WOTA

51CTO

World Of Tech 2017

全球架构与运维技术峰会

2017年4月14日-15日　北京富力万丽酒店

ARCHITECTURE

WOTA

51CTO

出品人及主持人：

赖春波 滴滴出行平台技术部 高级技术总监

高可用架构

WOTA

51CTO

大流量网站的高可用建设经验

君山

2017.4.15

WOTA

51CTO

君山

滴滴出行
技术研究员

分享主题：

大流量网站的高可用架构演进实践

关于我

- 许令波(君山)
- 在阿里待过，有大流量web系统的优化经验
- 关注大流量系统的高可用架构和性能优化工作。
- 在滴滴做容器化和资源调度建设

目录

- 高可用网站的挑战
- 有那些实践
- 我们的经验

何谓高可用

There are three principles of systems design in reliability engineering which can help achieve high availability.

1. Elimination of single points of failure. This means adding redundancy to the system so that failure of a component does not mean failure of the entire system.
2. Reliable crossover. In redundant systems, the crossover point itself tends to become a single point of failure. Reliable systems must provide for reliable crossover.
3. Detection of failures as they occur. If the two principles above are observed, then a user may never see a failure. But the maintenance activity must.

- 三个原则
  - 故障监测与排除
  - 消除单点故障
  - 互备和容灾

架构上的挑战—大流量

- 高伸缩性
- 读cache

架构上的挑战—业务复杂性

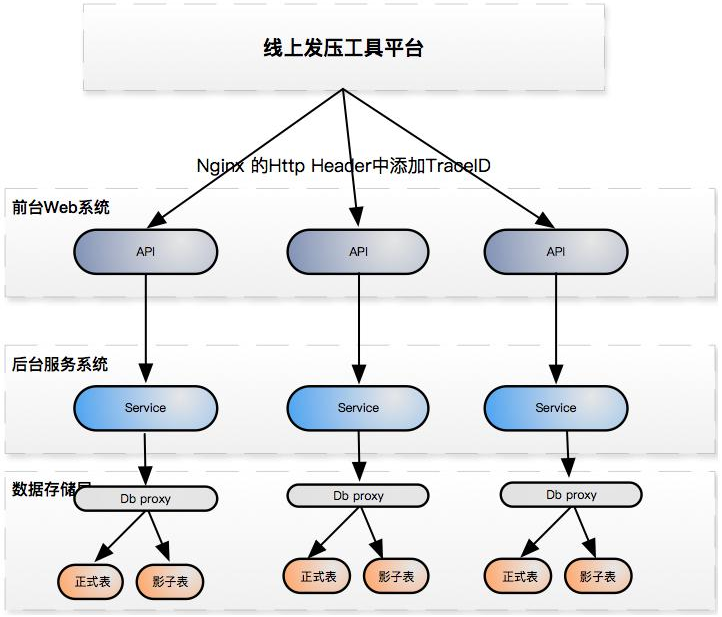
- 架构上
  - 分布式化
  - 业务功能域服务化
- 业务架构匹配组织架构
  - 开发、发布效率
  - 问题排查与责任划分

有哪些实践

- 故障检测与排除
  - 全平台压测
- 分布式化改造
  - 消除单点增加冗余
  - 系统拆分、大团队协作
- 大流量架构改造
  - 高伸缩性
  - 热点数据的自动检测

