★ 高等学校软件工程系列教材

软件测试
——基于问题驱动模式

SOFTWARE TESTING
Based on Problem-driven Mode

朱少民 编著

高等教育出版社

★ 高等学校软件工程系列教材

软件测试

——基于问题驱动模式

SOFTWARE TESTING

Based on Problem-driven Mode

朱少民　编著

高等教育出版社·北京

内容提要

本书以“建构主义”为指导，注重软件测试与应用背景相结合，遵循项目推进的时序过程，以问题驱动方式组织软件测试内容。全书共10章，覆盖软件测试的完整过程所需的知识和技能，包括单元测试、持续集成测试、系统功能测试、性能测试、安全性测试和测试管理等。为了适应当前软件技术的实际应用状态，用专门一章介绍移动App的测试方法、技术和工具；同时，为了进一步提高软件测试的核心技能，包括能够进行更彻底的自动化测试，特别编写了“基于模型的软件测试”这一章。

随书提供丰富的配套资源，包括重点知识的讲解视频、相关工具的说明文档、电子教案、示例源代码等。读者可扫描书中的二维码观看视频/阅读文档，也可从数字课程网站获取相关资源。

本书可作为高等学校计算机科学与技术专业、软件工程专业“软件测试”课程教材，也可供有关技术人员参考。

图书在版编目（CIP）数据

软件测试——基于问题驱动模式 / 朱少民编著
. --北京：高等教育出版社，2017.4
ISBN 978-7-04-047229-5

Ⅰ. ①软… Ⅱ. ①朱… Ⅲ. ①软件-测试-高等学校-教材 Ⅳ. ①TP311.5

中国版本图书馆CIP数据核字（2016）第324336号

Ruanjian Ceshi——Jiyu Wenti Qudong Moshi

策划编辑	倪文慧	责任编辑	倪文慧	封面设计	于文燕	版式设计	王艳红
插图绘制	杜晓丹	责任校对	杨凤玲	责任印制	田　甜		

出版发行	高等教育出版社	网　址	http://www.hep.edu.cn
社　址	北京市西城区德外大街4号		http://www.hep.com.cn
邮政编码	100120	网上订购	http://www.hepmall.com.cn
印　刷	北京北苑印刷有限责任公司		http://www.hepmall.com
开　本	787mm×1092mm　1/16		http://www.hepmall.cn
印　张	27.5		
字　数	510千字	版　次	2017年4月第1版
插　页	1	印　次	2017年4月第1次印刷
购书热线	010-58581118	定　价	49.00元
咨询电话	400-810-0598		

物 料 号　47229-00

数字课程资源使用说明

与本书配套的数字课程资源发布在高等教育出版社易课程网站，请登录网站后开始课程学习。

一、注册/登录

访问 http://abook.hep.com.cn/18663027，点击“注册”，在注册页面输入用户名、密码及常用的邮箱进行注册。已注册的用户直接输入用户名和密码登录即可进入“我的课程”页面。

二、课程绑定

点击“我的课程”页面右上方“绑定课程”，正确输入教材封底防伪标签上的 20 位密码，点击“确定”完成课程绑定。

三、访问课程

在“正在学习”列表中选择已绑定的课程，点击“进入课程”即可浏览或下载与本书配套的课程资源。刚绑定的课程请在“申请学习”列表中选择相应课程并点击“进入课程”。

与本书配套的易课程数字课程资源包括电子教案、教学视频示例源代码等，以便读者学习使用。

账号自登录之日起一年内有效，过期作废。如有账号问题，请发邮件至：abook@hep.com.cn。

前　言

2014年，我开始在网易云课堂做“软件测试”慕课（Massive Open Online Course，MOOC，面向大众的、开放的在线课程），重新审视软件测试课程的教学。在MOOC上线过程中，老师和学生没有面对面的交流和互动，在这种场合下，如果讲比较多的概念和理论，同学们可能听不下去，教学效果不好，达不到教学目标。即使用我自己编写的《软件测试方法和技术》作为参考教材，也不得不对内容进行适当的裁剪，才能满足MOOC教学的需要。MOOC教学特点显著，例如内容要生动，可操作性强，主题明确，每个主题不仅独立成篇，而且要短小精悍。要达到更好的教学效果，仅仅进行内容的裁剪也还不够，还需要改变过去那种平铺直叙的介绍方式，要尽量引起读者或听众的兴趣、激发学生的学习热情。这其中有效的方法之一，就是先摆出问题，让学生思考，然后再去解决问题。这样，学生的注意力就会被引导到问题上，进一步关心问题是如何解决的，这对MOOC的教学会很有帮助，也可以让学生更好地学以致用，将所学的知识或方法应用于实践中去。这就是本书写作的背景，虽然之前已经编写了软件测试方面的教材，但为了更好地适应MOOC教学，适应新形势下的软件测试课程教学需求，编写了这本全新的软件测试教材，并将教材定位于“问题驱动式教学方式的软件测试教材”。

