谈谈灰度发布的设计与实现

定义

灰度发布时互联网产品发布常用的一种方式，顾名思义，就是在黑与白之前平滑过渡的一种产品发布方式。铲平发布者根据某种规则，让一部分用户继续使用原来的产品功能，另一部分用户开始组建启用新的功能，在过度的过程中可能还会对产品做进一步的完善，灰度发布完成后，所有用户都将使用新的产品功能。

- 金丝雀发布
- A/B Test

灰度发布目的

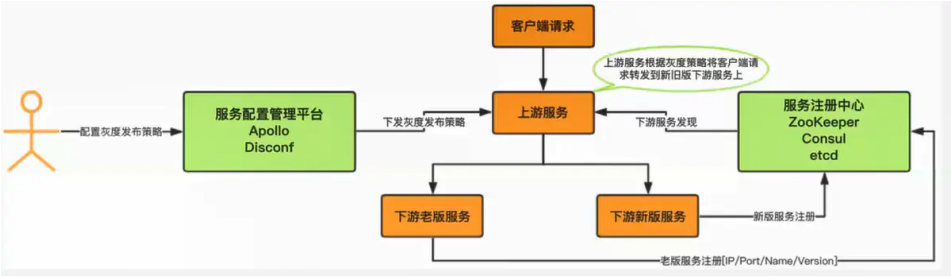
互联网产品需要快速迭代上线，又要保证质量，保证刚上线的吸引，一旦出现问题可以很快控制局面，就需要设计一套灰度发布系统。

作用

可以根据配置，将用户流量导到新的上线的系统上，来快速验证新的功能，而一旦出现问题，也可以马上的恢复，简单的说就是一套A/B test 系统

架构方案

- 上游服务介入客户端请求，根据下发的灰度配置将符合条件的请求转发到下游新旧版服务上。
- 下游新旧版服务是处理客户端请求的业务服务系统。
- 服务配置管理平台，此平台可以配置不同的灰度发布转发策略给上游服务。
- 注册中心，负责上下游服务的注册和发现。



image

难点 怎么去设计

协议设计

image

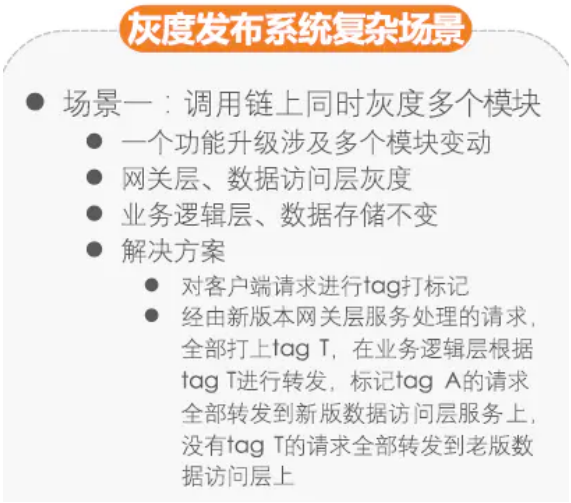
image

案例

- 上游服务
  - Nginx
    - 拓展Nginx实现灰度发布策略转发
    - 本地部署Agent
      - 接收服务配置管理平台下发的灰度策略，更新Nginx配置，优雅重启Nginx服务。
  - RPC 服务
    - 集成配置管理平台客户端SDK，接收服务配置管理平台下发的灰度策略。
    - 灰度策略执行计划。
      - 新版本的开启。
- 下游服务
  - 新版本服务注册到服务注册中心
    - 通过版本号控制是否开启

复杂场景

场景一



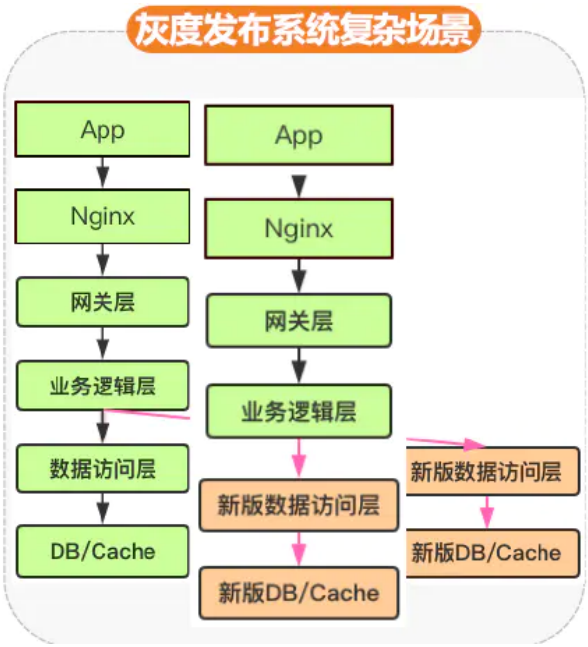
image

网关层改动，数据访问层也要改动。

image

场景二

image



image

灰度发布的链路

ABTest场景实施的层次

- 网关层(如果各个场景都使用，就选择网关层)
- 业务逻辑层
-

核心问题

```
本质是什么？ 如果只是一个具体的业务逻辑实现，那么在业务层做。如果是通用的，那么在网关层。
```

开源框架:

springCloud灰度发布神器

```
Nepxion Discovery
    服务注册和负载均衡的增强中间件
    包含了灰度发布
```

APP端如何发布

image