China · Beijing

ArchSummit全球架构师峰会北京站

张斌

好雨云CTO

ArchSummit 全球架构师峰会 2016

[北京站]

主办方 Geekbang. InfoQ

China · Beijing

40分钟了解传统架构如何过渡到微服务架构

[北京站]

主办方 Geekbang. InfoQ

自我介绍

张斌

好雨联合创始人&CTO

曾就职人人网、大街网和澳客网

12年互联网开发和架构经验，擅长分布式系统和微服务架构

2016.7 参加深圳架构师大会

大 纲

1、如何选择架构

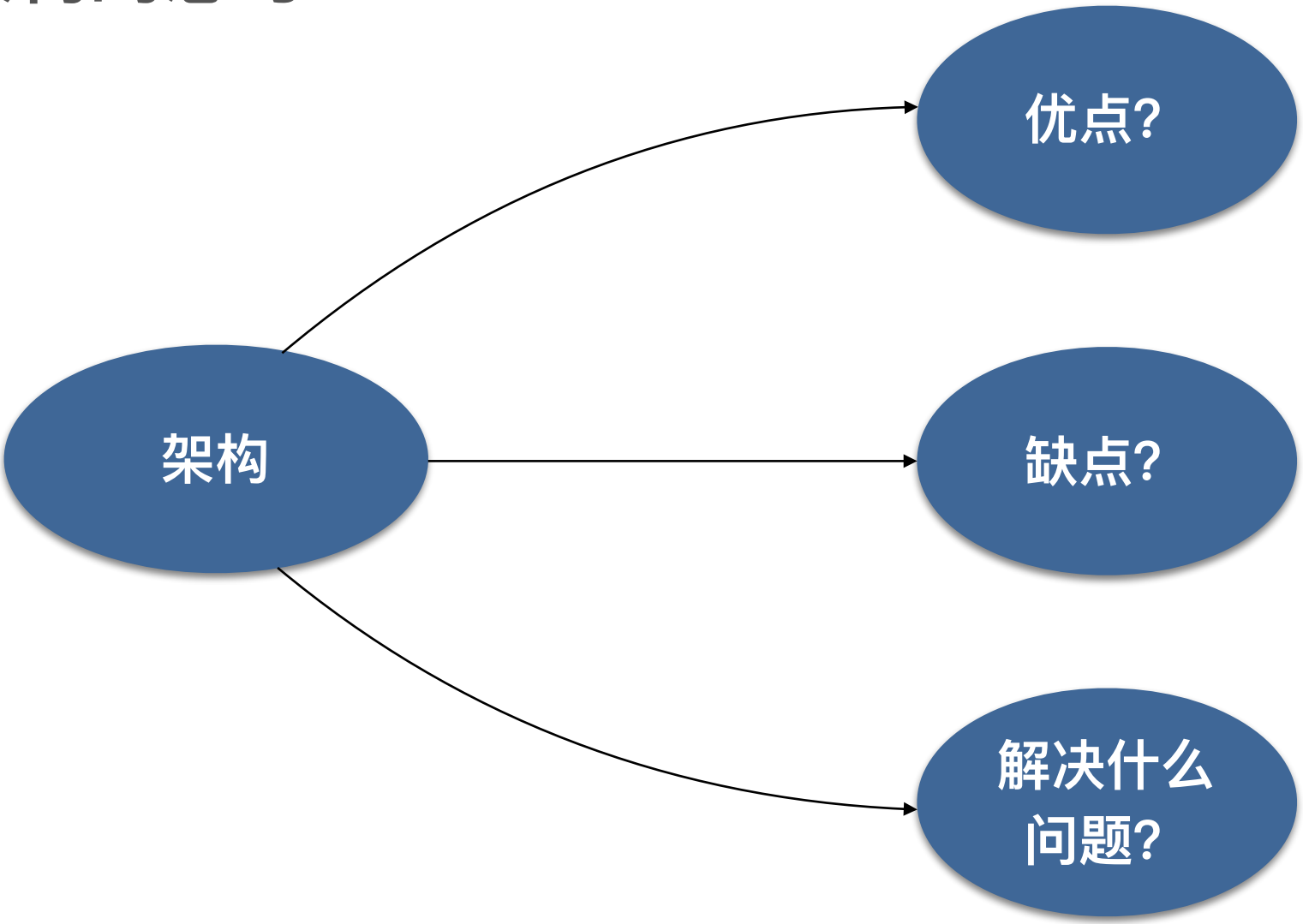
2、传统架构与微服务架构对比

3、微服务架构如何落地

4、客户案例

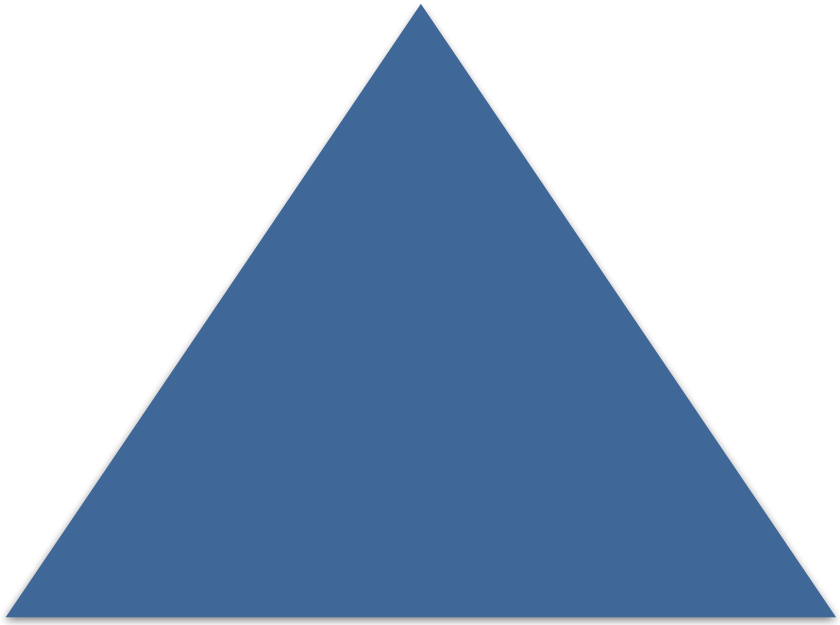
5、提问环节

架构的思考



理想的软件开发

架构



过程

敏捷

持续交付

自动化

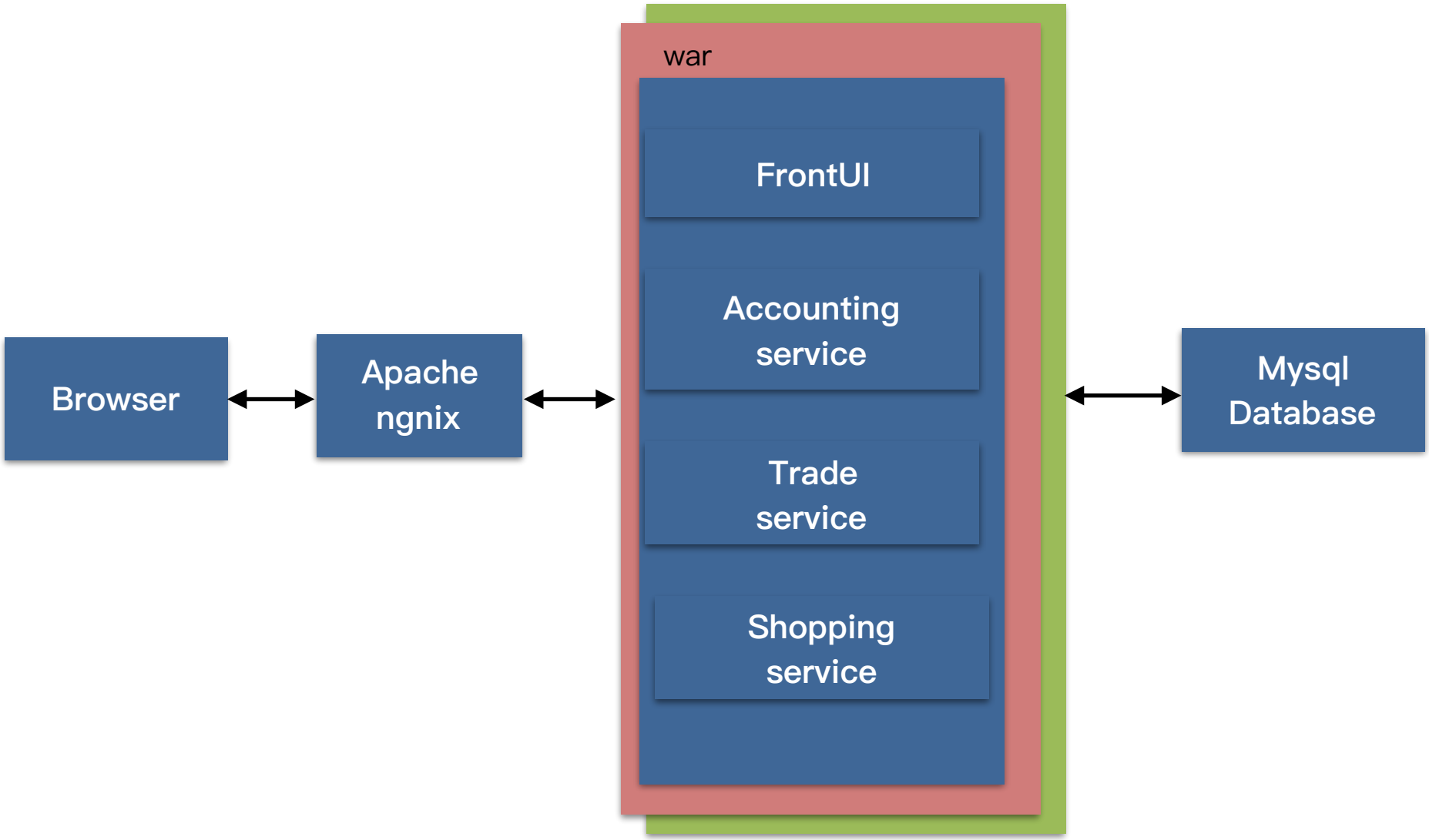
组织

