

# 目录

## 01 多活架构解析

## 02 多活最佳实践

## 03 多活管控治理

ITIL 先锋论坛

我们是谁?  
WHO ARE WE?

国内最大的数字化时代IT服务管理交流社区，自2010年底成立以来，始终致力于以 ITIL 为代表的IT管理方法论在国内的推广与落地。

我们的服务  
OUR SERVICES

数十个专业微信群、近千篇可一键下载的资料、视频号专家直播、全国一线城市巡回聚会、开源免费ITIL软件、国内最权威的ITIL知识库

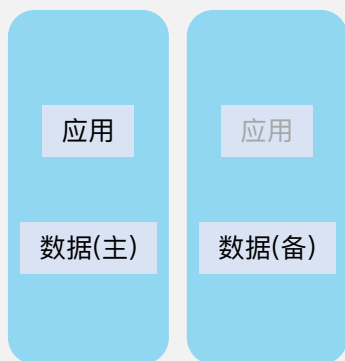
01

# 多活架构解析

## 多活方案类型

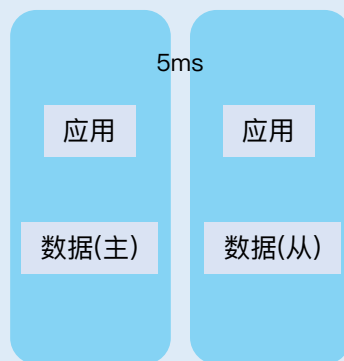
多活通常指不同地理位置上的数据中心，部署应用服务和数据，基于流量调度管控能够同时提供服务

### 单活/灾备



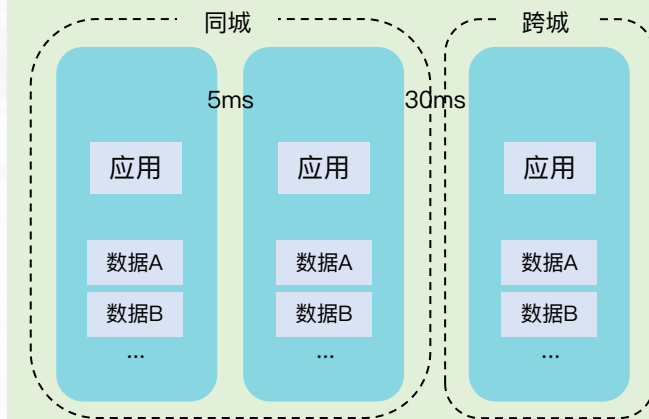
多数据中心

### 同城多活



同城多数据中心

### 异地多活



异地多数据中心

高效

## 同城多活

同城多活（同城双/三中心）：业务部署在同一个城市不同数据中心，可划分为两个/多个可用区，由数据中心间专线支持提供<5ms网络延迟。数据层具备同步和切换能力。

### 主要优势：

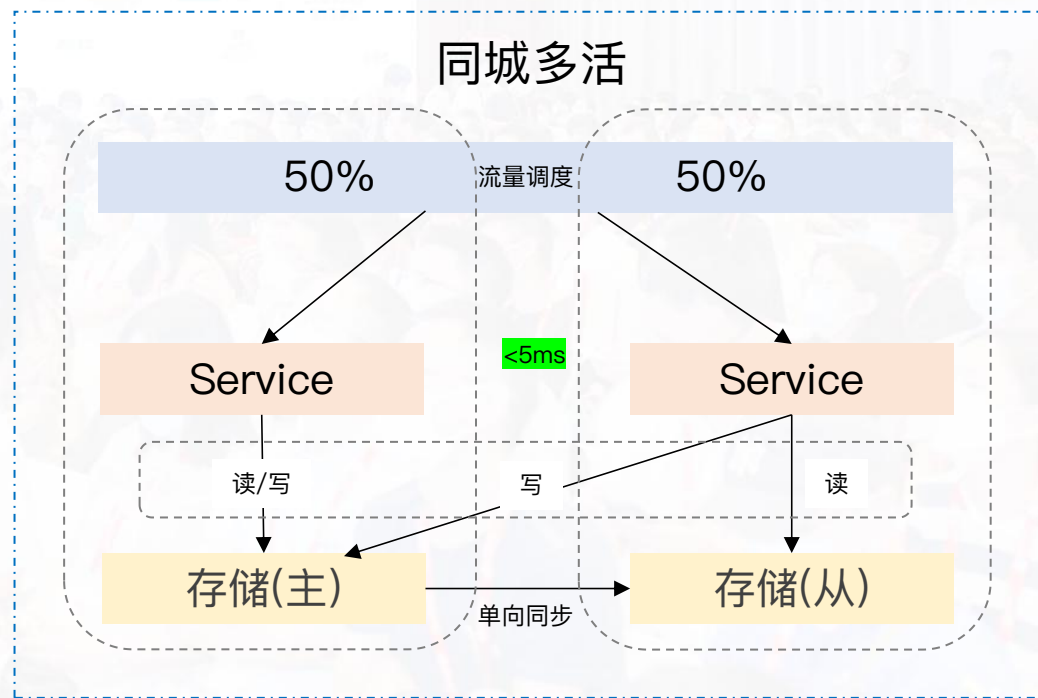
- 架构方案比较简单，业务改造接入成本低；
- 数据中心距离近，数据同步延迟低，易保证数据最终一致性；

