百度外卖

测试反模式

百度外卖 艾辉

百度外卖

背景

百度外卖

引言

什么是**测试反模式**？

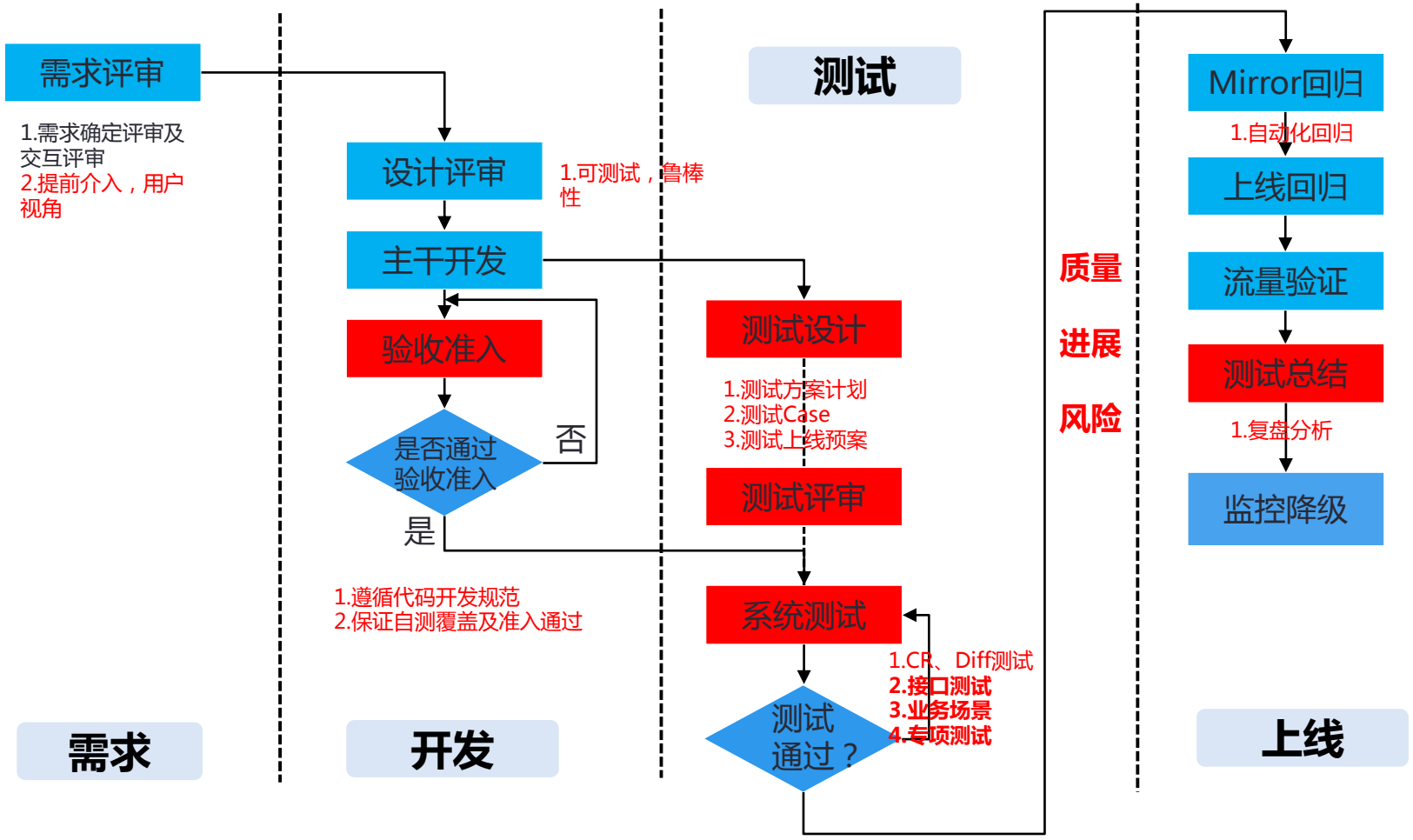
- 与我们所说的模式，或者正模式不同，反模式告诉你：“这么搞事情，一定会把事情搞砸”。

