

- ★ 提示、诀窍、路线和技术，妙趣横生，精彩纷呈
- ★ 软件测试艺术随轻松、幽默和机智的行文娓娓而来



探索式软件测试

(美) James A. Whittaker 著
方敏 张胜 钟颂东 郭艳春 译



清华大学出版社

探索式软件测试

(美) James A. Whittaker 著

方 敏 张 胜

译

钟颂东 郭艳春

清华大学出版社

北 京

内 容 简 介

探索式软件测试作为一种富有创新精神和现实意义的测试方法,引起越来越多软件测试人员、质量保证人员和项目经理的高度重视。本书作者结合自己二十年的经验,从多个角度结合丰富的实例阐述了探索式软件测试的使用技巧、提示和相关技术。全书共 8 章,3 个附录,为手工测试流程提供了重要的指导、技术和规划。

本书适合软件测试人员、OA 专家、开发人员、程序经理和架构师阅读,对他们的工作具有重要的启发作用。

Simplified Chinese edition copyright © 2010 by **PEARSON EDUCATION ASIA LIMITED and TSINGHUA UNIVERSITY PRESS.**

Original English language title from Proprietor's edition of the Work.

Original English language title: Exploratory Software Testing, by James A. Whittaker © 2010
EISBN: 978-0-321-63641-6

All Rights Reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as Pearson Education.

This edition is authorized for sale only in the People's Republic of China (excluding the Special Administrative Region of Hong Kong and Macao).

本书中文简体翻译版由 Pearson Education 授权给清华大学出版社在中国境内(不包括中国香港、澳门特别行政区)出版发行。

北京市版权局著作权合同登记号 图字:01-2010-7734

本书封面贴有 Pearson Education(培生教育出版集团)激光防伪标签,无标签者不得销售。

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

探索式软件测试/(美)惠特克(Whittaker J.)著;方敏,张胜,钟颂东,郭艳春译.一北京:清华大学出版社,2010.4

书名原文:Exploratory Software Testing

ISBN 978-7-302-22384-9

I. 探… II. ①惠… ②方… ③张… ④钟… ⑤郭… III. 软件—测试 IV. TP311.5

中国版本图书馆 CIP 数据核字(2010)第 054135 号

责任编辑:文开琪

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 刷 者:清华大学印刷厂

装 订 者:三河市李旗庄少明装订厂

经 销:全国新华书店

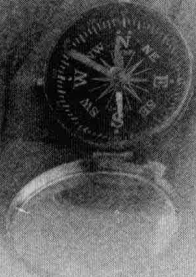
开 本:185×230 印 张:15.5 插 页:2 字 数:373 千字

版 次:2010 年 4 月第 1 版 印 次:2010 年 4 月第 1 次印刷

印 数:1~4000

定 价:35.00 元

产品编号:035379-01



献 辞

本书大部分内容是我在微软公司担任架构师时所写。在这里，将它献给我在微软时与我共事的所有天才测试人员。谢谢你们！你们改变了我思考和工作的方式，更改变了我对软件测试行业的展望。祝你们再接再厉，更创高峰！



序

我第一次见到詹姆斯·惠特克(James Whittaker)是在几年前,当时他是佛罗里达技术学院(Florida Institute of Technology)的教授。他当时是访问了雷德蒙(Redmond)的微软公司园区,向一个测试工程师小组演讲他钟爱的课题——软件测试。从第一次见面起,我就明显地感受到 James 风趣的谈话风格和渊博的软件测试知识。多年的教学生涯使他很容易与那些好学者打成一片。

詹姆斯在 2006 年加入微软公司,在过去的三年中,我有机会和詹姆斯在一起工作很长时间,对他有了更深的了解。我可以高兴地告诉大家,他的幽默和善于与测试人员交流的能力继续在他的教学和交流方面起着关键性的作用。每一次我想找他谈话,几乎都看到他在与某个测试工程师或一个测试团队进行交流或正在鼓励他们。虽然我们在微软从来没有在同一个团队里工作过,但曾经有过很多机会在一起为跨公司的倡议而工作,共同负责为公司的新员工做演讲。(当然了,我所谓的“共同”意味着詹姆斯写演讲稿,而我剽窃他的幽默。)他在微软公司工作期间,我们真正进行的长时间交流还是发生在微软的足球场上。我们在过去三年里,大概一起度过了一百多个小时,把球传过来踢过去,同时还切磋和交流各种改进软件测试和开发的想法。

