高等学校电子信息学科“十三五”规划教材·计算机类

软件测试技术

（第二版）

范　勇　兰景英　李绘卓　主编
韩永国　主审

西安电子科技大学出版社
http://www.xduph.com

高等学校电子信息学科“十三五”规划教材·计算机类

软件测试技术

（第二版）

范　勇　兰景英　李绘卓　主编
韩永国　主审

西安电子科技大学出版社

内 容 简 介

本书详尽地阐述了软件测试的基础知识及其相关的测试技术，内容包括软件测试基础、软件测试模型与过程、软件测试管理、黑盒测试、白盒测试、单元测试、集成测试、系统测试、面向对象软件的测试、自动化测试，Web 系统测试案例和软件测试实验。书中通过一个 Web 系统测试案例实践本书所论述的测试理论和技术。

本书内容全面、重点突出、理论简明、难易适中，注重基本概念和基础理论，强调测试技术的实用性。书中结合大量的测试案例，将理论与实践紧密结合，使读者可以更好地理解和掌握软件测试技术，并运用到实际测试工作中去。

本书可作为高等院校、示范性软件学院的计算机相关专业和软件技术专业的教材，也可作为软件测试技术初、中级培训教程，同时可供从事软件开发和软件测试的专业技术人员和管理人员参阅。

图书在版编目(CIP)数据

软件测试技术 / 范勇，兰景英，李绘卓主编. 2 版. —西安：西安电子科技大学出版社，2017.8
(高等学校电子信息学科“十三五”规划教材 • 计算机类)
ISBN 978-7-5606-4655-8

Ⅰ. ① 软… Ⅱ. ① 范… ② 兰… ③ 李… Ⅲ. ① 软件—测试 Ⅳ. ① TP311.55

中国版本图书馆 CIP 数据核字(2017)第 190553 号

策　　划　陈　婷
责任编辑　陈　婷
出版发行　西安电子科技大学出版社（西安市太白南路 2 号）
电　　话　(029)88242885　88201467　邮　编　710071
网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com
经　　销　新华书店
印刷单位　陕西华沐印刷科技有限责任公司
版　　次　2017 年 8 月第 2 版　2017 年 8 月第 3 次印刷
开　　本　787 毫米×1092 毫米　1/16　印张 22.5
字　　数　534 千字
印　　数　7001～10 000 册
定　　价　43.00 元
ISBN 978 – 7 – 5606 – 4655 – 8 / TP
XDUP 4947002-3

前　言

随着软件业的迅猛发展，计算机软件正被广泛地应用到社会的各个领域，软件产品的质量控制与管理正逐渐成为软件企业生存与发展的关键。如何在有竞争力的时间框架内向客户交付质量令人满意的软件产品，以及如何运用新技术来应对软件应用程序日益增长的复杂性等问题越来越受到软件企业、软件用户的关心与重视。由于软件开发周期变短，应用程序的使用和围绕应用程序的技术可能每天都在变化，所以在运作期间必须对应用程序的质量进行监控。软件测试成为了保证软件产品质量控制、管理与检测的重要手段。

软件测试并非是对软件产品的“找错”过程，而是贯穿于软件设计、开发过程的始终，是一个科学的质量控制过程。软件质量是指软件产品中能满足给定需求的各种特性的总和，这些特性称作质量特性。ISO/IEC 9126 中规定了软件的 6 个质量特性，即功能性、可靠性、易用性、效率、维护性和可移植性，每个特性包含若干子特性。由于软件质量特性的复杂性、抽象性及难以度量性等，使得软件测试内容繁多、技术复杂、过程繁杂。软件企业已越来越意识到软件测试的重要性。在微软内部，软件测试人员与软件开发人员的比率一般为 1.5～2.5，这也许出乎了大家对测试人员的理解，但微软软件开发的实践过程已经证明了这种人员结构的合理性。我国的软件企业也逐渐加大了软件测试在整个软件开发系统工程中的比重。近些年来，测试成本的比例更有上升趋势。

