

微博ServiceMesh大规模实践

— WeiboMesh

新浪微博 / 丁振凯

2018.12

msup[®]

2019 技术创新年度会议

携手国内外一线专家
打造有深度的实践分享



扫二维码
获得一线实践经验



人工智能与 机器学习创新峰会

集中在人工智能如何切实
帮助业务；指导技术管理者
开展AI技术战略布局

📍 上海 05.26~27



MPD技术管理工作坊

20位知名企业一线带头人
指导的实践沙盘演练；
现场实践的大时段课程

📍 北京 07.20~21
深圳 09.20~21
上海 11.26~27

TOP1

技术型企业案例研究智库

全球软件案例研究峰会

全球范围甄选、盘点
年度100个研发与创新案例

📍 北京 11.27~29

GIAC

全球互联网架构大会

国内最顶级的架构师峰会
70+首架带来的
架构领域的最佳实践

📍 深圳 06.14~15



最佳管理实践 公开课500+

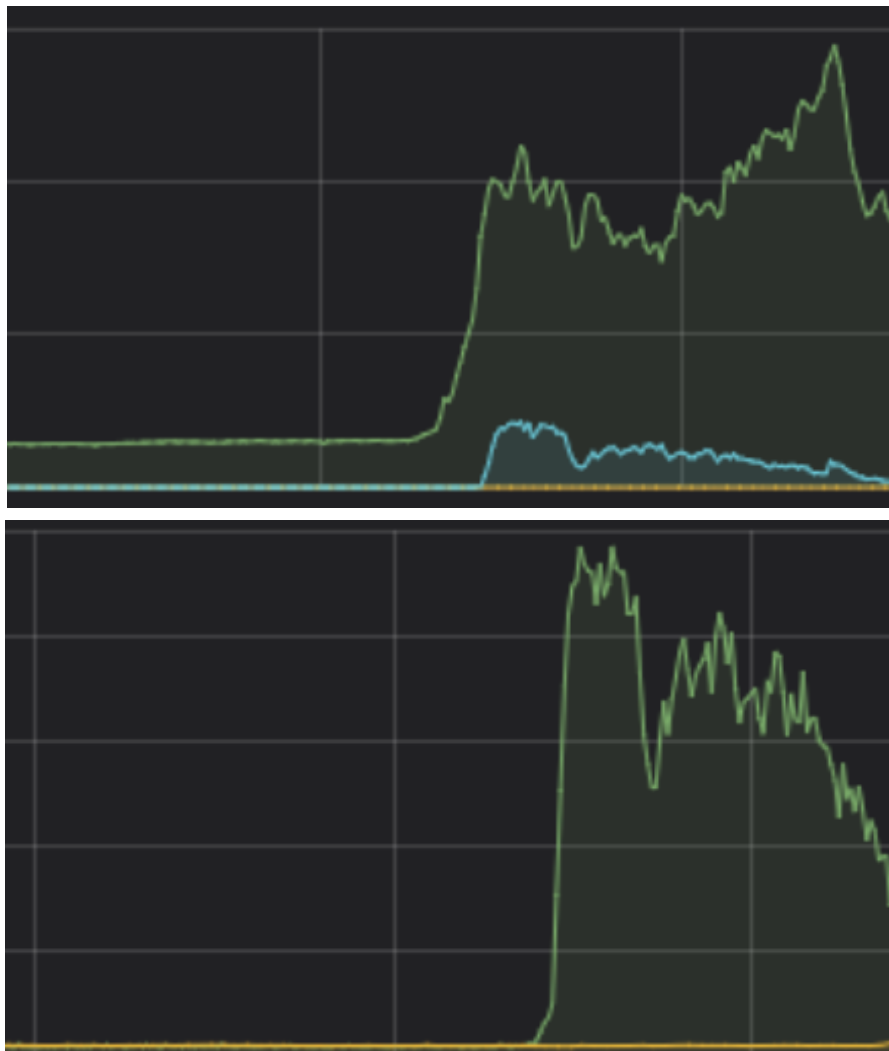
联盟创新公司共建“方法+”智库

内容提要

- 微博服务化挑战
- 服务化新思路
- WeiboMesh方案介绍
- 生产实践
- 总结

1

微博服务化挑战



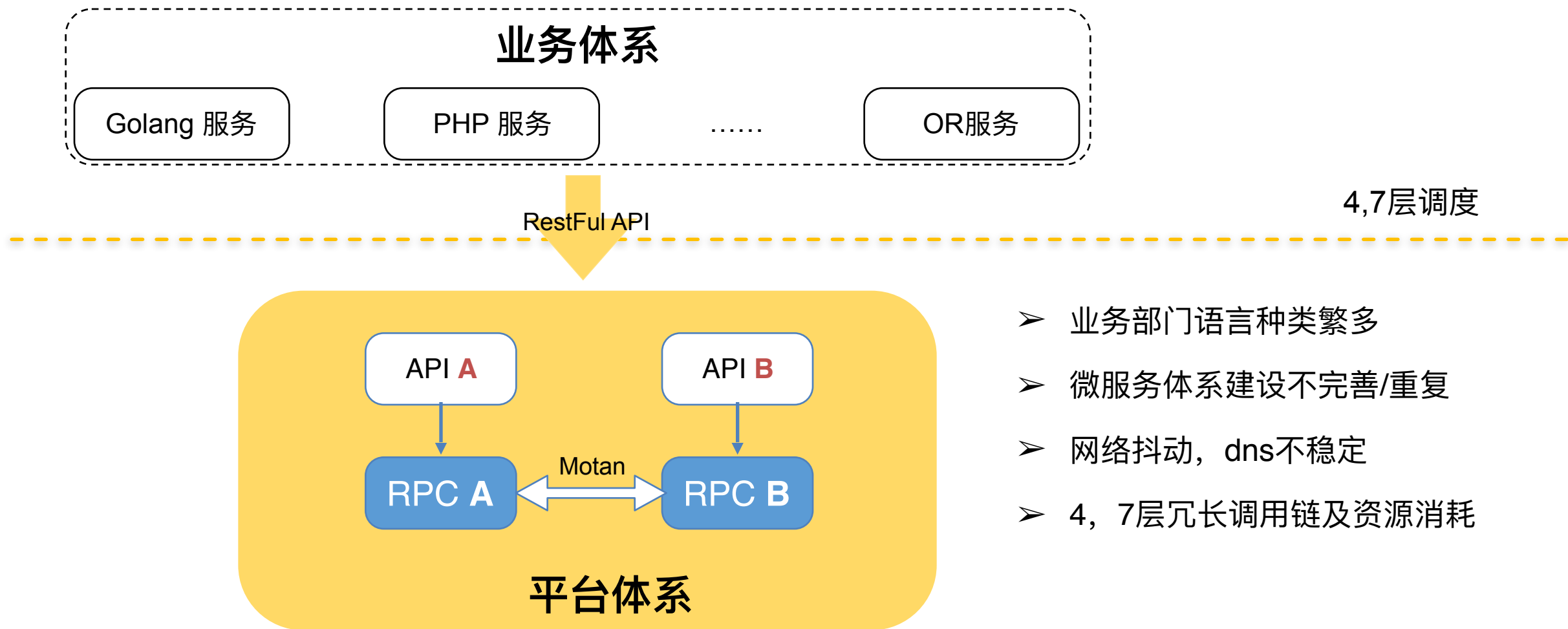
- 热点事件事发突然
- 短时间内流量暴涨
- 极易引发雪崩效应



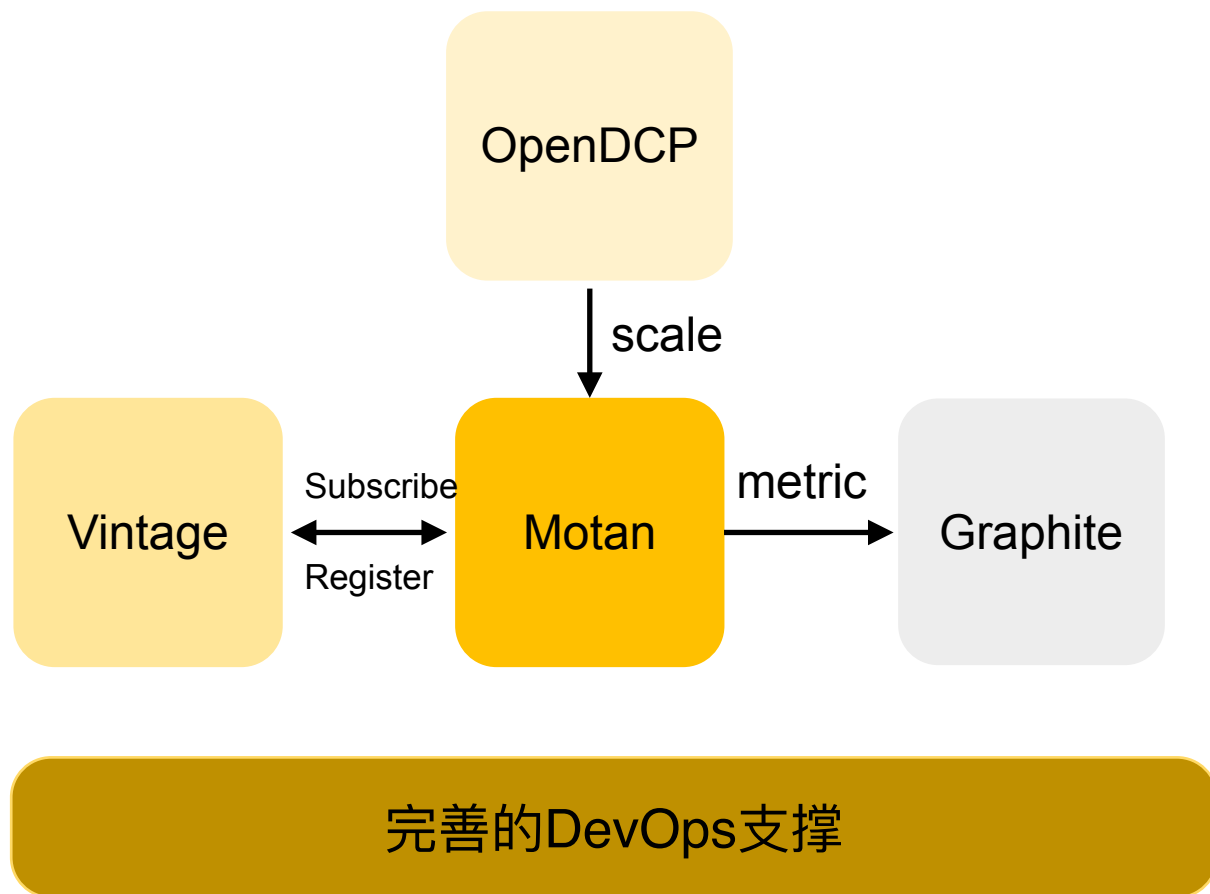
- 服务如何隔离拆分?
- 扩容? 降级?
- 运行现状感知
- 如何评价服务可用性?