詹姆斯最重要的特点是,一旦有了新想法,他总想去检验并证明它。(哪个伟大的软件测试大师不这样?)他如此成功的原因是,他不怕失败,也敢于承认一些想法不成熟。也许我的测试本能使我比普通的测试人员更多疑,但是说起来,我还是觉得很高兴在过去的几年里,我曾经驳倒过詹姆斯的很多“伟大的想法”。詹姆斯教导他的学生时经常使用如下名言:“大多数伟大想法的背后是一片埋葬着不成熟想法的墓地。”这句话不无道理,一个成功的创新者必须谦虚谨慎。

由于我在微软公司的角色,我有机会看到无数充满创意的新点子,也亲历了很多点子。其中很多是富有潜力的优秀发明,但是由于发明者没能具体实现他的想法而最终失败了。随着詹姆斯和我更多的会面以及相互讨论测试想法,我有机会观察到他把几个想法有条不紊地开发成实际有用的创新成果。这些创新成果被真实的微软测试人员所采用。他为测试人员设计的“提示显示”(Tester'Heads Up Display)技术就是其中一个例

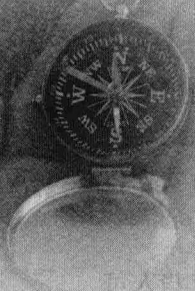


子。这些想法最初在足球场上酝酿，在实践中优化，最终产生实际的用途——凭借这项技术，测试人员在工作时可以获得并使用实时测试数据。由于这项发明，微软公司为詹姆斯颁发了大奖，Visual Studio 更有意要把这个概念用在其测试产品的未来版本上。

我亲身见证了詹姆斯设计出“漫游”这个隐喻(metaphor)来指导软件测试。他也许不是第一个谈论漫游的人，但他是我知道的第一个完整设计出漫游隐喻的人，他随后还指导了几十个测试团队将其成功地应用在真实的软件(非常复杂的软件)。他的漫游路径已从几条增加到几十条。他不断地开发和优化这些概念，直到它们变得完善。虽然詹姆斯提出的有些漫游并不成功，但是不用担心，詹姆斯敢于放弃那些想法，所以在这里就不再提及它们了。这本书里包括了软件测试漫游秘笈的全部工作。这些秘笈被检验过，修改过，然后再次被检验过。詹姆斯擅长以讲故事的方式来描述概念，这种能力在本书中表现得淋漓尽致。所以当我阅读这本优秀的软件测试书时，有时甚至不禁忘记它竟然是一本讲测试的书。虽然我弄不清在测试行为中使用隐喻为何会让漫游测试法变得这么有效，但是漫游技术在现实世界中表现得如此之好，我再怎么夸奖也不过分。这个概念太重要了，以致于微软已把“漫游测试法”加入专为新入职测试人员提供的技术培训课程。

如果你有兴趣提高自己的技能或者提高整个团队的技能，这本书会很有帮助。本书不仅生动有趣，更是一本可以常备手边供随时参考翻阅的手册。

微软卓越测试部门总监 阿伦·培智(Alan Page)



前言

“用户购买功能的同时也在忍受缺陷。”

——史考特·沃兹沃思(Scott Wadsworth)

任何一个使用过电脑的人都知道软件故障。从最初的第一个程序到最新的现代应用程序，软件从来没有完美过。

将来这一切也多半不会改变。不只是软件开发错综复杂和开发人员容易犯错的天性，还有硬件、操作系统、运行时环境、驱动程序、平台、数据库等不断变化，这些加在一起使软件开发任务成为全人类最让人称奇的专业技能之一。

不过，仅仅令人称奇是不够的，正如本书第 1 章指出的那样，这个世界还要求软件拥有高质量。

显然，关心质量不只是软件测试人员的事。我们应该用正确的方式构建软件，将可靠性、安全性和高性能等作为系统设计的一部分来考虑，而不是到开发末期才想起它们。然而，说到理解软件缺陷的本质，测试人员总是站在前沿。如果没有测试人员在测试的最前沿发挥他们的洞察力、技术以及应变措施，使这样的可能性变为现实，软件质量的全面解决方案差不多算是“镜中花，水中月”。

