工业和信息化
人才培养规划教材
Industry And Information
Technology Training
Planning Materials

高职高专计算机系列

# 软件测试任务驱动式教程

## Software Testing

陈承欢 ◎ 编著

- 典型工作场景，**8个**教学单元
- 提炼岗位核心技能，**34项**测试任务
- 介绍自动化测试工具等**新技术**、**新应用**
- 丰富的**教学资源**，面向教学的**全过程**教学环节设计

人民邮电出版社
POSTS & TELECOM PRESS

工业和信息化
人才培养规划教材
Industry And Information
Technology Training
Planning Materials

# 软件测试
# 任务驱动式教程
Software Testing

本书在对软件企业中软件测试岗位的岗位职责和岗位需求进行认真的调研分析， 对软件测试岗位必备的理论知识、必需的技能和素质、必用的测试工具进行深入的学习和分析，并对教学内容进行系统化重构的基础上编写而成。本书科学设计了8个教学单元，并精心设计了34项测试任务，可以帮助读者在真实的测试环境中完成真实应用程序和软件系统的测试工作，并在这个过程中掌握知识、训练技能、积累经验和固化能力。

本书以测试实践为主线，将测试方法指导与测试实践活动有机结合，强调“做中学” ，注重理论指导实践；关注软件测试行业的发展现状和未来方向，使用QTP、LoadRunner、JUnit等先进的自动化软件测试工具执行软件测试操作。

免费提供
PPT等教学相关资料

人民邮电出版社
教学服务与资源网
www.ptpedu.com.cn

教材服务热线：010-81055256
反馈／投稿／推荐信箱：315@ptpress.com.cn
人民邮电出版社教学服务与资源网：www.ptpedu.com.cn

封面设计：董志桢

ISBN 978-7-115-34670-4

ISBN 978-7-115-34670-4

定价：45.00 元

工业和信息化
人才培养规划教材
Industry And Information
Technology Training
Planning Materials

高职高专计算机系列

# 软件测试任务驱动式教程

## Software Testing

陈承欢 ◎ 编著

人民邮电出版社
北京

图书在版编目（CIP）数据

软件测试任务驱动式教程 / 陈承欢编著. -- 北京 : 人民邮电出版社, 2014.4
工业和信息化人才培养规划教材. 高职高专计算机系列
ISBN 978-7-115-34670-4

Ⅰ. ①软… Ⅱ. ①陈… Ⅲ. ①软件－测试－高等职业教育－教材 Ⅳ. ①TP311.5

中国版本图书馆CIP数据核字(2014)第025723号

## 内 容 提 要

本书在对软件企业中软件测试岗位的岗位职责和岗位需求进行认真的调研分析，对软件测试岗位必备的理论知识、必需的技能和素质、必用的测试工具进行深入的学习和分析，并对教学内容进行系统化重构的基础上编写而成。本书科学设计了 8 个教学单元，并精心设计了 34 项测试任务，可以帮助读者在真实的测试环境中完成真实应用程序和软件系统的测试工作，并在这个过程中掌握知识、训练技能、积累经验和固化能力。

本书以测试实践为主线，将测试方法指导与测试实践活动有机结合，强调“做中学”，注重理论指导实践；关注软件测试行业的发展现状和未来方向，使用 QTP、LoadRunner、JUnit 等先进的自动化软件测试工具执行软件测试操作。

书中每一个教学单元面向教学全过程设置了 6 个必要的教学环节：教学导航→方法指导→引导测试→探索测试→测试拓展→单元小结，适合于灵活多样的教学组织方式。

本书可以作为高等院校计算机类各专业以及其他各相关专业的软件测试教材，也可以作为软件测试技术人员的参考书。

◆ 编　　著　陈承欢
责任编辑　王　威
责任印制　杨林杰

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
中国铁道出版社印刷厂印刷

◆ 开本：787×1092　1/16
印张：19.5　　2014 年 4 月第 1 版
字数：512 千字　　2014 年 4 月北京第 1 次印刷

定价：45.00 元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315

# 前　言

随着计算机软件在各行各业的普及应用，软件的质量问题也越来越受到人们的关注，即使是被广泛使用的成熟软件产品，也难免有不尽人意之处。软件测试是一种以找出隐藏在软件中的缺陷和错误为主要目的的活动，软件测试是软件开发过程中不可缺少的一个重要环节，是保证软件质量的重要手段。由于软件的复杂程度不断增强，软件测试也变得越来越重要，受到了业界广泛的重视。

