软件测试的用例设计

胡志勇 高大葳 杨志杰 田玉琳

〔摘 要〕 本文介绍了白盒测试法、黑盒测试法和软件测试的五种用例设计方法。

〔关键词〕 软件测试;测试用例;黑盒测试;白盒测试

Test Case Design for Software

HU zhi - yong GAO Da - wei YANG Zhi - jie TIAN Yu - lin

Abstract: We introduce Blackbox Test, Whitebox Test and five kinds of test case methods for Software.

Keywords: Software Test; Test Case; Blackbox Test; Whitebox Test

1 引言

随着今天软件的规模和复杂性的日益增加,因特网技术的普遍应用,以及各个行业信息化建设步伐的不断加快,进行专业化高效软件测试的要求会越来越迫切,挑战性也会越来越强。本文将从测试用例的角度,系统地说明白盒测试技术、黑盒测试技术。

2 软件测试

2.1 软件测试的概念和目的

软件测试就是在软件投入运行前,对软件需求分析、设计规格说明和编码的最终复审,是软件质量保证的关键步骤。软件测试不等于程序测试;软件测试贯串于软件定义和开发的整个期间;需求规格说明、概要设计规格说明、详细设计规格说明、源程序都是软件测试的对象。

测试的目的是寻找错误,并且是尽最大可能找出最多的错误。好的测试方案是极可能发现至今为止尚未发现的错误的测试方案。成功的测试是发现了至今为止尚未发现的错误的测试。

2.2 软件缺陷

所有软件的问题通称为软件缺陷。只要符合下列5条规则中的任何一条,就叫做软件缺陷:1)软件未达到产品说明书中已经标明的功能;2)软件出现了产品说明书中指出不会出现的错误;3)软件功能超出了产品说明书指明的范围;4)软件未达到产品说明书虽未指出但应达到的目标;5)软件测试员认为软件难以理解、不易使用、运行速度缓慢,或者最终用户认为该软件使用效果不好。

据研究表明软件缺陷产生的原因依次为产品说明书、设计方案、代码和其它(见图1)。

2.3 软件测试的分类

按测试用例设计方法:白盒测试、黑盒测试;按测试策略和过程:单元测试、集成测试、确认测试、系统测试。

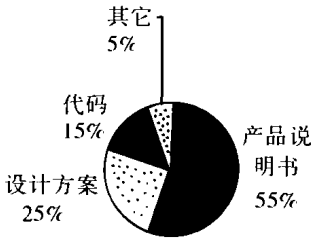


图 1 软件缺陷产生的原因

2.3.1 黑盒测试法

黑盒测试法把程序看成一个黑盒子,完全不考虑程序内部结构和处理过程。黑盒测试是在程序接口进行测试,它只是检查程序功能是否按照规格说明书的规定正常使用。黑盒测试又称功能测试。

黑盒主要是为了发现以下几类错误:是否有不正确或遗漏了的功能?在接口上,输入能否正确地接受?能否输出正确的结果?是否有数据结构错误或外部信息(例如数据文件)访问错误?性能上是否能够满足要求?是否有初始化或终止性错误?

常见的黑盒测试法有等价类划分法、边界值分析法、错误推测法、因果图法、比较测试法、功能图法等。

2.3.2 白盒测试法

白盒测试的前提是可把程序看成装在一个透明的白盒子里,也就是完全了解程序结构和处理过程,这种方法按照程序内部逻辑测试程序,检验程序中每条通路是否按预定要求正确工作。白盒测试又称结构测试。

使用白盒测试方法,主要想对程序模块进行如下的检查:对程序模块的所有独立的执行路径至少测试一次。对所有的逻辑判定,取"真"与取"假"的两种情况者能少测试一次。在循环的边界和运行界限内执行循环体。测试内部数据结构的有效性等。

常见的白盒测试法有逻辑覆盖测试法、路径测试法、程序结构分析法等。

3 测试用例

3.1 测试用例的概念

为达到最佳的测试效果或高效的揭露隐藏的错误而精心设计的少量测试数据,称之为测试用例。由于输入量太大、输出结果太多、软件实现途径太多、软件说明书没有客观标准等原因,我们不可能进行完全测试。为了节省时间和资源,提高测试效率,必须要从数量极大的可用测试数据中精心挑选出具有代表性或特殊性的测试数据来进行测试。一个好的测试用例是在于它能发现至今未发现的错误。