规模小

自主性

小组化

传统软件开发


war
FrontUI
Accounting service
Trade service
Shopping service
Browser
Apache ngnix
Mysql Database

传统软件开发

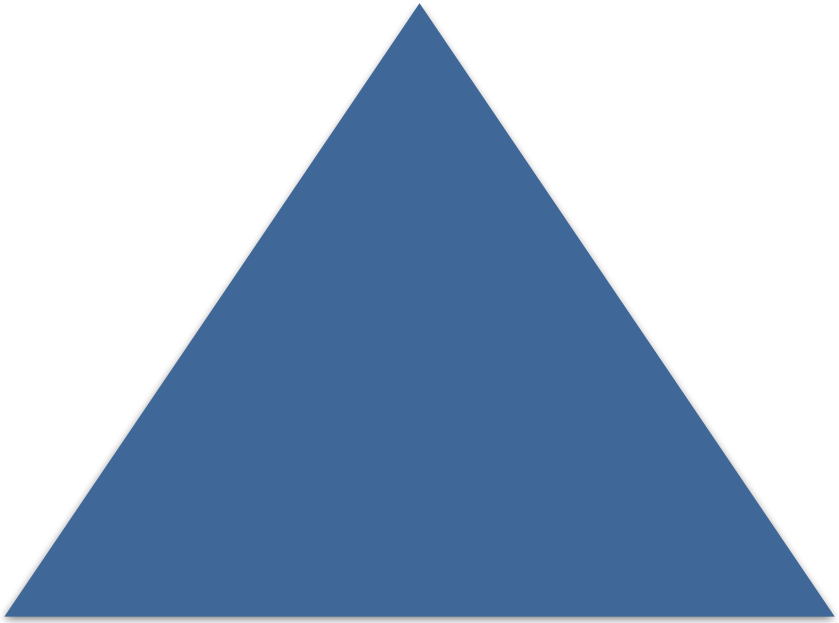
优点：

开发简单

部署简单

扩容简单

架构



缺点：

开发效率低

代码维护难

部署不灵活

稳定性不高

扩展性不够

过程

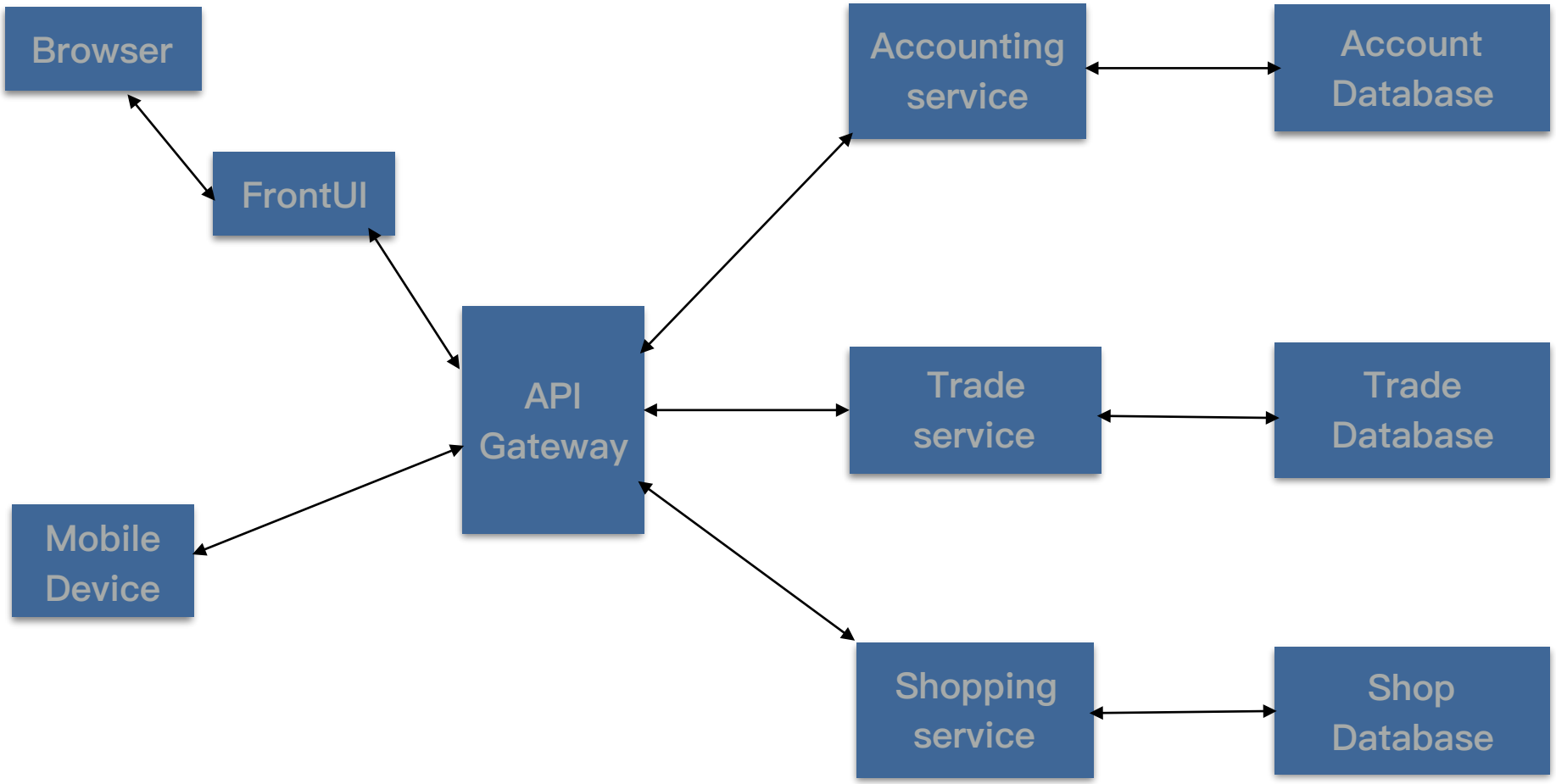
敏捷

组织

规模小

自主行

微服务架构

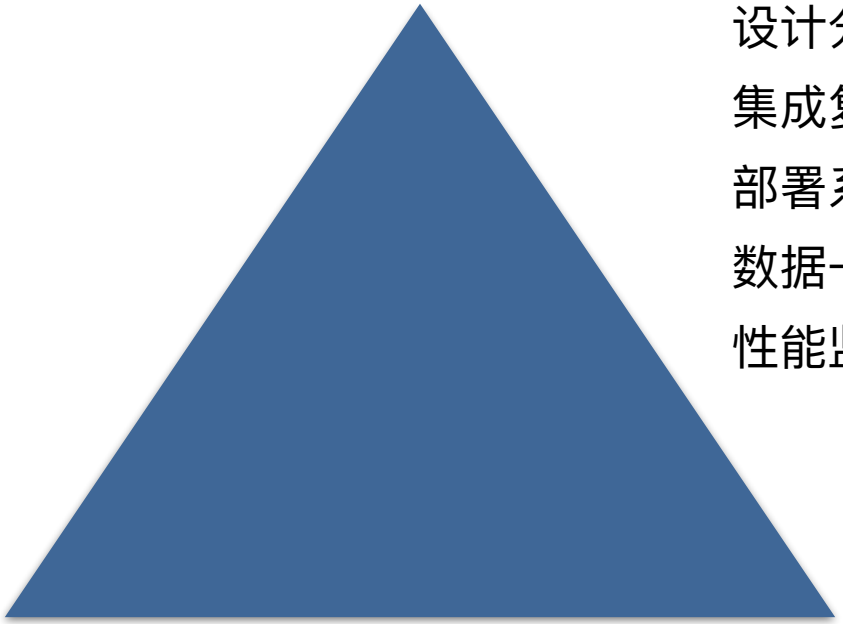

Browser
FrontUI
Mobile Device
API Gateway
Accounting service
Account Database
Trade service
Trade Database
Shopping service
Shop Database