本书主要有以下特色。

（1）认真调研分析软件测试岗位的知识、技能和素质需求，系统化重构教学内容，科学设计教学单元。

书中对不同城市 100 多家软件企业的软件测试岗位的岗位职责和岗位需求进行了认真的调研分析，对软件测试岗位必备的理论知识、必需的技能和素质、必用的测试工具进行了深入地了解和分析，在此基础上系统化重构教学内容，科学设计了 8 个教学单元：软件测试的认知与体验、结构化应用程序的黑盒测试与白盒测试、.NET 应用程序的单元测试与界面测试、Java 应用程序的单元测试与功能测试、Windows Mobile 应用程序的单元测试与功能测试、基于类的数据库应用程序的单元测试和性能测试、Web 应用程序的性能测试与负载测试、软件系统的集成测试与系统测试。

（2）在真实的测试环境中完成真实应用程序和软件系统的测试工作，在完成各项测试任务过程中掌握知识、训练技能、积累经验并固化能力。

本书精心设计了 34 项测试任务，这些测试任务主要涉及结构化应用程序、.NET 应用程序、Java 应用程序、Windows Mobile 应用程序、面向对象应用程序、数据库应用程序、Web 应用程序和软件系统，重点关注黑盒测试、白盒测试、单元测试、集成测试、系统测试、界面测试、功能测试、性能测试和负载测试，尽量做到涉及面广、突出重点、抓住关键。让学习者在完成各项测试任务的过程中，理解和掌握理论知识，逐渐具备软件测试的基本能力和积累软件测试的经验，从而适应软件测试岗位的需求。

（3）以测试实践为主线，测试方法指导与测试实践活动有机结合，强调“做中学”，注重理论指导实践。

软件测试岗位对测试技能和测试经验有较高的要求，对于《软件测试》课程，如果只重视软件测试的理论知识和方法原则，而没有足够的测试实践活动，则很难胜任软件企业测试岗位的工作。本书以测试实践为主线，同时也重视方法指导、注重理论指导测试实践。

（4）关注软件测试行业的发展现状和未来方向，使用先进的自动化软件测试工具执行软件测试操作。

软件测试自动化是软件测试未来发展的重要方向，目前也备受众多软件企业的关注，软件企业纷纷引入自动化测试工具，通过自动化测试提高测试效率和测试覆盖率。软件测试人员必须熟悉和掌握先进的自动化软件测试工具，并学会在软件测试实践中熟练使用。本书重点介绍和使用了 Quick Test Professional（QTP）、LoadRunner、JUnit、Microsoft Visual Studio 2008 这 4 种测试工具，一般介绍和使用了 StyleCop、FxCop、Nunit、NunitForms、DevPartner Studio Professional Edition 这 5 种测试工具，简要介绍了 QACenter、TestComplete、TestDirector、

Framework for Integrated Test、Jtest、JCheck、Hopper、ERWin Examiner、TSQLUnit 和 Robot 这 10 种测试工具。JUnit 和 Microsoft Visual Studio 2008 是常用的单元测试工具，QTP 是一种优秀的功能测试工具，LoadRunner 是一种优秀的性能测试工具，这些都是软件企业中普遍使用的软件测试工具，学习者都应熟练掌握。

（5）面向教学全过程设置完善的教学环节。

本书的每一个教学单元面向教学全过程设置了 6 个必要的教学环节：教学导航→方法指导→引导测试→探索测试→测试拓展→单元小结，突出了方法指导测试实践，符合学习者的认知规律和技能形成规律，有助于提高教学效果和操作技能。