纵观国内和国外的软件产业，技术上的差距当然是存在的，但更为明显的也是致命的差距在于产品质量的控制，其中软件测试是重要的一环。为了缩小国内软件测试水平与国际水平的差距，培养专业的软件测试人才，国内许多高校和培训机构都开设了各类软件测试课程。我们总结多年的软件测试技术教学和实践经验编写了本书。全书共包括 12 章。

第 1 章介绍软件质量和软件测试的相关概念，其中包括软件质量保证、软件质量成本以及软件测试的定义、目的、原则、分类和测试用例的相关知识。

第 2 章介绍软件测试模型与过程，分别介绍软件测试中的常见模型：V 模型、W 模型、X 模型、H 模型。

第 3 章介绍软件测试管理，重点介绍软件测试的组织和管理，包括测试小组的组建、测试环境的搭建、被测件的版本管理、测试计划、缺陷管理等相关内容。

第 4 章介绍黑盒测试的主要方法，包括边界值分析、等价类测试、基于判定表的测试、因果图、正交试验设计法、错误推测法等。

第 5 章介绍白盒测试的主要方法，其中包括逻辑覆盖测试、基本路径测试、数据流测试、程序插装、域测试等方法。

第 6 章介绍单元测试的相关知识，重点介绍单元测试的定义、环境、策略、方法、内容以及单元测试的意义，最后通过一个案例来实践单元测试的过程。

第 7 章介绍集成测试的相关知识，重点介绍集成测试的定义、集成测试的各类方法，包括基于功能分解的集成测试、基于调用图的集成、基于路径的集成，最后通过一个拼图游戏案例来实践集成测试的过程。

第 8 章介绍系统测试的相关知识，重点介绍系统测试的定义、内容，以及系统测试的方法。由于 Web 系统的广泛应用以及其软件的特点，本章还介绍了关于 Web系统测试的特有方法。

第 9 章介绍面向对象程序测试的相关知识，回顾了面向对象程序设计的特性：封装、继承、多态，介绍了这些特性对测试的影响，对比了面向对象的测试方法和传统测试方法。本章还介绍了面向对象软件测试的层次以及各个层次的测试方法。

第 10 章介绍自动化测试的背景、自动化测试的技术和常见的自动化测试工具。

第 11 章以一个博客网站为例，简要介绍 Web 系统的测试方法和流程。

第 12 章介绍软件测试实践教学中适合开展的软件测试实验，其中包括：软件测试管理实验、软件缺陷管理实验、单元测试实验、功能测试实验、性能测试实验、Web 安全测试实验和综合实验。教师可根据教学内容选择性地安排实验教学。

本书突出案例教学的特点，注重学生测试实践能力的培养。在阅读本书时，对任何测试技术，不仅要知其然，还要知其所以然。从理论到实践，再从实践回归理论。只有这样，才能更好地领悟到书中所涉及的理论和技术。

感谢刘自伟研究员审校了全书结构，感谢潘娅副教授提出了宝贵意见。感谢西安电子科技大学出版社为本书辛勤付出的所有编辑们。

鉴于作者水平有限，编写时间仓促，书中疏漏之处在所难免，恳请读者批评指正。

编　者

2017 年 5 月

目　录

第 1 章　软件测试基础

随着用户对软件产品质量要求的不断提高以及人们对软件质量的重视程度越来越高，软件测试在软件开发中的地位越来越重要。软件工程的总目标是充分利用有限的人力、物力和财力，高效率、高质量地完成软件开发项目。不足的软件测试势必使软件带着一些未揭露的隐藏错误投入运行，这将意味着让用户承担更大的风险。因此，尽可能多地发现软件中的 Bug，是软件测试工程师的终极目标之一。

故 事

2007 年末，奥运门票销售系统瘫痪事件在中国各大网站成为热点。奥运票务系统瘫痪，错不在“先到先得”的购票政策，而原因主要有两个：一是在设计时没有正确估计系统的访问量，二是对网站没有进行充分的测试，特别是性能测试。该瘫痪事件进一步引发了中国软件企业对软件性能和测试技术的关注，这个事件给软件设计人员带来了关于设计、测试等技术诸多问题的思考。