微服务架构特点

优点：

- 语言无关性
- 独立进程通讯
- 高度解偶
- 任务边界固定
- 按需扩展

架构



缺点：

- 设计分布式系统复杂度
- 集成复杂度
- 部署系统复杂度、devops
- 数据一致性
- 性能监控

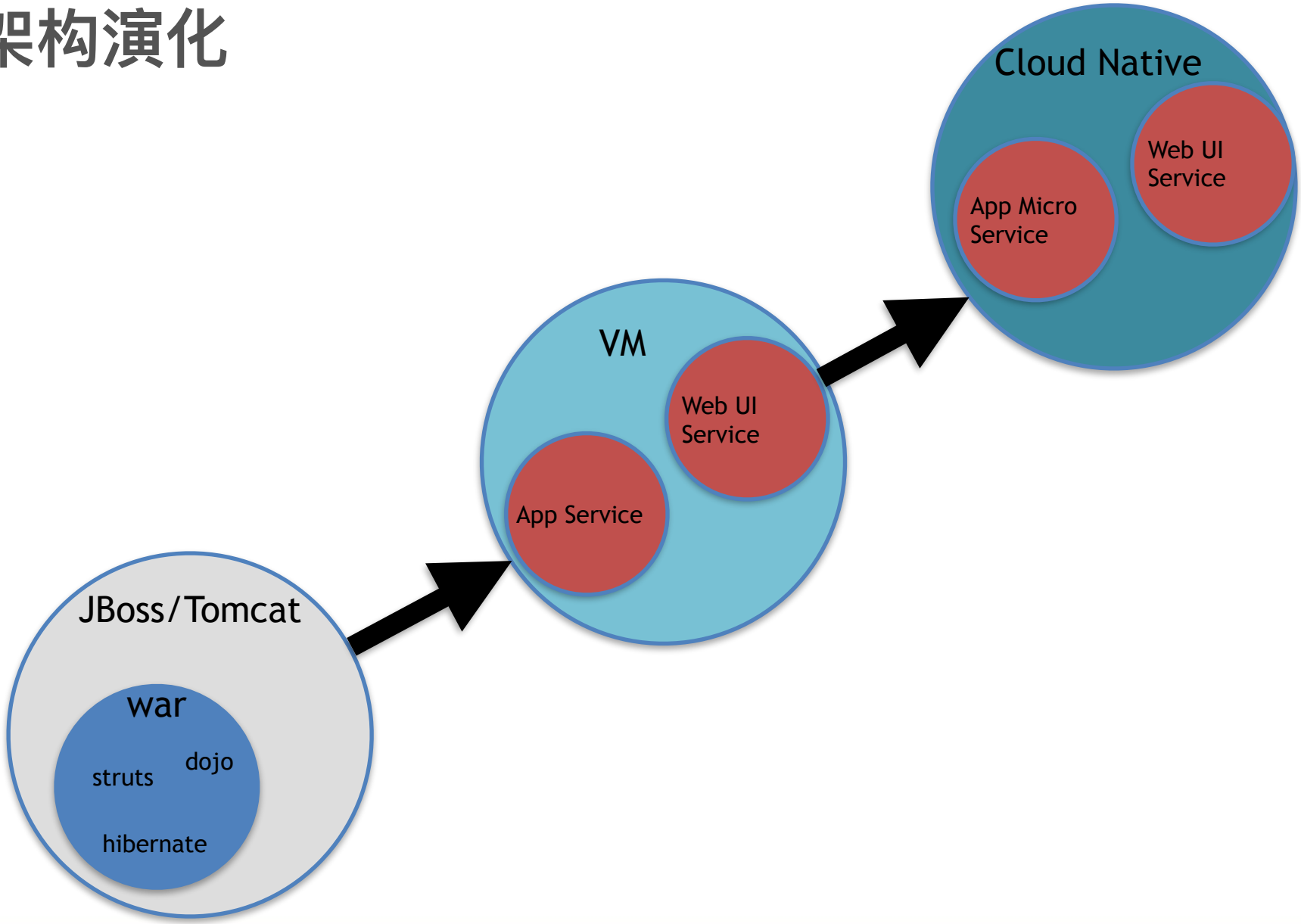
处理

- 敏捷
- 持续交付
- 自动化

组织

- 规模小
- 自主性
- 小组化

架构演化


Cloud Native
Web UI Service
App Micro Service
VM
Web UI Service
App Service
JBoss/Tomcat
war
dojo
struts
hibernate

