2019年的微服务

江南白衣

唯品会基础平台部资深架构师

极客时间 | 企业服务

这是一封正经“福利”贴

想做团队的领跑者 需要迈过这些“槛”

成长型企业，易忽视人才体系化培养 企业转型加快，团队能力又跟不上	VS	从基础到进阶，超100+一线实战 技术专家带你系统化学习成长
团队成员技能水平不一， 难以一“敌”百人需求	VS	解决从小白到资深技术人所遇到 80%的问题
寻求外部培训，奈何价更高且 集中式学习	VS	多样、灵活的学习方式，包括 音频、图文 和视频
学习效果难以统计，产生不良循环	VS	获取员工学习报告，查看学习 进度，形成闭环

课程顾问「橘子」

回复「QCon」
免费获取
学习解决方案

 # 极客时间企业账号 # 解决技术人成长路上的学习问题

自我介绍

20年经验

老派程序员

+ 开过源 SpringSide , VJTools
+ 写过公众号 春天的旁边

现状1：人人都有一套微服务

服务调度： 服务注册发现，负载均衡

服务治理： 超时，限流，熔断，降级

服务监控： 分布式调用链，指标，日志

基础设施： 配置中心，API网关

限流	限流
Guava RateLimiter	Alibaba Sentinel

现状2: 关于未来，只听过Service Mesh

Service Mesh 的 真实好处

- 跨语言接入，零成本接入
- 升级力，新即正义

业务同学 到底需要什么？

- 规模化之后的， 全生命周期的
- 开发效率，运维效率，性能，可用性

目录

目录

1 Spring Cloud 够了吗

我们要怎样的基础能力

Dubbo, Envoy 又如何

负载均衡

权重，经典而实用的技术

- 灰度
- 压测
- 摘除流量
- 新实例热身
- 不同性能的机器混合部署

语义上不支持权重： 加权响应时间，活跃调用数

一致性哈希

- Stick Session
- 本地缓存
- 本地计数

路由

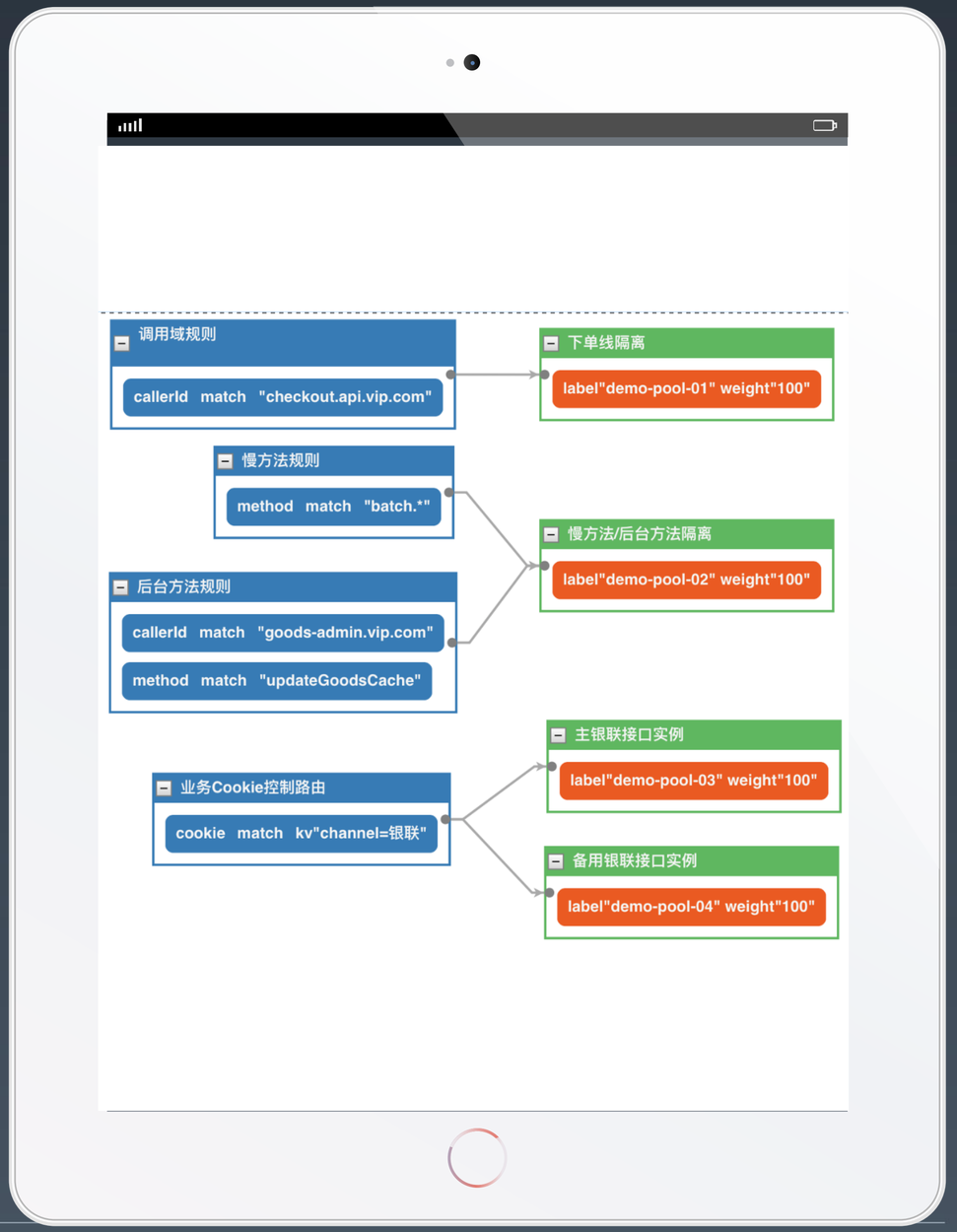
应用分流 — 上游应用标识

跨机房调度 — 上游 IP段 / Zone标识

