七、软件测试管理

1. 测试计划
2. 测试用例管理
3. 缺陷追踪管理
3. 管理测试小组

- 本章为自行了解内容
- 重点了解什么是缺陷管理（结合实验一中的缺陷跟踪实验）

1. 测试计划

- 测试计划概述
- 测试计划的主要内容
- 编写有效测试计划必须考虑的问题

1.1 测试计划概述

- 什么是测试计划
  - 项目范围内的**测试目的**和**测试目标**的有关信息
  - 确定实施和执行测试时所使用的**策略**以及所需**资源**
  - 对测试**风险**等做出预先的**计划**和**安排**

- 测试计划包括
  - **测试主计划**：项目开始时制订测试主计划
  - **阶段计划**：根据开发的过程和测试主计划对测试计划进行细化，制订各个阶段的测试计划

1.1 测试计划概述

- **制定测试计划的目的**
  1. 使软件测试工作进行更顺利
  2. 促进项目参加人员彼此的沟通
  3. 使软件测试工作更易于管理

- **制定测试计划的原则**

  制定测试计划是软件测试中最有挑战性的一个工作。以下原则将有助于制定测试计划工作。

  – 尽早开始
  – 保持测试计划的灵活性
  – 保持测试计划简洁和易读
  – 尽量争取多渠道评审测试计划
  – 计算测试计划的投入

1.2 测试计划的主要内容(国军标)

- 1 范围
- 2 引用文档
- 3 软件测试环境
- 4 正式合格性测试
- 5 数据记录、整理和分析

1.2 测试计划的主要内容(国军标)

1 范围

1.1 标识： 列出已批准的标识号；标题；缩略语；本文档适用的系统和计算机软件配置项（**CSCI**）等

1.2 系统概述：概述待测系统和CSCI 的用途

1.3 文档概述：概述本文档的用途和内容

1.4 与其它计划的关系：概述本计划与其它计划的关系

2 引用文档

按文档号和标题列出本文档引用的所有文档

1.2 测试计划的主要内容(国军标)

- **3 软件测试环境**

描述为执行测试所使用软硬件资源的实现和控制计划

3.1 软件项

软件环境：说明每个软件的用途、保密处理和安全性问题等。

3.2 硬件和固件项

硬件环境：说明每项的用途、保密处理和安全性问题等。

3.3 权限

标明软件测试环境相关的每个项目的专利和权限。

3.4 安装、测试与控制

环境建立相关的安装、测试与控制计划

其它

- **人员**：人数、经验和专长。他们是全职、兼职、业余还是学生？
- **办公室和实验室空间**：哪？多大？怎样排列？
- **其他资源**——电话、参考书、培训资料等。

1.2 测试计划的主要内容(国军标)

- **4 正式合格性测试**

分节对每个CSCI 作正式合格性测试的说明和要求

4.X （CSCI 名称和项目唯一标识号）

4.X.1 总体测试要求

进度、充分性、错误处理方面的测试要求等。

对不同的实际问题应外加相应的专门测试。

4.X.2 测试类

要进行的测试的种类（如：强度测试、时间性测试、错误输入测试、最大能力测试等）

4.X.3 测试级

描述要进行的正式合格性测试的级别，如：单元级、集成级、软硬件集成级、系统级等

1.2 测试计划的主要内容(国军标)

4.X.4 测试定义

分节标识和描述用于CSCI 的各项测试。

给出测试用例信息，可用图表：

- 测试用例信息；
- 测试种类或类型；
- 对需求规格说明中相关内容的引用；

4.X.5 测试进度

- **5 数据记录、整理和分析**
  - 指明数据整理和分析过程。
  - 说明根据数据整理和分析得到的信息和结果。
  - 结果应清楚地显示出是否达到测试目标。

测试计划模版

- 参考文档：
  - 测试计划1
  - 测试计划2

国家标准中的13类测试

国防软件的测评中，经常会展开以下13类测试

1. 数据和数据库完整性测试
2. 接口测试：只测输入不管输出的测试
3. 集成测试
4. 功能测试
5. 用户界面测试
6. 性能评测
7. 负载测试
8. 强度测试
9. 容量测试
10. 安全性和访问控制测试
11. 故障转移和恢复测试
12. 配置测试
13. 安装测试

1.3 编制有效的测试计划应考虑的问题

- 了解任务和测试目标
- 考虑风险
- 根据功能优先级安排测试工作
- 规划测试环境

了解任务和测试目标

- 怎样了解?
  - 理解系统：功能需求+非功能需求
    - 涉及整个系统的讨论会和文档
  - 及早介入（测试经理等）：增加对客户需求、客户问题、潜在的风险和功能的理解
  - 理解企业文化和过程
    - 测试组和开发组独立还是一体化？
    - 测试方法是否适应“极限编程”的方法？

