糗事百科技术架构及演进

周迁@糗百

qiushibaike.com

分享你身边发生的爆笑糗事

——糗事百科

一些数据

- 2000w注册用户，4000w非注册用户
  320w日启动用户，2000w日启动次数
  1亿动态请求/天，峰值30000/sec
  11台应用服务器(8核8G)[5台支撑web，6台支撑app]
  18G memcached
  6台mysql服务器(master4核4G，slave4核16G)

关于架构

- 没有最好的架构，只有最合适的架构
简单就是美(KISS)
不断改进和优化
可扩展很重要
强大的架构在处理增长时通过简单增加相同的东西(服务器)来应对，同时还能保证系统的正确性

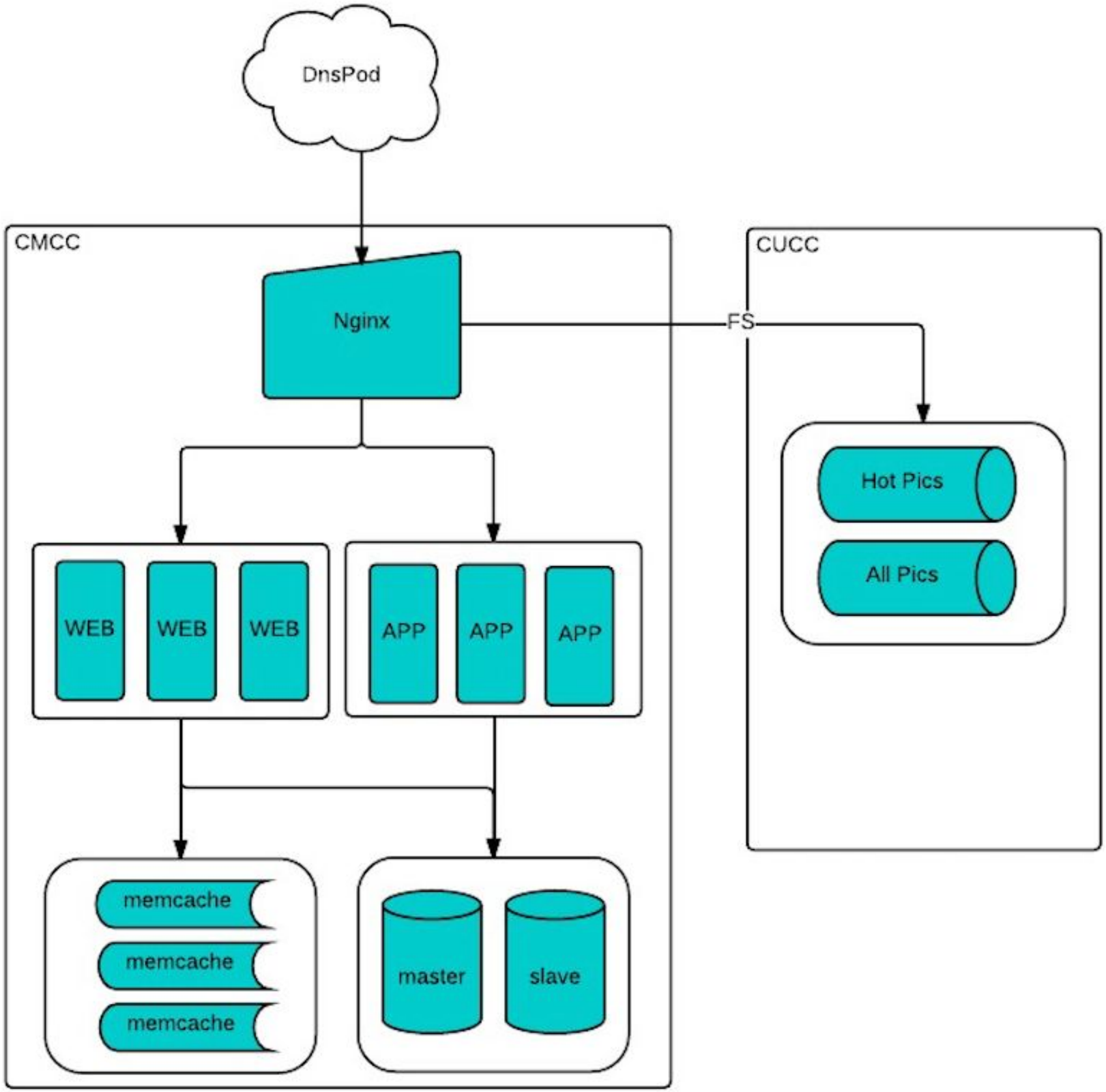
技术选择

- It will fail, keep it simple

工程师以一当十？服务器以一当十?

选择标准：成熟，简单，用的人多，良好的社区氛围，持续的优异性能，很少失败，开源，轻量级

nginx, tornado(python), memcached, redis, mysql



nginx

- 高并发连接，内存消耗少，稳定性高
缓存图片等静态文件
反向代理，快慢分离
所有的请求都通过80端口的nginx进程分发,动态内容走localhost上的torando进程

tornado

- 非阻塞式设计(epoll)，轻量级web框架
尽量少使用异步接口
多进程： 单服务器部署多个tornado进程
快慢分离
app，web服务独立

快慢分离

```
upstream appends{
        server 127.0.0.1:8716;
        server 127.0.0.1:8717;
        …
}
upstream appslowends{
        server 127.0.0.1:8714;
        …
}
…
location /
{
        proxy_set_header        Host $host;
        proxy_set_header        X-Real-IP $remote_addr;
        if ( $query_string ~* "slow" )
        {
                proxy_pass http://appslowends;
```

tornado

```
if cache :
    return self.write( cache )
elif self.request.uri.find("slow")<0 :
    arg = '?slow' if self.request.uri.find( '?' ) <= 0 else '&slow'
    logging.info("list acc Redirect 1")
    self.set_header("X-Accel-Redirect", self.request.uri + arg)
    return
…
```

