# 软件测试基本概念

## 1.1理解软件质量与软件测试

目前，在软件定义方面有很多定义。最主要的有以下几种。

 国际标准化组织ISO在质量特性国际标准ISO/IEC 9162中将软件质量定义为反应软件产品满足规定需求和潜在需求能力的特征和特性的总和。

MJ.Fisher 将软件质量定义为：所有描述计算机优秀程度的特性组合。 TomMcCabe 则认为软件质量是较高的用户满意程度以及较低的缺陷等级，这常常同较低的软件复杂程度有关。

SEI的Watts Humphrey倾向于将软件质量定义为具有很高的可实用性，同需求很好吻合，并具有高的可靠性和可维护性。

按照ANSI/IEEE std 1061-1992中的标准，软件质量定义为：与软件产品满足需求所规定的和隐含的能力有关的特征或特性的总体。

软件测试是软件工程不可或缺的重要环节之一, 是软件质量保证(Software Quality Assurance)的重要手段。

## 1.2理解软件测试定义

<https://www.cnblogs.com/liyongmei/p/10633744.html>

为了发现程序中错误而执行程序的过程

## 1.3理解软件测试目的

1）软件测试为了发现程序存在的代码或业务逻辑错误；

2）软件测试为了检验产品是否符合用户需求；

3）软件测试为了提高用户的体验

## 1.4理解软件测试原则

1）测试应该尽早介入；----需求分析

2）所有的测试都应追溯到用户需求；

3）程序员应该避免检查自己的程序。除了单元测试。因为程序员对于自己的作品，思维具有局限性。无法保证测试质量。交给第三方或者专业测试，运用各种测试技术，利用丰富的测试经验和对bug的敏感，去提高软件的质量；

4）设计测试用例时应考虑到合法的输入和不合法的输入以及各种边界条件，特殊情况下还要制造极端状态和意外状态。

5）二八原则，测试发现的错误中80%很可能起源于20%的模块中；

6）对错误结果要进行一个确认过程；

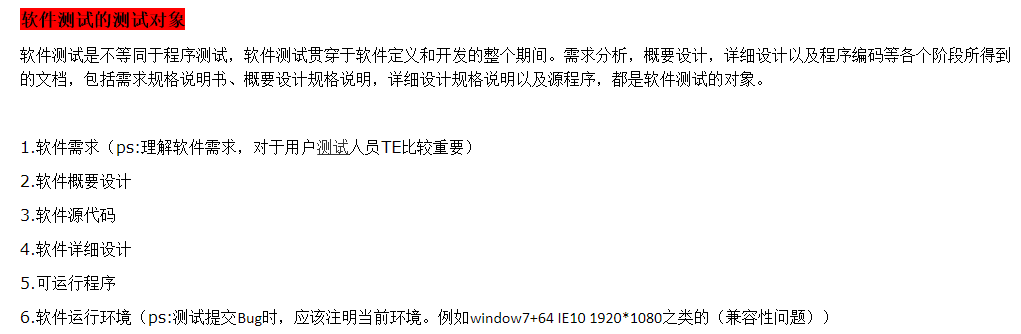
7）制定严格的测试计划；

8）完全测试是不可能的，测试需要终止；

9）妥善保存测试过程中的所有文档。

## 1.5理解软件测试对象

源程序、目标程序、数据及相关文档



# 软件测试类型

## 2.1理解单元测试、集成测试、系统测试

1、单元测试：针对一个软件单元的测试

1）测试重点：功能性测试、健壮性（逆向测试：无效值）、性能

2）典型技术：白盒测试技术（语句、分支覆盖）

黑盒测试技术（等价类划分、边界值分析等）

3）发现缺陷：在组件内的功能性缺陷、运行时的缺陷、（本机）性能问题、健壮性问题

4）遗漏缺陷：接口问题，‘大环境’中的问题，许多非功能性缺陷

2、集成测试：一种旨在暴露接口以及集成组件/系统间交互时存在的缺陷的测试

1. 测试重点：接口，系统内不同部分的相互作用
2. 典型技术：白盒测试技术（内部接口）

黑盒测试技术（等价类划分）

1. 发现缺陷：接口的缺陷，组件间的协调错误
2. 可能遗漏的缺陷：相关组件外的问题、对整个系统的需求的不满足、与外部系统的接口常常被忽略
3. 集成测试的策略：自顶向下的集成、自底向上的集成