了解任务和测试目标

- 测试对象的范围
- 测试的期望：管理层、客户的期望
- 吸取教训：以前的测试工作中学到了什么？（确定测试策略、设定实际的测试预期）
- 工作量大小：初步估计项目的复杂度的工作量
- 解决方案的类型：最终是实现了最复杂的解决方案？较短时间开发、更划算的解决方案？（决定采用的测试类型）

了解任务和测试目标

- 技术选择：
  - 实现技术？引起的问题？架构？系统类型？（确定测试策略、选择测试工具）
- 预算（确定测试类型、测试工作量）
- 时间表：系统测试的时间？截止日期？
- 测试环境所需的硬件和软件
  - 评估、购买和实现测试工具
- 分阶段的解决方案（迭代开发？发行许多版本？）

考虑风险

- 测试难以穷尽，分析风险将有助于：
  - 排定待测试项的优先顺序，集中精力去关注那些极有可能发生失效的领域。
  - 根据技能的高低、工作量、风险和质量目标分配测试人员

- 下面是一些潜在的问题和风险的例子：
  - 不现实的交付日期
  - 与其他系统的接口
  - 极其复杂、关键的软件
  - 有过缺陷历史的模块
  - 发生过许多或者复杂变更的模块
  - 安全性、性能和可靠性问题
  - 难于变更或测试的特征

考虑风险

- 考虑风险级别

  考虑损失发生的可能性和损失带来的影响的严重程度

  - 风险分类和来源
  - 风险评估
  - 降低风险的测试策略
    - 把重点放在可能会引起绝大多数问题的那些部分
    - 对风险低和影响小的功能，只执行必须的测试工作，并可为新手提供积累经验的机会
    - 产生可预测的、更高质量的测试结果

根据优先级安排测试工作

- 划分优先级的依据
  - 功能的重要性
  - 风险最高到最低
  - 复杂度最高到最低
  - 客户的需要（市场和销售）
  - 预算的限制
  - 时间的限制
  - 人员限制（特殊需求？谁来做？）
- 综合使用以上方法，得到一个功能的总体价值，并进行排序，得到功能优先级表。

优先级确定相关的常见问题

- 哪些功能是软件的特色？
- 哪些功能是用户最常用的？
- 如果系统可以分块卖的话，哪些功能块最昂贵？
- 哪些功能出错将导致用户不满或索赔？
- 哪些程序是最复杂、最容易出错的？
- 哪些程序是相对独立，应当提前测试的？
- 哪些程序最容易扩散错误？
- 哪些程序是全系统的性能瓶颈所在？
- 哪些程序是开发者最没有信心的？

规划测试环境

- 测试环境：支持测试工作的所有物质元素
- 设计测试环境
  - 获得客户环境的样本（支撑软硬件）
  - 确定网络特性（带宽、网络协议等）
  - 确定服务器OS
  - 确定执行某些测试过程需要的其他软件
  - 确定硬件环境时考虑测试数据的需求（规模）
  - 考虑配置测试需要的特殊资源（活动硬盘和图像库）

2 测试用例的管理

- 测试用例的管理属性有那些？
- 测试用例体本身的属性有那些？
- 测试用例管理系统

2.1 测试用例的管理属性

- 项目名称
- 目标行业：银行、电信、交通、电子、.....
- 系统类型：嵌入式、b/s、c/s、其它；
- 运行平台：操作系统、数据库管理系统、浏览器等
- 编码语言：Java、C/C++、Php、.....
- 测试类型：功能、兼容性、性能测试、安全性测试等
- 测试阶段
- 创建人/创建时间
- 重要级别
- 状态

2.2 测试用例体本身的属性

- 测试用例名称
- 测试用例目的
- 测试用例版本
- 相关附件
- 测试用例描述方式：文本、测试脚本、可执行程序等
- 测试用例程序文件、前置条件、输入、操作步骤
- 测试用例预期输出
- 与复用操作有关的属性：父测试用例id、修改原因、修改时间、修改人员

2.3 测试用例管理系统

- 国外著名的有
  - IBM Rational Test Manager
  - Compueware QADirector
  - HP TestDirector
- 国内比较有名的有
  - 中科院的I-test
  - 北航的QESuite等

