

# 微服务下的应用架构设计

Clean Object-oriented & Layered  
Architecture

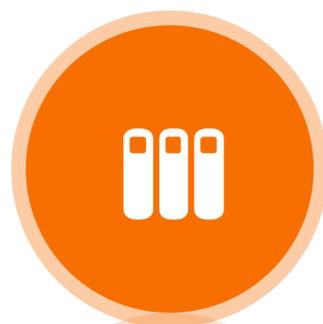
张建飞

零售通技术部-高级技术专家

2020/08/18-08/19



关注“阿里巴巴云原生”公众号  
获取第一手技术资料



## 微服务

什么是微服务?



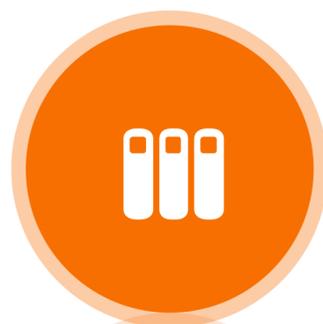
## 服务拆分

服务如何拆分?



## 应用架构

应用架构如何设计?



## 微服务

什么是微服务?



## 服务拆分

服务如何拆分?



## 应用架构

应用架构如何设计?

# 什么是微服务？



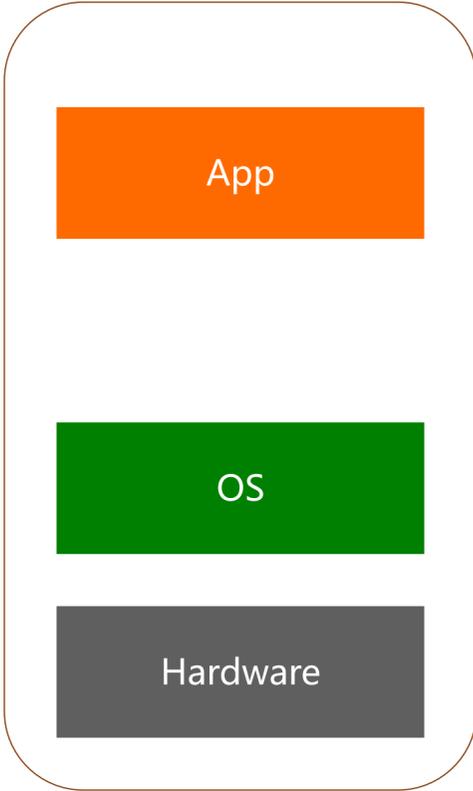
