

评估软件测试效果

问题思考：如果你是测试经理/组长，

- 你如何评估和度量**软件测试工作**的效果？
- 你如何评价**测试人员**的工作效果？

上午10时34分

提纲

- ◆ 测试评估概述
- ◆ 测试评估分类
 1. 覆盖评估
 2. 质量评估
- ◆ 软件测试总结报告

上午10时34分


提纲

- ◆ **测试评估概述**
- ◆ 测试评估分类
 1. 覆盖评估
 2. 质量评估
- ◆ 软件测试总结报告

上午10时34分

1.测试评估概述


- ◆ 对软件测试进行评估，目的是解决软件测试过程的疑问，比如：
 1. 每天发现的缺陷有多少？
 2. 发现缺陷的严重程度如何？
 3. 哪些功能引发的软件缺陷最多，哪些最少？
 4. 发现的缺陷中，处于打开状态的有多少，处于解决和关闭状态的又有多少？



上午10时34分

1.1 测试评估的概念

- ◆ 测试评估是在测试结束后（后期）**对整个测试过程与产品进行评估**的过程，主要包括对于测试工作的**总结**、**缺陷数据**的分析以及**测试过程的评估**。



1.2 测试评估的目的

- 检查软件产品是否满足客户需求——最终目的
- 发现软件产品的缺陷——直接目的
- 改进软件开发过程——附带目的/增值目的

测试目的

- ◆ 评估软件产品的质量状况
- ◆ 为产品放行提供决策依据
- ◆ 分析缺陷发现、分布情况
- ◆ 为过程流程改进提供依据
- ◆ 为项目管理提供分析数据
- ◆ 评估项目人员的工作绩效

软件质量评估

生产过程评估

测试评估目的

1.3 测试评估的内容

- ◆ 测试评估包括两方面内容：
 1. **量化测试过程**: 在测试过程中, 不断反馈和总结测试过程中遇到的情况, 然后给项目经理提供有关测试状况的信息, 这被称为**测试监控**
 2. **测试过程改进**: 在测试完成之后, 对整个测试进行**总结、分析**, 从而构建更好的测试模型, 为将来的测试做准备

上午10时34分

7

1.4 测试度量

- ◆ **测试度量 (Test Metrics)**: 用于描述软件测试特定属性的度量单位, 评价测试状况的指标, 包括:
 - 测试人员每天发现缺陷的平均数
 - 缺陷在不同功能区域的分布
 - 缺陷的消除速度等
 - 测试用例执行量
 - 需求或代码的测试覆盖率
 - 测试中严重缺陷发生的时间阶段
 - 测试成本
 -

上午10时34分

8

1.5 测试控制

- ◆ **测试控制 (Test Control)**: 根据收集的测试信息和度量, 需要采取的纠正活动
- ◆ 测试控制活动包括:
 1. 当有明确的风险发生时, 需要**重新设定测试的优先级**
 2. 根据测试环境可用性, **改变测试时间进度表**
 3. **设定入口准则**: 修改后的模块必须经过相关的测试人员测试后才能将它们集成到版本中去

上午10时34分

9

提纲

- ◆ 测试评估概述
- ◆ **测试评估分类**
 1. **覆盖评估**
 2. **质量评估**
- ◆ 软件测试总结报告

上午10时34分

10

2 测试评估分类

- ◆ 测试评估贯穿软件测试的整个过程
- ◆ 测试评估方法主要包括**覆盖评估**和**质量评估**
 1. **覆盖评估**是对测试完全程度的评价, 其建立在**测试覆盖**的基础上, 通常与完成计划的程度相关
 2. **质量评估**则是对测试软件的整体质量状况的评估, 其建立在测试过程中发现的软件缺陷的分析和修复的基础上

上午10时34分

11

2.1 覆盖评估

- ◆ **覆盖评估**用来度量软件测试的完成程度
- ◆ 最常用覆盖评估是:
 1. 基于**代码**的测试覆盖: Cover every statement
 2. 基于**需求**的测试覆盖: Cover every functional requirement