问题驱动式教学方式，可以追溯到“建构主义（constructivism）”。建构主义强调个体的主动性在建构认知结构过程中的关键作用，认为“情境”“协作”“会话”和“意义建构”是学习环境，在该学习环境中更多的是通过人际间的协作活动来完成学习。建构主义是学习理论中从行为主义发展到认知主义之后进一步发展的结果，根据斯皮罗（Spiro）等人的研究，可以将学习分为初级学习与高级学习：

- 初级学习只要求学生知道一些重要的概念和事实，在测验中只要求他们将所学的东西按原样再生出来。
- 高级学习则要求学生把握概念的复杂性，并广泛而灵活地运用到具体情境之中去。

从这个角度看，软件测试的学习属于高级学习，因为软件测试是一门实践性很强的专业课程，在大学高年级学习的课程，学习效果如何，仅仅靠笔试是不够的，必须靠实践检验。即学完课程之后，能否开展实际的测试工作、能否在适当的指导下完成系统的功能、性能、安全性等相关的测试任务，必须在实践中来检验效果。而且，软

件测试所解决的问题相对复杂，依赖于复杂的情景（上下文）——项目背景、软件研发技术和流程、实际业务的应用场景等，所以，不能简单地提取已有知识来解决实际问题，而需要根据具体应用场景来构建解决问题所需的知识；也不能以单个概念、原理为基础，而是通过多个概念、原理以及大量的课程试验，了解单元测试、系统测试和不同类型的测试等知识之间的关系，不断优化学生的知识结构，进而使其能真正掌握相关的高级知识。基于对“建构主义”的理解，形成本书的写作思路，即注重和应用背景的紧密结合，强调“做中学”，在实践（实验）中学习；尽量不去为了讲概念而讲概念，没有单纯的知识介绍，而是在问题解决过程中，当遇到相关的概念或需要相应知识时自然引出，目的就是解决问题。这样，概念和知识更容易理解、便于记忆，自然，教学效果也得到了提升。

本书共 10 章，覆盖软件测试的完整过程所需的知识和技能，包括单元测试、持续集成测试、系统功能测试、性能测试、安全性测试和测试管理等内容。为了适应当前软件技术的实际应用状态，用专门一章介绍移动应用的测试方法、技术和工具；同时，为了进一步提高软件测试的核心技能，包括能够进行更彻底的自动化测试，特别编写了“基于模型的软件测试”这一章。在内容组织上，本书遵循项目推进的时序过程，同时力求采用问题驱动方式，每一章都是提出问题，然后分析问题、解决问题。各章内容简介如下。

第 1 章软件测试入门，侧重介绍软件测试的必要性和一些重要的基本概念（或知识点），如软件缺陷、软件质量、内部/外部质量、使用质量、静态测试、软件测试层次和软件测试主要活动等，为后续内容的学习打下坚实的基础。

第 2 章需求与设计评审，引入了静态测试，包括分析、互为评审、走查和会议审查等，这样将软件测试扩展到软件研发的整个生命周期，从需求评审开始就启动软件测试，抓住产品开发的源头，能够更有效、更彻底地保证软件产品的质量。

第 3 章单元测试，是系统测试、集成测试的基础，围绕软件中最小的独立工作单元展开其软件测试的讨论，包括单元自动化测试框架 JUnit、逻辑覆盖和基本路径覆盖等测试方法、JMock 技术、类的测试、测试覆盖率分析、代码静态分析等。

第 4 章持续集成测试，主要讨论持续集成的概念和环境、自动化构建、自动部署、自动检查代码、自动单元测试、版本自动验证等内容，将单元测试中代码静态分析、动态测试和持续集成整合到一起，完成持续构建、持续集成、持续测试的一体化实现。

第 5 章系统功能测试，以 Web 应用为背景，介绍如何开展系统的功能测试，从功能测试的准备工作开始，逐步深入系统测试的分析、设计和执行，系统地覆盖了系统功能测试所需的框架、知识、方法和工具应用等。