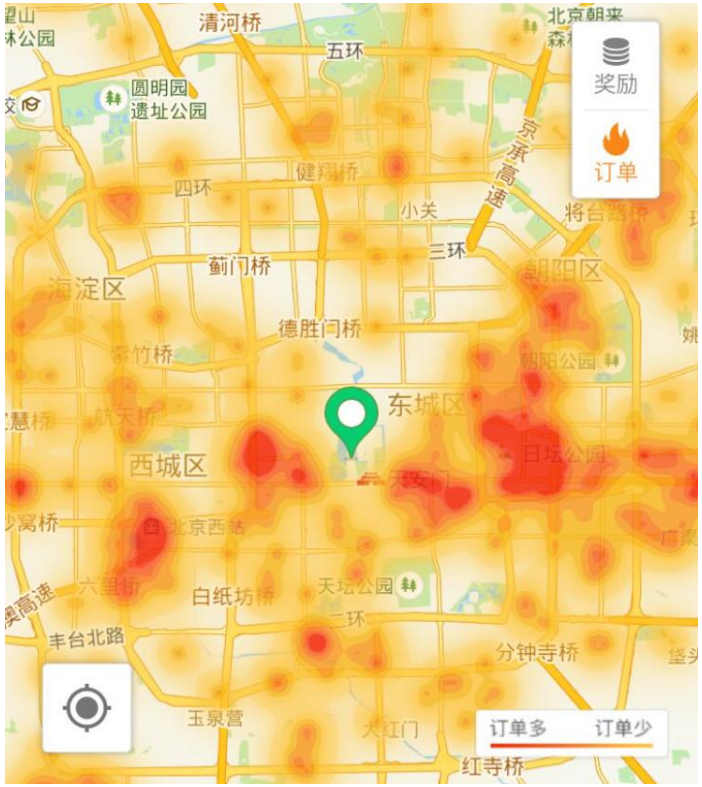
故障检测与排除—全平台压测

- 通过trace传递标识流量中间件等
- 通过标记将流量导到影子表



故障检测与排除—全平台压测数据构造

- 坐标
  - 压测坐标经纬度偏移到太平洋
- 虚拟乘客和司机
  - ID偏移、手机号替换
- 订单数据
  - 订单号在保留区间

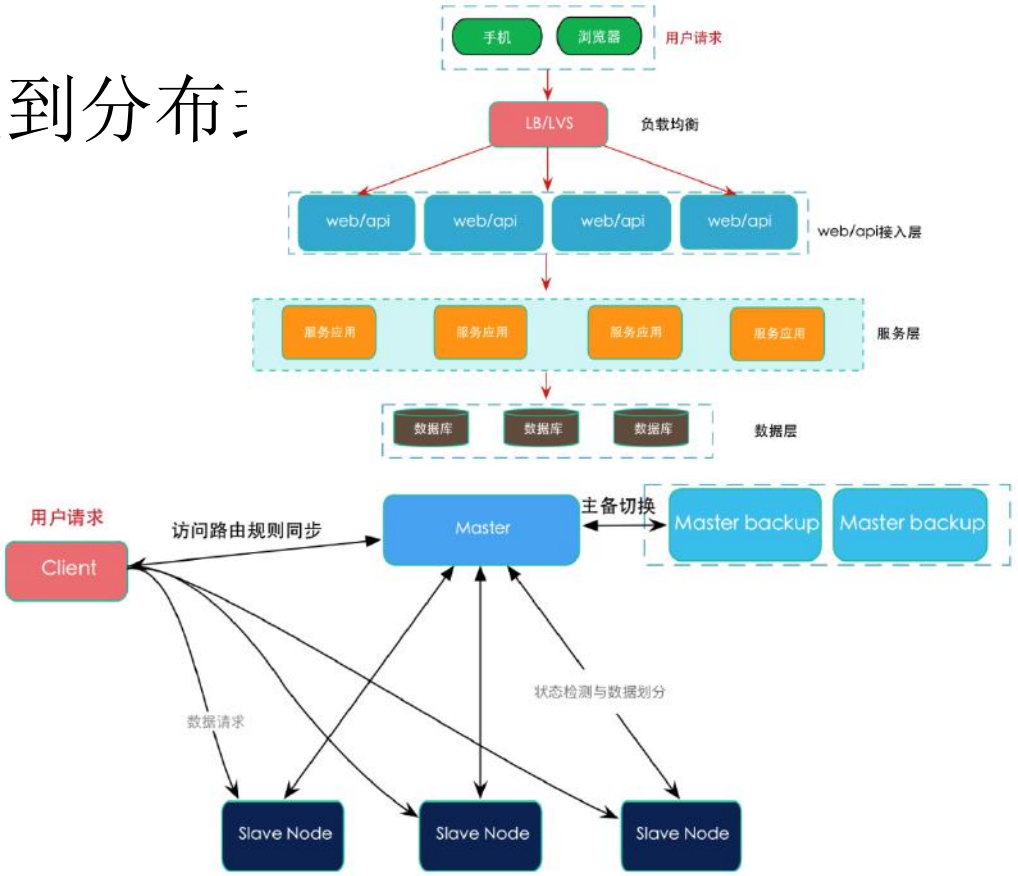


故障检测与排除—发现的问题

- 业务线
  - 顺风车：接口耗时增长，如列表页面: 100ms => 700ms
  - 顺风车：日志搜集的ftp—server夯死
  - 专快：STG访问codis出现超时
  - 出租车：getdriverstatus 触发限流
  - 出租车：STG单条日志量太大 影响性能
- 基础平台
  - NAT：2台NAT启动无用的内核模块，流量大时大量丢包
  - LBS：lbspp_index写入超时，lbspp_proxy查周边接口有超时
  - 地图：路径规划服务，到达容量瓶颈
- 压测工具导致的其他问题
  - 专快计算超时：由于工具问题，司机和订单陡增，km算法超时，主要是日志过多导致

