**测试流程及测试理论方法**

1. **测试流程**

1.软件开发流程：

需求分析—>概要设计—>详细设计—>编码开发—>测试—>维护

2.测试流程为：

单元测试/集成测试—>系统测试/自动化测试—>性能测试—>验收测试

 3.目标：

3.1制定完整且具体的测试路线和流程，为快速、高效和高质量的软件测试提供基础流程框架。

 3.2最终目标是实现软件测试规范化、标准化、自动化。

4.测试流程说明：



5.测试需求分析

 测试需求是整个测试过程的基础；确定测试对象以及测试工作的范围和作用。用来确定整个测试工作（如安排时间表、测试设计等）并作为测试覆盖的基础。而且被确定的测试需求项必须是可核实的。即，它们必须有一个可观察、可评测的结果。无法核实的需求不是测试需求。所以我现在的理解是测试需求是一个比较大的概念，它是在整个测试计划文档中体现出来的，不是类似的一个用例或者其他.

·测试需求是制订测试计划的基本依据，确定了测试需求能够为测试计划提供客观依据；

·测试需求是设计测试用例的指导，确定了要测什么、测哪些方面后才能有针对性的设计测试用例；

·测试需求是计算测试覆盖的分母，没有测试需求就无法有效地进行测试覆盖。

5.1测试方法与规范

 5.1.1 **测试方法**

 随着软件技术发展，项目类型越来越多样化。根据项目类型应选用针对性强的测试方法，合适的测试方法可以让我们事半功倍。以下是针对目前项目工程可以参考的测试方法：

* **β测试 （beta测试）--非程序员、测试人员**

β测试，英文是Beta testing。又称Beta测试，用户验收测试（UAT）。

　　β测试是[软件](http://baike.baidu.com/view/37.htm)的多个用户在一个或多个用户的实际使用环境下进行的测试。开发者通常不在测试现场，Beta测试不能由程序员或测试员完成。

当开发和测试根本完成时所做的测试，而最终的错误和问题需要在最终发行前找到。这种测试一般由最终用户或其他人员完成，不能由程序员或测试员完成。

* **α测试（**[**Alpha测试**](http://baike.baidu.com/view/415270.htm)**）****--非程序员、测试人员**

　　α测试，英文是Alpha testing。又称Alpha测试.

　　Alpha测试是由一个用户在开发环境下进行的测试，也可以是公司内部的用户在模拟实际操作环境下进行的受控测试，Alpha测试不能由该系统的程序员或测试员完成。

在系统开发接近完成时对应用系统的测试;测试后，仍然会有少量的设计变更。这种测试一般由最终用户或其他人员来完成，不能由程序员或测试员完成。

* **兼容性测试 --测试人员**

兼容性测试是指测试软件是否可以成功移植到指定的硬件或者软件环境中，例如在B/S项目中各个不同浏览器之间的测试。

* **用户界面测试-UI测试** **--测试人员**

　　用户界面测试，英文是User interface testing。又称UI测试。

　　用户界面，英文是User interface。是指软件中的可见外观及其底层与用户交互的部分（菜单、对话框、窗口和其它控件）。

　　用户界面测试是指测试用户界面的风格是否满足客户要求，文字是否正确，页面是否美观，文字，图 片组合是否完美，操作是否友好等等。UI 测试的目标是确保用户界面会通过测试对象的功能来为用户提供相应的访问或浏览功能。确保用户界面符合公司或行业的标准。包括用户友好性、人性化、易操作性 测试。

　　用户界面测试用户分析软件用户界面的设计是否合乎用户期望或要求。它常常包括菜单，对话框及对 话框上所有按钮，文字，出错提示，帮助信息 (Menu 和Help content)等方面的测试。比如，测试Microsoft Excel中插入符号功能所用的对话框的大小，所有按钮是否对齐，字符串字体大小，出错信息内容和字体大小，工具栏位置/图标等等。

* **冒烟测试 --版本编译者**

　　冒烟测试，英文是Smoke testing。

　　冒烟测试的名称可以理解为该种测试耗时短，仅用一袋烟功夫足够了。也有人认为是形象地类比新电路板功基本功能检查。任何新电路板焊好后，先通电检查，如果存在设计缺陷，电路板可能会短路，板子冒烟了。

冒烟测试的对象是每一个新编译的需要正式测试的软件版本，目的是确认软件基本功能正常，可以进行后续的正式测试工作。冒烟测试的执行者是版本编译人员。

* **随机测试** **--测试人员**

　　随机测试，英文是Ad hoc testing。

　　随机测试没有书面[测试用例](http://baike.baidu.com/view/106882.htm)、记录期望结果、检查列表、[脚本](http://baike.baidu.com/view/54.htm)或指令的测试。主要是根据测试者的经验对软件进行功能和性能抽查。随机测试是根据测试说明书执行用例测试的重要补充手段，是保证测试覆盖完整性的有效方式和过程。