### 劣势：

- 无法抵抗城市级别故障/灾难，但极端灾难发生概率比较低；

### 挑战：

- 流量调度管控、数据接入层能力支持
- 复杂业务场景下链路会出现频繁跨机房调用增加响应时间，影响业务耗时和用户体验。



## 异地多活

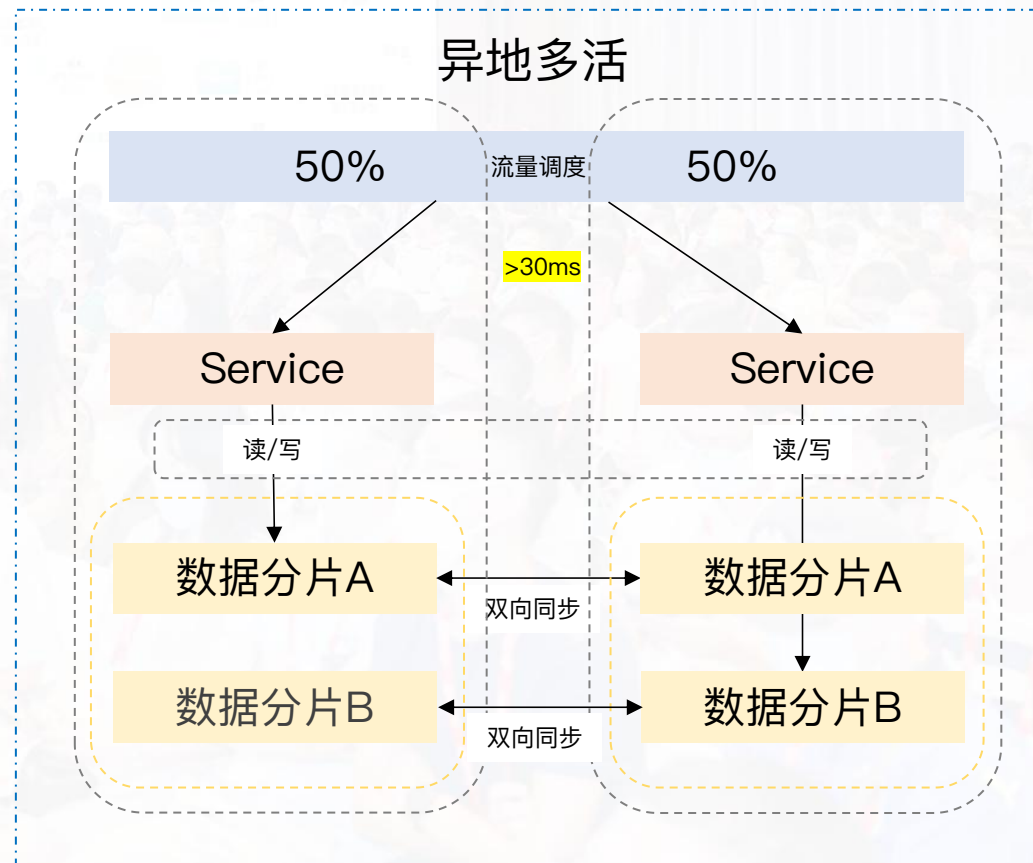
异地多活：业务部署在相对远端城市的数据中心，数据中心间有专线，但网络延迟通常会 $>30\text{ms}$ 。需要按用户分片划分流量，数据层具备单元分片能力，数据进行双向同步；通常也称为单元化架构。

### 主要优势：

- 可以抵抗城市或区域级别的灾难影响；
- 易于扩展，可以解决单城市/机房容量上限问题；
- 业务调用单元内封闭，少量跨机房调用，保障耗时和用户体验

### 挑战：

- 数据分片存储、双向同步能力支持
- 架构复杂度高，并非所有业务都适合，强一致业务仍旧依赖全局部署/数据存储，非强一致业务适合做单元化改造





# 同城多活方案

## 流量接入层

- DCDN
  - 南北向流量管控
  - 用户纬度Hash路由
- SLB&GW
  - 南北向流量管控
  - 故障降级重试
- Discovery
  - 东西向流量管控

