HZ BOOKS
华章教育

高等院校计算机课程案例教程系列

本书为教师配有电子教案

软件测试案例教程

吕云翔 王洋 肖咚 编著

- 针对不同的测试方法和技术选用不同的案例
- 介绍当前常见的软件测试工具

机械工业出版社
China Machine Press

高等院校计算机课程案例教程系列

软件测试案例教程

吕云翔 王洋 肖咚 编著

机械工业出版社
China Machine Press

本书以案例驱动，讲述了软件测试的相关概念、方法和技能。全书分为四个部分：基础篇、方法篇、策略篇和工具篇。基础篇讲述了软件测试的基础理论，为后面的学习奠定了一定的理论基础；方法篇通过案例“C0 编译器”讲述了软件测试中常用的黑盒测试和白盒测试技术的使用；策略篇分别通过案例“聚合文件管理工具”和“交互式实验室资源管理与服务网站”讲述了传统软件测试和面向对象软件测试的策略；工具篇讲述了自动化测试和常见工具，并选取了两个常用软件测试工具讲述其使用方法。每章的开始部分有本章要点，列出了章节中的重要内容，方便读者自学和教学选择；每章的结尾部分都附有练习题，供读者检验学习成果。

本书重视实践能力和操作能力的培养，并在案例讲述过程中穿插相关的基础知识和基本理论介绍，做到理论与实践相结合，方法与应用相结合。本书适合高等院校计算机、软件工程、测试等相关专业本科生作为教材学习，同时也可作为社会人员自学使用。

图书在版编目(CIP)数据

软件测试案例教程/吕云翔，王洋，肖咚编著. —北京：机械工业出版社，2010.10
（高等院校计算机课程案例教程系列）

ISBN 978-7-111-32099-9

Ⅰ.软…　Ⅱ.①吕…②王…③肖…　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（2010）第 193371 号

机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码　100037）
责任编辑：张少波
三河市明辉印装有限公司印刷

2011 年 1 月第 1 版第 1 次印刷
185mm×260mm · 13.5 印张
标准书号：ISBN 978-7-111-32099-9
定价：25.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：(010)88378991；88361066
购书热线：(010)68326294；88379649；68995259
投稿热线：(010)88379604
读者信箱：hzjsj@hzbook.com

前言

为了振兴中国的计算机和软件产业，培养具备软件工程思想和技术，并具有相应开发经验的人才，国家近年来一直十分重视软件工程相关课程的建设和人才培养。除了开设专门的软件工程专业，还倡导在计算机科学技术相关专业开设软件工程课程，使得软件工程思想和技术在中国的IT人才中得到普及。软件测试是软件工程中重要的组成部分，对保证软件质量具有重要意义。

本书以案例为驱动，讲述了软件测试的相关概念、方法和技能。全书分为四个部分：基础篇、方法篇、策略篇和工具篇。基础篇讲述了软件测试的基础理论，为后面的学习奠定了一定的理论基础；方法篇通过案例“C0编译器”讲述了软件测试中常用的黑盒测试和白盒测试技术的使用；策略篇分别通过案例“聚合文件管理工具”和“交互式实验室资源管理与服务网站”讲述了传统软件测试和面向对象软件测试的策略；工具篇讲述了自动化测试和常见工具，并选取了两个常用的软件测试工具讲述其使用方法。每章的开始部分有本章要点，列出了章节中的重要内容，方便读者自学和教学选择；每章的结尾部分都附有练习题，供读者检验学习成果。

本书重视实践能力和操作能力的培养，并在案例讲述过程中穿插相关的基础知识和基本理论介绍，做到理论与实践相结合，方法与应用相结合。本书适合高等院校计算机、软件工程、测试等相关专业本科生作为教材使用，同时也可作为社会人员自学使用。

相比于软件测试相关的同类教材，本书具有以下特色：

- 循序渐进。本书将内容分为“基础篇”、“方法篇”、“策略篇”和“工具篇”四个部分，层次分明，便于循序渐进地讲述知识，便于读者学习与理解。
- 实用性强。本书选用三个案例贯穿全书，做到案例驱动；又对不同的测试方法和技术选用不同的案例，做到有所针对；同时介绍了工具使用和文档撰写，具有很强的实用性。
- 理论结合。本书在案例介绍、工具介绍过程中穿插相关的理论知识和基本方法，使基础知识更具体形象，同时也更容易被理解和应用。
- 实时性强。本书所选案例均是近年来的真实案例，可以代表当代技术特征和需求环境；本书介绍的工具均是当前常见的软件测试工具；面向对象测试策略的内容符合软件测试技术的发展方向。