## 2.2理解确认测试、验收测试

系统测试：

## 2.3理解开发方测试、用户测试、第三方测试

## 2.4理解动态测试、静态测试

* 静态测试：不执行程序的测试方法，主要用于测试文档和代码（文档）

静态测试方法包括评审和静态分析方法

* 评审的分类：

文档审查

代码审查

代码走查

* 静态分析方法：

符合编程的原则和标准

与控制流（control flow）结合 ：控制流分析

程序代码复杂度：复杂度分析

静态测试可以发现的缺陷：

* 引用一个没有定义值的变量；
*  从未使用的变量；
*  模块和组件之间接口不一致；
*  不可达代码（unreachable code）或死代码（dead code）；
*  违背编程规则；
*  安全漏洞；
*  代码和软件模型的语法错误等
* 动态测试：通过运行程序来发现缺陷的测试方法。黑盒测试和白盒测试

## 2.5理解白盒测试、黑盒测试、灰盒测试

黑盒测试：

* 也称为功能测试、数据驱动测试、基于规格说明书测试。
* 从用户观点出发，主要以软件需求规格说明书为依据，对程序功能和接口进行测试，对输入输出数据之间的对应关系进行测试。
* 它不涉及到程序的内部结构，如果外部特性本身有问题或规格需求说明书有问题，则无法察觉。
* 安全性测试、互操作性测试也属于功能测试。
* 方法如大纲法、场景法、等价类、边界值、决策表、错误猜测等。
* 黑盒测试方法还用于测试软件的非功能性特性
* 非功能测试用于测试系统工作的怎么样，包括但不限于
* 可用性/可靠性/稳定性/健壮性/可恢复性测试
* 可维护性测试
*  易用性测试
*  可移植性/兼容性测试
*  配置测试
*  文档测试
*  国际化测试/本地化测试
* 当不涉及程序内部结构时，上述测试类型也使用黑盒测试方法

白盒测试：

也称结构测试、逻辑驱动测试、基于程序本身的测试、程序员测试。

结构测试需要完全了解程序内部机构和处理过程，按照程序内部逻辑测试程序，检测程序中每条通路是否按照预定要求工作。

# 开发与运行阶段的测试

## 3.1理解单元测试

单元测试用例的设计方法：

以白盒测试方法为主，并适当地结合黑盒测试方法

## 3.2理解集成测试

## 3.3理解系统（确认）测试

## 3.4理解验收测试

# 测试用例设计方法

## 4.1白盒测试设计

### **4.1.1了解白盒测试基本技术**

白盒测试方法的步骤

1. 获得需求、获得/画出程序流程图/算法图
2. 画出控制流图
3. 选择覆盖方法测试用例

### **4.1.2了解白盒测试方法**

* + - 逻辑覆盖法
* 语句覆盖
* 判定覆盖
* 条件覆盖
* 判定-条件覆盖
* 条件组合覆盖
  + - 路径覆盖法

## 4.2黑盒测试用例设计

### **4.2.1理解测试用例设计方法**

\*测试用例：就是将软件测试的行为活动，作一个科学化的组织归纳。 （即，测试用例就是设计一个情况，软件程序在这种情况下，必须能够正常运行并且达到程序所设计的执行结果。如果在此情况下不能正常运行且问题会重复发生，则表示有缺陷，需将此问题标注出来反馈给开发人员修改，版本更新后，需要利用同一个用例来测试之前的问题，确保问题已被修复）

