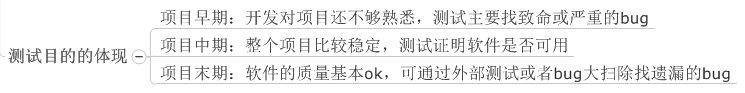
测试基础

1. 软件生命周期(掌握)
2. 计划:项目经理编写项目计划书,计划书内容3W(who/when/what)多少人花多少时间做什么事
3. 需求分析:需求分析工程师/产品人员,与客户反复沟通,将客户的需求转化成专业的表达,需求规格说明书(SRS)
4. 设计:系统架构工程师,设计软件的基本框架,概要设计说明书(HLD)
5. 编码:开发工程师,设计实现软件的思路,详细设计说明书(LLD),编写代码
6. 测试:测试工程师,检验实际结果与预期结果是否一致,需求是否有遗漏
7. 维护:技术支持工程师,版本更新
8. 软件测试的概念(掌握)

找出软件的毛病,提升软件质量,加深对被测软件的认识(优点,问题,瓶颈)

1. 软件测试的目的(了解)

证明软件可用,找bug,预防bug



1. 缺陷bug

缺陷不仅仅指功能问题(界面,用户体验,颜色,响应时间...)

错误通常是人为导致的,错误在一定场景下触发成为缺陷,缺陷可能引发故障,故障处理不好可能引发失效

1. 研发组织(大公司职位分明,小公司身兼数职)
2. 常用模型
3. 瀑布模型(适用需求全面且变更极少的情形,OA系统)(掌握)

* 特点:将软件生命周期各个阶段按照固定顺序从上到下依次执行
* 优点:应用广泛,流程清晰,易理解
* 缺点:测试介入较晚,前期的问题到后期才能发现,文档较多,项目后期才能看到研发成果,研发风险较大

1. V模型(适用需求变更不频繁的情形)

* 特点:瀑布模型的变种,将测试划分为多个阶段,并与相应开发阶段对应
* 优点:左边是开发活动,右边是测试活动,强调了开发和测试同等重要的思想
* 缺点:对测试介入时间指代不明确

1. 双V模型(掌握)

* 特点:一个V代表开发活动,一个V代表测试活动,开发和测试并行,测试设计和测试执行分离,先设计,再执行,测试设计和测试执行顺序相反
* 优点:测试介入较早,有利于发现和预防bug
* 缺点:测试设计对开发活动的依赖性较强

1. 螺旋模型(适合大型项目或软件的第一个版本)

* 特点:将项目分阶段实现,每个阶段可以看做是一个瀑布模型,增加了备选方案和风险分析
* 优点:增加了风险识别,用户参与度高
* 缺点:周期长

1. 敏捷模型(适合互联网项目)

* 特点:将需求划分成小需求,每个小需求由相应的团队去完成,开发效率高,重沟通,轻文档
* 优点:能够快速开发出软件版本
* 缺点:对团队要求高,不适合不成熟的团队

测试过程

1. 软件测试的四个阶段
2. 单元测试(一般由开发人员完成)

* 检查代码是否把详细设计说明书中的功能都实现了
* 检查点:函数功能,内部逻辑

1. 集成测试(一般由开发人员完成)

* 检查代码是否把概要设计说明书都实现了
* 检查点:内部接口,集成后的功能

1. 系统测试(由测试人员完成)

* 检查代码是否把需求规格说明书都实现了
* 检查点:需求中功能特性

1. 验收测试

* 检查代码是否把用户需求都实现了
* 第三方验收(正式验收):花钱请第三方专业团队做验收测试
* 用户验收(非正式验收):阿尔法α测试(内部人员在受控的环境[开发环境]下测试,测试结果可以得到立即反馈)贝塔β测试(用户在不受控环境[公网]下测试,测试结果不能得到立即反馈)

单元测试/集成测试/系统测试的关联与区别:

执行顺序:单元测试>集成测试>系统测试>验收测试

测试角度不一样:单元测试是针对最小单位函数来测试的,集成测试是针对集成后的模块功能以及模块与模块之间的接口来测试的,系统测试是针对系统功能来测试的

每个阶段测试方法不一样:单元测试以白盒测试方法为主,集成测试以灰盒测试方法为主,系统测试以黑盒测试方法为主

1. 回归测试
2. 回归测试的概念:
3. 检验开发是否把bug修改正确的过程,四个阶段都有回归测试
4. 检验版本的旧功能
5. 回归测试的目的:
6. 验证开发人员是否成功修复了bug
7. 验证开发人员修复bug后没有引入新的bug
8. 回归测试方案:
9. 完全回归

* 概念:测试版本的所有新旧功能
* 优点:覆盖全面
* 缺点:随着需求的增多,耗费的人力/时间越来越多,成本高

1. 选择性回归

* 概念:选择一部分功能进行测试
* 优点:节省成本
* 缺点:存在漏测的风险
* 选择哪些功能:已发现的bug,本版本的新需求,用户频繁使用的功能,主要业务流程,过去版本中经常出现问题的模块

1. 软件测试的四个活动
2. 测试计划

* 主要参与人员:测试经理
* 编写文档:测试计划书
* 文档内容:3W人员分工,时间安排,测试范围
* 参照文档:项目计划书,需求规格说明书(SRS)

1. 测试设计

* 主要参与人员:高级或资深测试工程师
* 编写文档:测试方案文档
* 文档内容:规划测试工作怎么做
* 参照文档:测试计划,需求规格说明书(SRS)

1. 测试实施

* 主要参与人员:普通测试工程师
* 编写文档:测试用例
* 文档内容:描述在什么条件下使用什么数据做什么操作达到什么预期结果
* 参照文档:测试计划,需求规格说明书(SRS),测试方案

1. 测试执行
2. 搭建测试环境

* 测试环境是指被测软件运行所需要的环境,通常是指一些软件包括操作系统的安装
* 有些公司是由专门的IT部门或运维人员配置
* 有些公司是由开发人员配置
* 若要测试人员配置,通常会提供文档,遇到问题,请教前辈

1. 执行测试用例:按照测试用例文档操作软件,检查测试结果是否与预期结果一致,一致则测试通过,不一致则需要提交bug,然后跟进bug解决.
2. 提交缺陷报告:将bug列表通过邮件发送给测试经理,开发经理,项目经理及其他相关人员,项目经理据此把控项目进度及节奏
3. 提交测试日报:测试工程师发邮件给相关人员,描述当天的工作内容,通常包括,执行了多少测试用例,发现了多少bug,遇到的问题,存在的风险,何时能测试完毕
4. 提交测试报告:可能分阶段提交,也可能是项目测试完毕只提交一次;通常由测试经理编辑邮件,内容包括使用了多少人力,花了多长时间测试,写了多少测试用例,执行了多少测试用例,发现了多少bug,解决了多少bug,有多少bug未解决,bug未解决的原因(最终遗留的bug一般是由于当前技术手段无法解决,是和开发经理确认过的),版本是否测试完毕或是否可发布的结论.项目经理据此判断当前版本是否可发布.
5. 项目文档
6. 需求规格说明书(SRS)

* 编写者:需求分析工程师
* 项目阶段:需求分析阶段
* 内容:描述软件的功能(运行环境/使用用户/开发语言/功能特性/非功能特性/界面...)

1. 概要设计说明书(HLD)

* 编写者:系统架构工程师
* 项目阶段:设计阶段
* 内容:描述软件有哪些模块构成,模块之间的关系,有哪些函数,函数之间的关系

1. 详细设计说明书(LLD)

* 编写者:开发工程师
* 项目阶段:编码阶段
* 内容:描述函数的内部逻辑

1. 代码

* 编写者:开发工程师
* 项目阶段:编码阶段
* 内容:开发工程师将伪码(为函数设计思路打草稿)转变成可以运行的代码