

**Gdevops**

# 全球敏捷运维峰会

新浪微博大规模  
基于Docker的混合云应用实践

演讲人：王关胜  
资深运维架构师

# 主要分享内容

---

目录  
content

---

- 一、业务背景及峰值应对
- 二、DCP的架构设计挑战
- 三、业务上云的应用实践



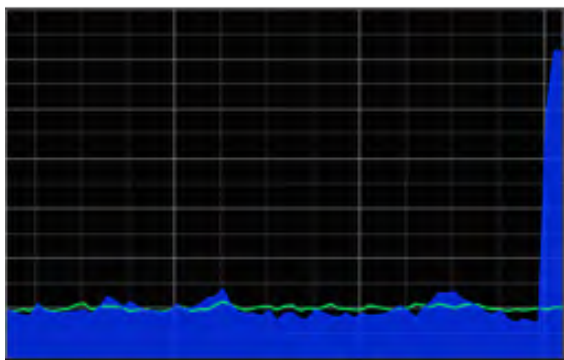
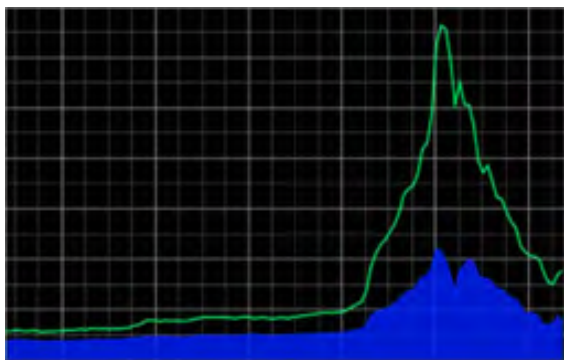
# Part 1

---

## 微博业务背景 及峰值应对

---

# 微博业务场景 – 极端流量常见



- 特点：
  - 瞬间峰值高
  - 互动时间短
- 挑战：传统应对手段成本高
  - 设备成本
  - 时间成本

# 总结：面临的挑战

1

## 产品上：迭代快

现状：功能多，依赖复杂

挑战：发布&变更频繁

2

## 运营上：大型活动&重要新闻的Push

现状：站内外，活动，运营，大V均有Push场景

挑战：全量极速下发，互动时间短

3

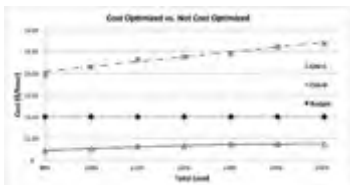
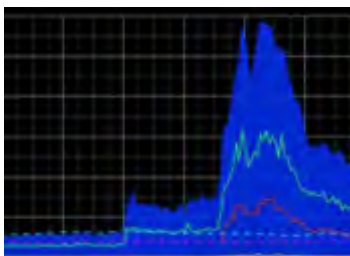
## 技术上：突发的极端流量