本书作者一直在北京航空航天大学（简称北航）软件学院担任教学工作，进行了大量的教学探索和研究。在此感谢北航软件学院在成书过程中提供的各种宝贵资料和建议。

由于能力和水平有限，书中难免存在疏漏和错误之处，恳请各位同仁和广大读者给予批评指正，也希望各位能将教学和学习过程中的经验和心得与我们交流。

作者联系方式：yunxianglu@ hotmail. com

教学建议

教学内容	学习要点及教学要求	课时安排
第1章 软件测试概述	• 软件定义和软件的分类 • 软件工程和瀑布过程模型 • 软件质量要素 • 软件可靠性和软件错误的概念 • 软件测试的定义和目的 • 软件测试的原则 • 软件测试的分类 • 常见的软件测试过程模型 • 测试用例的编写规范和设计原则	2
第2章 软件测试实施和管理	• 测试计划的内容和编写步骤 • 测试用例的实施和跟踪 • 测试用例的管理 • 测试报告与测试评估 • 测试文档的主要内容 • 测试团队的组织和建设 • 测试人员的角色和职责 • 测试管理相关知识	2
第3章 “C0编译器”案例概述	• 了解项目背景即编译原理的相关知识 • 了解C0语言和目标语言的定义 • 了解案例项目“C0编译器”的结构和实现	2
第4章 黑盒测试	• 了解黑盒测试的基本概念 • 通过案例掌握等价类划分法及其应用 • 通过案例掌握边界值分析法及其应用 • 通过案例掌握因果图法及其应用 • 通过案例掌握决策表法及其应用 • 通过案例掌握场景法及其应用 • 了解不同黑盒测试方法的优缺点和应用场合	4
第5章 白盒测试	• 了解白盒测试的基本概念 • 通过案例掌握代码检查法及其应用 • 通过案例掌握静态结构分析法及其应用 • 通过案例掌握程序插桩技术及其应用 • 通过案例掌握逻辑覆盖法及其应用 • 通过案例掌握基本路径法及其应用 • 了解不同白盒测试方法的优缺点和应用场合 • 了解白盒测试和黑盒测试的区别与联系	4
第6章 传统软件测试策略	• 了解传统软件测试策略包含的阶段和各阶段的内容 • 熟悉单元测试的概念和原则 • 熟悉单元测试关注的几个方面 • 了解单元测试环境建立和使用的主要方法 • 熟悉集成测试的概念和原则 • 熟悉集成测试分析的内容 • 了解几种常见的集成测试策略 • 熟悉系统测试的概念 • 了解常见的系统测试方法 • 熟悉验收测试的概念和分类 • 了解验收测试的主要内容 • 了解回归测试的相关概念	4

（续）

教学内容	学习要点及教学要求	课时安排
第7章 “聚合文件管理工具”测试	• 了解“聚合文件管理工具”的功能 • 了解“聚合文件管理工具”的概要和详细设计 • 通过“聚合文件管理工具”案例熟悉软件测试计划 • 通过“聚合文件管理工具”案例熟悉传统软件测试策略和阶段 • 通过“聚合文件管理工具”案例熟悉测试用例编写 • 通过“聚合文件管理工具”案例熟悉测试总结分析报告编写	4
第8章 面向对象软件测试策略	• 熟悉面向对象的软件工程的基本概念和特征 • 了解面向对象软件测试策略 • 了解面向对象软件测试常用工具 • 熟悉面向对象软件测试用例设计方法 • 了解网站测试的概念与过程 • 了解软件测试的基本内容	4
第9章 “交互式实验室资源管理与服务网站”测试	• 了解“交互式实验室资源管理与服务网站”案例的项目背景 • 了解“交互式实验室资源管理与服务网站”案例的主要需求与设计 • 通过“交互式实验室资源管理与服务网站”案例掌握面向对象的软件测试计划编写 • 通过“交互式实验室资源管理与服务网站”案例掌握面向对象的类测试 • 通过“交互式实验室资源管理与服务网站”案例掌握面向对象的交互测试 • 通过“交互式实验室资源管理与服务网站”案例掌握面向对象的确认测试 • 通过“交互式实验室资源管理与服务网站”案例掌握面向对象的系统测试	4
第10章 软件测试自动化	• 了解软件测试自动化的概念和优缺点 • 熟悉软件测试自动化的实施过程 • 熟悉常见软件测试工具分类和工具代表 • 了解常见的几款软件测试工具	2
第11章 单元测试工具 Unit Test	• 熟悉单元测试工具 Unit Test 的基本功能 • 熟悉单元测试工具 Unit Test 的基本使用方法 • 能够利用单元测试工具 Unit Test 进行实际代码的单元测试	2
第12章 负载测试工具 LoadRunner	• 熟悉 LoadRunner 的背景和基本功能 • 熟悉 LoadRunner 的主要特性、组件和相关术语 • 熟悉 LoadRunner 的安装过程 • 能够利用 LoadRunner 进行负载测试	2
	教学总学时建议	36