产品质量下降的结果不但让企业失去竞争力，而且给客户带来了不同程度的经济损失或人身伤害，远到迪斯尼公司耗资上百万的光盘全部回收、巴拿马中心医院数人因医疗事故丧生，近至卡巴斯基的“误杀门”……惨重的教训终于唤醒了软件企业对产品质量的重视和对软件测试技术的研究。

1.1　软 件 质 量

1983 年，ANSI/IEEE STD729 给出了软件质量的定义：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，包括：

(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度。

软件的质量需求，从根本上说是为了引导和满足客户的需求，而软件质量具体表现在软件产品固有的特性方面，如产品的功能性、可靠性、易使用性、效率、可维护性、可移植性和安全性等。对于软件的质量，客户、软件产品开发人员和软件开发企业的认识有不同的侧重点，但必须达到一个平衡点。

从客户角度看，主要关注产品的功能性需求和非功能性需求。功能性需求主要通过各种输入完成用户所需要的各项操作，包括数据的输入和结果的输出。同时对于这些功能的使用，要求易用性要高，界面要友好。对于非功能性需求，主要体现在软件产品的性能、有效性、可靠性等方面，对于不同种类的软件其非功能性需求有很大差异，如实时软件在实时性和可靠性上的要求就非常高。

从软件产品开发人员的角度来看，除客户所关注的性能外，还要关注产品的可维护性、兼容性、可扩展性和可移植性等。

对于软件开发企业来说，除了客户和开发人员所关注的重点外，软件的质量需求更多体现在市场竞争、成本控制等方面。提高软件的质量可以大大降低因质量问题产生的不良成本(如维护成本等)，提高企业的利润。因此，对企业而言，质量需求主要体现在软件的功能性和非功能性需求上，如软件的功能、可维护性、可移植性、可扩展性等。

综上所述，软件质量必须兼顾客户、软件开发人员和软件开发企业对软件质量的需求。一般来说，高质量软件应具备的特性包括：

(1) 满足用户的需求。这是最重要的一点，一个软件如果不能够满足用户的需求，设计得再好，采用的技术再先进，也没有任何意义，即应在软件开发中遵循以用户为中心的原则。

(2) 合理处理进度、成本、功能的关系。一个高质量的软件在开发过程中，项目成员一定能够客观地对待这三个因素，并通过有效的计划、管理、控制，使得三者之间达成一种匹配，保证产出的最大化。

(3) 具备一定的可扩展性和灵活性，能够适应一定程度的需求变化。有变化或变更就会对软件开发产生冲击，所以一个质量优秀的软件，应该能够在一定程度上适应这种变化，并保持软件的稳定性。

(4) 具备一定的可靠性，能够有效处理例外的情况，能够承受各种非法情况的冲击。

(5) 保持成本和性能的平衡。性能往往来源于客户的非功能需求，是软件质量的一个重要的评价因素。但是性能问题在任何地方都存在，所以需要客观地看待它。例如，代码可读性与可靠性之间的平衡。

软件的质量主要由项目和项目管理团队或企业专门负责质量的部门来负责，这就需要他们对项目质量有明确的认识，从而在项目执行过程中按照质量计划让项目朝着预先确定的质量目标前进。为达到软件的高质量目标，质量管理的方法、理念被不断提出、完善和创新。目前流行的软件质量管理有全面质量管理、6δ 管理等。

小贴士

从 1961 年菲根堡姆提出全面质量管理的概念开始，世界各国对它进行了全面深入的研究，使全面质量管理的思想、方法、理论在实践中不断得到应用和发展。概括地讲，全面质量管理的发展经历了以下四个阶段：

(1) 日本从美国引入全面质量管理。1950 年戴明博士在日本开展质量管理讲座，日本人从中学习到了这种全新的质量管理的思想和方法。到 1970 年，质量管理已经逐步渗透到了全日本企业的基层。