- 测试反模式：测试过程中常见的错误模式，典型的问题案例。

百度外卖

目录

研发测试流程


需求评审
1.需求确定评审及交互评审
2.提前介入，用户视角
设计评审
1.可测试，鲁棒性
主干开发
验收准入
是否通过验收准入
否
是
1.遵循代码开发规范
2.保证自测覆盖及准入通过
测试
测试设计
1.测试方案计划
2.测试Case
3.测试上线预案
测试评审
系统测试
测试通过？
1.CR、Diff测试
2.接口测试
3.业务场景
4.专项测试
质量
进展
风险
Mirror回归
1.自动化回归
上线回归
流量验证
测试总结
1.复盘分析
监控降级
需求
开发
上线

目录

研发反模式

线上运维

编码细节

架构设计

百度外卖

研发反模式-架构设计

■ 模块设计

× 模块臃肿、功能过多；自带大cache模块；自带大字典模块；主次不分模块；定时重启模块；“自定义”接口的模块；……

■ 模块交互

× 批量请求循环调用；对次要服务强依赖；完全信任下游返回的数据；不合理的均衡、重试算法；不合理的超时重试；……

■ 系统一致性

× 忽视容错处理导致的一致性问题；忽视运维导致的一致性问题；忽视业务升级导致的一致性问题；忽视不同业务逻辑耦合导致的一致性问题；忽视多任务导致的一致性问题；忽视模块交互协议导致的一致性问题；……