随机测试主要是对被测软件的一些重要功能进行复测，也包括测试那些当前的测试样例 (TestCase)没有覆盖到的部分。另外，对于软件更新和新增加的功能要重点测试。重点对一些特殊点情况点、特殊的使用环境、并发性、进行检查。尤其 对以前测试发现的重大Bug，进行再次测试，可以结合回归测试 (Regressive testing)一起进行。

* **黑盒测试（功能测试）--测试人员**

　　黑盒测试，英文是Black Box Testing。又称功能测试或者数据驱动测试。

黑盒测试是根据软件的规格对软件进行的测试，这类测试不考虑软件内部的运作原理，因此软件对用户来说就像一个黑盒子。

在完全不考虑程序内部结构和特性的情况下，测试软件的外部特性。根据软件的需求规格说明书设计测试用例，从程序输入和输出特性上检查程序是否满足设定的功能。黑盒测试常采用的方法是设计适量有效和无效的输入数据进行测试，以期用最小的代价发现最多的错误。

[软件测试人员](http://baike.baidu.com/view/4005893.htm)以用户的角度，通过各种输入和观察软件的各种输出结果来发现软件存在的缺陷，而不关心程序具体如何实现的一种软件测试方法。

* **个人复查**

　　个人复查是指程序员自行设计**测试用例**，对源代码、详细设计进行仔细检查，并记录错误、不足之处等。个人复查主要包括检查变量的正确性、检查标号的正确性、检查子程序、宏、函数、常量检查、标准检查、风格检查、比较控制流、选择、激活路径、对照详细说明书，阅读源代码和补充文档等方面的测试内容。

* **走查**

　　走查是指测试人员先阅读相应的文档和源代码，然后人工将测试数据输入被测试程序，并在纸上跟踪监视程序的执行情况，人工沿着程序的逻辑走查运行一遍，跟踪走查运行的进程来发现程序的错误。走查的具体测试内容包括模块特性、模块接口、模块的对外输入或输出、局部数据结构、数据计算错误、控制流错误、处理出错和边界测试等方面。

* **会审**

　　会审是指测试人员在会审前仔细阅读软件的有关资料，根据错误类型清单（根据以往的经验、对源程序的估计等，并在以后测试中给以丰富补充）填写检测表，提出根据错误类型要提出的问题。会审时，由程序设计人员讲解程序的设计方法，由程序编写人员逐个讲解程序代码的编写，测试人员需要逐个审查，提问，讨论可能出现的问题。会审对程序的功能、结构、逻辑和风格都要进行审定。会审的测试内容与“走查”的内容相同。

。

* **白盒测试**

　　白盒也称结构测试，这是将软件看成一个透明的白盒子，按照程序的内部结构和处理逻辑来选定测试用例，对软件的逻辑路径及过程进行测试，检查它与设计是否相符。

* **性能测试**

　　性能测试，英文是Performance Testing。

　　性能测试是在交替进行负荷和强迫测试时常用的术语。理想的“性能测试”(和其他类型的测试)应在需求文档或质量保证、测试计划中定义。性能测试一般包括负载测试和压力测试。

　　通常验证软件的性能在正常环境和系统条件下重复使用是否还能满足性能指标。或者执行同样任务时新版本不比旧版本慢。一般还检查系统记忆容量在运行程序时会不会流失(memory leak)。比如，验证程序保存一个巨大的文件新版本不比旧版本慢。

5.1.2 测试规范

 测试规范是根据开发规范而制定的测试标准，测试规范也是后期测试用例编写的重要依据。因为开发规范因公司而异，因产品而异，所以测试规范的标准程度每个公司都不一样。

从理论到方法到各类流程到各类报告模版，都属于测试规范的范畴，当一整套规范形成之后，可使得测试工作进行更加稳健，所有问题有据可查。

5.2 软件需求规格说明书

 软件需求规格说明书是软件达到的各项功能的目标。是测试人员各项工作的依据，没有需求就无法判断测试结果是正确的。

5.3软件设计说明（概要与详细设计）

 设计说明书包含软件的一些框架、字段、数据库设计等。软件设计说明对测试工作开展有很大影响，没有软件设计说明很多问题将无法溯源，测试准备的前期工作也是根据软件设计说明来制定的。