微服务定义

微服务架构（Microservices Architecture）是将应用拆分成小业务单元开发和部署，使用轻量级协议通信，通过协同工作实现应用逻辑的架构模式。

微服务使用者


Google


Netflix Inc.
NETFLIX


amazon.com


ebay


GILT


HAIL

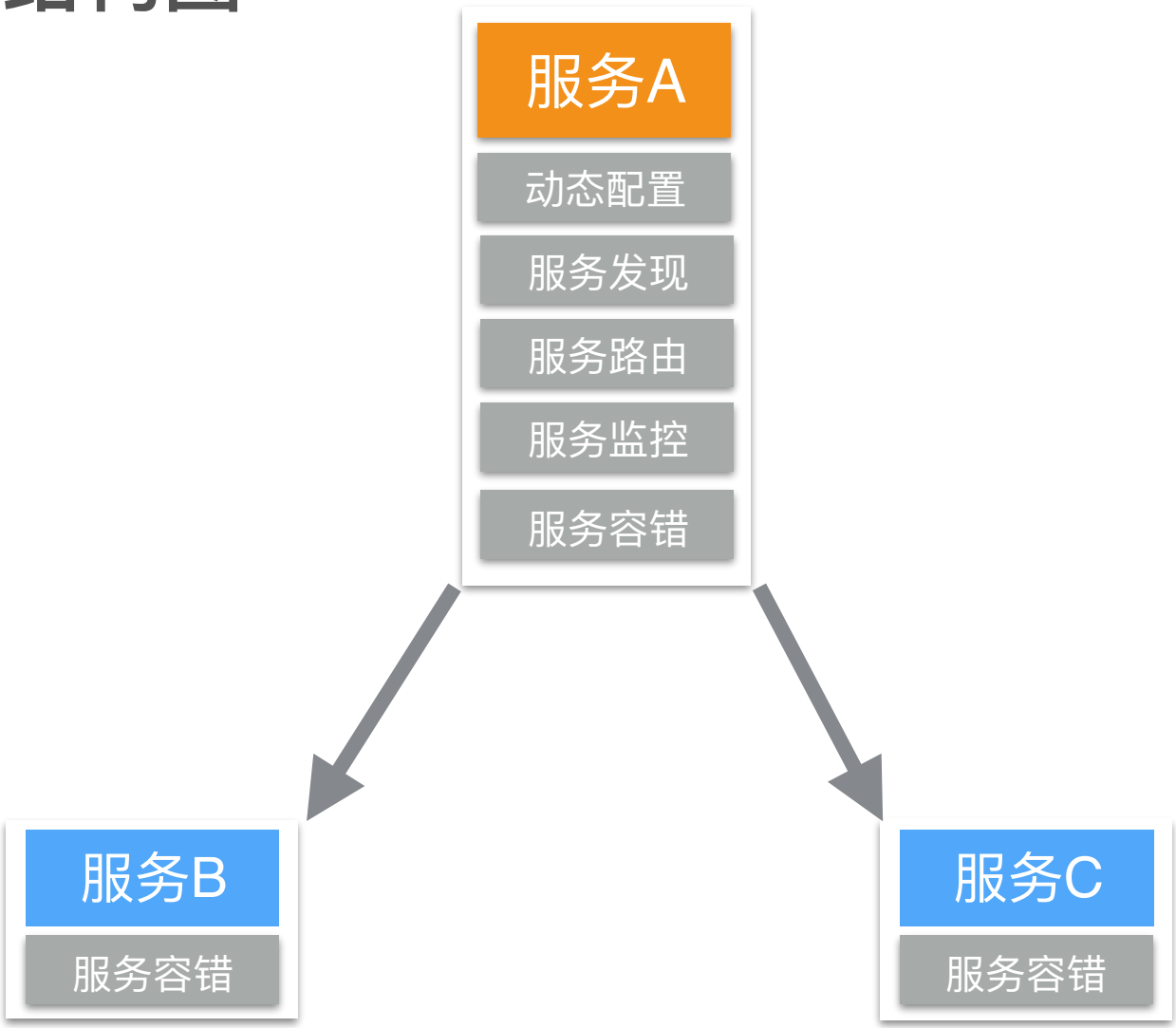

Uber Technologies Inc.
UBER


好雨
goodrain

微服务特征

服务组件化	应用拆分成不同服务运行在不同进程中，明确定义服务边界
按业务划分服务与组织	业务领域的全栈（从前端到后端）软件实现
去中心化	针对业务特征选择不同的技术水平，针对性的解决问题
设计失败	服务的消费方需要优雅的处理错误
智能终端	业务逻辑在服务内部处理；服务间通信尽轻量化，不添加任何额外的业务规则
自动化	开发、调试、测试、集成、监控和发布
API Gate	统一暴露服务接口；对服务认证、授权、监控、路由等

服务结构图



微服务落地条件

一、如何访问这些服务?

二、服务如何发现?

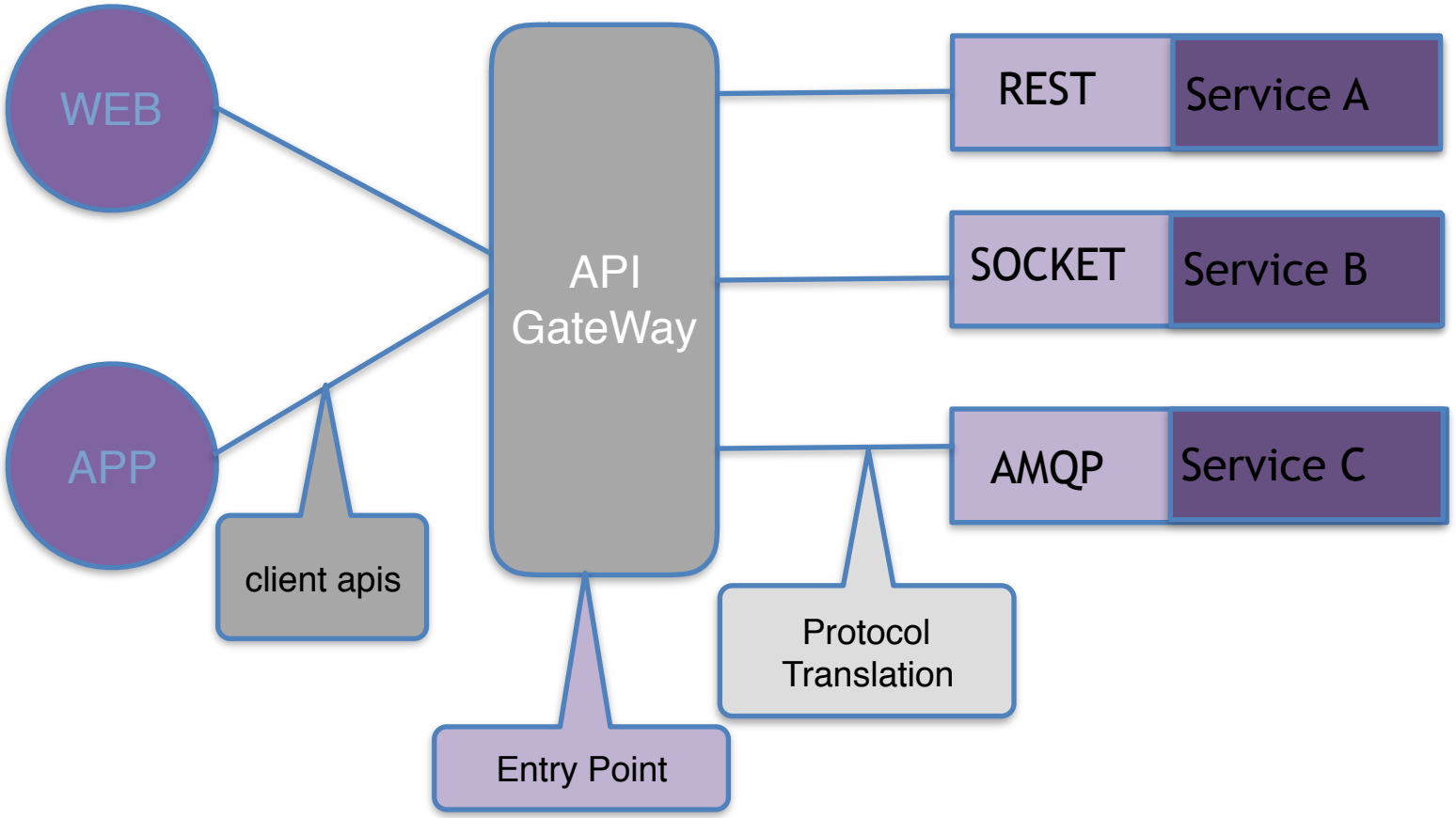
三、服务如何通信?

四、数据如何管理?

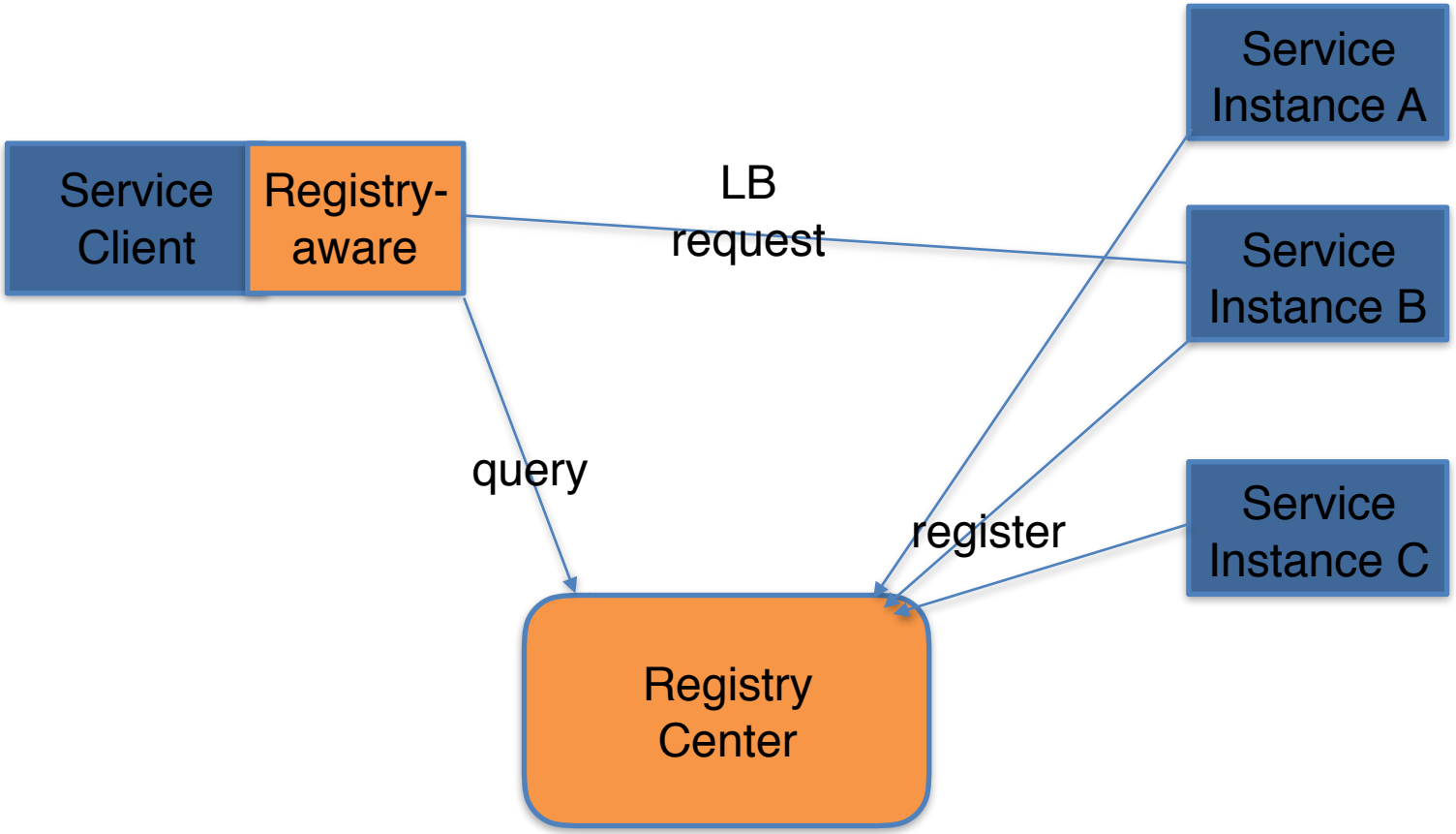
五、服务如何容错?