分布式化改造—典型的分布式架构

- 最重要的业务2层要做到分布
  - 服务的无状态化
  - 每个节点都是对等的
- 数据层有状态
  - 要解决冗余和备份
  - 故障切换

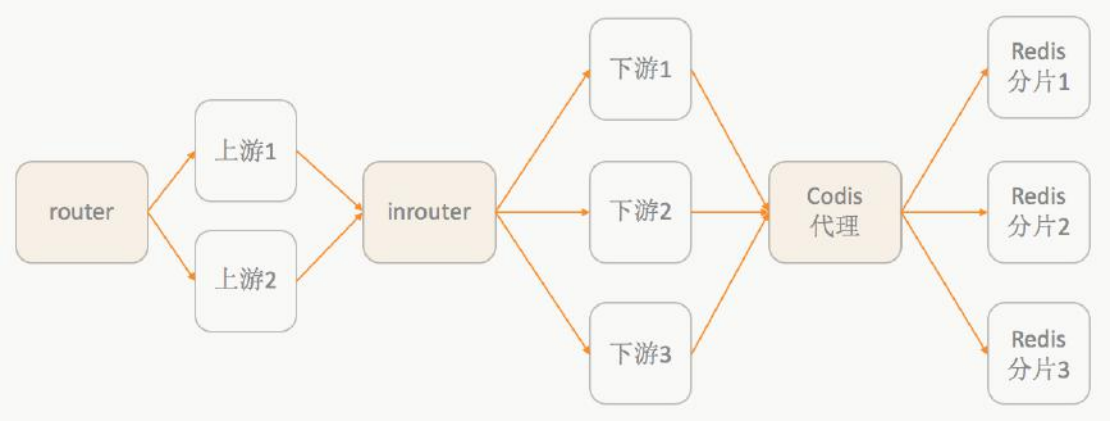


分布式化改造—关键技术点

- 分布式RPC框架
- 分布式消息框架
- 分布式配置框架

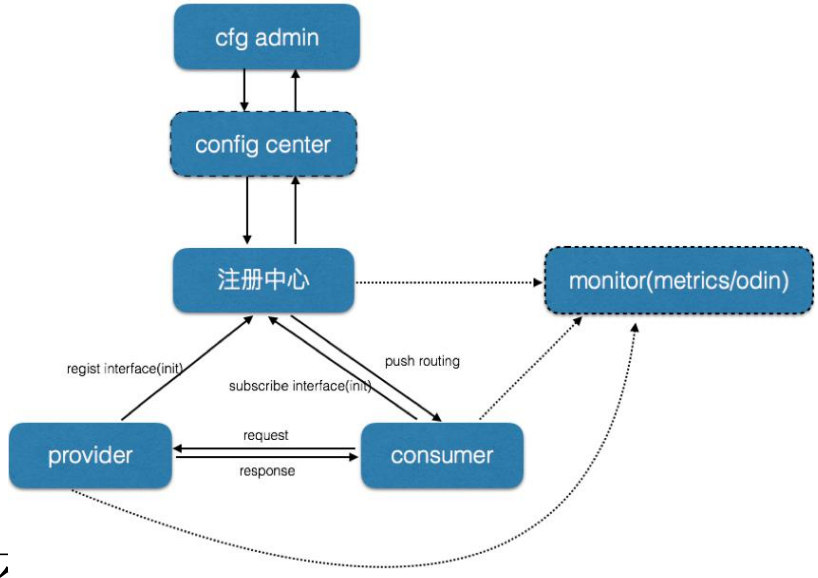
分布式化改造—服务治理

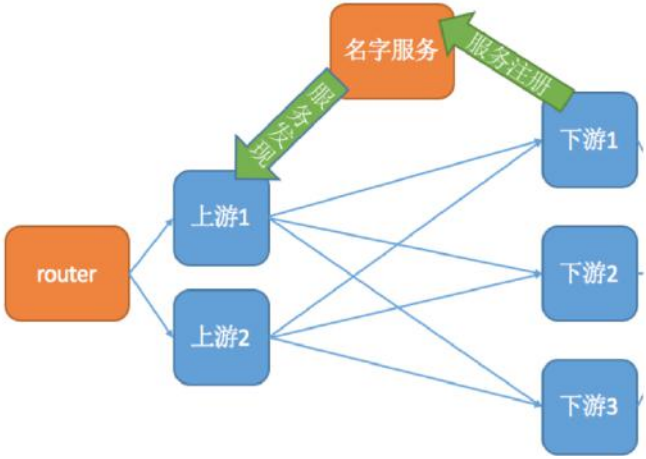
- 上下游依赖硬编码在代码里；
- 没有使用inrouter/tgw的ip:port 摘除和扩容需要代码重新上线；
- Inrouter有网络链路稳定性隐患，以及效率上的损失；