谈论软件质量的方法有很多，感兴趣的听众也有很多。本书是为软件测试人员而写的，写的是一种我认为比其他任何缺陷都重要的特殊缺陷：即逃过所有各种检测手段而最终存在于发布产品中的缺陷。

任何一个软件公司发布的产品都有缺陷。缺陷是怎么引入的？为什么没有在代码审核、单元测试、静态分析或其他面向开发人员的活动中把它们找出来？为什么自动化测试没有找出它们？那些缺陷有些什么特质使其能逃过手工测试？

什么是找出产品缺陷的最好方法？

本书针对的正是最后一个问题。在第 2 章“手工测试”中，我提出了一个观点：因



为用户是在使用软件过程中找到这些缺陷的，所以我们的测试人员也应该通过使用软件来找到它们。无论使用自动化测试和单元测试，还是其他一些手段，都难以接触到这些缺陷。无论测试人员怎么实现自动化测试，即使全部都自动化，这些缺陷还是会处处作怪，并在产品中屡屡重现从而伤害最终用户。

问题在于很多现代化手工测试实践都缺乏目的性，随机性强且重复性强。有些人可能还会加上一条：手工测试无聊透顶。本书试图为手工测试流程提供一些指导、技术和规划。

在第3章“局部探索式测试法”中，针对测试人员在运行任何一个测试用例时都需要做出很多细微的战术层面决定，我给出了详尽的指导建议。测试人员必须决定对于某个特定的输入字段应该使用什么输入值，或者给应用程序使用的文件提供什么数据。在测试过程中，必须做出许多这样的小决定。在缺乏指导的情况下，这些决定常常是未经分析且不是最优化的。在向一个文本框内输入一个数时，选择整数4难道就胜过整数400么？应该用长度为32字节的字符串还是长度为256字节的字符串？选择一个而不选另一个是有一定道理的，这一切都取决于处理该输入的软件的具体情况。鉴于测试人员每天都要做出数百次这样的小决定，在这里提供有效的指导建议显得至关重要。

在第4章“全局探索式测试法”中，针对测试人员在编制测试计划和测试用例设计时需要考虑哪些广泛的战略性问题，我也给出了一些指导建议。这些技术都基于“漫游测试”(tour)概念，如同一个导游带领旅游团队参观大都市中一系列著名景点一样，这种漫游测试法指出的路线可以指导测试人员如何探索软件的方方面面。这里的探索并不一定是随机的或者漫无目的的。本书所记录的方法已经成为微软和谷歌的许多测试人员日常工作的一部分。诸如“地标测试法”(landmark tour)和“极限测试法”(intellectual's tour)等词汇已经列入了手工测试人员的标准词汇表中。测试技术以前确实被称作“漫游”，但是用整个旅游业来隐喻软件测试，并在测试实际发布的应用程序时，大规模使用这些隐喻的名称，还属于本书的一个创举。

全局探索式测试法对于制定完整的测试策略给出了指导建议。例如，如何创建一组特性覆盖率(feature coverage)较高的测试用例？如何确定是否要在一个单独的测试用例中使用多个特性？如何创建一个完整的测试用例套件(test case suite)，从而使软件尽可能地满负荷工作以便能找到更多重要的缺陷？这些都是设计测试用例和保证测试套件质量时必须解决的重大问题。

在第5章“混合探索测试技术”中，通过把探索式测试和传统的脚本或基于场景的

测试技术相结合，进一步扩展了漫游的概念。我们将讨论如何修改各种端到端场景(end-to-end scenario)、测试脚本(test script)或用户故事(user story)，来创造更多的变化情况，以激发传统静态测试技术查找缺陷的潜力。

在第6章“探索式测试的实际应用”中，来自微软不同产品组的五位客串作者提供了他们使用漫游技术后得到的经验报告。这些作者和他们的团队在真实的开发环境中，把漫游方法应用在真实的软件上。他们记录了各自是如何使用漫游、修改漫游甚至创建自己的漫游的。这些内容来自于使用漫游法测试重要的关键软件产品的测试人员，属于真正的第一手资料。