3 缺陷追踪管理

- 缺陷属性
- 缺陷的处理流程
- 缺陷的分离和重现
- 缺陷的度量及其意义
- 缺陷管理系统

3.1 缺陷属性

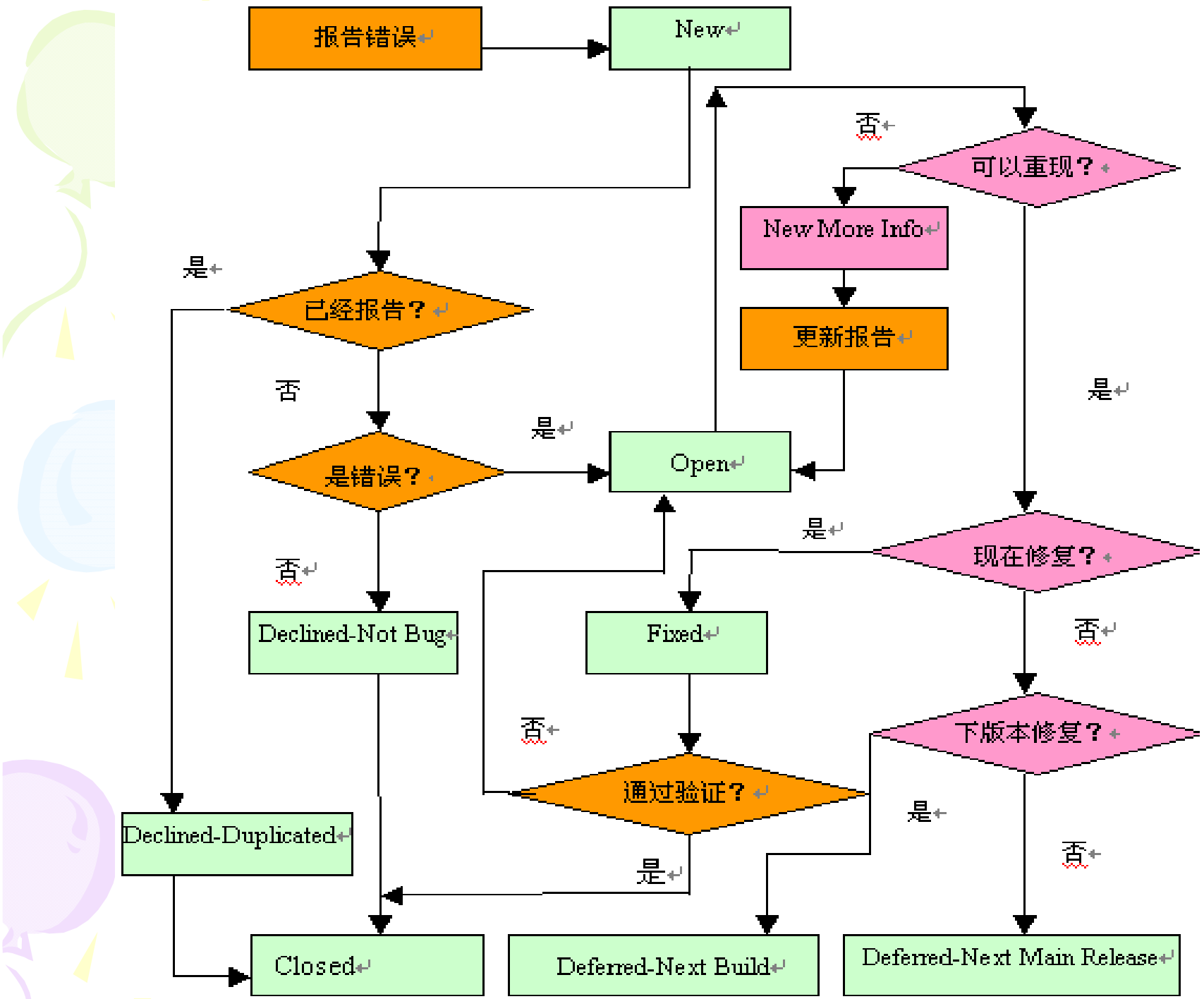
- 缺陷ID
- 状态：新发现、正在修改、待确认、修改完毕
- 测试人员
- 提交时间
- 缺陷归属
  - 所属项目、所属模块、开发人员
- 缺陷类型
- 优先级： Urgent， High，Medium， Low
- 缺陷修复信息
  - 修改人、修改时间、修改次数
  - 解决方案：提出解决当前缺陷的方案并给出修改部分代码
- 确认结果： 缺陷已修复、缺陷还需再次修改

缺陷优先级和严重性划分

缺陷类别	标识/权值	说明
A 类	A1/5.6	由程序执行引起的死机、非法退出
	A2/5.5	死循环
	A3/5.4	数据库发生死锁
	A4/5.3	数据库设计未达第三范式的要求或需求规格说明的水平
	A5/5.2	数据功能实现错误
	A6/5.1	与数据库连接错误
	A7/5.0	数据通讯错误
B 类	B1/4.3	程序语法错误
	B2/4.2	因错误操作迫使程序中断
	B3/4.1	程序接口错误
	B4/4.0	数据库的表、业务规则、缺省值未加完整性等约束条件
C 类	C1/3.4	操作界面错误
	C2/3.3	打印内容、格式错误
	C3/3.2	简单的输入限制未放在前台进行控制
	C4/3.1	删除操作未给出提示
	C5/3.0	数据库表中有过多的空字
D 类	D1/2.5	界面不规范
	D2/2.4	辅助说明描述不清楚
	D3/2.3	输入输出不规范
	D4/2.2	长操作未给用户提示
	D5/2.1	提示窗口文字未采用行业术语
	D6/2.0	可输入区域和只读区域没有明显的区分标志
E 类	E1/6.1	遗漏部分功能
	E2/6.1	实现功能与需求不相吻合

