

# 软件测试教程 第三节 基础篇

从这节课开始，我们将作为一名刚刚加入测试团队的菜鸟，开始一次测试之旅。

在这里我们将回答以下问题：

- 软件测试的生命周期
- 如何描述一个bug
- 如何定义bug的级别
- bug的生命周期
- 如何开始第一次测试
- 测试的执行和bug管理
- 产生争执怎么办

## 软件测试的生命周期

软件测试的生命周期：需求分析→测试计划→测试设计、测试开发→测试执行→测试评估

## 软件测试&软件开发生命周期

- 需求阶段
  - 测试人员了解需求、对需求进行分解，得出测试需求
- 计划阶段
  - 根据需求编写测试计划/测试方案
- 设计阶段
  - 测试人员适当的了解设计，对于设计测试用例是很有帮助的，测试人员搭建测试用例框架，根据需求和设计编写一部分测试用例
- 编码阶段
  - 测试人员一般是不需要编码的，但已经编码的模块，专业的白盒测试人员可以计划执行单元测试，完善、细化测试用例以及调整测试计划和方案。
- 测试阶段
  - 测试阶段是软件测试人员最为重要的工作阶段，根据测试用例和计划执行测试，在执行的过程中记录、管理缺陷，测试完成后编写测试报告。
- 运行维护
  - 测试人员需要参与项目的实施工作。测试人员对项目产品的业务和操作非常了解，加上测试人员的沟通表达能力一般都比较强，所以测试人员可以参与用户使用软件的培训，在试运行项目时收集问题并及时反馈给相关负责人。

## 如何描述一个bug

一个合格的bug描述应该包括以下几个部分：

### 1、发现问题的版本

开发人员需要知道出现问题的版本，才能够获取对应版本的代码来重现故障。并且版本的标识也有利于统计和分析每个版本的质量。

## 2、问题出现的环境

环境分为硬件环境和软件环境，如果是web项目，需要描述浏览器版本，客户机操作系统等，如果是app项目，需要描述机型、分辨率、操作系统版本等。详细的环境描述有利于故障的定位。

## 3、错误重现的步骤

描述问题重现的**最短**步骤。

## 4、预期行为的描述

要让开发人员指导怎么样才是正确的，尤其要以用户的角度来描述程序的行为是怎样的。如果是依据需求提出的故障，能写明需求的来源是最好的。

*要相信：测试人员是最懂需求的。*

## 5、错误行为的描述

描述错误的现象。crash等可以上传log，UI问题可以有截图。

## 6、其他

某些公司会有一些其他的要求，例如故障的分类：功能故障，界面故障，兼容性故障等。有些有优先级的分类，严重影响测试需要开发人员优先修改的，可以设置优先级为高。

## 7、不要把多个bug放到一起

在无法确认是同一段代码造成的故障时，不要将bug放在一起提交。

案例：

提交了如下bug：

- 1、在短信列表，选择一条短信，进行删除，删除失败
- 2、在短信列表，选择一条短信，进行查看，在查看页面，进行删除，删除失败

故障发现版本：VPS20180226\_01

故障类别：兼容性

故障优先级：中

故障标题：ie下界面显示异常，界面文字有重叠

故障描述：

测试环境：win7+IE8

测试步骤：1、打开vps首页，点击“通知”链接，进入通知页面

预期结果：通知页面显示正确，一页显示10条通知，按时间顺序倒序排列

实际结果：页面显示10条通知，通知顺序正确，但是页面文字有重叠

附件：上传截图

## 如何定义bug的级别

bug的定义每个公司都不一致，在定义级别之前需要查看公司规范。

以下为样例：

### 1、Blocker（崩溃）：

阻碍开发或测试工作的问题；造成系统崩溃、死机、死循环，导致数据库数据丢失，与数据库连接错误，主要功能丧失，基本模块缺失等问题。如：代码错误、死循环、数据库发生死锁、重要的一级菜单功能不能使用等（该问题在测试中较少出现，一旦出现应立即中止当前版本测试）。