■ 对外依赖

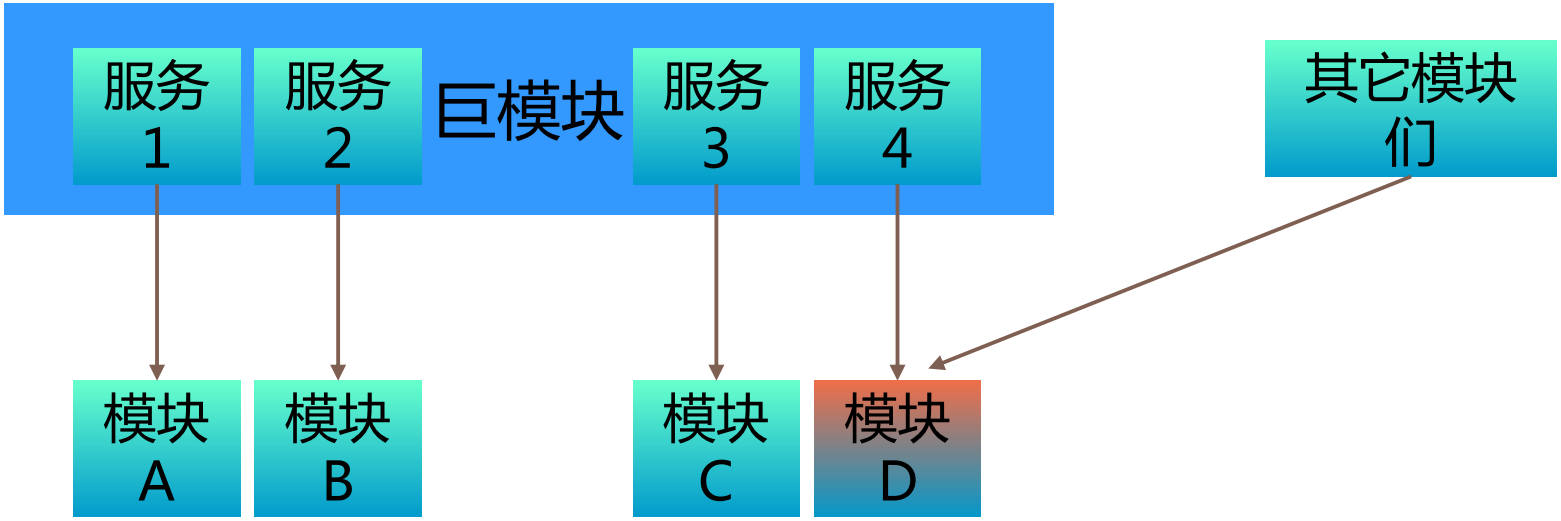
× 爆发式的访问外部服务；不以标准的方式使用外部服务；忽视外部服务的时效性；

■ 对外服务

× 把样例当接口规范；对外接口缺少请求来源字段；……

百度外卖

架构设计-模块臃肿，功能过多

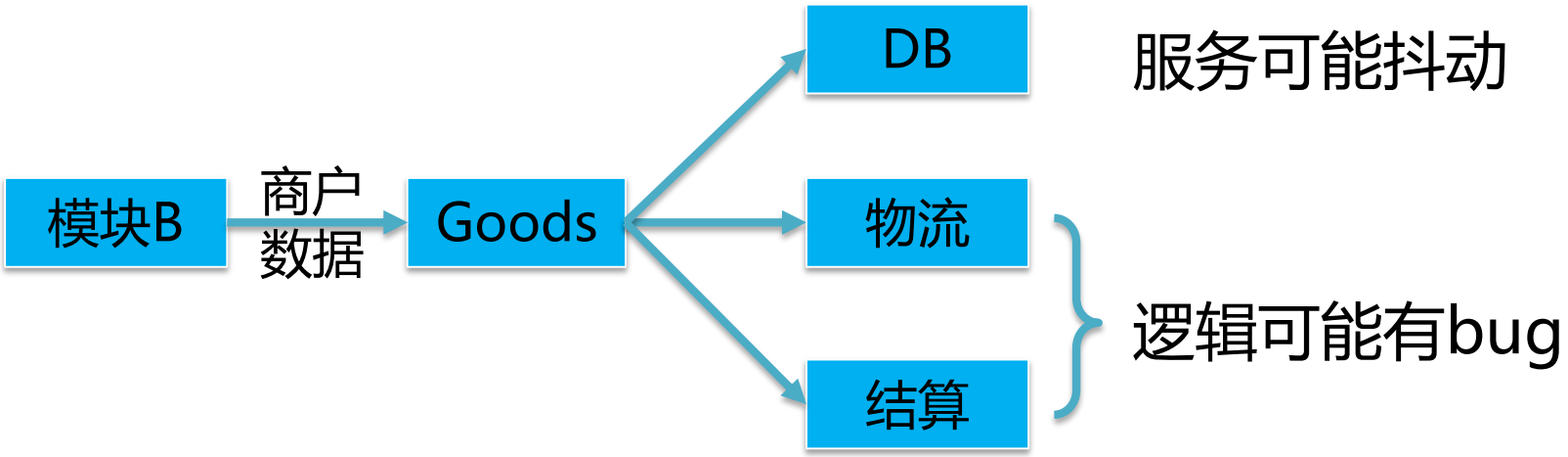


单个模块功能过多，上线频繁易出事故
鸡蛋放在同一个篮子里，拖垮整个系统

模块拆分
独立部署

百度外卖

架构设计-完全信任下游返回的数据



Goods容易出现数据不一致，
大部分都因为对下游请求没有异常处理

百度外卖

研发反模式-编码细节(设计)

1. **初始化建立过多链接，而不是需要使用时分配，造成资源浪费**
2. **继承过于臃肿的基类，而不是组合**
3. **不区分核心/非核心服务**
4. **接口升级不考虑兼容性(返回值)**
5. **异常处理**

   ×不检查其他模块传来的数据；不注意检查函数返回值；不做异常处理

百度外卖

研发反模式-编码细节(基础组件-Mysql)

1. sql不合理导致慢查询

X 没有索引/没有用到合理索引

X 查询结果集过大

- 导数据、查全库，不带where条件，或区分度较低

X 瞬时大量更新操作

- delete，update

X 复杂业务下的复杂sql

- like/join/group by，前后台功能公用存储

百度外卖

研发反模式-编码细节(基础组件-Mysql)

2. 没有正确使用事务

X 没事务

- 数据要求一致的场景没有事务或重试
- 容易引起数据不一致

X 长事务

- 稳定性风险，降低性能

X 事务挂起

- 事务未提交、也没有回滚

3. 逻辑不严谨导致安全漏洞

X 手动拼sql

- 无转义

X 请求参数无校验

- 分页参数没有intval强转
- 请求参数没有类型校验

百度外卖

研发反模式-编码细节(基础组件-Redis)