（6）适合于灵活多样的教学组织方式。

本书适合于灵活多样的教学组织方式，可以为串行方式（连续安排 2~3 周）组织教学，也可以为并行方式（每周安排 6~8 课时，安排 8 周左右，每周完成一个教学单元）组织教学。

本书使用的编程语言为 C#和 Java，软件开发环境为 Microsoft Visual Studio 2008、Eclipse-SDK-3.7.1 和 JDK1.7，数据库开发环境为 Microsoft SQL Server 2008。

本书由陈承欢教授编著，参加本书测试程序的设计、优化和部分章节的编写、校对和整理工作的还有吴献文、谢树新、颜谦和、刘钊、颜珍平、宁云智、肖素华、林保康、王欢燕、张丹、冯向科、林东升、刘荣胜、郭外萍、侯伟、唐丽玲、陈雅、张丽芳等老师。

由于编者水平有限，书中难免存在疏漏之处，敬请各位专家和读者批评指正。为方便教学，本书还配备了 PPT 等教学辅助资源，请登录人民邮电出版社教学服务与资源网（www.ptpedu.com.cn）下载使用。感谢您使用本书，期待本书能成为您的良师益友。

陈承欢

2014 年 3 月

# 目 录 CONTENTS

# 单元 1
# 软件测试的认知与体验



软件测试是软件开发过程中不可缺少的一个重要环节，是确保软件质量的重要手段。由于软件的复杂程度不断增强，软件测试也变得越来越重要，受到了业界广泛的重视。

## 【教学导航】

| | |
|---|---|
| 教学目标 | （1）熟悉软件测试的基本概念以及软件测试在整个软件开发生命周期中的地位和作用<br>（2）了解软件测试的分类和对软件测试人员的要求<br>（3）掌握软件测试的目的、原则和流程<br>（4）掌握场景设计法与软件测试的基线<br>（5）学会对 Windows 操作系统自带的计算器进行功能测试和界面测试<br>（6）学会应用场景法对 ATM 机进行黑盒测试<br>（7）学会应用场景法对 QQ 登录的界面和功能进行测试 |
| 教学方法 | 讲授分析法、任务驱动法、探究学习法 |
| 课时建议 | 6 课时 |
| 测试阶段 | 验证测试、确认测试 |
| 测试对象 | 计算器、ATM 机、QQ 登录 |
| 测试方法 | 黑盒测试法、功能测试、界面测试、场景法 |

## 【方法指导】

## 1.1 软件测试概述

### 1．软件的概念

软件是计算机系统中与硬件相互依存的一个部分，它是源程序、数据及其相关文档的集合。源程序是按事先设计的功能和性能要求执行的指令序列；数据是使程序能正常操纵信息的数据结构；文档是与程序开发、维护和使用相关的图文资料，包括《用户需求规格说明书》、《系统概要设计》、《系统详细设计》、《数据库设计》、《用户操作手册》等。

### 2．软件缺陷的概念

软件缺陷（Defect）是指计算机软件中存在的某种破坏其正常运行的问题、错误，或者其中隐藏的功能缺陷，称为“Bug”，“Bug”在英语中是臭虫的意思，通常用“Bug”表示计算机

系统硬件或软件中隐藏的错误、缺陷或问题。

软件缺陷的存在会导致软件产品在某种程度上不能满足用户的需要，在软件开发过程中，产生软件缺陷是不可避免的，产生软件缺陷的原因比较复杂，主要包括以下几个方面。

（1）软件项目复杂。

计算机技术的进步，使软件规模、功能、结构日益复杂，算法的难度不断增加，而软件却要求高精确性，任何一个环节出现了差错都会导致软件出现错误。

（2）沟通交流不够。

系统需求分析时对客户的需求理解不透彻或者与用户沟通不畅，不同阶段的开发人员相互理解不一致，也会造成软件缺陷。例如：软件设计人员对需求分析的理解有偏差；编程人员对系统设计说明书中的某些内容重视不够，或存在误解；对于设计与编程上的一些假定或依赖性，相关人员没有充分沟通；项目组成员技术水平参差不齐、交流不够或交流上有误解，导致软件开发和维护过程中出现问题，从而产生软件缺陷。

