软件测试与维护

报告所发现的软件缺陷

软件缺陷的描述

软件缺陷是什么？

软件缺陷指的是系统或系统部件中那些导致系统或部件不能实现其功能或者性能的缺陷。如果在执行中遇到一个缺陷，可能引起系统的失效。

那么准确有效的定义和描述软件缺陷，可以使软件缺陷得以快速修复，节约了软件测试项目的成本和资源，提高产品质量。

软件缺陷的基本描述

- 软件缺陷的描述是软件缺陷报告中测试人员对问题的陈述的一部分并且是软件缺陷报告的基础部分。同时，软件缺陷的描述也是测试人员就一个软件问题与开发小组交流的最初且最好的机会。
- 一个好的描述，需要使用简单的、准确的、专业的语言来抓住缺陷的本质。

  以下是软件缺陷的有效描述规则：
  - 单一准确
  - 可以再现
  - 完整统一
  - 短小简练
  - 特定条件
  - 补充完善
  - 不做评价

软件缺陷属性

- 软件缺陷属性包括
  - 缺陷标识
  - 缺陷类型
  - 缺陷严重程度
  - 缺陷优先级
  - 缺陷产生可能性
  - 缺陷状态
  - 缺陷起源
  - 缺陷来源
  - 缺陷原因。

软件缺陷标识和类型

- 缺陷标识：是标记某个缺陷的唯一的表示，可以使用数字序号表示。
- 缺陷类型：是根据缺陷的自然属性划分缺陷种类。见软件缺陷类型列表：

缺陷类型	描述
功能	影响了各种系统功能、逻辑的缺陷
用户界面	影响了用户界面、人机交互特性，包括屏幕格式、用户输入灵活性、结果输出格式等方面的缺陷
文档	影响发布和维护，包括注释，用户手册，设计文档
软件包	由于软件配置库、变更管理或版本控制引起的错误
性能	不满足系统可测量的属性值，如执行时间，事务处理速率等。
系统/模块接口	与其他组件、模块或设备驱动程序、调用参数、控制块或参数列表等不匹配、冲突。

软件缺陷产生的可能性(频率)

- 缺陷产生的可能性：指缺陷在产品中发生的可能性，通常可以用频率来表示。

缺陷产生可能性	描述
总是 (Always)	总是产生这个软件缺陷，其产生的频率是100%
通常 (Often)	按照测试用例，通常情况下会产生这个软件缺陷，其产生的频率大概是80-90%
有时 (Occasionally)	按照测试用例，有的时候产生这个软件缺陷，其产生的频率大概是30-50%
很少 (rarely)	按照测试用例，很少产生这个软件缺陷，其产生的频率大概是1-5%

软件缺陷状态

- 缺陷状态：指缺陷通过一个跟踪修复过程的进展情况，也就是在软件生命周期中的状态基本定义，如软件缺陷状态列表所示:

缺陷状态	描述
激活或打开（Active or Open）	问题还没有解决，存在源代码中，确认“提交的缺陷”，等待处理，如新报的缺陷。
已修正或修复(Fixed or Resolved)	已被开发人员检查、修复过的缺陷，通过单元测试，认为已解决但还没有被测试人员验证
关闭或非激活(Closed or Inactive)	测试人员验证后，确认缺陷不存在之后的状态。
重新打开(Reopen)	测试人员验证后，还依然存在的缺陷，等待开发人员进一步修复
推迟(Deferred)	这个软件缺陷可以在下一个版本中解决
保留(on hold)	由于技术原因或第三者软件的缺陷，开发人员不能修复的缺陷
不能重现(Cannotduplicate)	开发不能复现这个软件缺陷，需要测试人员检查缺陷复现的步骤。
需要更多信息(Needmoreinfor)	开发能复现这个软件缺陷，但开发人员需要一些信息，例如：缺陷的日志文件，图片等。
重复（Duplicate）	这个软件缺陷已经被其他的软件测试人员发现。
不是缺陷（Notabug）	这个问题不是软件缺陷
需要修改软件规格说明书（Spec modified）	由于软件规格说明书对软件设计的要求，软件开发人员无法修复这个软件缺陷，必须要修改软件规格说明书。

软件缺陷起源

- 缺陷起源：缺陷引起的故障或事件第一次被检测到的阶段，如软件缺陷起源列表所示。

缺陷起源	描述
需求	在需求阶段发现的缺陷
构架	在系统构架设计阶段发现的缺陷
设计	在程序设计阶段发现的缺陷
编码	在编码阶段发现的缺陷
测试	在测试阶段发现的缺陷
用户	在用户使用阶段发现的缺陷