(2) 质量管理中广泛采用统计技术和计算机技术。从 20 世纪 70 年代开始，日本企业从质量管理中获得巨大的收益，他们充分认识到了全面质量管理的好处。日本人开始将质量管理当做一门科学来对待，并广泛采用统计技术和计算机技术进行推广和应用。全面质量管理在这一阶段获得了新的发展。

(3) 全面质量管理的内容和要求得到标准化。随着全面质量管理理念的普及，越来越多的企业开始采用这种管理方法。1986 年，国际标准化组织 ISO 把全面质量管理的内容和要求进行了标准化，并于 1987 年 3 月正式颁布了 ISO 9000 系列标准，这是全面质量管理发展的第三个阶段。因此，我们通常所熟悉的 ISO 9000 系列标准实际上是对原来全面质量管理研究成果的标准化。

(4) 质量管理上升到经营管理层面。随着质量管理思想和方法往更高层次发展，企业的生产管理和质量管理被提升到经营管理的层次。无论是学术界还是企业界，很多知名学者，如朱兰、石川馨、久米均等人，都提出了很多有关这个方面的观念和理论，“质量管理是企业经营的生命线”这种观念逐渐被企业所接受。

1.1.1　软件质量保证

质量保证是为了提供信用、证明项目将会达到有关质量标准，而在质量体系中进行的一系列有计划、有组织的工作活动。软件质量保证是由各种任务组成的，这些任务分别与两种不同的参与者紧密相关——进行软件开发的工程师和负责质量保证的计划、监督、记录、分析及报告工作的软件质量保证小组。软件开发工程师通过可靠的技术方法和措施，进行正式的技术评审、执行计划周密的软件测试来考虑质量问题，并保证软件的质量。而软件质量保证小组的职责是辅助软件工程小组得到高质量的最终产品。美国 CMU 大学的软件工程研究所推荐了一组有关质量保证中的计划、监督、记录、分析及报告的质量保证活动。这些活动将由一个独立的质量保证小组(SQA，Software Quality Assurance)来执行：

(1) 为项目准备质量保证计划；

(2) 参与开发该项目的软件过程描述；

(3) 复审各项软件工程活动，对其是否符合已定义好的软件过程进行核实；

(4) 审计软件工作产品，对其是否符合定义好的软件过程中的相关部分进行核实；

(5) 确保软件工作及其工作产品中的偏差已被记录，并按预定流程进行处理；

(6) 记录所有与相关规范或制度不符合的部分，并报告给高级管理者。

软件质量保证的目标是以独立审查方式，从第三方的角度监控软件开发任务的执行，即软件项目是否正遵循已制定的计划、标准和规程，给开发人员和管理层提供反映产品和过程质量的信息和数据，提高项目透明度，同时辅助软件工程组取得高质量的软件产品。软件质量保证的目标主要包括以下四个方面：

(1) 通过监控软件开发过程来保证产品质量；

(2) 保证开发出来的软件和软件开发过程符合相应标准与规程；

(3) 保证软件产品、软件编制过程中存在的与规范或制度不符合的问题得到处理，必要时将问题反映给高级管理者；

(4) 确保项目组制定的计划、标准和规程不仅适合项目组的需要，同时还满足评审和审计的需要。

除了以上四点之外，SQA 最好还能作为软件工程过程小组(SEPG)在项目组中的延伸，能够收集项目中好的实施方法和发现实施不利的原因，为修改企业内部软件开发整体规范提供依据，为其他项目组的开发过程提供先进方法和样例。

软件企业中的 SQA 人员既可以由全职人员担任，也可以由企业内具有相关素质、经过 SQA 培训的人员兼职担任。由此组成的 SQA 小组可能是一个真正的物理上存在的独立部门，也可以是一个逻辑上存在的平台。但不管是真正的独立部门还是逻辑上的平台，它都需要有一个灵魂人物——SQA 小组组长，来组织 SQA 小组的日常活动。

在给一个项目组指派 SQA 人员时，一定要注意一点：指派的 SQA 人员不能是该项目组的开发人员、配置管理人员或测试人员，一个项目的 SQA 除了监控项目过程，完成 SQA 相关工作以外，不应该参与项目组的其他实质性工作，否则他会与项目组捆绑在一起，很难保持客观性。