上午10时34分

12

2.1.1 基于代码的测试覆盖

- ◆ 基于代码的测试覆盖：测试执行了多少代码，已经测试了多少路径
- ◆ 白盒测试中基于控制流测试的技术是代码测试覆盖的基础
- ◆ 基于代码测试覆盖率的计算公式如下：

$$\text{Coverage_Code} = (\text{Tc} / \text{TP})$$

Coverage_Code：代码测试的覆盖率

Tc：已执行的代码数（路径或代码）

TP：测试计划确定的被测代码数

上午10时34分

13

2.1.2 基于需求的测试覆盖

- ◆ 基于需求的测试覆盖：完成的测试需求百分比，可以用来监控测试的完成进度
- ◆ 基于需求的测试覆盖度量指标包括3种：
 1. 计划的测试覆盖程度
 2. 已实施的测试覆盖程度
 3. 已执行成功的测试覆盖程度

上午10时34分

14

2.1.2.1 计划的测试覆盖程度

- ◆ 计划的测试覆盖程度

$$\text{Coverage_Plan} = (\text{TP} / \text{RfT})$$

Coverage_Plan：计划测试需求的覆盖率；

TP：测试计划确定的测试用例的数量

RfT：测试需求的总数

- ◆ RfT是进行完全需求测试的理论测试用例总数，如，为了覆盖所有的功能，共需设计10000个用例。但在具体测试设计时，可能只设计5000个用例，则计划的测试覆盖率将是50%。

上午10时34分

15

2.1.2.2 已实施的测试覆盖程度

- ◆ 已实施的测试覆盖程度是指完成的测试覆盖
- ◆ 已实施的测试覆盖程度

$$\text{Coverage_Done} = (\text{TD} / \text{TP})$$

Coverage_Done：已实施测试的覆盖率；

TD：到目前为止已完成的测试用例数

TP：测试计划确定的测试用例数量

上午10时34分

16

2.1.2.3 成功的测试覆盖程度

- ◆ 成功的测试覆盖程度表示执行成功的测试比率。
- ◆ 成功的测试是指测试时没有发现软件缺陷，或者即使发现了缺陷，但是程序员进行了修改并重新测试通过。
- ◆ 计算公式

$$\text{Coverage_Success} = (\text{TS} / \text{TP})$$

Coverage_Success：已成功测试的覆盖率；

TS：已成功完成测试的测试用例数量

TP：测试计划确定的测试用例数量

上午10时34分

17

2.2 测试质量评估

- ◆ 覆盖评估能够有效地评价测试完成的量，但对于测试的质不能很好地评价。在测试的过程中，还需评估缺陷本身的状况，与缺陷相关的测试评估称为测试质量评估
- ◆ 测试质量评估与缺陷的分析统计密切相关。
- ◆ 缺陷分析提供了软件的可靠性指标，为揭示软件的缺陷趋势及缺陷分布提供了判定数据

上午10时34分

18

2.2.1 质量评估的度量

- ♦ 用于缺陷分析的4类软件测试度量

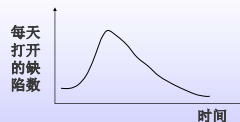
 1. 缺陷发现率 (Defect Discovery Rate)
 2. 缺陷潜伏期 (Software Defect Latent Period)
 3. 缺陷密度 (Software Defect Density)
 4. 整体软件缺陷的累积及清除率

上午10时34分

20

2.2.2 缺陷发现率

- ♦ **缺陷发现率**：平均每天发现的缺陷数与时间的关系。通过对缺陷发现率作图，可以看到随着时间的推移，缺陷发现率的变化趋势



上午10时34分

21

2.2.3 缺陷潜伏期

- ♦ **缺陷潜伏期**：缺陷引入阶段与缺陷发现阶段之间的时间差
- ♦ 通常而言，缺陷发现的时间越晚，修复缺陷所需要花费的代价越大

缺陷造成阶段	缺陷发现阶段 (缺陷潜伏期 = 缺陷发现阶段 - 缺陷造成阶段)							
	需求分析	概要设计	详细设计	编码	单元测试	集成测试	系统测试	验收测试
需求分析	0	1	2	3	4	5	6	7
概要设计		0	1	2	3	4	5	6
详细设计			0	1	2	3	4	5
编码				0	1	2	3	4

上午10时34分

22

2.2.4 软件缺陷密度

- ♦ **软件缺陷密度**：单位代码量所引入的软件缺陷个数。代码量通常以千行为单位
- ♦ 计算公式：

$$\text{Defect_Density} = (D / T)$$

Defect_Density：缺陷密度

D：软件缺陷总数

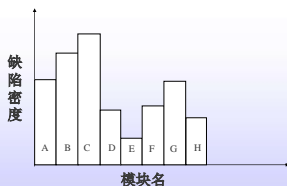
T：被测软件的代码行或功能点数量

比如，某项目有100千行代码，测试共发现300个缺陷，则每千行的缺陷密度为3 (=300/100)