服务治理水平直接影响到服务可用性

服务调用链



- 业务部门语言种类繁多
- 微服务体系建设不完善/重复
- 网络抖动, dns不稳定
- 4, 7层冗长调用链及资源消耗



Motan

- 服务治理
- 动态路由

Vintage

- 注册中心

OpenDCP

- 智能弹性调度

Graphite

- 实时统计监控

平台内部微服务体系建设比较完善

趋势



云原生生态

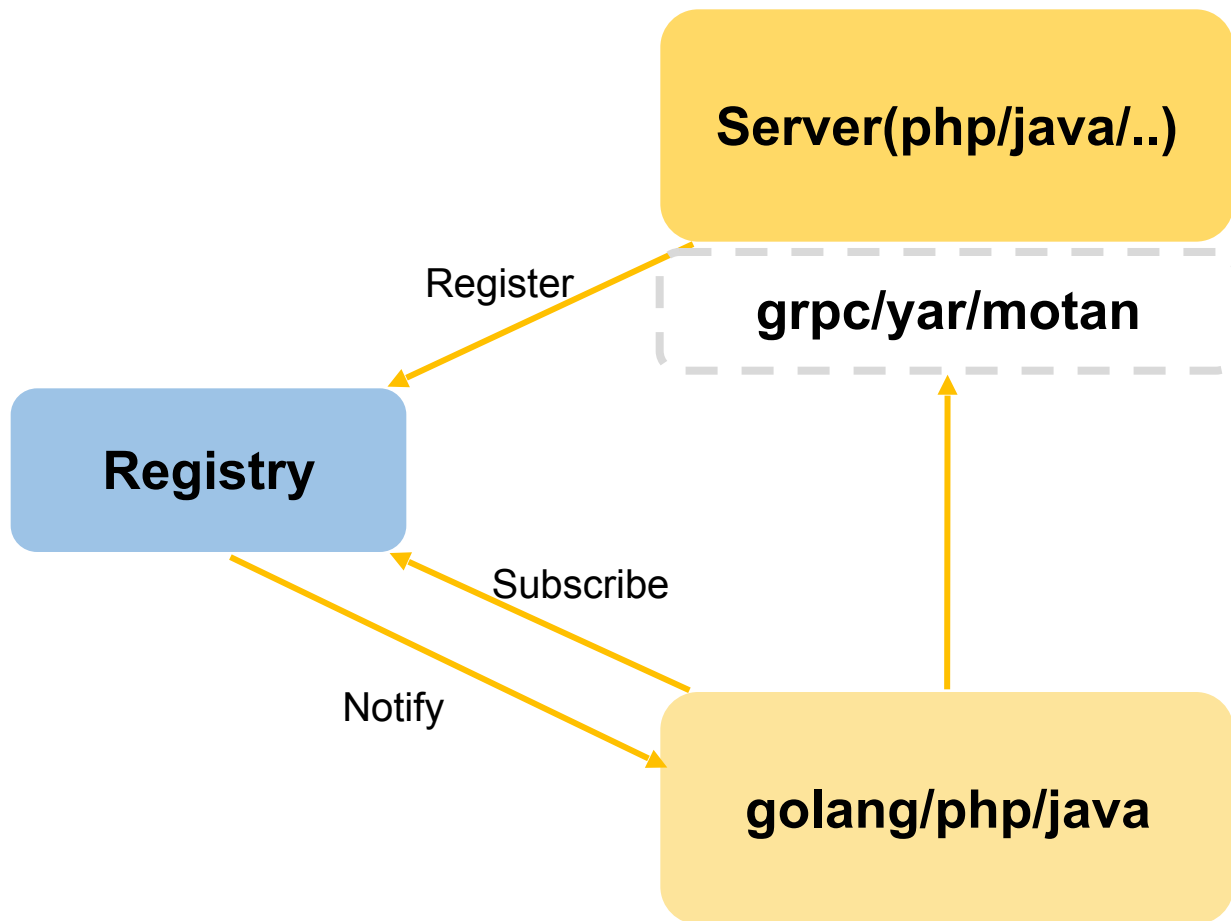
微服务是云原生生态重要的一个技术手段

2

微博服务化新思路

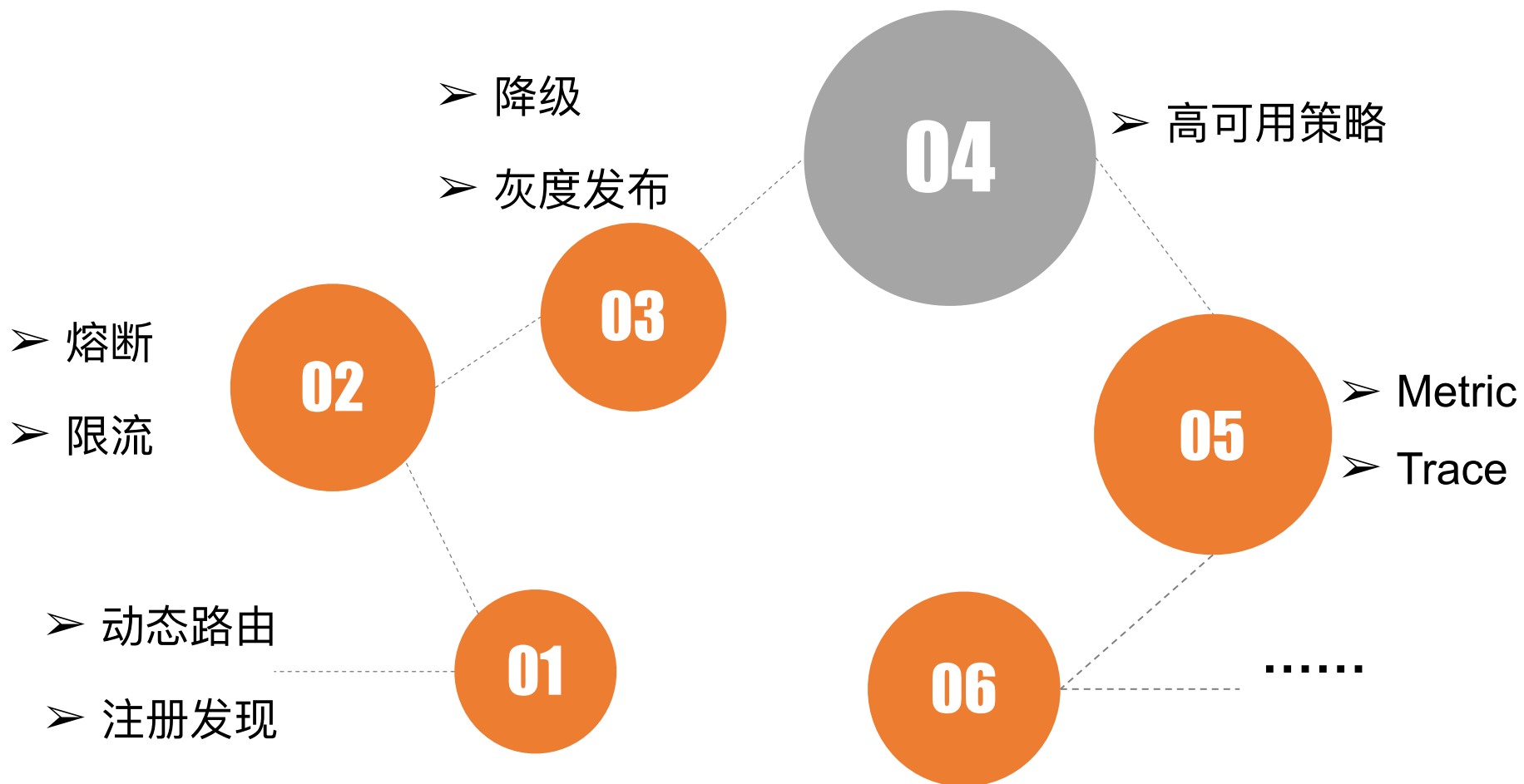
- 跨语言
- 服务治理

跨语言的尝试