1.1.2 质量成本

质量成本包含所有质量工作或者进行与质量有关的活动所导致的成本。质量成本可以划分为预防成本、质量评估成本和缺陷修复成本。

预防成本主要包括：

(1) 质量计划；

(2) 技术评审/管理评审；

(3) 测试设备/工具；

(4) 培训。

质量评估成本包括为深入了解各个过程中产品的质量而开展的活动，主要包括：

(1) 过程内和过程间的审查；

(2) 测试设备，工具的维护；

(3) 测试。

缺陷修复成本是指在开发过程中和将产品交付给客户之后修复发现的缺陷所导致的成本，可以进一步划分为内部修复成本和外部修复成本。内部修复成本指产品交付前发现缺陷而引发的成本，主要由返工、修复和失败模式分析等组成。外部修复成本指产品交付给客户后所发现的缺陷带来的相关成本，如因解决客户的抱怨、退换产品、技术支持和维护等而产生的成本。

图 1-1-1 为 Boehm 所收集的数据，发现和修复一个缺陷的成本将随着我们从预防成本到质量评估、从内部修复到外部修复工作的开展而急剧增加。大量统计数据表明，质量成本三部分中，其预防成本所占比例最低，修复缺陷成本最高。因此，为有效降低质量成本，应将更多的精力和关注点放在质量预防上，其次是质量评估上，采用缺陷修复是不得已而为之。因此有人提出了零缺陷管理方法。

质量保证中最有效的办法是预防。预防胜过检查。质量计划、设计和实施是保证项目的关键，不是项目质量出了问题采取检查弥补。预防质量问题的成本要少于纠正质量问题的成本。

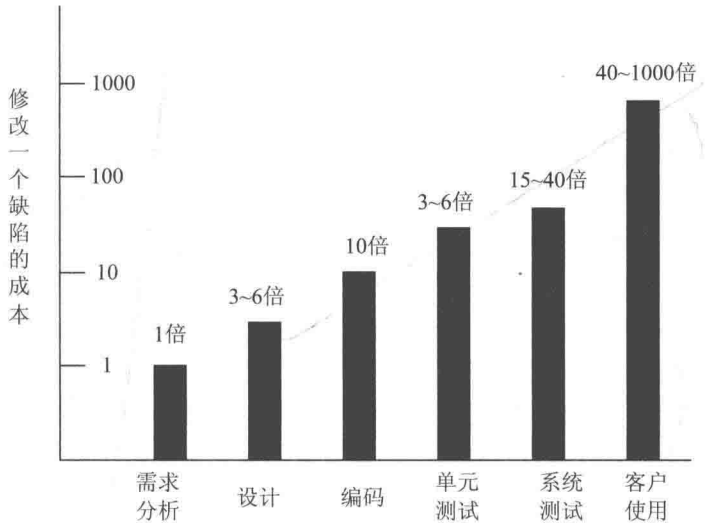

图 1-1-1　改正一个缺陷的相对成本示意图

小贴士

“零缺陷管理”的基本内涵和基本原则，大体可概括为：基于宗旨和目标，通过对经营各环节各层面的全过程全方位管理，保证各环节各层面各要素的缺陷趋向于“零”。其具体要求是：

(1) 所有环节都不得向下一道环节传送有缺陷的决策、信息、物资、技术等；

(2) 每个环节每个层面都必须建立管理制度和规范，按规定程序实施管理，责任落实到位，不允许存在失控的漏洞；

(3) 每个环节每个层面都必须有对产品或工作差错的事先防范和事中修正的措施，保证差错不延续并提前消除；

(4) 在全部要素管理中以人的管理为中心，完善激励与约束机制，充分发挥每个员工的主观能动性，使之不仅是被管理者，而且是管理者，以零缺陷的主体行为保证产品、工作和企业经营的零缺陷；