快/慢 接口分离 — 方法名

前/后台 接口分离 — 方法名

业务自定义规则 — Header 信息

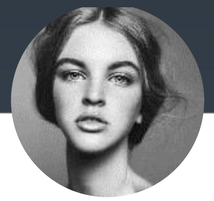


超时（1）- 谁做主

基本但其重要性，

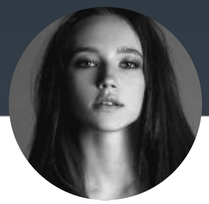
在不怕死就怕慢的分布式系统里，

占了三分之一



- 我的服务，性能我清楚
- 公共可见，工具友好
- 可为个别上游定制

服务端 @ 治理中心

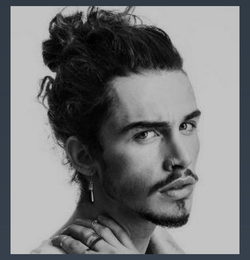


- 为我定制？别当康威定律不存在
- 不同场景，不同超时

客户端 @ 代码

经验教训： 客户端猛报超时，服务端岁月静好，因为不知晓对方设置

超时(2) - 各有增强



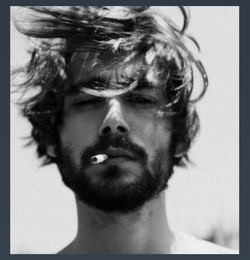
1. 框架：

“我知道你已经超时了”

执行前超时：不调用业务代码

执行后超时：不序列化

不传输结果



2. 业务代码：

“框架，我还有多少时间？”

办大事前 - 如 RPC/DB call

Context.getTimeLeft()

省得白干活，还得回滚补偿

3. 调用链：

“让我来一路传递上游的剩余时间”

如果上游已超时

除了补偿操作 ，

其他什么都不要再做了

重试

明明是好东西，为何设置的同学，

眼里总是 饱含挣扎？

重试限流

不做压垮系统的最后一根稻草

服务总体超时

下游重试爽，上游等得慌

连接异常默认重试

非幂等的服务如何抢救一下

牲口一样重启

优雅停机易，首次调用超时难

1

预热

根据重启前的记录

1. 下游服务的元信息
2. 下游服务的TCP连接
3. Java Class

2

新实例热身

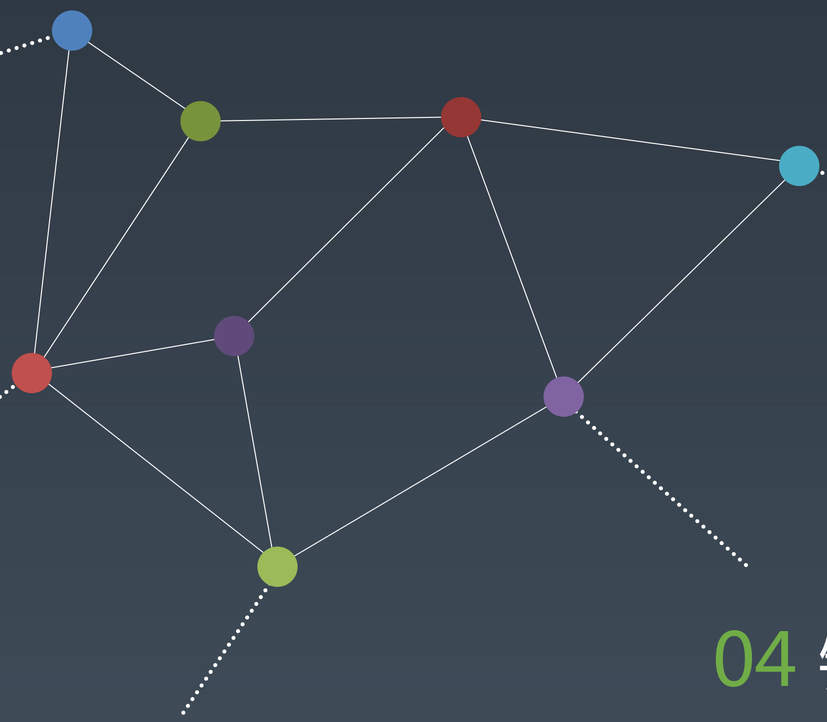
第一分钟的权重逐渐放大

3

GC热身

连续GC到全部晋升

单机故障处理



01 注册中心心跳

微务框架的基础

02 健康检查

容器化的基础

03 单实例熔断

我还坚强活着，但是....