- 没有清晰的服务目录，API文档以及SLA和监控。



分布式化改造—RPC框架

- Naming服务
  - 服务命名
  - 注册和发现
- RPC通道
  - 私有协议
  - 可扩展性
- 服务路由与容灾
  - 动态路由
  - 故障节点摘除





分布式化改造—RPC框架实践经验

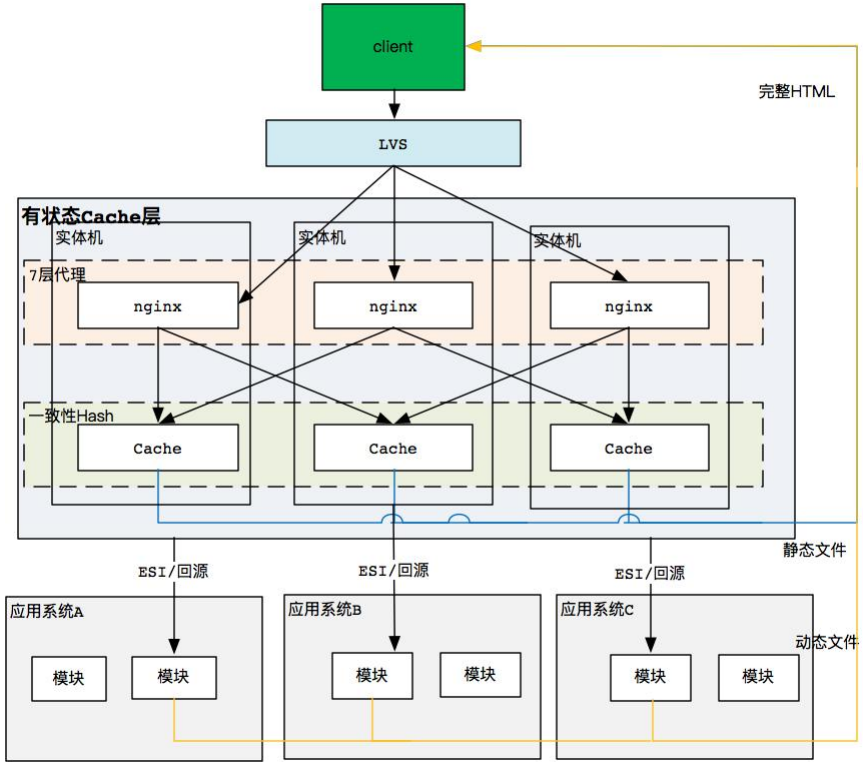
- 业务技术栈最好统一：Java或者Go等
  - Java成熟的Dubbo，其他多语言的DiSF（后期会开源）
- 服务命名要规范
  - 服务名自描述
  - 要构建统一服务树
- RPC协议是Http还是私有协议？
  - 方便性VS滥用
- 服务路由是全局摘除还是本地拆除？
  - 自动化VS风险