Memcached

- 多服务器，多节点
  一致性哈希
  每个机房单独一组memcached节点

Mysql

- 读写分离
  双master，多slave
  跨机房同步：每个机房有主力slave，也有备份slave

tools

- dnspod　　- 网段、地区流量分配
  监控宝(等) - 常规服务,性能监控
  360网站宝 - CDN,攻击防御
  scribe　　- 日志收集

问题出现

- 帖子生命周期无法跟踪
  积压大量审核通过待投放的帖子
  部署事故频现，*cap*不再适用

解决方案

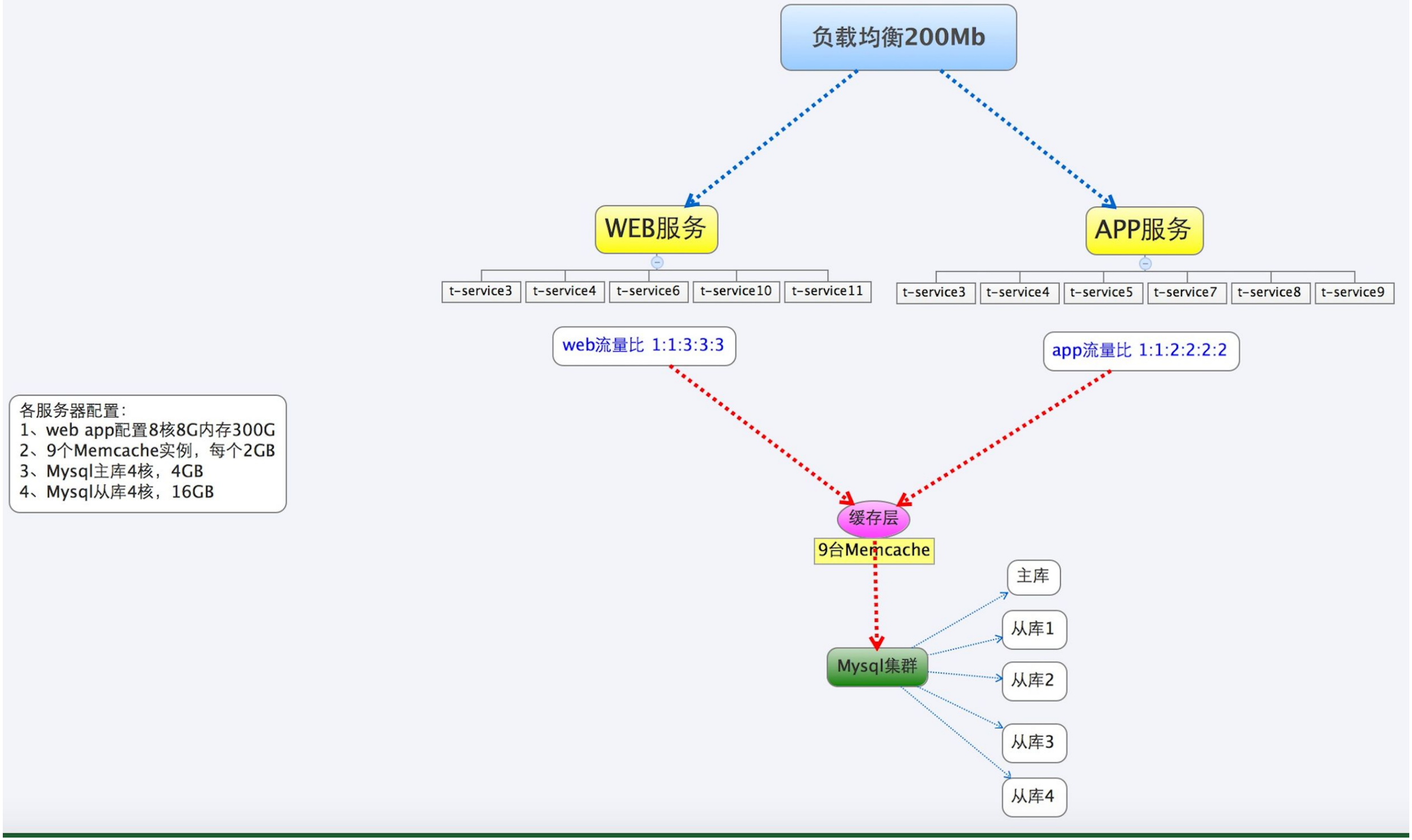
- 所有代码迁到gitlab，开发完善的部署系统(Qber)
开发帖子快照服务，完善审核投放回收的监控
开发LogServer，收集帖子操作相关数据
监控帖子生命周期
顺便解决错误日志收集告警，用户生命周期监控

问题出现

- 开始推送后服务器过载
  跨机房同步延时大
  图片中心磁盘io成为瓶颈

解决方案

使用商用CDN
nginx与tornado之间改为长连接
接入层增加频率限制和最大连接数限制(防雪崩)
线上服务迁到腾讯云
只维护一个memcache集群
网络连通性、数据库同步等问题得到解决
上线新服务器速度更快



问题出现

- 扫黄打非，图片风险大
平胸大赛导致审核流量上升10倍，老代码扩展性有问题，加机器不能解决问题
A/B Test需求(wap取消图文后流量暴跌)

