21世纪高职高专计算机专业教材

王爱平 主 编
徐占鹏 李天辉 刘凤玲 副主编

软件测试

清华大学出版社
http://www.tup.com.cn
北京交通大学出版社
http://press.bjtu.edu.cn

21世纪高职高专计算机专业教材

软件测试

王爱平　主编

徐占鹏　李天辉　刘凤玲　副主编

清华大学出版社

北京交通大学出版社

·北京·

内容简介

本书比较全面地介绍了软件测试方法，首先介绍了测试技术的发展历史和现状；然后，作为测试的基础，介绍了白盒测试、黑盒测试及测试覆盖率等几个重要概念，并充分分析了业界在这几个概念方面的研究成果；之后从全流程测试的角度详细介绍了面向对象的测试技术。又从目前实际情况出发，介绍了较为流行的 WEB 测试技术。为了使读者更快地掌握测试技术，第 7 章用一个实例，给出了完整的与软件测试相关的文档。最后，作者总结了测试的基本原则和一些好的实践经验。

本书内容充实，实用性强，可作为高职高专院校计算机软件专业软件测试技术课程的教材，也可作为有关软件测试的培训教材，对从事软件测试实际工作的相关技术人员也具有一定的参考价值。

图书在版编目（CIP）数据

软件测试 / 王爱平主编. —北京：清华大学出版社；北京交通大学出版社，2008.4
（21 世纪高职高专计算机专业教材）
ISBN 978-7-81123-230-1

Ⅰ. 软… Ⅱ. 王… Ⅲ. 软件-测试-高等学校：技术学校-教材 Ⅳ. TP311.5

中国版本图书馆 CIP 数据核字（2008）第 016917 号

责任编辑：谭文芳　　特邀编辑：赵彩云
出版发行：清华大学出版社　　邮编：100084　　电话：010-62776969　　http://www.tup.com.cn
　　北京交通大学出版社　　邮编：100044　　电话：010-51686414　　http://press.bjtu.edu.cn
印 刷 者：北京东光印刷厂
经　　销：全国新华书店
开　　本：185×260　　印张：10.5　　字数：263 千字
版　　次：2008 年 4 月第 1 版　　2008 年 4 月第 1 次印刷
书　　号：ISBN 978-7-81123-230-1/TP·405
印　　数：1～4 000 册　　定价：17.00 元

本书如有质量问题，请向北京交通大学出版社质监组反映。对您的意见和批评，我们表示欢迎和感谢。
投诉电话：010-51686043，51686008；传真：010-62225406；E-mail：press@bjtu.edu.cn。

前　言

信息技术业已成为国家经济发展的支柱产业之一，作为其重要组成部分的软件产业取得了长足的发展，并且越来越广泛地应用于国民经济和国防建设的各个领域。然而，在实际应用中，由于计算机软件缺陷而造成计算机系统故障并导致严重后果的事例屡见不鲜。因此，如何保证软件产品的质量就成了必须解决的一个问题，而对软件进行有效的测试就是解决软件质量问题的方法之一。

软件测试是软件质量保证的关键步骤。软件测试研究的结果表明：软件中存在的问题发现越早，其软件开发费用就越低；在编码后修改软件缺陷的成本是编码前的 10 倍，在产品交付后修改软件缺陷的成本是交付前的 10 倍；软件质量越高，软件发布后的维护费用越低。另据对国际著名 IT 企业的统计，它们的软件测试费用占整个软件工程所有研发费用的 50% 以上。

中国软件企业在软件测试方面与国际水准相比仍存在较大差距。首先，在认识上重开发、轻测试，忽略了如何通过流程改进和软件测试来保证产品或系统的质量，也没有认识到软件项目的如期完成不仅取决于系统设计水平和代码实现能力，而且还取决于设计、代码、文档等各方面的质量。其次，在管理上随意、简单，没有建立规范、有效的软件测试管理体系。另外，缺少自动化工具的支持，大多数企业在软件测试时并没有采用软件测试管理系统。所以对软件企业来说，不仅要提高对软件测试的认识，同时要建立起独立的软件测试组织，采用先进的测试技术，充分运用测试工具，不断改善软件开发流程，建立完善的软件质量保证的管理体系。只有这样，才有可能达到软件开发的预期目标，降低软件开发的成本和风险，提高软件开发的效率和生产力，确保及时地发布高质量的软件产品。

