团队编号：202118

团队名称：泰山

**软件功能测试报告**

**RESUME**

目录

[第1章  软件测试  1](#_Toc10179)

[1.1  软件测试的概念  1](#_Toc1998)

[1.2  软件测试技术的广度和深度  1](#_Toc3716)

[1.2.1  按生命周期 1](#_Toc8012)

[1.2.2  按测试方法 2](#_Toc29162)

[1.2.3 按执行测试方式 2](#_Toc20513)

[1.3  黑盒测试理论  2](#_Toc17524)

[1.3.1  黑盒测试概念  2](#_Toc18179)

[1.3.2  黑盒测试的测试用例设计方法  3](#_Toc15216)

[第2章  黑盒测试流程  5](#_Toc22984)

[2.1  用例项  5](#_Toc31766)

[2.2  设计测试用例  5](#_Toc3229)

[第3章  测试表格 6](#_Toc26214)

[3.1  用例项划分 6](#_Toc8672)

[3.2 测试用例表格 6](#_Toc5151)

[3.3  关注点  8](#_Toc4552)

[3.3.1  文本输入框  8](#_Toc5194)

[3.3.2  增加数据  8](#_Toc28734)

[3.3.3  修改数据  8](#_Toc28710)

[3.3.4  删除数据  8](#_Toc14998)

[3.3.5  查询数据  9](#_Toc11390)

[3.3.6  数据导入导出  9](#_Toc5134)

[3.3.7  其他  9](#_Toc21172)

# 第1章  软件测试

## 1.1  软件测试的概念

软件测试其实应该是伴随着软件生产而产生，有了软件生产就必然有软件测试，但直到1957年，软件测试才和软件调试区分开来，软件测试的概念也有很多版本。测试目的演变如下：

（1）证明：表明软件能工作；

（2）检测：发现错误；

（3）预防：管理质量。

到上世纪80年代，软件质量“号角”吹响之后，软件测试的概念才逐步的稳定下来。1983年IEEE提出了软件工程标准术语定义如下：

“使用人工或自动化工具运行和测试某个系统的过程，目的在于验证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。”这个定义明确提出了软件测试以检验是否满足需求为目标。

有的方案需要采用复杂的测试设备和技巧才能完成，甚至需要专门的实验室。但对于一个要做出好产品的团队来说这些测试是必要的。总之开发人员通过设计来实现预期的需求目标，测试人员通过测试来验证预期的需求是否满足。

## 1.2  软件测试技术的广度和深度

对于复杂软件产品，特别是大型软件项目的测试就会涉及到很多技术环节，比如测试技术的广度和深度。

测试技术的广度体现在软件产品的种类繁多和业务领域的复杂性上。针对与不同的软件产品需要应用不同的软件测试技术。软件测试技术本身也是在不断发展，尤其是近几年。各种测试技术发展迅猛，这就使得软件测试知识体系越来越庞大。大体上软件测试可以分为以下类型和维度：

### 1.2.1  按生命周期

生命周期测试可分为单元测试，集成测试，系统测试，验收测试。

（1）单元测试：又称为模块测试,是针对软件结构中独立的基本单元进行测试。 是对单元设计文档的验证过程，通常在编码阶段进行的。

（2）集成测试：又称为组装测试，可以根据集成策略对软件模块进行组装后测试。

（3）系统测试：是将已经集成好的软件系统，作为整个基于计算机系统的一个 元素，与计算机硬件，外设，某些支持软件，数据和人员等其他系统元素结合在一起，在实际运行环境下，对计算机系统的一系列测试。系统测试根据测试类型又分为功能，性能，容量，GUI测试等测试技术。

### 1.2.2  按测试方法

测试方法可分为黑盒测试，白盒测试，灰盒测试。

（1）黑盒测试包括功能测试用例设计技术。

（2）白盒测试包括词法语法分析，静态错误分析，测试驱动技术，程序分析技术。

（3）灰盒是介于黑盒与白盒之间的测试方法。

### 1.2.3 按执行测试方式

执行测试方式可分为人工手动测试，自动化测试技术。

其中自动化测试技术又包含了功能自动执行和回归自动化测试，性能模拟自动化测试，数据自动生成，测试过程管理自动化技术，嵌入式自动化。另外还包括整套的测试过程管理解决方案等。

