05测试与测试点与测试用例

测试：探索程序可能存在的错误（使用测试用例）

测试点：一切需要测试的功能或接口----测什么

测试用例：为了测试在特定场景下的功能的一组操作和预期---怎么做

测试步骤：寻找测试点--形成测试用例--执行测试用例--迭代测试用例

测试点

模块拆分　　单元细化　　逻辑补充

测试拆分

模块拆分：整体系统可以拆分成多个组成部分

属性拆分：将一个独立的单位拆分成它所具有的属性

流程拆分：按照流程拆分

测试点细化

需要细化到直接指导测试用例编写

方法：等价类，边界值分析

等价类

输入的条件集合是对于验证功能是等价的

有效等价类：对程序有效的输入

无效等价类：对程序无效的输入

边界值分析

某些程序输入是有规定输入范围 ，在边界条件处着重分析

测试点补充

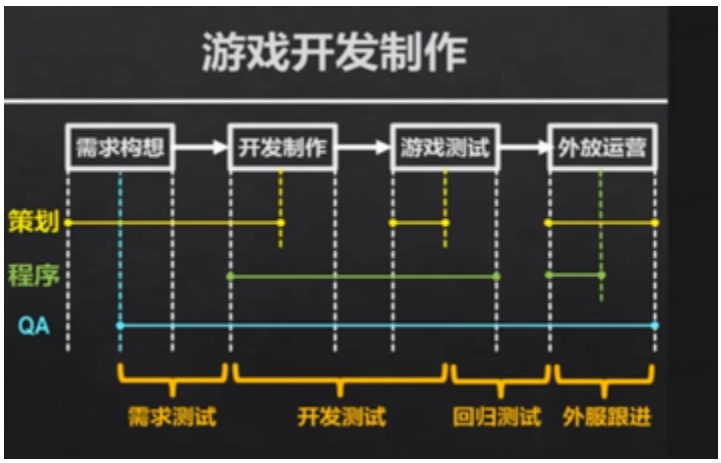
方法：组合测试，换位测试，破坏性测试（主动行为：内存修改，代码注入；被动行为：顶号，断网，延迟）