我们将多年来所积累的软件测试经验与技术实践整理成书，与大家共享，希望成为软件测试的实际应用参考书。同时，也将作者在大学软件学院的软件测试专业课、在全国性软件测试和质量保证高级培训班及其他培训班等的授课经验与体会，融入本书之中。

本书参考教学时数为 40～50 学时，全书共分为 7 章：第 1 章讨论了软件测试的一些基本概念；第 2 章介绍了软件开发过程及特征；第 3 章介绍了软件测试的基础知识；第 4 章详细描述了面向对象测试技术；第 5 章讨论了目前较为流行的 Web 信息系统测试技术；第 6 章介绍软件测试的组织与管理；第 7 章通过一个实例，给出了完整的与软件测试相关的文档。本书最后附有软件测试术语。第 1 章和第 3 章由抚顺职业技术学院的王爱平老师编写，第 2 章和附录由抚顺职业技术学院的刘凤玲老师编写，第 4 章和第 5 章由青岛职业技术学院的徐占鹏老师编写，第 6 章和第 7 章由沈阳师范大学职业技术学院的李天辉老师编写。抚顺职业技术学院的张海伟老师参与了本书的编写及校对工作。

本书在编写过程中，参阅了很多国内外同行的著作和文章，汲取了该领域最新的研究成果。在此，对这些成果的作者表示深深的感谢！

由于水平和时间的限制，书中不可避免地会出现一些错误，请各界同仁不吝赐教。

编　者

2008 年 3 月

目　录

第 1 章　软件测试概述

软件测试是软件开发过程的重要组成部分，用来确认一个程序的品质或性能是否符合开发之前所提出的一些要求。软件测试就是在软件投入运行前，对软件需求分析、设计规格说明和编码的最终复审，是软件质量保证的关键步骤。软件测试是为了发现错误而执行程序的过程。软件测试在软件生存期中横跨两个阶段：通常在编写出每一个模块之后就对它作必要的测试（称为单元测试），编码和单元测试属于软件生存期中的同一个阶段；在结束这个阶段后对软件系统还要进行各种综合测试，这是软件生存期的另一个独立阶段，即测试阶段。

1.1　软件错误与缺陷

计算机技术的发展使计算机渗透到人们生活中的各个方面，帮助人们解决了各种难题。人们在欣喜地享受计算机带来的巨大变化的同时，也承受着由于软件错误而产生的灾难。

1.1.1　著名的软件错误案例

1．“爱国者”导弹防御系统

美国“爱国者”导弹防御系统首次应用在海湾战争中对抗伊拉克“飞毛腿”导弹。尽管大家对此导弹系统赞誉有加，但是它在实战中还是出现了失利，其中一枚在沙特阿拉伯的多哈误杀了 28 名美国兵。通过调查分析，专家发现原因是一个软件缺陷。一个很小的系统时钟错误累积起来就可能拖延 14 小时，造成跟踪系统失去准确度。在多哈战中，系统被拖延 100 多个小时。

2．Windows XP 漏洞

随着大家越来越多地使用 Windows XP 系统，其本身的漏洞也越多地暴露出来，例如，浏览器 IE 6.0 的漏洞、Windows XP 内建的“即插即用”功能的漏洞……日本微软甚至在支持技术信息中指出，当用户重新安装、修复及升级 Windows XP 时有可能导致保存在电脑中数据文件丢失……

美国微软公司承认，其最新推出的 Windows XP 操作系统存在巨大安全隐患，Windows XP 的用户只要上网，黑客就可以完全控制电脑，并利用它发动网上攻击行动。Gartner 公司网络安全评定中心已经把这两个漏洞标为高危险级，估计到 2002 年春季末，一名“合格的”黑客就能利用这些漏洞开发出专门针对 Cable Modem（电缆调制解调器）和 DSL（Digital Subscriber Loop，数字用户环路）网络设备的攻击工具。

3．美国航天局火星基地登陆失败

1999 年 12 月 3 日，美国航天局火星基地登陆飞船在试图登陆火星表面时失踪。错误修正委员会观测到故障，并认定出现误动作的原因极可能是某一个数据位被意外更改。大家一