X key管理混乱

- 无集中管理，不清楚业务都在使用哪些key，迁移的痛
- key前缀起名随意

X 大key

- 单key内存占用无法收敛，链表、集合等
  例：消息去重、UGC Tag筛选

X 死key

- 业务不使用也不删除，无过期时间

X 业务混部

- 例：c_default

X redis批量操作随意

- 万级别mset
- 批量delete
- delete一个set集合

百度外卖

研发反模式-编码细节(基础组件-MQ)

X 下游处理超时

- 一次接收数据量过大
- 单命令点业务逻辑过于复杂
- 下游依赖服务异常，超过mq超时

X 业务自身抛异常

- 该throw excetion还是return？

X 瞬时写入

- 大量消息积压泄洪，瞬间冲垮DB

X 下游消费不幂等

X module 拆分不合理

X 配置不合理（超时、flag、窗口和线程、延迟）

百度外卖

编码设计-初始化过多链接/对象，资源浪费

X 构造函数做太多事情

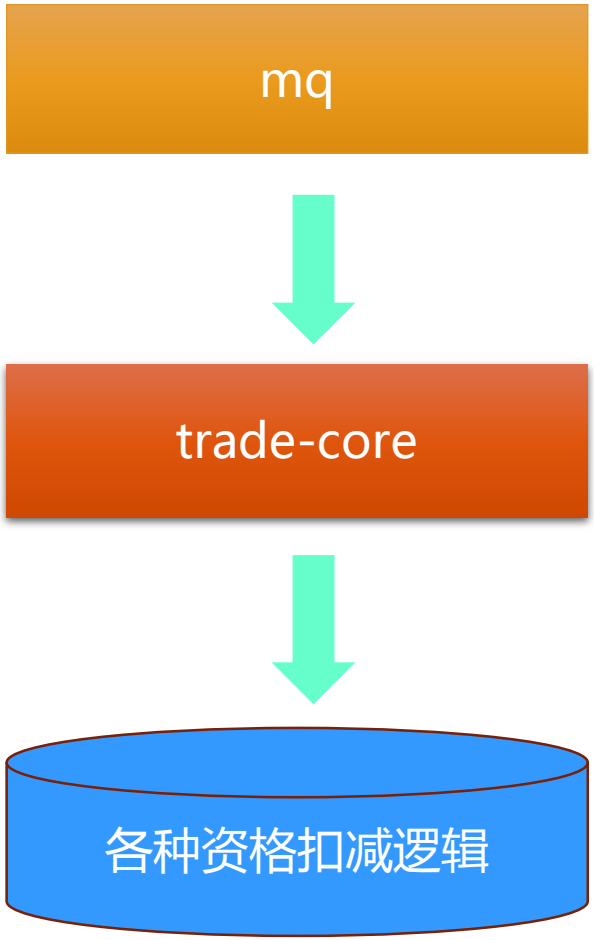
- db空链接
- redis空链接
- 默认开启事务/默认请求ps
- new失败阻塞服务（例：redis迁移）

```
35    public function __construct() {
36        $this->[redacted]Shop = new S[redacted]Proxy();
37        $this->[redacted]Order = new [redacted]Order();
38        $this->[redacted]ng = new [redacted]ting();
39        $this->[redacted]Pay = new [redacted]ay();
40        $this->[redacted]unt = new [redacted]nt();
41        $this->[redacted]ist = new [redacted]t();
42        $this->[redacted]ceDataLogistics = new [redacted]stics();
43        $this->[redacted]er = new [redacted]();
44        $this->[redacted]am = new [redacted]spam();
45        $this->[redacted]Cost = new [redacted]st();
46        $this->[redacted]roc = new [redacted]lloc();
47        $this->[redacted]y = new [redacted]y();
48    }
49
```

[redacted]	0.9	9	http	200	GET	gzhxy	无	
[redacted]	0.15	12	http	200	GET	gzhxy	无	
[redacted]	0.15.1	0.23	mysql		connect	gzhxy	冗	
[redacted]	0.15.2	0.071	redis		connect	gzhxy	冗	
[redacted]	0.15.3	39.13	http	200	GET	gzhxy	无	
[redacted]	0.15.3.1	0.236	mysql		connect	gzhxy	#0	
[redacted]	0.15.3.2	0.345	mysql		query	gzhxy	#0	