分布式化改造—大团队协作问题

- 从单业务系统做分布式改造的一个出发点就是解决大团队分工和协作问题
  - 代码分支分拆，减少代码冲突
  - 系统独立，打包和发布效率都会提高
  - 部署独立，线上故障排查和责任认定更加明确
- 带来的问题
  - 依赖的不确定性增加
  - 性能上一些损耗

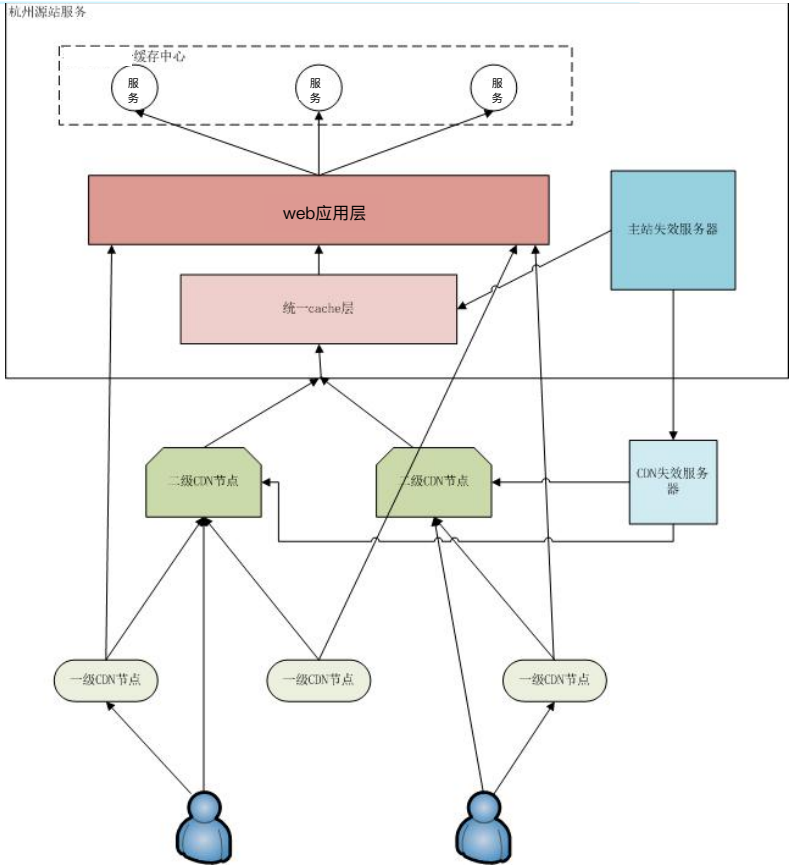
大流量系统的高可用迭代—从无状态到有状态

- 单纯的横向扩展已经不能解决问题
- 需要带有分组功能的一致性Hash的cache



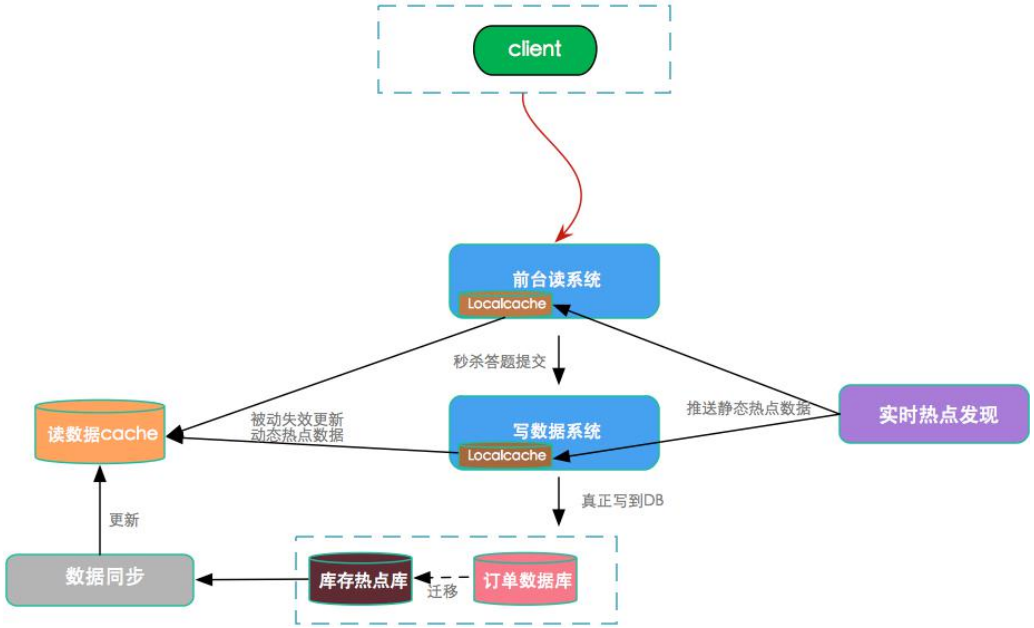
大流量系统的高可用迭代—更大的伸缩性

- 静态内容前置到CDN，具备更大的伸缩性
  - 可以抗百万级别流量
  - 节省带宽
  - 节点分散，可用性更高



大流量系统的高可用迭代—热点侦测

- 大流量网站都会遇到热点击穿问题
  - 即1%的热点会影响99%
  - 上游热点自动侦测
  - 下游热点自动cache



大流量系统的高可用架构经验

- 热点隔离
- 先做数据的动静分离
- 将99%的数据缓存在客户端浏览器
- 将动态请求的读数据cache在web端