(5) 整个企业管理系统根据市场要求和企业发展变化及时调整、完善，实现动态平衡，保证管理系统对市场和企业发展有最佳的适应性和最优的应变性。

1.2　软 件 测 试

由于软件缺陷带来的高额修复代价使得人们更注重于规划良好的软件测试，因此软件开发组织将 30%～50%的项目精力花在测试上也就不足为奇，对于那些与人的生命有关的软件(如飞行控制和医疗检测软件)，在测试上所花的时间往往是需求、设计、开发等其他软件工程活动时间之和的 3～5 倍。软件测试就好比工厂的质量检验工作，是对软件产品和阶段性工作成果进行质量检验，力求发现其中的各种缺陷，并督促修正缺陷，从而控制和保证软件产品的质量。因此，软件测试是软件公司提高软件产品质量的重要手段之一。

小贴士

软件测试与质量保证的关系：

规范的软件测试活动一般包括测试计划创建、测试用例设计、执行测试、更新测试文档等；而软件质量保证的活动主要有协调度量、风险管理、文档检查、促进/协助流程改进、监察测试工作。软件质量保证(SQA)的职能是向管理层提供正确的可视化的信息，从而促进与协助流程改进。SQA 还充当测试工作的指导者和监督者，帮助软件测试建立质量标准、测试过程评审方法和测试流程，同时通过跟踪、审计和评审，及时发现软件测试过程中的问题，从而帮助改进测试或整个开发的流程等，因此有了 SQA，测试工作就可以被客观地检查与评价，同时也可以协助测试流程的改进。而测试为 SQA 提供数据和依据，帮助 SQA 更好地了解质量计划的执行情况、过程质量、产品质量和过程改进进展，从而使 SQA 更好地做好下一步工作。

二者相同点：都是贯穿整个软件开发生命周期的。

二者不同点：SQA 侧重对流程中各过程的管理与控制，是一项管理工作，侧重于流程和方法。而测试是对流程中各过程管理与控制策略的具体执行与实施，其对象是软件产品(包括阶段性的产品)，即测试是对软件产品的检验，是一项技术性的工作。测试，常常被认为是质量控制的最主要手段。

1.2.1　软件测试的定义

1．软件测试

1979 年，G.J.Myers 对软件测试的定义：程序测试是为了发现错误而执行程序的过程。

1983 年，IEEE 对软件测试的定义：使用人工或者自动的手段来运行或测定某个系统的过程，其目的在于检验它是否满足规定的需求或者是弄清预期结果与实际运行结果之间的差别。

1983 年，B.hetzel 对软件测试的定义：以评价一个程序或系统的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，测试的定义：使用人工或者自动手段来运行或测试被测试件的过程，其目的在于检验它是否满足规定的需求并弄清预期结果与实际结果之间的差别。它是帮助识别开发完成(中间或最终版本)的计算机软件(整体或部分)的正确度(correctness)、完全度(completeness)和质量(quality)的软件过程。

从上面的定义可以看出，软件测试的内涵在不断丰富，对软件测试的认识在不断深入。要完整理解软件测试，就要从不同角度去审视。软件测试就是对软件产品进行验证和确认的活动过程，其目的就是尽快尽早地发现软件产品在整个开发生命周期中存在的各种缺陷，以评估软件的质量是否达到可发布水平。软件测试是软件质量保证的关键元素，代表了需求规格说明书、设计和编码的最终检查。

2. 软件测试的狭义观点

G.J.Myers 在《软件测试之艺术》(The Art of Software Testing)给出的软件测试定义，是传统意义上的测试定义，即在代码完成后通过运行程序或软件来查找程序代码或者软件系统中的错误。这种传统意义上的测试主要是受软件开发瀑布模型的影响，而且非常不利于保证软件的质量，主要原因是这种测试不能在代码完成前发现软件系统在需求、设计等上的缺陷，图 1-1-1 的统计表明这将导致后期的软件质量成本很高。