微服务是涉及业务+技术的综合性问题

# 微服务-底层技术-演进之路

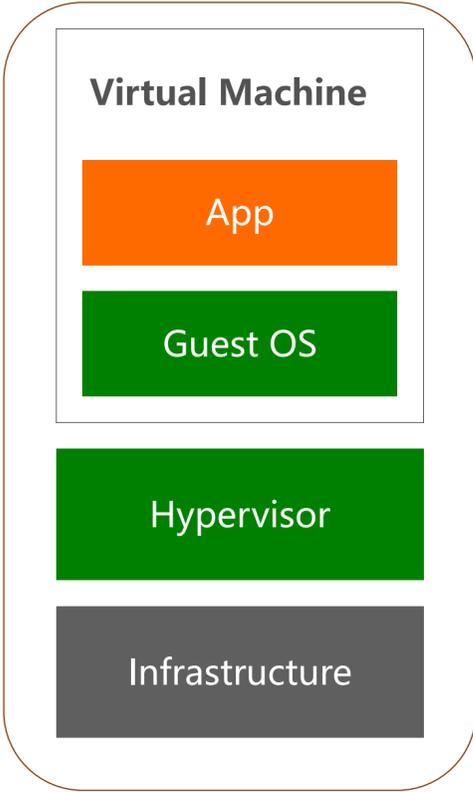
## 应用底座



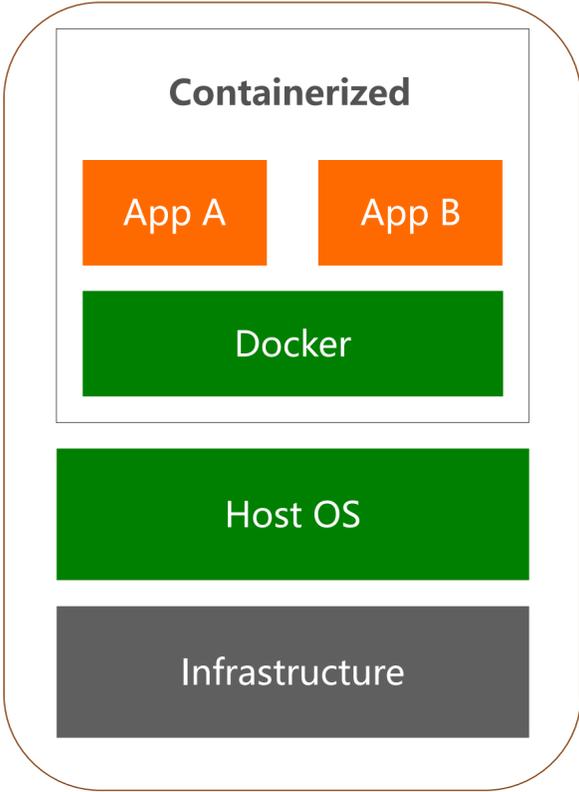
OS



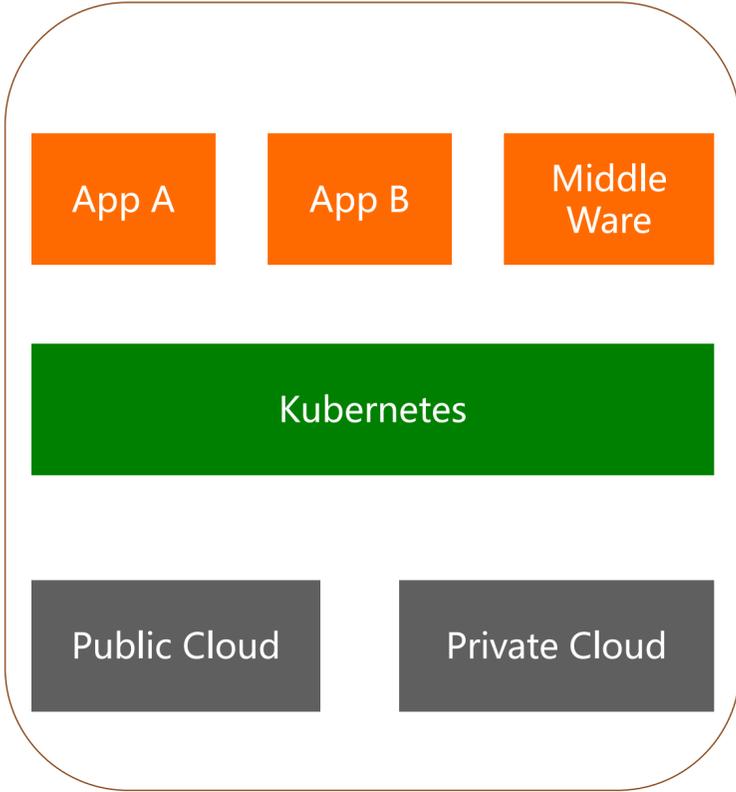