六、服务如何监控?

如何访问这些服务


WEB
APP
client apis
API
GateWay
Entry Point
Protocol
Translation
REST
Service A
SOCKET
Service B
AMQP
Service C

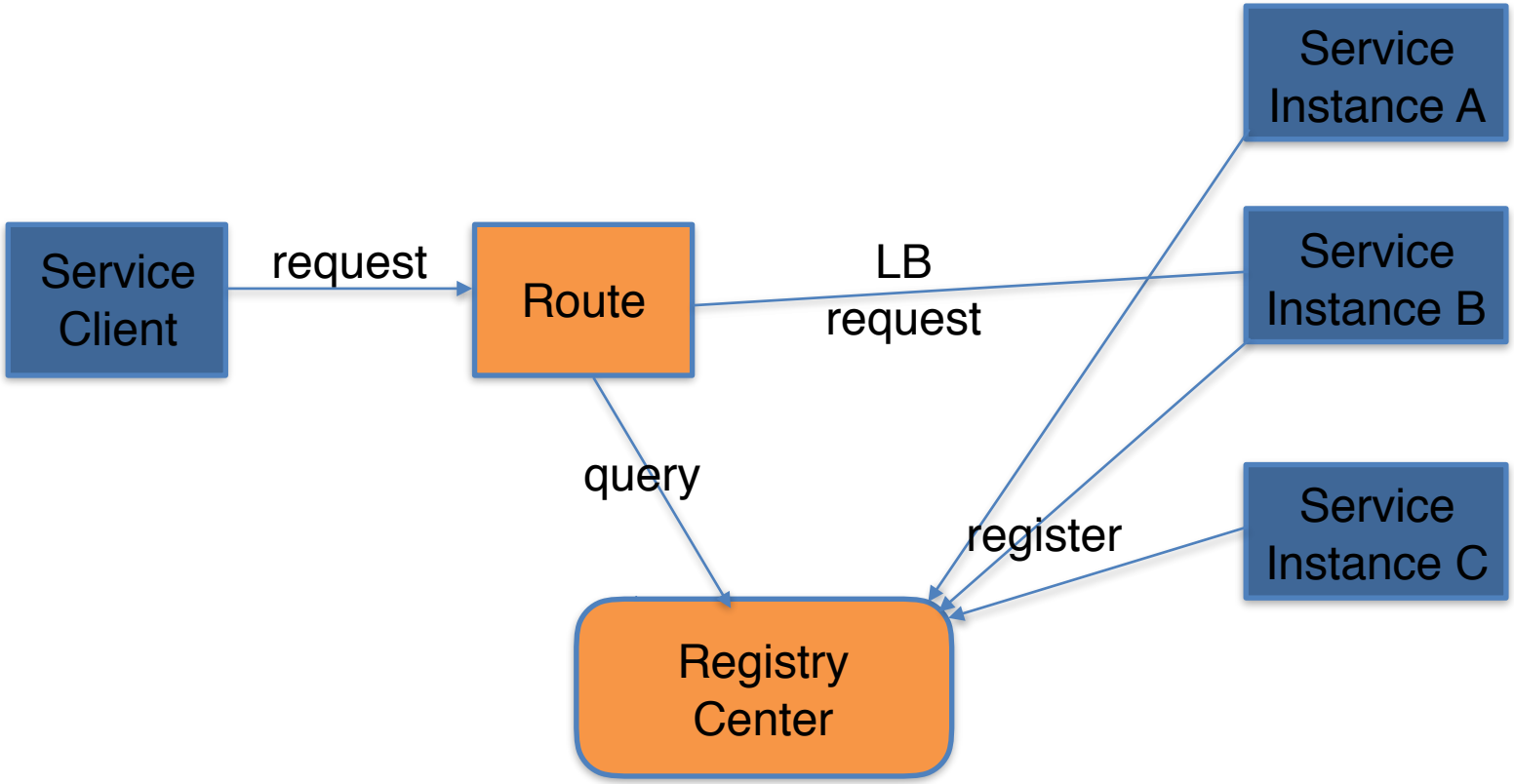
服务如何发现（一）



spring cloud

dubbo

服务如何发现（二）



Goodrain microservice

服务如何通信

	一对一	一对多
同步	请求/相应	——
异步	通知	发布 / 订阅
	请求/异步响应	发布/异步响应

同步调用：

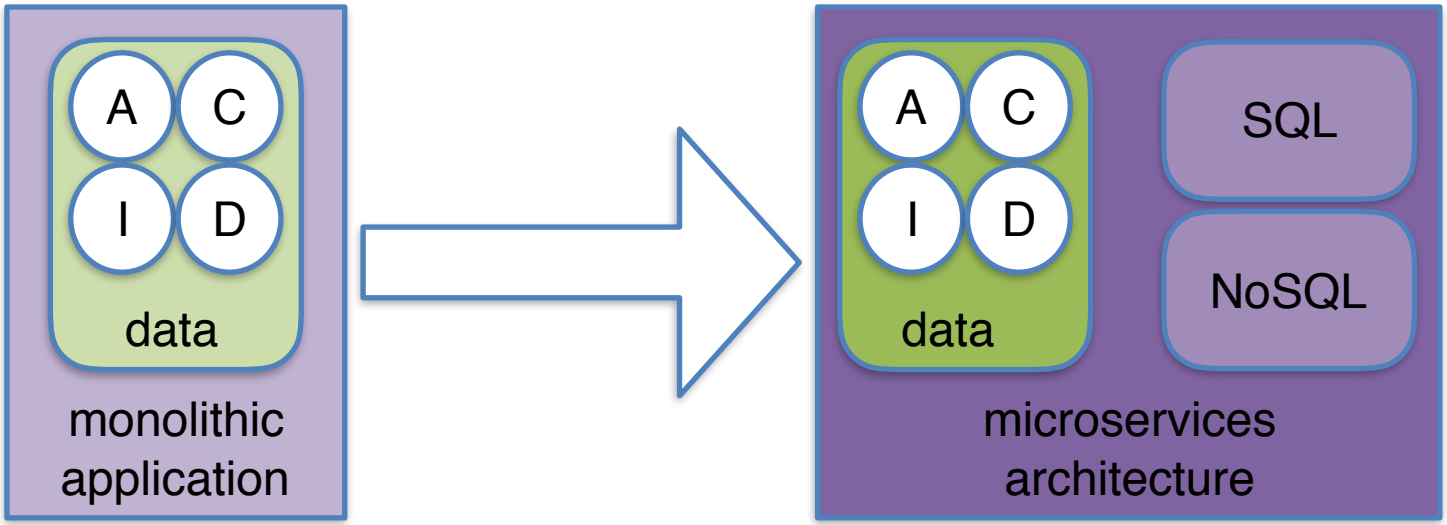
REST（JAX-RS，Spring Boot）

