高可信软件测试

—软件缺陷模式与测试

http://www.51testing.com
上海博为峰软件技术有限公司
2009年11月

- 软件缺陷
- 影响功能的缺陷：难以模型化
- 不影响功能的缺陷：部分可以模型化

软件测试

- 代码走差：人工测试
- 结构测试：白盒覆盖测试
- 功能测试：黑盒功能测试
- 非功能属性测试：系统测试等
- 缺陷测试：属于非功能测试。由程序员遗漏产生，或由低级程序员产生，能影响系统的可靠性、安全性和代码质量。

一、背景

- 高可信软件测试有广泛的需求，在高可信计算领域如航空、航天、电信、电力、核动力、武器装备等，大型软件如ERP，基础软件如操作系统等，一旦发生故障，将会造成巨大的损失，严重时会危及人的生命。对此类软件实施有针对性的面向缺陷的测试，可以大大减少软件的缺陷密度，提高软件的可信性。

软件中某些故障是很难测试的，如存储器泄漏、内存泄漏、数组越界、空指针等，这些类型故障的特点是发生概率小、单个测试用例难以测试，应用传统软件测试方法几乎检测不出来这些故障，而这些故障一旦被激活，往往会导致系统瘫痪。本系统将完成该使命。

二、传统软件测试的局限性

- 自动化程度低
- 发现缺陷能力低
- 主要面向功能缺陷
- 测试评估比较困难
- 小概率故障等检测效率很低
- 测试需要耗费大量的资源（目前美国软件测试已经占软件总成本的53%～87%）
- 有些错误必须通过专门的技术才能测试出来，而用传统软件测试计划测不出来
- 需要很专业的测试人才

三、软件测试目的及发展趋势

一、目的

- 发现软件缺陷
- 验证软件的功能和和非功能属性

二、发展趋势

- 测试能力更强
- 自动化程度更高
- 测试生产线
- 更便于使用

四、面向缺陷的软件测试技术

面向缺陷的软件测试是2000年以来发展起来的全新软件测试技术，该技术的特点是：

- 故障检测效率高
- 可以检测传统软件测试难以检测的故障
- 故障定位准确
- 自动化程度高
- 易学、易用
- 适合高可信软件领域：航空、航天、武器装备、电信、电力、医疗、金融、交通、铁路、核工业、工业控制、办公自动化，基础软件、大型软件等
- 与传统软件测试具有互补性
- 主要面向非功能性缺陷

五、缺陷模式

- 经过传统软件测试后，残留在软件中的缺陷一般都是小概率、开发人员疏忽造成的。我们对8类软件故障的测试实践表明，经过严格的传统软件测试，残留在软件中的故障密度为1~2个故障/10KLOC，而没有经过严格测试的软件，其故障密度一般在5～10个故障/10KLOC，而这些故障一旦发生，往往会导致系统崩溃。软件缺陷模式就是经过理论分析、实践总结归纳出来的，我们目前将其分为故障、漏洞、疑问和规则模式。这是基于缺陷模式测试的核心技术之一。

缺陷模式的概念

(1) 缺陷模式的定义

缺陷模式是缺陷的语法或语义特征的抽象，具有一定代表性或者会造成严重后果 。

(2) 缺陷模式中缺陷产生的原因

- 疏忽：缺陷模式中的缺陷一般是由开发人员疏忽造成的。由于涉及可能多条路径、多个约束条件，疏忽往往是不可避免的。这类缺陷往往都是和路径或多个约束条件相关的。
- 不理解：缺陷模式中的缺陷也可能是由于开发人员对程序语言本身不理解造成的。如对某些规则不了解、对某些语言现象不理解都可能会产生缺陷。

缺陷模式中缺陷的特点：

- 一般都是非功能性的缺陷
- 路径敏感的缺陷一般都是有由疏忽造成的
- 路径不敏感的缺陷一般疏忽或不理解造成的
- 在大量工程软件的测试结果统计中，模式中的缺陷是会存在于大多数的软件中，并且有一定发生概率。这个概率要大于人们容忍的范围。