VM



Docker



Cloud Native



# 微服务-业务技术-演进之路

## 应用自身



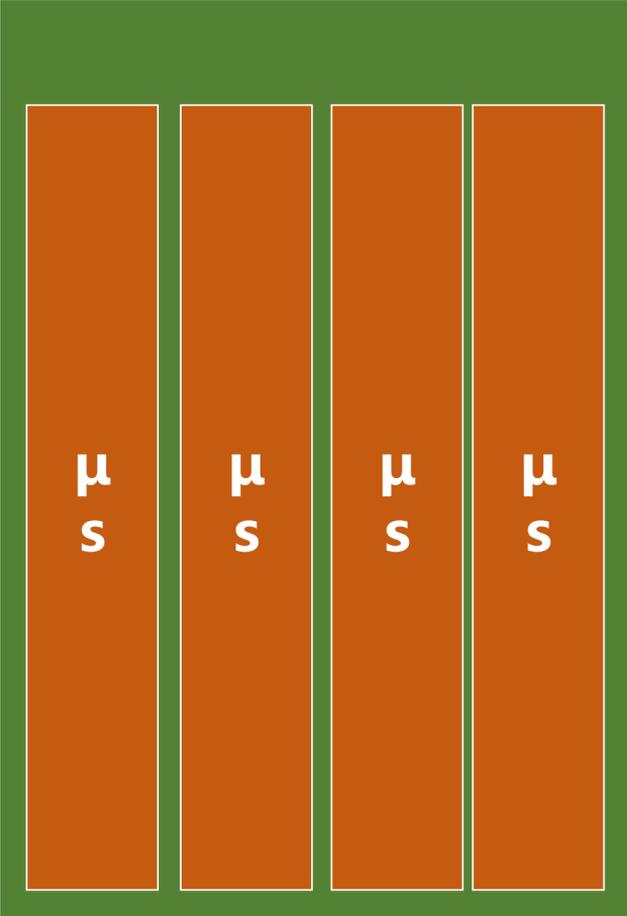
Monolithic



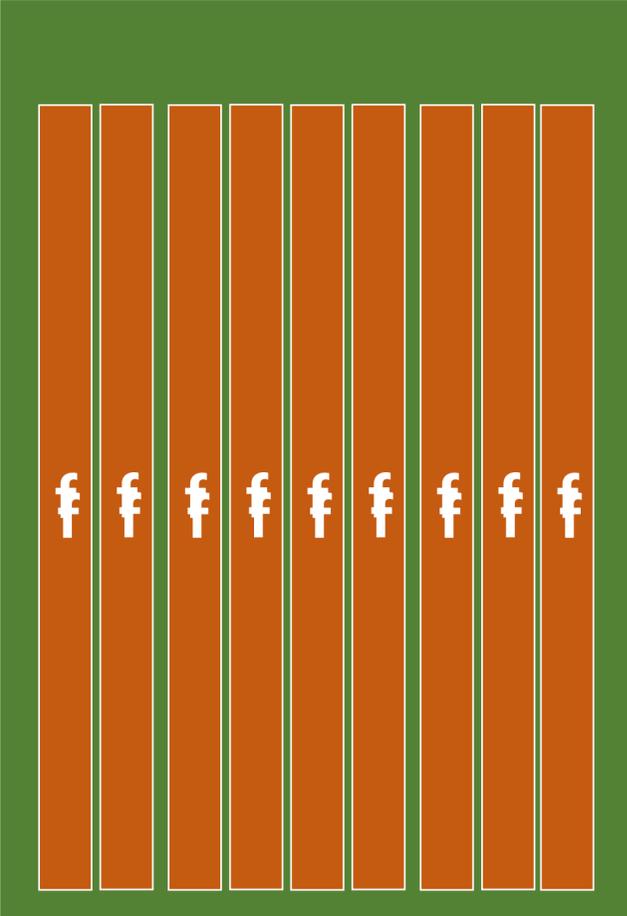
SOA