百度外卖

编码设计-下游消费不幂等



下游处理消息可能遇到各种异常，
重发消息是常态

业务处理缺乏去重逻辑

没有可重入就会重复处理消息，
极端情况可能导致系统雪崩
例：批量push，msg，注券等

百度外卖

研发反模式-线上运维

1. 重要次要模块，服务混部

2. 基础服务跨机房交互

3. 机器冗余度不够

4. 不及时清理冗余数据

5. 线上环境

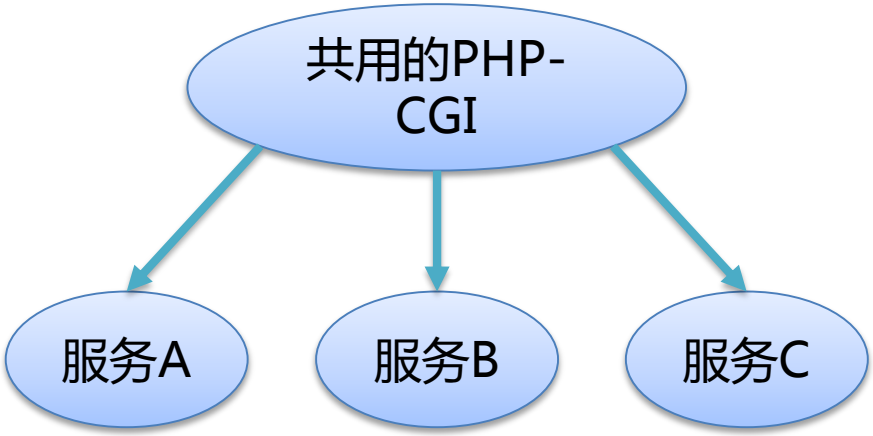
X 没有防攻击预案；重要数据无权限控制；重要数据无备份；不清理过期数据；线上执行大IO类任务

6. 线上监控

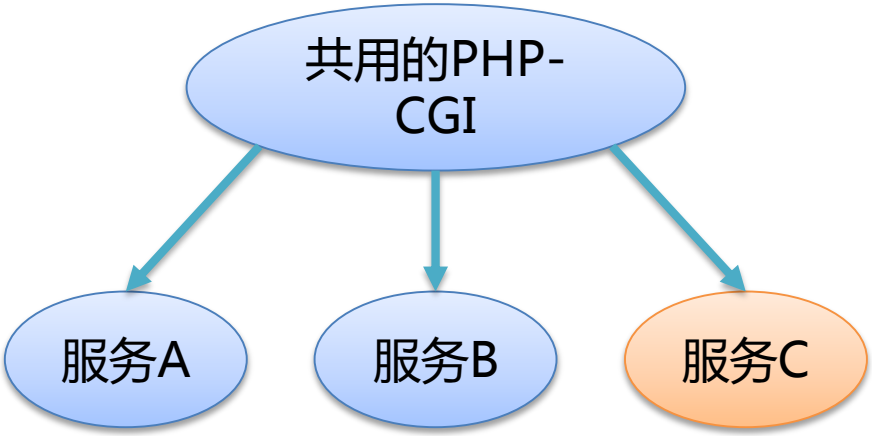
X 缺少机器监控；不监控模块日志；不监控产品线核心指标

百度外卖

线上运维-重要次要模块，服务混部



正常的时候，大家和谐共处
你好我好大家好



服务C异常，占完了PHP-CGI
于是整个服务都不好了

百度外卖

线上运维-基础服务跨机房交互

模块名	spanid	类型	状态	服务/方法	远程ip	大小	时间轴(ms)
shopui	0		200				125.53
	0.1	redis-redis	0				0.74
session	0.2	nshead-mcpack2	0	session/getData	:9081	req:340,res:542	70.56
	0.3	redis-redis	0				0.1
	0.4	redis-redis	0				0.19
	0.5	redis-redis	0				0.84
	0.6	redis-redis	0				1.02
	0.7	redis-redis	0				0.84
	0.8	redis-redis	0				0.9
session	0.9	nshead-mcpack2	0	session/getData		req:340,res:542	1.74
	0.10	mysql-mysql	0				1.45