3.2 测试用例的作用

在开始实施测试之前设计好测试用例,可以避免盲目测试并提高测试效率。测试用例的使用令软件测试的实施重点突出、目的明确。在软件版本更新后只需修理部分的测试用例即可展开测试工作,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,而相对于功能模块的测试用例的通用化和复用化则会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。

3.3 测试用例的选用策略

1) 在任何情况下都必须使用边界值分析方法。经验表明用这种方法设计出测试用例发现程序错误的能力最强。

2) 必要时用等价类划分方法补充一些测试用例。

3) 用错误推测法再追加一些测试用例。

4) 对照程序逻辑,检查已设计出的测试用例的逻辑覆盖程度。如果没有达到要求的覆盖标准,应当再补充足够的测试用例。

5) 如果程序的功能说明中含有输入条件的组合情况,则一开始就可选用因果图法。

4 测试用例的设计

4.1 等价类划分法

等价类划分的办法是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当作测试用例,每一类的代表性数据在测试中的作用等价于这一类中的其他值,也就是说,如果某一类中的一个例子发现了错误,这一等价类中的其他例子

也能发现同样的错误;反之,如果某一类中的一个例子没有发现错误,则这一类中的其他例子也不会查出错误。怎样划分等价类,有下列原则:

1)如果输入条件规定了取值的范围或值的个数,则可确定一个有效等价类和两个无效等价类。

2)如果一个输入条件说明了一个"必须成立"的情况。

3)如果输入条件规定了输入数据的一组可能的值,而且程序是用不同的方式处理每一种值,则可为每一种值划分一个有效等价类,并划分一个无效等价类。

4)如果我们确知,已划分的某等价类中的各元素(例子)在程序中的处理方式是不同的,则应据此将此等价类进一步划分成更小的等价类。

5)在确立了等价类之后,建立等价类表,列出所有划分出的等价类:

输入条件	有效等价类	无效等价类
…	…	…
…	…	…

确定等价类测试用例的步骤如下:

1)为每个等价类规定一个惟一的编号

2)设计一个新的测试用例,使其尽可能多地覆盖未覆盖的有效等价类。重复这一步,最后使得所有有效等价类均被测试用例所覆盖;

3)设计一个新的测试用例,使其只覆盖一个无效等价类,重复这一步使所有无效等价类均被覆盖。

4.2　边界值分析法

边界值分析法是一种补充等价划分的测试用例设计技术,它不是选择等价类的任意元素,而是选择等价类边界附近的处理必须给于足够的重视,为检验边界附近的处理专门设计测试用例,常常取得良好的测试效果。边界值分析法不仅重视输入条件边界,而且也从输出域导出测试用例。

边界值设计遵守的几条原则:

1)如果输入条件规定了取值范围,应以该范围的边界内及刚刚超范围的边界外的值作为测试用例,如a和b为边界,测试用例应当包含a和b及略大于a和略小于b的值;

2)若规定了值的个数,分别以最大、最小个数及稍小于最小、稍大于最大个数作为测试用例;

3)针对每个输出条件使用前面的第1和第2条原则;

4)如果程序规格说明中提到的输入或输出域是个有序的集合(如顺序文件、表格等)、就应注意选取有序集的第一个和最后一个元素作为测试用例;

5)分析规格说明,找出其他的可能边界条件。

4.3　因果图法

什么是因果图法:等价类划分方法和边界值分析法是着重考虑输入条件,并没有考虑到输入情况的各种组合,也没考虑到各个输入情况之间的相互制约关系。因果图方法的思路是:从用自然语言书写的程序规格说明的描述中找出因(输入条件)和果(输出或程序状态的改变),通过因果图转换为判定表。

因果图用法的几个步骤:

1)分析程序规格说明的描述中,哪些是原因,哪些是结果,原因常常是输入条件或是输入条件的等价类,而结果是输出条件;

2)分析程序规格说明的描述中语义的内容,并将其表示成连接各个原因与各个结果的"因果图";