说明：

① 本课程建议主要针对计算机类本科生教学，累计为36学时，不同专业根据不同的教学要求和计划教学时数可酌情对教材内容进行适当取舍。

② 非计算机类本科专业使用本教材可适当降低教学要求。

③ 本教材理论授课学时数36学时，包含案例讲授、课堂讨论、练习等必要的课内教学环节。

④ 建议授课时间比例为：基础部分50%，案例部分50%。

目 录

前言
教学建议

第一部分 基础篇

第 1 章 软件测试概述 …… 2
1.1 软件测试背景 …… 2
1.1.1 软件 …… 2
1.1.2 软件工程 …… 3
1.1.3 软件质量 …… 4
1.1.4 软件可靠性和软件错误 …… 5
1.2 软件测试基本概念 …… 6
1.2.1 软件测试的目的 …… 6
1.2.2 软件测试的原则 …… 7
1.2.3 软件测试的分类 …… 8
1.2.4 软件测试过程模型 …… 10
1.3 测试用例 …… 11
1.3.1 测试用例编写 …… 12
1.3.2 测试用例设计 …… 13
1.4 小结 …… 13
1.5 本章习题 …… 14
第 2 章 软件测试实施和管理 …… 15
2.1 软件测试实施 …… 15
2.1.1 软件测试计划 …… 15
2.1.2 测试用例实施和管理 …… 16
2.1.3 测试报告与测试评估 …… 18
2.1.4 软件测试文档 …… 20
2.2 测试团队和人员 …… 21
2.2.1 软件测试团队 …… 21
2.2.2 软件测试人员 …… 22
2.3 软件测试管理 …… 23
2.4 小结 …… 24
2.5 本章习题 …… 25

第二部分 方法篇

第 3 章 “C0 编译器”案例概述 …… 28
3.1 编译原理简介 …… 28
3.2 C0 语言和目标代码定义 …… 30
3.3 “C0 编译器”程序结构 …… 32
3.4 小结 …… 35
3.5 本章习题 …… 35
第 4 章 黑盒测试 …… 36
4.1 等价类划分法 …… 36
4.2 边界值分析法 …… 40
4.3 因果图法 …… 41
4.4 决策表法 …… 44
4.5 场景法 …… 45
4.6 黑盒测试方法选择 …… 46
4.7 小结 …… 46
4.8 本章习题 …… 47
第 5 章 白盒测试 …… 48
5.1 代码检查法 …… 48
5.2 静态结构分析法 …… 59
5.3 程序插桩技术 …… 61
5.4 逻辑覆盖法 …… 64
5.5 基本路径法 …… 69
5.6 白盒测试方法选择 …… 73
5.7 白盒测试和黑盒测试比较 …… 74
5.8 小结 …… 74
5.9 本章习题 …… 75

第三部分 策略篇

第 6 章 传统软件测试策略 …… 78
6.1 单元测试 …… 78
6.1.1 单元测试概述 …… 78
6.1.2 单元测试内容 …… 79
6.1.3 单元测试方法 …… 81
6.2 集成测试 …… 82
6.2.1 集成测试概述 …… 82
6.2.2 集成测试分析 …… 82
6.2.3 集成测试策略 …… 84
6.3 系统测试 …… 85
6.3.1 系统测试概述 …… 85
6.3.2 系统测试方法 …… 85
6.4 验收测试 …… 87
6.4.1 验收测试概述 …… 87
6.4.2 验收测试内容 …… 88

6.5 回归测试 …… 90
6.6 小结 …… 91
6.7 本章习题 …… 91
第7章 “聚合文件管理工具”测试 …… 92
7.1 案例概述 …… 92
7.2 测试计划 …… 94
7.3 测试用例 …… 99
7.3.1 单元测试用例 …… 99
7.3.2 功能测试用例 …… 102
7.4 测试报告和分析 …… 105
7.5 小结 …… 108
7.6 本章习题 …… 108
第8章 面向对象软件测试策略 …… 109
8.1 面向对象的基本特征 …… 109
8.2 面向对象软件的测试策略 …… 112
8.2.1 面向对象的单元测试 …… 112
8.2.2 面向对象的集成测试 …… 113
8.2.3 面向对象的系统测试 …… 114
8.2.4 面向对象系统的回归测试 …… 114
8.2.5 面向对象测试的相关模型 …… 115
8.3 面向对象软件的测试用例设计 …… 116
8.3.1 面向对象测试用例设计的基本概念 …… 117
8.3.2 面向对象编程对测试的影响 …… 117
8.3.3 基于故障的测试 …… 118
8.3.4 基于场景的测试 …… 118
8.3.5 表层结构和深层结构的测试 …… 118
8.4 网站测试 …… 119
8.4.1 网站测试概念 …… 119
8.4.2 网站测试过程 …… 120
8.4.3 数据库测试 …… 122
8.4.4 用户界面测试 …… 123
8.4.5 构件级测试 …… 124
8.4.6 配置测试 …… 124
8.4.7 安全性测试 …… 125
8.4.8 系统测试 …… 126
8.5 小结 …… 127
8.6 本章习题 …… 128
第9章 “交互式实验室资源管理与服务网站”测试 …… 129
9.1 案例概述 …… 129
9.1.1 项目背景 …… 129
9.1.2 项目目标 …… 129
9.1.3 系统功能性需求 …… 129
9.1.4 系统总体设计及实现 …… 132
9.1.5 系统环境 …… 137
9.1.6 条件与限制 …… 138
9.2 项目测试计划 …… 138
9.3 测试过程 …… 140
9.3.1 类测试 …… 141
9.3.2 交互测试 …… 151
9.3.3 确认测试 …… 153
9.3.4 系统测试 …… 155
9.4 测试报告和分析 …… 156
9.5 小结 …… 158
9.6 本章习题 …… 159