最后，我用两章内容总结前面各章所讨论的内容。在第7章“漫游测试的棘手问题”中，描述了我认为的测试中最困难的几个问题，以及如何将那些具有高度针对性的探索式测试方法融入一个更广泛的解决方案中。在第8章“软件测试的未来”中，我更进一步讨论在未来几年中，诸如虚拟化、可视化甚至电视游戏之类的技术将如何改变测试的面貌。附录包括我对测试职业生涯的看法，收集了我以前一些深受读者喜爱的文章(加入了一些新的注解)，其中一些文章已经无法在其他地方看到了。

写这本书对我来说是一种享受，我希望你阅读本书也是一种享受。



致 谢

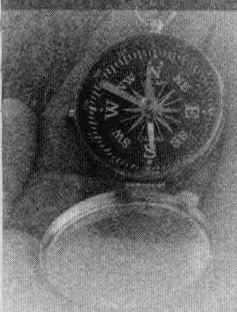
我要感谢所有不断追求微软软件质量提高的微软测试人员。我还要感谢那些允许本书合作者尝试新鲜事物的微软经理。它的成功运作正说明了该公司测试经理们的智慧和远见。

我还要感谢下列微软人士，他们通过阅读、评论、审阅和协作等其他方式帮助我全面认识了漫游测试法：

David Gorena Elizondo	Mark Mydland	Ahmed Stewart	Geoff Staneff
Joe Alan Muharsky	Naysawn Naderi	Anutthara Bharadwaj	RyanVogrinec
Hiromi Nakura	Nicole Naugen	Alan Page	Vessie Djambazova
Shawn Brown	Kyle Larson	Habib Heydarian	Bola Agbonile
Michael Fortin	Ratnaditya Jonnalagadda		Dan Massey
Koby Leung	Jeremy Croy	Scott Wadsworth	Craig Burley
Michael Bilodeau	Brent Jensen	Jim Shobe	Vince Orgovan
Tracy Monteith	Amit Chatterjee	Tim Lamey	Jimbo Pfeiffer
Brendan Murphy	Scott Stearns	Jeff Mac Dermot	Chris Shaffer
Greg B. Jones	Sam Guckenheimer		Yves Neyrand

其他一些非微软人士也给予了我很大的帮助，他们是：Gitte Ottosen, Rob Lambert, Beth Galt, Janet Gregory, Michael Kelly, Charles Knutson 以及 Brian Korver。

最后，感谢我在谷歌的新同事 Alberto Savoia 和 Patrick Copeland，感谢他们对我的鼓励以及他们将来在谷歌为探索式测试做出的新贡献。



作者介绍

詹姆斯·惠特克(James Whittaker)的全部职业生涯都致力于软件测试,在该学科的许多方面都留下了他的印记。他是基于模型的测试领域的先驱,他在田纳西大学的博士学位论文是该主题的标准参考资料。他在错误注入(error injection)方面的工作,创造了备受欢迎的运行时错误注入工具 Holodeck。他是软件安全和渗透测试(penetration testing)的早期创导者。作为教师和演讲者,他也为人们称道,他曾在国际会议上赢得过多个最佳论文和最佳演讲奖。在佛罗里达技术学院担任教授期间,他的软件测试课程吸引了来自业界和世界各国政府的几十位赞助者,而他的学生们后来也因他们在测试方面的深厚技术功底而被争相聘用。

惠特克博士还著有《如何攻破软件》(How to Break Software)及其一系列后续著作《如何破坏软件安全》(Break Software Security, 与 Hugh Thompson 合著)和《如何破坏网络软件》(How to Break Web Software, 与 Mike Andrews 合著)。在十年的教授生涯后,他于 2006 年加入微软,于 2009 年离开并加入谷歌担任科克兰(Kirkland)和西雅图办公室的测试工程部门总监。他住在华盛顿州的 Woodinville 市,为了将来某一天软件能完全正确工作而孜孜不倦奋斗着。