3)由于语法或环境的限制,有些原因和结果的组合情况是不可能出现的,为表明这些特定的情况,在因果图上使用若干个特殊的符号标明约束条件;

4)把因果图转换成判定表;

5)为判定表中每一列表示的情况设计测试用例。

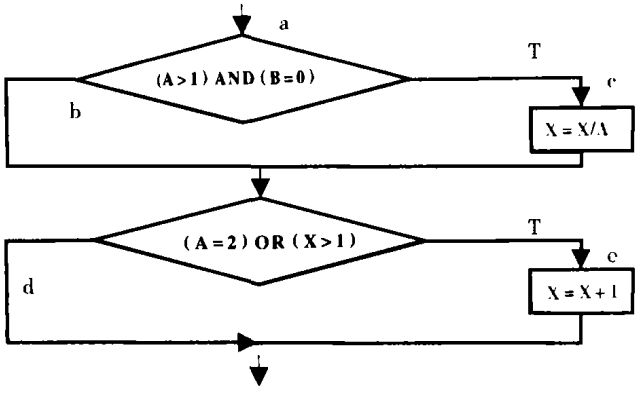
4.4　逻辑覆盖测试法

目前比较常见的逻辑覆盖测试法包括语句覆盖、判定覆盖、条件覆盖、判定/条件覆盖、组合覆盖和路径覆盖。下面用经典的小程序作为参考示例:

```
IF  ((A > 1)  AND  (B = 0))  THEN
        X = X/A
IF  ((A = 1)  OR  (X > 1))  THEN
```

X = X + 1

其对应的流程图如下：



为了清楚起见，分别对第一个判断的取假分支、取真分支及第二个判断取假分支和取真分支命名为 b、c、d 和 e。这样所有 4 条路径可表示为：L1(ace)、L2(abd)、L3(abe)及 L4(acd)。

对于第一个判断：条件 A > 1 取真值为 T1，取假值为¬ T1

条件 B = 0 取真值为 T2，取假值为¬ T2

对于第二个判断：条件 A = 2 取真值为 T3，取假值为¬ T3

条件 X > 1 取真值为 T4，取假值为¬ T4

测试用例的设计格式如下：[输入的(A,B,X)，输出的(A,B,X)]

4.4.1 语句覆盖

语句覆盖方法即足够多的测试用例，使程序中的每个可执行语句至少执行一次。满足语句覆盖的一组测试用例是：

[(2,0,4),(2,0,3)] 覆盖 L1(ace)

4.4.2 判定覆盖

判定覆盖是比语句覆盖较强一些的覆盖方法，即通过执行足够的测试用例，使得程序中的每个判定至少都获得一次“真”值和“假”值，也就是使程序中的每个取“真”分支和取“假”分支至少经历一次。

满足判定覆盖的一组测试用例是：

[(2,0,4),(2,0,3)] 覆盖 L1(ace)

[(1,1,1),(1,1,1)] 覆盖 L2(abd)

4.4.3 条件覆盖

条件覆盖的目的是设计足够的测试用例，在执行被测程序以后，要使每个判定中每个条件的可能取值至少满足一次。

满足条件覆盖的一组测试用例是：

[(1,0,4),(1,0,3)] 覆盖路径 L3(abe) 覆盖条件 ¬ T1、T2、¬ T3、T4 覆盖分支 b,e

[(2,1,1),(2,1,2)] 覆盖路径 L3(abe) 覆盖条件 T1、¬ T2、T3、¬ T4 覆盖分支 b,e

4.4.4 判定/条件覆盖

判定/条件覆盖要求设计足够的测试用例，使得判定中每个条件的所有(真/假)至少出现一次，而且每个判定本身的判定结果(真/假)至少出现一次。

满足判定/条件覆盖的一组测试用例是：

[(2,0,4),(2,0,3)] 覆盖路径 L1(ace) 覆盖条件 T1、T2、T3、T4 覆盖分支 c,e

[(1,1,1),(1,1,1)] 覆盖路径 L2(abd) 覆盖条件¬ T1、¬ T2、¬ T3、¬ T4 覆盖分支 b,d

4.4.5 组合覆盖

组合覆盖的目的是通过执行足够的测试用例，使得每个判定中条件的各种可能都至少出现一次。因此，满足组合覆盖的测试用例一定满足“判定覆盖”、“条件覆盖”、“判定/条件覆盖”。