第四部分 工具篇

第10章 软件测试自动化 …… 162
10.1 软件测试自动化概述 …… 162
10.1.1 软件测试自动化优缺点 …… 162
10.1.2 软件测试自动化实施过程 …… 163
10.2 软件测试工具分类 …… 164
10.3 常用工具介绍 …… 165
10.3.1 功能测试工具 WinRunner …… 165
10.3.2 黑盒测试工具 QACenter …… 166
10.3.3 白盒测试工具 Logiscope …… 170
10.3.4 测试管理工具 TestDirector …… 171
10.4 小结 …… 172
10.5 本章习题 …… 173
第11章 单元测试工具 Unit Test …… 174
11.1 Unit Test 功能介绍 …… 174
11.2 Unit Test 使用流程 …… 177
11.3 小结 …… 181
11.4 本章习题 …… 181
第12章 负载测试工具 LoadRunner …… 182
12.1 LoadRunner 概述 …… 182
12.2 LoadRunner 主要特征 …… 183
12.3 LoadRunner 组件和术语 …… 184
12.4 LoadRunner 安装 …… 185

12.5 使用 LoadRunner 对 Web 应用进行负载/压力测试 …… 187
12.5.1 制定负载测试计划 …… 188
12.5.2 开发测试脚本 …… 189
12.5.3 创建运行场景 …… 193
12.5.4 运行测试场景 …… 197
12.5.5 监视场景 …… 199
12.5.6 分析测试结果 …… 200
12.6 小结 …… 202
12.7 本章习题 …… 203
参考文献 …… 204

第一部分

基 础 篇

软件测试是软件工程中重要的组成部分，对保证软件质量具有重要意义。本部分介绍软件测试的基础知识内容，包括软件测试的概念、原则、分类、测试管理等内容，为后面的学习奠定一定的理论基础。本部分内容比较简略，很多内容没有进行详细的阐述和展开，只需要读者对软件测试和相关知识有一个大概的了解，然后在接下来的章节中结合各个案例进行更深入全面的学习和实践。

第1章

软件测试概述

本章要点

- 软件定义和软件的分类
- 软件工程和瀑布过程模型
- 软件质量要素
- 软件可靠性和软件错误的概念
- 软件测试的定义和目的
- 软件测试的原则
- 软件测试的分类
- 常见的软件测试过程模型
- 测试用例的编写规范和设计原则

1.1 软件测试背景

作为软件工程中一个重要组成部分，软件测试是保证软件质量的一个关键步骤。要了解软件测试，首先要了解其所在的背景和所针对的对象。因此，本章首先对软件和软件工程进行简单的介绍。读者在阅读和学习这些概念和知识时，应注意思考它们能够带给软件测试的指导和启示。

1.1.1 软件

计算机技术的发展与革新带动了近几十年来科技的快速发展，也给人们的生活带来了翻天覆地的变化，被誉为第三次工业革命。计算机技术主要由软件技术和硬件技术两部分组成，软件与硬件结合形成一个完整的计算机系统，并实现相应的功能。

在计算机发展的初期，计算机的功能主要是由计算机的各个硬件部件有机地协调工作来完成的，当时所谓的软件就是程序，它的作用并没有得到人们足够的重视。而随着计算机技术的发展，人们越来越充分地认识到高质量的软件会使计算机系统的功能和效率大大提高，于是，软件在计算机系统中的作用也日益重要。

1. 软件定义

人们通常把各种不同功能的程序，包括系统程序、应用程序、用户自己编写的程序等称为软件。然而，随着计算机的应用日益普及，软件日益复杂且涉及规模日益增大，人们意识到软件并不仅仅等于程序。程序是人们为了完成特定的功能而编制的一组指令集，它由计算机的语言描述，并且能在计算机系统上执行。而软件不仅包括程序，还包括程序的处理对象——数据，以及与程序开发、维护和使用有关的图文资料（文档）。Roger S. Pressman 对软件给出了这样的定义：

计算机软件是由专业人员开发并长期维护的软件产品。完整的软件产品包括了在各种不同容量和体系结构计算机上的可执行程序，运行过程中产生的各种结果，以及以硬复制和电子表格等多种方式存在的软件文档。

2. 软件分类

软件本身的含义对软件测试的指导意义在于，软件测试活动不应只局限于对程序的测试，也要充分考虑软件涉及的数据和描述软件的各相关文档。

可以按照不同的角度对软件进行分类。

按照在计算机系统中所处应用层次的不同，软件可以分为系统软件、支撑软件和应用软件三类。系统软件是居于计算机系统中最靠近硬件的一层，为其他程序提供最底层系统服务，如编译程序和操作系统等；支撑软件以系统软件为基础，以提高系统性能为主要目标，支撑应用软件开发与运行，主要包括环境数据库、各种接口软件和工具组；应用软件是提供特定应用服务的软件，如字处理程序等。