3.2 缺陷的处理流程

- 缺陷报告最初生成的状态为“新”；
- 选择缺陷优先级
- 评估缺陷，为缺陷分配状态
- 若状态为“打开”，则把缺陷分配给负责的人，变为“开发”状态
- 开始改正缺陷了，变为“正在开发”状态
- 缺陷改正完了，改为“修改完毕”状态；或者“工作正常”、“缺陷不能重现”
- 若创建了新版本，所有改正的缺陷改为“返测”状态
- 测试工程师返测这些改动，设置状态为“关闭-改正”、“返测失败”


报告错误
New
否
可以重现？
New More Info
是
已经报告？
更新报告
否
是
是
是错误？
Open
是
现在修复？
否
Declined-Not Bug
Fixed
否
下版本修复？
否
通过验证？
是
Declined-Duplicated
是
否
Closed
Deferred-Next Build
Deferred-Next Main Release

Red Hat Bugzilla – Browse: Fedora

Home | New | Search | Browse | Search [?] | Reports | Requests | Help | New Account | Log In | Forgot Password

Note: This product browsing page is BETA release and subject to change

Browse: Fedora

Fedora ▼ all ▼ All Bugs ▼ Change

- Create a new bug in this product
- Show full component descriptions for this product

Bugs related to the components of the Fedora distribution. If you are reporting a bug against a stable release or a branched pre-release version please select that version number. The currently maintained released versions are Fedora 20, Fedora 21 and Fedora 22. If you have a bug to report against the daily development tree (rawhide) please choose 'rawhide' as the version.

Summary | Recents | Components/Versions | Duplicates | Road Map

Recents

Activity within the last 7 days (between 1 and 100) or from [ ] to [ ] Change

Recently Opened | Recently Closed

771 Recently Opened Bugs

<< first < prev 1 2 3 4 5 6 7 8 9 10 next > last >>

ID	Status	Version	Component	Severity	Summary
1227588	NEW	22	ark	unspecified	Cannot Unrar Files - Looking for Incorrectly Spelled "Unar"
1227585	NEW	22	kmenuedit	low	KMenuEdit missing on Fedora KDE
1227568	NEW	21	isight-firmware-tools	unspecified	[abrt] isight-firmware-tools: extract(): ift-extract killed by SIGTRAP
1227554	NEW	22	kernel	unspecified	use of USB DVD-RW drive in rhythmbox causes ATA reset, system crash
1227553	NEW	22	gnome-shell	medium	The spinning wheel mouse pointer is being draw at the wrong position.
1227550	NEW	rawhide	python-zope-proxy	unspecified	python-zope-proxy-4.1.6 is available
1227548	NEW	rawhide	azureus	unspecified	azureus-5610 is available

Bug 1227560 – [abrt] evolution: _dl_map_object_deps(): evolution killed by SIGTRAP

Status: CLOSED NOTABUG
Aliases: None
Product: Fedora
Component: evolution (Show other bugs)
Version: 21
Hardware: x86_64 Unspecified
Priority unspecified **Severity** unspecified
Target Milestone: ---
Target Release: ---
Assigned To: Milan Crha
QA Contact: Fedora Extras Quality Assurance
Docs Contact:
URL: https://retrace.fedoraproject.org/faf...
Whiteboard: abrt_hash:e3136c65a8846a5467a6c3de277...
Keywords:
Depends On:
Blocks:
Show dependency tree / graph

Reported: 2015-06-02 22:50 EDT by Ernie D
Modified: 2015-06-03 00:39 EDT (History)
CC List: 4 users (show)
See Also:
Fixed In Version:
Doc Type: Bug Fix
Doc Text:
Clone Of:
Environment:
Last Closed: 2015-06-03 00:39:06 EDT

Attachments (Terms of Use)		
File: backtrace (61.69 KB, text/plain) 2015-06-02 22:50 EDT, Ernie D	*no flags*	Details
File: cgroup (190 bytes, text/plain) 2015-06-02 22:50 EDT, Ernie D	*no flags*	Details
File: core_backtrace (37.20 KB, text/plain) 2015-06-02 22:50 EDT, Ernie D	*no flags*	Details
File: dso_list (27.66 KB, text/plain) 2015-06-02 22:50 EDT, Ernie D	*no flags*	Details
File: environ (1.30 KB, text/plain) 2015-06-02 22:50 EDT, Ernie D	*no flags*	Details
	flags	Details

Ernie D **2015-06-02 22:50:16 EDT** **Description**

Description of problem:
I Simply clicked on the minimized icon to view latest incoming messages. When i did, Evolution just closed.