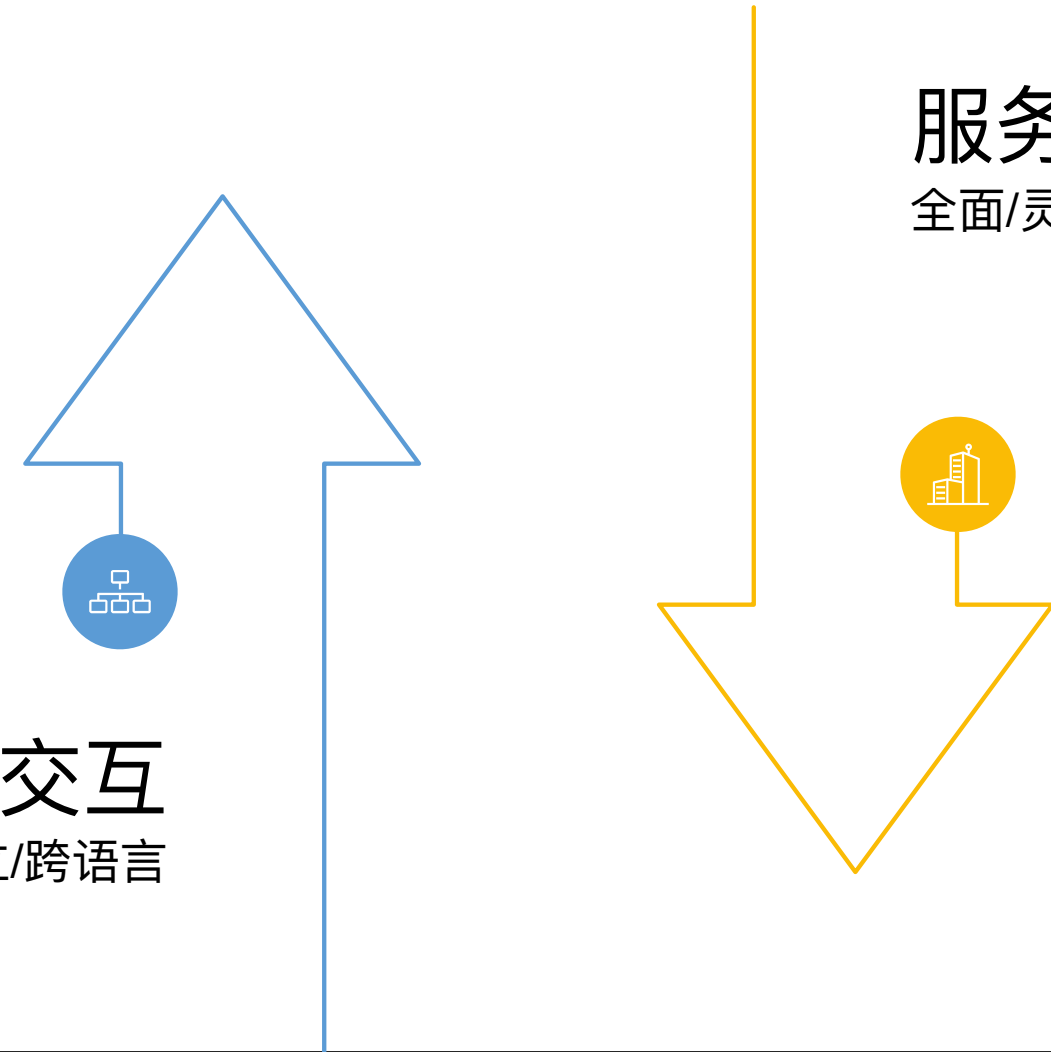


- 语言特性
- 历史积累
- 业务侵入较大，client太重
- 性能
- 扩展性差
- 推广困难

相同的治理功能，不同语言的服务都要做一遍？



跨语言服务化本质



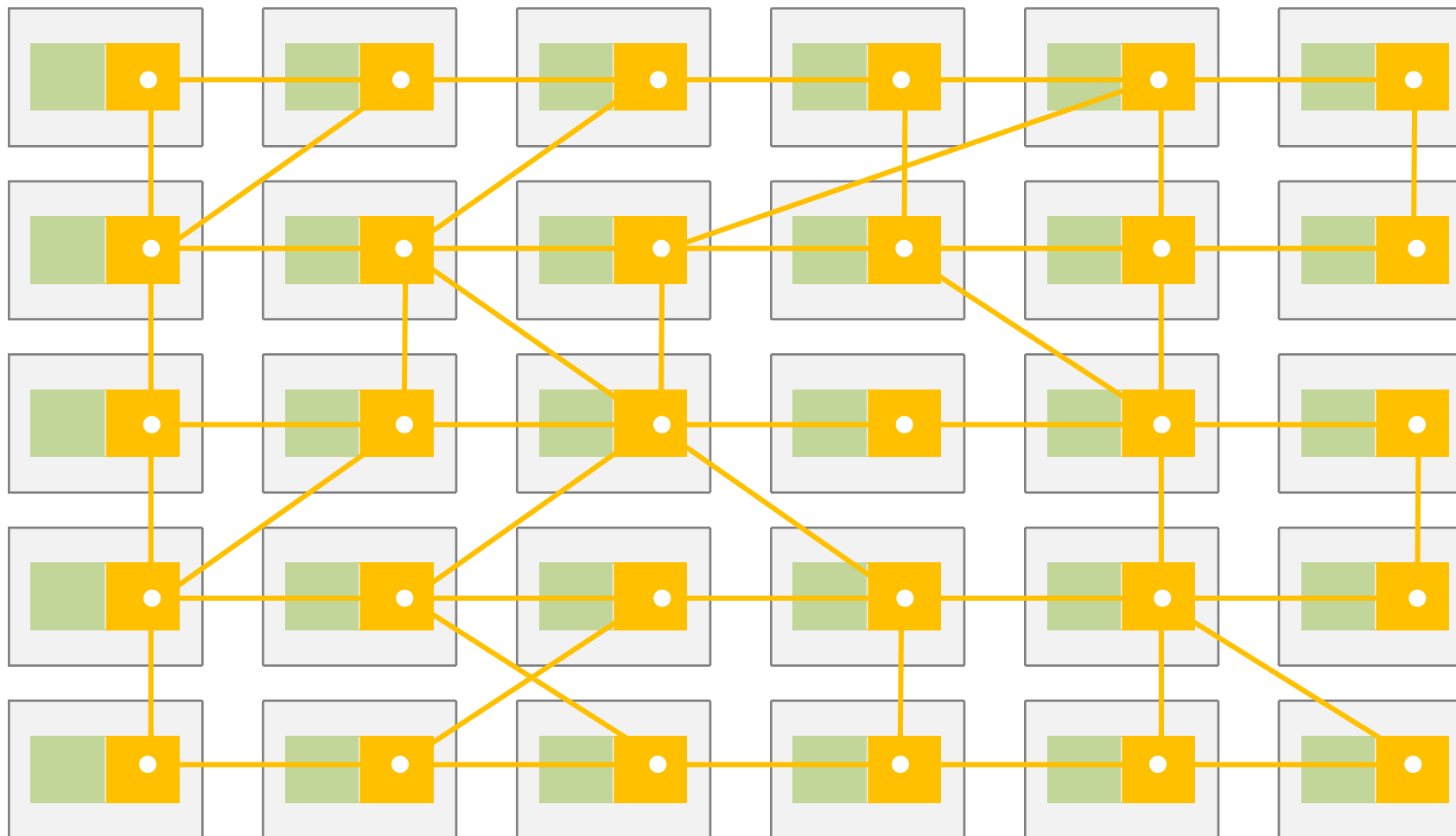
数据交互
协议中立/跨语言

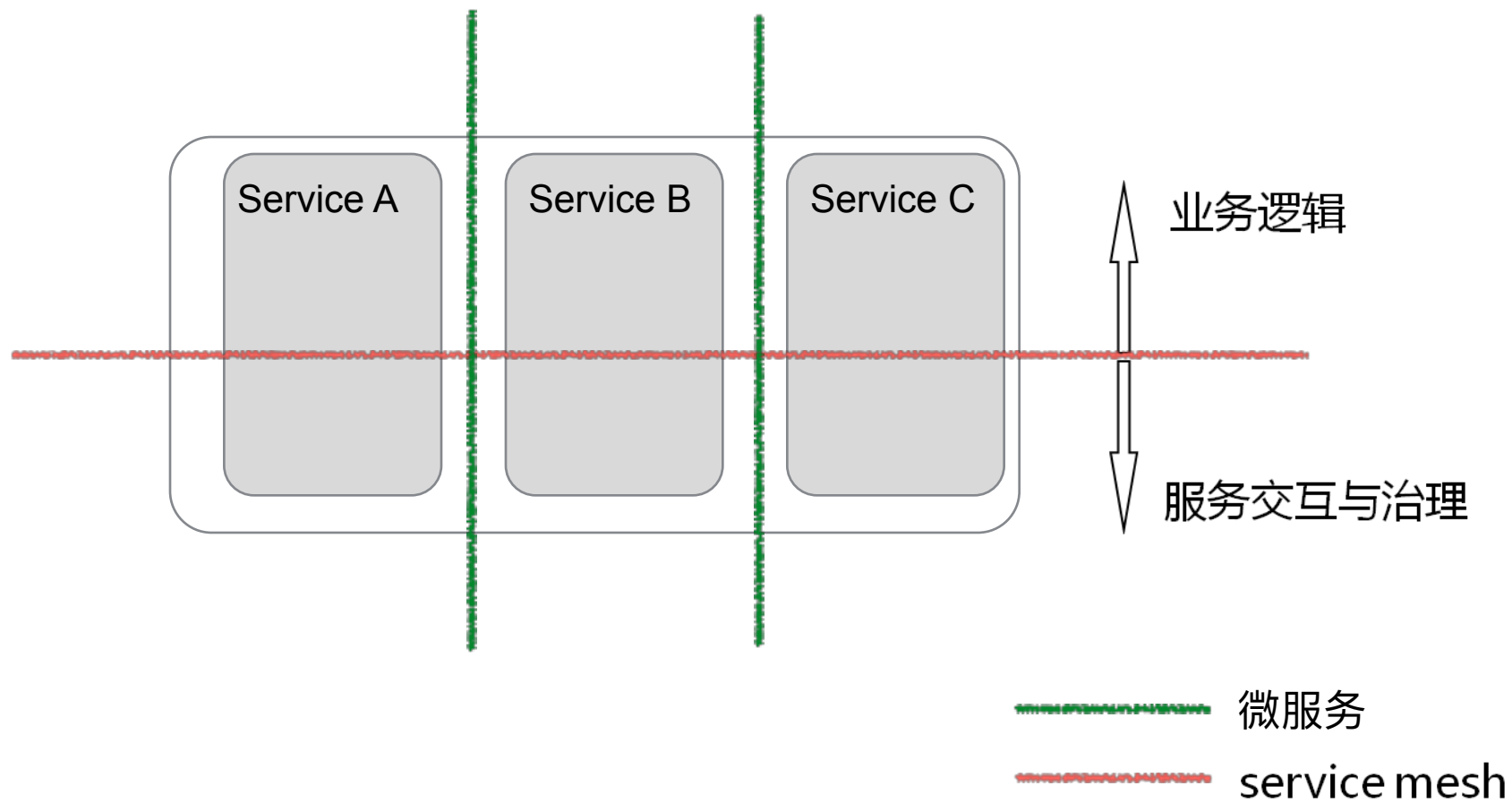
服务治理
全面/灵活可扩展

跨语言服务化方式对比

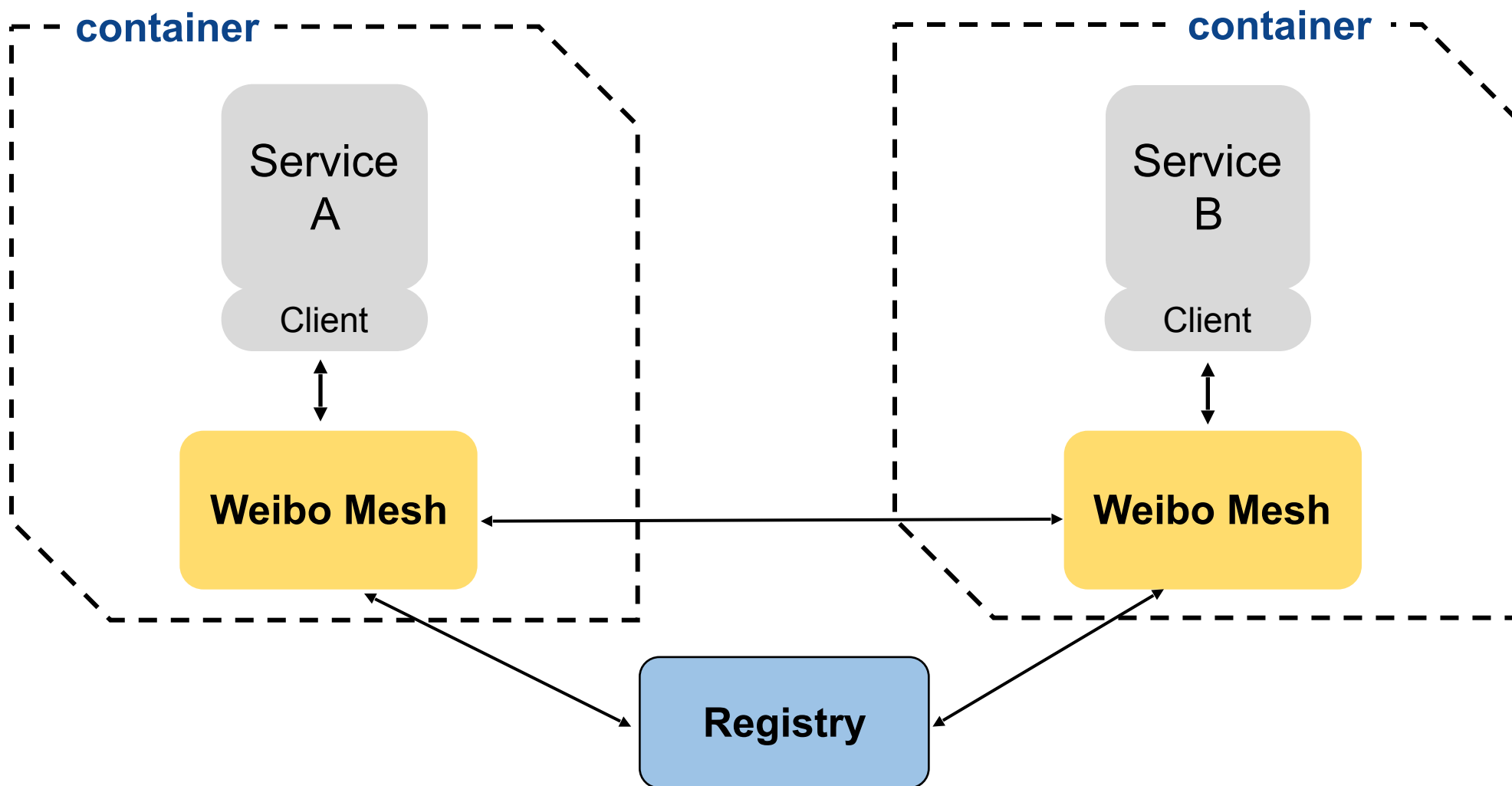
	Http代理	RPC模块	Agent代理
研发成本	低	高	中
维护成本	低	高	中
使用成本	低	低	中
治理功能	中	高	高
扩展能力	低	中	高