跨机房调用耗时明显耗时高，并且稳定性也差很多
导致系统不稳定，影响数据一致性

百度外卖

目录

测试反模式

不标准 环境	不严格 准入	不关注 异常	不关注 日志
不测试上 线&回滚	不考虑 性能	不考虑 安全	不约束不边界 联调测试
上线和测试文 件不一致	不考虑 线上线下隔离	不考虑 上线后流量	不关注 监控告警
.....			

百度外卖

测试反模式：不标准环境

案例1

rd提测，qa搭建环境，[illegible]的机器上会较独立、好区分，且按自己的[illegible]这样部署环境存在什么问题？

原则：按照上线步骤进行环境搭建，若条件不允许，尽量保持等价。

案例2

故障：频道页513运[illegible]券已抢光，无法正常领取。

原因：线下测试环境搭建时未及时同步C端最新上线的代码，未测试覆盖及时发现问题。

原则：环境&依赖服务和线上保持一致。

百度外卖

测试反模式：环境搭建的错误与解决

常见错误

- 不按上线步骤搭建。
- 未同步最新代码和数据库。
- 绝对路径不一样。
- 基础依赖不一样。
- 不注意与线上环境的隔离，影响线上服务。

环境搭建原则：
1.环境一致；2.按照线上步骤搭建。

解决思路

- ✓ 服务器环境一致（操作系统等）。
- ✓ 基础服务版本一致（基础库）。
- ✓ 同步线上最新代码和数据库结构。
- ✓ 按照上线步骤搭建。
- ✓ 本地化配置。
- ✓ 注意线上线下环境隔离。
- ✓ 注意文件的权限一致。

百度外卖

测试反模式：不严格准入

案例

环境搭建完后，qa着手准入验证，发现其中某些重要功能验证不通过，联想到之前rd自测被准入不通过时不高兴，qa犹豫是否让提测项目通过。

与rd沟通，说明问题及影响，与rd达成一致，并按流程发出准入报告结论。

常见问题

qa不好意思驳回；未和rd沟通确认就直接打回。

解决建议

流程是无数经验的总结，尽量遵守；准入case与rd达成共识；及时沟通、提升效率。

百度外卖

测试反模式：不关注异常

血的教训

- 2014年6月，由于支付生[illegible]，某支付渠道通过建行支付生[illegible]断为支付成功，持续3天，引起[illegible]重事故，RD和QA绩效降级和罚款[illegible]辞退。

支付业务关系用户和产品利益，异常测试很重要。

- b端nmq堵塞影响商户发布菜品，原因是url的某个字段输入异常时候返回异常。

- .....

百度外卖

测试反模式：异常测试错误与解决

常见错误

- 不做异常测试，只关注正常逻辑。
- 不关注依赖服务异常，包括：依赖服务不可用、依赖系统响应慢、依赖系统数据异常等。
- 未处理依赖函数所有返回值。
- 不考虑业务实际情况。

解决建议

- 梳理清楚需求（流程图），深入了解异常逻辑。
- 用流程图梳理代码，异常点是否考虑。
- 用时序图梳理多模块交互，看异常处理是否包含。
- 用状态转移图梳理状态变更，遵循有的状态不可逆原则。
- 结合积累线上问题或经验判断。

百度外卖

测试反模式：异常测试方法

异常测试方法

- 输入异常测试（必填项缺少，输入长度、参数范围越界、参数类型错误、参数个数错误、特殊参数、异常的操作顺序等）。
- 模块系统测试：系统中某个模块出现异常影响系统整体服务，包含：慢节点、模块异常处理，交互异常。
- 依赖服务异常：依赖服务不可用、响应慢、依赖服务数据异常等。
- 操作系统异常：文件系统异常、进程异常。
- 硬件异常：网络延时、抖动，磁盘IO错误，CPU打满，内存泄漏等。