Version-Release number of selected component:
evolution-3.12.11-1.fc21

Additional info:
reporter: libreport-2.3.0
backtrace_rating: 4
cmdline: evolution
crash_function: _dl_map_object_deps
executable: /usr/bin/evolution
kernel: 3.19.5-200.fc21.x86_64
runlevel: N 5
type: CCpp
uid: 1000
var_log_messages: [System Logs]:\n-- Logs begin at Mon 2015-05-04 13:03:25 PDT, end at Tue 2015-06-02 19:35:38 PDT. --

Truncated backtrace:
Thread no. 1 (1 frames)
#26 _dl_map_object_deps at dl-deps.c:470

Milan Crha **2015-06-03 00:39:06 EDT** **Comment 10**

Thanks for a bug report. this crashed due to:
> Settings schema 'apps.gecko-mediaplayer.preferences' is not installed
which was tried to be opened somehow through libasound. Why evolution tries to load it I do not know, it might be some other library's dependency or a 3rd party browser plugin you've probably installed. This is not a bug in the Evolution, it's a bug in the installation of that software.

3.3 缺陷的分离和重现

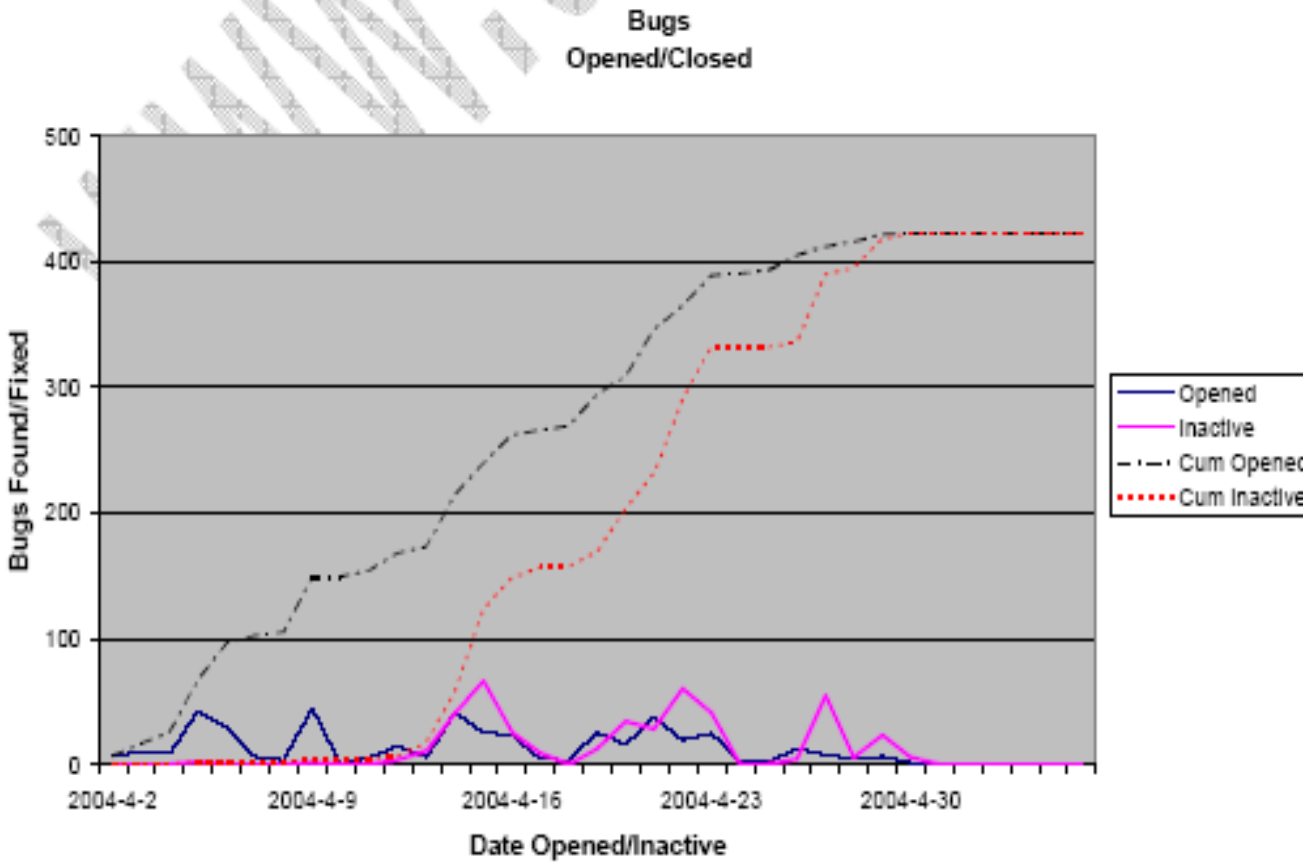
- 如何有效地报告缺陷?
  - 描述清晰，包含再现步骤
- 分离缺陷
  - 记录每一个步骤\每一个停顿\每一件工作
  - 查找时间依赖和竞争条件的问题(时间发生次序)
  - 与负荷相关的边界条件\内存泄露\数据溢出
  - 考虑资源依赖性和内存\网络\硬件共享的相互作用

3.4 缺陷的度量及其意义

- 缺陷管理的意义
  - 确保发现的缺陷都能够得到适当的处理
  - 对软件开发过程进行管理控制
- 缺陷的分析和度量
  - 了解缺陷集中的区域
  - 明晰缺陷发展趋向，预测软件质量
  - 度量软件开发过程中各阶段工作产品的质量
  - 评估开发人员的效率、测试人员的效率
  - 评估项目进展的情况，改进软件过程
- 软件开发只有引入了度量机制和定量化的管理，才能称为真正意义上的“工程”。

缺陷度量—缺陷的发展趋势

- 新发现缺陷数量增长趋势和关闭缺陷数量的增长趋势
  - 对于软件产品发布而言，发展趋势图是辅助决策的重要依据。
  - 一般来说，软件发布的必要条件是新缺陷的数量增加呈下降趋势.