RPC（Thrift, Dubbo）

异步调用：

Akka Actor,Kafka, Notify, MQ

数据如何管理



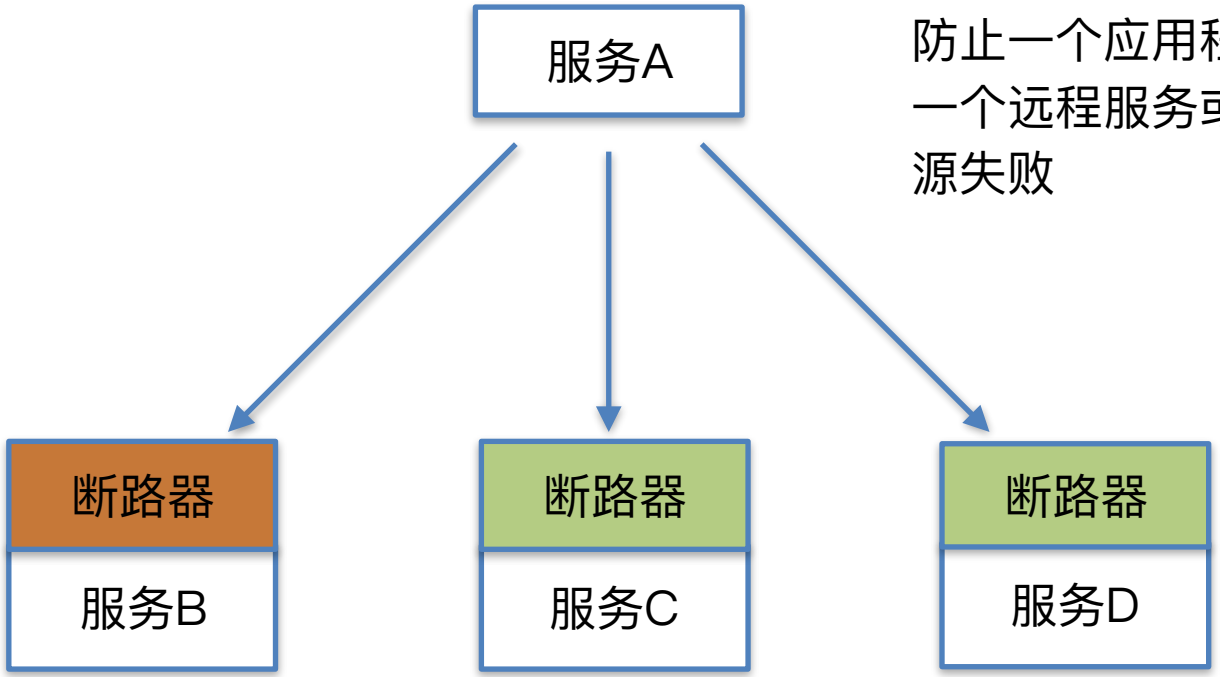
常见方式：

- 共享数据库
- 消息队列

事件驱动：

- Event Sourcing
- CQRS

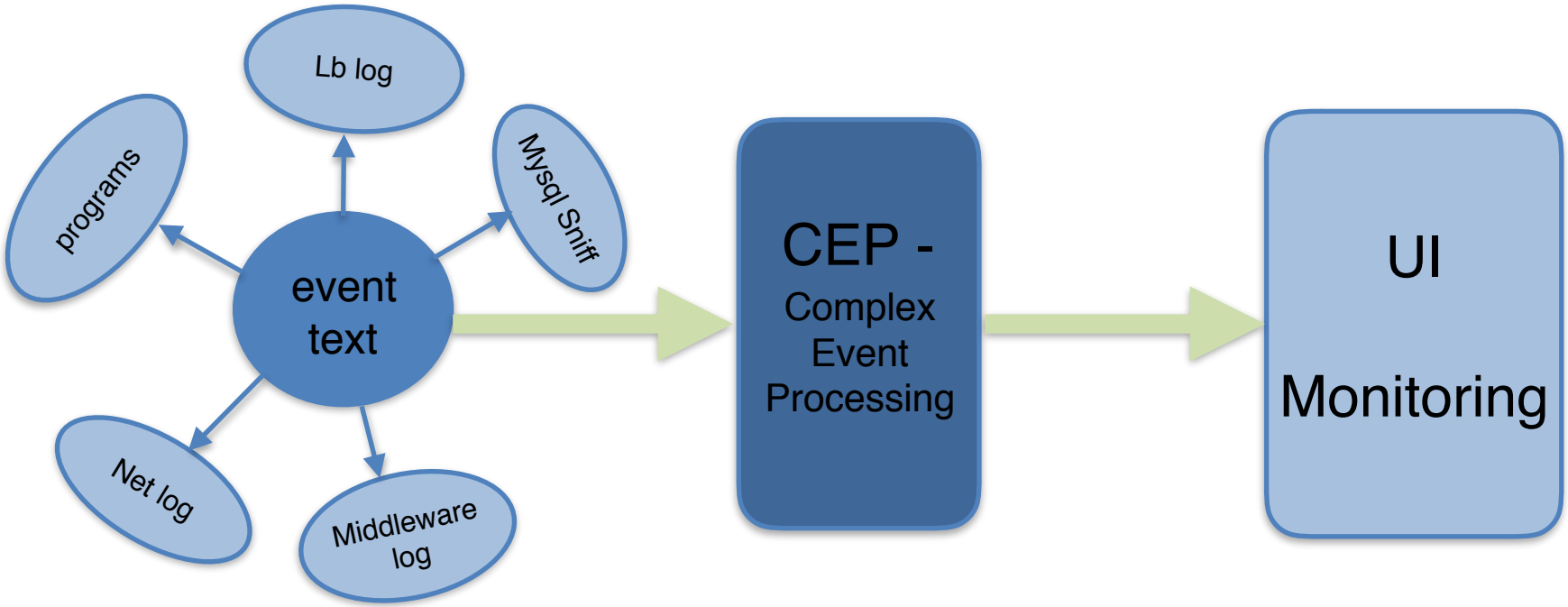
服务如何容错



防止一个应用程序试图调用一个远程服务或访问共享资源失败

异常处理、日志记录、测试失败操作、资源分化、并发 等等

服务如何监控


Lb log
programs
Mysql Sniff
event
text
Net log
Middleware
log
CEP -
Complex
Event
Processing
UI
Monitoring