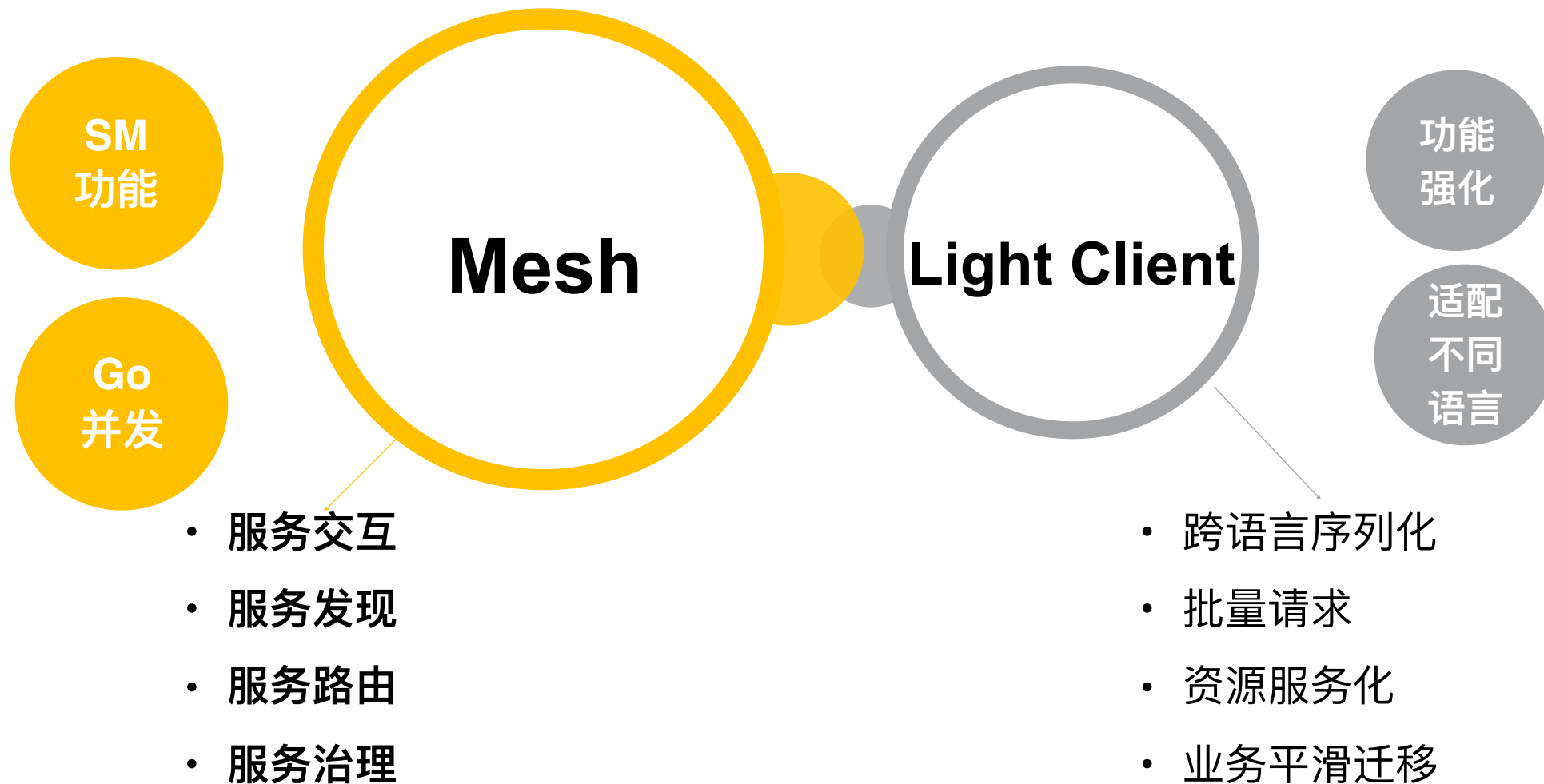
ServiceMesh





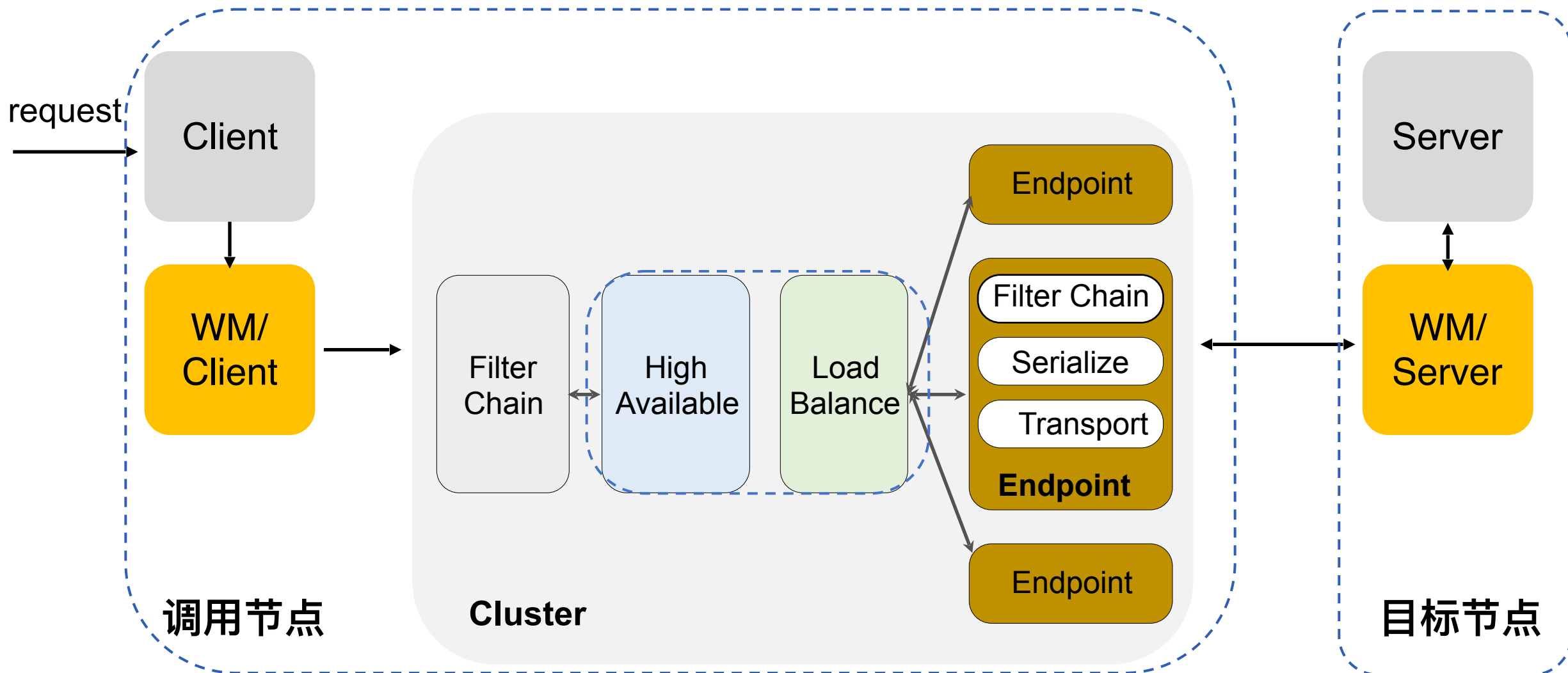
3 | WeiboMesh方案介绍



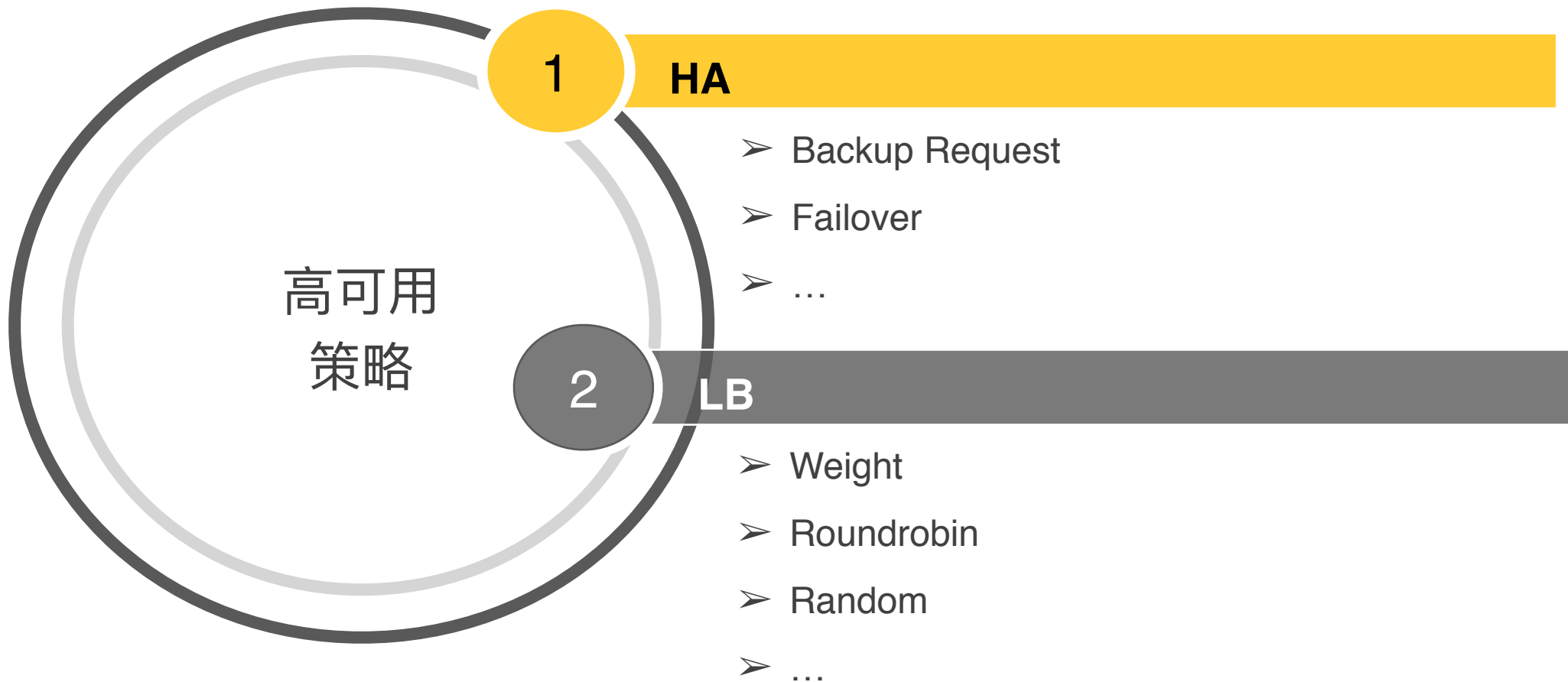


- Cluster（集群管理）
- HA（高可用策略）
- LB（负载均衡）
- Endpoint（服务节点的抽象）
- Protocol（Motan2/传输协议+Simple/序列化协议）

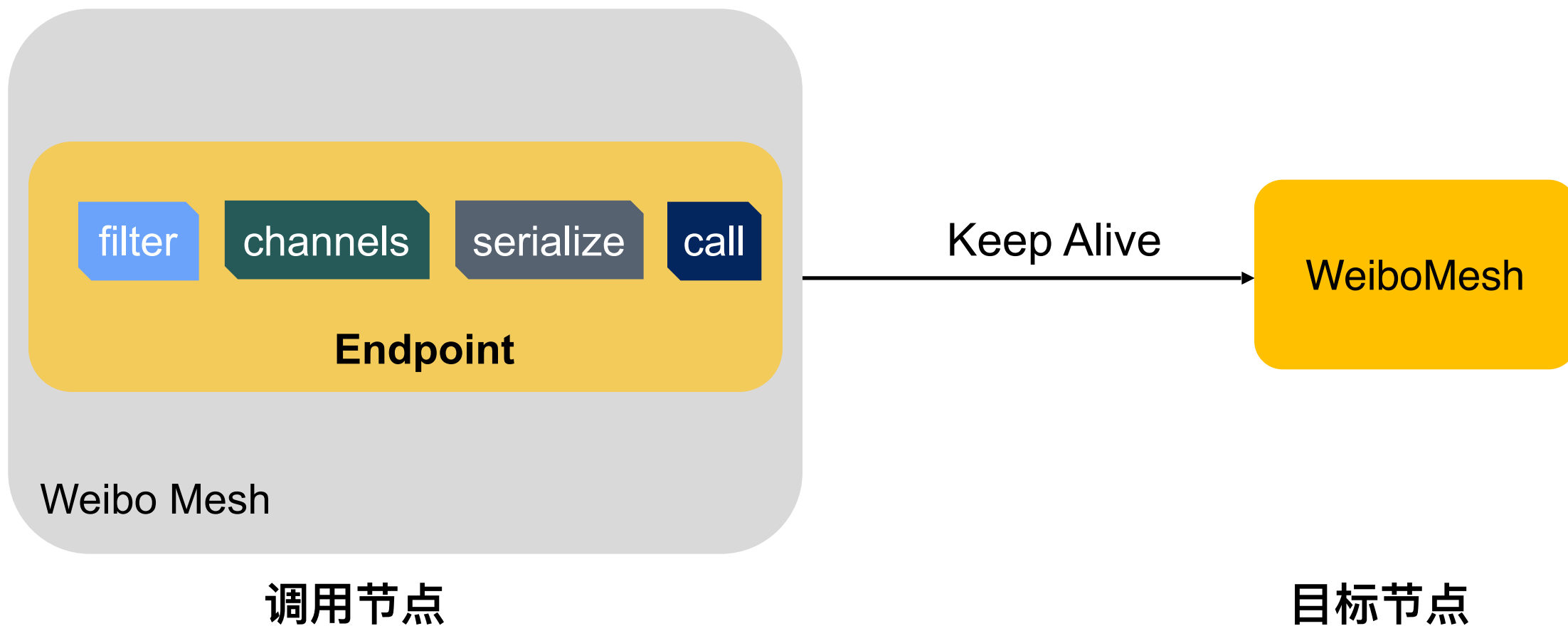
Cluster模块



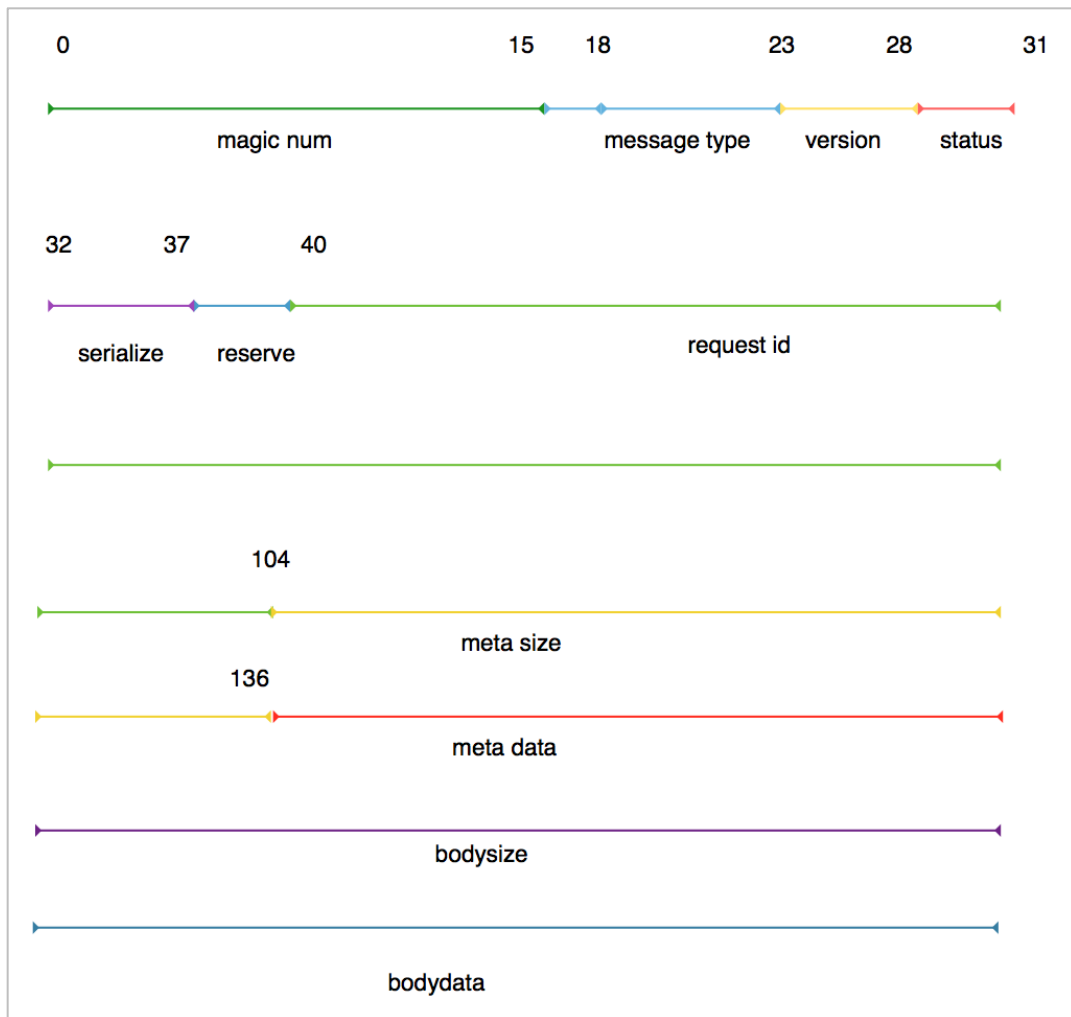
高可用策略



节点抽象



Motan2 传输协议



Header

- 消息类型
- 协议版本
- 序列化协议 (body)

