**交互测试**

**一.功能测试**

**1.定义：**

Functional testing（功能测试），也称为behavioral testing（行为测试），根据产品特性、操作描述和用户方案，测试一个产品的特性和可操作行为以确定它们满足设计需求。本地化软件的功能测试，用于验证应用程序或网站对目标用户能正确工作。使用适当的平台、浏览器和测试脚本，以保证目标用户的体验将足够好，就像应用程序是专门为该市场开发的一样。功能测试是为了确保程序以期望的方式运行而按功能要求对软件进行的测试，通过对一个系统的所有的特性和功能都进行测试确保符合需求和规范。

功能测试也叫黑盒测试或数据驱动测试，只需考虑需要测试的各个功能，不需要考虑整个软件的内部结构及代码.一般从软件产品的界面、架构出发，按照需求编写出来的测试用例，输入数据在预期结果和实际结果之间进行评测，进而提出更加使产品达到用户使用的要求。

**2.黑盒测试：**

[黑盒测试](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=126595&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)（Black-box Testing，又称为功能测试或[数据驱动测试](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=168028352&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)）是把测试对象看作一个[黑盒子](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=846279&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)。利用[黑盒测试法](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=7476029&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)进行动态测试时，需要测试软件产品的功能，不需测试软件产品的内部结构和处理过程。

比如黑盒技术设计测试用例的方法有：等价类划分、边界值分析、错误推测、[因果图](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=204053&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)和综合策略。

黑盒测试注重于测试软件的功能性需求，也即黑盒测试使[软件工程师](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=316626&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)派生出执行程序所有功能需求的输入条件。黑盒测试并不是[白盒测试](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=87714&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)的替代品，而是用于辅助白盒测试发现其他类型的错误。

黑盒测试试图发现以下类型的错误：

（1）功能错误或遗漏

（2）界面错误

（3）数据结构或外部数据库访问错误

（4）性能错误

（5）初始化和终止错误

### 用例设计

（1）等价类划分方法

（2）边界值分析方法

（3）错误推测方法

（4）因果图方法

（5）[判定表](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=470635&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)驱动分析方法

（6）[正交实验设计](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=7951721&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)方法

（7）[功能图](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=76638799&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)分析方法

**等价类划分**

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[测试用例设计](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=7706544&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)方法。

**划分等价类**

等价类是指某个输入域的[子集合](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=103498&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank).在该子集合中，各个输入数据对于揭露程序中的错误都是等效的.并合理地假定：测试某等价类的代表值就等于对这一类其它值的测试.因此，可以把全部输入数据合理划分为若干等价类，在每一个等价类中取一个数据作为测试的输入条件，就可以用少量代表性的测试数据.取得较好的测试结果.等价类划分可有两种不同的情况：有效等价类和[无效等价类](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=63203076&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)。

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。

无效等价类：与有效等价类的定义恰巧相反。

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。

**方法**

下面给出六条确定等价类的原则。

①在输入条件规定了取值范围或值的个数的情况下，则可以确立一个有效等价类和两个无效等价类。

②在输入条件规定了输入值的集合或者规定了“必须如何”的条件的情况下，可确立一个有效等价类和一个无效等价类。

③在输入条件是一个布尔量的情况下，可确定一个有效等价类和一个无效等价类。

④在规定了输入数据的一组值（假定n个），并且程序要对每一个输入值分别处理的情况下，可确立n个有效等价类和一个无效等价类。

⑤在规定了输入数据必须遵守的规则的情况下，可确立一个有效等价类（符合规则）和若干个无效等价类（从不同角度违反规则）。

⑥在确知已划分的等价类中各元素在程序处理中的方式不同的情况下，则应再将该等价类进一步的划分为更小的等价类。

**3.测的内容：**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 测试项 |  |  |  |  |  |  |  |  |  |
| 检查项 |  |  |  |  |  |  |  |  |  |
| 前置条件 |  |  |  |  |  |  |  |  |  |
| 操作步骤 |  |  |  |  |  |  |  |  |  |
| 期待结果 |  |  |  |  |  |  |  |  |  |
| 优先级 |  |  |  |  |  |  |  |  |  |
| 真实结果 |  |  |  |  |  |  |  |  |  |