（3）程序设计错误。

程序设计时出现算法错误、语法错误、计算和精度问题、系统结构不合理、接口参数不匹配等问题，导致软件存在隐藏的缺陷。

（4）软件需求变化。

软件需求变化的影响是多方面的，需求变化的后果可能会造成软件的重新设计，设计人员日程重新安排，已经完成的工作可能要重做或者完全抛弃等。如果有多次小的改变或者一次大的变化，项目各部分之间已知或未知的依赖性可能会相互影响而导致更多问题的出现，需求改变带来的复杂性也可能导致出现错误。

（5）代码文档缺陷。

不规范或不完整的文档使得代码维护和修改变得非常麻烦，其结果是带来许多错误。

（6）开发工具错误。

可视化软件开发工具、类库、编译器、脚本开发工具等自身的错误被带到开发的软件中，从而产生软件缺陷。

对于软件来讲，不论采用什么技术和方法，软件中仍然会有错。采用先进的编程语言、先进的开发方法、完善的开发过程，可以有效减少错误的产生，但是不可能完全杜绝软件中的错误，这些错误需要通过测试来找出，软件中的错误密度也需要通过测试来估计。

### 3．软件测试的概念

简单地说，软件测试就是为了发现错误而执行程序的过程。软件测试是一个找错的过程，测试只能找出程序中的错误，而不能证明程序无错。测试要求以较少的用例、时间和人力找出软件中潜在的各种错误和缺陷，以确保软件系统的质量。

在 IEEE 所提出的软件工程标准术语中，软件测试的定义为“使用人工或自动手段来运行或测试某个系统的过程，其目的在于检验它是否满足规定的需求或弄清楚预期结果与实际结果之间的差别”。软件测试与软件质量密切相连，软件测试归根到底是为了保证软件质量，软件质量通过与否是以“满足需求”为基本衡量标准的，该定义明确提出了软件测试以检验是否满足需求为目标。

软件测试的主要工作是验证（Verification）和确认（Validation）。验证是保证软件正确实现特定功能的一系列活动，即保证软件做了所期望的事情；确认是一系列的活动和过程，其目的是证实在一个给定的外部环境中软件的逻辑正确性。

软件测试的对象不仅仅是程序，还包括整个软件开发期间各个阶段所产生的文档。

**4．测试用例的概念**

测试用例是为某个特定目标而编制的一组测试输入、执行条件以及预期结果，以便测试某个程序路径或核实是否满足某个特定需求。测试用例的目的是确定应用程序的某个特性是否能够正常工作，并且达到程序所设计的结果。

一个好的测试用例会使测试工作达到事半功倍的效果，并且能尽早发现一些隐藏的软件缺陷。测试用例（Test Case）可以用一个简单的公式来表示：

测试用例＝输入＋输出＋测试环境

其中，输入是指测试数据和操作步骤；输出是指系统的预期执行结果；测试环境是指系统环境配置，包括硬件环境、软件环境和数据，有时还包括网络环境。

**5．测试环境的概念**

简单地说，测试环境就是软件运行的平台，即进行软件测试所必需的工作平台和前提条件，可用如下公式来表示：

测试环境＝硬件＋软件＋网络＋历史数据

## 1.2 软件测试的地位和作用

软件测试在整个软件开发生命周期中占据着重要的地位，软件工程采用的生命周期方法把软件开发划分成多个阶段，把整个开发工作明确地划成若干个开发步骤，可以把复杂的问题按阶段分别加以解决，为中间产品提供了检验的依据，各阶段完成的软件文档成为了检验软件质量的主要依据。很显然，表现在程序中的错误，并不一定是编码所引起的，也可能是详细设计、概要设计阶段，甚至是需求分析阶段的问题引起的。因此，软件中所出现问题的根源可能在开发前期的各个阶段。解决问题、纠正错误也必须追溯到前期的工作。正因如此，软件测试工作才应该着眼于整个软件生命周期，特别是着眼于编码以前各个开发阶段的工作来保证软件的质量。如果不在早期阶段进行测试，错误的延时扩散常常会导致最后成品测试产生巨大困难。所以说，软件测试并非传统意义上产品交付前单一的“找错”过程，而是贯穿于软件生产全过程的一个科学的质量控制过程。一个软件项目的需求分析、概要设计、详细设计以及编码等各阶段所得到的文档，包括需求规格说明、概要设计规格说明、详细设计规格说明以及源程序，都应该是软件测试的主要对象，软件开发的整个过程都需要软件测试人员的介入。