Metadata

- 服务名
- 方法名
- 系统参数及用户参数

Body

- response
- request

Simple 序列化

null
00

string "hello"
01 00 00 00 05 68 65 6c 6c 6f

type(1byte)+size(4byte)+content(\${size} byte)

map {name:ray, code: xxx}
02 00 00 00 1e 00 00 00 04 63 6f 64 65 00 00 00 03 78 78 78 00 00 00 04 6e
61 6d 65 00 00 00 03 72 61 79

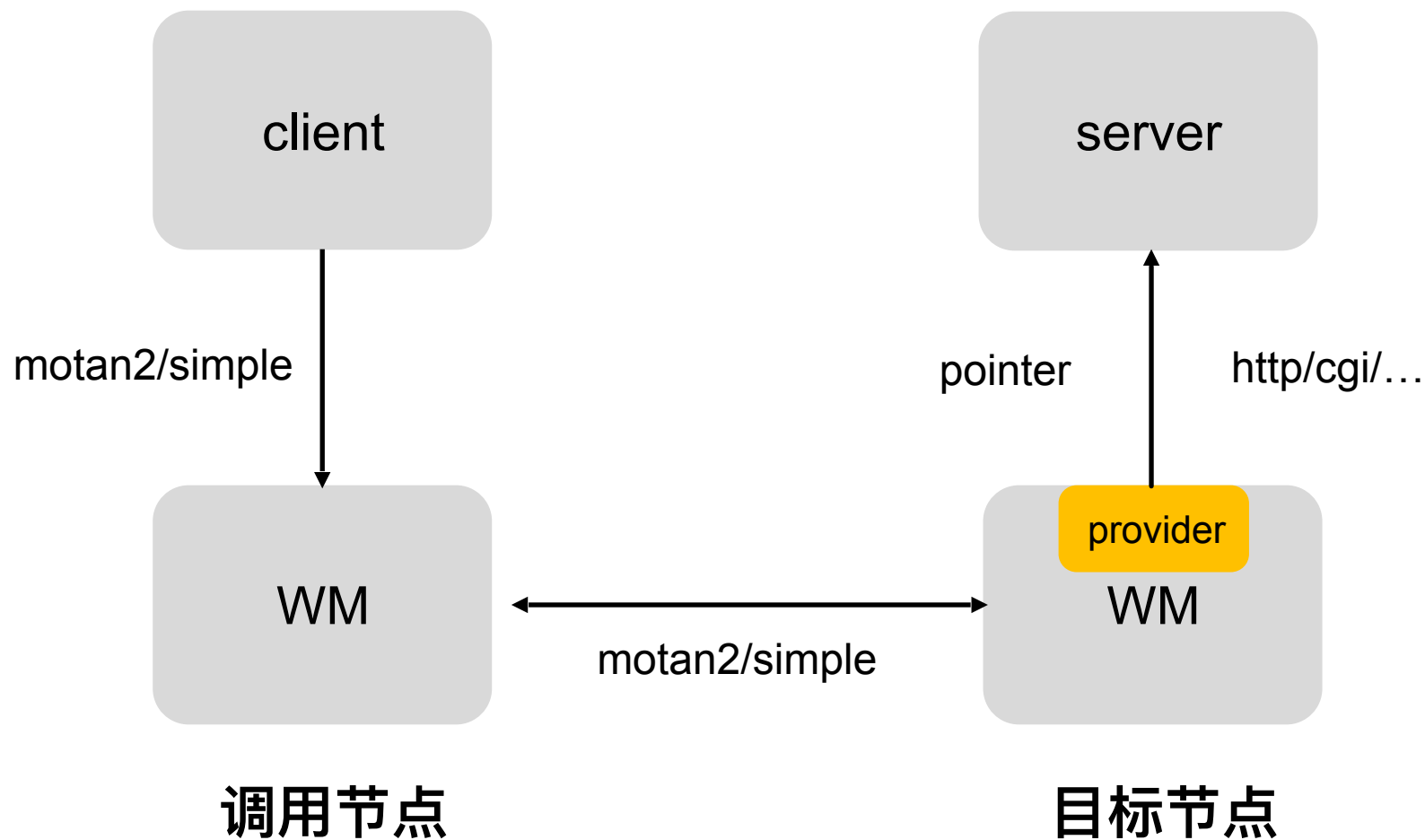
基础类型

组合类型

```
// serialize type
const (
    sNull = iota
    sString
    sStringMap
    sByteArray
    sStringArray
    sBool
    sByte
    sInt16
    sInt32
    sInt64
    sFloat32
    sFloat64

    // [string]interface{}
    sMap    = 20
    sArray  = 21
)
```

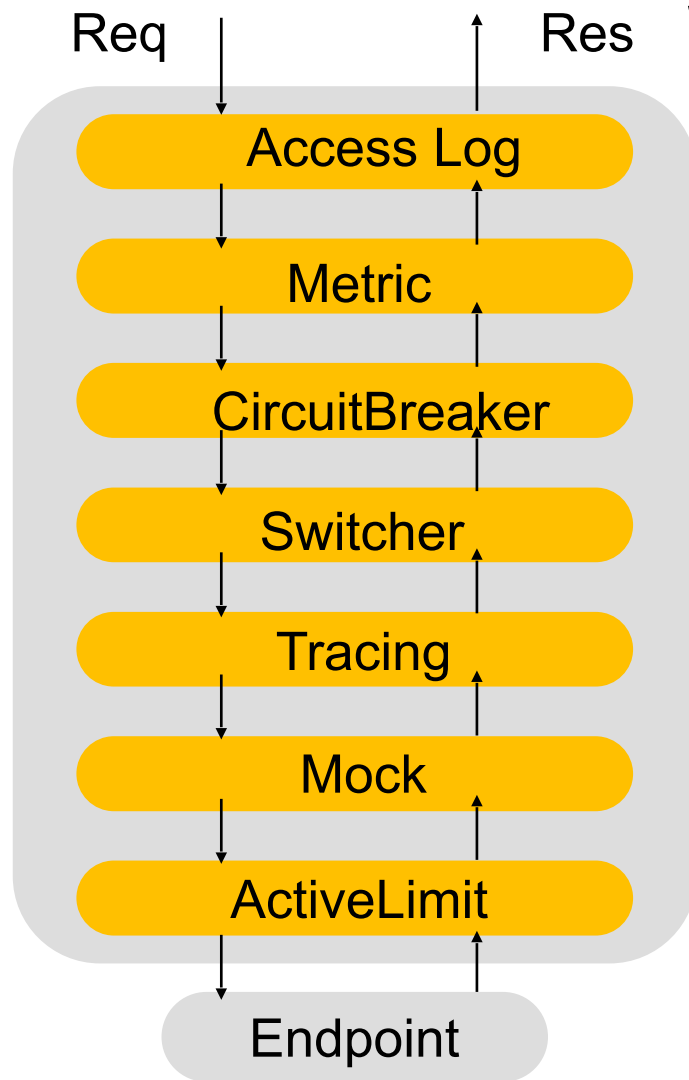
协议转换过程



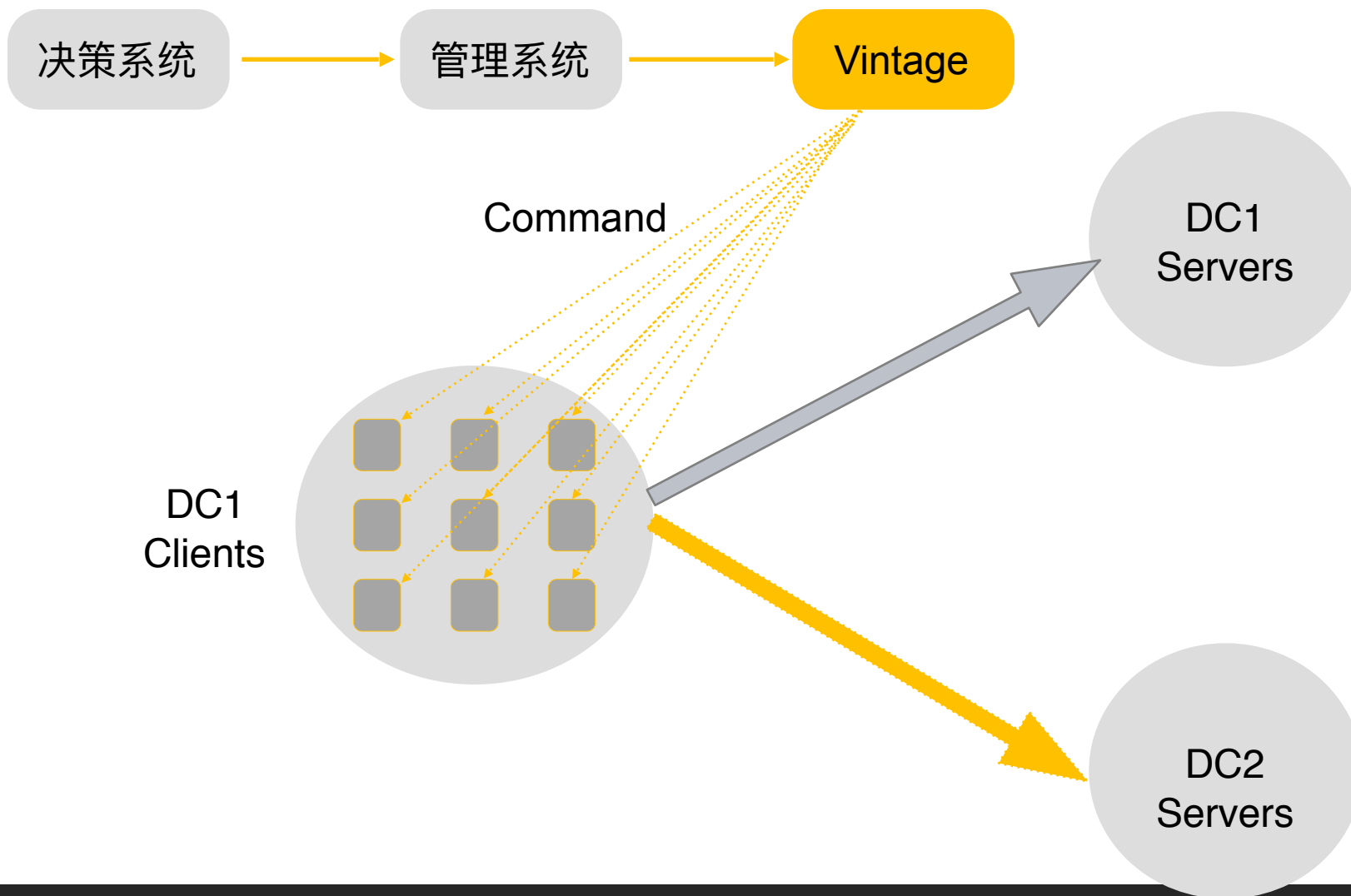
- 策略扩展：Filter Chain
- 流量调度：MCS（Mesh Command System）