致批评：问题为什么没有在内部测试时解决。

从理论上看，登陆计划是这样的：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，并在预定地点着落；当飞船离地面 1 800 米时，它将丢弃降落伞，点燃登陆推进器，在余下的高度缓缓降落地面。

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”，登陆推进器会一直工作。

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，机械震动在大多数情况下也会触发着地开关，设置错误的数据位。飞船开始着陆时，计算机极有可能关闭登陆推进器，导致飞船下坠 1 800 米之后冲向火星表面，撞成碎片。

结果是悲惨的，但原因很简单。登陆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。前一个小组不去注意着地数据位是否置位，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，但从未相互沟通协调。

1.1.2 软件缺陷是什么

软件错误是在软件设计开发过程中由于理解不到位造成的错误或编写代码时人为引入的错误。软件缺陷是错误的结果，即错误的表现。

符合下边 5 个规则任何一个或数个的叫作软件缺陷：

① 软件未达到产品说明书标明的功能；

② 软件出现了产品说明书指明不会出现的错误；

③ 软件功能超出产品说明书指明范围；

④ 软件未达到产品说明书虽未指出但应达到的目标；

⑤ 软件测试员认为软件难以理解、不易使用、运行速度缓慢或者最终用户认为不好。

软件缺陷是如何产生的呢？通过对大量软件及软件缺陷的研究，人们发现大多数软件缺陷并非来源于编程错误。导致软件缺陷最大的原因是产品说明书；第二大原因是设计方案；第三个原因是代码；第四个原因是某些软件缺陷产生的条件被错误地认定。软件缺陷产生原因分布如图 1-1 所示。

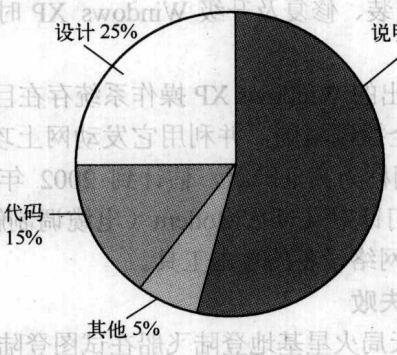

图 1-1 软件缺陷产生原因分布

软件说明书是造成软件缺陷的主要原因。在许多情况下，没有编写软件说明书；也可能是软件说明书不够全面，经常更改，或是各个开发小组没有很好地沟通。

软件缺陷的第二大来源是设计方案。这是程序员开展软件计划的好地方，好比建筑师为建筑物绘制蓝图。这里产生软件缺陷的原因和软件说明书是一样的：片面、易变、沟通不足。

第三个原因是代码。代码错误可以归咎于软件的复杂性、说明文档不足、进度压力或者普通的低级错误。另外，由于代码基本上是由程序员人工编写完成，出现代码错误也是在所难免的。许多看上去是编程错误的软件缺陷实际是因软件说明书或者设计方案造成的。

第四个是某些软件缺陷产生的条件被错误地认定。由于这个原因，就可能反复出现许多软件缺陷。另外，不少软件缺陷可以归咎于测试错误。

按产生的原因可将软件缺陷分为如下类别。

① 文档缺陷：是指对文档的静态检查过程中发现的缺陷，通过测试需求分析、文档审查发现的被分析或被审查的文档中的缺陷。

② 代码缺陷：是指对代码进行同行评审、审计或代码走查过程中发现的缺陷。

③ 测试缺陷：是指在测试执行活动中发现的被测对象（被测对象一般是指可运行的代码、系统，不包括静态测试发现的问题）的缺陷，测试活动类型主要包括内部测试、连接测试、系统集成测试、用户验收测试。