\*设计测试用例的目的：为了能将软件测试行为转换为可管理的模式。

（测试用例可将测试行为具体量化，以便进一步让管理阶层掌握所需要的测试过程）

\*使用测试用例的好处：可避免盲目测试并提高测试效率；

令软件测试的实施重点突出、目的明确；

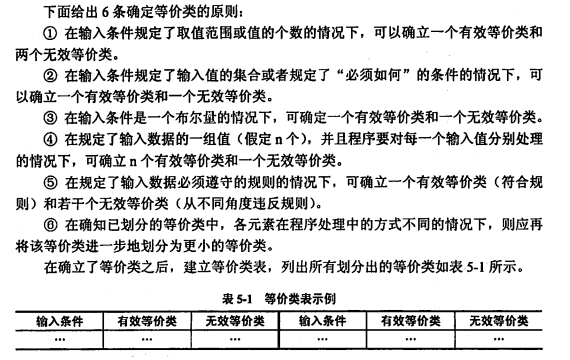
版本更新后只需修正少部分的用例便可展开测试工作，降低工作强度，缩短项目周期；

功能模块的通用化和复用化使软件易于开发，测试用例的通用化和复用化使软件测试易于开展，并随着测试用例的不断精化，其效率也不断攀升。

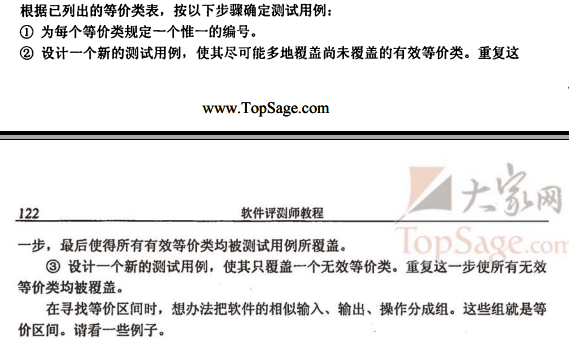
\*黑盒测试用例设计方法包括：等价类划分法、错误推测法、因果图法、判定表驱动法、 正交试验设计法、功能图法等。