记①A > 1，B = 0 作 T1、T2；　②A > 1，B < > 0 作 T1、¬ T2；

③A < = 1，B = 0 作¬ T1、T2；④A < = 1，B < > 0 作¬ T1、¬ T2；

⑤A = 2，X > 1 作 T3、T4；　⑥A = 2，X < = 1 作 T3、¬ T4；

⑦A < > 2，X > 1 作¬ T3、T4；⑧A < > 2，X < = 1 作¬ T3、¬ T4。

满足组合覆盖的一组测试用例是：

[(2,0,4),,(2,0,3)]覆盖路径 L1(ace) 覆盖条件 T1、T2、T3、T4 覆盖组合①，⑤

[(2,1,1),(2,1,2)]覆盖路径 L3(abe) 覆盖条件¬ T1、¬ T2、T3、¬ T4 覆盖组合②，⑥

[(1,0,3),(1,0,4)]覆盖路径 L3(abe) 覆盖条件¬ T1、T2、¬ T3、T4 覆盖组合③，⑦

[(1,1,1),(1,1,1)] 覆盖路径 L2(abd) 覆盖条件¬ T1、¬ T2、¬ T3、¬ T4 覆盖组合④, ⑧

4.4.6 路径覆盖

路径覆盖的目的是指设计足够多的测试用例,要求覆盖程序中所有可能的路径。路径能否全面覆盖是软件测试中的一个非常重要的问题。

满足路径覆盖的一组测试用例是:

[(2,0,4),(2,0,3)] 覆盖路径 L1(ace) 覆盖条件 T1、T2、T3、T4

[(1,1,1),(1,1,1)] 覆盖路径 L2(abd) 覆盖条件¬ T1、¬ T2、¬ T3、¬ T4

[(1,1,2),(1,1,3)] 覆盖路径 L3(abe) 覆盖条件¬ T1、¬ T2、¬ T3、T4

[(3,0,3),(3,0,1)] 覆盖路径 L4(acd) 覆盖条件 T1、T2、¬ T3、¬ T4

4.5 路径测试法

路径测试法就是从一个程序的入口开始,执行所经历的各个语句的完整过程。路径测试法常用于循环结构,对简单循环(设其循环的最大次数是 j)采用如下策略:

1) 跳出整个循环;

2) 只循环 1 次;

3) 循环 2 次;

4) 循环 i 次,其中 i < j;

5) 分别循环 n - 1,n 和 n + 1 次。

5 结语

在软件开发过程中,无论多么努力,软件缺陷总会出现。但我还是希望本文对诸位程序员有所帮助,使他们更快、更早的发现软件缺陷,降低改正成本。

参考文献

[1] 王健,苗勇,刘呈.软件测试员培训教材.电子工业出版社,2003.9.

[2] 冯玉琳,黄涛,金蓓弘.网络分布计算与软件工程.科学出版社,2003.5.

[3] 郑人杰,殷人昆,陶永雷.实用软件工程.清华大学出版社,1997.4.

[4] 郑人杰.软件工程.清华大学出版社,1999.8.

[5] 朱三元,钱乐秋,宿为民.软件工程技术概论.科学出版社,2002.1.

◀◀◀上接第 4 页◀

表 3 Φ6 × 10 样品测试结果

样品号	Br(KGs)	Hcb(Koe)	Hcj(Koe)	(BH)max(MGO)
1#	13.51	12.706	14.836	43.8
2#	13.56	12.665	14.836	44.2

6 结论

采用合适的氢破碎工艺,是制备小规格高性能烧结 NdFeB 产品的一条有效的途径。利用氢破碎工艺,打破了一次成型小规格产品的性能限制,无疑是高性能小规格产品制备工艺的一次创新。

参考文献:

[1] 肖耀福等.氢在 NdFeB 磁体制造中的应用.磁性材料及器件.Vol 32 No.2,2001.

[2] 李波等.NdFeB 母合金吸氢和脱氢过程的研究.金属功能材料.Vol.10 No.2.April,2003.

[3] 周寿增.超强永磁体.冶金出版社.