现状：热点多，#马航370# #刘翔摔倒# #王宝强#

挑战：峰值应对，考验服务的弹性伸缩能力

# 峰值应对 - 关注点

- 快速扩容
- 及时回收



- 可伸缩的业务利用公有云
- 私有云内弹性部署

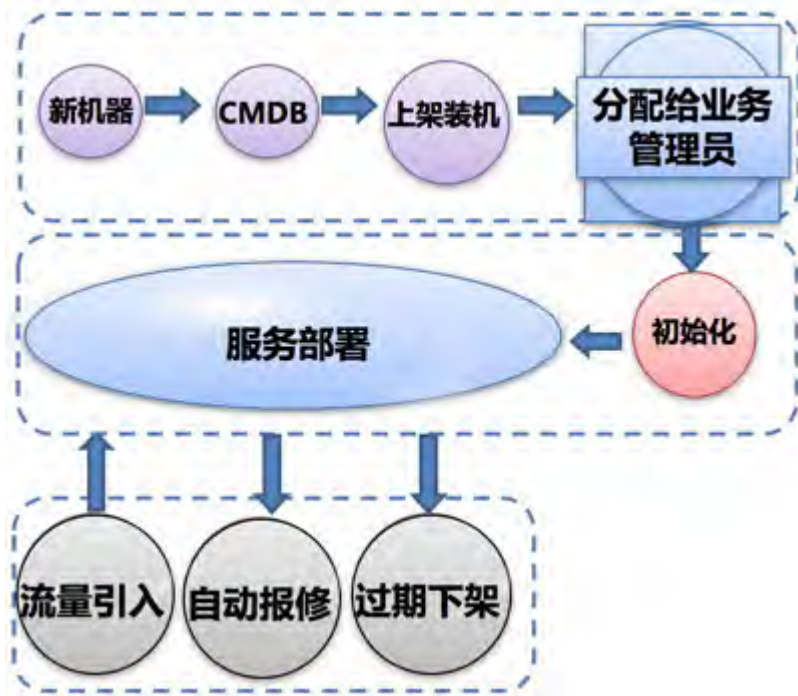


- 打通多语言环境
- 全公司统一平台



- 标准化基础设施
- 提高发布效率

# 峰值应对 - 传统手段



## 业务运维可控

### Step1 设备申请

- 设备申请, 项目评审
- 入CMDB, 上架装机

### Step2 机器初始化

- 设备录入资源池, 初始化

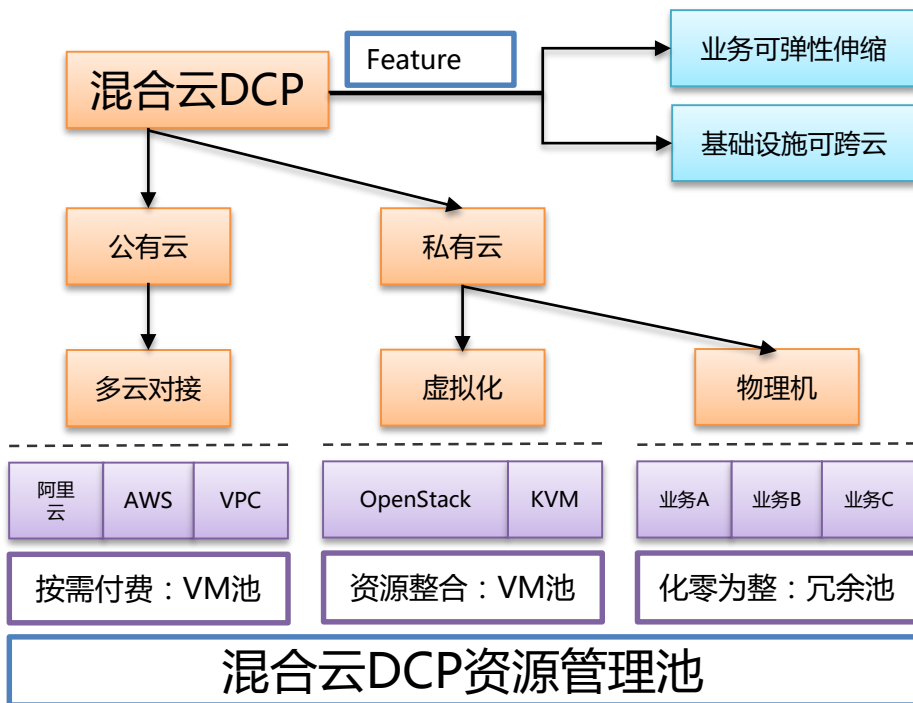
### Step3 服务部署

- 环境部署
- 监控部署
- 服务部署 (代码 & confs)
- 流量引入

### Step4 自动报修&下架

- 服务自动上下线
- 设备置换或下架

# 峰值应对 – DCP的弹性伸缩



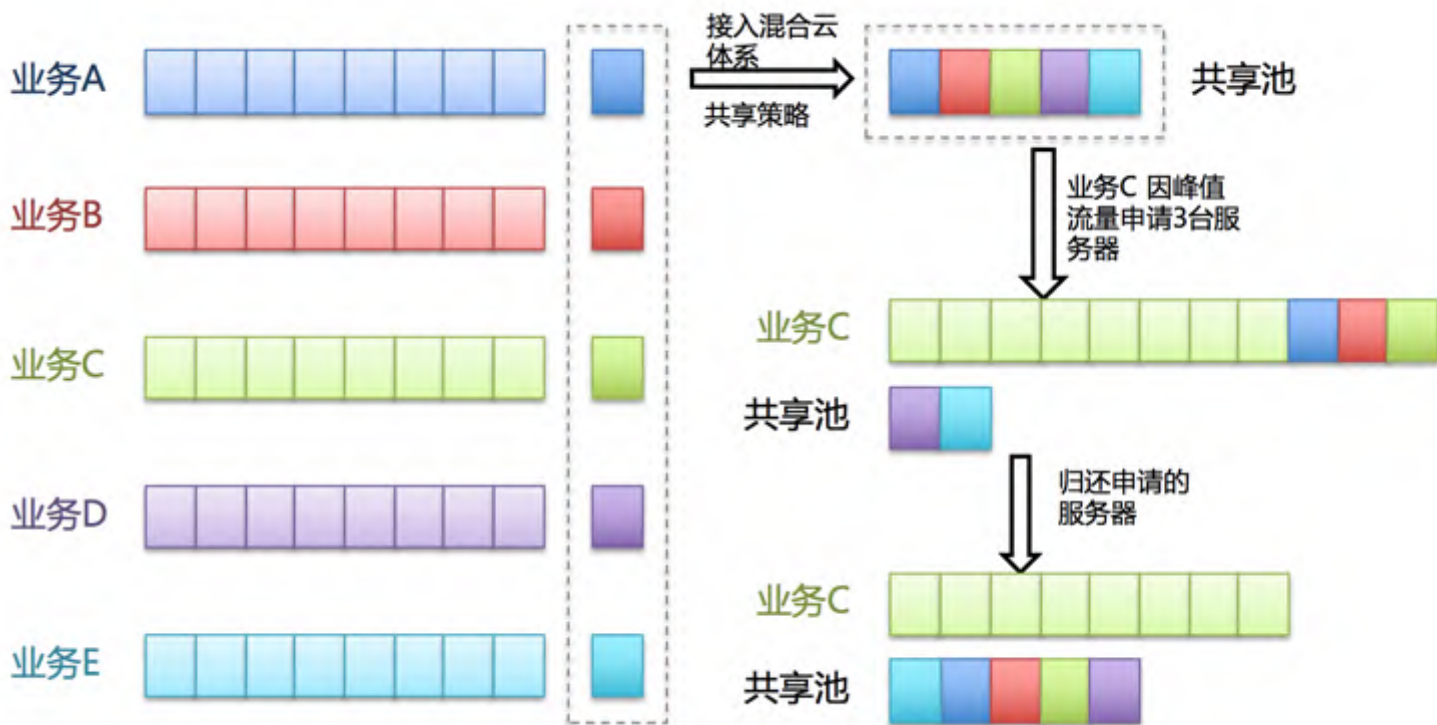
云化架构—基于docker

- 业务服务化
- 微服务化
- 业务消息化
- 多机房部署

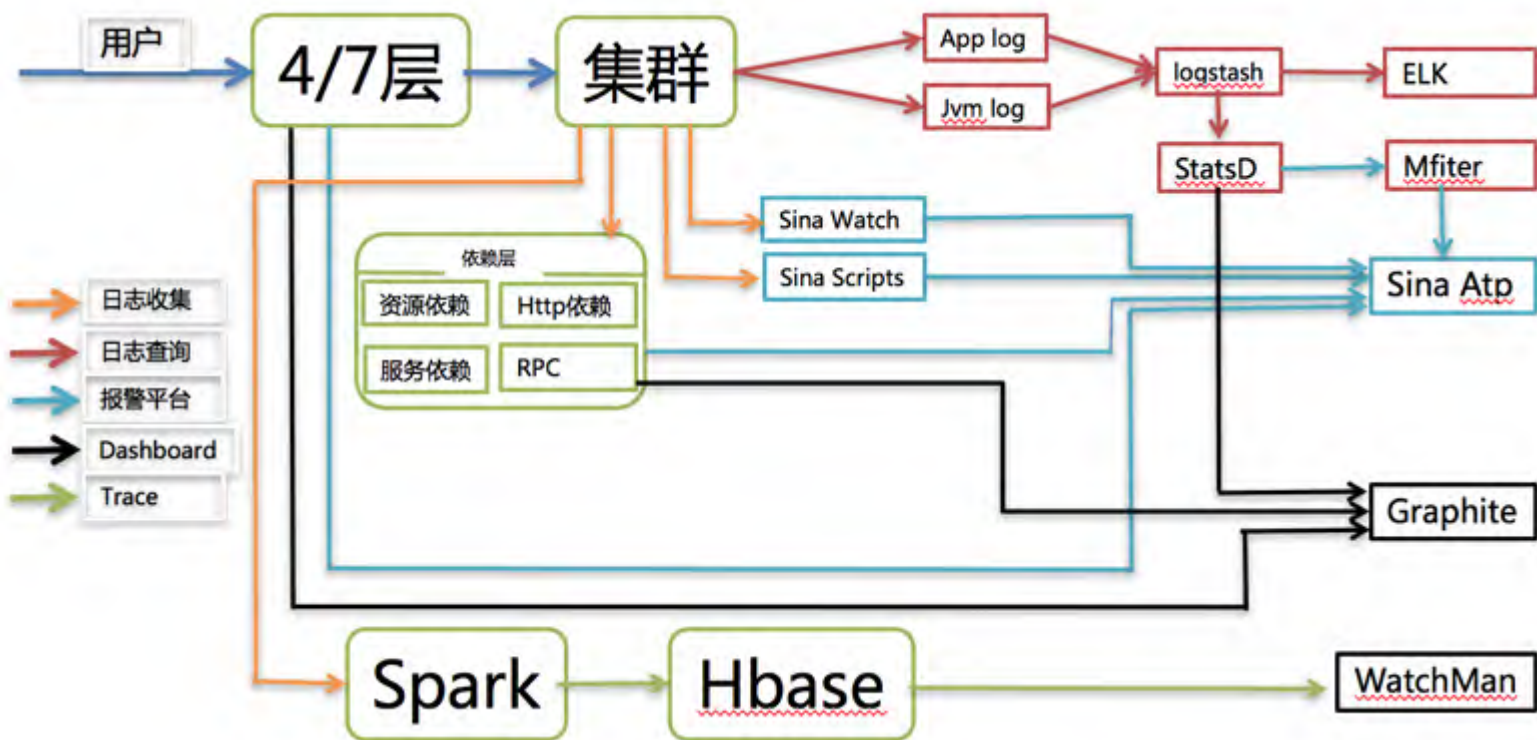
- 敏捷基础设施
- 持续集成
- 多租户&隔离
- 弹性伸缩
- 故障自愈



# DCP的弹性伸缩 - 私有云也弹性



# 峰值服务保障 - 统一监控平台



# 峰值服务保障 - 核心链路服务自动伸缩

The screenshot displays the DCP Elastic Scheduling (弹性调度) interface. A modal window titled "输入任务信息" (Enter Task Information) is open over a task list. The modal contains the following configuration fields:

- Disable:  on  off
- Name: aliyun-v4-action
- Owner: guansheng
- MailRecipients: [empty]
- AddStrategy: StrategyNone
- RemoveStrategy: StrategyNone
- PlutoAppId: [redacted]
- PlutoAppKey: [redacted]
- Machine:  16核16G  8核8G  16核64G
- ScheduleType:  crontab  depend  auto
- Jobs: Time: 23:49 InstanceNum: 0

At the bottom of the modal, there is a "+ 添加job" button. The background shows a task list table with columns for No., Sid, and 修改时间.