按照软件本身规模的不同，软件可以划分为微型、小型、中型、大型和超大型软件。一般情况下，微型软件只需要一名开发人员，在4周以内完成开发，并且代码量不超过500行；小型软件一般需要2~3名开发人员，开发周期可以持续到半年，代码量一般控制在5000行以内；中型软件的开发人员控制在10人以内，要求在2年以内开发5000~50000行代码；大型软件的开发人员在10~100名，开发周期为1~3年，代码量在50000~100000行；超大型软件往往涉及上百名甚至上千名开发人员，开发周期可以持续到3年以上，甚至5年。

按照软件运行平台的不同，软件可以分为个人计算机软件、嵌入式软件、基于Web的软件等。个人计算机软件运行在PC上，为使用者提供各种应用，包括字处理、电子表格、计算机图形、多媒体、娱乐等；嵌入式软件驻留在嵌入式设备的只读内存中，用于控制智能产品和系统，功能相对简单，规模较小，要求有很高的系统性能；基于Web的软件以整个网络环境为应用平台，依托浏览器和各类网络协议，结合可执行指令和数据，提供了几乎是无限的、可被任何人通过浏览器访问的软件资源。

3. 软件与软件测试

针对软件的应用不同、规模不同、运行平台不同，需要选择不同的测试策略、测试方法，制定测试计划，编写测试用例，组织测试活动。比如，大型软件往往比小型软件需要进行更多的测试，并需要精心制定测试计划，有组织地执行测试活动；系统软件和支撑软件往往需要具有很好的兼容性、准确性和性能，为上层应用软件提供服务；基于Web的软件，往往负载能力成为决定其性能的核心指标之一；而嵌入式软件，需要严格控制其对运算能力和存储容量的需求，具有较高的性能。

1.1.2 软件工程

在20世纪60年代，计算机软件的开发、维护和应用过程中普遍出现了一些严重的问题，如开发出来的软件产品不能满足用户的需求；相比越来越廉价的硬件，软件价格过高；软件质量难以得到保证，且难以发挥硬件潜能；难以准确估计软件开发、维护的费用以及开发周期，往往导致软件产品不能在预算范围之内按照计划完成开发；难以控制开发风险，开发速度赶不上市场变化；软件产品修改维护困难，对遗留的系统进行集成更加困难等。人们将其称为“软件危机”。

这些问题严重影响了软件产业的发展，制约着计算机的应用。人们通过对导致“软件危机”的各种因素进行分析，发现软件在需求分析、开发过程、文档撰写、人员交流、测试、软件维护等很多方面都存在严重不足。为了解决“软件危机”，人们开始尝试用工程化的思想去指导软件开发，于是诞生了软件工程思想。

IEEE对软件工程的定义为：1）将系统化、严格约束的、可量化的方法应用于软件的开发、

运行和维护，即将工程化应用于软件；2）对1）中所述方法的研究。具体来说，软件工程是以借鉴传统工程的原则、方法，以提高质量、降低成本为目的来指导计算机软件开发和维护的工程学科。软件工程层次图如图1-1所示。

在图1-1中，软件质量被放在了最为基础的位置，是整个软件开发过程和开发方法所关注的核心内容，围绕着如何提高软件质量而对软件过程、软件方法进行的研究和相应软件工程辅助工具的开发也成了人们不断探索的课题。

在传统软件工程中流行的“瀑布模型”定义了开发一个软件的基本活动和它们的执行流程。如图1-2所示。

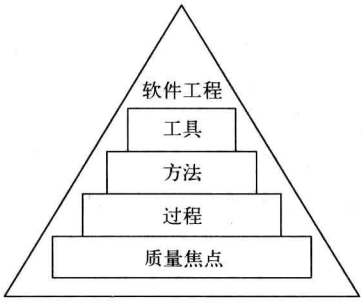

图1-1　软件工程层次图

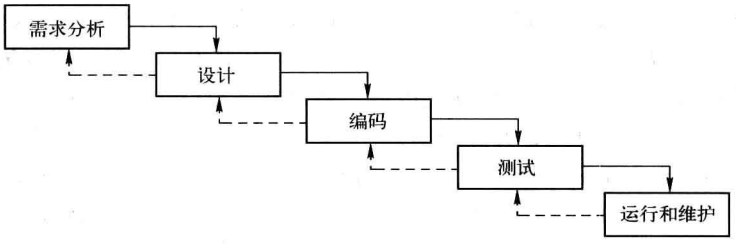

图1-2　瀑布模型图

1）需求分析产生软件的运行特征（功能、数据和行为）的规约，指明软件和其他系统元素的接口并建立软件必须满足的约束。

2）设计把软件需求描述转换为软件表示，包括数据设计、体系结构设计、接口设计和构件设计。

3）编码将设计产生的解题逻辑转换为可以执行的机器代码。

4）测试是动态验证软件的过程，包括内部测试和外部测试。对内部实现逻辑测试，以发现程序错误；对外部实现功能测试，以确保所有输入都生成与需求一致的实际输出。

5）运行和维护是将软件交付用户使用，并改正软件错误或为满足用户新需求而进行修改。

在瀑布模型中，软件测试作为一个重要步骤被执行，占据整个软件开发近40%的时间和工作量。可以说在早期的软件工程活动中，软件质量主要是通过测试活动保证的。

随着软件工程的发展，之后又出现了增量开发模型、演化开发模型、螺旋模型、统一过程模型、敏捷开发模型等，每一种模型的不同主要集中在活动之间的组织关系和执行顺序上，而作为活动本身的构成，基本没有太大的变化，测试作为核心活动之一，始终扮演着重要的角色。