\*等价类划分法：把程序的输入域划分成若干部分，然后从每个部分中选取少数代表性数据作为测试用例。

步骤：1）必须在分析需求规格说明的基础上划分等价类，列出等价类表，如：有效等价类和无效等价类；



2）确定测试用例



（完全不考虑程序内部结构，只根据需求规格说明书，仔细分析和推敲，然后把说明书中对输入的要求和输出的要求区别开来并加以分解）

\*边界值分析法：相对输入等价类和输出等价类而言，稍高于其边界值及稍低于其边界值的一些特定情况。

1）边界条件

2）

### **4.2.2理解测试用例的编写**

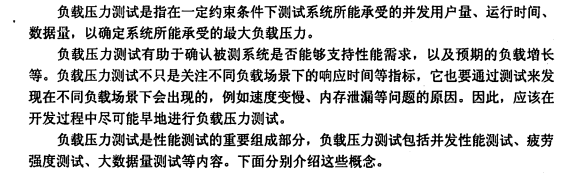
## 4.3测试方法选择的策略

### **4.3.1理解黑盒测试方法选择策略**

### **4.3.2了解白盒测试方法选择策略**

## 4.4负载压力测试

### **4.4.1理解性能/压力测试基本概念**



### **4.4.2理解性能/压力测试解决方案**

\*并发性能测试

\*疲劳强度测试

\*大数据量测试

### **4.4.3理解性能/压力测试指标分析**

\*交易处理性能指标

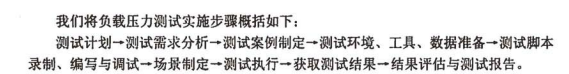
\*服务器操作系统资源监控

\*数据库资源监控

\*web服务器监控

\*中间件服务器监控

### **4.4.4理解性能/压力测试实施**



# Web应用测试

## 5.1理解链接测试

链接是web应用系统的一个主要特征，它是在页面之间切换和指导用户去一些不知道地址的页面的主要手段。

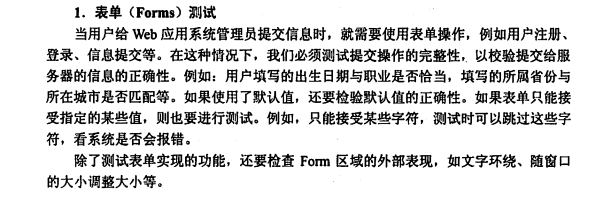
\*分为三个方面：首先，测试所有链接是否按指示那样确实链接到了该链接的页面；

其次，测试所链接的页面是否存在；

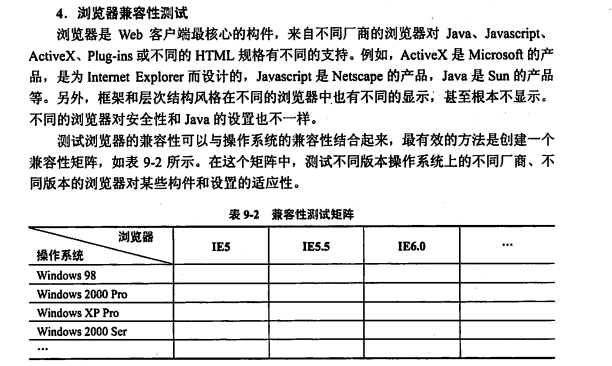
最后，保证web应用系统上没有孤立的页面，即没有正确链接指向该页面。

测试时，可逐一检查链接的有效性、可达性、正确性等。