插件化

Filter Chain



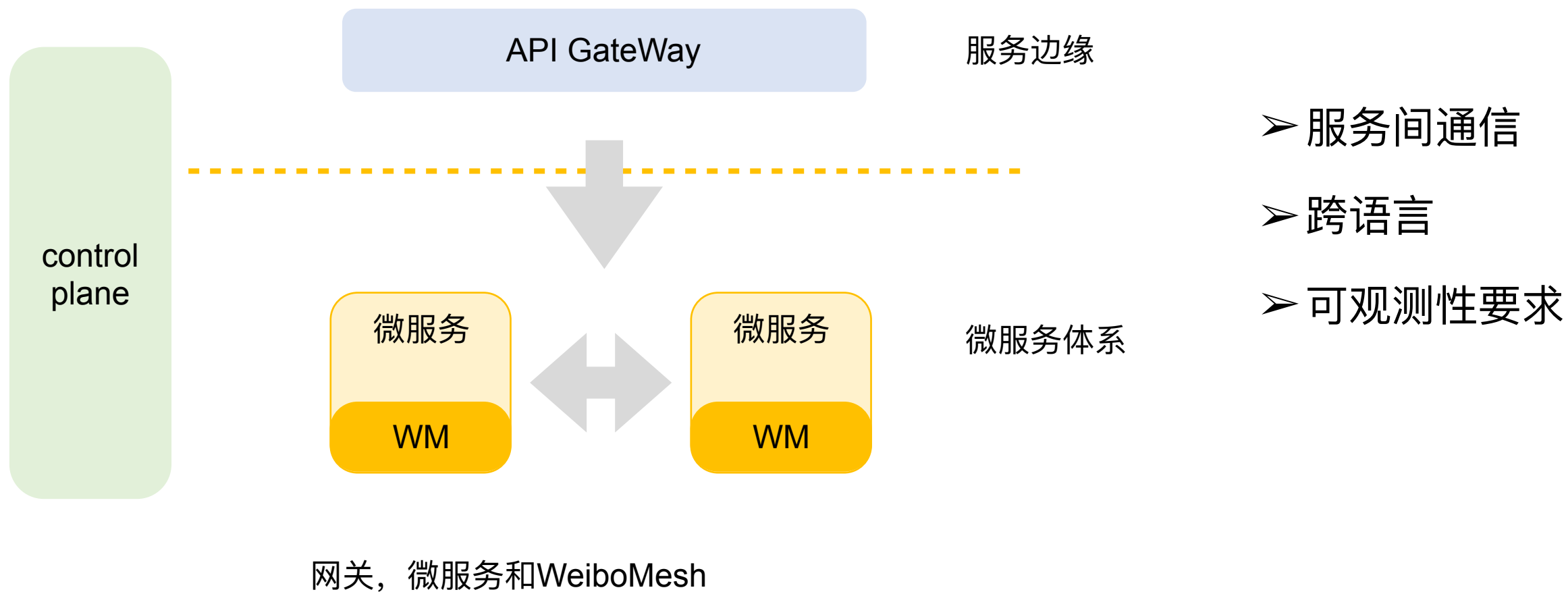
流量调度



4 | 生产实践

- Mesh适用场景
- 迁移成本的考量
- 业务Mesh实践

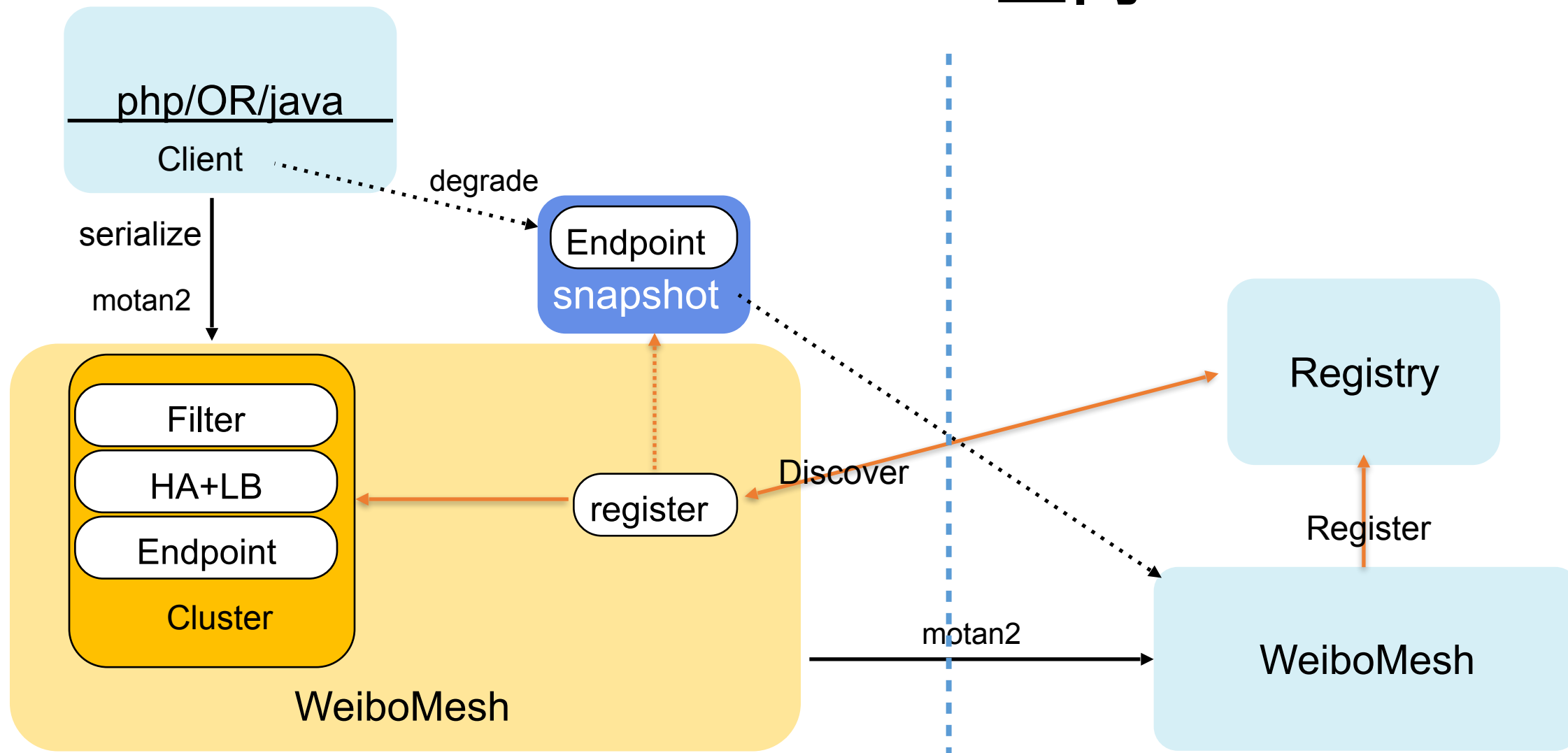
典型场景



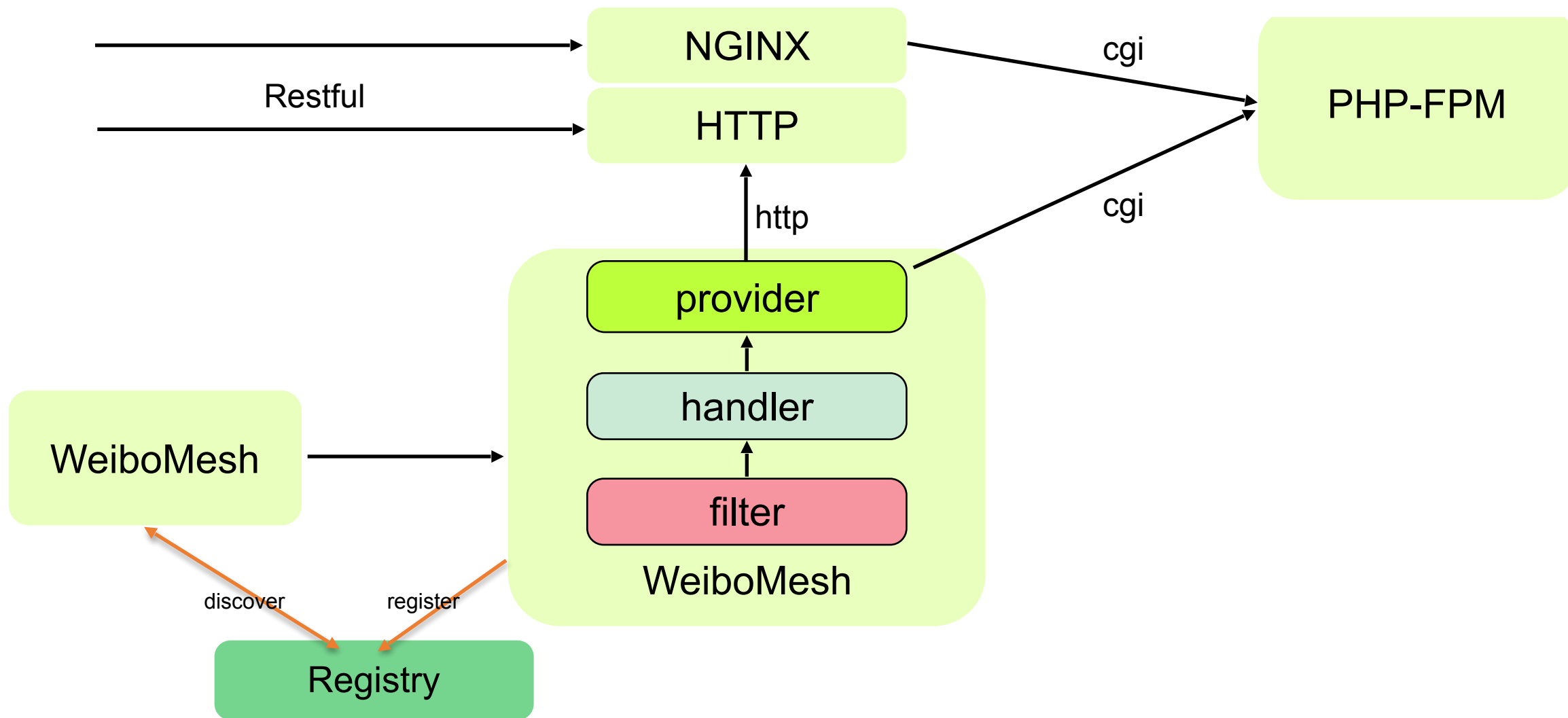
迁移成本的考量

- 业务部署模式？非云，混合云，云原生？
- 适配注册中心
- 适配client
- mesh故障转移
- DevOps

正向Mesh



反向Mesh



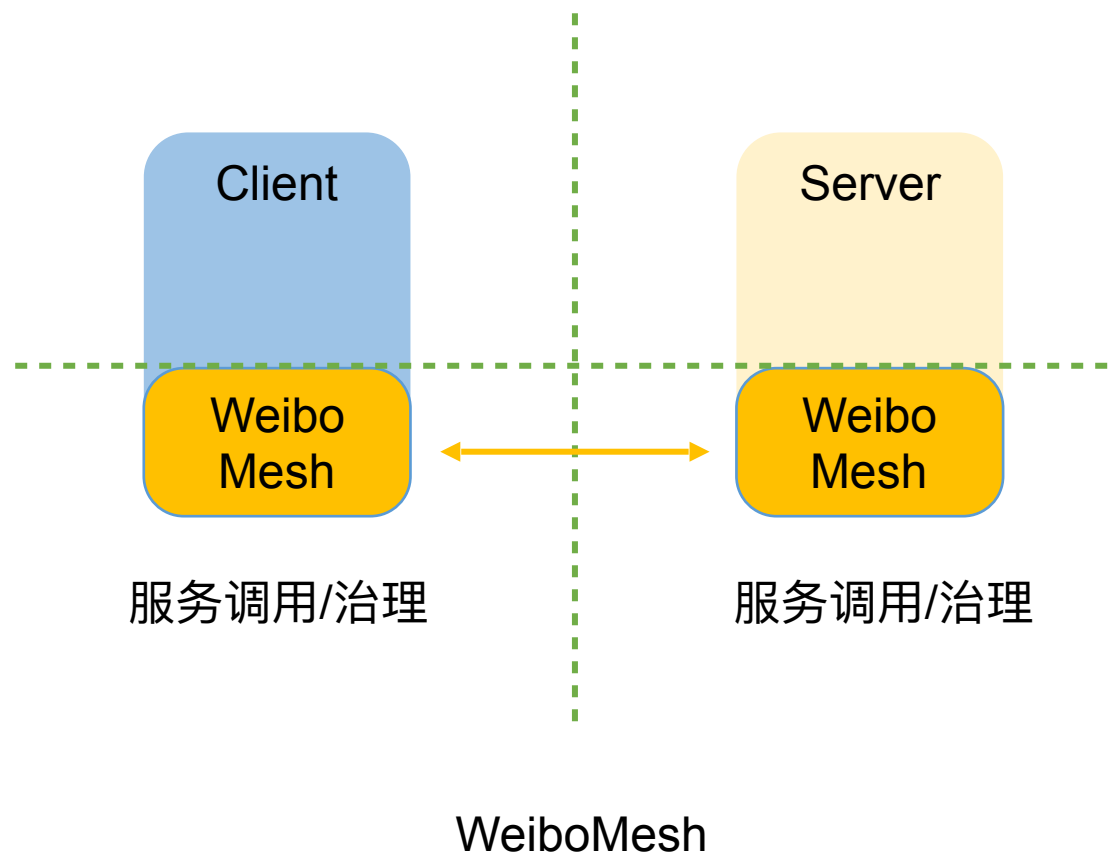
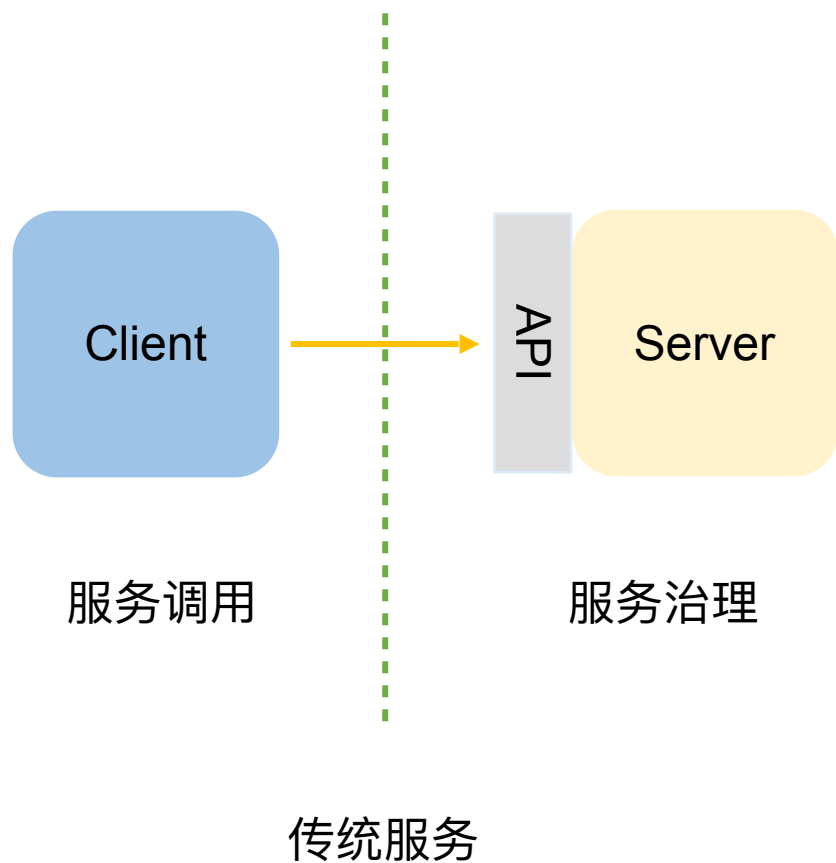
反向Mesh特色