(3) 软件缺陷模式中的缺陷发生密度

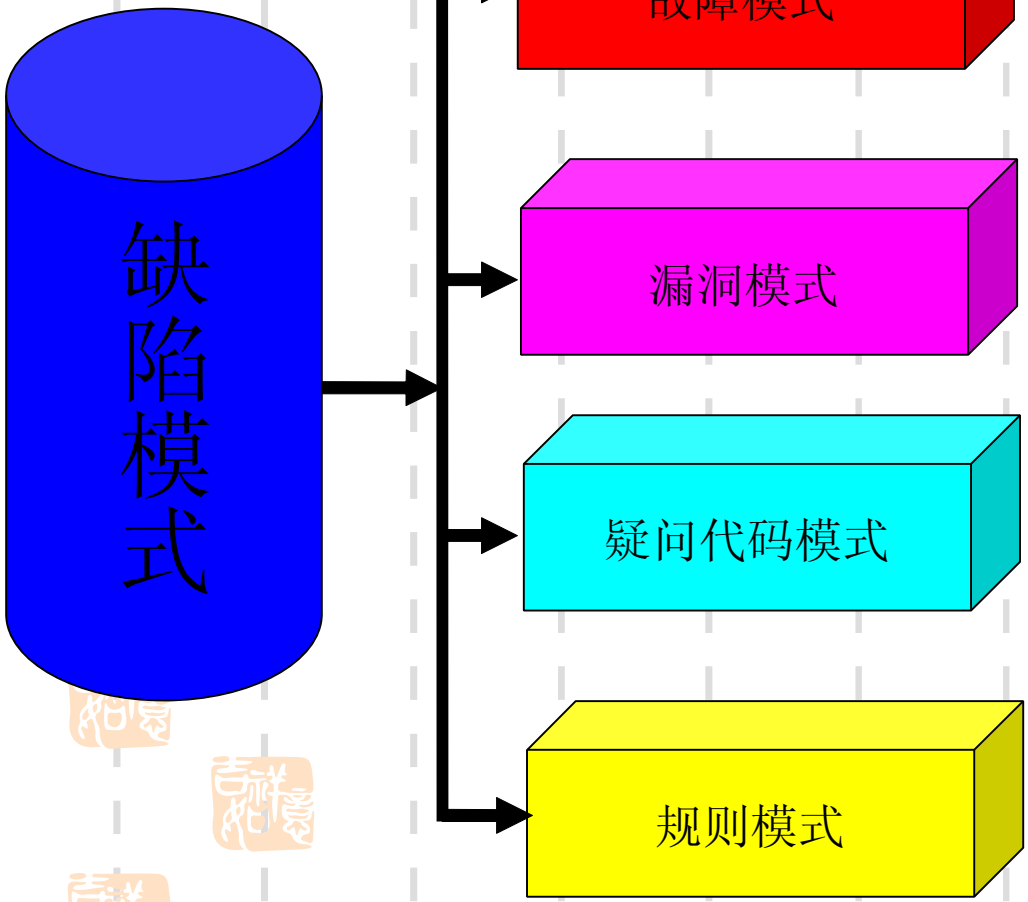
- 在以往的对N个软件测试活动中，该类缺陷的总个数除以N个软件的总行数×1000，称为该类缺陷的缺陷密度，用缺陷个数/KLOC表示，N越大，其缺陷密度统计的准确性就越高。

(4) 缺陷模式的分类——按缺陷产生后的严重程度：故障模式、安全漏洞模式、疑问代码模式和规则模式

(5) 缺陷模式分类——按是否是路径敏感

- 是路径敏感：需要人工确认
- 路径不敏感：不需要人工确认

软件缺陷模式的分类：按缺陷严重程度



会导致系统崩溃
一般由疏忽造成的

会导致系统被攻击
一般由疏忽、不理解造成的

多余、低效的代码
不理解、低水平程序员造成的

不符合规则的代码
低水平程序员造成的

（1）故障模式---此类缺陷是故障，一经产生,会导致系统出错

- 资源泄露模式（Java）
- 指针使用错误模式
- 数组越界模式
- 非法计算模式
- 使用未初始化变量模式
- 死循环结构模式
- 死锁模式
- 存储器泄露模式（C/C++/VC）

（2）安全漏洞模式---此类缺陷会给系统留下安全隐患，为攻击该系统开了绿灯

- 缓冲区溢出模式
- 未验证的外部数据模式
- 竞争条件模式
- 风险操作模式
- 滥用API模式
- 口令泄露模式
- 异常处理不当模式
- 封装不当模式

（3）疑问代码模式---此类缺陷不会产生故障或安全隐患,但是相关代码会引起质疑

- 性能缺陷模式
- 坏习惯模式
- 死码模式

（4）规则模式---软件开发总要遵循一定的规则，开发团队也有一些开发规则

- 代码规则
- 复杂性规则
- 控制流规则
- 命名规则
- 可移植性规则
- 资源规则

六、面向缺陷模式的软件测试的原理

假设$M=\{M_1,M_2,\ldots,M_n\}$是缺陷模式集合，D是检测算法，S是被测软件。基于缺陷模式的测试算法即是D（S,M),即从S中计算出和M中定义的模式相匹配的程序元素$IP=\{IP_1,IP_2,\ldots,IP_L\}$。

- 如果M_i是和路径相关的，则IP_i是否是缺陷需要进一步确认。（目前一般需要靠人工来完成）
- 如果M_i是和路径无关的，则IP_i即是缺陷。

七、系统的精度

- 适用的语言：C/C++/Java约占50%
- 缺陷模式数量：数量与定义的准确性
- 缺陷误报率：一般在70～80%，好的系统可以达到50～60%
- 缺陷漏报率：通过多个系统的比较才能给出。
- 计算复杂性：与误报率和漏报率相关

八、基于缺陷模式测试的三个研究领域