软件缺陷来源

- 缺陷来源：指缺陷所在的地方，如文档、代码等，如软件缺陷来源列表所示。

缺陷来源	描述
需求说明书	需求说明书的错误、或不清楚引起的问题
设计文档	设计文档描述不准确、和需求说明书不一致的问题
系统集成接口	系统各模块参数不匹配、开发组之间缺乏协调引起的缺陷
数据流(库)	由于数据字典、数据库中的错误引起的缺陷
程序代码	纯粹在编码中的问题所引起的缺陷

软件缺陷根源(背后因素)

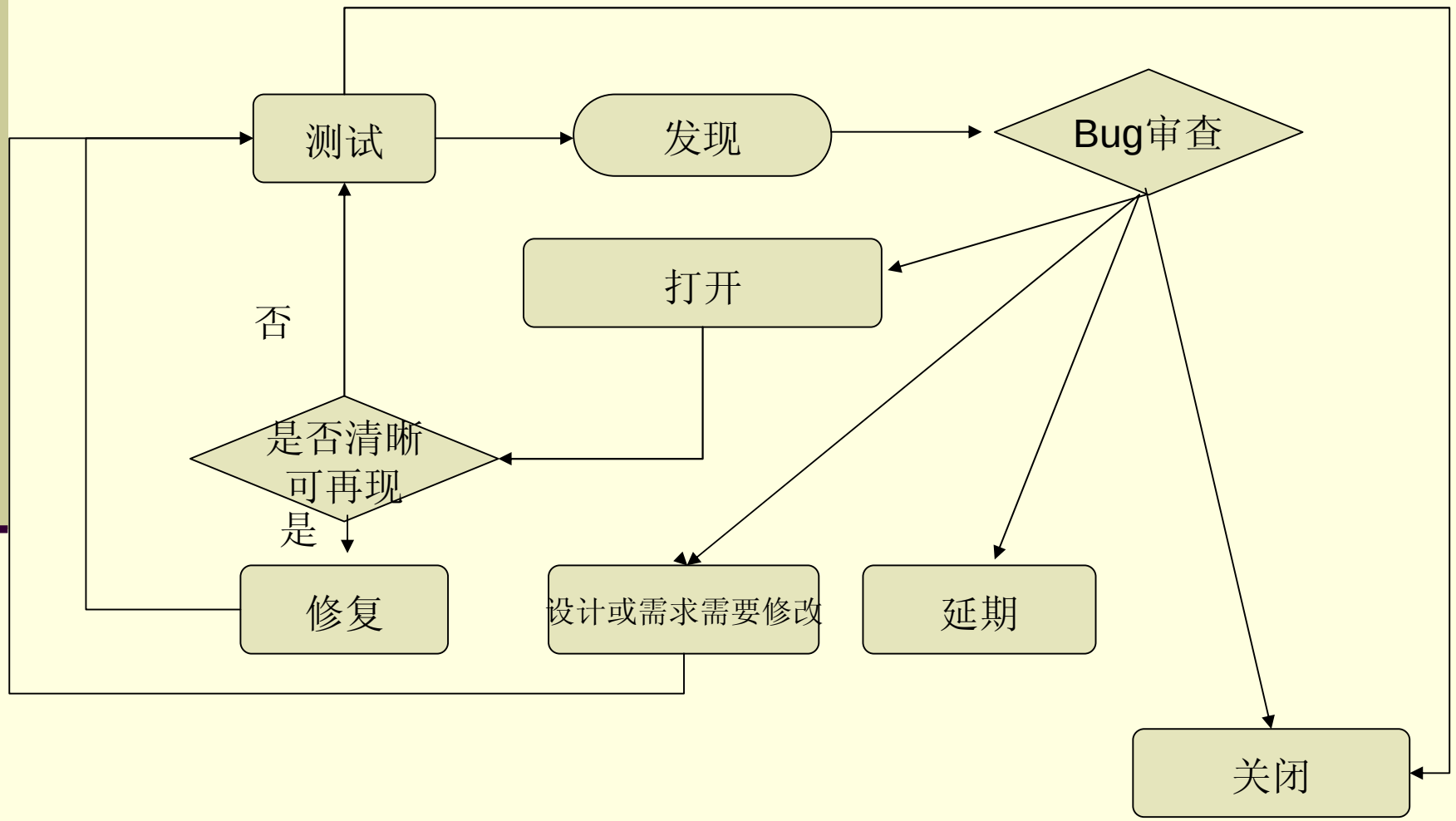
- 缺陷根源：指造成上述错误的根本因素，以寻求软件开发流程的改进、管理水平的提高，如**软件缺陷根源列表**所示。