第 6 章系统性能测试，侧重讨论如何进行系统的性能测试以及完整的系统性能测

试过程，包括明确系统性能的需求、确定系统关键业务流程、设计测试场景和负载模式、开发测试脚本和设置准确的测试环境、执行和监控、测试结果分析等内容。

第7章系统安全性测试，介绍系统常见的安全性漏洞、系统的安全性保障措施、微软的软件安全性开发生命周期等内容；重点讨论各种安全性测试方法和工具，包括静态代码分析方法（如SAST、DAST、IAST和RASP等）和动态的渗透测试方法，以及基于模型的测试方法、基于故障注入测试方法、形式化测试方法、模糊测试方法等如何应用在安全性测试之上。最后介绍Web应用的安全性测试技术与实践，如XSS、注入式攻击、失效的身份认证等检测技术。

第8章移动App的测试，以移动App的测试项目为背景，完整全面地讨论一个被测系统（两大平台Android、iOS的智能终端应用）的各项软件测试任务，包括移动App的功能测试、性能测试、易用性测试（用户体验测试）、安全性测试等特点及其自动化测试工具，还讨论了移动App的专项测试，如流量测试、耗电量测试等。

第9章基于模型的软件测试，主要介绍什么是基于模型的测试（MBT）方法、有哪些MBT方法以及如何建模等内容，侧重介绍基于业务建模的MBT方法、基于事件流/应用场景建模的MBT方法、基于UML的MBT方法等，并介绍了一些形式化方法，如有限状态机、符号执行、定理证明、模型检验等，较为全面地覆盖了MBT体系。

第10章测试与缺陷管理，在介绍测试计划内容时，重点讨论了测试需求的分析和测试风险的识别，包括如何确定测试项及其优先级。在此基础上对测试工作量、所需的资源和测试进度等进行合理的估算与安排。而缺陷的管理包括缺陷报告、缺陷跟踪和处理、缺陷分析和缺陷管理工具等内容。整个测试管理以测试用例库、缺陷库为核心，力求达到有效性、可追溯性、可视化等，覆盖整个测试过程。

从软件测试教学来看，全书主要章节都视为必需的内容，其中第1~6章和第10章更是最基本的内容。如果本课程只有32学时，就选择这7章内容作为主要讲授内容，其他章节可以作为课外阅读材料。如果有48学时，就可以完成全部10章内容的讲解。软件测试课程和软件工程的许多课程类似，实践性很强，需要学生多动手、多做实验，强调“做中学”。所以，在学时分配上，课堂上需要拿出三分之一的学时来进行教师演示和学生实验，另外，要求学生在课外时间还需要用三分之一的学时去实践，侧重测试工具的运用。总体看，理论教学和实践/实验的时间接近1:1，而且第3、5、6、7、8章是重点实践的章节。

从当前工业界的实践来看，“安全性测试、移动应用测试”越来越受到关注，作为教学，自然也要加强这方面的教学。而第9章“基于模型的软件测试”内容难度相对比较高，如果缺乏相关工具的演示和练习，那么教学会比较抽象，学生不容易掌握。希望授课老师多做些准备，通过具体生动的实例和工具的演示来帮助学生理解该章内

容。关于本书的内容概览、教学指导详见书后所附的思维导图。

本书提倡问题驱动的教学模式，不是为了讲述概念和方法，而是为了解决软件测试中的实际问题。所以，建议授课老师在讲解每章或每个主题时都能摆出问题。例如，在讲第 3 章单元测试时，首先就摆出问题“单元测试为什么很重要？”，可以举一个例子：假如系统有 20 个组件且它们是串联的，如果：

- 每个组件的可靠性很高的话（Re=0.999），系统整体可靠性如何？
- 如果每个组件的可靠性相对比较低的话（Re=0.80），系统整体可靠性如何？

前者是 $0.999^{20} = 0.98$，结果还好，后者是 $0.8^{20}=0.012$，系统可靠性就非常低。从测试方法看，也可以换一个纬度看，提出不同的问题，如“谁来测？测什么？为什么测？如何测？”等基本问题，见下列表中所示。

问题	分析	具体的方法
由谁来测？	开发人员、用户、测试人员、专家等都可以做测试	用户测试、吃自己狗食、α 测试、β 测试、Bug bashes、领域专家测试、结对测试
测什么？	测试的对象或关注点，如业务、功能、接口、数据、路径、状态……	端到端测试、基于需求的直接验证、交互特性测试、面向接口测试、基于输入域的测试、路径覆盖、遍历状态、组合测试等
为什么测？	I/O、功能、性能、安全性等各方面问题的存在	每个 I/O 限制检验、性能测试方法、安全性测试方法等
如何测试？	采用不同的方式、不同的手段等进行测试	基于脚本的测试、探索式测试、手工测试、自动化测试、基于场景的测试、长序列测试、极限测试等