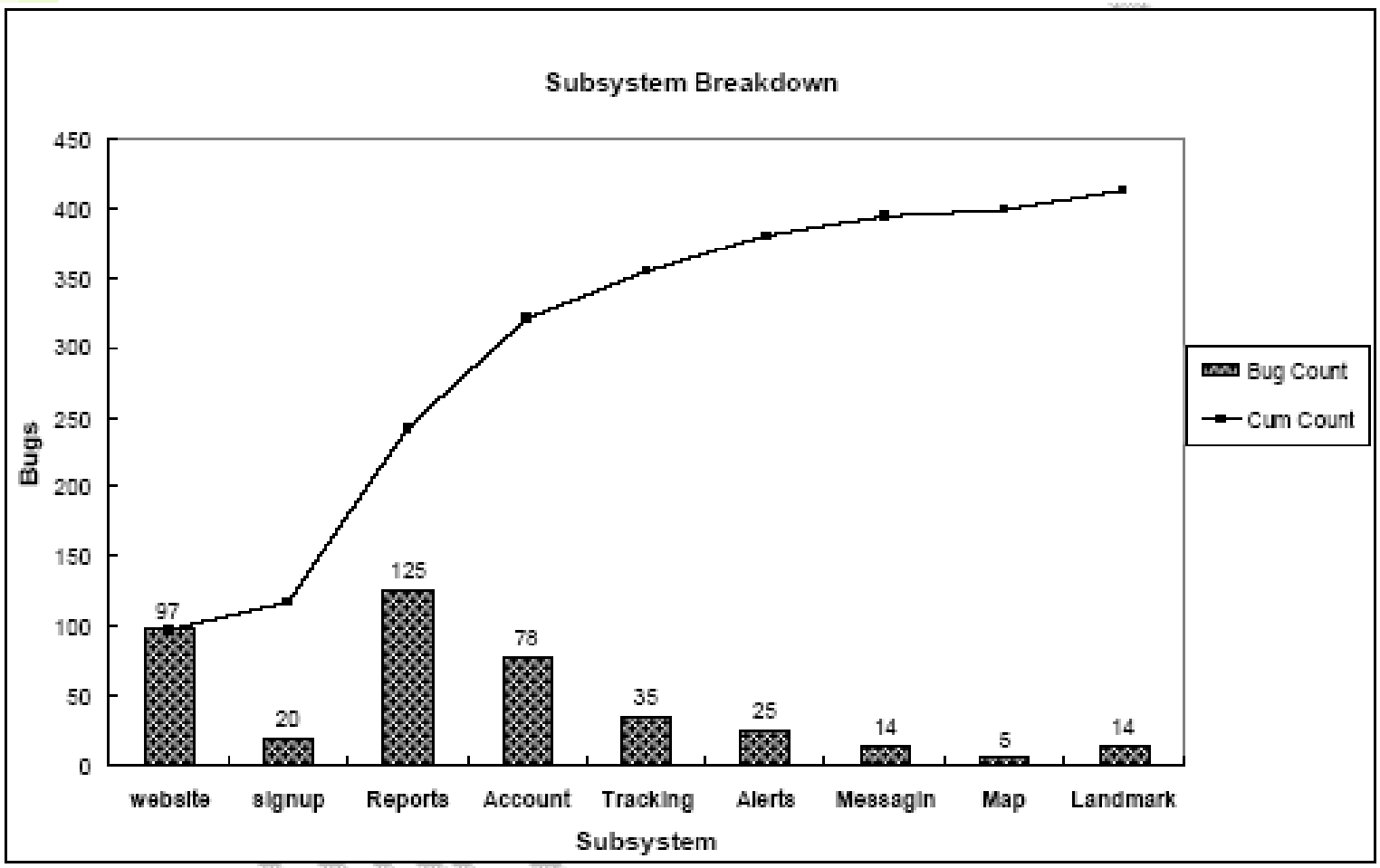
缺陷度量—缺陷的分布状况

- 缺陷分布状况图
  - 有助于了解各版本中缺陷数量的分布
  - 回归测试阶段中，缺陷分布可以直接反映出版本的质量状况

- 缺陷分布状况图有两种
  - 一种是缺陷按模块的分布状况

    评估各模块质量水平，开发难度。

    从侧面反映出测试资源在各模块分布的情况
  - 一种是缺陷按产生的根本原因的分布状况


Subsystem Breakdown
Bugs
450
400
350
300
250
200
150
100
50
0
97
20
125
78
35
25
14
5
14
website
signup
Reports
Account
Tracking
Alerts
Messagin
Map
Landmark
Subsystem
Bug Count
Cum Count