Micro Services

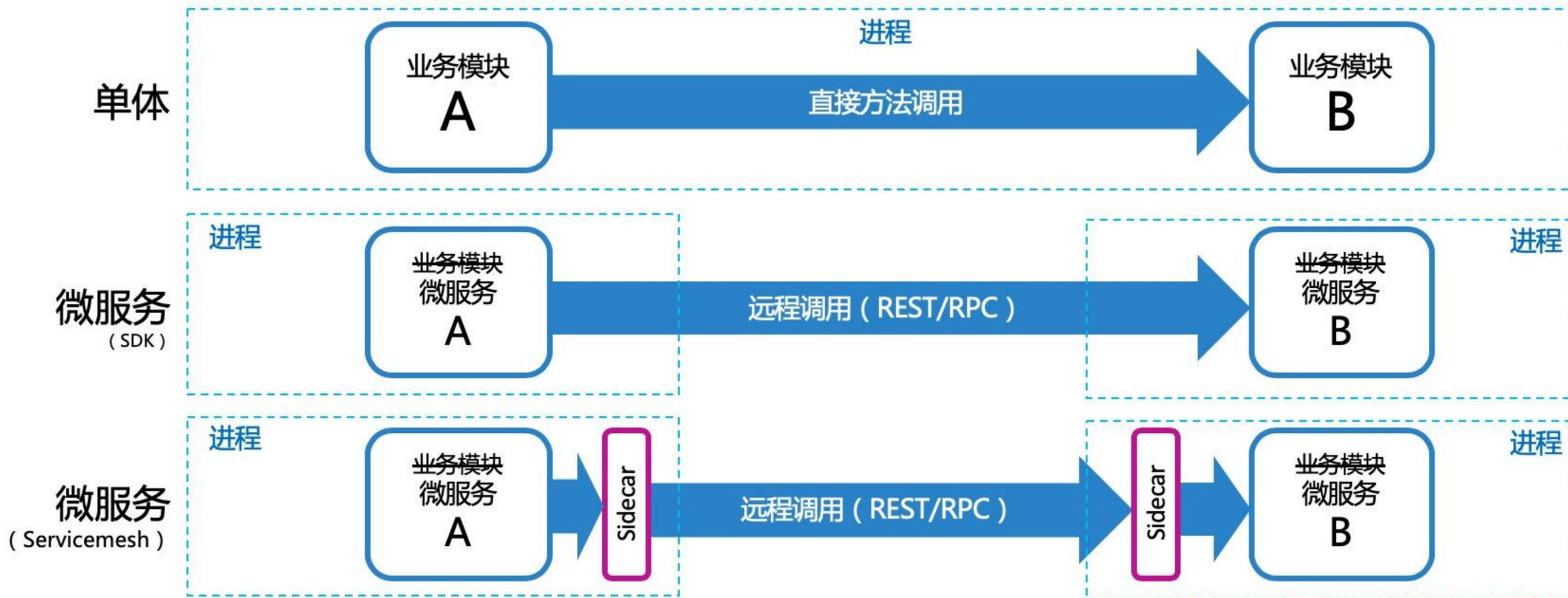


FaaS

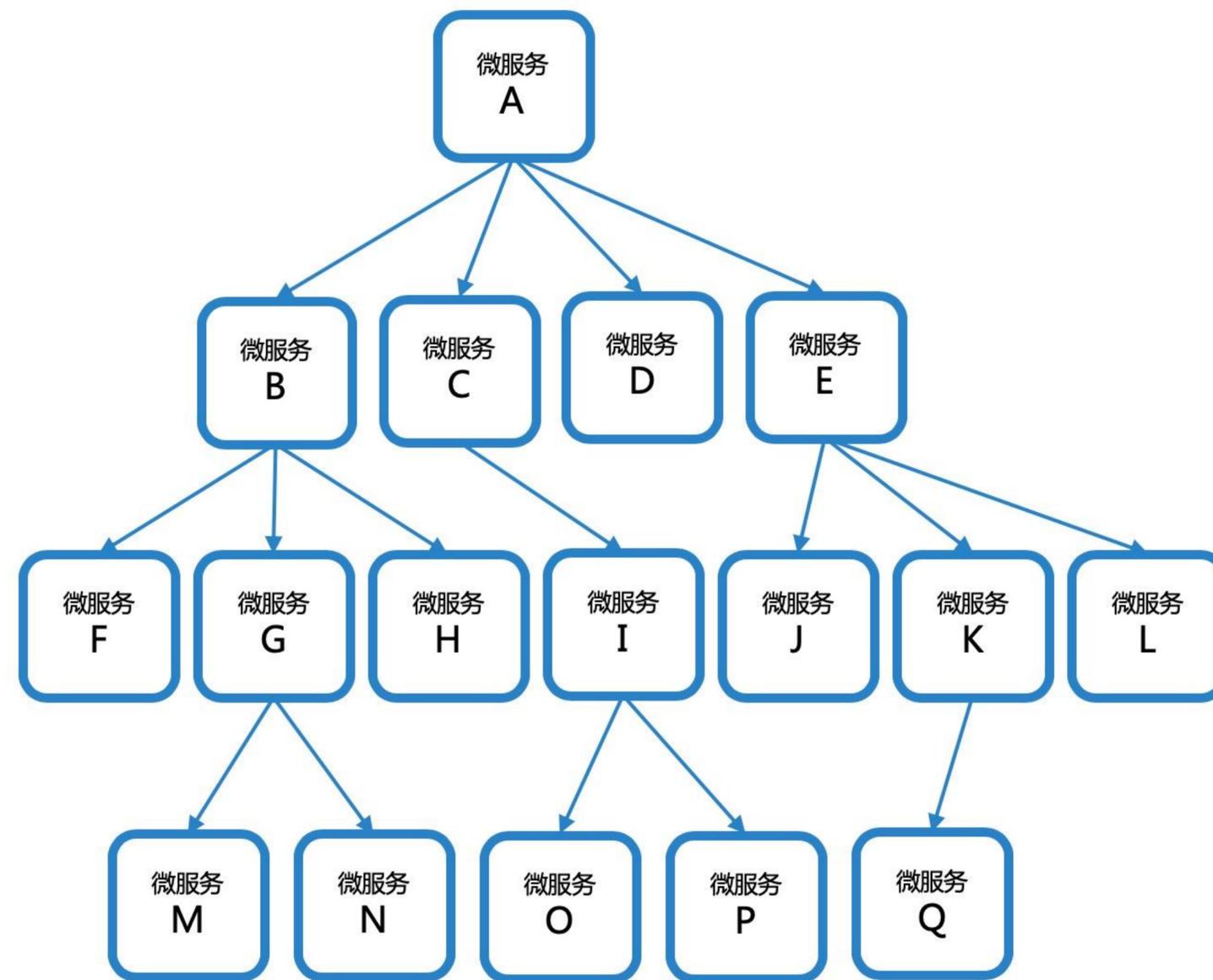
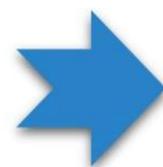
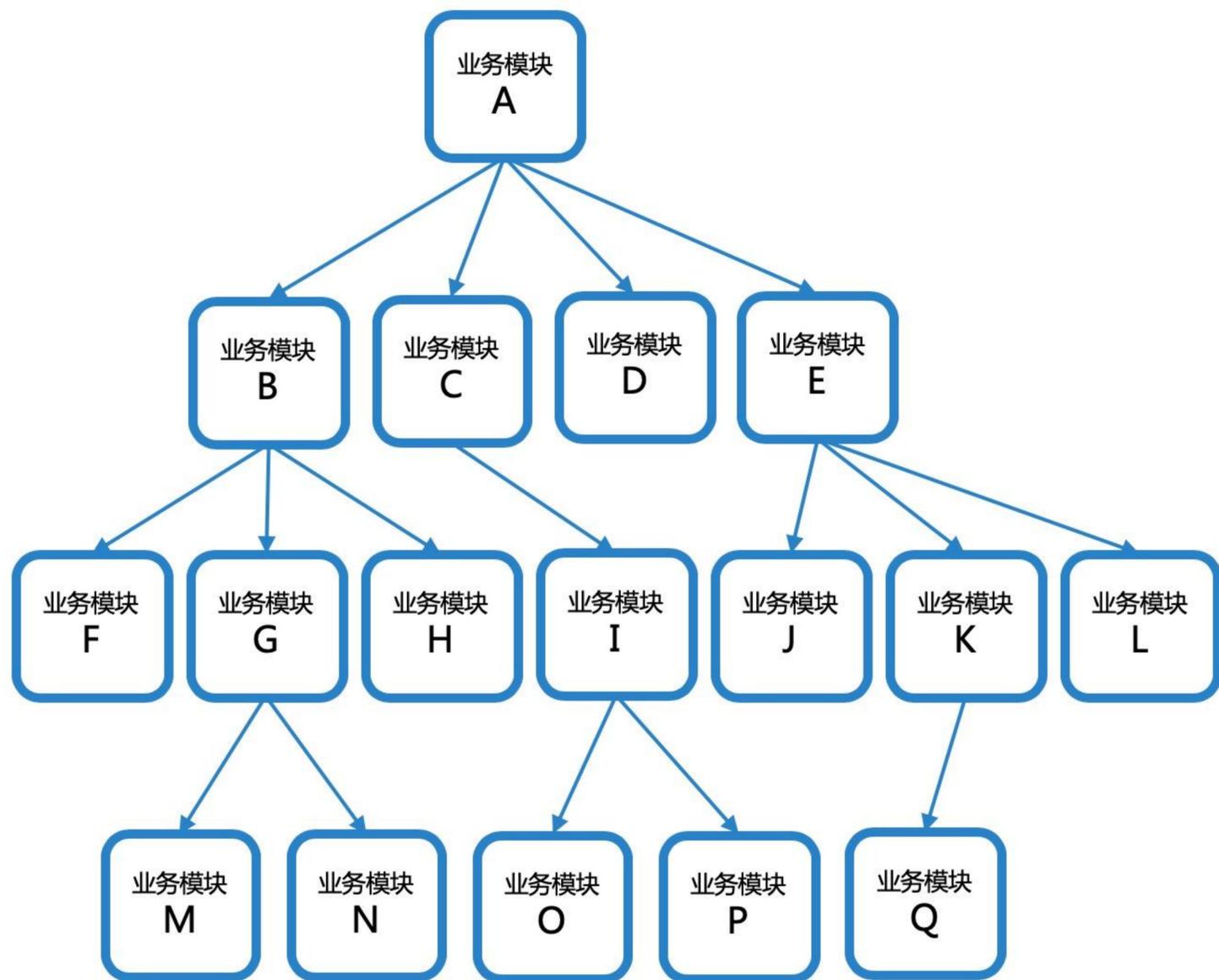