缺陷根源	描述
测试策略	错误的测试范围，误解了测试目标，超越测试能力等
过程，工具和方法	无效的需求收集过程，过时的风险管理过程，不适用的项目管理方法，没有估算规程，无效的变更控制过程等
团队/人	项目团队职责交叉，缺乏培训,没有经验的项目团队，缺乏士气和动机不纯等
缺乏组织和通讯	缺乏用户参与，职责不明确，管理失败等
硬件	硬件配置不对、缺乏，或处理器缺陷导致算术精度丢失，内存溢出等
软件	软件设置不对、缺乏，或操作系统错误导致无法释放资源，工具软件的错误，编译器的错误，2000千年虫问题等。
工作环境	组织机构调整，预算改变，工作环境恶劣，如噪音过大。

软件缺陷的处理和跟踪

- 软件缺陷跟踪管理是测试工作的一个重要部分，测试的目的是为了尽早发现软件系统中的缺陷，而对软件缺陷进行跟踪管理的目的是确保每个被发现的缺陷都能够及时得到处理。
- 软件测试过程简单说就是围绕缺陷进行的，对缺陷的跟踪管理，一般而言需要达到以下目标：
  - 确保每个被发现的缺陷都能够被解决，“解决”的意思不一定是被修正，也可能是其他处理方式（例如，延迟到下一个版本中修正或者由于技术原因不能被修正），总之，对每个被发现的BUG的处理方式必须能够在开发组织中达到一致；
  - 统计数据：
    - 收集缺陷数据并根据缺陷趋势曲线识别测试处于测试过程中的哪个阶段；
    - 决定测试过程是否结束，通过缺陷趋势曲线来确定测试过程是否结束是常用并且较为有效的一种方式。
    - 收集缺陷数据并在其上进行数据分析，作为组织过程改进的财富。

复杂的软件缺陷生命周期



缺陷跟踪工具

- 项目中使用Microsoft Excel 电子表格或Word 文档来记录和跟踪软件缺陷，
- 一些缺陷管理工具
- 自己开发缺陷跟踪系统
  - 一个缺陷跟踪数据库的基本表，将要包括多达几十项的数据项，如bug的ID号、标题(Title)、状态、严重程度、优先级、重现步骤、期望结果、实际结果、项目名称、模块、报告作者、日期等等。

缺陷管理工具

- 常见免费缺陷管理工具:
  Buggit
  http://www.pb-sys.com/
  Buggit 是一个十分小巧的C/S结构的Access应用软件，仅限于intranet，十分钟就可以配置完成，使用十分简单，查询简便，能满足基本的缺陷跟踪功能，还有十个用户定制域，有十二种报表输出。
  数据库要求:MS Access
- Mantis
  http://mantisbt.sourceforge.net/
  工具描述:
  Mantis是一款基于WEB的软件缺陷管理工具，配置和使用都很简单，适合中小型软件开发团队
  环境: MySQL

  Bugzilla
  http://www.mozilla.org/projects/bugzilla/
  工具描述: 则是一款开源免费的bug跟踪工具。

软件缺陷报告

任何一个缺陷跟踪系统的核心都是“软件缺陷报告”，一份软件缺陷报告详细信息如表：

软件缺陷项目列表

分类	项目	描述
可跟踪信息	缺陷ID	唯一的、自动产生的缺陷ID，用于识别、跟踪、查询
软件缺陷基本信息	缺陷状态	可分为“打开或激活的”、“已修正”、“关闭”等
	缺陷标题	描述缺陷的最主要信息
	缺陷的严重程度	一般分为“致命”、“严重”、“一般”、“较小”等四种程度
	缺陷的优先级	描述处理缺陷的紧急程度， 1是优先级最高的等级，2是正常的，3是优先级最低的
	缺陷的产生频率	描述缺陷发生的可能性1%-100%
	缺陷提交人	缺陷提交人的名字（会和邮件地址联系起来），一般就是发现缺陷的测试人员或其他人员
	缺陷提交时间	缺陷提交的时间

软件缺陷报告

软件缺陷基本信息	缺陷所属项目/模块	缺陷所属的项目和模块，最好能较精确的定位至模块
	缺陷指定解决人	估计修复这个缺陷的开发人员，在缺陷状态下由开发组长指定相关的开发人员；也会自动和该开发人员的邮件地址联系起来，并自动发出邮件
	缺陷指定解决时间	开发管理员指定的开发人员修改此缺陷的时间
	缺陷验证人	验证缺陷是否真正被修复的测试人员；也会和邮件地址联系起来
	缺陷验证结果描述	对验证结果的描述（通过、不通过）
	缺陷验证时间	对缺陷验证的时间
缺陷的详细描述	步骤	对缺陷的操作过程，按照步骤，一步一步地描述
	期望的结果	按照设计规格说明书或用户需求，在上述步骤之后，所期望的结果，即正确的结果
	实际发生的结果	程序或系统实际发生的结果，即错误的结果
测试环境说明	测试环境	对测试环境描述，包括操作系统、浏览器、网络带宽、通讯协议等
必要的附件	图片、Log文件	对于某些文字很难表达清楚的缺陷，使用图片等附件是必要的；对于软件崩溃现象，需要使用Soft_ICE工具去捕捉日志文件作为附件提供给开发人员。