## 1.3  黑盒测试理论

### 1.3.1  黑盒测试概念

黑盒测试（Black-box Testing，又称为功能测试或数据驱动测试）是把测试对象看作一个黑盒子。利用黑盒测试法进行动态测试时，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。

采用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。

### 1.3.2  黑盒测试的测试用例设计方法

黑盒测试的测试用例设计方法包括：等价类划分方法、边界值分析方法、错误推测方法、因果图方法、判定表驱动分析方法、正交实验设计方法、功能图分析方法。

#### 1.3.2.1  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方法。

（1）划分等价类：等价类是指某个输入域的子集合。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。

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。

（2）划分等价类的方法：参考六条确定等价类的原则。

（3）设计测试用例：在确立了等价类后，可建立等价类表，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，然后从划分出的等价类中按以下三个原则设计测试用例：

（1）为每一个等价类规定一个唯一的编号。

（2）设计一个新的测试用例，使其尽可能多地覆盖尚未被覆盖地有效等价类，重复这一步。直到所有的有效等价类都被覆盖为止。

（3）设计一个新的测试用例，使其仅覆盖一个尚未被覆盖的无效等价类，重复 这一步。直到所有的无效等价类都被覆盖为止

#### 1.3.2.2  边界值分析法

  边界值分析法边界值分析方法是对等价类划分方法的补充。

（1）边界值分析方法的考虑：长期的测试工作经验告诉我们，大量的错误是发生在输入或输出范围的边界上，而不是发生在输入输出范围的内部。因此针对各种边界情况设计测试用例，可以查出更多的错误。

（2）使用边界值分析方法设计测试用例，首先应确定边界情况。通常输入作为测试用例。如果程序中使用了一个内部数据结构，则应当选择这个内部数据结构的边 界上的值作为测试用例。分析规格说明，找出其它可能的边界条件。

#### 1.3.2.3  错误推测法

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

错误推测方法的基本思想：列举出程序中所有可能有的错误和容易发生错误的特殊情况，根据他们选择测试用例。例如，在单元测试时曾列出的许多在模块中常见的错误。以前产品测试中曾经发现的错误等，这些就是经验的总结。还有，输入数据和输出数据为0的情况。输入表格为空格或输入表格只有一行。这些都是容易发生错误的情况。可选择这些情况下的例子作为测试用例。

因果图方法前面介绍的等价类划分方法和边界值分析方法，都是着重考虑输入条件，但未考虑输入条件之间的联系，相互组合等。考虑输入条件之间的相互组合，可能会产生一些新的情况。但要检查输入条件的组合不是一件容易的事情，即使把所有输入条件划分成等价类，他们之间的组合情况也相当多。因此必须考虑采用一种适合于描述对于多种条件的组合，相应产生多个动作的形式来考虑设计测试用例。这就需要利用因果图（逻辑模型）。

# 第2章  黑盒测试流程

功能性测试也叫黑盒测试，它的简要流程如下：

1. 搭建软件测试平台；
2. 根据需求确认软件功能的完整性；
3. 划分用例项；
4. 设计测试用例，进行测试；
5. 记录测试情况，得出测试结论。

黑盒测试的核心部分就是用例项划分和测试用例的设计，这两部分的设计直接影响了测试的全面性和准确性，下面具体介绍用例项划分和测试用例的设计。

## 2.1  用例项

用例项代表了一个大的测试单元，该单元能够独立实现软件的一个功能。用例项的划分往往与软件功能、界面相关联，在建立划分测试项时需要综合考虑以上两个因素。  按照功能划分用例项是比较科学的方法，也是在测试中经常使用的方法。可以很好保证软件测试的覆盖范围。  按照界面划分是基本的划分方法，可以为每一个界面建立一个用例项，也可以根据功能进行合并和拆解。比如，一个软件具有人员维护的功能，它可能包括两个界面，表示某一个特定人员的界面和所有人员的列表，这里可以将两个界面划分为同一个用例项。

## 2.2  设计测试用例

等价类划分设计方法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量具有代表性的数据作为测试用例。等价类是指某个输入域的子集合。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他值的测试。