5.4页面原型（demo）

页面原型是项目人员快速熟悉项目的最佳路径。在需求不够明确，设计说明书不够全面的情况下，页面原型也是后期测试用例编写思想的重要根据。

6.测试过程设计

 明确测试目的，最终达成目的并验证结果是测试要做的事情。包括：

1. 测试范围：描述本次测试中的测试范围，如：测试软件功能范围、测试种类等。
2. 简单的描述如何搭建测试平台以及测试的潜在的风险。
3. 项目信息：说明要测试的项目的相关资料，如：输入输出文档，产品描述，软件主要功能。
4. 人力资源的分配。
5. 测试需求：笼统说，就是测试中的所有设计和需求文档。作为本次测试的依据

6.1测试策略制定

* 这一阶段在于需求、详细设计、测试计划完成之后，主要是本次测试的策略阶段。很多公司缺少这个阶段，需要有计划性的分出产品的功能扣出测试的功能点，现阶段大多公司都是直接拿着文档就开始做用例设计。
* 对需求进行分析，列出具体的功能列表。（一般根据功能交互文档就能明确出此功能的大体功能，一层层的分下去，一直到每个功能表单。然后考虑到使用哪些测试方法？工作一旦做到执行阶段，我们可以更好的根据这些功能表一点一点的覆盖。也能让我们在用例评审时，充分的证实我们的工作是有效的能够保证产品的质量。）一般在此之前，一些业务培训和需求评审是有必要是听一下的。这样能够更早更熟练的理解需求，也能保证产品设计中出现的一些误区。
* 对于一个个测试该如何进行测试？如下：
1. 功能测试
* 功能范围（划分出各自负责的功能模块）
* 使用测试方法（等价类、边界值等测试方法）
* 测试标准（符合设计、需求和规范文档对该功能的描述）
1. 界面测试
2. 兼容性测试
3. 性能测试

6.2测试计划

1. 要充分考虑测试计划的实用性，即测试计划与实际之间的接近程度和可操作性。编写测试计划的目的在于充分考虑执行测试时 的各种资源，包括测试内容、测试标准、时间资源、人力资源等等，准确地说是要分析执行时所能够调用的一切资源以及受各种条件限制，可能受到的各种影响。
2. 测试内容：对一个软件来说测试计划中会明确本次测试做哪些测试？

如：系统测试：在整个系统测试中会有（界面测试、功能测试、性能测试、兼

容性测试、安装卸载测试、可靠性测试等测试）。

1. 测试目的：一般多为保证产品质量是否达到预期的指标。这个指标也就是在测试中定义的结束标准。
2. 测试标准：需要考虑本次测试需要输入那些文档，该项目结束标准定义、测试结束标准的定义？bug级别定义、优先级定义、bug管理流程定义。这个都需要在执行测试时明确。计划中应该包含这些内容。
3. 资源分配：这里分为人力资源、软硬件资源等划分。一般会把人力资源的利用写入一个测试人员任务分配表里，按照不同的阶段，每个阶段提交相应的成果（难度很大）。软硬件资源中主要是在做计划时考虑到需要多少电脑或别的工具，列出清单。
4. 测试风险：大多考虑到的就是项目开发延期、测试人员不足用例无法全面覆盖测试点、时间不足用例无法全部执行、bug无法及时修改导致无法验证、测试人员技能不足导致测试进度拉长。
5. 软件测试策略一般都是分开来做相关测试方案。

6.3测试附件

* + 用例模板、缺陷报告模板
	+ 测试环境的搭建
	+ 缺陷管理流程和缺陷级别定义

缺陷状态一般分为：新建、打开、已分配、已修复、关闭、重新打开

中间会有：延期、重复、拒绝等状态

**缺陷管理流程：**



1. 测试人员或开发人员发现bug后，判断输入哪个模块的问题，填写bug报告后，系统会自动通过Email通知开发组长和该模块开发者。
2. 开发组长根据具体情况，重新reassigned分配给bug所属的开发者。
3. 开发者收到email信息后，判断是否为自己的修改范围。
* 若不是，重新reassigned分配给开发组长或应该分配的开发者。
* 若是，进行处理，resolved并给出解决方法。（可创建补丁附件及补充说明）
1. 测试人员查询开发者已修改的bug，进行回归测试。
* 经验证无误后，修改状态为verified。待整个产品发布后，修改为closed。
* 还有问题，reopened，状态重新变为“new”，并发送邮件通知。
1. 如果这个bug一周内一直没被处理过。Bugzilla就会一直用email骚扰它的属主，直接采取行动。管理员可以设定最迟采取行动的期限，比如3天，系统默认7天。

