# 软件测试的概念

## 软件的6大质量特性

功能性、可靠性、可用性、效率、可维护、可移植

## 软件测试的定义

使用人工和自动手段来运行或测试某个系统的过程，其目的在于检验它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。

## 测试目的

尽可能发现并排除软件中潜藏的错误，提高软件的可靠性。

## 测试用例

1. **定义：**

测试用例是为了某个特定的目标而编制一组测试输入、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 = 输入 + 输出 + 测试环境。

测试环境 = 硬件 + 软件 + 网络 + 历史数据。

1. **重要性：**
   1. 指导测试的实施
   2. 规划测试数据的准备
   3. 编写测试脚本的《设计规格说明书》
   4. 降低工作强度
2. **测试用例的标准：**
   1. 有效性
   2. 经济性
   3. 可仿效性
   4. 可修改性
   5. 独立性
   6. 可跟踪性
3. **设计的基本原则**
   1. 测试用例的代表性
   2. 测试结果的可判断性
   3. 测试结果的可再现性

## 测试原则

* 1. 所有的软件测试都应追溯到用户需求
  2. 应当把“尽早地和不断地进行软件测试”作为测试者的座右铭
  3. 完全测试是不可能的，测试需要终止
  4. 测试无法显示软件潜在的缺陷；
  5. 充分注意测试中的群集现象
  6. 程序员应避免检查自己的程序
  7. 尽量避免测试的随意性
  8. 妥善保存测试计划、[测试用例](https://baike.baidu.com/item/%E6%B5%8B%E8%AF%95%E7%94%A8%E4%BE%8B)、出错统计和最终分析报告，为维护提供方便。

## 测试对象

程序开发过程中的各个文档、源程序。

## 测试阶段

[**单元测试**](https://baike.baidu.com/item/%E5%8D%95%E5%85%83%E6%B5%8B%E8%AF%95)

单元测试是对软件组成单元进行测试，其目的是检验软件基本组成单位的正确性，测试的对象是软件设计的最小单位：模块。

[**集成测试**](https://baike.baidu.com/item/%E9%9B%86%E6%88%90%E6%B5%8B%E8%AF%95)

集成测试也称联合测试，将程序模块采用适当的集成策略组装起来，对系统的接口及集成后的功能进行正确性检测的测试工作。其主要目的是检查软件单位之间的接口是否正确，集成测试的对象是已经经过单元测试的模块。

[**系统测试**](https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E6%B5%8B%E8%AF%95)

系统测试主要包括功能测试、界面测试、可靠性测试、易用性测试、性能测试。 功能测试主要针对包括功能可用性、功能实现程度（功能流程&业务流程、数据处理&业务数据处理）方面测试。

**验收测试**

测试内容：根据任务书或合同、供需双方约定的验收依据文档进行对整个系统的测试与评审，确认是否接收或拒绝系统；

[**回归测试**](https://baike.baidu.com/item/%E5%9B%9E%E5%BD%92%E6%B5%8B%E8%AF%95)

回归测试指在软件维护阶段，为了检测代码修改而引入的错误所进行的测试活动。回归测试是软件维护阶段的重要工作，有研究表明，回归测试带来的耗费占软件生命周期的1/3总费用以上。

与普通的测试不同，在回归测试过程开始的时候，测试者有一个完整的测试用例集可供使用，因此，如何根据代码的修改情况对已有测试用例集进行有效的复用是回归测试研究的重要方向，此外，回归测试的研究方向还涉及自动化工具，面向对象回归测试，测试用例优先级，回归测试用例补充生成等。

## 测试方法的分类

**白盒测试：**

通过对程序内部结构的分析、检测来寻找问题，检查程序的结构及路径是否正确，检查程序的内部动作是否按照设计说明的规定正常进行。

**黑盒测试：**

又称功能测试，通过运行程序发现其缺陷和错误，在程序界面处进行测试。

**灰盒测试：**

介于白盒和黑盒测试之间，关注输出对于输入的正确性，也关注程序的内部结构，但没有白盒测试那样详细、完整

## 测试方法

### 等价类

**1.定义**

是把所有可能的输入数据，即程序的输入域划分成若干部分（子集），然后从每一个子集中选取少数具有代表性的数据作为测试用例。该方法是一种重要的，常用的[黑盒测试](https://baike.baidu.com/item/%E9%BB%91%E7%9B%92%E6%B5%8B%E8%AF%95)用例设计方法。