解决方案

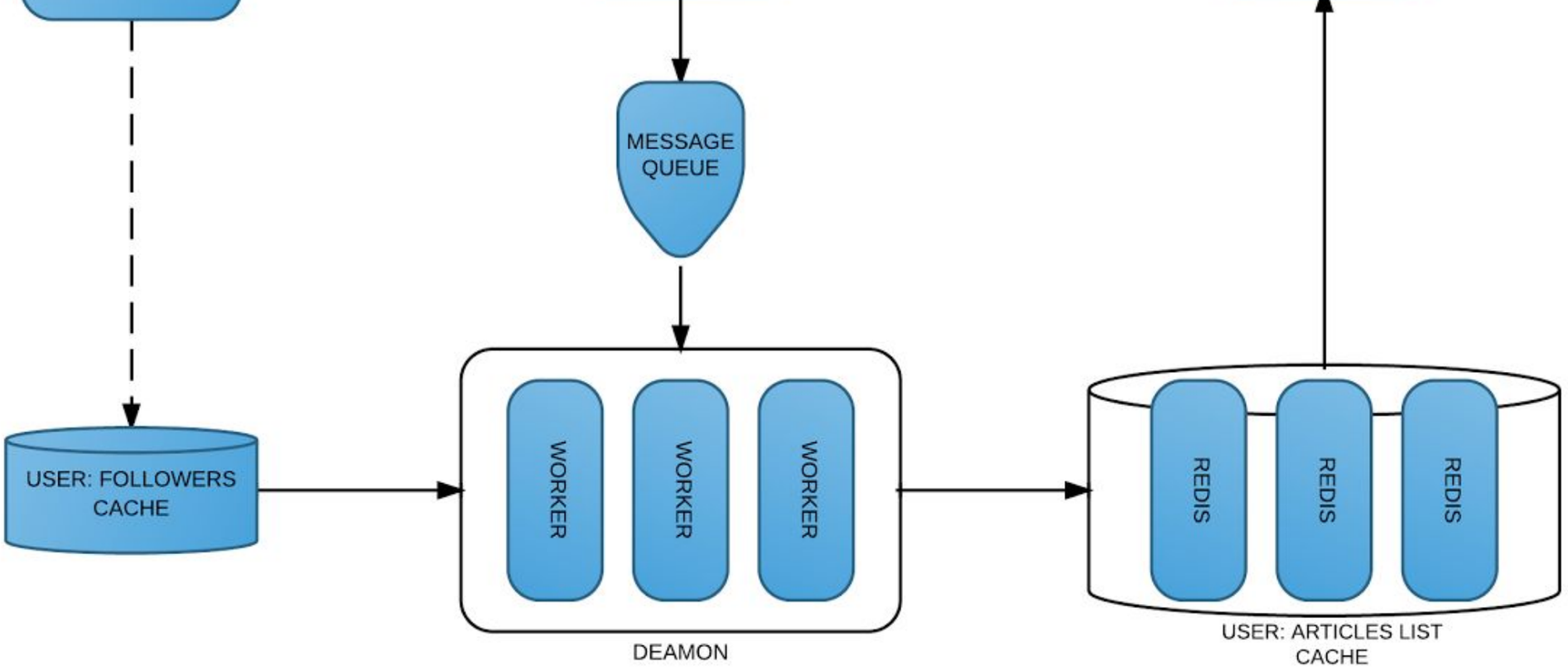
- 开发图片人工审核系统，图片用户审核前先人工审核
重新实现审核系统，增加审核举报功能，解决扩展性问题同时性能提升5倍
开发abtest-proxy

abtest-proxy

- golang实现
  放在负载均衡的下一层，由abtest-proxy转发请求
  abtest-proxy与backend之间通信使用http1.1，长连接
  支持配置文件热更新
  内置常用测试规则，特殊规则自行实现Rule接口
  支持实验用户和对照用户
  abtest-proxy将指定测试用户和对照用户的请求根据测试规则分别转到testServer和compareServer，其他请求根据加权轮询调度算法转到服务列表(serverList)的一台服务器