软件测试应该从生命周期的第一个阶段开始，并贯穿于整个软件开发生命周期的每个阶段，尽可能早地发现错误并加以修正，早期检测和纠错是系统开发中最有效的方法。

（1）需求分析阶段。

在软件开发初期，与问题定义和需求分析同时进行的验证行为极其重要，该验证必须对需求进行彻底的分析，并在初始测试时得到预期的回答。进行这些测试有助于明确系统需求，这些测试将成为最终测试单元的核心。

（2）系统设计阶段。

系统设计阶段要阐明一般测试策略，如测试方法和测试评价标准，并创建测试计划。另外，重大测试事件的日程安排也应在这一阶段构建，同时还要建立质量保证和测试文档的框架。

在详细设计阶段，要确定相应的测试工具并产生测试规程，同时还要构建功能测试所需的测试用例。除此之外，设计过程本身也要经过分析和检查以排除错误。设计中所遗漏的情况、不完善的逻辑结构、模块接口不匹配、数据结构不一致、错误的输入输出设计和不恰当的接口

等都是需要考虑的内容。

（3）系统编码阶段。

在编码阶段要进行足够的单元测试，很多测试工具和技术会应用于这一阶段。代码走查和代码审查都是有效的人工测试技术；静态分析技术通过分析程序特征来排除错误；对于大型程序，需要用自动化工具来完成这些分析。

（4）系统测试阶段。

测试应用系统应着眼于功能上的测试，严格控制和管理测试信息是最重要的。

（5）系统安装阶段。

系统安装阶段的测试必须确保投入运行的程序是正确的，例如，程序是正确的版本，确保数据被正确地更改和增加。

（6）系统维护阶段。

系统启用以后，无论是纠正系统错误还是扩充原系统，都需要对系统进行更改。系统在每一次更改之后都需要重新测试，这种重新进行的测试称为回归测试。虽然一般情况下，只对由于更改而影响系统的部分进行重新测试，但是任何程序的变化都有必要进行重新测试、重新确认并更新文档。

## 1.3 软件测试的目的

软件测试的目的是为了在软件开发过程中，对软件产品进行质量控制并保证软件产品的最终质量。测试可以完成许多事情，但最重要的是可以衡量正在开发软件的质量。

对于软件测试目的，Grenford J. Myers 提出以下观点。

（1）软件测试是一个为了发现错误而执行程序的过程。

（2）软件测试是为了证明程序有错，而不是证明程序无错。

（3）一个好的测试用例在于它能发现至今尚未发现的错误。

（4）一个成功的测试是发现了至今尚未发现错误的测试。

这些观点提醒人们，测试要以查找错误为中心，而不是为了演示软件的正确功能。软件测试并不仅仅是为了要找出错误，也是对软件质量进行度量和评估，以提高软件的质量。软件测试是以评价一个程序或者系统属性为目标的活动，从而验证软件的质量满足用户的需求的程度，为用户选择与接受软件提供有力的依据。通过分析错误产生的原因和错误的分布特征，可以帮助软件项目管理者发现当前所采用的软件过程的缺陷，以便改进软件过程。同时，通过分析也能帮助我们设计出有针对性的检测方法，改善测试的有效性。即使是没有发现错误的测试也是有价值的，完整的测试是评定测试质量的一种方法。

## 1.4 软件测试的原则

为了进行有效的测试，测试人员应理解和遵循以下基本原则。

（1）应当把“尽早地和不断地进行软件测试”作为软件开发者的座右铭。

由于软件系统的复杂性和抽象性，软件开发各个阶段工作的多样性，以及开发过程中各种层次的人员之间工作的配合关系等因素，使得软件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，坚