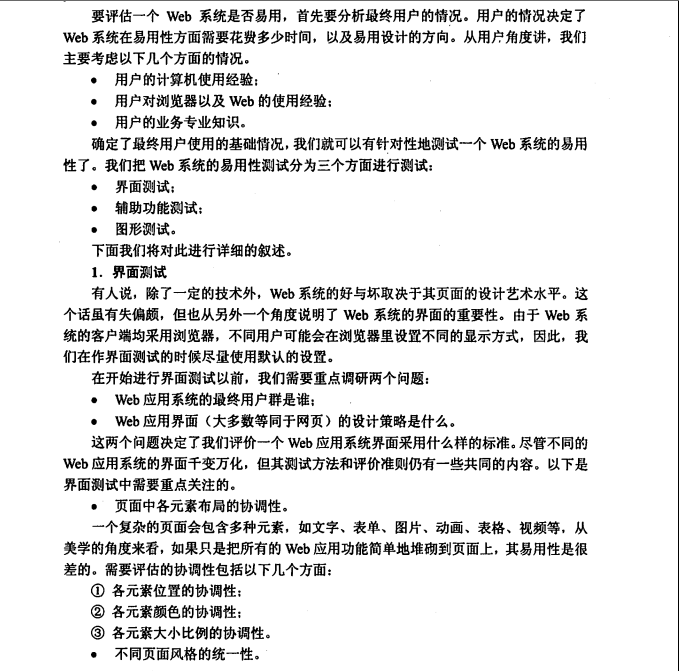
## 5.2理解表单测试

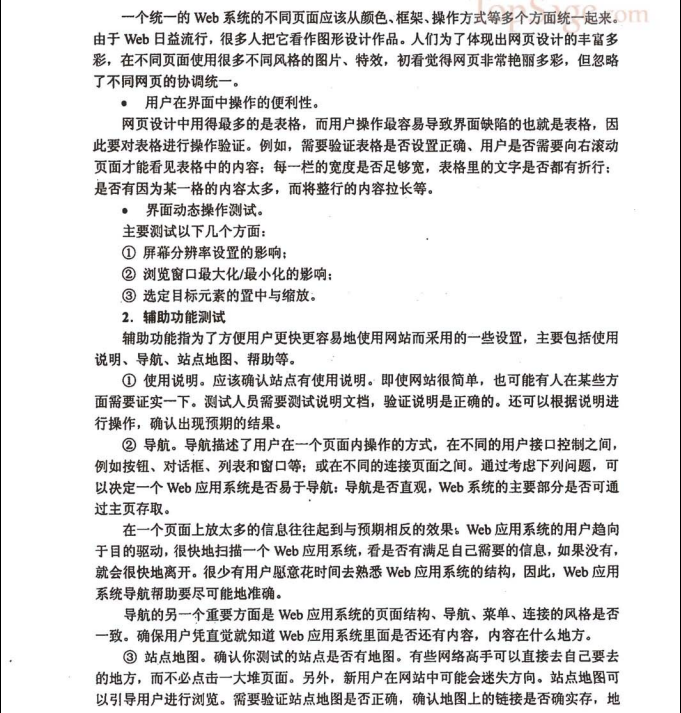


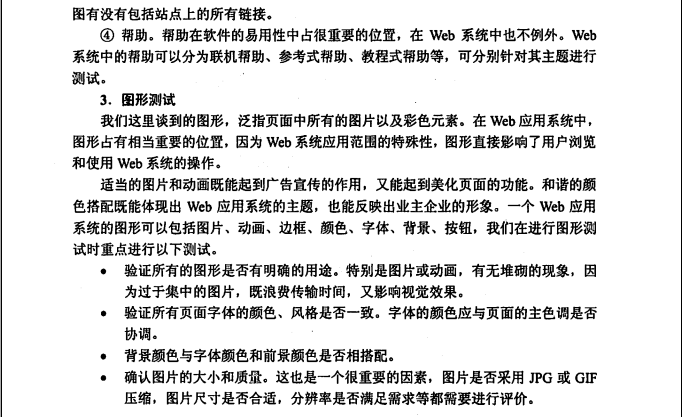
## 5.3理解浏览器兼容性测试



## 5.4理解其他测试（界面输入、风格、布局、式样、颜色搭配）



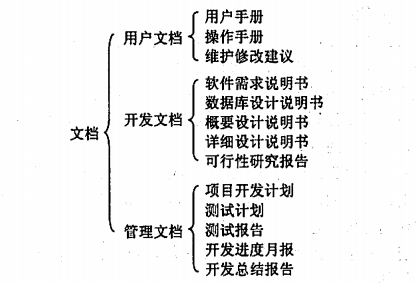


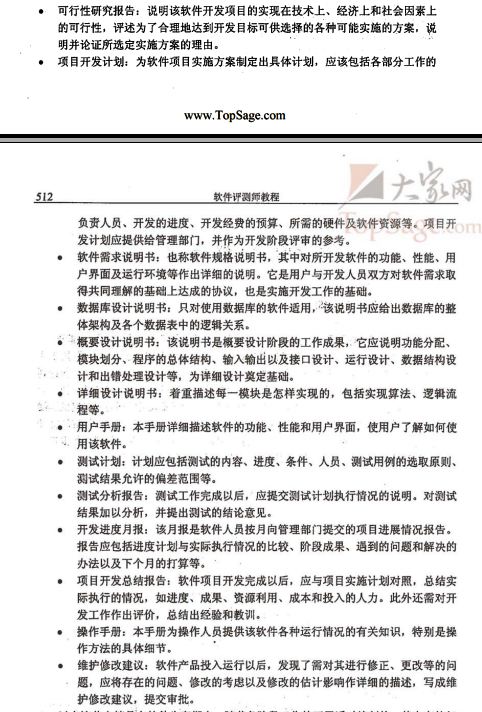


# 文档测试

## 6.1理解文档测试的范围

\*软件产品由可运行的程序、数据和文档组成。文档是软件的一个重要组成部分。

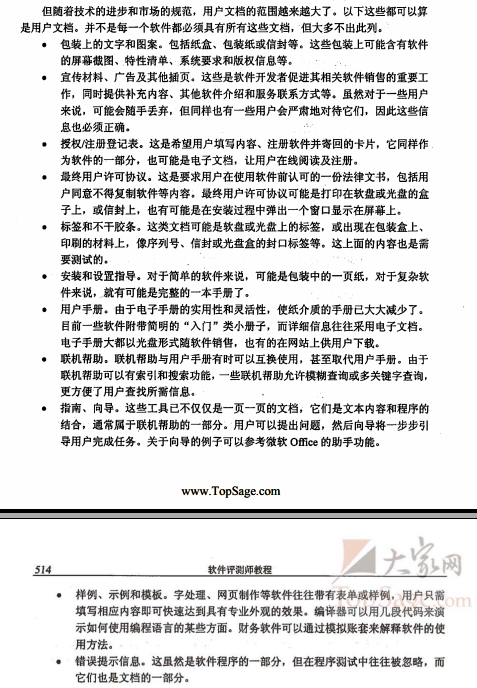




## 6.2理解用户文档的内容

当用户文档仅包含一个Readme文件时，只需对此进行拼写检查，确认涉及到的技术准确无误，最多对Readme文件进行病毒扫描，确保其不带病毒即可。

一般用户文档内容包括：包装上的文字图案；宣传材料、广告及其他插页；授权/注册登记表；最终用户许可协议；标签和不干胶条；安装和设置指导；用户手册；联机帮助；指南、向导；样例、示例和模板；错误提示信息。



用户文档的作用：改善易安装性、提高软件的易用性、改善软件可靠性、促进销路、降低技术支持的费用。

## 6.3理解用户文档测试的要点

\*读者群

\*术语

\*正确性

\*完整性

\*一致性

\*易用性

\*图表与界面截图

\*样例和示例

\*语言

\*印刷与包装

## 6.4理解用户手册的测试

\*准确按照手册的描述使用程序

\*尝试每一条建议

\*检查每条陈述

\*查找容易误导用户的内容

## 6.5理解在线帮助的测试

很大程序上与用户手册测试相同，但帮助并不只是用户手册的电子版，还应包含以下几点补充说明：

\*准确性

\*帮助是文档编写和程序编写的结合

\*帮助索引

\*超链接

\*链接的意义

\*帮助的风格