上午10时34分

23

软件缺陷密度在各模块的分布



上午10时34分

24

软件缺陷严重性分布

- ♦ 通过饼图对缺陷的严重性进行总结，以便知道缺陷的严重程度，修复的优先级



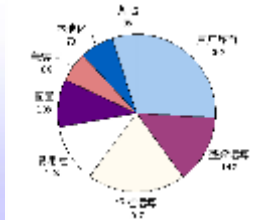
可以看出，这个软件的缺陷非常严重，需要立即修复

上午10时34分

25

软件缺陷的功能分布

- 统计缺陷主要发生在哪些功能区，从而为缺陷的改进指明方向



上午10时34分

23

2.2.5 软件缺陷的累积及清除率

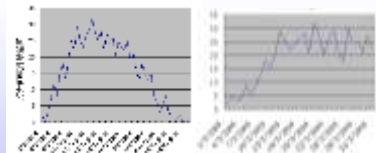
- 为了表达缺陷的清除率，引入一些参数
 - F: 软件规模，代码行或功能点数
 - D1: 软件开发过程中（含测试）发现的缺陷
 - D2: 软件发布后发现的缺陷
 - D: 发现的总缺陷数， $D = D1 + D2$
- 根据几个公式来评估软件项目的质量：
 - 软件质量（每功能点的缺陷数） $= D2 / F$
 - 软件缺陷注入率 $= D / F$
 - 整体软件缺陷清除率 $= D1 / D$

上午10时34分

24

对缺陷的趋势进行统计

- 对每天发现的缺陷数量进行统计，形成缺陷趋势图，从而评估当前软件是否可以发布



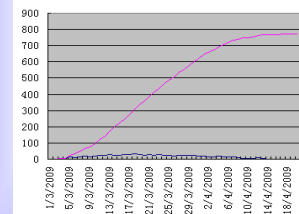
软件缺陷减少

软件缺陷平稳

上午10时34分

27

对缺陷的累计趋势进行统计



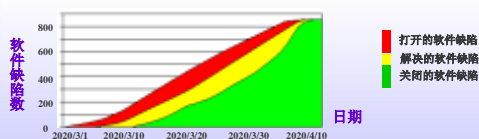
红色曲线为累计打开的软件缺陷，可以看到随着时间的推移，其累计趋于平稳

上午10时34分

28

统计软件缺陷修复情况

- 将三种状态（打开，解决和关闭）的缺陷进行比较，从而指导当前的测试和开发工作



上午10时34分

29

提纲

- 测试评估概述
- 测试评估分类
 - 覆盖评估
 - 质量评估
- 软件测试总结报告

上午10时34分

30

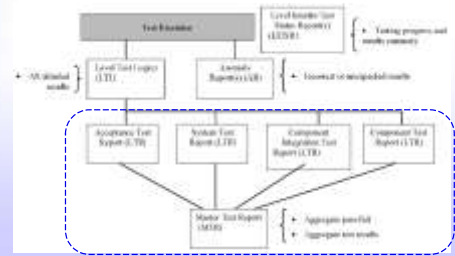
3 软件测试总结报告

- ◆ 测试完成后，需要书写软件缺陷总结报告
- ◆ 测试总结报告包含以下内容：
 1. 测试总结报告标识符
 2. 概述
 3. 差异
 4. 广泛评价
 5. 结果总结
 6. 评估
 7. 活动总结
 8. 批准

上午10时34分

31

IEEE829-2008推荐的测试报告类型



上午10时34分

32

IEEE829-2008 Master Test Report Outline

- | | |
|---|--|
| <ol style="list-style-type: none"> 1. Introduction <ol style="list-style-type: none"> 1.1. Document identifier 1.2. Scope 1.3. References 2. Details of the Master Test Report <ol style="list-style-type: none"> 2.1. Overview of all aggregate test results 2.2. Rationale for decisions 2.3. Conclusions and recommendations 3. General <ol style="list-style-type: none"> 3.1. Glossary 3.2. Document change procedures and history | <ul style="list-style-type: none"> •Summary: testing activities, testing task results, anomalies and resolutions, final metrics collected •Assessment of release/increment quality Specify the reasons for the pass /fail /conditional-pass conclusions for the system. Specify an overall evaluation of the system. |
|---|--|

上午10时34分

33

The End
Any Question?



34