3. 软件测试的广义观点

为了尽早发现问题，降低软件质量成本，可将传统的软件测试范围延伸到需求评审、设计评审、代码评审等活动中。根据广义观点，软件测试可分为静态测试和动态测试。静态测试主要的活动是评审，即通过对需求、设计、代码和其他软件开发文档的评审来检验相应的内容是否满足用户的需求，由于静态测试不需要运行软件或程序，故具有静态特性特征。动态测试通过运行软件或程序来发现存在的问题，由于是在运行过程中发现问题，故具有动态性特征。

4. 软件测试的辩证统一观点

G.J.Myers 给出的软件测试定义，被软件测试业界认可，并经常被引用，但属于软件测试的狭义范畴。后来 G.J.Myers 进一步提出了程序测试的 3 个重要观点：

(1) 测试是为了证明程序有错，而不是证明程序无错；

(2) 一个好的测试用例在于它发现至今没有发现的错误；

(3) 一个成功的测试是发现了至今未发现的错误的测试。

从质量保证观点来看软件测试，就是证明或者验证软件的功能特性和非功能特性满足用户的需求，主要是针对软件的所有功能点逐一验证其正确性，对于非功能点要满足用户的要求；从软件测试的目标和降低测试成本等方面来看，就是尽早尽快地发现更多的软件缺陷，主要采取试图破坏系统、摧毁系统等手段，发现系统中存在的各种缺陷。软件测试就是在这两者之间获得平衡，但对于不同行业领域，两者的比重是不一样的。对于航天、电信计费系统等要求有很高的软件质量，特别是系统的高可靠性，而一般的应用软件或服务软件，其质量目标一般设置在用户可接受的水平，以降低软件开发成本，加快软件的发布速度。

从软件测试的辩证统一观点来看，对不同的软件产品，应制定相应的可发布的质量标准，以评估软件是否可发布。

5. 软件测试的经济成本观点

“一个好的测试用例在于发现至今未发现的错误”，这体现了软件测试的经济成本观点。软件测试的成本一直是业界关注的问题之一。根据辩证统一观点来看，不充分的测试是不负责任的；过分的测试是一种资源的浪费，同样也是一种不负责任的表现。在实际操作中的困难在于：如何界定什么样的测试是不充分的，什么样的测试是过分的。在目前软件测试技术状况下，唯一可用的答案是：制定最低测试通过标准和测试内容，然后具体问题具体分析。对于相对复杂的产品或系统来说，零缺陷是一种理想，应在测试成本范围内进行更充分的测试和更全面的质量评估。

1.2.2　软件测试的目的

软件测试的目标，概括地说，就是尽快尽早地将被测件中所存在的缺陷找出来，并促进系统分析工程师、设计工程师和程序员等尽快地解决这些缺陷，并评估被测试件的质量水平。软件测试是软件质量保证过程中的重要一环，同时也是软件质量控制的重要手段之一，测试工程师与整个项目团队共同努力，确保按时向客户提交满足客户要求的高质量的软件产品。

软件测试的目标之一是尽快尽早地找到至今没有被发现的缺陷，而不是确保没有缺陷。主要原因有：

(1) 测试的覆盖率几乎不可能达到100%，即软件测试不可能穷举所有的测试用例，不能将程序中的所有路径测试一遍；

(2) 去除现有的缺陷可能会产生新的缺陷，同时系统的需求总是不断在变化，这种需求的不稳定性也将带来新的缺陷；

(3) 测试工程师对产品的理解不能完全代表用户的理解，由于两者之间的差异，故意味着可能存在测试工程师没有发现的缺陷；

(4) 测试的模拟环境不能完全代表用户的实际使用环境，由于两者之间的差异，故意味着可能存在测试工程师没有发现的缺陷。

软件测试的另外一个重要的目标是评估软件是否达到可发布水平，即何时停止软件测试。

每当讨论软件测试技术的时候都会引发一个经典问题的讨论："我们怎么知道我们的测试已经足够了呢？"。遗憾的是，到目前为止这个问题还没有一个非常明确的答案，但还是有一些基于实践经验的答案。通过在软件测试过程中收集的数据，利用现有的统计理论和可靠性模型，就可能得到"测试什么时候完成"这种问题有意义的指导原则。根据这些指导原则，不同的软件企业根据项目的特征指定了软件的可发布标准，对软件的可发布性进行了有益的探索。