微  微小？

# 微服务和单体的本质差异



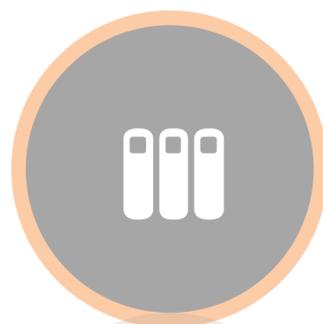
# “分布式单体” 陷阱



# 总结：微服务成功的关键

1. 功能完备的基础实施。  
(云原生)

2. 正确的认知。  
(服务拆分 + 应用架构)



## 微服务

什么是微服务?



## 服务拆分

服务如何拆分?



## 应用架构

应用架构如何设计?

# 服务拆分的三个考量维度

## 功能维度

- 纵向
- 横向

## 质量维度

- 可靠性
- 性能
- 易变性

## 团队维度

- 组织结构
- 人员数

# 功能维度：纵向



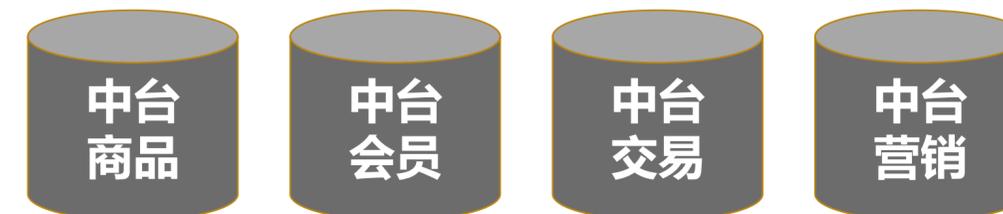
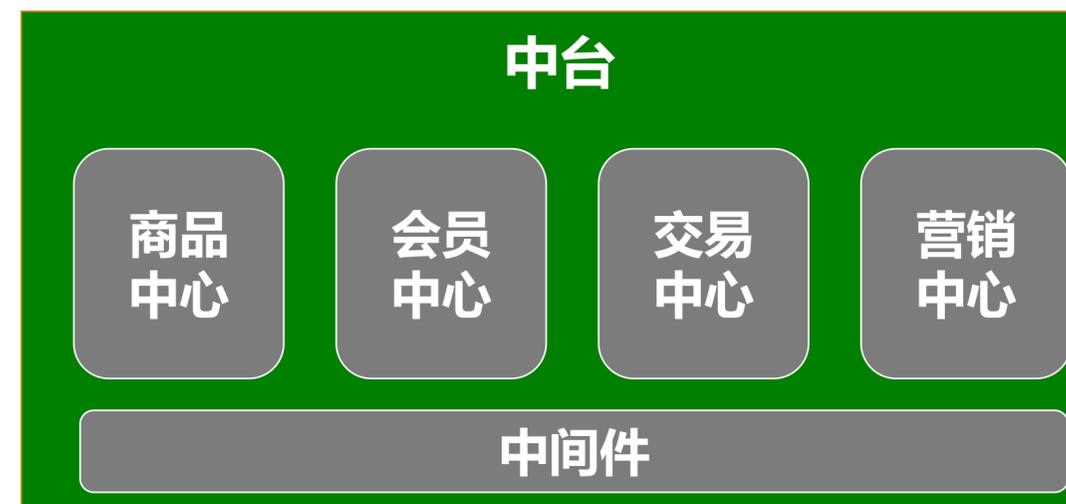
DDD领域划分