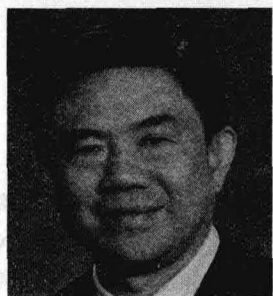


译者介绍

本书中文版执行局成员

CEO: 方敏

CTO: 张胜、钟颂东、郭艳春



方 敏

现任微软亚洲工程院 UIS 项目首席测试部门主管，拥有 20 年软件测试管理和开发的丰富经验，曾参加过微软多项重大产品和技术的研究，包括 UIS, Windows Server/Client/Security, SQL Server, Exchange Server, MSN, COM+ Services, Windows Medi 和微软内部 IT 工具等。方敏曾在清华大学获得电子工程学士和硕士学位，在美国新墨西哥技术学院获得计算机硕士学位。

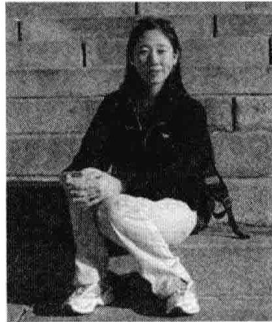


张 胜

现任微软总部高级软件开发测试主管，拥有 10 余年软件开发测试和团队管理经验，参与 Visual Studio, SQL Server 和 Office Live 的开发测试与发布，现主管 Office Communications Server 本地化软件开发测试工作。张胜拥有复旦大学计算机系硕士和学士学位。



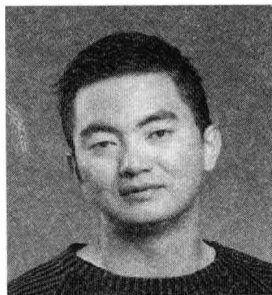
钟 颂东 2006年至今任微软总部SQL团队引擎部的开发测试工程师，负责SQL 2005，2008，2008 R2等版本的测试任务，关注领域有SQL整合与虚拟化、性能和恢复等。曾在NACT和Verso等公司任职。拥有10余年软件设计、开发、网络及数据库管理和测试经验。钟颂东毕业于中国科技大学，拥有物理学学士学位，曾在杨百翰大学进修过计算机硕士及物理博士课程。英文名 Antony Zhong。



郭 艳春 现任微软总部 Windows 部门软件开发测试工程师。拥有10余年软件开发测试经验，参与过多个Windows XP, Vista 以及 Windows 7 的正版增值项目。郭艳春拥有哈尔滨工业大学计算机系硕士和硕士学位。



林 俊彦 毕业于复旦大学计算机专业，此后加入微软中国研发集团服务器与开发工具事业部，现为 Visual Studio Team System 软件开发测试工程师，曾参与 Visual Studio 和 .Net Framework 多个版本的开发测试工作，业余时间热心于专业资料的翻译工作。



林 涛 现为微软总部软件开发工程师，从事云计算广告平台开发工作。曾参与过 Office 部门用户行为智能分析核心模块的开发测试。他拥有华中科技大学学士学位和加拿大里贾纳大学博士学位。



目 录

第 1 章 软件质量	1	小结	41
软件的魔力	1	练习题	42
软件失效	4	第 4 章 全局探索式测试法	45
小结	9	探索软件	45
练习题	9	旅游者比喻	47
第 2 章 手工测试	11	漫游测试	49
软件缺陷的根源	11	商业区测试类型	51
缺陷预防和检测	12	历史区测试类型	58
缺陷预防	12	娱乐区测试类型	60
缺陷检测	13	旅游区测试类型	63
手工测试	15	旅馆区测试类型	66
手工测试中使用脚本	16	破旧区测试类型	68
探索式测试	16	漫游测试法实战	70
小结	21	小结	72
练习题	21	练习题	72
第 3 章 局部探索式测试法	23	第 5 章 混合探索式测试技术	73
想不想测试软件?	23	场景和探索	73
测试就是有所变, 有所不变	25	使用基于场景的探索式测试	75
用户输入	26	通过场景操作引入变化	76
状态	36	插入步骤	76
软件状态的基本知识	36	删除步骤	77
如何测试软件状态	37	替换步骤	77
代码路径	39	重复步骤	78
用户数据	39	替换数据	78
运行环境	41	替换环境	78