百度外卖

测试反模式：不关注日志

案例

- 项目测试期间，qa在测试到某个异常，无任何异常日志信息。
- 某业务线线上出现问题，rd定位问题，发现日志中有用信息甚少，需要重新加日日志定位问题。（那偶发问题怎么办？）

不打印日志或者日志混乱，出现问题，定位花很久时间，耗时耗力。

测试建议

- 日志是程序质量的一部分；需考虑日志级别、标准和规范。
- 日志对于定位问题可起到事半功倍的效果，良好的日志规范可快速的辅助问题定位、日志监控。
- 日志是qa测试的一部分，如测试出现异常，没有打印日志或者打印日志不符合日志规范，这属于bug。

百度外卖

测试反模式：不测试上线&回滚

案例

案例1：某业务，上线前和上线后的状态分别是状态(1)和(3)，这两个是ok的，(2)是上线过程中的状态，这个时候模块v1 服务是不正常的，由于各种原因，状态(2)停留的时候可能会比较长， 影响线上服务，造成回滚。

案例2：某业务线rd上线，没准备回滚方案，结果上线出现异常，按照经验回滚代码，不能及时回滚，导致线上服务2小时存在问题，影响20w用户。

测试建议

- 上线步骤和回滚方案是qa测试的重点，qa严格按照上线步骤部署测试环境。
- 记录测试过程中的注意事项，关注线上线下的差异性。
- 关注发布过程中的服务及业务兼容性。
- 各个步骤有明确的操作时间和负责人，及对应的回滚方案。

百度外卖

测试反模式：不考虑性能

案例

故障：2016年7月23日中午11点05分开始，商户营销活动数据展示异常。

原因：11点运营吃货节发用户push，发送范围200W用户。11点01分PV陡增，11点05到高峰，活动页面qps高达6000qps，页面对营销活动商户请求有4个，对营销服务4倍压力，营销活动处理逻辑存在db空连接，导致db连接数打满，出现服务异常。

测试建议

- 新开发的系统，需做性能测试摸底，给出性能评估。
- 核心服务接口，需定期做性能测试，给出服务承载和容量规划。
- 系统重构项目，需有针对的压测评估。
- 性能调优项目，需要给出性能调优前后的性能分析对比。
- 营销运营活动类项目，合理评估流量压力，对依赖核心服务做压测评估。

百度外卖

测试反模式：不约束不边界联调

常见问题

排期不match，互相等待；未界定边界、有遗漏Case；环境问题多等。

测试建议

- 测试计划：注重排期，尤其是联调时间点，不要出现等待联调。
- 联调case：双方review，查漏补缺。
- 联调环境：集约部署，集中解决。
- 联调过程：主要业务方QA主动Owner，及时通报问题，推动解决，注重可见性。
- 下游约束上游！

百度外卖

测试反模式：上线和测试文件不一致

案例

- 某服务上线到mirror机到发现有php fatal，回滚 现问题发现上线文件和测试文件不一致。
- qa测试完成，pm有需求变更，只同步了 上线后出问题。

原则：上线文件=测试文件

常见错误

- 未按照上线步骤上线。
- 上线文件和测试文件分支不一样。
- 上线配置是线下配置。

解决方案

- 上线文件一定是最后测试完成的文件，配置文件要仔细check。
- 要求rd提前及时合代码，合完后再做回归。

百度外卖

测试反模式：不考虑线上线下隔离

案例

- 某qa同学在做push测试过程中，对线上与线下环境配置错误，导致500W用户收到“test测试”推送，负面影响极大，定性为严重事故，接口QA被开除。

测试建议

- 线上环境隔离，如：白名单。
- 线上测试，也必须基于测试数据。
- 环境搭建和线下测试，一定做配置本地化。如果一定要用线上配置或数据，走流程按规定取数据，取线上数据会对线上机造成风险，合理预判。
- qa对线上服务和数据要有敬畏之心。

百度外卖

目录

百度外卖

QA六大意识

- 质量意识
- 产品意识
- 推动&沟通意识
- 团队意识
- 时间意识
- 进取意识

百度外卖

QA目标要求

- 不会开发的QA，已是行业新时代的”文盲”
- 比开发更懂业务，比产品更懂技术

百度外卖

Thanks!

Q&A

百度外卖