# 峰值服务保障 – 预案&干预手段

- 预案：100+
  - ◆ 日常&应急预案
  - ◆ 重大活动，三节等预案手册
- 服务降级：5000+开关
- 有效的干预手段





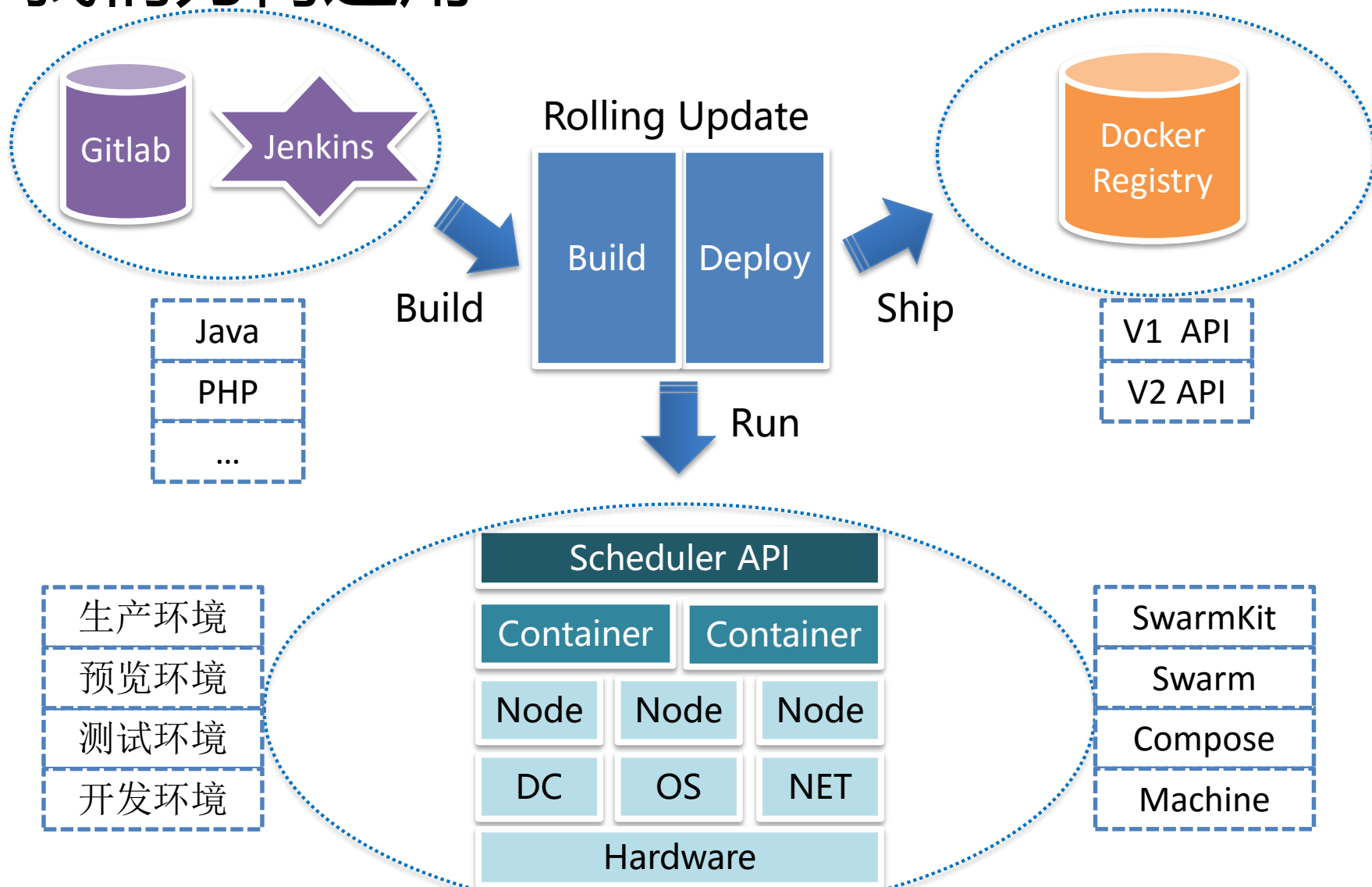
# Part 2

---

## Weibo DCP 架构设计挑战

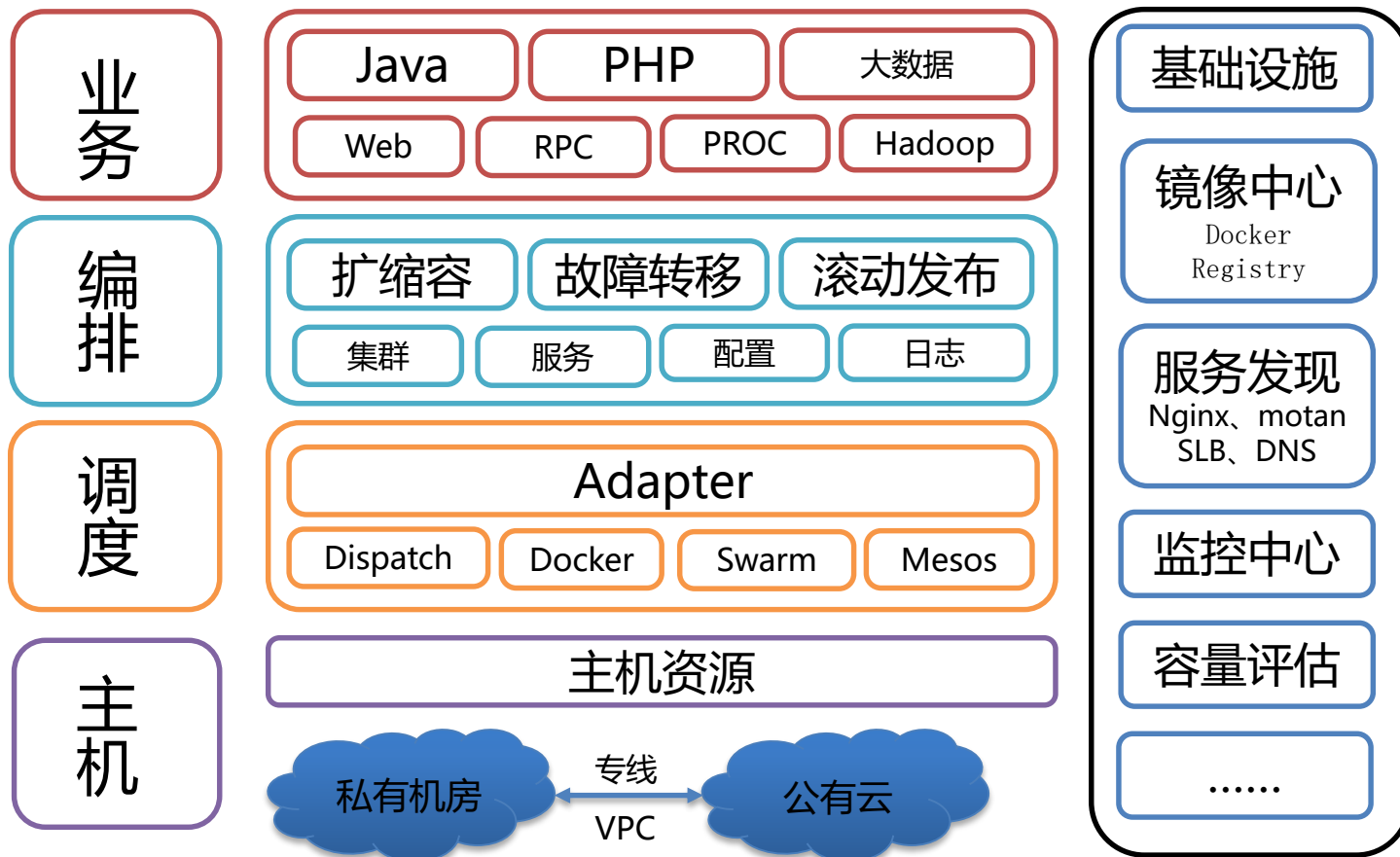
---

# 我们为何选用Docker

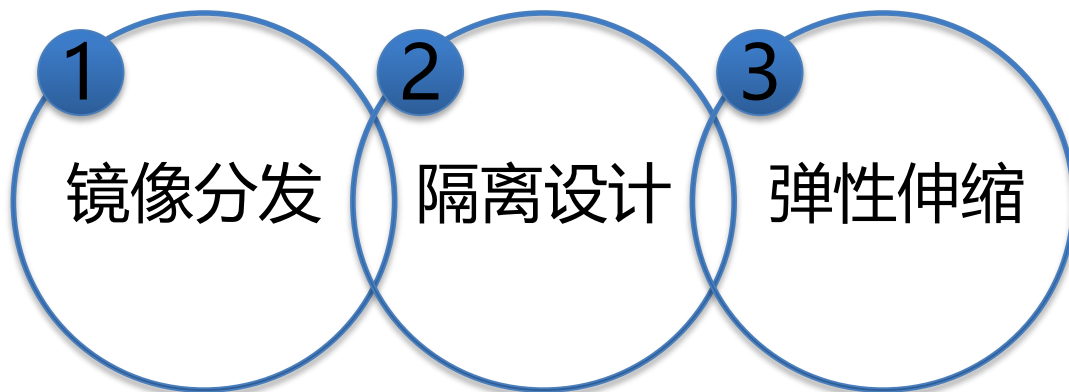


# Weibo DCP - 整体架构介绍

- 方案设计：来源于官方三驾马车（Machine + Compose + Swarm）



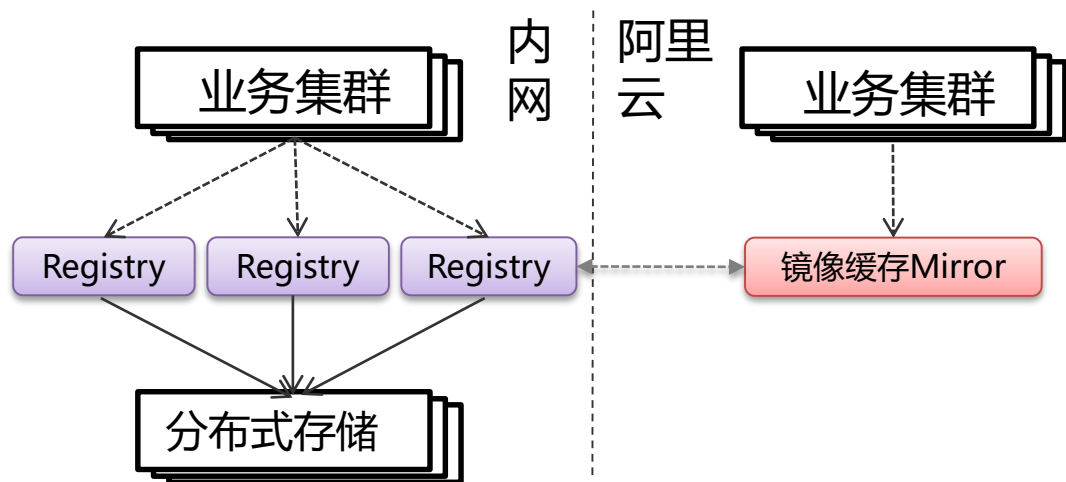
# Weibo DCP – 架构设计挑战



- 镜像分发
  - 镜像优化
  - 分发速度