## 缓存层

- 缓存一致性维护
- 纯缓存场景改造

## 消息层

- Topic Sync消息双向同步
- Global/Local/None消费模式

## 数据访问层

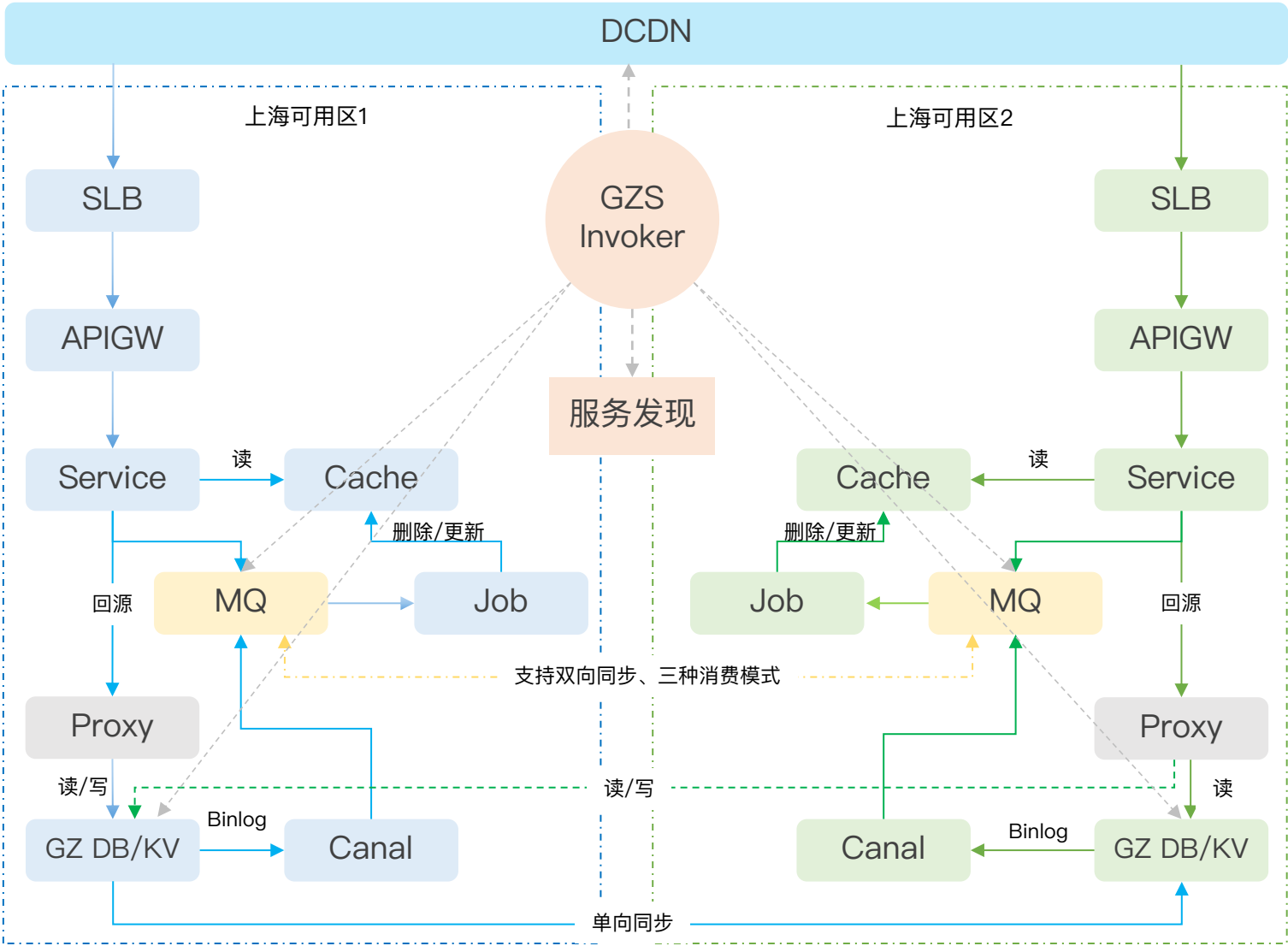
- Proxy读写路由、容灾切换
- 支持MySQL/TiDB/KV

## 数据存储层

- 主从单向同步、双向同步
- 高可用容灾切换

## 多活管控

- 多活资源可视化
- 多活定义、编排
- 切流预检可观测



# 流量接入层

南北向

东西向

## DCDN

- 基于边缘CDN做路由管控，动态最佳选路
- 自研Picker Hash模块，用户ID、设备ID等信息路由
- 支持多机房流量权重灵活调整

## SLB

- 当单可用区SLB故障，支持CDN从其他可用可用区回源
- 支持发现多可用区服务/APIGW节点
- 支持API降级、限流

## APIGW

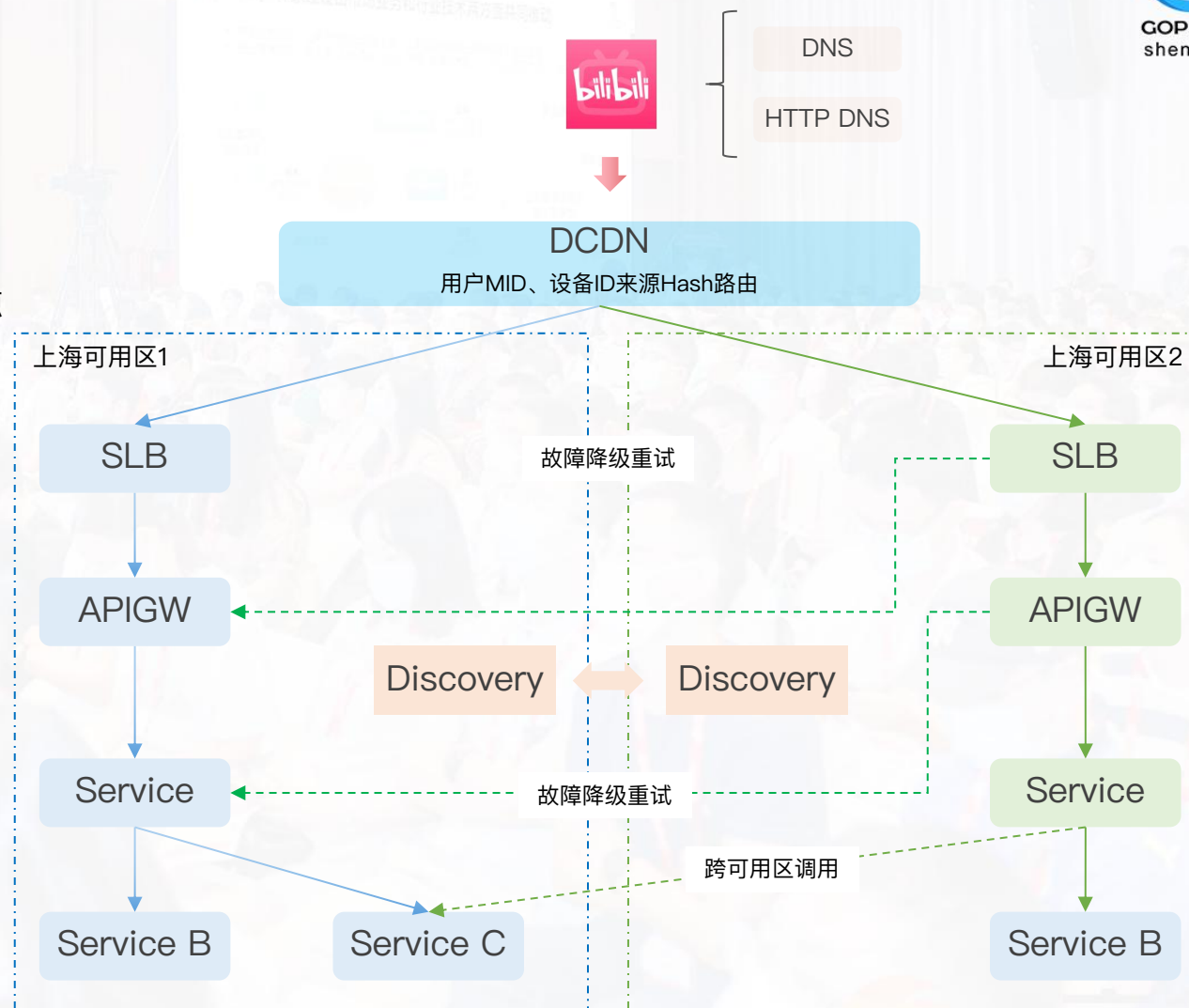
- 发现多可用区服务节点，单可用区服务故障重试
- 支持服务API降级、熔断、限流、客户端流控
- PaaS平台部署，支持HPA弹性扩容