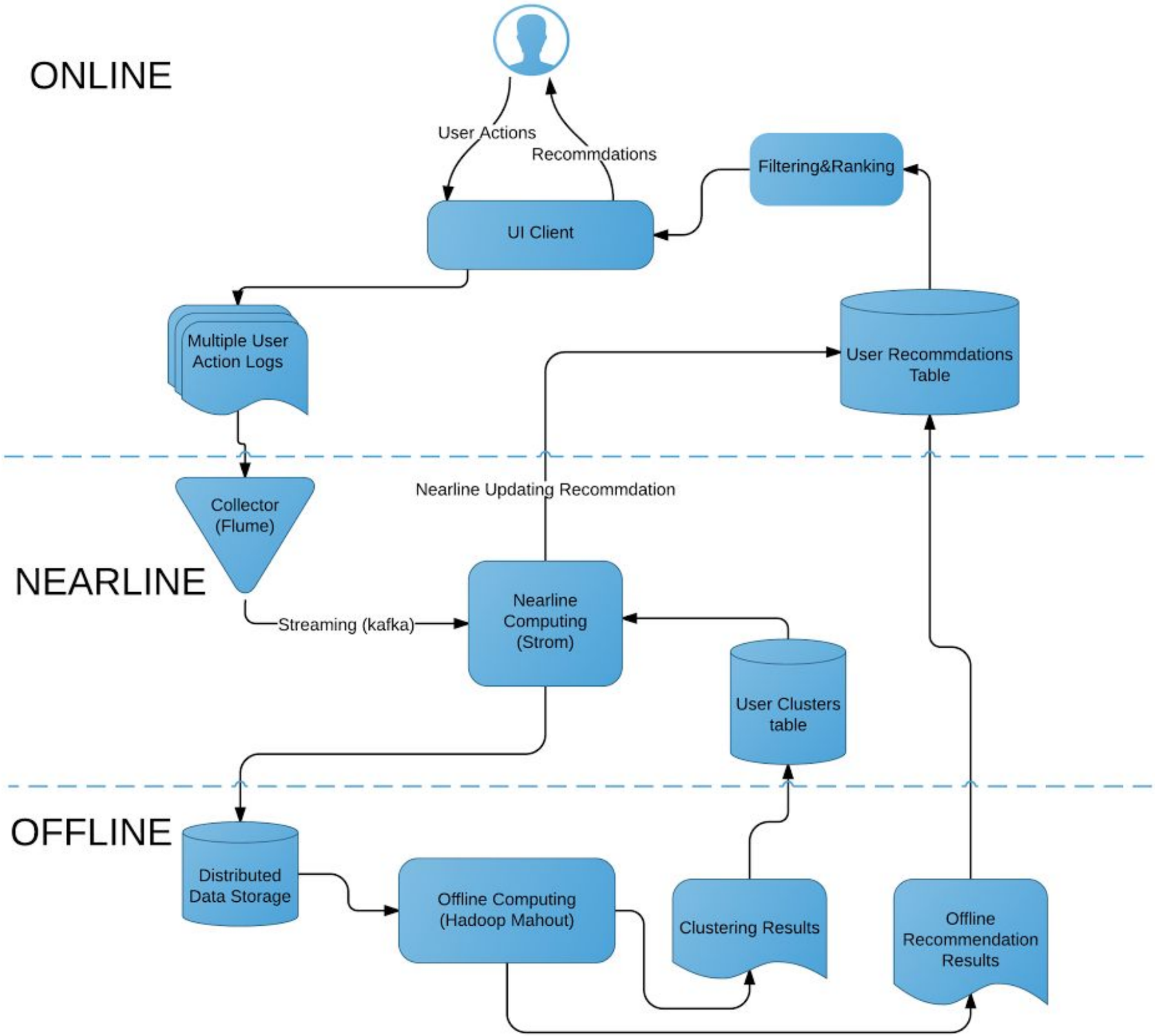
最新动作

- 订阅系统
  推荐系统


MESSAGE QUEUE
USER: FOLLOWERS CACHE
WORKER
WORKER
WORKER
DEAMON
REDIS
REDIS
REDIS
USER: ARTICLES LIST CACHE

订阅系统

- 来源：feeds，附近的糗事，热门
数据聚合: Redis, MongoDB, Memcached
过滤：看过的不再展示
排序：各种Tag的优先级



推荐系统

- ing...
  目前进度：hadoop集群，用户画像
  Welcome to join us!

广告时间

后端工程师

算法工程师

运维工程师

web前端工程师

Android工程师

来扫一扫吧

HR的微信号

kevincechen

Thanks!