04 链路空闲心跳

大家都活着，链路断了

莫等 TCP KEEPALIVE

05 运维监控的自动化处理

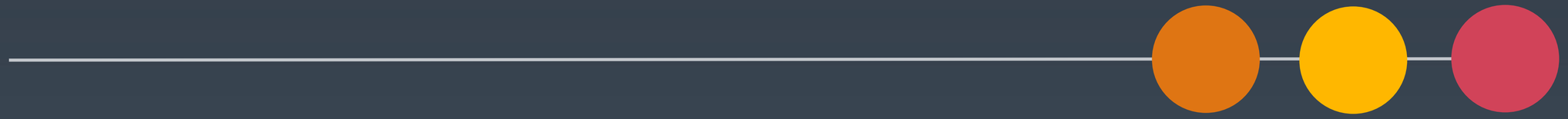
硬盘故障，网卡掉速

开放流量摘除接口

超越Tomcat - 优雅的方法隔离线程池

正常时：高利用率的，公共池

缓慢时：互相隔离的，方法独立池



总是将任务先提交给 公共池 (QueueLength=0)

拒绝时将任务提交给 方法池 (CoreSize=0)

业务同学喜欢的自动 Thread Dump

继续超越Tomcat

与业务代码隔离的 ClassLoader

- 基础组件依赖的3PP库，与业务代码依赖的冲突
- 基础组件不敢自动升级

夜半无人的 FullGC

- 减少白日 CMS GC 的概率
- 整理老生代碎片
- 执行之前反注册

服务配置中心（1）- 另一个配置中心

功能和配置中心一样： 独立的UI，相同的后台

- 动态下发
- 灰度下发
- 版本管理回滚
- 工单系统集成

变更时间窗口控制，高风险变更审批， 根因分析回溯

配置报表

- 谁配了复杂路由
- 谁改过了熔断的默认值
- 谁配了两次以上的重试

服务配置中心（2）- 条件表达式配置

示例： 超时配置

```
{"method": "getCart", "callerId": "checkout.api.vip.com", "value": 700 },
{"method": "getCart", "value": 400 },
{"value": 200 }
```

“从 checkout 应用来的“获取购物车“，**700** ms

其他应用来的“获取购物车”，**400** ms

偷个懒，其他方法，**200** ms”