## Service

- 服务本地完整支持读写请求
- 强依赖服务本地部署
- 弱依赖跨专线回主机房

## Discovery

- 同可用区调用优化
- 非多活服务回主可用区调用
- 东西向流量权重管理、灰度等



# 缓存一致性

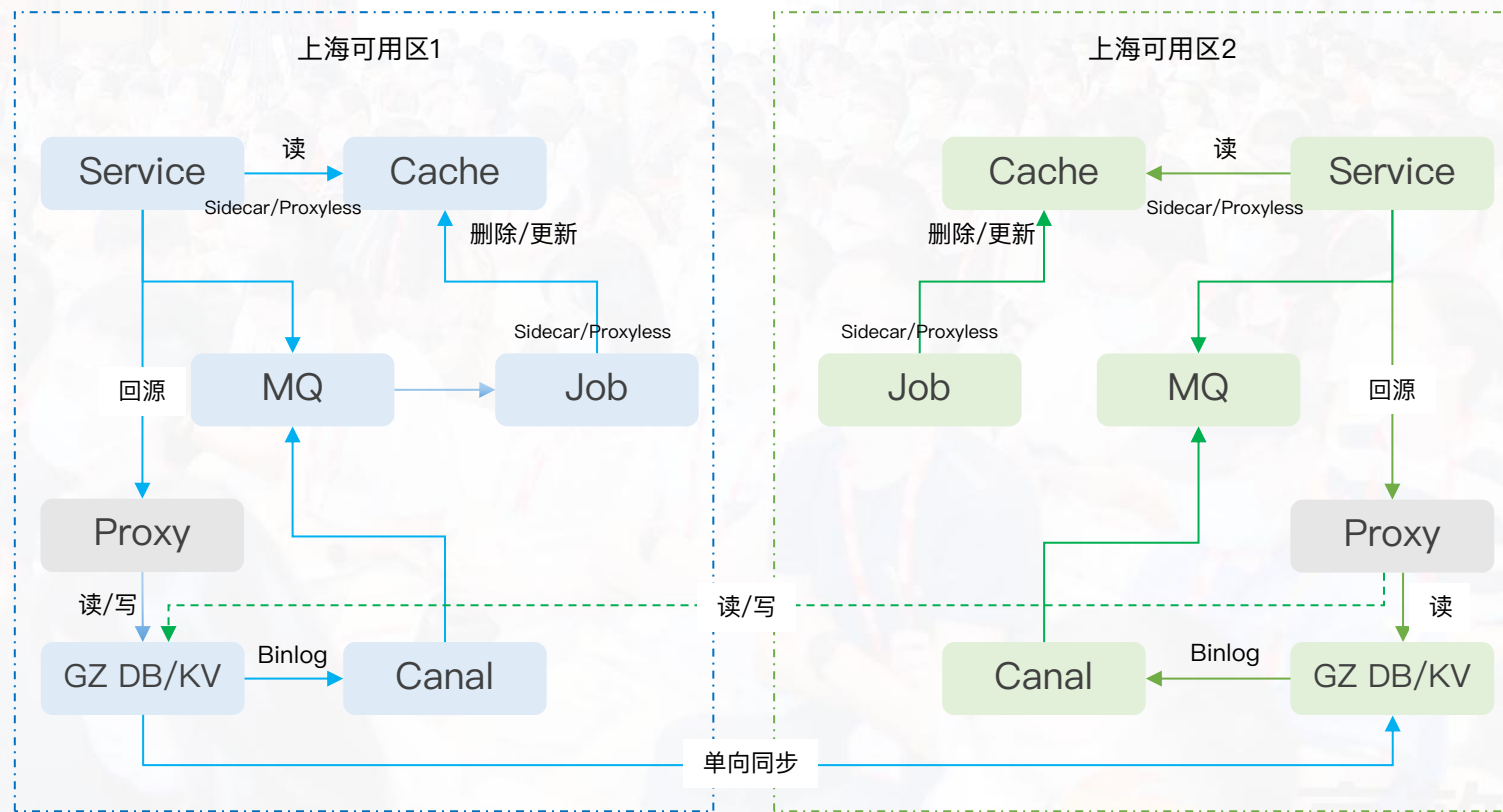
缓存不支持多机房同步，多活场景下缓存独立部署；服务需要订阅同机房数据源，进行数据更新/删除，维护缓存数据最终一致

## 缓存相关中间件支持：

- 统一Proxy，Sidecar/Proxyless SDK模式部署
- 支持Redis、Memcache

## 缓存一致性处理方式：

- **DB/KV+Cache**
  - Canal订阅同可用区存储Binlog，投递消息队列，由业务Job解析处理后删除/更新缓存；
- **纯缓存场景**
  - 热数据：多可用区独立缓存，由Job定时刷新
  - 存储：改造为Cache Aside模式或KV进行存储
- **分布式锁**
  - 使用中心缓存集群
  - 改造为KV，支持CAS、持久化、跨可用区同步

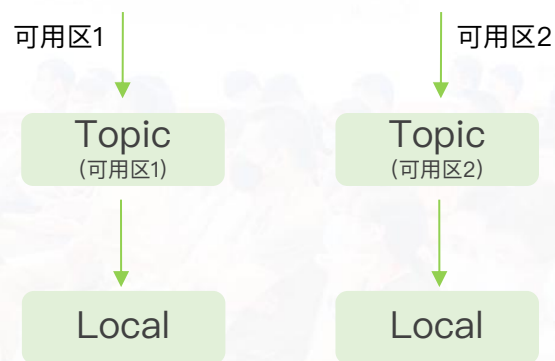




# 消息多活

遵循单可用区内自产自消，支持多可用区间消息双向同步，根据业务场景下游三种消费处理模式

## 各可用区自产自销 (Local)

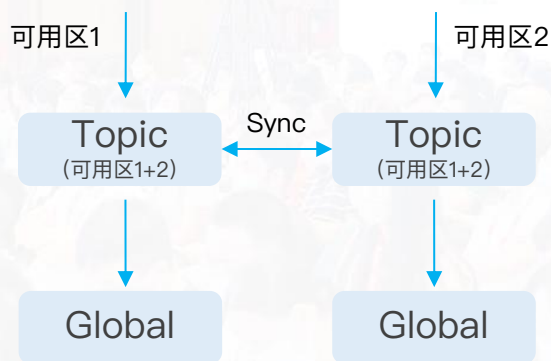


- 可用区之间不进行任何数据同步
- 各可用区消费者消费当前可用区Topic全量数据

### 适用场景:

- Service至Job异步消息
- ServiceA投递给已多活下游ServiceB

## 多可用区全量消费 (Global)

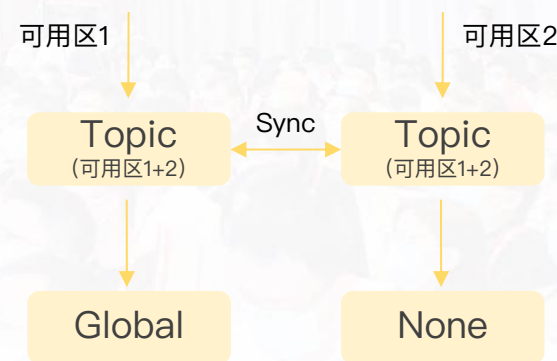


- 可用区之间数据双向同步
- 各可用区消费者消费当前可用区Topic全量 (可用区1+2) 数据

### 适用场景:

- ServiceA投递给未多活下游ServiceB

## 全量消费 (Global) 自定义不消费可用区



- 可用区之间数据双向同步
- 消费者可以消费当前可用区Topic全量 (可用区1+2) 数据; 支持选择单可用区不消费 (None)

### 适用场景:

- 消息由解析Binlog触发的全量数据, 同城双活可用区2不消费

# 数据访问/存储层

## 存储相关中间件支持:

- Proxy提供多可用区路由, 并具备动态切换能力;
- 支持MySQL、TiDB、Taishan(KV);
- 流量切换禁写

## 同城多活 (GZone): 数据全局, 主从读写分离, 强一致场景读主

### MySQL/TiDB 强一致场景:

- SQL Hint级别: select /\*master\*/ name from table where id < 10 ;