- 对读数据不做强一致性校验
- 对写数据进行基于时间的合理分片
- 实时热点发现
- 对写请求做限流保护
- 对写数据进行强一致性校验

高可用建设经验—需要体系化

- 建立规范和责任体系
  - 高可用指标（KPI）
  - 故障等级
  - 责任与荣耀体系
- 工具要完善和体系化
- 需要要配套的组织保障

高可用建设经验—需要积累和沉淀

➢变更之中出现问题第一时间回滚

➢在变更之前必须制定回滚方案

- 对变更内容设置开关，出现问题可以快速通过开关关闭新功能
- 接口变更、数据结构变更，回滚要考虑第三方依赖

➢指导原则：

- 将故障清晰描述和暴露出来，获取第一手资料，找到问题反馈源头
- 先解决问题，消除故障
- 找到对应系统和业务的直接负责人

➢处理流程：

- 问题发现后第一时间上报到“消防群”
- 组建应急处理小组
- 跨团队合作，通知到对方系统的负责人，P1故障要通知到客服、公关接口人，尽量做到集中办公
- 问题处理完毕，立即总结和制定改进方案
- 系统TL负责，改进方案的执行情况

WOTA

51CTO

Thank you !

- 邮箱:xulingbo0201@163.com
- http://xulingbo.net