字幕组

服务治理中心

注册中心 + 服务配置中心 + 文档中心 + ？

监控中心？ 发布系统？ 混沌测试系统？

No！

从整个运维体系布局，功能内聚

以开放API，与运维体系互通

目录

ServiceMesh 美丽与哀愁

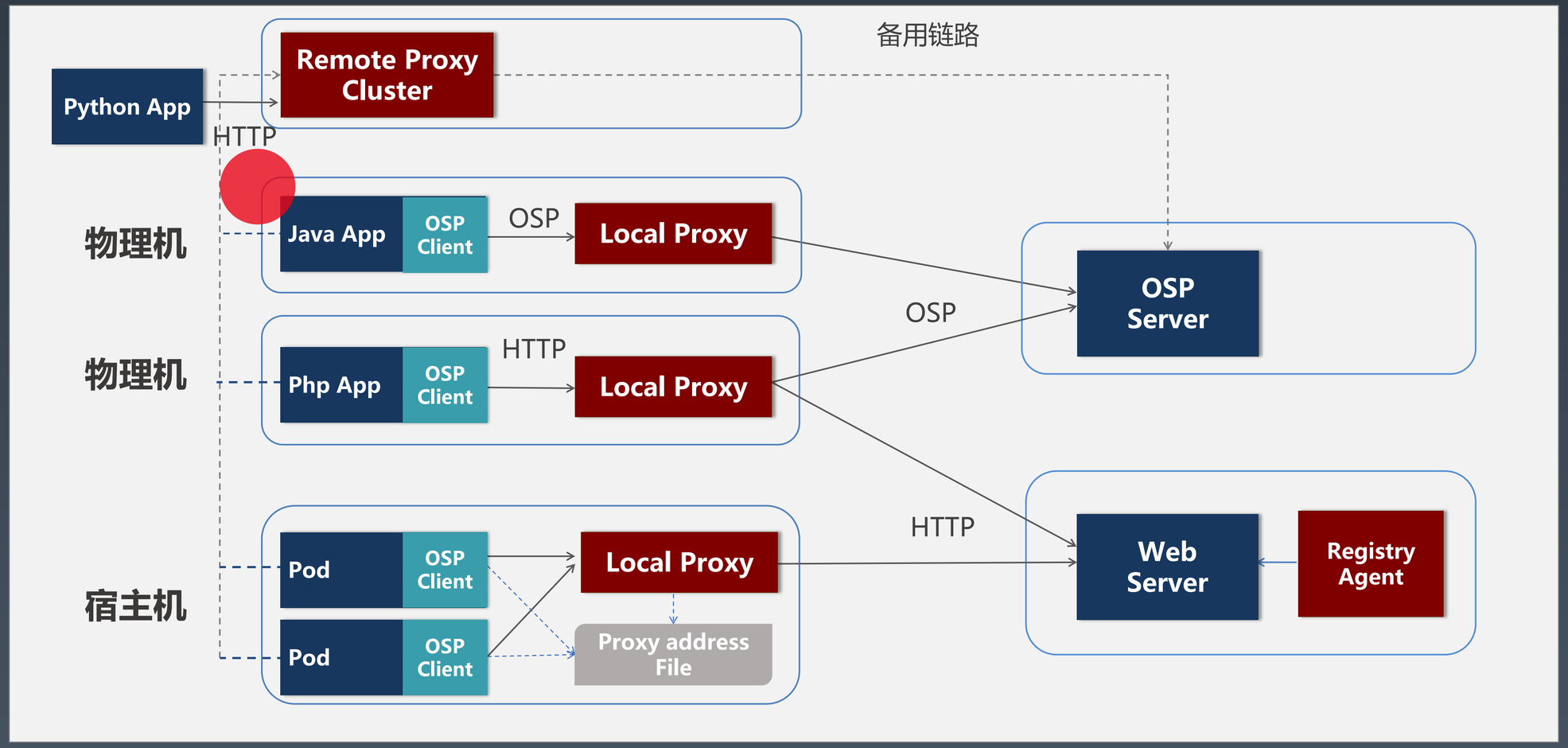
2

三年，自下而上的进化

ServiceMesh 架构的折衷

我们的“类 ServiceMesh”架构

Java、PHP、C++多语言 / Proxy 快速升级 / 未改造 Web Server / 容器、物理机混合部署

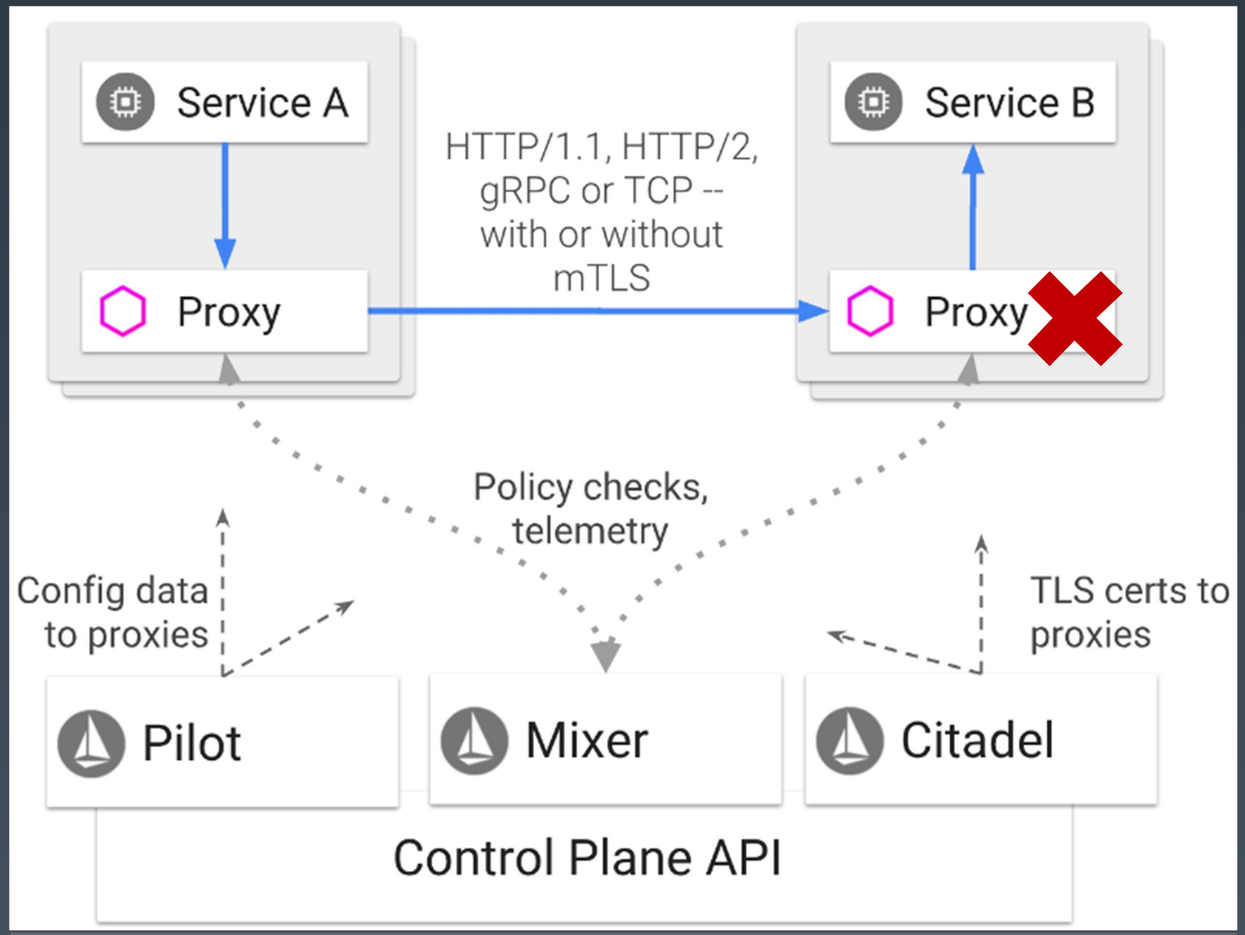


Server Side Proxy ?

Server端 零改造成本， 但...