本书各章均提供参考文献作为课外读物，此外还提供了电子教案、关键内容教学视频、测试工具、实验指导及相关文档，尽量帮助大家做好教学，达到良好的教学效果。

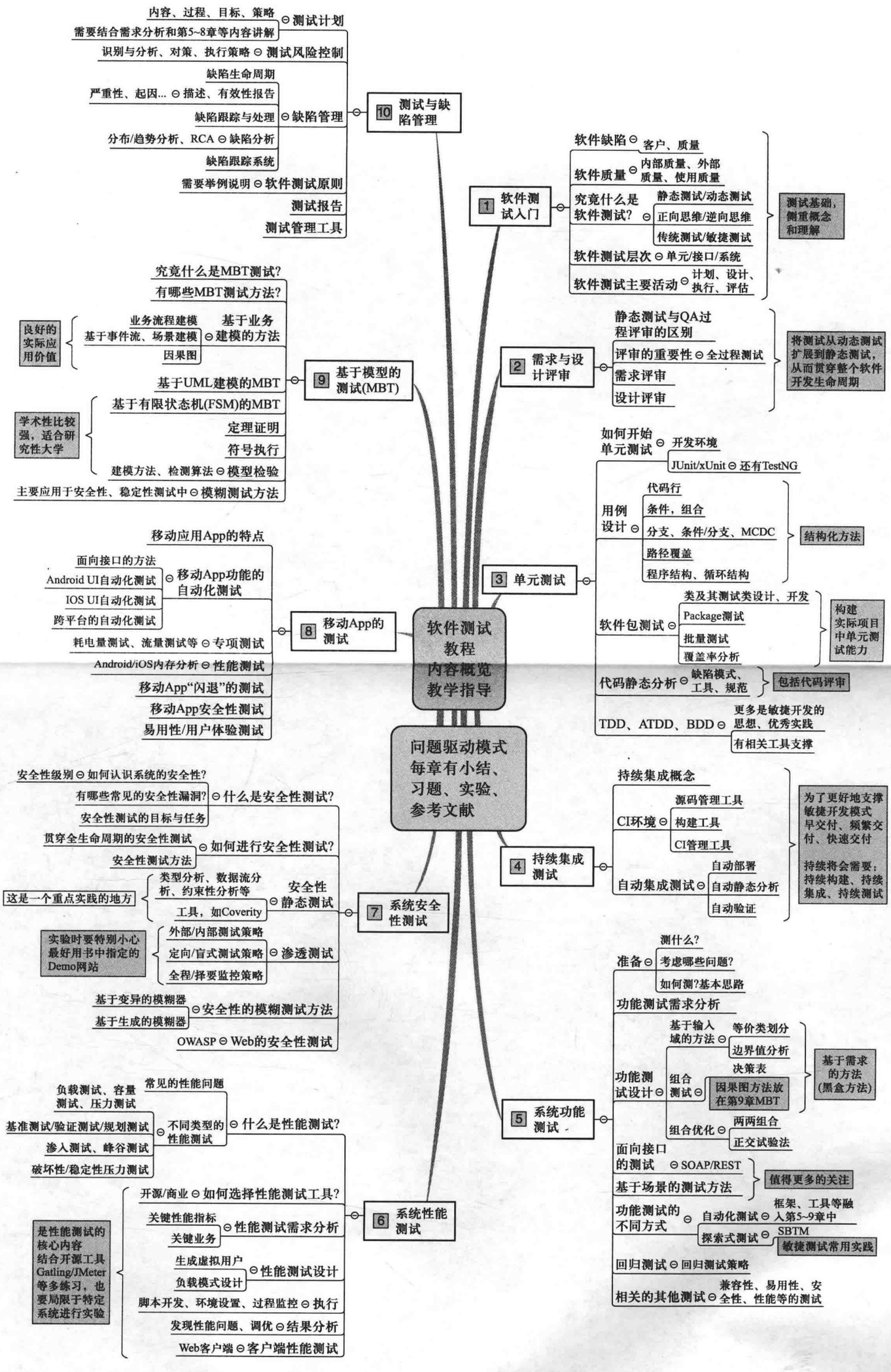
本书能够顺利出版，得到很多同行和朋友的帮助与支持。福建师范大学林岭老师基本独立完成了第 2 章，清华大学刘强老师、天津大学王赞老师、微软总部史亮等在本书写作过程中提出了宝贵意见，还有高等教育出版社编辑的不断推动，在此向他们表示衷心的感谢。

