效性确认”活动主要通过各种软件评审活动来实现，包括让客户参加评审、测试活动。

从以上内容可以看出，软件测试不仅要通过运行软件程序或软件系统来进行检验，而且要对软件相关文档，特别是需求定义和设计规格说明书等进行评审，以确认这些文档所描述的内容都是客户所需要的。这就说明，Myers 明显受到传统的“软件开发瀑布模型”影响，早期所给出的测试定义——*程序测试是为了发现错误而执行程序的过程*是片面的、不足的，它限制了软件测试活动。将需求和设计阶段产生的问题，留到编程之后去发现、去解决，其结果将造成设计、编程的部分或全部需要返工，给软件开发带来巨大的劣质成本。实际上，我们也清楚地知道，软件不等于程序或软件包，软件应当包括软件开发过程中产生的文档，软件是程序、文档和数据构成的一个集合或系统。

V&V 将软件测试扩展到用户需求、设计等早期阶段，实质上就是将软件测试扩展到整个软件生命周期。软件测试不再是发生在编程之后的某个阶段，而是贯穿于整个软件生命周期。所以，软件测试不仅包含了动态测试，而且也包含了静态测试。

（1）“动态测试”是通过运行程序来发现软件系统中的问题。这种测试是在程序运行过程中将缺陷发现出来，具有动态性，所以称为动态测试。

（2）“静态测试”主要活动是评审，即通过对需求、设计、配置、程序和其他各类文档的审查来检验相应内容是否满足用户的需求。不需要运行程序，测试对象是属于静态的。

概括地说，测试可以看作“验证”和“有效性确认”的统一，以客户的需求为基准，参照设计规格说明书，不仅对程序进行检验，而且要对各种软件文档进行评审，从而更早地发现软件中存在的各种缺陷，降低软件开发成本，全面提高软件质量。

1.6 软件测试和软件开发

软件测试和软件开发是软件生命周期中最主要的活动，两者是不可分离的，是相辅相成的。在软件开发中，虽然有很多角色，有各种各样的人员，包括项目经理、产品经理、UI（用户界面）设计人员、文档人员等，但最大的两个团队就是测试团队和开发团队（由设计人员/程序员组成），也就是说，在软件公司研发团队中，测试人员和程序员是主体。

在详细介绍软件测试和软件开发之间的关系之前，先介绍一些常用的软件测试概念。