④ 过程缺陷：是指通过过程审计、过程分析、管理评审、质量评估、质量审核等活动发现的关于过程的缺陷和问题。过程缺陷的发现者一般是质量经理、测试经理和管理人员。

表 1-1 显示了对各类软件缺陷的描述。

表 1-1　对各类软件缺陷的描述

缺陷名称	对　象	发现活动	主要发现人
文档缺陷	包括最终产出物和中间产出物文档。具体包括：项目组的文档，如需求文档、设计文档、计划、报告等；测试文档，如测试计划、测试需求分析、测试设计、测试案例、测试分析报告等	同行评审 产品审计	同行评审人员 测试经理
代码缺陷	程序代码：包括程序单元、数据库脚本、配置文件等	同行评审 产品审计 代码走查	同行评审人员 测试经理
测试缺陷	可运行的程序代码、系统、原型等	单元测试 集成测试 系统测试 性能测试等	测试人员
过程缺陷	测试管理体系 测试项目实施过程	过程审计 过程分析 管理评审 质量评估 质量审核等	质量经理、项目经理、管理人员

1.2　什么是软件测试

在软件开发中无论怎样强调软件测试的重要性及它对软件可靠性的影响都不过分。在开发大型软件系统的漫长过程中，面对极其错综复杂的问题，开发者认为正确的却不一定完全符合客观事实，与软件开发工程密切相关的各类人员之间的沟通及配合也不可能没有漏洞，

因此，在软件生命周期的每个阶段都不可避免地会产生差错。虽然开发者在每个阶段结束之前，通过严格的技术审查，力求尽可能早地发现并纠正差错，但是经验表明审查并不能发现所有差错。此外，在编码过程中还不可避免地会引入新的错误。如果在软件投入生产性运行之前，没有发现并纠正软件中的大部分差错，那么这些差错迟早会在生产过程中暴露出来，那时不仅改正这些错误的代价更高，而且往往会造成很严重的后果。因此，这些软件错误需要通过测试来发现。

1.2.1 软件测试定义

软件测试是软件开发过程中的一个重要阶段，是软件质量保证的重要环节。那么软件测试的含义是什么？在软件发展的不同阶段有不同的理解，在软件界也有不同的定义。

1979 年，Glenford J. Myers 对软件测试给出了定义："软件测试是为了发现错误而运行程序的过程。"这一定义明确指出了软件测试的目的是为了发现软件中的错误。1983 年，IEEE （国际电气电子工程师学会）提出了软件测试的定义："测试是使用人工或自动的手段来运行或检测某个系统的过程，其目的在于检验它是否满足约定的需求或是比较预期结果与实际结果之间的差别。"这个定义明确提出了软件测试以检验软件是否满足用户需求为目标。

大量统计资料表明，软件测试的工作量往往占软件开发总工作量的 40%以上。因此，必须高度重视软件测试工作，绝不能以为写出程序之后软件开发工作就接近尾声了，实际上，大约还有同样多的开发工作量需要完成。所以，在软件开发者中间流行这样一句话：用多长时间编写程序，就要用多长时间测试、调试程序。

1.2.2 软件测试的特性

与分析、设计、编码等工作相比，软件测试具有若干特殊的性质。了解这些性质，将有助于正确处理测试中出现的问题，做好测试工作。

1. 挑剔性

软件测试是对软件质量的监督与保证，所以"挑剔"和"揭短"很自然地成为测试人员奉行的信条。测试是为了发现程序中的错误，而不是证明程序无错。因此，对于被测程序就是要"吹毛求疵"，就是要在"鸡蛋里面挑骨头"。只有抱着为发现程序中的错误的目的去测试，才能最大限度地把程序中潜在的错误找出来。

2. 复杂性

人们常常认为开发一个程序是困难的、复杂的，测试一个程序则是比较容易的。其实这是一种误解。设计测试用例就是一项需要细致和高度技巧的工作，稍有不慎就会顾此失彼，产生不应有的疏漏。例如，假设一个程序的功能是输入三角形的 3 条边长，然后判定这个三角形的类别。如果输入 3 个"7"，程序回答"等边三角形"，但若输入 3 个"0"，程序也回答"等边三角形"，就真假不分了。又如，三条边分别为 3、4、5 时应判断是"不等边三角形"，但如果对 2、3、7 也判断为"不等边三角形"，也会出现错误。小程序尚且如此，大型程序就可想而知了。因此，进行软件测试时，要认识到测试的复杂性，需要考虑到各种可能出现的情况，对被测程序进行多方面的考核，切忌把原本复杂的问题想得过于简单，将