### 2、Critical（严重）：

系统主要功能部分丧失、数据库保存调用错误、用户数据丢失，一级功能菜单不能使用但是不影响其他功能的测试。功能设计与需求严重不符，模块无法启动或调用，程序重启、自动退出，关联程序间调用冲突，安全问题、稳定性等。如：软件中数据保存后数据库中显示错误，用户所要求的功能缺失，程序接口错误，数值计算统计错误等（该等级问题出现在不影响其他功能测试的情况下可以继续该版本测试）。

### 3、Major（一般）：

功能没有完全实现但是不影响使用，功能菜单存在缺陷但不会影响系统稳定性。如：操作时间长、查询时间长、格式错误、边界条件错误，删除没有确认框、数据库表中字段过多等(该问题实际测试中存在最多)

### 4、Minor（次要）：

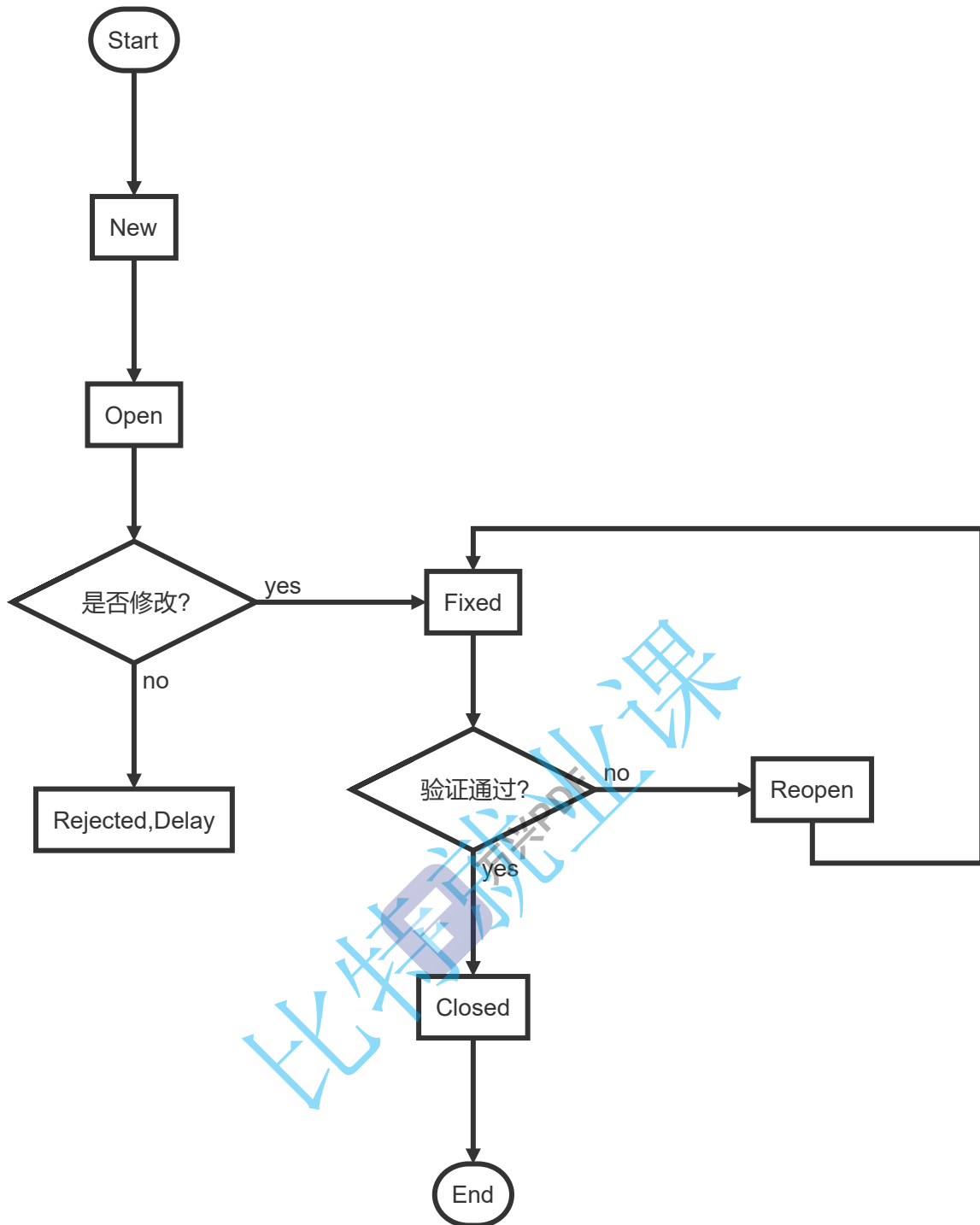
界面、性能缺陷，建议类问题，不影响操作功能的执行，可以优化性能的方案等。如：错别字、界面格式不规范，页面显示重叠、不该显示的要隐藏，描述不清楚，提示语丢失，文字排列不整齐，光标位置不正确，用户体验感受不好，可以优化性能的方案等（此类问题在测试初期较多，优先程度较低；在测试后期出现较少，应及时处理）