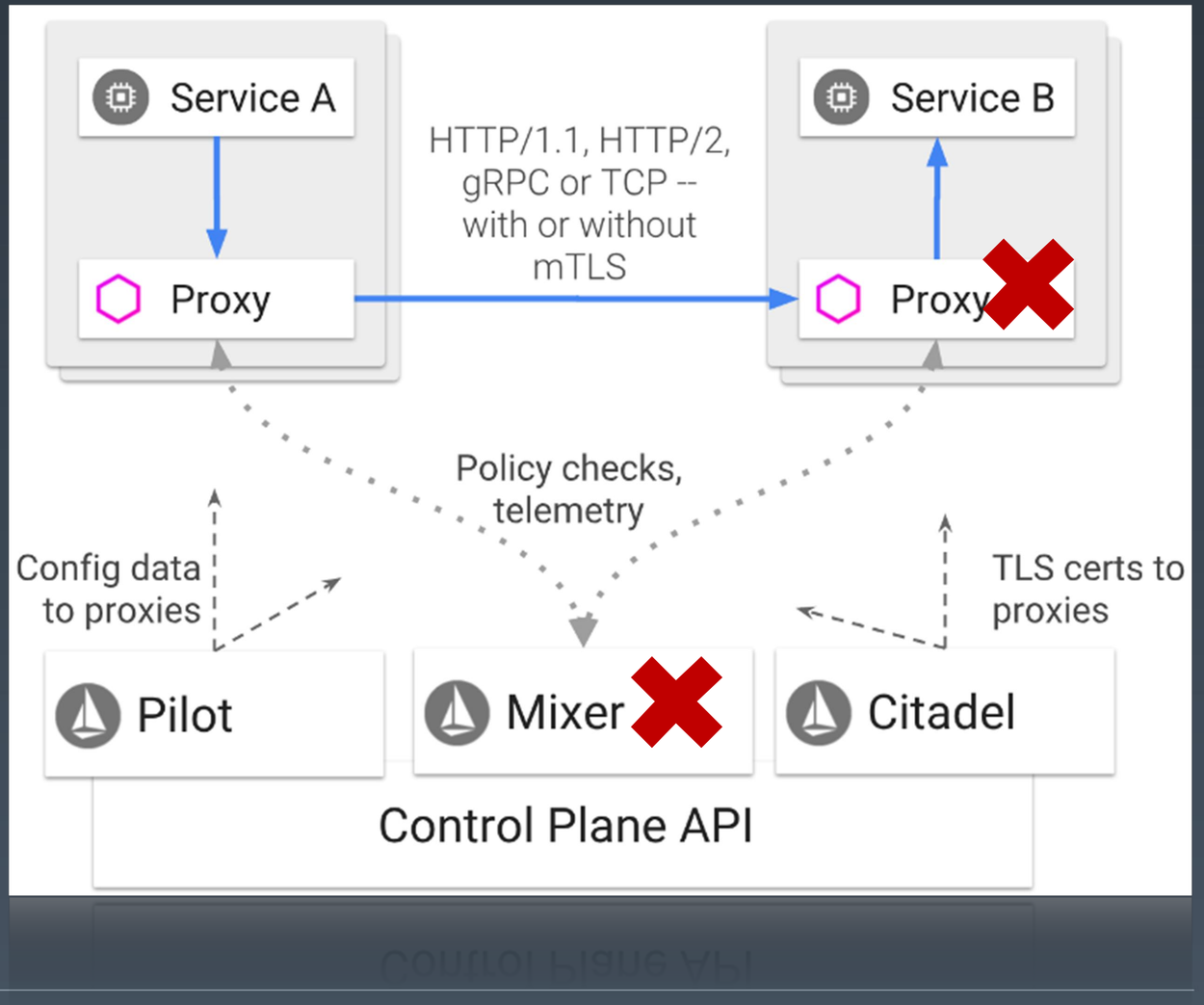
- Client Proxy 已加一跳，Server 还来？
- 服务端的治理能力，Proxy 形式足够吗？
- 分布式调用链/故障注入，不限于 RPC

跨语言WebServer： 轻量级注册Agent



中央Mixer ?

- 巨大的时间消耗
  新的中央瓶颈
- 漂亮架构图的产物
  数据面可替换性大饼的牺牲品
- 社区努力改进中
  下沉， 缓存，异步发送



基于IPTable拦截？

客户端 跨语言，零改造成本，但....

IPTalbe 性能总是不好，服务 越多越慢

应用不想和 Proxy 同生共死

静态路由，防火墙穿透路由，Proxy 隔离排查

主流SDK：提供Local / Remote Proxy切换
非主流语言：访问 Remote Proxy Cluster

SideCar VS DaemonSet

良好的 隔离性, 但...

- Java Agent 吃内存
- SideCar 升级， 要把全网应用重启一遍

Proxy对来源IP限流 :高出阈值的流量，
重定向到 Remote Proxy Cluster

目录

MicroService 往何处去

3

那些 微服务 与 SM 框架
短期还没顾上的

基于契约编程（1） - 服务定义

SDK First， API First， Java First

- 用 Java接口/实体类 定义 Thrift/PB 协议
- JSR-349 Bean Validation 1.1
- 其他增强
  - Entity 类继承
  - 全局动态枚举
  - Annotation 定义参数抽取到Header...