- 隔离设计
  - 平台层隔离
  - 部署/实例隔离
- 弹性伸缩
  - 自动扩缩容
  - 故障转移



# 挑战一：镜像分发 - 镜像优化



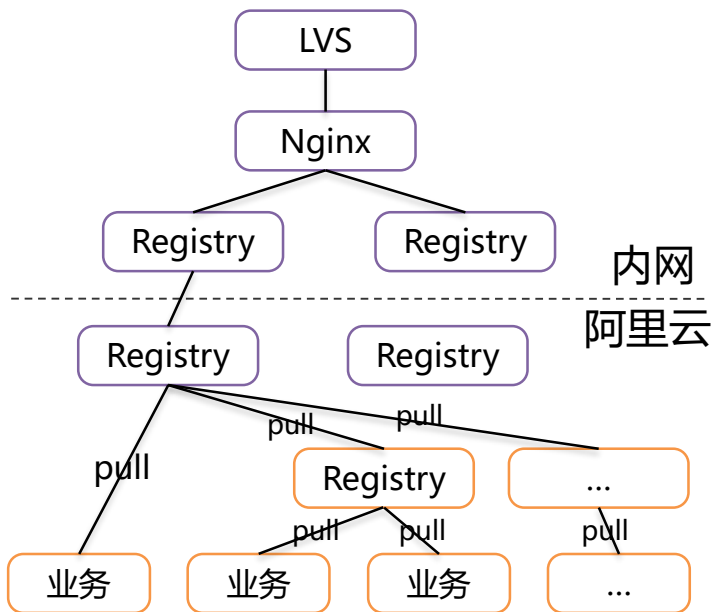
- 镜像制作优化

- 镜像分层，逐层复用
- 制作微镜像

- 仓库部署优化

- Storage driver : Ceph
- 多机房部署：镜像缓存Mirror

# 挑战一：镜像分发 - 分发速度



## ● 部署模式

- 常规部署：最小化模式
- 弹性扩缩容：依赖模式

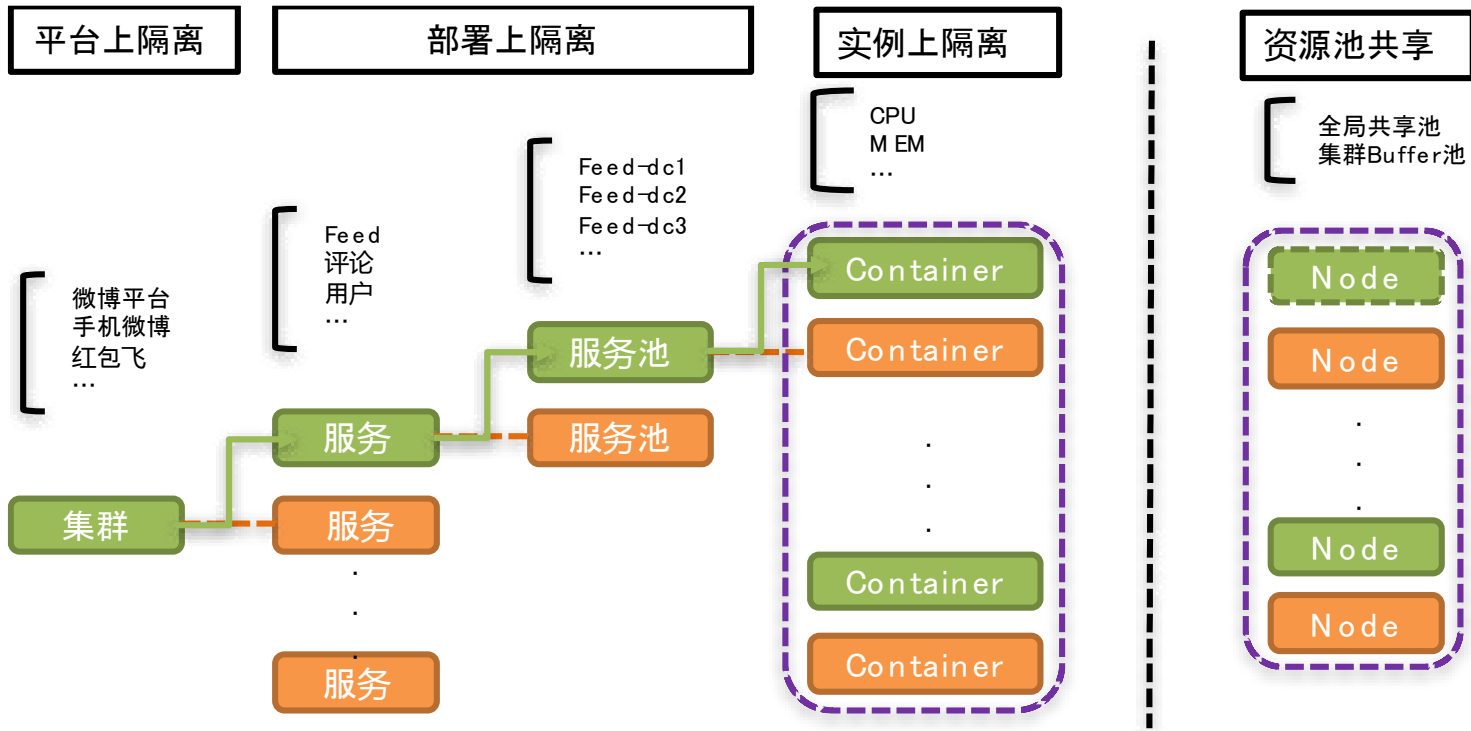
## ● 核心优势

- 镜像穿透：级联&预热
- 带宽优化：打散
- 分发速度：千台规模分钟级

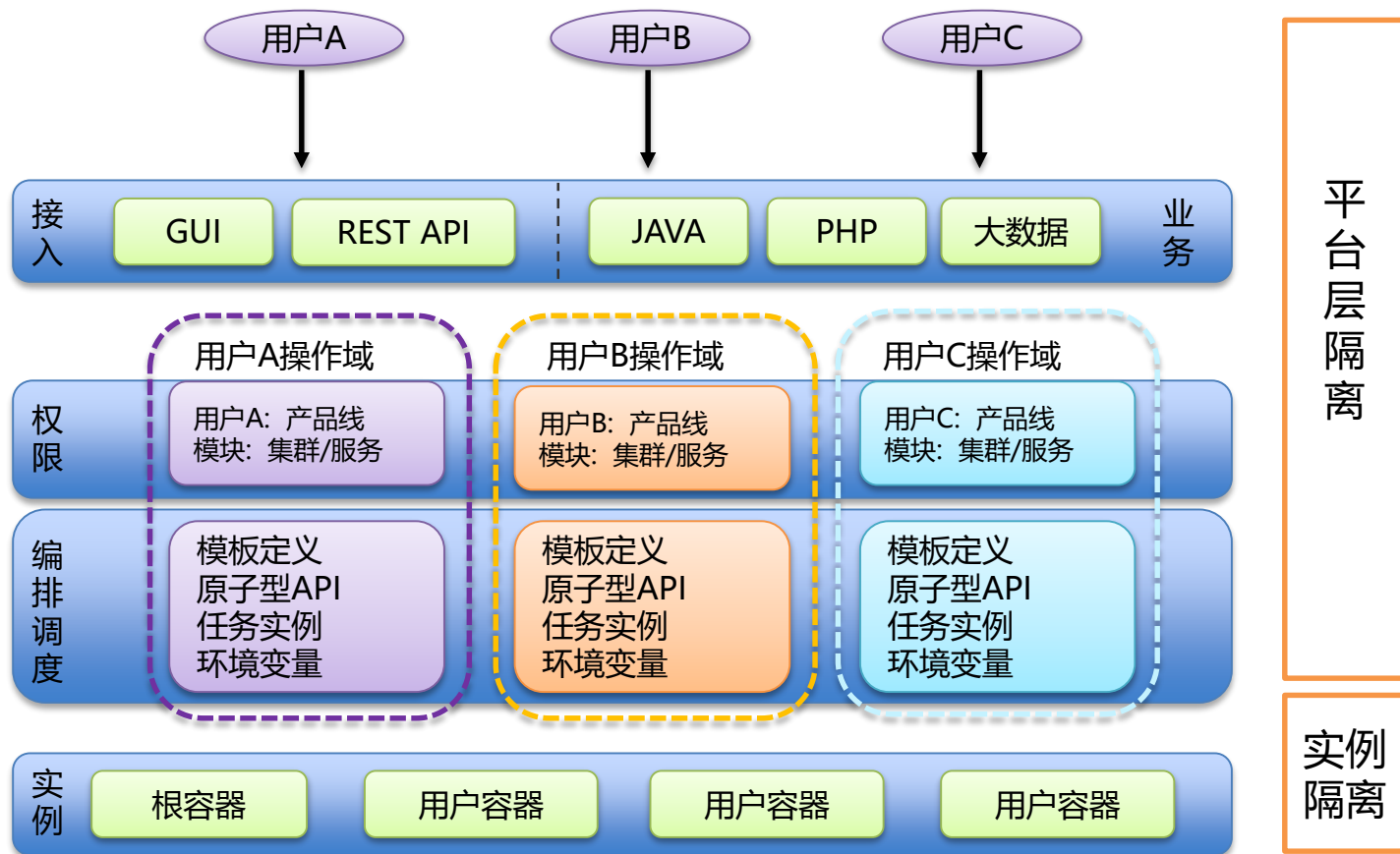
## ● 未来方向

- 支持p2p