测试用例

应该包含：编号（标记），测试内容（测什么），执行步骤（如何操作），预期结果（预期结果），执行结果（实际结果）

> 😀 测试用例需要规范：
>
> 表述清晰，其他人能够按照步骤执行
>
> 步骤明确，结果唯一
>
> 充分覆盖所有测试用例

06游戏测试流程与执行



开发测试

测试环境

客户端，服务端，权限&指令

客户端：内服客户端（资源--手动覆盖，脚本---下载补丁，引擎---重装），外服客户端（向外）

服务器：内部测试服务器（环境不同，被指标，DNS，内网），外部服务器

指令（可自己写指令）：内服，运维

测试对象

执行测试

😀 tips：

可以改变某些测试用例顺序，来减少指令使用

多个用例一起执行

使用测试工具

简化复杂指令

编写自动化脚本

明确策划需求和程序实现

了解程序实现

BUG

侠义：只包含软件存在的漏洞

广义：还包含用户，测试等发现可改进细节，需求文档存在差异的功能实现

发现bug后----------确定bug并重现

提单：一个bug对应一个单，标题与内容一致

bug修复完成后需要更新文档

回归测试

BUG需要在文档更新后在完整测试一边-------回归测试

外服测试

玩游戏，评估数据，跟进BUG，压力测试，事故报告

07易协作

易协作基于任务和交付物的讨论，聚焦性更强，不同于即使通讯的私聊

易协作群可以方便的发起项目协作

团队成员都可以查询到所有讨论的历史内容，不存在信息孤岛失真

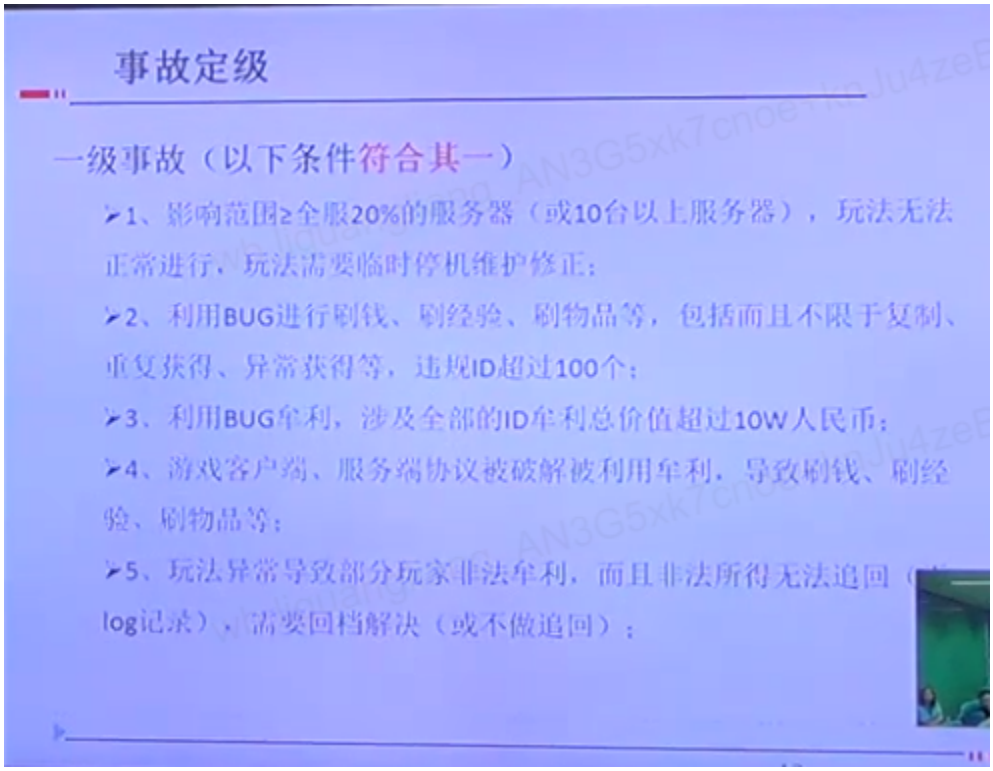
可以建单，杜绝事后忘记

10手游事故库

定义

事故---外放的BUG演变而成的，取决于BUG的扩散成都以及BUG的影响范围

定级：



手游事故

手机对手游影响特性-----分辨率，CPU，内存，OS版本

手游测试关注点：兼容性测试，设备环境&异常处理

BUG情况：

规则缺失：尽管程序没有问题，但在某些内容上缺少限制或规定

规则冲突：实际情况与文档内容冲突

😀 注意点：

明确规则，文档编写的时候应该明确所有规则

熟悉游戏，多玩自己负责的游戏，了解哪些地方可能出问题，熟悉游戏默认规则

加强沟通

重视玩家

判定缺失：判定规则对某些判定条件不够全面

判定冲突：服务器和客户端之间的判定规则不同

😀 注意点：

所有信息服务器都应该校验

测试覆盖全面

log

流程管理：内容修改没有通知，程序修改为通知

😀 注意点：

监控重要信息

外放前确认所有修改内容

版本管理：游戏版本众多，回归版本众多，没有模拟外服环境

😀 注意点：

加强版本管理

外放内容至少在最近本版本进行测试

对于分渠道玩法，关注对其他版本的影响

建立patch回归流程，patch外放前预跑

事故处理

发现，处理，重现，修复BUG

17Linux使用技巧

18SVN+git

SVN：一个开源的版本控制系统。它可以帮助用户管理和跟踪项目中的文件和目录的历史变更记录

git：开源的分布式版本控制系统，被广泛使用于小型到大型项目的源代码管理

作用：

增强CTRL+S：保存每一个提交节点的版本

增强CTRL+Z：回到上次提交状态，回退到某次之前的修改

增强CTRL+C/V：分支（branch）别人的产出，或者合并内容

版本库：

本地版本库：SVN，GIT

第三方托管

私有版本库

客户端：

tortoiseSVN：直接与版本库交流

tortoiseGIT：先安装git，通过git与版本库通信