基于契约编程（2）- 接口文档编写

Annotation文档， 接口/文档一体化，版本管理，但...

Java的字符串的局限： 不支持多行，转义字符，可视化编辑

文档中心 在线编辑， 与Java文件 双向转换

格式漂亮，还能带长长的使用示例

基于契约编程 - GraphQL的尝试

剪裁GraphQL： 由客户端定制返回的内容

SDL： { allBooks { id, title } }

保留契约化编程特征：基于生成的SDK

REST不是一场革命

GraphQL也不是

```
GqlQuery query =  GqlQuery.newBuilder(BookServiceClient.ALLBOOKS)
                                .add(BookField.id, BookField.title)
                                .build();
```

全链路治理

- 分布式调用链的汇总

核心业务流程建模

强弱依赖，流量漏斗模型

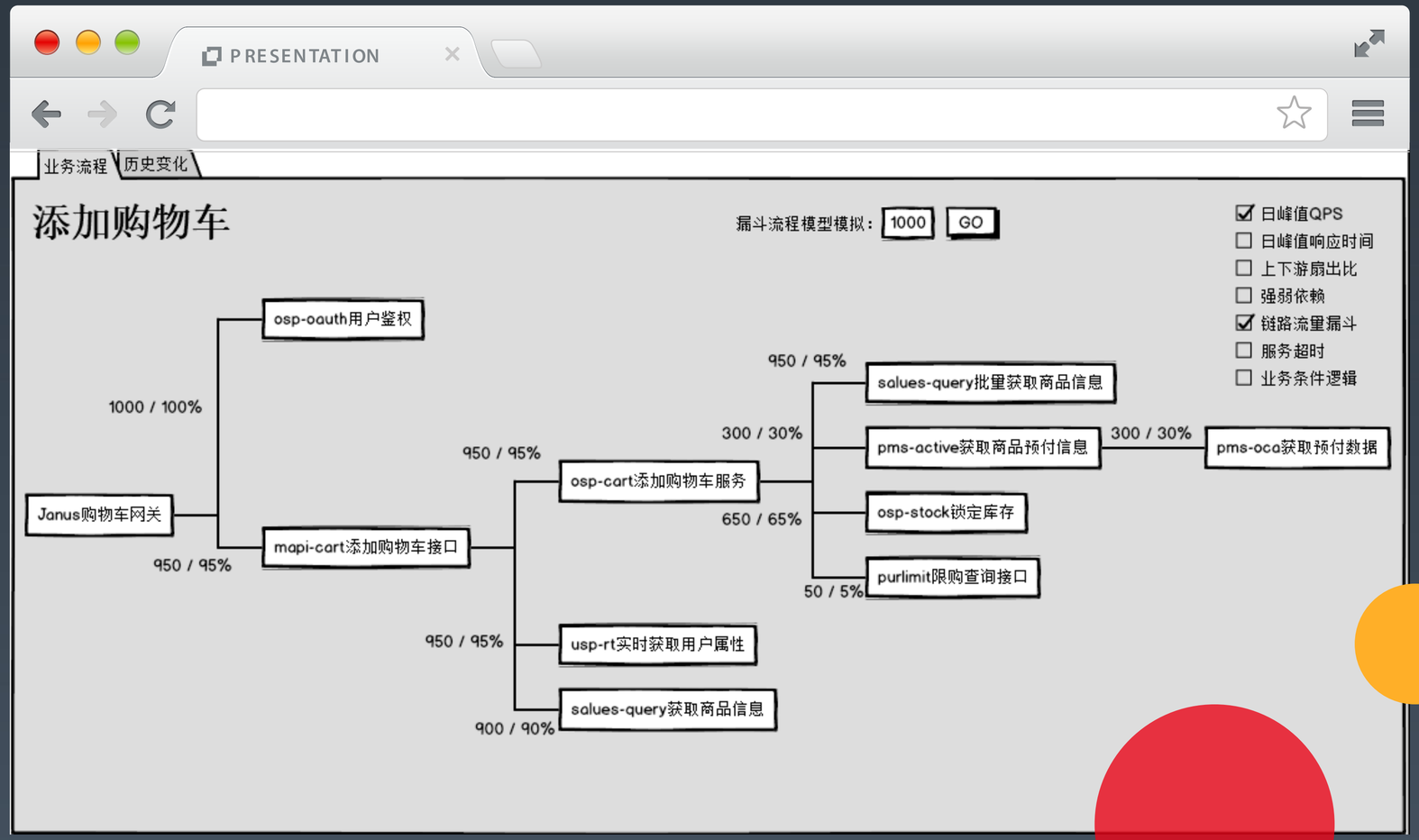
流量模型变化的历史比对

流量模型变化的实时通知

链路超时设置治理