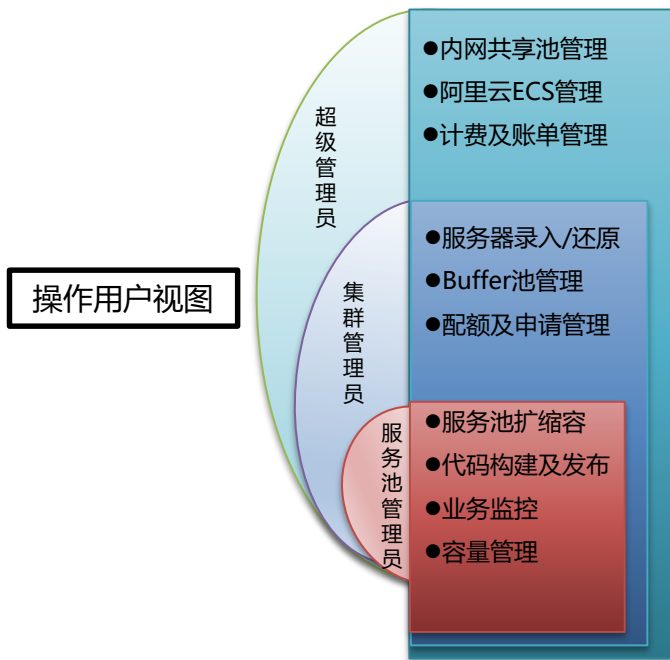
# 挑战二：隔离设计 - 隔离模型



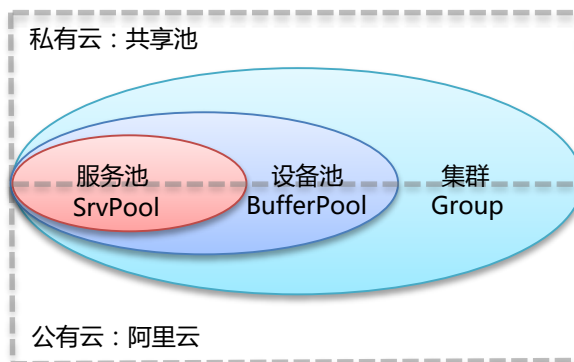
# 挑战二：隔离设计 - 平台层实现



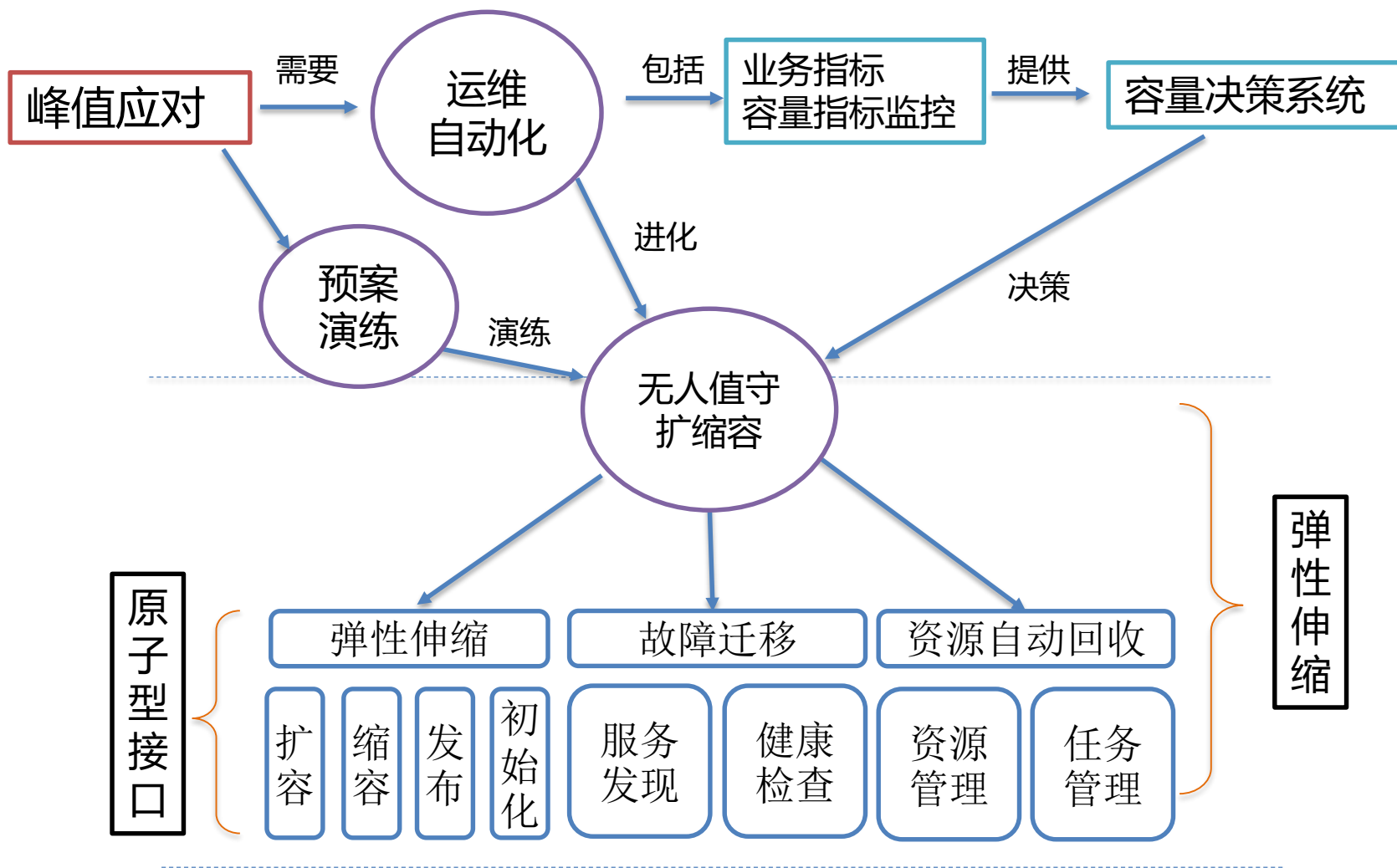
# 挑战二：隔离设计 – 平台用户操作域



- 集群内：自由扩缩容（可跨SrvPool）
- 集群外：配额调度
  - 集群有配额：自动获取资源/归还
  - 集群无配额：先申请配额，再申请资源

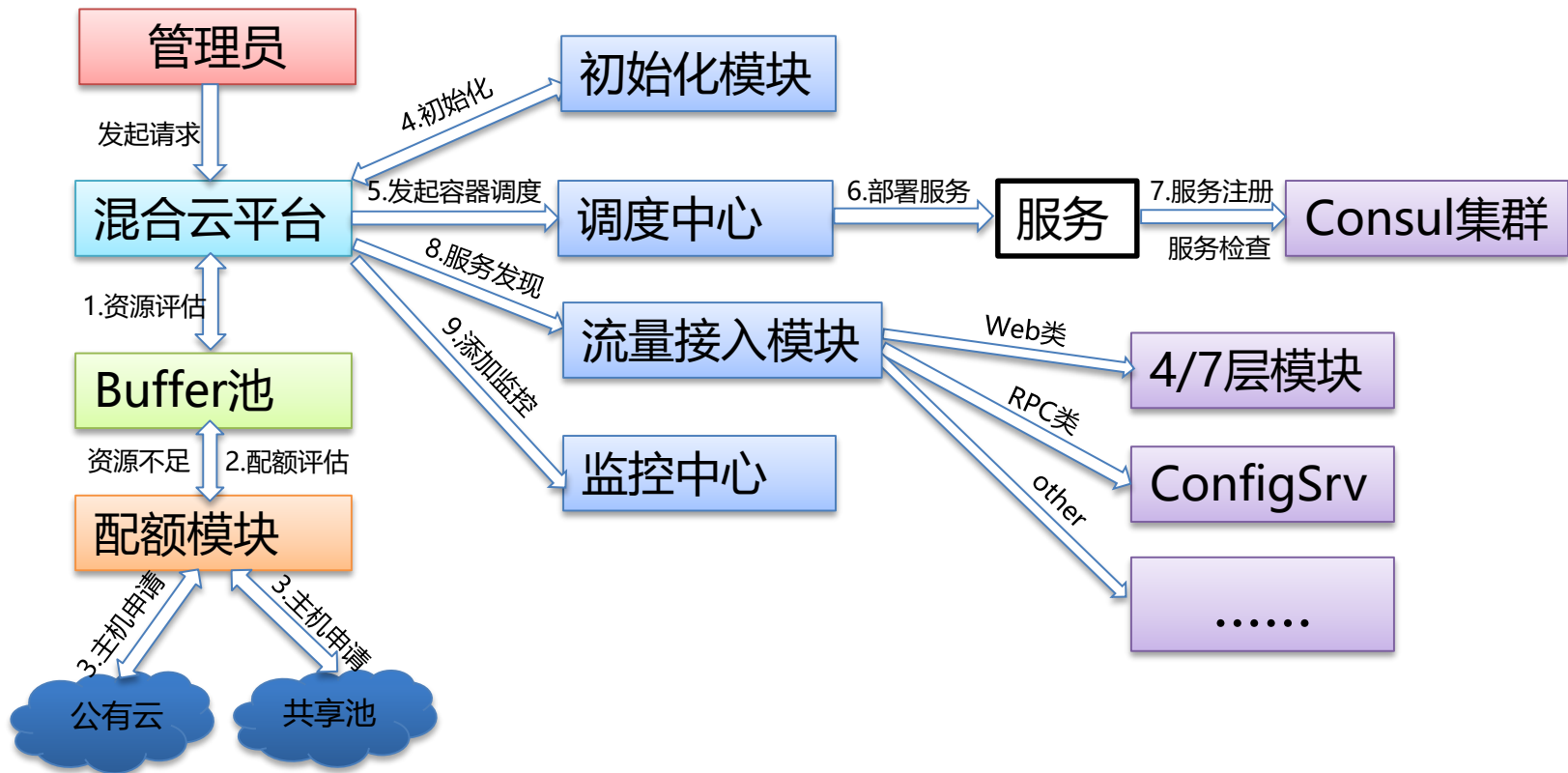


# 挑战三：弹性伸缩 - “无人值守” 扩缩容



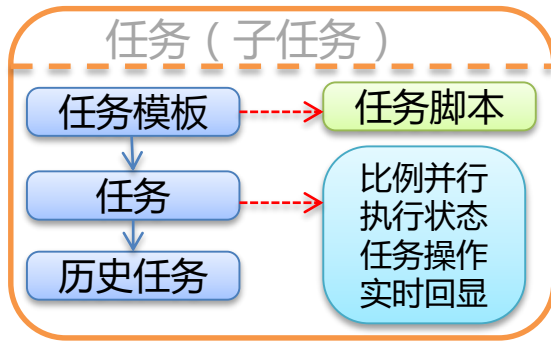
# 挑战三：弹性伸缩 - 扩容模板

## 原子型API任务



# 挑战三：弹性伸缩 - 原子型任务系统

新浪自研：C++编写

