**1.概述**

黑盒测试也称功能测试或数据驱动测试，它是在已知产品所应具有的功能，通过测试来检验每个功能都是否能够正常使用。

在测试时，把程序看作一个不能打开的黑盒子，在完全不考虑程序内部结构和内部特性的情况下，测试人员进行直接测试，检查系统功能是否按照需求规格说明书的规定正常使用、是否能适当的接收输入数据而输出正确的结果等，检查相应的文档是否采用了正确的模板、是否满足规范需求。

**2. “黑盒”的两种基本方法**

黑盒测试有两种基本方法，即“通过测试”和“失败测试”。

在进行“通过测试”时，实际上是确认软件能做什么，而不会考验其能力如何。软件测试员只运用最简单、直观的测试案例。

在设计和执行测试案例时，总是先进行通过测试。在进行破坏性试验之前，看一看软件基本功能是否能够实现。

在确信了软件正确运行之后，就可以采取各种手段通过搞垮软件来找出缺陷。纯粹为了破坏软件而设计和执行的测试案例，被称为失败测试或迫使出错测试。

**3.黑盒测试的优、缺点**

**(1）优点**

a.比较简单，不需要了解程序内部的代码以及实现。

b.与软件内部实现无关。

c.从用户角度出发，很容易的知道用户会用到哪些功能，会遇到哪些问题。

d.在做软件自动化测试时较为方便。

e.基于软件开发文档，所以也能知道软件实现了文档中的哪些功能。

**(2)缺点**

a.不可能覆盖所有代码，覆盖率较低，大概只能达到总代码量的30%。

b.自动化测试的复用性较低。

**4.黑盒测试的测试用例设计方法**

（1）等价类划分法

（2）边界值分析法

（3）错误推测法

（4）因果图法

（5）判定表

（6）功能图法

**4.1等价类划分法**

**（1）概念**

等价类划分法就是解决如何选择适当的数据子集来代表整个数据集的问题，通过降低测试的数目去实现“合理的”覆盖，覆盖了更多的可能数据，以发现更多的软件缺陷。

等价类是指某个输入域的一个特定的子集合，在该子集合各个输入数据对于揭露程序中的错误都是等效的。

**（2）等价类划分的过程**

a.分类：即将输入域按照具有相同特性并用实例来表征这个特性。

b.抽象：即在各个子类中去抽象出相同特性并用实例来表征进行分类。

**（3）等价类划分的优缺点**

a.优点：基于相对较少的测试用例，就能够进行完整覆盖，很多程度上减少了重复性。

b.缺点：缺乏特殊用例的考虑，同时需要深入的系统知识，才能选择有效的数据。

（**4）有效等价类和无效等价类**

（1）有效等价类是指输入完全满足程序输入的规格说明、有意义的输入数据所构成的集合，利用有效等价类，可以检验程序是否满足规格说明所规定的功能和性能。

（2）无效等价类和有效等价类相反，即不满足程序输入要求或者无效的输入数据构成的集合。使用无效等价类，可以测试程序/系统的容错性——对异常输入情况的处理。

**4.2边界值分析法**

**（1）概念**

边界值分析法就是在某个输入输入变量范围的边界上，验证系统功能是否正常运行的测试方法。

**（2）与等价划分类的区别**

a.边界值分析不是从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。

b.边界值分析不仅考虑输入条件，还有考虑输出空间产生的测试情况。

**（3）边界值分析的处理技巧**

a.如果输入条件规定了值的范围，则取刚刚到达这个范围的边界值。

b.如果输入条件规定了值的个数，则用最大个数、最小个数、比最大个数多一个，比最小个数少一个的数等作为测试数据。

c.根据规格说明的每一个输出条件，分别使用以上两个规则。

d.如果程序的规格说明给出的输入域或输出域是有序集合（如有序表、顺序文件等），则应选取集合的第一个和最后一个元素测试数据。

**（4）边界值附近的数据确定的几种方法**

|  |  |  |
| --- | --- | --- |
| 项 | 边界值 | 测试用例的设计思路 |
| 字符 | 起始-1个字符/结束+1个字符 | 假设一个文本输入区域要求允许输入1-255个字符，输入1个和255个作为有效等价类，输入0个和256个作为无效等价类，这几个数值都属于边界条件值。 |
| 数值 | 开始位-1/结束位-1 | 例如，数据的输入域为1-999，其值最小为1，最大值为999,1000刚好在边界值附近。 |
| 方向 | 刚刚超过/刚刚低于 |  |
| 空间 | 小于空余空间一点/大于空余空间一点 | 例如，测试数据存储时，使用比最小剩余空间大一点（几KB）的文件作为最大值检验的边界条件。 |

**4.3错误推测法**

**（1）概念**

错误推测法是基于经验和直觉推测程序中所有可能存在的各种情况，从而有针对性的设计，列出测试用例的方法。

错误推测法基于这样一个思想，在某处发现了缺陷，则可能隐藏更多的缺陷。在实际操作中列出所有可能出现的错误和容易发现错误的地方，然后语句测试者经验做出选择。

**（2）优点与缺点**

优点：充分发挥人的直觉与经验，在一个测试小组中集思广益，方便实用，特别是在软件测试基础较差的情况下，很好地组织测试小组进行错误推测，是一种有效的测试方法。

使测试者能够快速且容易地切入，并且能够体会到程序的易用与否。

缺点：错误推测法不是一个系统的测试方法，所以只能用着辅助手段，即先用上述方法设计测试用例，在没有其他方法可用的情况下，再采用错误推测法，补充一些例子进行额外的测试。

难以知道测试的覆盖率，可能丢失大量未知的区域，并且这种测试行为带有主观性且难以复制。

**4.4功能图法**

**（1）概念**

功能图法是用功能图FD形式化地表示程序的功能说明，并机械地生成功能图的测试用例。

功能图模型由状态迁移图和逻辑功能模型构成。状态迁移用于表示输入数据序列以及相应的输出数据，在状态迁移图中，由输入数据和当前状态绝对输出数据和后续状态。

**（2）功能图**

功能图由状态迁移图和布尔函数组成。状态迁移图用状态和迁移来描述，一个状态指出数据输入的位置（或时间），而迁移则指明状态的改变，同时要依靠判定表或因果图表示的逻辑功能。

**（3）测试用例的生成方法**

从功能图生成测试用例，得到的测试用例数是可接受的。问题的关键是如何从状态迁移图中选取测试用例，若用节点代替状态，用弧线代替迁移，则状态迁移图可转化成一个程序的控制流程图形式。问题就转化称为程序的路径测试了。

**（4）测试用例生成规则**

为了将状态迁移（测试路径）的测试用例与逻辑模型（局部测试用例）的测试用例组合起来，从功能图生成实用的测试用例，须定义以下规则：在一个结构化的状态迁移中，定义三种形式的循环结构：顺序、选择和重复。但分辨一个状态迁移中所有循环是有困难的。

（5）从功能图中生成测试用例的过程

a.生成局部测试用例：在每个状态中，从因果图生成局部测试用例，局部测试用例由原因值（输入数据）组合与对应结果值（生输出数据或状态）组成。

b.测试路径生成：利用上面的规则（三种）生成从初始状态到最后状态的测试路径。

c.测试用例合成：合成测试路径与功能图中每个状态局部测试用例，结果是初始状态到最后状态的一个状态序列，以及每个状态中输入数据与对应的输出数据的组合。