故障影响分析辅助

容量规划辅助



全链路生态圈

1. 版本共知

- 提供者查看所有消费者SDK版本
- 提供者一键提醒消费者升级SDK

2. 变更共知

- 消费者获知 服务文档 变更
- 消费者获知 服务配置 变更

3. 配置共管

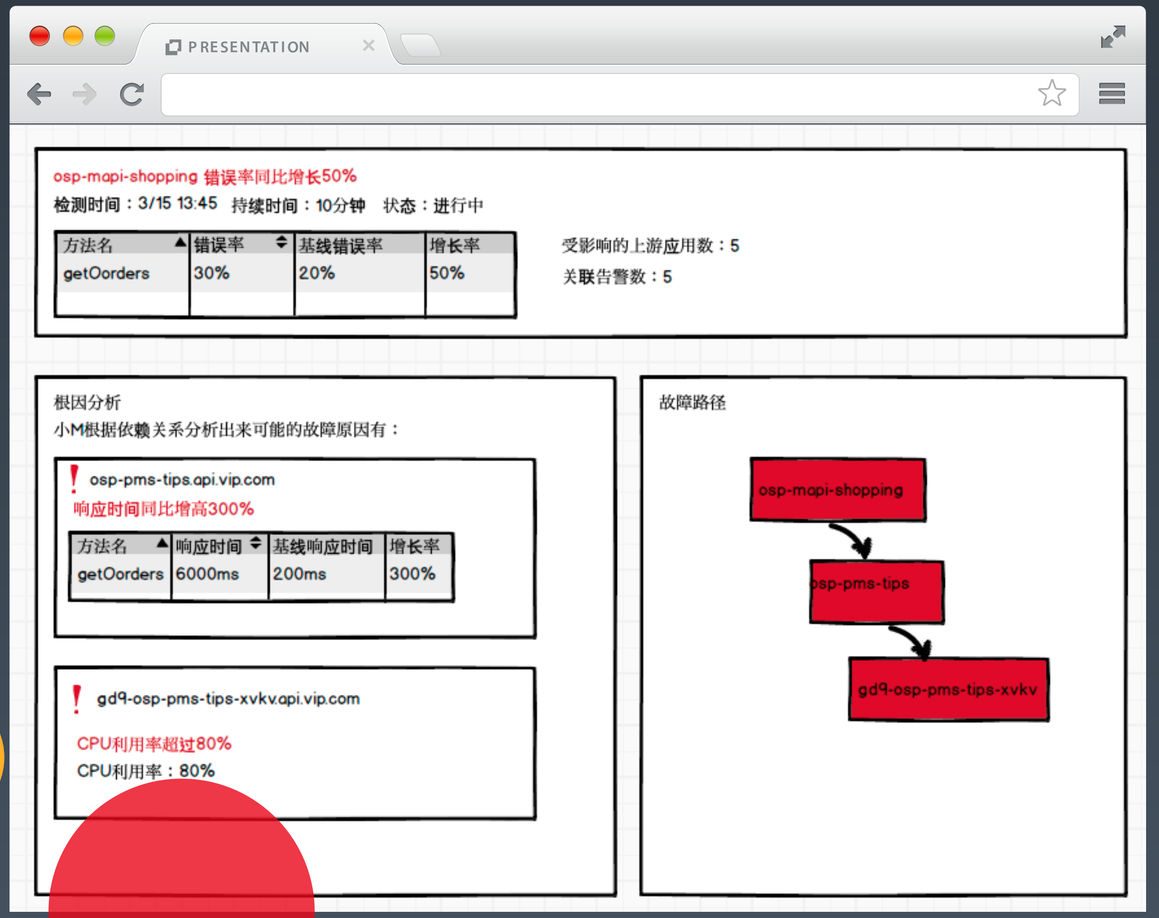
- 消费者在服务配置中心定制客户端配置

智能化根因分析

人工智能 并非必需品

简单的 统计学算法 和 文本分析

- 链路拓扑，时间顺序
- 统计学上的指标分析
  - EWMA，线性回归，Holt-Winters 等
- 分布式调用链
- 异常日志文本分析
- 变更事件



智能化参数治理

默认值不能在任何场景都是最好
让业务来配置又痛苦

基于历史指标的自动推荐

- 超时
- 熔断
- 最大线程数

基于实时指标的动态调整

- 限流
  - Latency
  - Threads
  - CPU usage / CPU load
- 扩容
  - 在CPU和网络之外，考虑应用指标
- 分布式日志采样率
  - 非繁忙时提高采样率

分布式事务

需求：

1. 跨分片/跨服务的 DB 事务
2. 跨服务的 DB / NoSQL / MQ ...