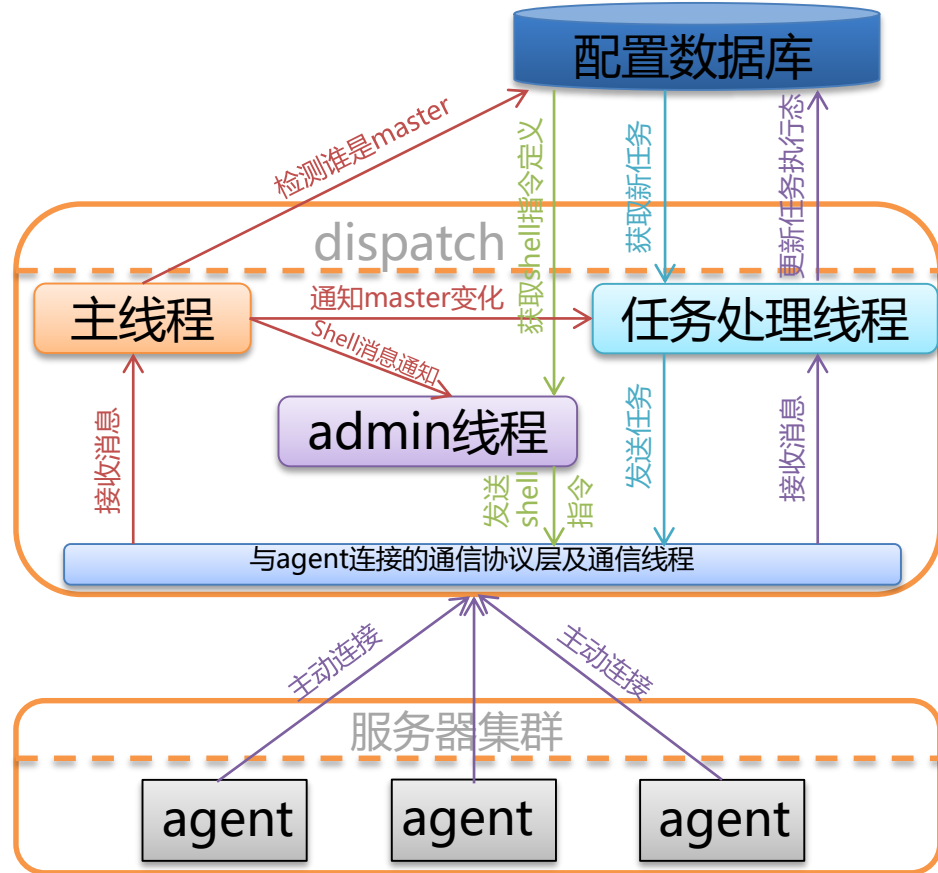


→ 主线程

→ agent上报

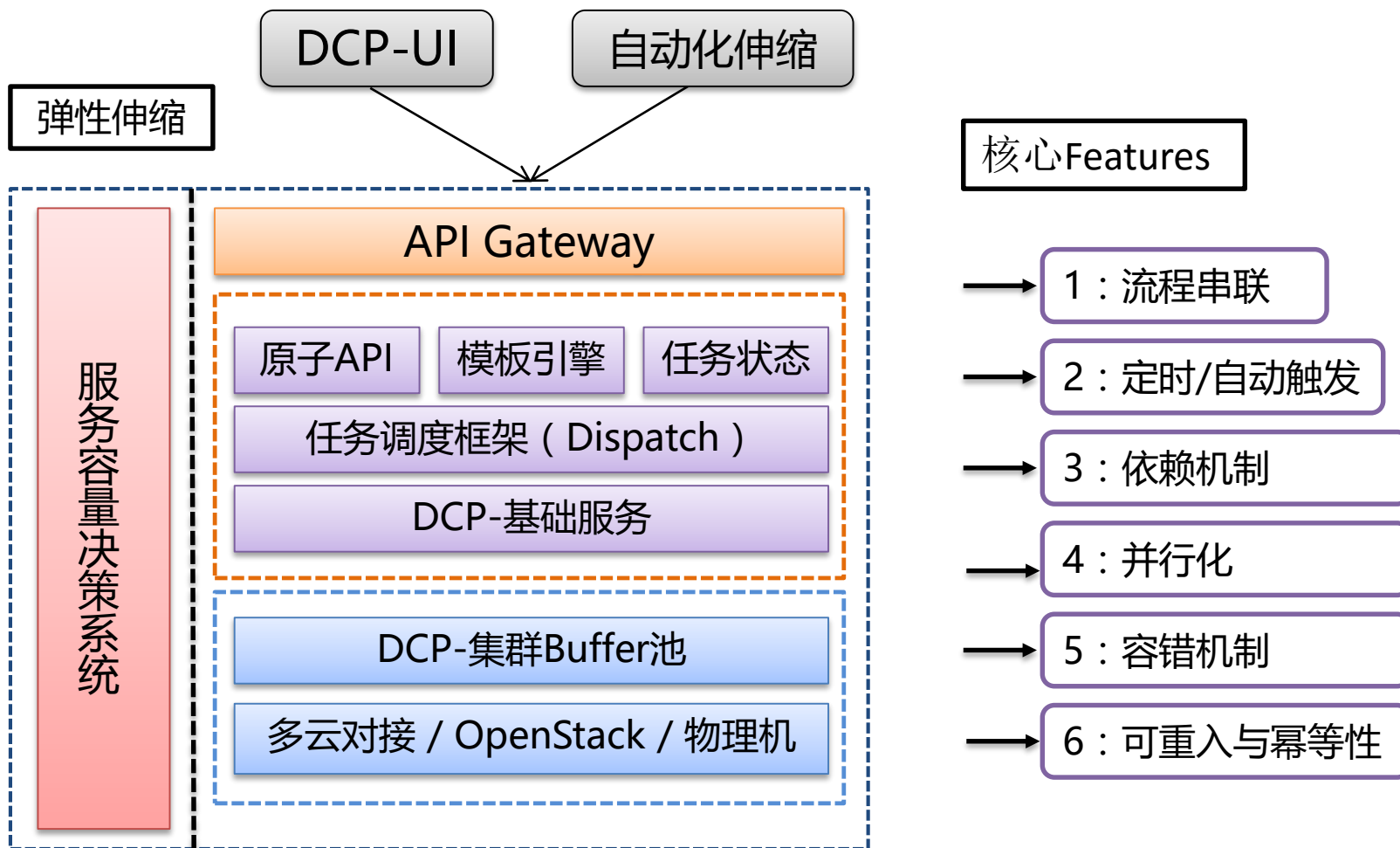
→ Shell调度

→ 任务调度





# 挑战三：弹性伸缩 - 系统框架



# 挑战三：弹性伸缩 - 容量决策

## ● 两种决策方式

### ◆ 自动压测：类Cron方式

- 压测方法：减少服务池可服务的实例数
- 压测机制：503.sh/200.sh
- 压测机指标：数据来源于监控中心（粒度10s）

系统	Load	<12	12<X<24	>24
	Cpu idle	<30%	10%<X<30%	<10%
	Iowait	<20%	20%<X<35%	>50%
	Swap	<500M	1G<X<2G	>2G
业务	5xx错误比率	<1%	1%<x<5%	>5%
	接口平均耗时	<100ms	100-500ms	>1s