## bug的生命周期

每个公司、每一个工具对bug生命周期的定义都是不一致的，下面仅是一个常见的例子

测试人员应该跟踪一个Bug的整个生命周期，从Open到Closed的所有状态。

### BUG状态转换图



- **New:** 新发现的Bug，未经评审决定是否指派给开发人员进行修改。
  - **Open:** 确认是Bug，并且认为需要进行修改，指派给相应的开发人员。
  - **Fixed:** 开发人员进行修改后标识成修改状态，有待测试人员的回归测试验证。
  - **Rejected:** 如果认为不是Bug，则拒绝修改。
  - **Delay:** 如果认为暂时不需要修改或暂时不能修改，则延后修改。
  - **Closed:** 修改状态的Bug经测试人员的回归测试验证通过，则关闭Bug。
  - **Reopen:** 如果经验证Bug仍然存在，则需要重新打开Bug，开发人员重新修改。
- 无效的bug: open->closed    open-rejected-closed

缺陷状态变更流程每个项目团队的实际做法可能不大一样。并且需要结合实际的开发流程和协作流程来使用。

例如，测试人员新发现的Bug，必须由测试组长评审后才决定是否Open并分派给开发人员。测试人员Open的Bug可以直接分派给Bug对应的程序模块的负责人，也可以要求都先统一提交给开发主管，由开发主管审核后再决定是否分派给开发人员进行修改。Bug的跟踪以及状态变更应该遵循一些基本原则：

- 测试人员对每一个缺陷的修改必须重新取一个包含更改后的代码的新版本进行回归测试，确保相同的问题不再出现，才能关闭缺陷。
- 对于拒绝修改和延迟修改的Bug，需要经过包含测试人员代表和开发人员代表、用户方面的代表（或代表用户角度的人）的评审。

## 如何开始第一次测试

作为一个菜鸟在进入测试团队开始第一次测试的时候，我们需要做很多的准备：

- 1、阅读所有项目有关的文档，包括：需求文档、设计文档、用户手册
- 2、尽可能参加各种项目会议，了解项目的背景、人员组成、尽可能的了解需求和业务。特别针对业务专业性较强的项目，例如银行业务，需要了解各种业务知识，如高低柜、一二三类账户等、存款、贷款等。
- 3、熟悉项目所使用的测试管理工具、配置管理工具，获取对应的地址和登录方式
- 4、阅读已有的测试方案和测试案例
- 5、阅读旧有的bug库，了解系统功能。尤其重要的是和现有的测试团队保持一致的故障定级原则
- 6、了解公司的规范要求，特别是用例编写规范、用例执行规范、bug提交规范、测试工具工具使用规范等