- 提供HTTP/cgi provider，可定制扩展
- HTTP框架自动转RPC，业务无需开发新RPC框架
- mesh对server端无侵入

5

总结

治理模式的差异



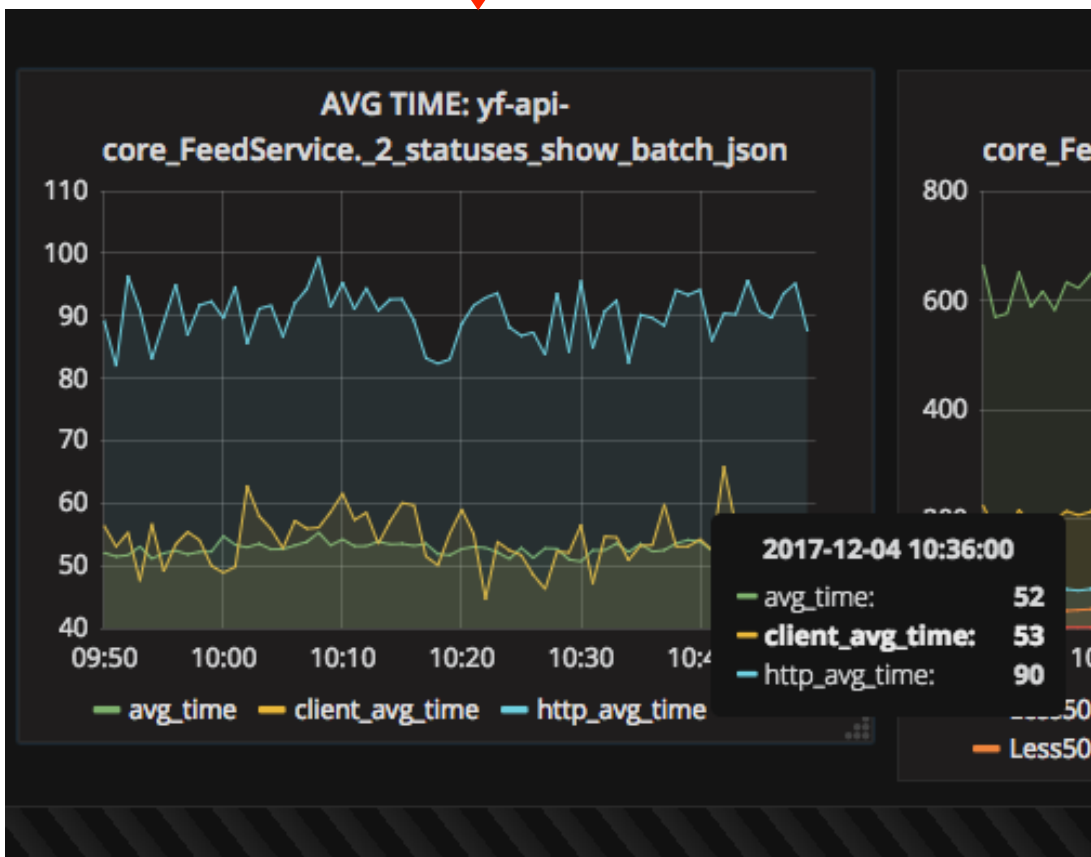
WeiboMesh优势

- 抽象治理/交互基础层 (L5)
- 跨语言服务化
- 业务逻辑解藕
- 可持续交付
- 可观察性
- 业务迁移成本极低，web自动转rpc，可定制
- 适合非云，混合云，适配Registry支持云原生