测试工作简单化。所以说，做好一个大型软件的测试，其复杂性不亚于对这个软件的开发，应该挑选最有才华的程序员来参加测试工作。

3．不彻底性

“程序测试只能证明错误的存在，但不能证明错误不存在。”E. W. Dijkstra 的这句名言揭示了软件测试所固有的一个重要性质——不彻底性。

软件测试的彻底性就是将程序的所有可能输入作为测试用例对程序进行测试，即穷举测试。穷举测试是否可行呢？现举一个例子说明。假如有一个程序，其功能是输入 X 和 Y，求 X 与 Y 的和，X 与 Y 均是 16 位二进制数。采用穷举测试需要设计 2^{32} 个测试用例，如果 1 秒钟测试一个用例，一天测试 24 小时，将需要 136 年才能测试完毕。由此可见，穷举测试是不可行的。

实际测试中，穷举测试不是根本无法实现，就是工作量太大（如一个小程序要连续测试许多年），在实践上行不通。这就注定了一切实际测试都是不彻底的，当然也就不能保证测试后的程序不存在遗留的错误。

4．经济性

穷举测试行不通，因此在软件测试中，总是选择一些典型的、有代表性的测试用例，进行有限的测试。通常把这种测试称为“选择测试”，为了降低测试成本（一般占整个开发成本的 1/3 左右），选择测试用例时必须遵守“经济性”原则。第一，要根据程序的重要性和一旦发生故障将造成的损失来确定它的可靠性等级，不要随意提高等级，从而增加测试的成本；第二，要认真研究制定好的测试策略，以便能使用尽可能少的测试用例，发现尽可能多的程序错误。

需要明确的是软件测试并不等于程序测试。软件测试贯穿于整个软件开发过程。在需求分析、概要设计、详细设计及程序编码等各阶段所产生的文档，包括需求规格说明、概要设计规格说明、详细设计规格说明和源程序等，都是软件测试的对象。另外，在对需求理解与表达的正确性、设计与表达的正确性、实现的正确性及运行的正确性的验证中，任何一个环节发生了问题都可能在软件测试中表现出来。

1.2.3　测试的目标

软件测试的目标就是发现软件中的错误，但是，发现错误并不是软件测试的最终目的。软件测试的根本目标是开发出完全符合用户需要的软件。G. J. Myers 在《软件测试技巧》中指出了软件测试的目标：

① 测试是为了发现程序中的错误而执行程序的过程；

② 好的测试方案使测试很可能发现尚未发现的错误；

③ 成功的测试是发现了尚未发现的错误的测试。

人类的活动具有高度的目的性，如果测试的目的是要证明程序中有错误，则测试时就会选择一些容易发现错误的测试数据进行测试。与此相反，如果测试的目的是要证明程序中没有错误，在测试时就会选择使程序不容易出现错误的测试数据进行测试。如果认为“没有发现错误就是成功的测试”，则很可能因为没有发现存在的错误而以为测试是成功的，进而使得程序存在错误隐患。

另外，在软件测试时应该注意：

① 测试目标必须在决定发布或接受软件之前，在系统预期的作用（如功能要求、安全要求、业务要求、保密性要求、性能要求等）和系统运行失败的风险之间进行权衡；

② 不同应用场合其测试目标是不同的；

③ 测试目标应该与组织的质量目标一致。

1.2.4 软件测试的原则

现在，我们对软件测试的目的、目标及测试的重要性都有所了解，那么在软件测试时应该遵循哪些原则呢？下面就列出几个重要原则。

1. 要尽早地并且不断地进行测试

由于开发软件的复杂性和抽象性，软件开发各个阶段工作的多变性，以及在参与软件开发各阶段人员之间工作的配合关系等因素，使得开发的每个环节都可能产生错误。所以不应把软件测试仅仅看作是软件开发的某个阶段，而应当把它贯穿到软件开发的始终。坚持在软件开发各个阶段的技术评审，才能在开发过程中尽早发现和预防错误，从而提高软件质量。

2. 测试用例应由测试输入数据及与之对应的预期输出结果两部分组成

测试用例用来检验开发人员所编写的程序，应该在测试程序之前就选择好测试用例，它需要测试的输入数据，且针对每个输入数据要有对应的预期输出结果。从而有一个检验结果的基准，来检验程序的正确性。