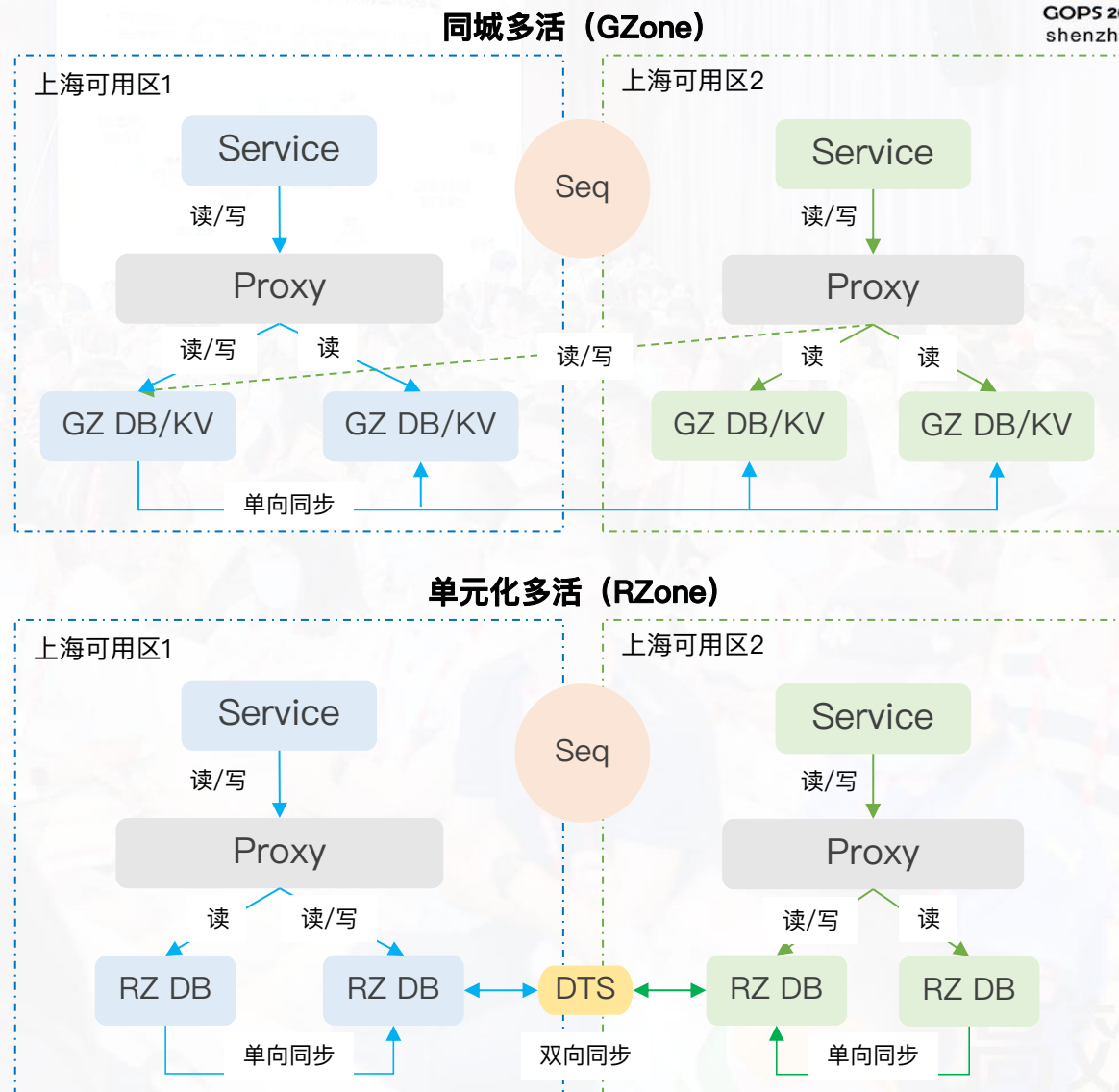
### Taishan(KV)强制读主场景:

- Table粒度读写路由固定主表

## 单元化多活 (RZone): 数据分片存储, 双向同步、冲突处理

### DTS同步组件:

- 实时双向数据复制, 延迟小于10s
- 支持数据冲突检测, 支持暂停同步、发送通知、覆盖等策略
- 支持回环规避、主动限流、数据校验、数据过滤等



02

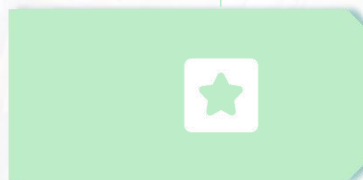
# 多活最佳实践

# 多活推进模式

1、业务分级：业务访问量、用户体验及营收影响

2、区分核心/非核心功能，与产研商定多活改造的场景

## 确定业务场景



## 接口链路梳理

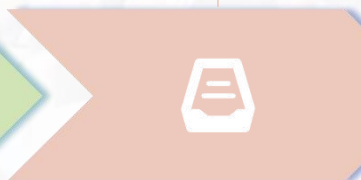
1、场景涉及业务流程梳理

2、涉及服务、接口梳理

1、依赖应用、接口，强弱依赖划分、多活情况

2、依赖中间件、组件梳理

## 接口依赖梳理



## 数据存储梳理

1、数据分类、一致性要求

2、数据可丢失性、可恢复性

3、涉及存储、缓存使用方式

1、涉及的消息队列梳理

2、消息生产/消费者梳理

3、下游处理消息方式

## 消息场景梳理



## 改造实施上线

1、业务架构、中间件改造方案

2、多活管控接入、多活验证方案

3、多活切量执行预案



# 同城多活实践

## 单活

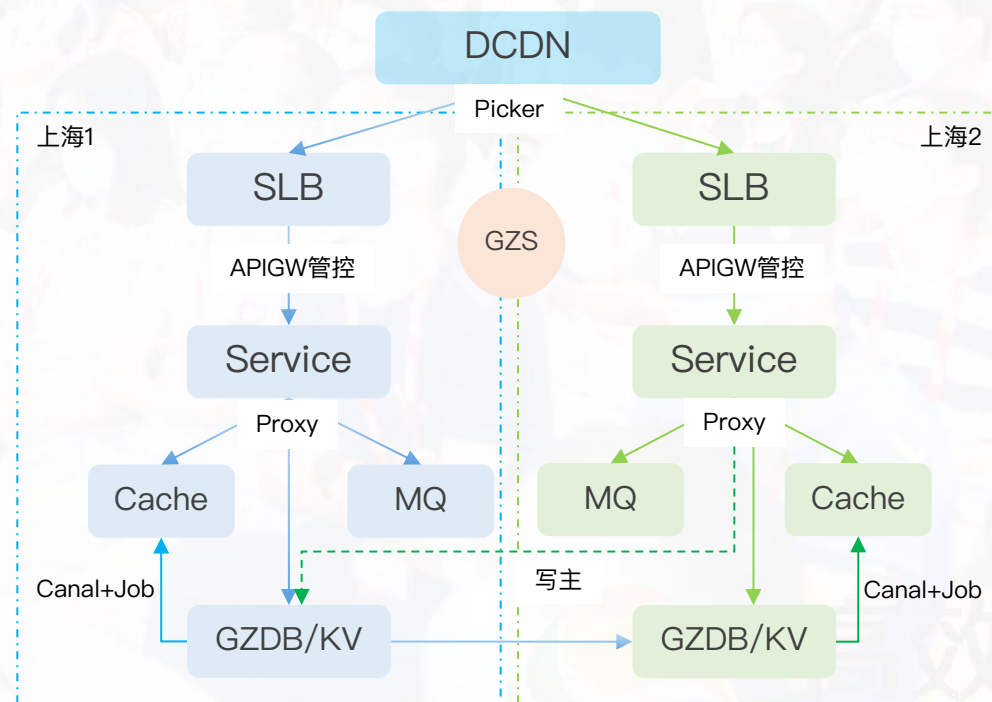
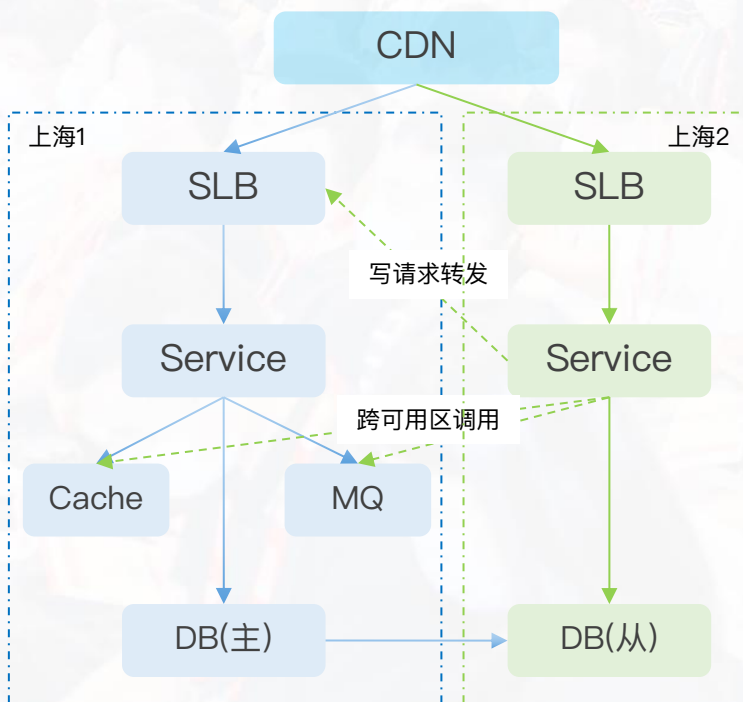
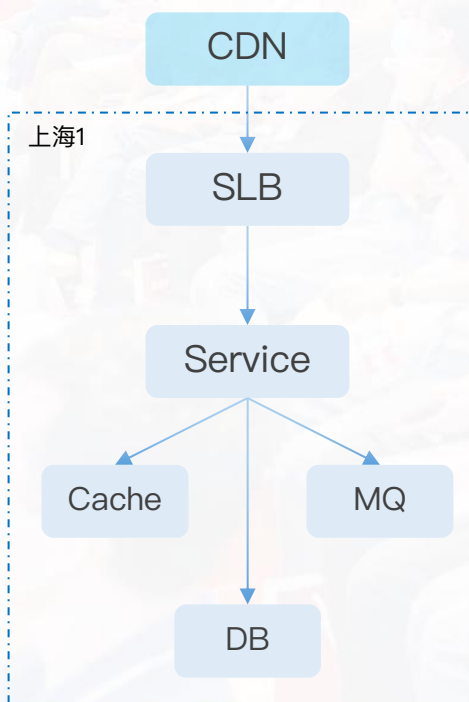
- 服务仅在单个可用区部署
- 服务直连/域名接入存储
- SLB+CDN配置规则发布