常见设计范式

流水号模式

元数据驱动

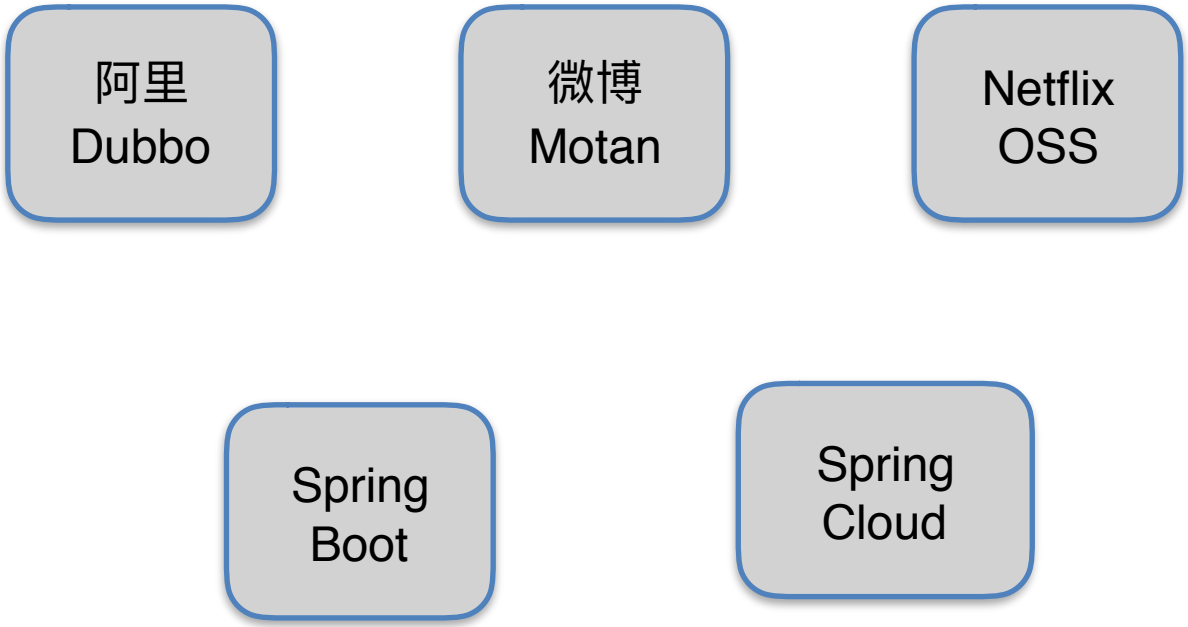
异步事件

服务无状态

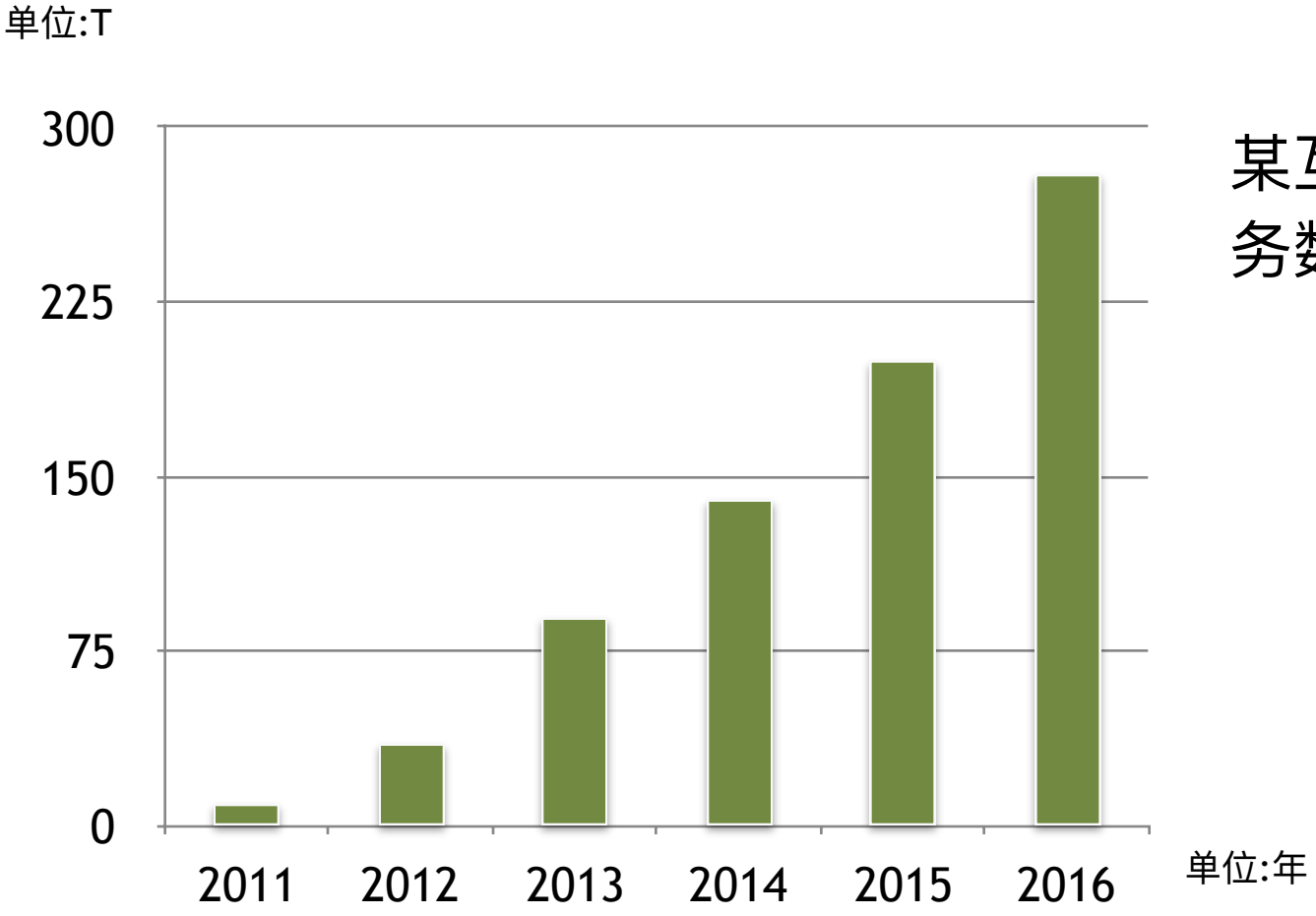
编排模式

统一配置

常见微服务框架

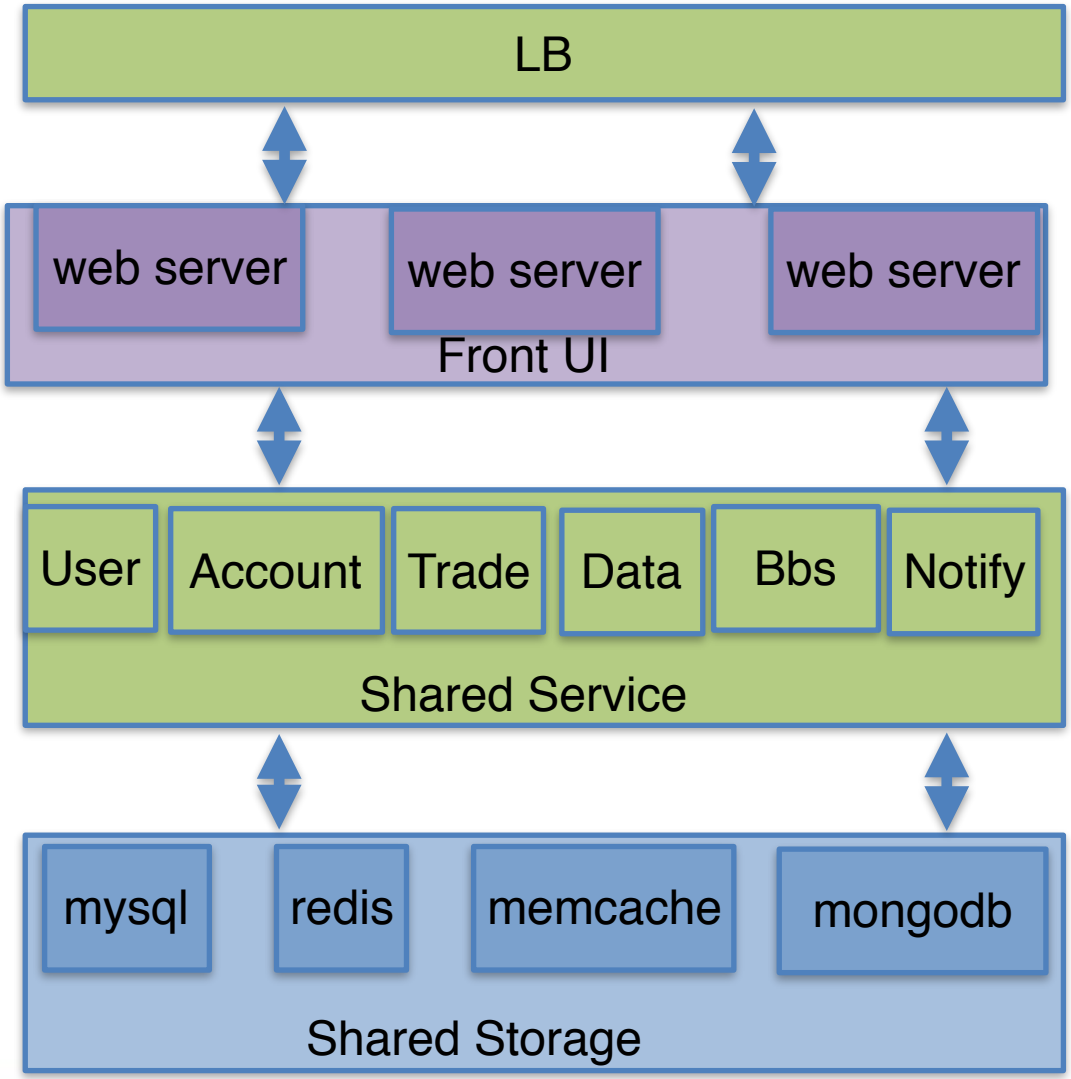


客户案例



某互联网平台业务数据增长趋势

历史现状



1、服务耦合在一起

2、业务数据共享

3、共享一个项目

4、产品更新周期长

5、错误排查难

6、无业务监控

7、伸缩困难

期待愿景

项目快速迭代

排查问题简单

业务级监控

高可用

应对业务快速增长

跟上技术步伐

系统拆分

业务拆分	核心业务与边缘业务拆分
	前后端交易分离
	数据分析与数据展示拆分
	用户体系和账户体系分离
架构拆分	服务无状态化
	基于事件机制异步处理
	数据进行sharding、冷热分离
	前后端开发分离
	采用微服务架构