持在软件开发的各个阶段进行技术评审，这样才能在开发过程中尽早发现和预防错误，杜绝隐患，提高软件质量。

（2）程序员应避免检查自己的程序。

人们常常由于各种原因而产生一些不愿意否定自己的心理，认为揭露自己程序中的问题不是一件令人愉快的事情。这一心理状态就成为测试自己编写的程序的障碍。另外，由于程序员对软件规格说明理解错误而引入的程序中的错误则更难发现。如果由别人来测试程序员编写的程序，可能会更客观、更有效并更容易取得成功。但程序测试和程序调试（Debuging）要区别对待，调试程序由程序员自己来做可能更有效。

（3）测试用例应由测试输入数据和与之对应的预期输出结果两部分组成。

在进行测试之前应当根据测试的要求选择测试用例（Test Case），测试用例不但需要测试的输入数据，而且需要针对这些输入数据的预期输出结果。如果对测试输入数据没有给出预期的输出结果，那么就缺少检验实测结果的基准，就有可能把一个似是而非的错误结果当成正确结果。

（4）在设计测试用例时，应当包括合理的输入条件和不合理的输入条件。

合理的输入条件是指能验证程序正确性的输入条件，而不合理的输入条件是指异常的、临界的、可能引起问题变异的输入条件。在测试程序时，人们常常倾向于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，软件系统在投入运行以后，用户的使用往往不遵循事先的约定，使用了一些意外的输入，如用户在键盘上按错了键或输入了非法的命令。如果软件对这种异常情况不能做出适当的反应，给出相应的信息，那么就容易产生故障，轻则输出错误的结果，重则导致软件失效。因此，软件系统处理非法命令的能力也必须在测试时受到检验。用不合理的输入条件测试程序时，往往比用合理的输出条件进行测试更能发现错误。

（5）充分注意软件测试时的群集现象。

测试时不要以为找到了几个错误问题就不需要继续测试了。在所测试的程序中，若发现的错误数目较多，则残存的错误数目也比较多，这种错误群集现象已被许多程序的测试实践所证实。针对这一现象，应当对错误群集的程序进行重点测试，以提高测试效率。

（6）严格执行测试计划，排除测试的随意性。

测试计划的内容要完整、描述要明确，并且不要随意更改。

（7）应当对每一个测试结果做全面检查。

有些错误的征兆在输出实测结果时已经明显地出现了，但是如果不仔细、全面地检查测试结果，就会使这些错误被遗漏掉。所以必须对预期的输出结果明确定义、对实测的结果仔细分析检查、暴露错误。

（8）妥善保存测试过程中产生的各种数据和文档。

对于测试过程中产生的测试计划、测试用例、出错统计和分析报告等数据和文档应妥善保存，为日后维护提供方便。

（9）注意回归测试的关联性。

回归测试的关联性一定要引起充分的注意，修改一个错误而引起更多错误出现的现象并不少见。

## 1.5 软件测试的分类

软件测试有多种分类方式，例如，按测试阶段分类、按是否需要运行被测试软件分类、按是否需要查看代码分类、按测试执行时是否需要人工干预分类、按测试目的分类等，表 1-1 描述了软件测试的各种分类。

表 1-1　　软件测试的各种分类

| 分类方式 | 测试类型 |
| --- | --- |
| 按测试阶段分类 | 单元测试、集成测试、确认测试、系统测试、验收测试 |
| 按是否需要执行被测试软件分类 | 静态测试、动态测试 |
| 按是否需要查看代码分类 | 白盒测试、黑盒测试、灰盒测试 |
| 按测试执行时是否需要人工干预分类 | 手工测试、自动测试 |
| 按测试目的分类 | 功能测试、界面测试、易用性测试、兼容性测试、安全性测试、接口测试、文档测试、安装与卸载测试、配置测试、恢复测试、性能测试、负载测试、压力测试、强度测试、并发测试、可靠性测试、健壮性测试等 |
| 其他测试类型 | 冒烟测试、随机测试、回归测试 |