3. 程序员应避免检查自己的程序

在某种意义上讲人是不愿意否定自己的，而且在做某一件事情时又可能用同一思路，不难想像带有错误认识的程序员是很难发现自己程序的缺陷的。另外，每个人的测试思路或认识思路都存在局限性，在这个时候，如果选择由别人来测试程序员编写的程序，可能会更客观，处理问题更严谨更有效、更容易取得成功。

4. 设计周密的测试用例

测试用例是测试工作的核心，测试用例设计得好与坏直接关系到测试的质量，测试用例要考虑到合理的输入和不合理的输入，合理的输入条件是指能验证程序正确的输入条件，而不合理的输入条件是指异常的、临界的及可能引起问题异变的输入条件。在测试程序时，可能会过多地考虑合法的和期望的输入条件，以检查程序是否做了它应该做的事情，而忽视了不合法的和预想不到的输入条件。事实上，软件在投入使用以后，用户的多样性、复杂性和所期望的大有差异，有些使用者不遵循或不知道软件的使用约定，而用了一些意外的输入，如用户在键盘上输入了非法的命令或进行数据的增删后忘记存盘等。遇到这种情况时应作出相应处理，给出相应的提示信息。因此，在测试时软件系统处理非法命令的能力也必须受到检验。用不合理的输入条件测试程序往往比用合理的输入条件测试程序能发现更多的错误。

5. 注意测试中错误集中的现象

错误集中的 80%很可能起源于程序模块中的 20%，本着这个原则，在测试过程中若某一部分发现了很多错误时，应该对这一部分进行进一步的测试，以确定其中是否还包含了更多的错误，提高测试投资的效益。

6. 严格执行测试计划，排除测试的随意性

成功者都是有计划的。因此，应该确立一个正确的测试目标，本着严肃、准确的原则，周到细致地做好测试前的准备工作，制订一个比较完善的测试计划。在制订计划时要注意把测试时间安排得尽量宽松，不要希望在极短的时间内完成一个高水平的测试。合理的测试计划应包括：软件功能的测试、输入输出测试、各项测试的进度安排、资源要求、测试工具描述、测试用例的选择、测试的控制方式和过程、系统组装方式、回归测试的规定及评价标准等。在测试工作中，要严格执行测试计划，排除测试的随意性，保证测试工作有序地进行。

7. 对测试错误结果一定要有一个确认的过程

对测试出来的错误，一定要有另一个测试来确认，特别严重的错误要召开评审会进行讨论和分析。

8. 妥善保存测试计划、测试用例、出错统计和最终分析报告

测试工作中产生的各种文档是评价测试工作的重要依据，因此，要妥善保存好这些文档。另外，在测试过程中错误的重现性也是有的，为了给维护提供方便，也要求妥善保留测试的各种文档资料。

1.3 软件质量保证

软件质量是贯穿软件生存期的一个极为重要的问题，是软件开发过程中使用的各种技术和测试方法的最终体现。

1.3.1 软件质量的定义

开发出高质量的软件是软件开发者所追求的目标。何谓软件质量呢？1983 年，IEEE 给出了软件质量的定义：软件系统或软件产品满足规定的和隐含的与需求能力有关的全部特征和特性。它包括以下几方面：

① 软件产品质量满足用户要求的程度；

② 软件各种属性的组合程度；

③ 用户对软件产品的综合反映程度；

④ 软件在使用过程中满足用户要求的程序。

从这个定义中可以看出软件质量包含多层含义：

① 软件能够达到需求说明书中所作的要求；

② 在使用软件过程中能够达到用户所期望的程度；

③ 软件内部各部分之间的组合程度要高。

换句话说软件质量不是单一的，它是软件所有特性的集合。

软件质量有多种不同的方面。用户主要感兴趣的是如何使用软件、软件性能和使用软件的效用，特别是在指定的使用环境下获得与有效性、效率、安全性和满意度有关的规定目标的能力，即使用质量。开发者负责生产出满足质量要求的软件，所以他们对中间产品和最终产品的质量都感兴趣，同时也要体现软件维护者所需要的质量特性。管理者更注重总的质量而不是某一特性，必须从管理的角度，考虑诸如进度拖延或成本超支与提高质量之间的权衡，以达到用有限的人力、成本和时间使软件质量优化的目的。