**缺陷等级划分：**

|  |  |  |  |
| --- | --- | --- | --- |
| **分级** | **Bug等级** | **Bug等级说明** | **分类说明** |
| 致命问题 | Blocker | 导致整个产品无法进行测试。修改优先级为最高，该级别需要程序员立即修改 | ○ 模块无法启动或异常退出 |
| ○ 其它导致无法测试的错误 |
| Critical | 死机，数据丢失，主要功能完全丧失，系统悬挂等错误。修改优先级为最高，该级别需要程序员立即修改 | ○ 运行过程中系统崩溃/死机/重启 |
| ○ 功能设计与需求严重不符 |
| ○ 严重花屏 |
| ○ 内存泄漏 |
| ○ 影响手机语音或数据通讯等 |
| ○ 严重的数值计算错误 |
| 严重问题 | Major | 主要功能丧失，导致严重的问题，或致命的错误声明。修改优先级为高，该级别需要程序员尽快修改 | ○ 功能未实现或者存在错误 |
| ○ 轻微的数值计算错误 |
| ○ 系统所提供的功能或服务受明显的影响 |
| ○ 用户数据丢失或破坏 |
| 一般问题 | Normal | 次要功能丧失， 不太严重，如提示信息不太准确。修改优先级为中，该级别需要程序员修改 | ○ 操作界面错误（包括数据窗口内列名定义、含义是否一致） |
| ○ 边界条件下错误 |
| ○ 功能存在错误，但出现概率很低 |
| ○ 提示信息错误（包括未给出信息、信息提示错误等） |
| ○ 长时间操作无进度提示 |
| ○ 系统未优化（性能问题） |
| Minor | 微小的问题，对功能几乎没有影响，产品及属性仍可使用。修改优先级为低，该级别需要程序员修改或不修改 | ○ 界面格式等不规范 |
| ○ 操作时未给用户提示 |
| ○ 文字排列不整齐等一些小问题 |
| ○ 光标跳转设置不好，鼠标（光标）定位错误 |
| 轻微问题 | Trivial | 提示信息格式不符合要求, 违背正常习俗习惯的，界面不美观，控件排列、格式不统一 | ○ 辅助说明描述不清楚 |
| ○ 个别不影响产品理解的错别字 |
| ○ 可输入区域和只读区域没有明显的区分标志 |
| Enhancement | 功能性建议，功能使用性、方便性、易用性不够 | ○ 建议 |

**7.测试实施**

**7.1执行**

* 开发就会转版本给我们测试部门进行系统测试了。拿到版本我们首先搭建测试环境
* 做一个预测试，目的是来评断这个版本是不是可测试的。如果预测试不通过，打回开发部返工，如果通过了，就开始我们第一轮的系统测试。
* 第一轮系统测试我们会执行我们所编写的所有测试用例，做好测试结果的记录，发现缺陷了提交缺陷报告。当第一轮测试结束后，我们把所有的bug单提交给开发人员，由他们进行修改。
* 在他们修复bug期间，我们会对第一轮系统测试做一个测试评估，出一个测试报告。还要根据实际情况，对我们写的测试用例进行修改和增加。开发改bug结束，提交一个新的版本给我们，我们重新搭建测试环境开始第二轮系统测试。首先是回归我们提交的缺陷报告，然后会在用例中挑选一些优先级别比较高的用例来进行测试，发现问 题了继续提交缺陷报告，直到缺陷率低于用户要求了，我们就进行最后一轮的回归测试，结束系统测试。具体测试轮次是根据版本质量和项目复杂度而决定的。

**8.测试评估**

* 执行阶段结束了进入测试评估阶段，我们会出一个总的测试报告对我们测试的这个过程和版本的质量做一个详细的评估
1. 需求需要评审那些？
2. 用例需要评审那些？
3. 计划应该评审那些？
4. 缺陷评审那些？
5. bug评估？

测试总结报告文档的输出：

1、可以让具体的任务负责人对本次测试中个人负责的模快进行评价，提出相关建议。给出总体的评估

2、整体上的bug按照不同等级统计出来、用例数量、用例执行数量

3、对项目中测试人力资源的统计。（单位：人/天）

1. 项目中软硬件资源统计。

5、提出软件总体的评价。

**8.1测试报告**

测试报告包括对软件功能的结论，说明为满足此项功能而设计的软件能力以及经过一项或多项测试已证实的能力。

说明该项目软件的开发是否达到预定目标，是否可以交付使用。

总结测试工作的资源消耗数据，如工作人员的水平级别数量、机时消耗等。

记录测试结果与发现本项目测试工作所得到的各项输出的承载体，根据输入与计划、要求的对比来总结此次项目所或得的经验。

**9.测试维护**