原始时代：

1. 执行顺序，补偿
2. 对账, 消息表

TCC，Saga

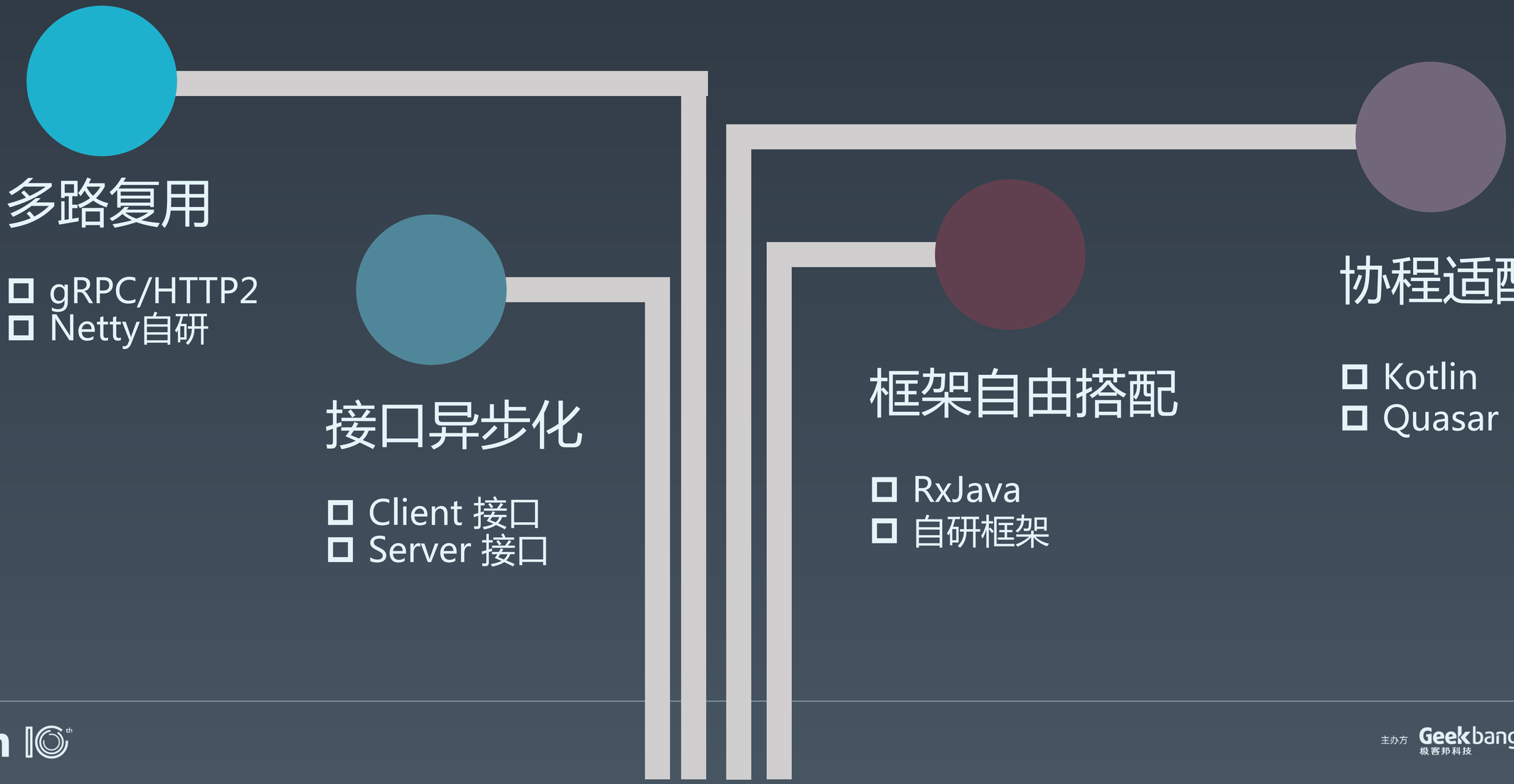
Apache ServiceComb Saga

Apache ShardingSphere for DB

Seata

阿里巴巴 plus 蚂蚁金服
期待超越 DB 的事务

全异步化


多路复用
gRPC/HTTP2
Netty自研
接口异步化
Client 接口
Server 接口
框架自由搭配
RxJava
自研框架
协程适配
Kotlin
Quasar

公共基础服务集

- 唯一 ID 生成
- Leader 选举
- 分布式锁
- 全局限流
- 敏感信息加密

测试工具集

线下

- 在线测试
  类似 Swagger
- 服务模拟器
  类似 Moco
- 线下压测工具
  基于 JMeter等

线上

- 引流压测
  自动逐步调整权重，按条件自动终止
- 流量录制播放
  按需录制, 协议头处理，速度控制
- 全链路压测
  流量染色，影子库
- 故障注入混沌测试

发布工具集

自动发布系统

- 至少三批滚动发布
- 每批自动测试
- 关键指标正常，自动下一批
- 关键指标异常，人工决定回滚

（“有内鬼，停止交易.jpg”）

变更巡检系统

- 发布后半小时，1小时，2小时执行
- 综合昨天同期 及 今日发布前数据
- 关键指标的趋势检查
  - CPU，内存，线程，GC ...
  - 响应时间，4XX/5XX，错误日志
  - 关联业务KPI，上下游域的关键指标

Geekbang. 极客邦科技 InfoQ

极客邦科技
会议推荐2019

5月

QCon 北京
全球软件开发大会
大会：5月6–8日
培训：5月9–10日

QCon 广州
全球软件开发大会
培训：5月25–26日
大会：5月27–28日

6月

GTLC GLOBAL TECH LEADERSHIP CONFERENCE 上海
技术领导力峰会
时间：6月14–15日

GMTC 北京
全球大前端技术大会
大会：6月20–21日
培训：6月22–23日

7月

ArchSummit 深圳
全球架构师峰会
大会：7月12–13日
培训：7月14–15日

10月

QCon 上海
全球软件开发大会
大会：10月17–19日
培训：10月20–21日

11月

GMTC 深圳
全球大前端技术大会
大会：11月8–9日
培训：11月10–11日

AiCon 北京
全球人工智能与机器学习大会
大会：11月21–22日
培训：11月23–24日

12月

ArchSummit 北京
全球架构师峰会
大会：12月6–7日
培训：12月8–9日

THANKS!

QCon 10th