**测试缺陷管理规范**

1. **目的**
   1. 缺陷是产品与规定要求不相符的部分，会存在于软件产品的整个生命周期中，本文规范软件测试过程中的出现的缺陷，通过测试活动及早发现软件系统中的缺陷，并确保缺陷被有效标识、跟踪、和修改，保证软件系统能够达到要求的质量。
2. **适用范围**
   1. 适用于软件的整个生命周期。
   2. 不限于测试过程发现的缺陷。评审，用户使用等过程中发现的缺陷都是应当按照本流程进行登记跟踪管理。
3. **术语和定义**
   1. 软件缺陷：软件或程序中存在的某种破坏正常运行能力的问题、错误，或者隐藏的功能缺陷。
   2. 严重程度：缺陷严重程度是指因缺陷引起的故障对[软件](http://baike.baidu.com/view/37.htm)产品的影响程度。
   3. 优先级：缺陷必须被修复的紧急程度。
4. **职责**
   1. **技术部**
      1. 测试工程师：主要是指发现缺陷和报告缺陷的测试人员。在一般流程中，需要对这个缺陷后续相关的状态负责：包括相关人员对这个缺陷相关信息的询问回答，以及验证测试。
      2. 开发工程师：主要是指对这个缺陷进行研究和修改的开发人员。同时，需要对修改后的缺陷在提交测试人员正式测试验证之前需要进行验证测试。
      3. 其他参与人员：项目负责人、用户组等，他们对缺陷进行优先级划分，负责人进行确认并调节争议。
      4. 配置管理员：负责缺陷库的创建和权限管理，并监督指导缺陷库的定制。
5. **缺陷流程**
   1. **登记**
      1. 缺陷发现后，由测试人员或者其他发现缺陷的人员登记到缺陷库。
      2. 缺陷登记后，提交前可以反复编辑，补充缺陷记录的信息。
      3. 登记缺陷描述的要求为分类准确、叙述简洁、步骤清楚、有实例、可再现、复杂问题有据可查（截图或上传附件的形式）

具体要求为：

单一：尽量一个报告只针对一个软件缺陷。

简洁： 每个步骤的描述应简洁明了。

再现：描述重现的步骤和条件，比如具体输入参数值，以便进行回归验证。应提供截图。

期望结果。

实际结果。

其它信息，可依实际情况增加。

* 1. **提交**
     1. 测试人员确认缺陷已经表述清楚，可以提交缺陷。
     2. 提交后的缺陷状态时“已提交”。
     3. 缺陷提交前必须分配一个具体的开发人员负责，如果测试人员不确定谁负责，可以把缺陷分配给开发负责人，由开发负责人重新分配责任人。
  2. **处置**
     1. 开发人员确认缺陷是自己负责后，开始着手处理，并修改缺陷的状态为“打开”，表示缺陷正在处理中。
     2. 开发人员对缺陷处置完成后，需做处置记录：

原因：说明缺陷产生的原因，比如：设计考虑不周，边界处理不严密，逻辑判断不合理。要求描述具体简洁，以便总结经验。

解决方法：修改稿涉及的文件、源代码、配置、脚本等。

概括：缺陷是否可能存在于其他位置，或引起其他问题。

* + 1. 已打开的缺陷也可以修改负责人。
  1. **解决**
     1. 问题解决后，填写解决处理记录，写明造成缺陷的原因和解决方案，改变缺陷状态为“已解决”。
     2. 如果开发人员发现如下情况，可以把缺陷驳回给测试人员：

缺陷不可再现

与先前登记的缺陷重复

不是缺陷，是测试人员理解错误

缺陷轻微，且修改困难、或修改易导致更大的潜在问题

如果按照开发计划，缺陷发生的功能不属于当前开发阶段必须完成的（需与项目负责人确认）。

* 1. **验证**
     1. 测试人员对“已解决”状态的缺陷进行重新测试，测试步骤应当按照等级的可重现步骤进行。
  2. **关闭**
     1. 测试人员确认缺陷已经解决后，关闭缺陷。
     2. 对于被开发人员驳回的缺陷，测试人员需和项目负责人讨论，项目负责人同意的可以关闭，否则需驳回给开发人员；
  3. **驳回开发人员重新修改**
     1. 验证测试不通过的缺陷，应当驳回给开发人员，状态为“重新打开”。
     2. 关闭了的缺陷再次出现时（通常因为解决缺陷的方法导致相同位置出现不同形式的缺陷时），测试人员重新打开缺陷，开发人员需要继续解决。

1. **附件**
   1. **缺陷严重程度**

|  |  |  |
| --- | --- | --- |
| **严重程度** | **标示** | **含义** |
| 1 | 致命 | 导致软件无法使用问题，例如整个程序崩溃，导致无法使用，测试阻塞。 1.问题会自发的影响整个系统。 2.用户使用正常的操作步骤，就会影响整个系统提供的服务。 3.具有操作先后顺序的功能，已开始的步骤出现故障，导致后续步骤无法使用。 |
| 2 | 严重 | 某个功能未实现或导致一个特性或导致一个特性不能运行并且没有替代方案 |
| 3 | 一般 | 错误导致了一个特性不能运行但可有一个替代方案。 功能特征设计不符合系统的需求，不影响系统的业务，并且有相应的补救方法。 |
| 4 | 轻微 | 错误是表面化或微小的（提示信息不太准确友好、不准确、误导、错别字、界面布局或罕见故障等），对功能几乎没有影响，产品及属性仍可使用。 |
| 5 | 建议 | 建设性的意见或建议。 需求文档没有规定的特性，如果实现会对系统功能或易用性有所提高。 |

* 1. **缺陷优先级**

|  |  |
| --- | --- |
| **优先级** | **含义** |
| 1 紧急 | 如果故障妨碍开发人员的进一步开发活动，应立即修复。 如果阻塞测试，应立即修复。 |
| 2 必须的 | 必须修改，版本发布前必须修正 |
| 3 应该的 | 必须修改，不一定马上修改，但需确定在某个特定版本发布前必须修正 |
| 4 可选的 | 如果时间允许应该修改 |
| 5 不需要 | 允许不修改 |

* 1. **缺陷状态**

|  |  |
| --- | --- |
| 缺陷状态 | 描述 |
| 初始状态 | 测试或开发人员提交一个新的缺陷，等待开发人员或项目经理分配修改负责人 |
| 驳回 | 要求缺陷的提交者再次对缺陷进行说明 |
| 已分配 | 已分配给开发人员，待修改状态 |
| 已解决 | 缺陷已被开发人员修复，等待测试人员验证 |
| 关闭 | 测试人员验证已修复的缺陷 |
| 重新打开 | 测试人员验证，缺陷没有修改正确 |
| 遗留 | 经项目负责人验证此缺陷在本版本中不用修改 |