技术选型

开发语言	Php、Java、Scala、C++、Golang
数据存储	Mysql、Redis、MongoDB
消息队列	RabbitMQ、ZeroMq
使用框架	AKKA、ThinkPhp、Spring、Dubbo、Cep、Angularjs
数据分析	Hadoop、Spark、R
运行环境	Docker、Kuberntes

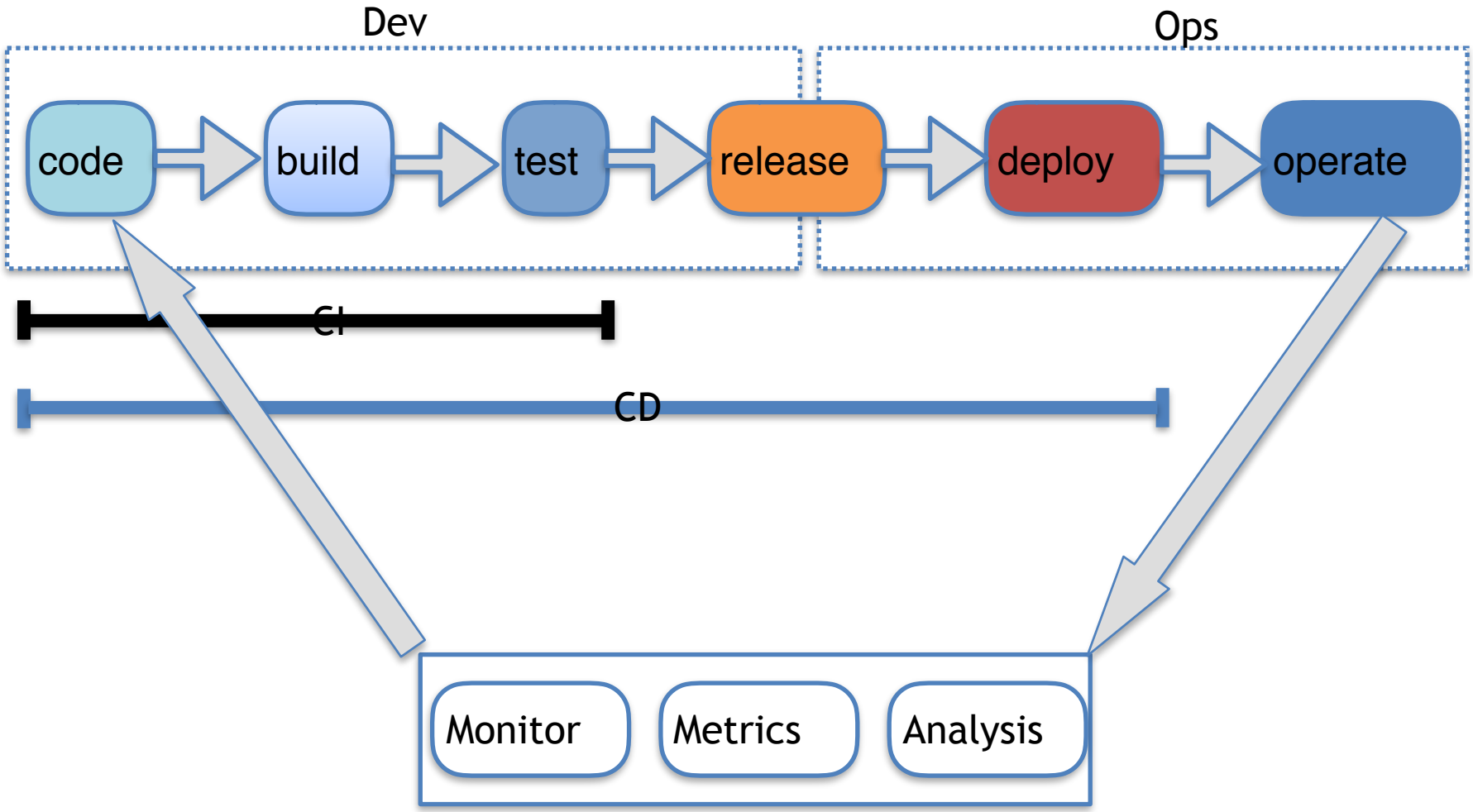
微服务实施

程序逻辑改造

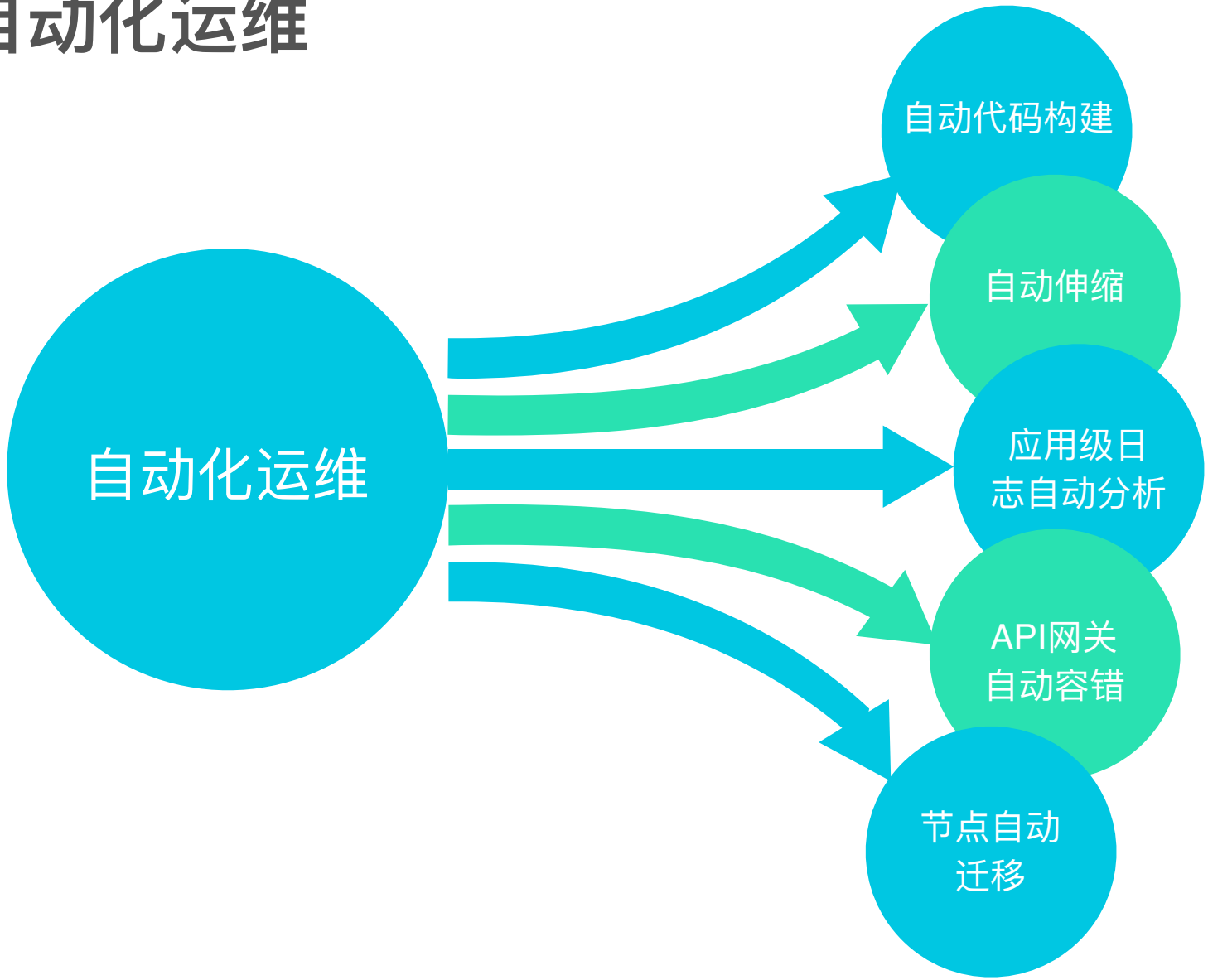
DevOps流程搭建

自动化运维实施

DevOps


Dev
Ops
code
build
test
release
deploy
operate
CI
CD
Monitor
Metrics
Analysis

自动化运维



踩过的坑

数据存储、一致性

冷热数据加载

docker中Dubbo 注册IP

docker日志收集

服务发现、监控、容错

THANKS

ArchSummit
全球架构师峰会 2016

[北京站]

主办方 Geekbang InfoQ