在进行了以上的准备工作之后，第一次测试工作到来了，我们需要与测试组长确认具体的工作内容：

- 1、测试的计划是什么？
- 2、测试的内容是什么？test case有多少？安排了几天执行？有没有自由测试的时间？
- 3、我要测试的内容开发人员是谁？需求人员是谁？
- 4、分配给我的测试内容是否需要特殊的测试资源？资源是否满足需要？

在我们确认了以上内容之后，就可以开始测试的执行了

## 测试的执行和BUG管理

现在我们开始进行测试了：

1. 打开待测试的系统
2. 打开测试管理工具用例模块，开始执行用例
3. 发现bug！进行复现并确认bug的复现步骤
4. 记录bug
5. 沟通bug
6. 验证以前提交的bug
7. 确认本次测试完成
8. 编写测试报告

执行测试时处理要做到测试用例和需求的覆盖外，还要有临时发挥的能力。根据自己的经验、对测试的感悟以及随机测试可以发现很多根据测试用例无法发现的缺陷。

不能拘泥于测试用例或者已有的测试方法，在测试执行过程中要不断总结测试方法和测试故障模型。真正优秀的测试人员在执行测试时是想着做，做着想，这样的测试效果才好，尤其是在测试过程中，对程序的处理相当了解的情况下，测试的思路会更加清晰和全面。

### 如何发现更多的bug？

- 1、软件测试同样存在二八原则，80%的故障集中于20%的模块，如果某部分问题较多，加强测试广度和深度！
- 2、开发人员也存在二八原则，80%的故障集中于20%的开发人员，如果某些开发人员的bug较多，加强他开发模块的测试广度和深度！
- 3、多进行逆向思维和发散性的思维
- 4、不要局限于用例和需求文档
- 5、尽早介入项目,不要等到开发的差不多了再介入项目

发散一下：以概念篇中注册的需求来进行一次测试？

功能：所有需求描述的功能

功能其他：需求未考虑到的：邮件内容是否正确？连续注册？

边界：最大值、最小值等

界面：美观，整齐

校验：email格式校验，错误校验，已注册校验、输出校验，为空等

兼容性：IE, CHROME, 360.....

安全性：验证码能否起效？http请求直接发送？

性能：多用户并发

其他？

48小时真的是48小时么？

## 产生争执怎么办（处理人际关系）

### 能让开发人员解决最多Bug的测试人员是最优秀的测试人员

在测试工作中，最常遇到的是和开发人员的PK，作为测试经理还会和项目经理、产品经理的PK进度、质量。

作为一名测试人员，一般会遇到以下几种情况：

- 这不是bug
- 这个bug的级别太高了
- bug影响不大，暂不修改

### 遇到争执不要怕，记住批判性思维：清楚--准确、切题--深刻，有意义，有逻辑性--公正、全面

#### 1、先检查自身，是否bug描述不清楚

如果能正确地、高质量地录入一个Bug，那么基本上已经成功地与开发人员沟通了一大半的关于Bug的信息。但是总有“书难达意”的耐候，这时就需要测试人员主动与开发人员进行沟通了。如果测试人员发现在写完一个缺陷后，好像还有很多关于Bug的信息没有表达出来，或者很难用书面语言表达出来时，就应该在提交Bug后，马上找相关的程序员解释刚才录入的Bug，确保程序员明白Bug描述的意思，而不要等待开发人员找自己了解更多的信息。

2、站在用户角度考虑问题 应该让开发人员了解到Bug对用户可能造成的困扰，这样才能促使开发人员更加积极地、高质量地修改Bug。在争执时，可以问一句：如果你是用户，你可以接受么？

案例：

例如：需求要求可以上传图片作为头像，但是没有定义格式。开发人员在上传时限制为只能传png格式的。站在用户角度考虑一下：png，jpg那种格式更多？是否要用户自己进行格式转换再上传？