## 读多活

- 服务在同城可用区部署、前端多活
- 读请求支持多活，写请求转发
- 部分Cache/MQ组件跨机房调用

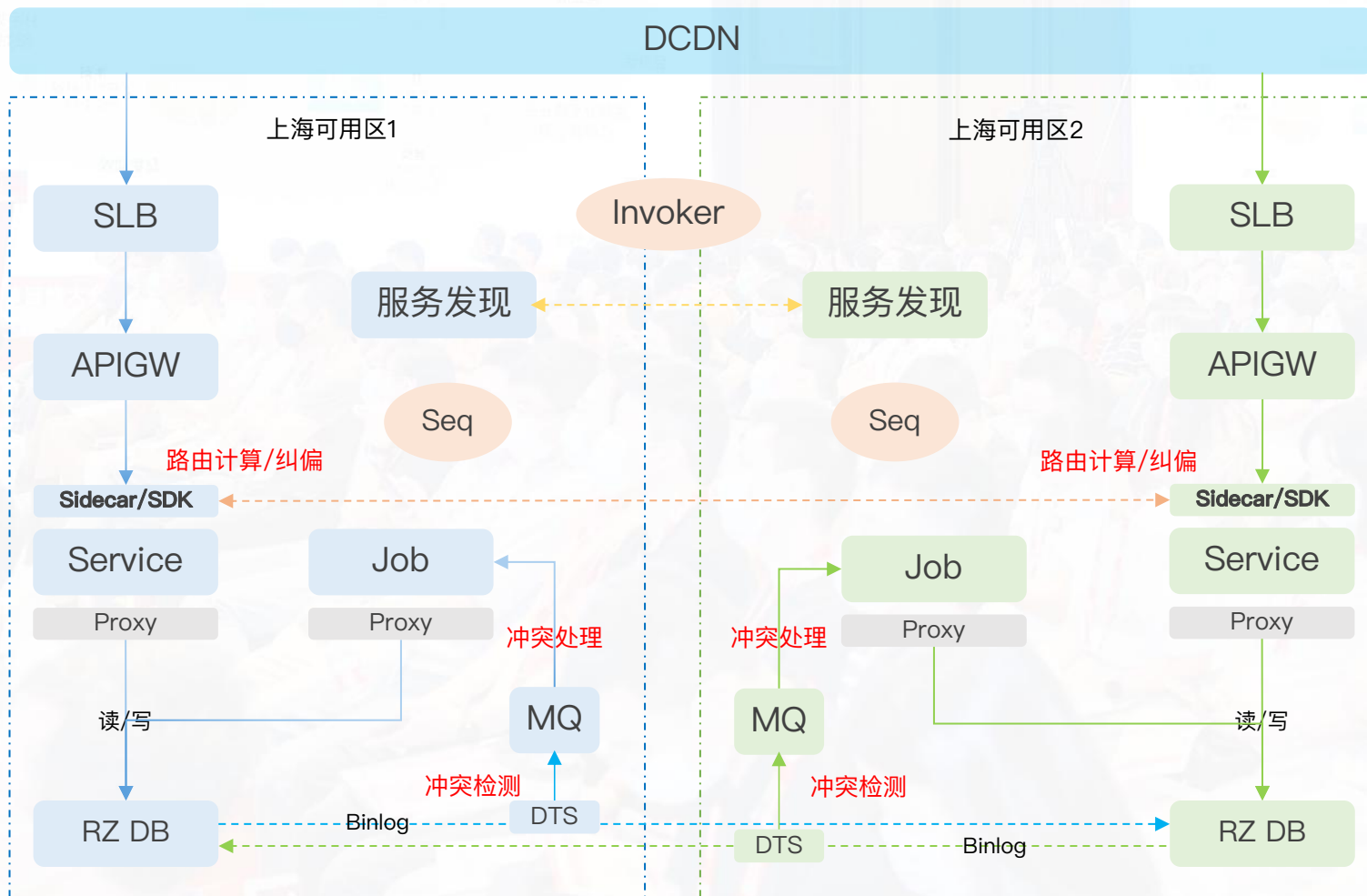
## 同城多活

- 服务接入Proxy，支持处理写请求；支持容灾自动切换
- 缓存一致性由Canal+Job维护；消息生产/消费可用区内闭环
- 服务接口发布由GW管理，Invoker统一发布至DCDN