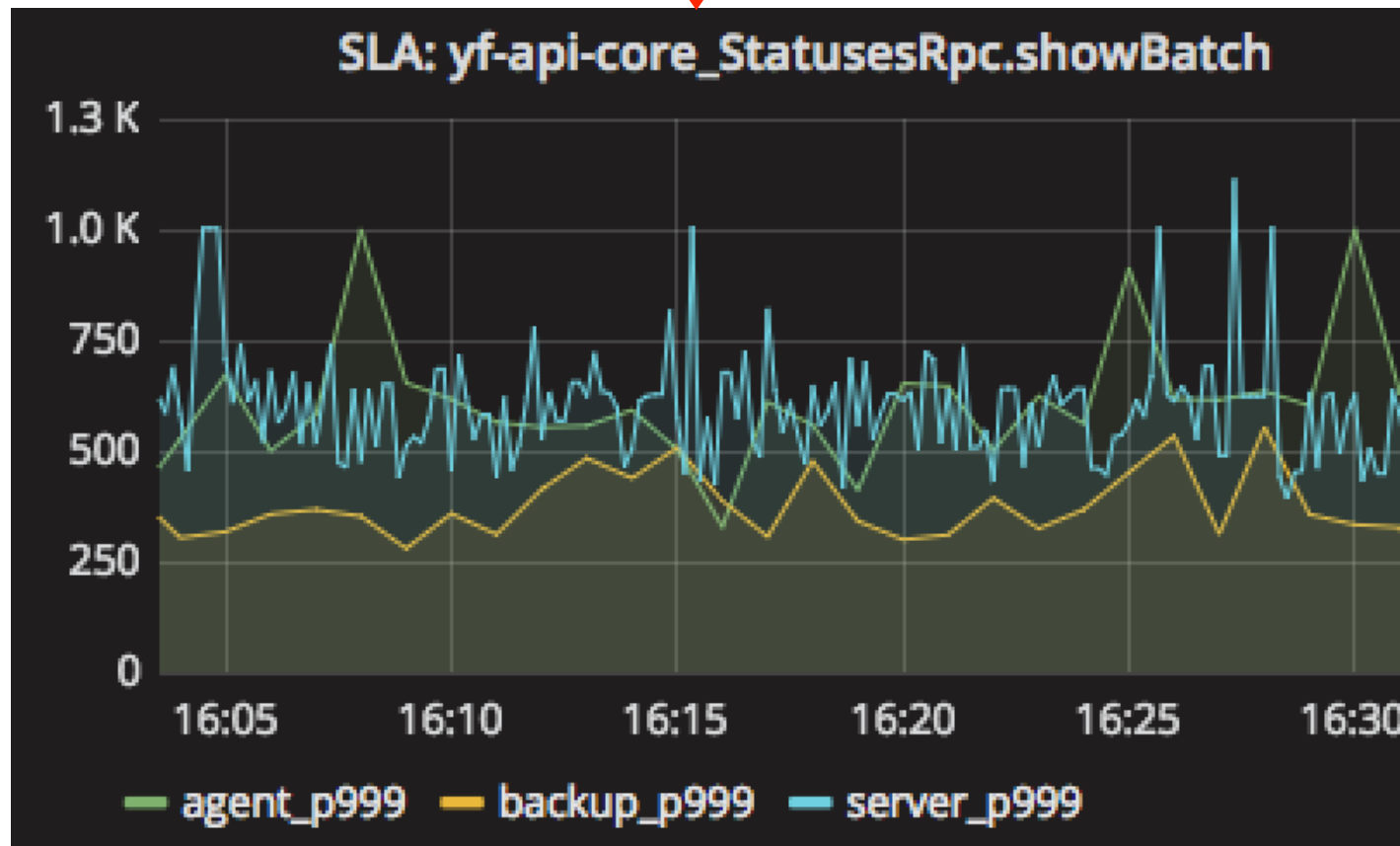
平均耗时  20%~40%

实战效果

SLA999  15%~50%

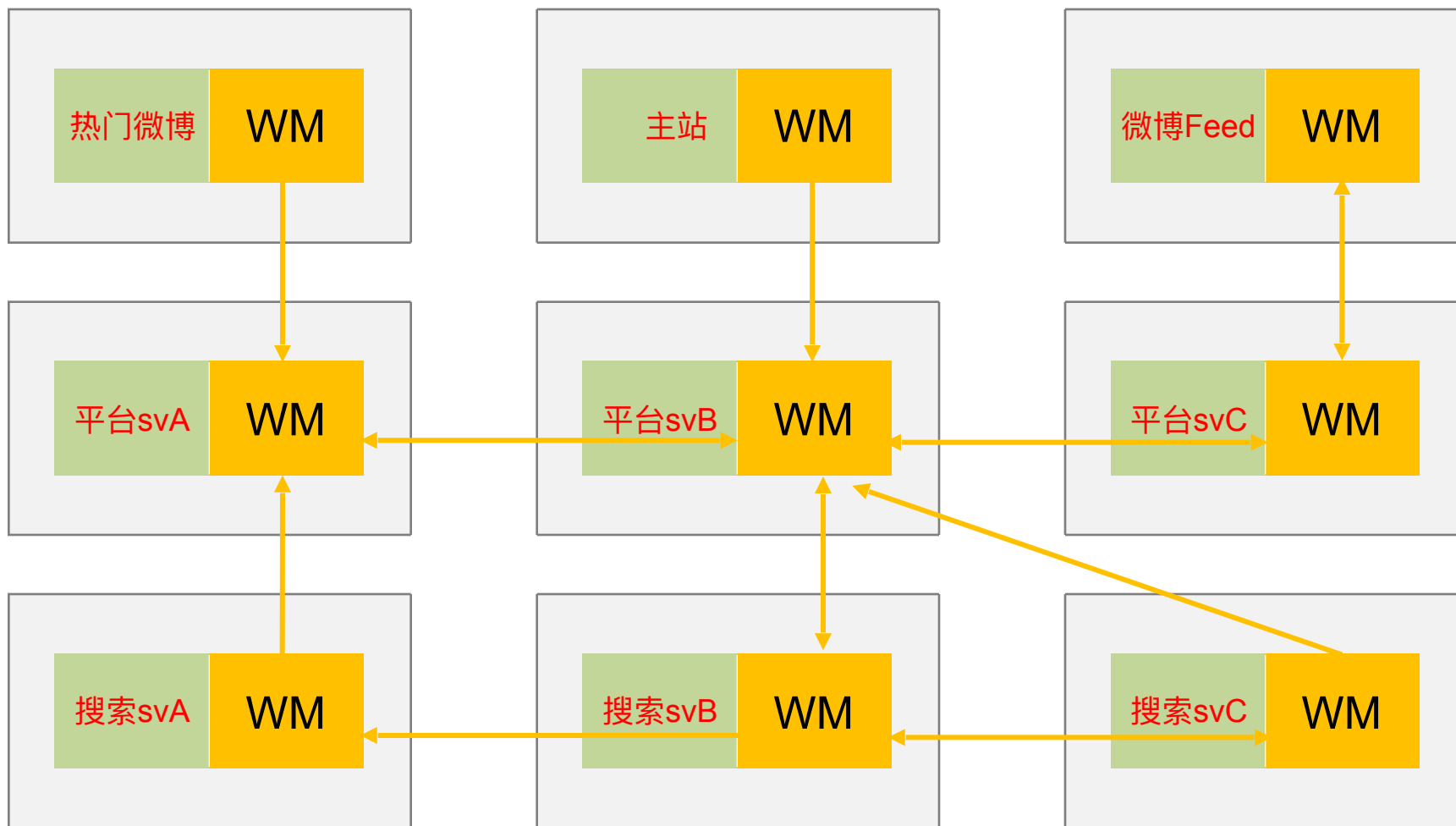


Mesh VS HTTP

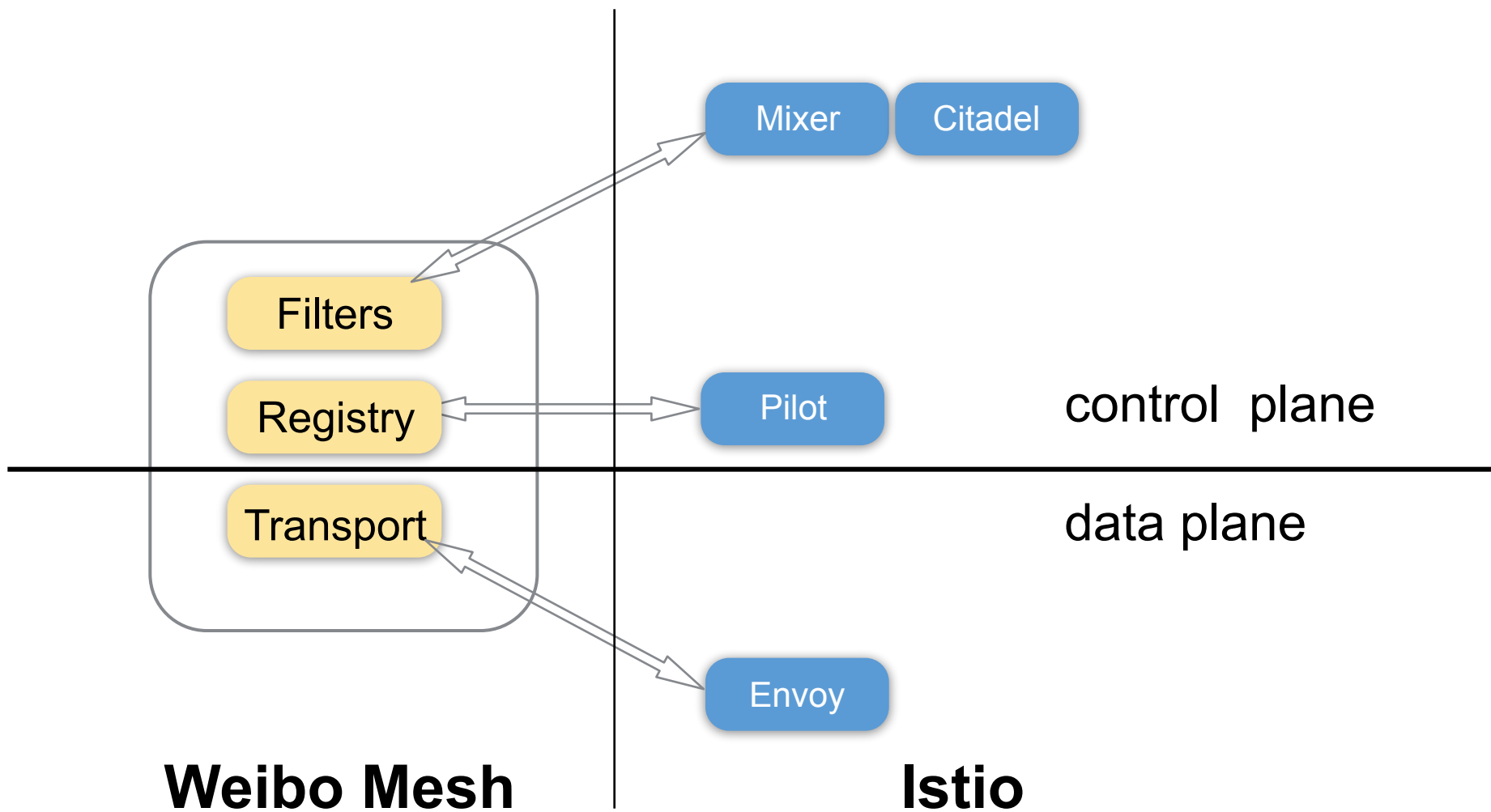


Backup Request 效果图

WeiboMesh集群

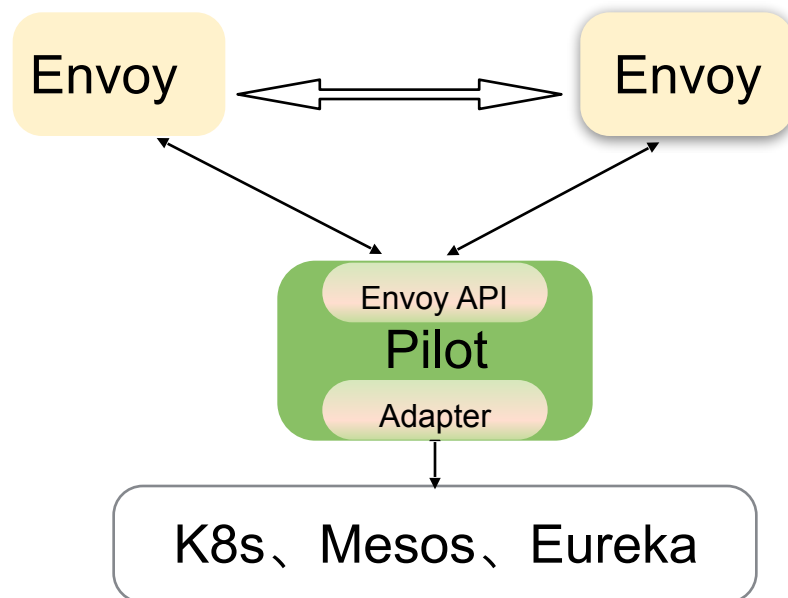


和Istio的区别-结构

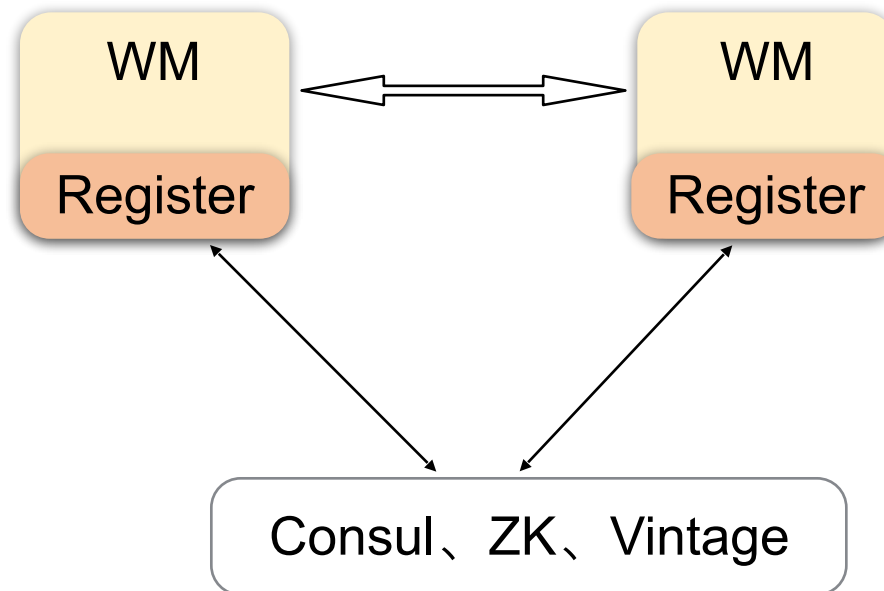


和Istio的区别-发现

云 VS 非云



Istio: Pilot适配云平台



WM: 注册中心

服务透明 VS 模块耦合

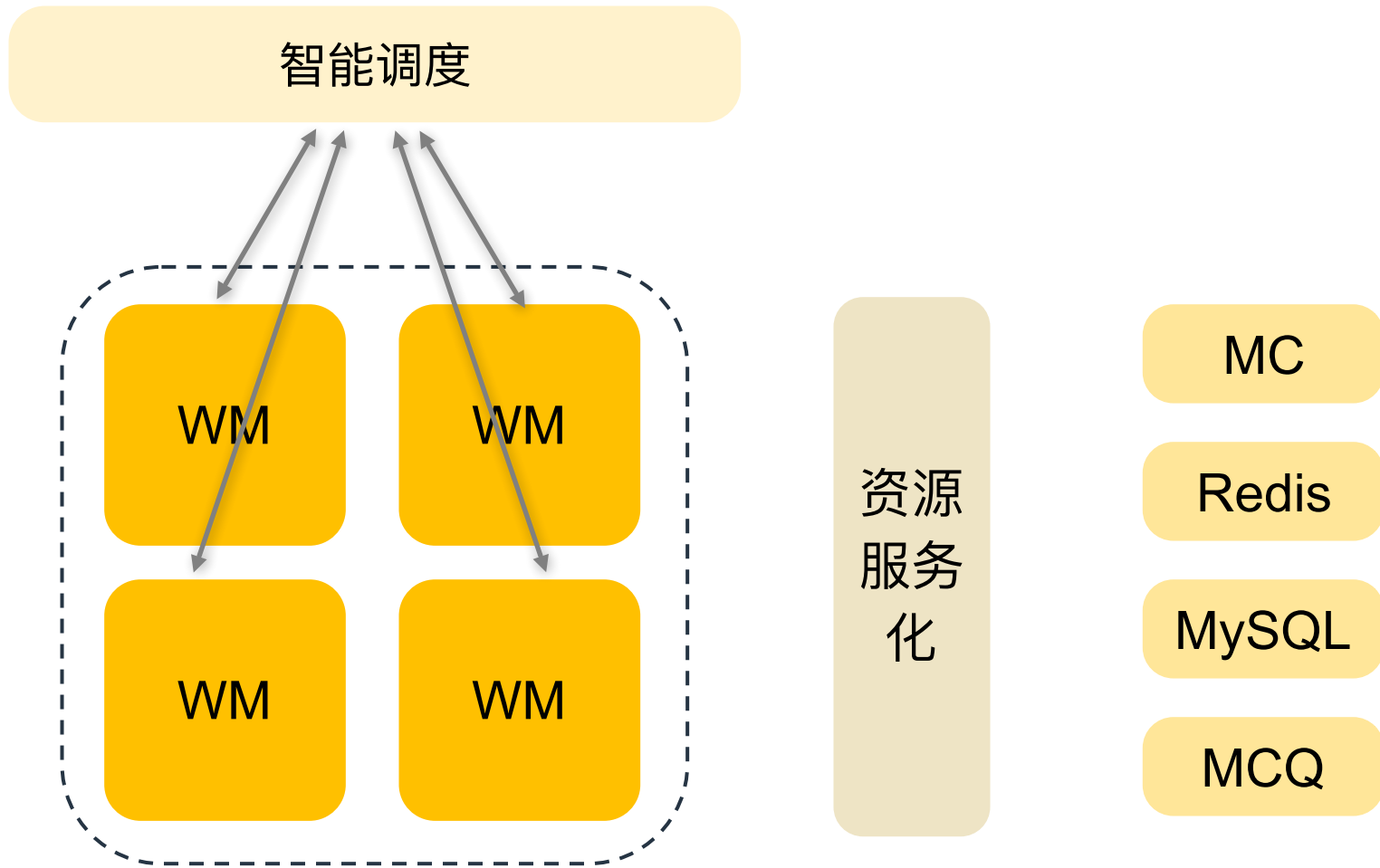
- 云原生
- IPtables流量拦截
- 服务无感知

Istio：对服务透明

- 耦合度可选
- 定制化开发

WM：模块化耦合

WM进行中



WM未来发展方向

结合容器编排进行优势互补
整合进L5层

云原生

易用性

流量拦截

更广泛的语言支持

更小的部署迁移成本

更高的性能

WeiboMesh: <https://github.com/weibocom/motan-go>

JAVA: <https://github.com/weibocom/motan>

OR: <https://github.com/weibocom/motan-openresty>

PHP: <https://github.com/weibocom/motan-php>

Examples: <https://github.com/motan-ecosystem/motan-examples>



100

**TOP100 CASE STUDIES
OF THE YEAR**