### 1.5.1 按测试阶段分类

软件测试按测试阶段可划分为单元测试、集成测试、确认测试和系统测试，最后进行验收测试。其中，单元测试主要是分别完成每个单元的测试任务，以确保每一个模块能正常工作。集成测试是把已测试过的模块组装起来，进行集成测试，其目的在于检验与软件设计相关的程序结构问题，这时较多地采用黑盒测试方法来设计测试用例。确认测试主要是在完成集成测试之后，对开发工作初期制定的确认准则进行的检验，是检验所开发软件是否满足所有功能和性能需求的最后手段，通常采用黑盒测试方法。系统测试是针对在完成确认测试以后，交付的应该是合格的软件产品，但为检验它能否与系统的其他部分（如数据库、硬件）协调工作而需要进行的测试；严格地说，系统测试已经超出了软件工程范围；验收测试是检测软件产品质量的最后一道工序，它更加突出了客户的作用，同时软件开发人员也应有一定程度的参与。

**1．单元测试**

单元测试（Unit Testing）又称模块测试（Module Testing），是指对软件中的最小可测试单元进行测试，目的是检查每个单元是否能够正确实现详细设计说明中的功能、性能、接口和设计约束等要求，发现各个模块内部可能存在的各种缺陷。

软件开发过程中，程序编码完成且通过编译后就可以进行单元测试了。单元测试的主要依据是程序代码和详细设计文档，根据设计文档、编码规范和注释，从程序内部结构出发，查看代码是否符合设计、规范以及注释的相关说明。

单元测试具有以下优点。

（1）是一种管理和组合测试元素的手段。通过单元测试，可以实现测试从“小规模”向“大规模”的逐步转变，从而降低测试难度，提高测试效率。

（2）可以减轻调试的难度。调度是准确定位并纠正某个已知缺陷的过程，在测试用例指导

下的单元测试针对性更强，更加有利于缺陷的定位和对失效原因的分析。

（3）提供同时测试多个单元的可能。多个相互独立或关联性不大的单元可以并行地、独立地进行单元测试，从而加快整体的测试速度和推进软件项目开发的进度。

**2．集成测试**

集成测试（Integration Testing）又称为组装测试，是在单元测试的基础上，按照设计要求，将通过单元测试的单元组装成系统或子系统而进行的测试，其目的是检验不同程序单元或部件之间的接口关系是否符合概要设计的要求，能否正常运行。

集成测试并不是等到所有单元测试完成之后才开始的，而是同步进行的，即在单元测试中先对几个单元进行独立的单元测试，然后将其组装起来进行集成测试，查看其接口是否存在缺陷。

集成测试的主要依据是概要设计文档，与单元测试相比，集成测试主要考查单元的外部接口，而单元测试主要从单元内部进行测试。

由于软件的集成是一个持续的过程，会形成多个临时版本，在不断集成的过程中，功能集成的稳定性是需要解决的主要问题，在提交每个版本时，应先进行冒烟测试（Smoke Testing，即版本验证测试）来验证主要功能是否能够正常执行。

**3．系统测试**

系统测试（System Testing）是为了验证和确认系统是否达到其原始目标，而对集成的硬件和软件系统进行的测试，是在真实或模拟系统运行的环境下，检查完整的程序是否能和系统（包括系统软件、支持平台、硬件、外设和网络）正确配置、连接，并满足用户需求。

系统测试的主要依据是软件的需求规格说明文档，是在整个系统集成好之后进行的、前期主要看系统功能是否满足需求，称为功能测试；后期主要测试系统运行是否满足要求，以及系统在不同硬件和软件环境中的兼容性等，分别称为性能测试、兼容性测试、用户界面测试等。一般将功能测试从系统测试中独立出来，作为集成测试和系统测试之间的一个测试环节。

**4．确认测试**

确认测试是通过检验和提供客观证据，证实软件是否满足特定预期用途的需求，检测与证实软件是否满足软件需求说明书中规定的要求。

**5．验收测试**

验收测试（Acceptance Testing）又称接受测试，是在系统测试后期，以用户测试为主，或有质量保证人员共同参与的测试。验收测试是软件正式交付给用户使用前的最后一个测试环节，并决定用户是否最终验收签字和结清所有应付款。