### ◆ 容量预估：数据来源于业务量监控

- 同比分析
- 环比分析

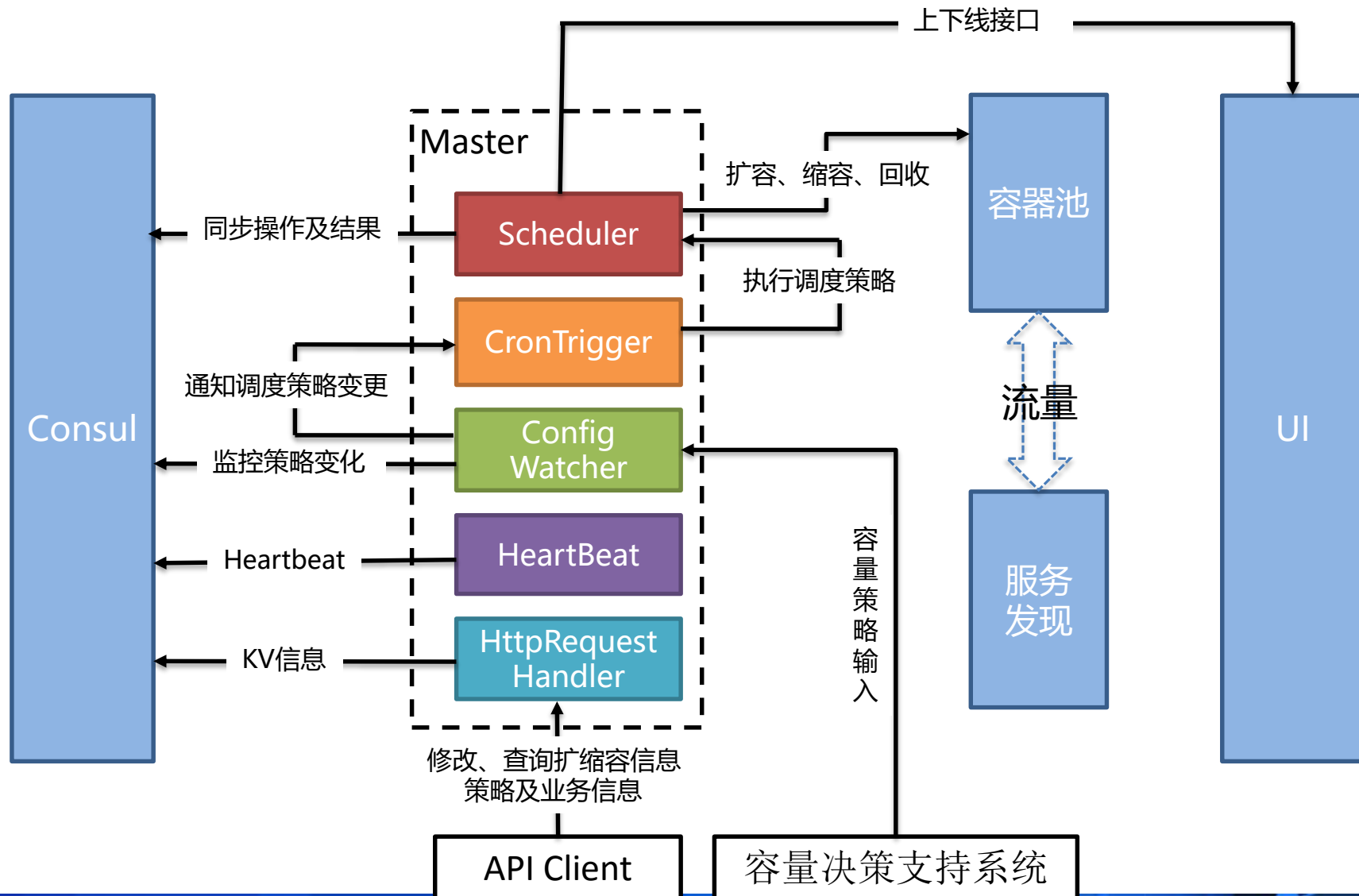
## ● 容量评估产出：

- 水位预警工具
- 容量报表
- 容量API

## ● 集群容量数据一览表



# 挑战三：弹性伸缩 - 自动化Job编排框架





# Part 3

---

## Weibo 业务上云 应用实践

---

# 业务上云的标准姿势

## 上云可行性

安全上：敏感数据

部署上：业务依赖

数据上：数据同步

自动化：弹性伸缩



标准姿势



## 核心特点

计算类业务上云

数据传输/同步

基础设施跨云

混合云平台

业务链路全上云

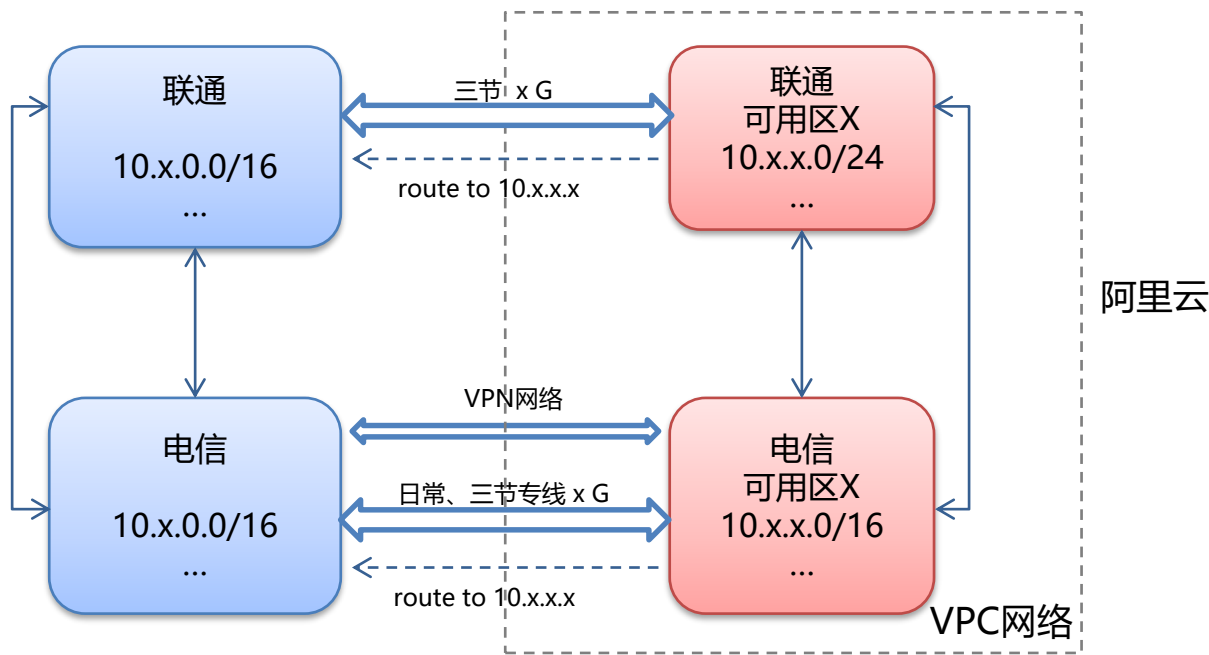