1.2.3　软件测试的原则

软件测试从不同的角度出发会派生出两种不同的测试原则。从用户的角度出发，通过软件测试能充分暴露软件中存在的问题和缺陷，从而考虑是否可以接受该产品；从开发者的角度出发，就是希望测试能表明软件已经正确地实现了用户的需求，达到软件正式发布的要求，以确立人们对软件质量的信心。

根据软件测试的广义观点来看，软件测试不是仅对源程序进行测试，开发各阶段得到的文档包括需求规格说明书、概要设计说明书、详细设计说明书等都是软件测试的对象。因此，在软件测试中应力求遵循以下原则：

(1) 可追溯性。所有的测试都应追溯到用户需求。软件测试揭示软件的缺陷，一旦修正这些缺陷就能更好地满足用户需求；如果软件实现的功能不是用户所期望的，将导致软件测试和软件开发做了无用功，这种情况在软件开发和软件测试中时有发生。

(2) 尽早开展预防性测试。测试工作进行得越早，越有利于提高软件的质量和降低软

件的质量成本，这是预防性测试的基本原则。由于软件的复杂性和抽象性，在软件生命周期各阶段都可能产生错误，所以不应把软件测试仅仅看做是软件开发的一个独立阶段，而应当把它贯穿到软件开发的各个阶段中。在需求分析和设计阶段就应开始进行测试工作，这样才能尽早发现和预防错误，杜绝某些缺陷和错误，尽量避免将软件缺陷遗留到下一个开发阶段，提高软件质量。

(3) 投入/产出原则。根据软件测试的经济成本观点，在有限的时间和资源下进行完全测试找出软件所有的错误和缺陷是不可能的，也是软件开发成本所不允许的，因此软件测试不能无限进行下去，应适时终止。即不充分的测试是不负责任的；过分的测试是一种资源的浪费，同样也是一种不负责任的表现。因此在满足软件预期的质量标准时，应确定质量的投入/产出比。

(4) 回归测试。由于修改了原来的缺陷，将可能导致新的缺陷产生。因此修改缺陷后，应集中对软件的可能受影响的模块/子系统进行回归测试，以确保修改缺陷后不引入新的软件缺陷。

(5) 80/20 原则。测试实践表明，系统中 80%左右的缺陷主要来自 20%左右的模块/子系统，因此应当花较多的时间和代价测试那些具有更多缺陷数目的程序模块/子系统。

(6) 设立独立的测试机构或委托第三方测试。由于思维定势和心理因素等原因，开发工程师难以发现自己的错误，同时揭露自己程序中的错误也是件非常困难的事。因此，测试一般由独立的测试部门或第三方机构进行，但需要软件开发工程师的积极参与。

1.3 软 件 缺 陷

1.3.1 软件缺陷的定义

软件缺陷是软件产品预期属性的偏离现象，它包括检测缺陷和残留缺陷。检测缺陷(Detected Defect)是指软件在进入用户使用之前被检测出的缺陷。残留缺陷(Residual Defect)是指软件发布后存在的缺陷，包括在用户安装前未被检测出的缺陷和已被发现但还未被修复的缺陷。

软件故障(Software Failure)是指用户使用软件时，由于残留缺陷引起的软件失效症状。

不要将软件缺陷和软件错误两个概念混淆起来。软件缺陷的范围更广，它涵盖了软件错误、不一致性问题、功能需求定义缺陷和产品设计缺陷等。软件错误仅是软件缺陷的一种，即程序或系统的内部缺陷，通常是软件代码本身的问题，如算法错误、语法错误、内存泄漏、数据溢出等。软件错误必须被修正，但软件缺陷不一定被修正。

术 语

缺陷类型(Type)：根据缺陷的自然属性划分的缺陷种类。
缺陷严重程度(Severity)：因缺陷引起的故障对软件产品的影响程度。
缺陷优先级(Priority)：缺陷必须被修复的紧急程度。
缺陷状态(Status)：缺陷通过一个跟踪修复过程的进展情况。