3.5 缺陷管理系统

- 代表性的有：
  - 开源软件Bugzilla、Jira
  - Compuware TrackRecord
  - Rational ClearQuest
  - ……

- 共同缺点：

  没有充分利用缺陷数据，不能以一种主动的、精确量化的方式对软件缺陷进行预防并提供软件项目管理者所需的有关产品和过程的度量信息。

4. 管理一个测试小组

1.企业的测试策略

2.测试人员的组织

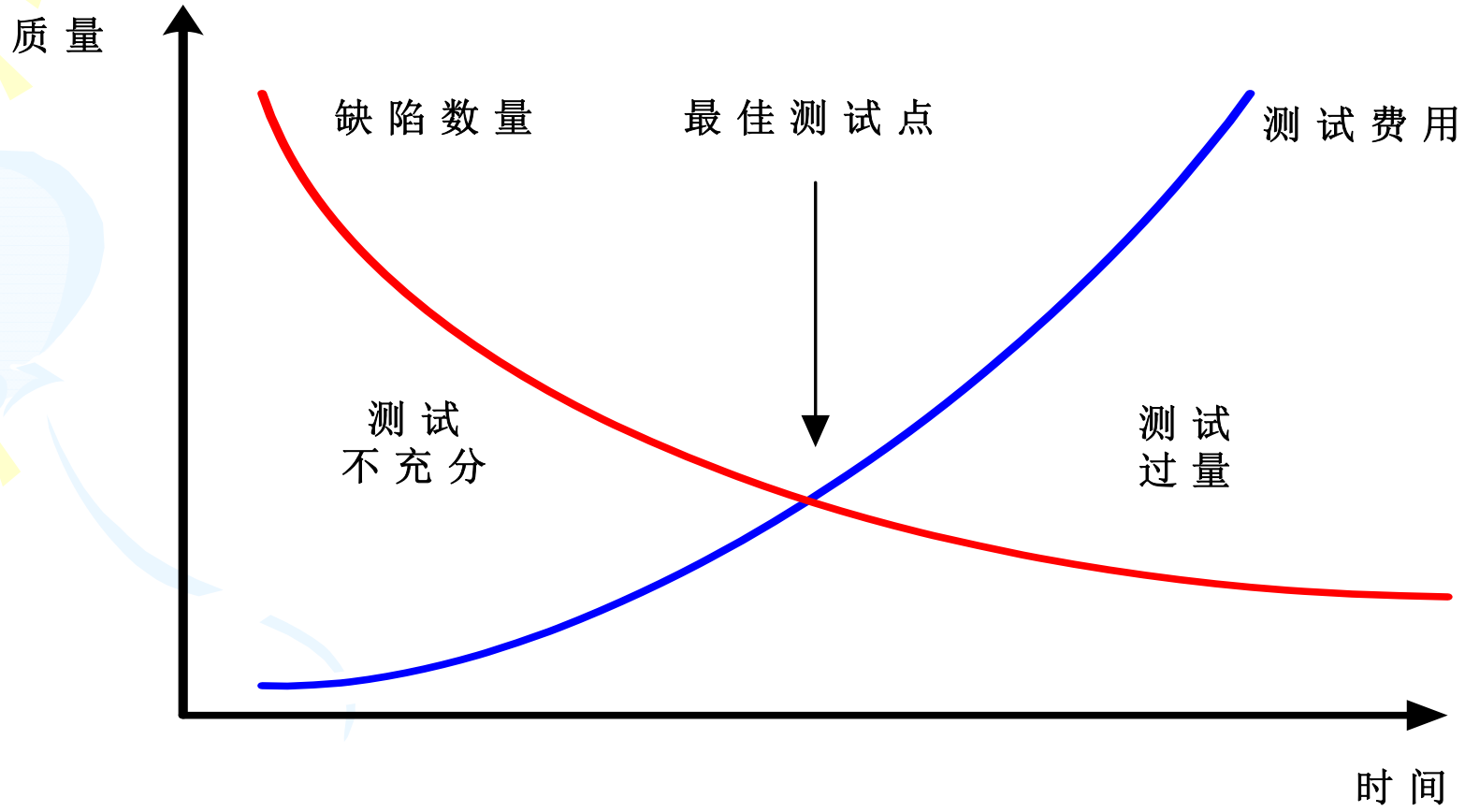
3.测试管理的实施

4.测试部门的评估

4.1 企业的测试策略

- 理念
  - 用较低的代价实现有效的测试
  - 不应为了追求完美的测试而不失一切代价
- 如何合理地减少测试工作量
  - 减少冗余的测试
    - 测试目标、达到的效果一样——冗余
    - “回归测试”中应避免冗余
  - 减少无价值的测试
    - 错误的目标、错误的方式
  - 确定测试的优先主次

测试的经济学



“Too little testing is a crime—too much testing is a sin.”