# 第3章  测试表格

## 3.1  用例项划分

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 主要功能 | 用例项名称 | 用例项编号 | 功能描述 | 实现情况描述 |
| 用户登录 | 登录功能 | a-1 | 根据用户名密码确认是否登录系统。 |  |
| 管理员操作 | 管理员功能 | a-2 | 管理用户信息、订单信息、管理排片信息；进行系统设置。 |  |
| 个人中心 | 个人中心 | a-3 | 用户点击可进入订单中心，关注电影，购票信息页面。 |  |
| 订单中心 | 订单中心 | a-4 | 用户可以查看自己的订单信息 |  |
| 关注列表 | 关注列表 | a-5 | 用户可以查看自己加入关注的商品信息，选择取消关注或进行购买。 |  |

## 3.2 测试用例表格

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 用例项名称： | |  | | 用例项编号： | |  | |
| 测试员： | |  | | 测试日期： | |  | |
| 功能说明： | |  | | | | | |
| 子功能（按钮） | 目的 | 测试数据 | 预期结果 | 实际结果 | 是否改正 | 更正日期 | 修改人员 |
|  |  |  |  |  |  |  |  |
| 主要功能实现情况： |  | | | | | | |
| 总测试项数 |  | | | OK项数 |  | | |
| NG项数 |  | | | NG—>OK项数 |  | | |

以系统的登录功能为例，首先需要显示登录界面：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 用例项名称： | | 用户登录 | | 用例项编号： | | a-1 | |
| 测试员： | | 韩腊梅 | | 测试日期： | | 2020年6月25日 | |
| 功能说明： | | 根据用户输入的用户名和密码确定用户登录的合法性，决定是否允许用户进入系统 | | | | | |
| 子功能（按钮） | 目的 | 测试数据 | 预期结果 | 实际结果 | 是否改正 | 更正日期 | 修改人员 |
| 登录系统 | 验证登录功能 | 用户名：hlm  密码：lrh | OK |  |  |  |  |
| 验证登录功能 | 用户名：hlm  密码：lrh | OK |  |  |  |  |
| 用户名 | 用户名为空时，程序处理 | 用户名：Null | OK |  |  |  |  |
| 测试 特殊字符处理 | 用户名：)12ab，--&‘“ ” | OK |  |  |  |  |
| 用户名过长时，程序处理 | >50个字符 | OK |  |  |  |  |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 主要功能实现情况： | 程序主要功能实现 | | | | | | |
| 总测试项数 | 9 | | | OK项数 | 3 | | |
| NG项数 | 4 | | | NG—>OK项数 | 2 | | |

3.3  关注点

### 3.3.1  文本输入框

（1）检测空数据；

（2）检测过长数据（超出控件本身允许的长度和数据库中该字段所允许的长度 Varchar(50)）；

（3）检测特殊字符，尤其是数据库中不允许的字符（！@#￥%„„&\*--），甚至回车字符、空格字符等；

（4）检测字符类型，比如：应该输入数字的文本框输入‘abc‘；

（5）中文字符的处理；

（6）对于日期时间型数据，检查格式正确性以及时间日期的合理性。比如：开始时间不能晚于结束时间等。

### 3.3.2  增加数据

（1）数据个数的上限；

（2）重复数据处理，尤其是键值的重复；

（3）相关表格的更新；

（4）检查多次使用back键的情况：在有back的地方，back，回到原来页面，再back，重复多次，看会否出错。

### 3.3.3  修改数据

（1）不能破坏数据库数据的关联和完整；

（2）重复数据处理，尤其是键值的重复；

（3）修改登陆用户本身信息时，对系统的影响；

（4）修改正在使用的数据；

（5）检查多次使用back键的情况：在有back的地方，back，回到原来页面，再back，重复多次，看会否出错。

### 3.3.4  删除数据

（1）不能破坏数据库数据的关联和完整；

（2）删除正在使用的数据；

（3）删除登陆用户本身。

### 3.3.5  查询数据

（1）多条件组合查询的正确性；

（2）多次连续查询正确性。

### 3.3.6  数据导入导出

（1）导入数据格式要求不应太苛刻，提示明确；

（2）导出数据不应乱码。

### 3.3.7  其他

（1）对网络故障的提示；

（2）同一用户多处登陆；

（3）压力测试，系统承受力，多用户同时登陆。