1.1.3　软件质量

软件工程的目标是生产出高质量的软件。而对于软件质量本身的定义，却是一件十分困难的事情。Roger S. Pressman对软件质量的定义为：

软件要符合明确规定的功能和性能需求，符合已清晰文档化的开发标准，并且具有专业人员开发的软件所应有的隐含特征。

事实上，当我们在现实软件开发中评价软件的质量时，经常从多个方面来考察，这些方面称为软件的质量属性。软件的质量属性按其在运行时是否可见又分为：运行时可观察到的质量属性，包括正确性、性能、安全性、可用性、易用性；运行时不可观察的质量属性，包括可修改性、可移植性、可测试性、可集成性、可重用性等。

1）正确性：软件能够做正确的事情，并且能够正确地运行。

2）性能：系统的响应时间和硬件资源的占用率。

3）安全性：在对合法用户提供服务的同时，阻止未授权用户的使用企图。

4）可用性：能长时间正确运行并快速从错误状态恢复到正确状态。

5）易用性：使最终用户容易使用和学习。

6）可修改性：系统很容易被修改从而适应新的需求或采用新的算法、数据结构的能力。

7）可移植性：软件可以很简单地在平台间移植。

8）可测试性：软件能够被测试的容易程度。

9）可集成性：让分别开发的组件在一起正确工作。

10）可重用性：能够在新系统中应用已有的组件。

在现代软件工程中，将软件质量保证作为一个单独的活动执行，以确保软件质量在软件开发的全过程中都受到重视和验证，称之为软件质量保证（Software Quality Assurance，SQA）活动。SQA 包含：一种质量管理方法，有效的软件工程技术，在整个软件开发过程中采用的正式技术评审；一种多层次的测试策略，对软件文档及其修改的控制，保证软件遵从软件开发标准的规程，度量和报告机制。

有了 SQA 活动，软件测试是否还有必要呢？答案是肯定的。评审和其他 SQA 活动确实能够发现错误，减少错误数量，但这还是远远不够的。程序永远不能被证明是正确的（想想看为什么），用户的使用，实际上就是对系统的一种测试，为了能尽量消除软件中存在的错误，就必须在交付用户前对软件进行测试。软件测试活动和软件质量保证活动相互补充和协作，共同促进软件质量的改善和提高。

1.1.4 软件可靠性和软件错误

在介绍软件测试之前，还有两个概念需要了解，即软件可靠性和软件错误。

IEEE 对软件可靠性的定义为：

在特定环境和特定时间内，计算机程序无故障运行的概率。可以用“平均故障间隔时间”来作为软件可靠性的度量。其中，平均故障间隔时间 = 平均故障时间 + 平均修复时间。平均故障间隔时间的长度，可以较好体现对于用户来说软件可靠性的高低。

软件错误是一个统称，其中涉及很多概念。主要包括软件错误、软件缺陷、软件故障和软件失效。

1）软件错误：在软件生命周期内不希望或不可接受的人为错误，其结果是软件缺陷的产生，相对于软件是一种外部行为结果。

2）软件缺陷：存在于软件中的不希望或不可接受的偏差，其结果是软件运行于某一特定条件时出现软件故障，称为软件缺陷被激活。

3）软件故障：软件运行过程中出现的一种不希望或不可接受的内部状态，此时若未及时采取措施加以处理，便产生软件失效。

4）软件失效：软件运行时产生的一种不希望或不可接受的外部行为结果。

可以这样理解上述几个概念的关系：由于人为的软件错误，导致软件在开发过程中产生了软件缺陷；如果这些缺陷在交付用户之前没能被检测和纠正，就会在用户使用过程中的特定条件下被激活，变成软件故障；如果没有能够及时解决软件故障，便会导致软件失效，给用户带来损失和不好的结果。SQA 活动主要预防软件错误，发现软件缺陷；软件测试活动主要在软件交付前尽可能地发现软件缺陷；软件维护活动在软件失效前对软件故障进行

修正。

1.2　软件测试基本概念

有了前面的知识积累，相信读者已经对软件测试有了一定的认识，并掌握了相关的知识背景。那么，究竟什么是软件测试呢？

概括来说，软件测试是为了发现错误而执行程序的过程。或者说，软件测试是根据软件开发各阶段的规格说明和程序的内部结构，而精心设计一批测试用例，并利用这些测试用例去执行程序，以发现程序错误的过程。IEEE 对软件测试的定义为：