通过漫游测试引入变化.....	80	破坏测试法实例.....	102
卖点测试法.....	80	超模测试法实例.....	103
地标测试法.....	81	Windows 媒体播放器的漫游测试	
极限测试法.....	81	实践.....	105
深巷测试法.....	81	Windows 媒体播放器.....	105
强迫症测试法.....	81	遍历测试法.....	106
通宵测试法.....	81	超模测试法.....	108
破坏测试法.....	82	极限测试法.....	109
收藏家测试法.....	82	与 WMP 相关的 25 个“假如”	
超模测试法.....	82	类型的问题.....	109
配角测试法.....	82	极限测试法：边界之旅.....	110
取消测试法.....	83	停车场测试法及其在 Visual Studio	
混票测试法.....	83	Team System 测试版的应用.....	112
小结.....	83	Sprint 中的测试.....	112
练习题.....	83	停车场测试法.....	114
第 6 章 实践中的探索式测试.....	85	漫游测试中的测试规划与管理.....	115
漫游测试.....	85	定义地貌.....	115
Dynamics AX 客户端的漫游.....	86	旅行计划.....	116
有用的探索漫游.....	87	让漫游测试运转起来.....	118
收藏家测试法和收集缺陷.....	89	漫游结果的分析.....	118
漫游测试提示.....	92	判断：里程碑和发布.....	119
利用漫游查找隐错.....	94	在实践中.....	119
测试用例管理解决方案的测试.....	94	小结.....	120
取消测试法.....	95	练习题.....	120
破坏测试法.....	96	第 7 章 漫游与测试中的棘手问题.....	121
快递测试法.....	97	软件测试的五个棘手问题.....	121
测一送一测试法.....	98	漫无目的.....	122
在 Windows Mobile 设备中的		重复性.....	124
漫游实践.....	98	暂时性.....	126
我的测试方法和哲学.....	99	单调性.....	127
漫游测试法找到的有趣缺陷.....	101	健忘.....	128

小结	130	正式方法	164
练习题	130	工具	164
第 8 章 软件测试的未来	131	流程改进	165
欢迎来到未来世界	131	第四种提案	166
测试人员的专有提示显示	132	软件测试是艺术、技巧或学科?	166
测试百科	134	恢复对软件行业的尊重	169
测试用例的重用	135	事与愿违的过去	170
测试原子和测试分子	136	寻找更好的方法	171
虚拟化的测试资产	137	分析安全漏洞和质量问题的 流程	171
可视化	138	附录 3 JW 微软博客修订版	175
未来的测试	141	加入博客圈	175
发布之后的测试	142	2008 年 7 月	176
小结	143	开篇	176
练习题	144	PEST(泡吧与软件测试)	177
附录 1 经营成功的测试职业生涯	145	测试人员评估	179
你是如何开始做测试工作的?	145	预防与治疗(一)	181
回到未来	146	用户与 John	182
上山	147	手工测试人员的赞歌	182
巅峰	149	预防与治疗(二)	185
下山	150	欧洲, 你好!	186
附录 2 JW 的专业博客摘录	151	测试赋	187
教我一些东西吧	151	预防与测试(三)	189
软件诫律	151	回到测试	190
测试错误代码	157	2008 年 8 月	192
真正的职业测试人员, 请上前一步	160	预防与治疗(四)	192
我找到的一些常见的共同特性 (无特别顺序)	161	如果微软擅长测试, 为什么软件 依然糟糕呢?	194
建议总结	162	预防与治疗(五)	197
三击不中出局, 是新的打击手上场的 时候了	163	自由式探索式测试	198
		基于场景的探索式测试	198



基于策略的探索式测试	198	再议手工测试与自动化测试	216
基于反馈的探索式测试	199	2008 年 11 月	218
软件测试的未来(一)	199	不再需要测试人员?	218
软件测试的未来(二)	201	让测试人员继续测试	219
2008 年 9 月	203	2008 年 12 月	220
测试认证	203	谷歌与微软的开发：测试	
软件测试的未来(三)	205	比例之争	220
软件测试的未来(四)	207	2009 年 1 月	221
软件测试的未来(五)	208	Zune 的问题	221
2008 年 10 月	210	解释探索式测试	223
软件测试的未来(六)	210	(未来的)测试用例重用	224
软件测试的未来(七)	212	测试用例重用(续)	226
软件测试的未来(八)	214	休假归来	227
谈到谷歌	216	鼯鼠和受感染的花生	228