# 单元化多活实践



## • 存储表结构改造

- 增加分片字段、校验时间字段
- 写入DB加入分片ID，拆分多行，各可用区只写本可用区的数据

## • 分布式ID生成器

- 自增、雪花，支持服务端+本地SDK

## • 流量路由

- Sidecar/SDK流量代理
- 业务维度路由分片计算

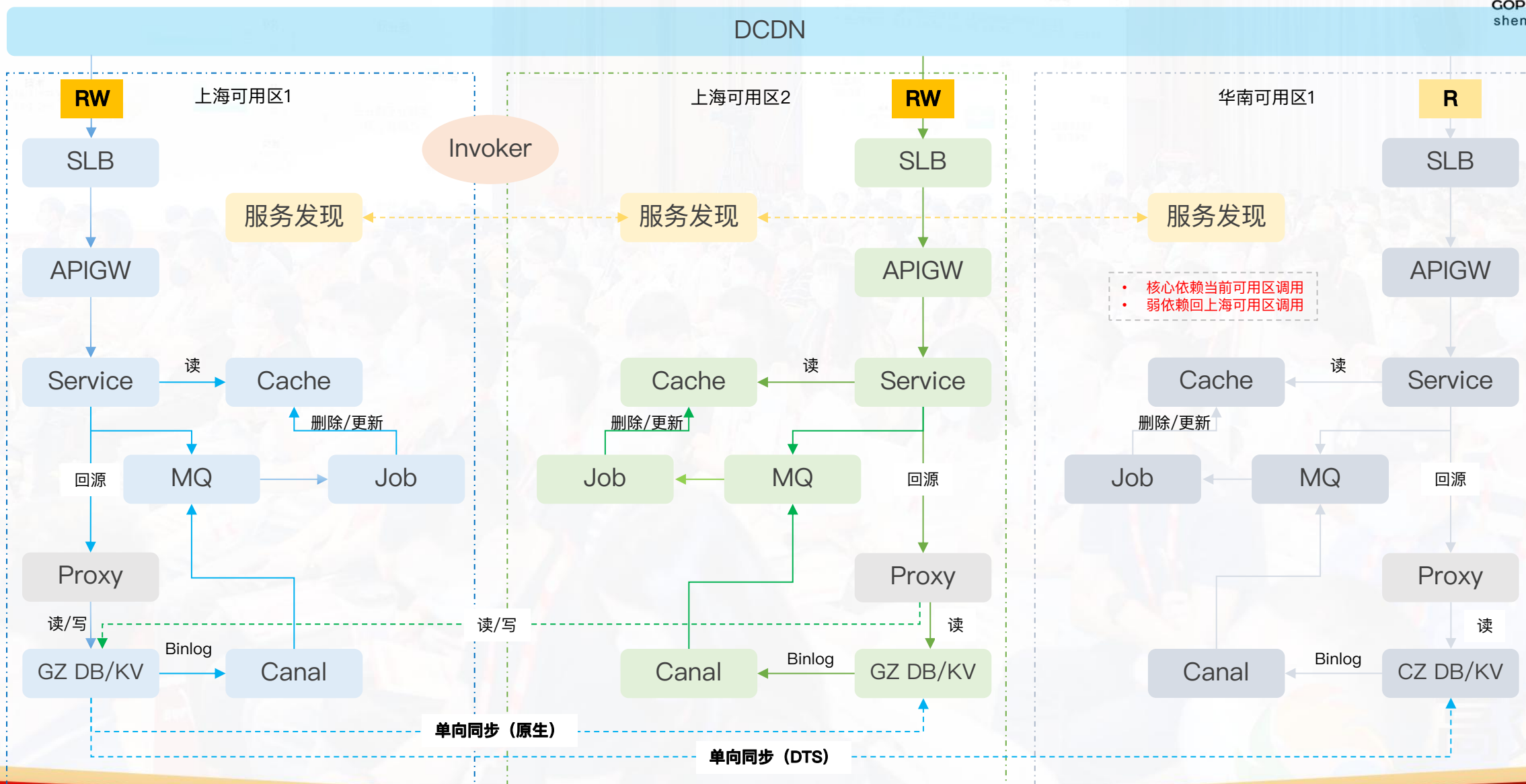
## • DTS双向同步冲突检测

- 校验时间字段冲突判断
- 根据需要执行投递冲突消息/覆盖/暂停

## • 缓存、消息计数场景改造

- 单元独立部署
- 全部分片数据合并计算

# 异地多活实践



03

# 多活管控治理

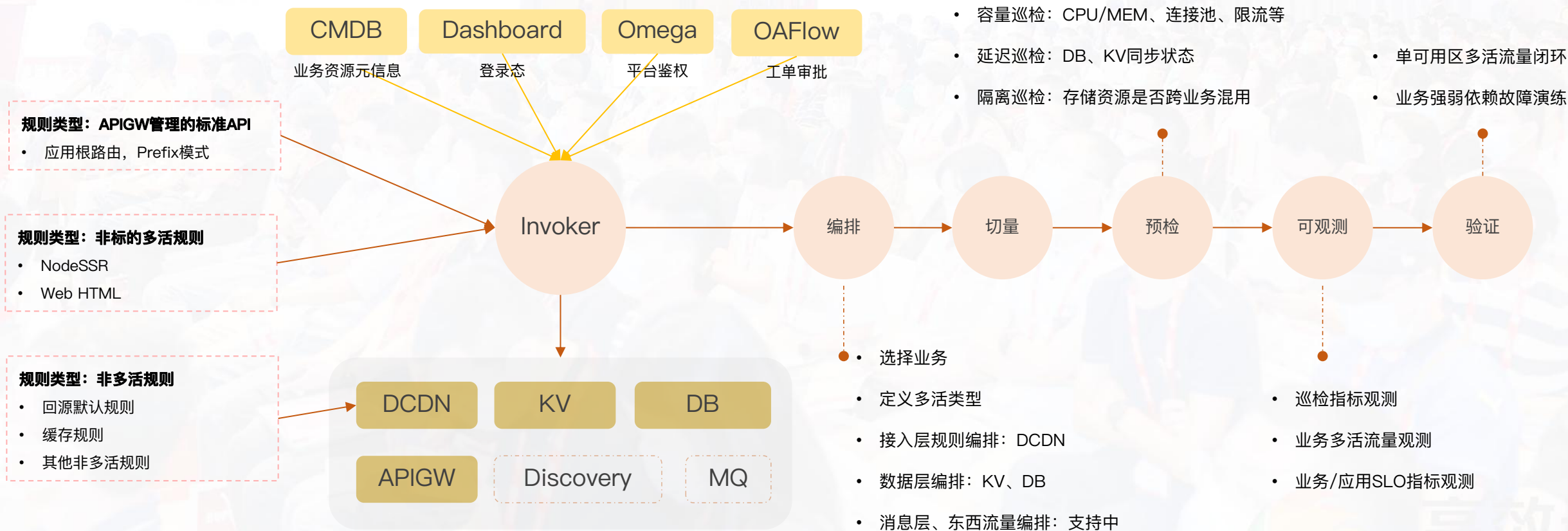
# 多活流量管控

多活元信息规则治理

Invoker平台多活&强依赖降级&演练

多活编排、接入流程化

预检、可观测、验证能力加强





# 多活定义编排

## 业务范围

- 业务，支持编排
- 业务域，不编排

## 多活类型

- CZone/GZone/RZone
- 地域、可用区选择

## 接入层编排

- 多活路由规则
- DCDN 调度模式

## 数据层编排

- 存储：KV、DB
- 存储下游消费任务

## 切量申请

- 数据延迟
- 单机房服务故障
- 基础设施故障等

## 切量编排

- 选择业务对象
- 切量权重、灰度
- 切量对象：DCDN、KV、DB

### 业务定义、多活资源编排



< 业务名称: [redacted]

编辑

所属组织: [redacted]

多活模式: GZONE

状态: 审核通过

地域: 上海

主可用区: [redacted]

同城双活可用区: [redacted]

更新时间: [redacted]

多活比例: [redacted]

自定义规则

应用名称	路径	域名	匹配方式	CDN规则检查	CDN路由方式
[redacted]	[redacted]	a, [redacted]	[redacted]	[redacted]	[redacted]
[redacted]	[redacted]	grpe, [redacted]	[redacted]	[redacted]	[redacted]

\* 任务名称

请输入任务名称

\* 业务名称

请选择业务名称

\* 流量比例

主可用区: 上海(sh0)

50

%

\* 灰度比例(单次发布比例)

请选择灰度比例

\* 切流维度

业务

\* 同城双活可用区: 上海

50

%

\* 同时切换存储

☒ 否 ☐ 是

多活接口规则

应用名称	路径	域名	CDN规则检查
<input type="checkbox"/>			

### 切量申请、切量对象选择





# 切流预检与可视化

## 容量预检

- 应用CPU
- Cache CPU
- DB/KV CPU
- DB连接数

## 延迟预检

- DB同步延迟
- KV同步延迟
- DTS任务延迟

## 限流预检

- 接入层限流
- 微服务框架限流
- 强依赖下游服务限流

## 流量观测

- 多活规则维度QPS
- 多机房流量调度观测

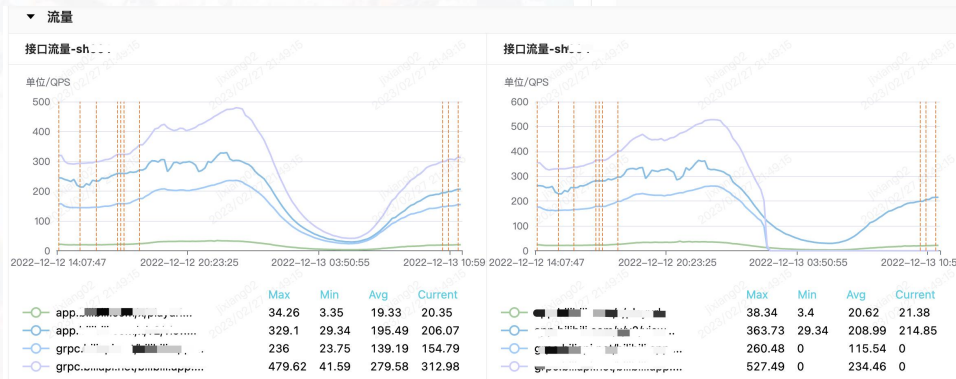
## 巡检观测

- 资源CPU容量
- 连接池容量
- 限流容量

## SLO观测

- 链路依赖
- 链路SLO

### 容量/延迟/限流等预检



### 服务/Cache/DB等容量观测



### 流量/限流观测



# 多活有效性验证

## 依赖展示

- 服务依赖展示
- 组件依赖展示

## 服务/组件依赖可视化



## 依赖检查

- 依赖多活检查
- 强弱依赖标记

## 流量来源/调用检查



## 流量闭环

- 跨可用区调用发现
- 核心依赖可用区闭环
- 弱依赖排期演练

## 故障演练

- 依赖自动发现
- 自动故障演练
- 报告输出确认

## 强弱依赖故障演练



# 多活建设思考

1 同城多活



2 单元化多活 + 异地读多活



3 异地多活



1. 高可用：故障应急时的有效快恢手段
2. 解决机房容量、组件连接池限制，满足业务增长
3. 变更灰度管控能力、重大活动稳定性保障

1. 数据中心、基础设施、硬件资源投入
2. 业务架构改造、基础架构及组件改造成本
3. 架构复杂度带来的资源管理、运维管理成本