使用人工和自动手段来运行或测试某个系统的过程，其目的在于检测系统是否满足规定的需求或是弄清预期结果与实际结果之间的差别。

结合前面对软件的理解可以看到，所谓的软件测试，绝不仅仅是针对程序的测试。需求规格说明、概要设计规格说明、详细设计规格说明、程序等都是软件测试的对象。而实际统计数据表明，属于程序编写的缺陷只占到软件总缺陷的36%左右。可以把软件测试简单地理解成图 1-3 所示的过程。

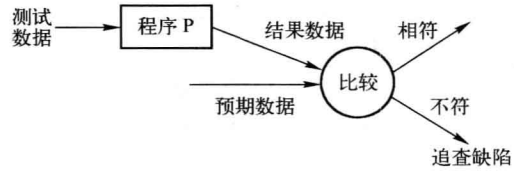

图 1-3　软件测试概念抽象图

1.2.1　软件测试的目的

软件危机导致了软件工程的产生，而软件质量是软件工程最注重的目标。在开发软件的过程中，导致产生各种软件缺陷的原因有很多，其中包括：开发人员之间、开发人员与用户之间缺乏有效的沟通；软件复杂度过高；编码错误；需求不断变更；时间有限；缺乏文档描述；没有合适的软件开发工具等。软件缺陷可能在软件开发的各个阶段被引入，如果没能及时发现和纠正，就会传递到软件开发的下一阶段。如图 1-4 所示。

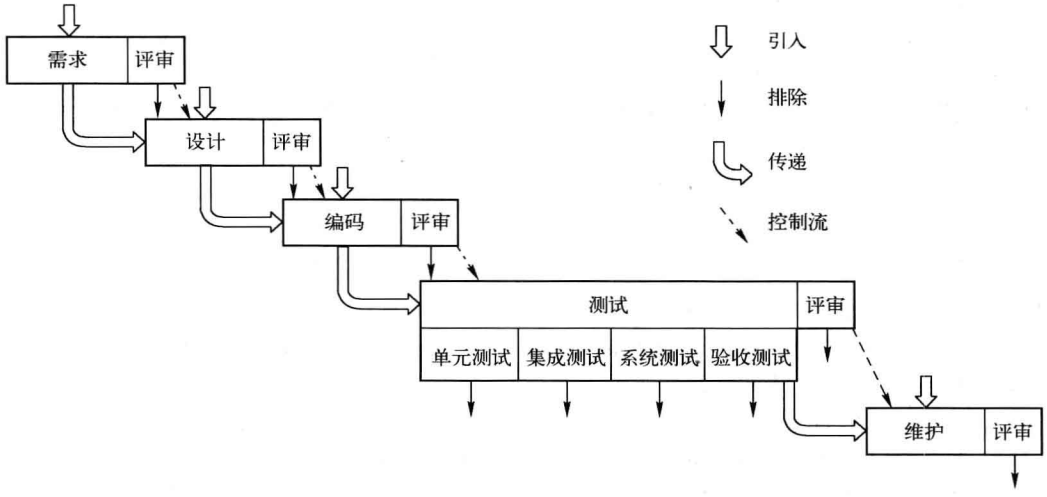

图 1-4　软件缺陷的引入和传递

随着软件缺陷的传递，会带来更多的问题，也会增加缺陷改正的难度和成本。IBM 公司给出的其对软件缺陷改正成本的研究成果表明：假设在分析阶段发现一个错误，其修正成本

为1个货币单位；则在测试之前发现一个错误，其修正成本约为6.5个货币单位；在测试时发现一个错误，其修正成本为15个货币单位；而在发布阶段发现一个错误，其修正成本则为60～100个货币单位。该比例也适合于发现一个错误需要的时间代价。

软件测试的目的，就是要发现软件中存在的缺陷和系统不足，定义系统的能力和局限性，提供组件、工作产品和系统的质量信息；提供预防或减少可能错误的信息，在软件开发过程中尽早检测错误以防止错误传递到下一阶段，提前确认问题和识别风险；最终获取系统在可接受风险范围内可用的信息，确认系统在非正常情况下的功能和性能，保证一个工作产品是完整的并且是可用或者可被集成的。

1.2.2 软件测试的原则

要达到软件测试的目的，就要了解和遵守一些软件测试原则：

1. 不可能进行完全测试

基于时间、人员、资金或设备等方面的限制，不可能对软件产品进行完全测试，即不可能考虑或测试到软件产品的所有执行情况或路径。而对于程序本身，在很多情况下，由于其运算复杂性和逻辑复杂性，在有限的时间内穷举测试也是不可能的。所以在测试过程中，应该采用具有代表性、最有可能查出系统问题的测试用例。

2. 测试中有风险存在

基于所使用的测试工具、测试方法或测试用例的局限性，在某些情况下，软件缺陷不会被发现。因此，正确的设计测试用例，并保证其满足一定的覆盖率是十分重要的。不同方法设计的测试用例应该用于同一被测试对象，从而避免某一种测试方法带来的局限性。