**2.划分等价类**

*1)有效等价类*

是指对于程序的规格说明来说是合理的、有意义的输入数据构成的集合。利用有效等价类可检验程序是否实现了规格说明中所规定的功能和性能。

*2)无效等价类*

与有效等价类的定义恰巧相反。无效等价类指对程序的规格说明是不合理的或无意义的输入数据所构成的集合。对于具体的问题，无效等价类至少应有一个，也可能有多个。

设计测试用例时，要同时考虑这两种等价类。因为软件不仅要能接收合理的数据，也要能经受意外的考验，这样的测试才能确保软件具有更高的可靠性。

### 边界值

**1. 定义：**边界值分析法就是对输入或输出的边界值进行测试的一种黑盒测试方法。通常边界值分析法是作为对等价类划分法的补充，这种情况下，其测试用例来自等价类的边界。

**2. 与等价划分的区别**

1) 边界值分析不是从某等价类中随便挑一个作为代表，而是使这个等价类的每个边界都要作为测试条件。

2) 边界值分析不仅考虑输入条件，还要考虑输出空间产生的测试情况。

**4. 常见的边界值**

1) 对16-bit 的整数而言 32767 和 -32768 是边界

2) 屏幕上光标在最左上、最右下位置

3) 报表的第一行和最后一行

4) 数组元素的第一个和最后一个

5) 循环的第 0 次、第 1 次和倒数第 2 次、最后一次

**5. 边界值分析**

1) 边界值分析使用与等价类划分法相同的划分，只是边界值分析假定错误更多地存在于划分的边界上，因此在等价类的边界上以及两侧的情况设计测试用例。

6

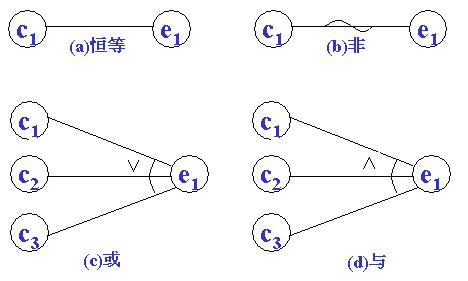
### 决策表方法

**决策表的组成**

* 1. 条件桩：列出问题的所有条件，除了某些问题对条件的先后次序有特定要求外，通常在这里列出的条件先后次序无关紧要。
  2. 条件项：针对条件桩给出的条件列出所有可能的取值。
  3. 动作桩：列出了问题规定可能采取的操作。这些操作的排列顺序没有约束。
  4. 动作项：列出在条件项的各种取值情况下应该采取的动作。

### 因果图方法

1、4种符号分别表示了规格说明中向4种因果关系。

****

1. 因果图中使用了简单的逻辑符号，以直线联接左右结点。左结点表示输入状态（或称原因），右结点表示输出状态（或称结果）。
2. ci表示原因，通常置于图的左部；ei表示结果，通常在图的右部。ci和ei均可取值0或1，0表示某状态不出现，1表示某状态出现。

**因果图关系**

1. 恒等：若ci是1，则ei也是1；否则ei为0。
2. 非：若ci是1，则ei是0；否则ei是1。
3. 或：若c1或c2或c3是1，则ei是1；否则ei为0。“或”可有任意个输入。
4. 与：若c1和c2都是1，则ei为1；否则ei为0。“与”也可有任意个输入。

**因果图约束**

A.输入条件的约束有以下4类：  
① E约束（异）：a和b中至多有一个可能为1，即a和b不能同时为1。  
② I约束（或）：a、b和c中至少有一个必须是1，即 a、b 和c不能同时为0。  
③ O约束（唯一）；a和b必须有一个，且仅有1个为1。  
④R约束（要求）：a是1时，b必须是1，即不可能a是1时b是0。  
  
B.输出条件约束类型  
输出条件的约束只有M约束（强制）：若结果a是1，则结果b强制为0。

## 测试流程

1、制定测试计划

2、编辑测试用例

3、执行测试用例

4、发现并提交BUG

5、开发组修正BUG

6、对已修正BUG进行返测

7、修正完成的BUG将状态置为已关闭，未正确修正的BUG重新激活

## 测试模型

### 软件测试过程模型－V模型**：**

是软件开发瀑布模型的变种，主要反映测试活动与分析和设计的关系；

局限性：把测试作为编码之后的最后一个活动，需求分析等前期产生的错误直到后期的验收测试才能发现

### ****软件测试过程模型－W模型****

在V模型的基础上，增加千开发阶段的同步测试，形成W模型；测试与开发同步进行，有利用尽早的发现问题

局限性：仍把开发活动看成是从需求开始到编码结束的串行活动，只有上一阶段完成后，才可以开始下一阶段的活动，不能支持迭代，自发性以及变更调整

### ****软件测试过程模型－H模型****

在V模型的基础上，增加千开发阶段的同步测试，形成W模型；测试与开发同步进行，有利用尽早的发现问题

局限性：仍把开发活动看成是从需求开始到编码结束的串行活动，只有上一阶段完成后，才可以开始下一阶段的活动，不能支持迭代，自发性以及变更调整

**健壮性**: 是指在异常情况下，软件还能正常运行的能力