发散一下：

如下图，当选择了1.png，2.png，2-2.png时，任意点击一个删除按钮，所选的三个都被删除

## 编辑版本信息

主版本名称:	Testedition	主版本号:	v0.1	构建版本号:	0006
<div>+ 添加 - 删除</div>					
操作	工程名	版本类型	自动打包/创建基线	附件	
<input type="checkbox"/>	Test	<input type="button" value="全量"/> <input type="button" value="增量"/>	<input type="button" value="点击获取打包参数"/>	<input type="button" value="上传"/> <input type="button" value="删除"/>	<input type="checkbox"/> 1.PNG
送测清单:	<input type="button" value="上传"/> <input type="button" value="删除"/>	<input type="checkbox"/> 2.PNG			
SIT测试报告:	<input type="button" value="上传"/> <input type="button" value="删除"/>	<input type="checkbox"/> 2-2.PNG			
版本内容描述:	<div></div>				
<div>确定 关闭</div>					

### 3、BUG定级要有理有据

BUG定级时，不仅要参考BUG级别，还要考虑BUG是否会影响到流程，往往用户的BUG级别和我们的是有区别的，需站在用户的角度定考虑定位级别。

### 4、提高自身的技术和业务水平. 不光要提出问题, 最好也能提出解决方案

提高自身的技术和业务水平，不但要做到能提出问题，还能够提出解决问题的思路。这样才能更让人信服。

在工作中，你会发现同一个bug，资深测试工程师提出和初级测试工程师提出，两者的结果完全不同，两者最大的差别是资深测试工程师往往会提出解决方案。而长此以往，权威性逐渐的建立起来，那么开发人员看到bug的第一反应，就是这是一个bug，而不是这是一个bug吗？

案例：

某网站经常隔几天访问时会出现500错误，但是之后就不会复现。

测试人员会提出问题：网站偶发性出现500错误。

开发人员回答：不常见，不影响使用，暂不修改。

资深测试人员提出问题：网站偶发性500错误，查看日志，是由于mysql数据库8小时超时问题造成。需要修改连接池配置定期校验连接。

开发人员处理：修改xml，增加校验配置项。

### 5、开发人员不接受时，不要争吵

可能你已经经过了多轮沟通，但是开发人员仍然拒不接受。此时可以发起Bug评审。

Bug评审要注意的问题 缺陷的评审应该包括以下两个层面 ● 决定如何处理Bug。 ● 分析缺陷产生的原因，找出预防的对策。

(1)决定如何处理Bug。 这一方面评审需要项目组各个方面的代表参加，通常不可缺少的是测试代表、开发代表、产品代表。

测试代表主要从Bug的具体表现、严重程度等方面提供信息，并提出自己对Bug的处理意见。需要注意的是，测试人员不应该一味地要求对Bug进行修改，因为修改可能带来回归的风险，同时带来的是回归测试的工作量，如果时间比较紧迫，修改后剩余的时间若不足以做一次有效的回归测试，可能不修改是个明智的选择。

开发代表主要从修改缺陷的难度和风险出发，考虑缺陷修改需要付出的代价，以及可能影响的范围、可能引发的风险等，如果决定要修改，还要讨论出修改的初步方案。

产品代表主要从产品的整体计划、用户的要求等方面对缺陷的修改必要性、缺陷修改的时间和版本提出自己的意见。这在微软的做法叫“Bug三方讨论会”，参加者一般是测试人员、开发人员和项目经理。





(2)分析缺陷产生的原因，找出预防的对策。缺陷评审还应该包括原因分析，找出Bug出现的原因，尤其是那些重复出现的Bug。应该找出出现错误的根源，并且制定出相应的预防措施，确保同类型的Bug不再出现。例如：有些Bug出现的原因不是简单的“引用为空”之类，而是开发人员的编码不规范或者编程习惯不好而导致，所以必须建立起正确的编程方式才能预防这些错误的出现，否则只是在玩无聊地重复发现相同的Bug的游戏。

万兴PDF  
比邻作业课