3. 软件测试并不能保证产品没有缺陷

软件测试只是查找软件缺陷的过程，即使测试人员使用了大量的测试用例、不同的测试方法对软件产品进行测试，测试成功以后也不能说明软件产品已经准确无误，完全符合用户的需求。也就是人们常说的“软件测试只能说明错误，不能说明正确”。

4. 软件产品中所存在的缺陷数与已发现的缺陷数成正比

软件测试所发现的缺陷越多，说明软件产品中存在的缺陷越多。一般情况下，潜在的缺陷数与发现的缺陷数存在着正比关系。并且软件缺陷的发生具有一定的群聚现象，在发现软件缺陷的地方，往往还存在其他的软件缺陷。

5. 要避免软件测试的杀虫剂现象

所谓杀虫剂现象是指，如果长期使用某种药物，那么生物就会对这种药物产生抗药性。同理，如果同一个软件产品总是由固定的测试人员去测试的话，那么基于这个测试人员思维方式、测试方法的局限性，有些缺陷是很难被发现的。所以，在软件测试的过程中，最好由不同的测试人员参与到测试工作中，不要由开发人员进行软件测试。

6. 及早地和不断地进行软件测试

根据软件缺陷的传递性和软件缺陷改正成本的递增性，及早地进行软件测试，可以及时发现和修改软件某一阶段产生的缺陷，从而避免其传递到下一阶段，降低缺陷改正成本。开发过程应该始终伴随着测试过程。

7. 进行回归测试

程序员在编写程序时经常有这样的经验：通过调试发现了一个程序bug并进行了修改，而重新调试时，发现由于上一个改动而导致了更多bug的出现。同样，软件测试发现缺陷并被改正后，很可能引入新的软件缺陷，往往是因为程序之间的关联性，或者缺陷的表现和缺陷的原因不

在同一个地方等。所以任何一次软件缺陷改正并提交后，都要进行回归测试来确保修改后没有引入新的软件缺陷。

8. 软件测试应该有计划、有组织地进行

作为软件开发中的重要活动，软件测试应该有软件测试计划进行指导，成立合适的软件测试团队，妥善保存一切软件测试过程文档，并建立有效的软件缺陷发现、上报、改正、跟踪、统计机制。避免软件测试过程中的盲目性、随意性和重复劳动。

1.2.3　软件测试的分类

软件测试按照不同的角度，有不同的分类方法。

1. 按开发阶段

按照开发阶段划分，软件测试可分为单元测试、集成测试、确认测试、系统测试和验收测试。

1）单元测试是对软件设计中最小的单位——程序模块进行的测试，它着重检查程序单元是否符合软件详细设计规约中对于模块功能、性能、接口和设计约束等方面的要求，发现各模块内部可能存在的各种错误。由于程序模块间应该具有低耦合、高内聚的特性，所以单元测试一般是可以并行进行的。

2）集成测试是在单元测试的基础上，将通过单元测试的各个模块有序地、递增地进行测试，它着重发现各模块接口关系和相互协作中是否存在错误。在很多情况下，通过单元测试的模块集成到系统中往往还存在问题，就是由于它没能正确地与其他模块协作，或者出现了接口错误。集成测试依据的标准是软件概要设计规约。

3）确认测试是通过检验和提供客观证据，验证软件是否满足特定预期用途的需求。它依据软件需求规格说明书，包括用户对软件的功能、性能和某些特性的要求。如果说前两种测试主要是验证软件是否在“正确地做事”，那么确认测试主要是验证软件是否在“做正确的事”。

4）系统测试是将通过确认测试的软件，作为整个基于计算机系统的一个元素，与计算机硬件、外部设备、网络和系统软件、某些支持软件、数据和人员等其他系统元素结合在一起，在实际运行环境下，检测其是否能够进行正确的配置、连接，满足用户需求。系统测试一般依据系统规格说明书。

5）验收测试是指按照项目说明书、合同、软件供需双方约定的验收依据文档等进行的对整个系统的测试和评审，决定是否接受该系统。

上述测试过程也是传统软件测试采用的过程，如图 1-5 所示。有时也把确认测试和系统测试归为一个过程，统称为系统测试。

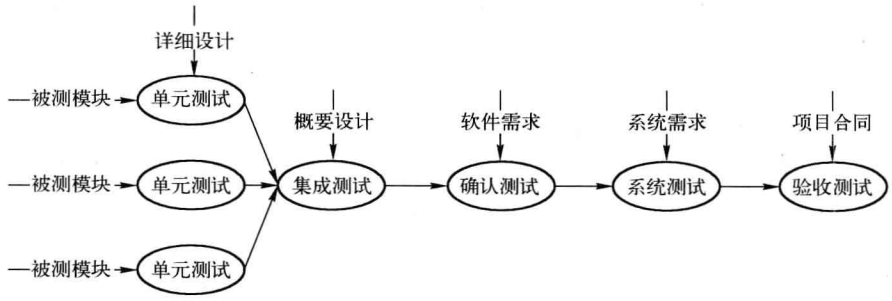

图 1-5　软件测试过程