验收测试的主要依据是软件需求规格说明文档和验收标准。验收测试的测试用例可以直接采用内部测试组所设计和系统测试用例的子集，也可以由验收人员自行设计。

验收测试又分为α测试和β测试。

α测试也称为开发方测试，开发方通过检测和提供客观证据，证明软件运行是否满足用户规定的需求。它是在软件开发环境下，由用户、测试人员和开发人员共同参与的内部测试，属于软件产品早期性测试。该测试一般在可控制环境下进行，可以是用户在开发环境下进行的测试，也可以是软件公司内部用户在模拟实际操作环境下进行的受控测试。

β测试是内部测试之后的外部公开测试，是将软件完全交给用户，让用户在实际使用环境下进行的对产品预发版本的测试。该测试是在开发者无法控制的环境下进行的软件现场测试，其实施过程是先将软件产品有计划地分发到目标市场，让最终用户大量使用、评价和检查软件，从而发现软件缺陷，然后从市场收集反馈信息，把关于反馈信息的评价制成容易处理的数据表，

再将这些数据分发给所涉及的各个部门进行修改。

β 测试通常被看成是一种“用户测试”，它使得“实际”客户有机会将自己的意见渗透到公司产品的设计功能和使用过程中，这些意见不但可对测试软件起到非常重要的作用，还有利于将收集的数据有效利用并促进公司未来新产品的研发。β 测试可以发现一些在测试实验室无法发现、甚至重复出现的缺陷，使软件公司更了解用户的需求，并为产品设计提供指南，有助于对产品的未来做出重要决策。

### 1.5.2 按是否需要执行被测试软件分类

#### 1．静态测试

静态测试（Static Testing）又称为静态分析（Static Analysis），是不实际运行被测软件，而是直接分析软件的形式和结构，从而查找缺陷的测试，主要包括对源代码、用户界面和各类文档及中间产品（如产品规格说明书、技术设计文档等）所做的测试。

（1）测试程序代码。

测试程度代码主要是为了查看代码是否符合相应的标准和规范，如可读性、可维护性等，其工作过程类似一个编译器，随着语法分析的进行，分析模块调用图、程序的控制流图等图表，度量软件的代码质量等。源代码中含有大量设计信息，以及程序异常的信息。利用静态测试，不仅可以发现程序中明显的缺陷，还可以帮助程序员重点关注那些可能存在缺陷的高风险模块，如多出口的情况、程序复杂度过高的情况、接口过多的情况等。

（2）测试界面。

测试界面主要是查看软件的实际操作和运行界面是否符合需求中的相关说明，是否符合用户的要求。

（3）文档测试。

文档测试主要是检查需求规格说明书、用户手册与需求说明是否真正符合用户的实际需求。

静态测试采用走查、同行评审、会审等方式来查找错误或收集所需要的数据。不需要运行程序，所以相对于动态测试，可以更早地进行。

静态测试的查错和分析功能是其他方法所不能替代的，静态分析能发现文档中的问题，通过文档中的问题或其他软件评审来找出需求分析、软件设计等存在的问题，而且能有效检查代码是否具有可读性、可维护性，是否遵守编程规范，包括代码风格、变量/对象/类的命名、注释内容等。

#### 2．动态测试

动态测试（Dynamic Testing）又称为动态分析（Dynamic Analysis），是指需要实际运行被测软件，通过观察程序运行时所表现出来的状态、行为等发现软件缺陷的测试，包括在程序运行时，通过有效的测试用例来分析被测程序的运行情况或进行跟踪对比，发现程序所表现的行为与设计规格与客户需求不一致的地方。

无论是单元测试、集成测试，还是系统测试、验收测试，动态测试都是一种经常使用的测试方法。但动态测试也存在很多局限性，与静态测试相比，动态测试增加了测试用例的设计、执行和分析，以及由测试用例所带来的用例组织与管理等一系列活动。动态测试需要搭建软件特定的运行环境，增加了有关测试环境的配置、维护和管理的工作量。