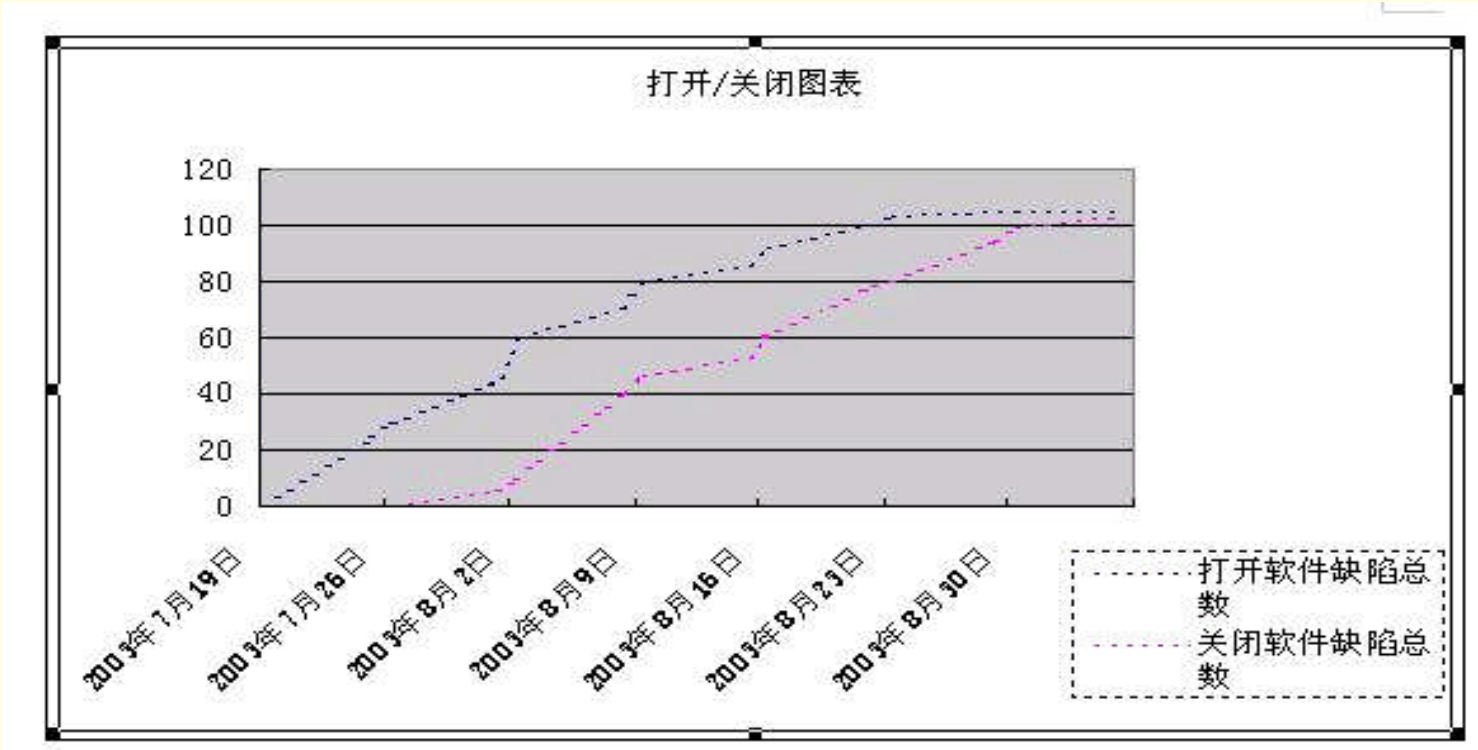
缺陷跟踪的方法和图表

打开/关闭的累积缺陷图

打开曲线顶端稳定，说明测试基本完成；

二者之差稳定，说明项目进展稳定，二者在顶端会合，修复完成；如果相差太远，说明研发与测试不停协调，没有及时修复；

打开曲线突起，说明某个阶段缺陷突增，开发上有问题，值得关注；



小结

本章讲解了应该遵循正规过程正确地描述、分离、分类、记录和跟踪软件缺陷，以保证它们最终得到解决，最好被修复。当项目处于测试执行阶段时，测试管理员管理缺陷主要由分析数据和收集数据组成，因此我们需要建立软件缺陷跟踪数据库存储、搜索和分析软件缺陷，从而生成一系列的图表，分析项目的发展趋势，找到薄弱的领域，合理的修复它。

关键词：

软件缺陷属性

软件缺陷生命周期

软件缺陷报告

软件缺陷管理系统

软件缺陷统计曲线