虽然高度认可“建构主义”给教学带来的积极作用，也努力提倡问题驱动式教学方式，但由于时间和能力所限，这方面还做得不够好、不够彻底，希望大家不吝赐教，多多提出宝贵意见（作者 Email 为 kerryzhu@tongji.edu.cn），希望下一版会有更大的改进。

朱少民

于同济大学美丽的校园

本书内容及教学指导概览



目　录

第 1 章　软件测试入门

相信每位拥有智能手机的用户都遇到过“闪退”的问题，即打开手机上某个应用程序（app），其主界面刚显示就退出去了，应用程序一闪而过。实际上，该 App 崩溃（crash）了，根本不能正常运行。为此，腾讯公司还专门开发了一个网站“Bugly”，帮软件开发者解决这类问题。

Bugly 网站

闪退还好，不好用就把该 App 卸载掉，而有些问题就更糟糕了，直接影响着用户的使用体验，甚至产生经济损失。例如：

- 程序卡死在进入界面并且无法关闭，只有重启手机才能解决。而有时想重启手机都没办法，因为重启选项被刚才的操作界面挡住了，如图 1-1 所示。
- 用户在打游戏、正要打赢怪物的关键时刻，游戏突然崩溃，瞬间欲哭无泪。
- 程序编译很久都没问题，到最后几分钟出现错误，直接跳出。
- 某浏览器打开网页时资源占用迅速上升导致电脑卡死，原因是之前安装了一个手势插件。
- 微信客户端在手机开启英文模式后，打飞机游戏模块消失。
- 打开某应用时，启动时间很长，操作时响应也很慢，无法容忍。
- 某手机卫士 App 在点击其“一键加速”功能后，手机桌面上其他 App 的快捷方式都失效了，再也没法启动相应的应用程序。

图 1-1　手机无法重启的界面

- 某影音软件在插有耳机时，播放完第一条广告后卡住，无法跳转至下一条广告，导致无法观看其他节目。
- 填完资料后保存报错，特别是遇到必填项很多的情况；或突然死机，丢失数据。
- 网购时，在线支付，实际上已经扣款了，系统却显示付款失败，用户再付一次

后又被提醒没有完成，导致重复扣款，而卖家却说只收到一次的钱。

凡此种种问题存在，特别是一些质量事故[1]，给用户的生活、工作带来很大影响。因此需要在软件发布前把这些问题找出来，在产品交付给用户后，使其能够愉快、流畅或正常地使用这些软件。这些问题就是我们常说的软件缺陷，为了发现这些缺陷，或通过了解软件中存在的缺陷情况判定产品质量的好坏，就需要进行软件测试。

1.1　什么是软件缺陷？

正如前面所说，软件测试就是为了发现软件产品中的缺陷。要了解测试，首先就要了解“什么是软件缺陷”。前面列出的“闪退”“卡死”“错误提示”“丢失数据”等各种问题，都是缺陷。像这种“数据计算错误、运行出错、功能不能正常运行”等缺陷比较容易理解，还有一些缺陷就不太容易判断了，例如：

- 需求描述不清楚、前后矛盾；
- 设计不合理；
- 用户界面不友好、不美观；
- 操作过程不够流畅；
- 在某种特定条件下没能给出正确或准确的结果；
- 实际结果和预期结果不一致。

所以，判断某种问题、事件或现象是不是缺陷，就会提到另一个概念，即测试的判断准则（test oracle，多数翻译为“测试预言”）。简单地说，测试预言就是判断产品的某个特性是否通过其测试的某种机制或准则（test oracle as a mechanism for determining whether a test has passed or failed）。如果没有通过，就说明这个地方可能存在缺陷，即实际结果和预期结果不一致。测试预言一般可以分为以下几类：


测试预言

① 测试需求文档，如需求规格说明书（specifications）。

② 竞品（competing products）或类似的产品所实现的功能特性。

③ 启发式准则（heuristic oracle），根据多个输入因素（如产品愿景、用户的期望、业务流程等）来提供近似或准确的结果。

④ 统计准则（statistical oracle）：通过统计数据结果或特征（如均值、方差等）做出判断。许多时候人为判断会出现偏差，这就需要借助数据统计分析了解真相。

⑤ 一致性准则（consistency oracle）：相似产品的执行结果比较，如某个家族产品中可比较的各个功能是否一致，产品的当前版本是否与已有版本相一致。