4.2 测试人员的组织

- 如何组织测试人员
  - **条件特别好的公司：** 为每个开发人员分配一名测试人员。测试人员职业化程度很高，可以完成单元、集成和系统测试，能够实现开发与测试同步进行。
  - **条件比较好的公司：** 设置一个独立的测试小组，轮流参加各个项目的系统测试。单元、集成测试工作由开发小组承担。
  - **条件一般的公司：** 单元、集成测试由开发小组承担。系统测试阶段，可以从项目外抽调一些人员，加上开发人员，临时组织系统测试小组。
  - **条件比较差的公司：** 开发人员一直兼任测试人员的角色，相互测试对方的程序。开发者测试自己的程序.

4.2 测试人员的组织

- 避免开发人员与测试人员产生矛盾
  - 开发人员的注意事项：
    - 不要敌视测试人员
    - 不要轻视测试人员
  - 测试人员的注意事项：
    - 发现缺陷时不要嘲笑开发人员，别说他的程序烂
    - 开发人员压力大或心情不好时不要火上浇油，发现缺陷时别大声嚷嚷
  - **请留意另一种极端：**如果测试人员与开发人员的关系好，可能会在测试时“手下留情”，这对项目也是一种伤害。

4.3 测试管理的实施

- 测试管理中的PDCA
  - P：测试计划
  - D：测试案例及测试步骤的设计
  - C：测试实施和错误跟踪
  - A：测试总结与报告

- 软件测试文件描述要执行的软件测试及测试的结果
- 测试文件的编写是测试工作规范化的一个组成部分
- 开始于需求分析阶段、使用于整个生命周期中

4.4 测试部门的评估

- 测试经理必须跟踪、监督和评估测试工作的实现，并在必要时进行改进。
  - 评估测试人员的有效性
    - 对测试人员的期望
    - 评估测试人员测试工作的要点
  - 评估测试组的有效性
    - 角色和职责
    - 行业知识、测试技巧和经验
  - 评估测试组测试活动质量的几个方面

A. 评估测试人员的有效性

- 对测试人员的期望
  - 遵守测试标准和测试过程
  - 保持进度（提交各种产品的时间）
  - 达到目标和完成指派的任务（为每人分配的任务必须形成文档，确定截止期限和完成目标）
  - 控制预算（购买测试工具时）

评估测试人员测试工作的要点

- 行业知识
  - 是否深入了解系统的功能
  - 是否具备应用程序的行业领域的知识
- 技术知识
  - 熟悉各种测试技术？
  - 熟悉测试工具?
- 技术能力
  - 测试用例设计的如何？覆盖面、检错能力、检错深度
  - 开发的测试脚本、测试工具质量如何?
  - 能否用测试技术提高手头测试任务的测试效率?
- 交流能力
  - 是否听取了来自需求人员、开发人员和其他测试人员的反馈意见？
  - 撰写的缺陷报告质量如何？

缺陷相关的指标

- 发现缺陷的数量
- 发现缺陷在代码中的比例
- 发现缺陷的效率
- 发现缺陷的时间：主要缺陷是否发现太晚了？
- 是否遗漏了缺陷?
- 缺陷类型：是否发现重要的缺陷？
- 缺陷报告：是否标准化，是否易重现陷？

B. 评估测试组的有效性

- 定义角色和职责是否明确
  - 是否定义并文档化测试组成员的角色和职责
  - 是否将角色分配给了合适的人
  - 是否有细致的任务描述：任务的构成、技术方法、时间表、费用、每个人的时间分配、使用的标准和流程
- 是否具有充分的测试技巧、行业知识和经验
  - 行业知识：很重要
  - 技术知识：了解技术平台和系统架构，自动测试的基本编写、理解性能和安装之类的问题
  - 经验等级：初学者需要培训，可测试风险较低的部分。
  - 专业测试人员和行业专家测试人员应该互相协作。

成功测试组的10大因素

- **业务知识：**测试工程师应具备业务知识，并和用户紧密接触。
- **技术知识：**熟悉所测试的产品用到的技术，并掌握测试工具、方法等相关技术。
- **任务划分：**将业务任务和技术任务相互独立。
- **资源管理：**业务资源和技术资源相互结合。
- **与开发组的关系：**同开发人员协同工作。
- **生存周期早期介入：**测试应在开发周期的早期介入。
- **测试过程：**有成熟的测试过程管理规范。
- **灵活性/适应性：**能够适应不同的测试项目。
- **度量：**掌握度量的方法，以改进工作。
- **过程改进：**应致力于工作的不断改进。

C. 评估测试组测试活动质量

- 几个方面
  - 达到测试目标
  - 软件发布时系统无重大缺陷
  - 有一些缺陷在用户手册中的预防
  - 软件的测试工作保持了进度
  - 保持了预算