云资源API化

云管控平台

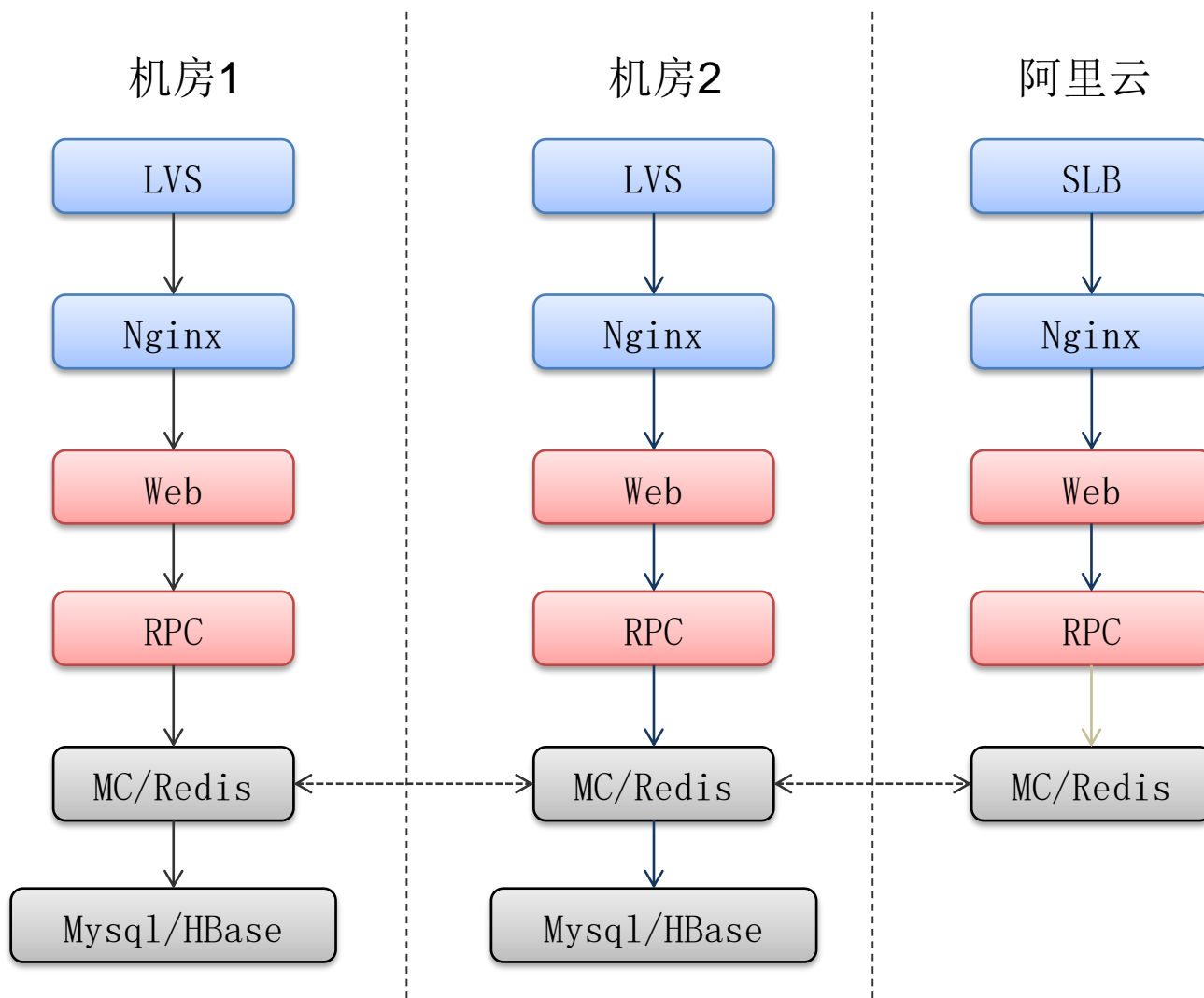
混合云

XaaS

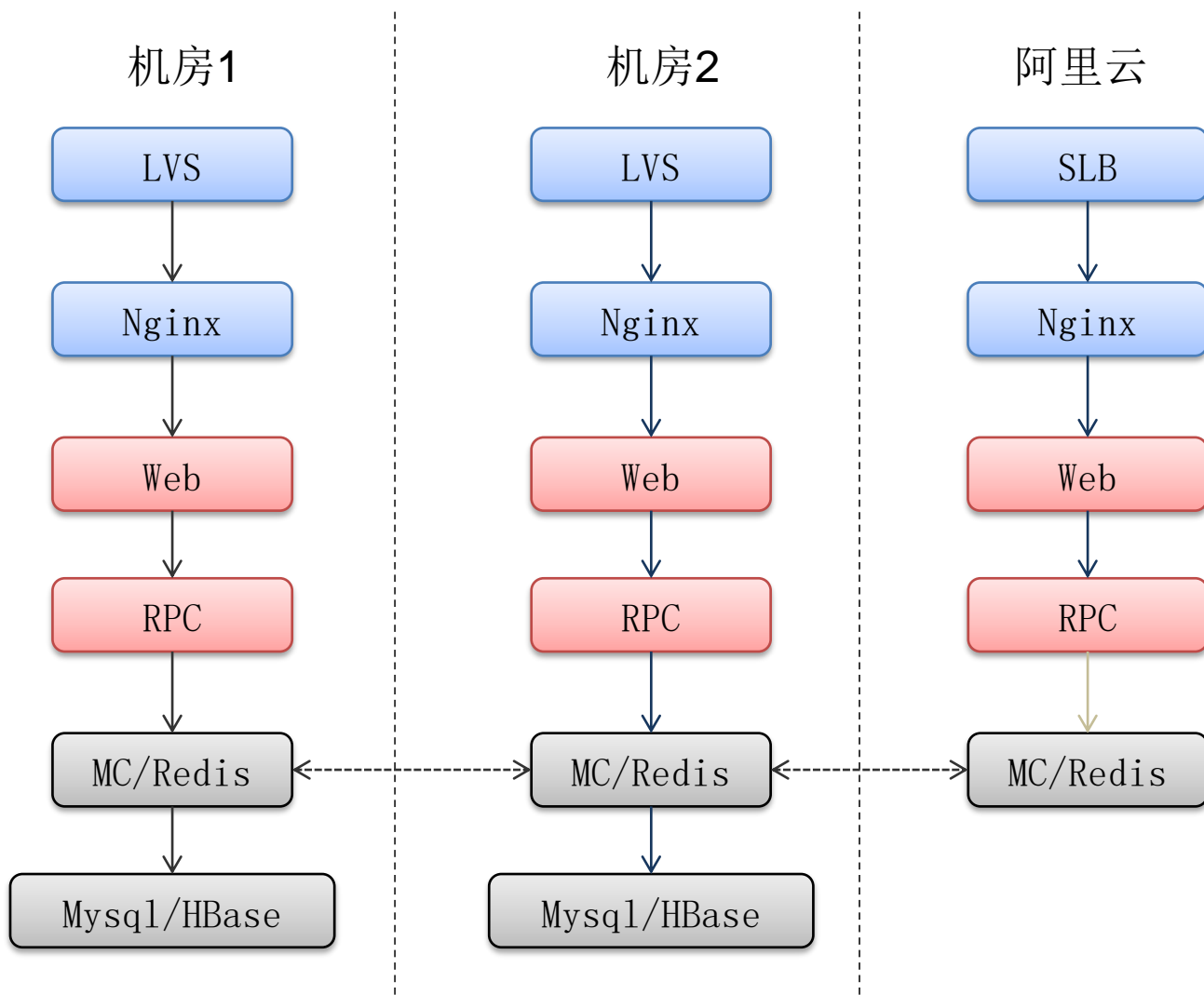
# 微博案例：混合云 - 核心关键是专线



# 微博案例：混合云部署方案一



# 微博案例：混合云部署方案二





# 微博案例：服务治理

四七层、RPC服务跨IDC、跨集群按权重流量切换

全链路在线压测

建立防御体系：

降级、分流、隔离…

监控、报警：快速发现定位

资源监控（WMB、MCQ、MC…）

服务池与单机监控（可用性监控、业务指标（Slow、Top、SLA…）、系统指标）

# 微博DCP开源





**G***devops*

# 全球敏捷运维峰会



THANK YOU !