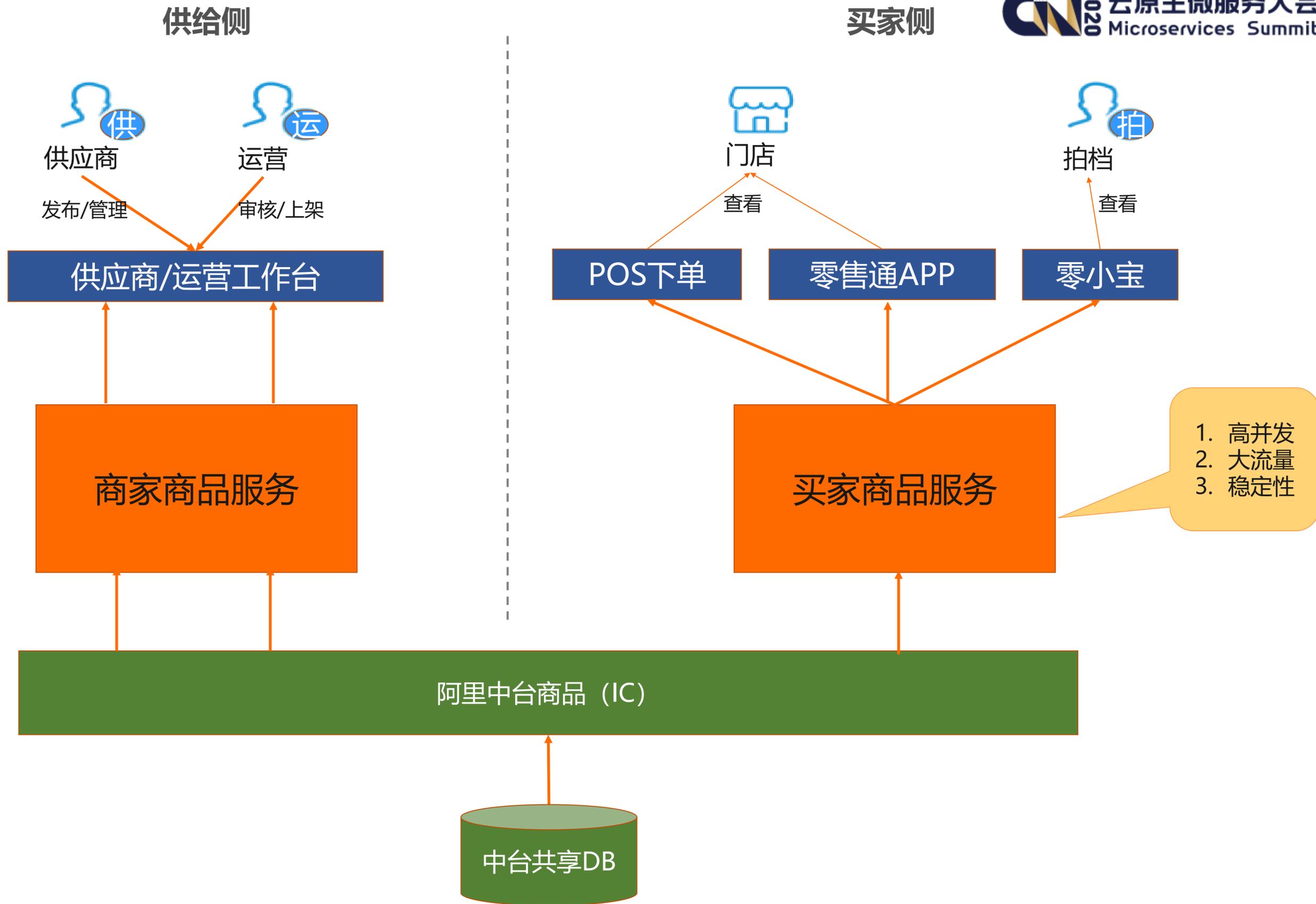
业务功能：纵向



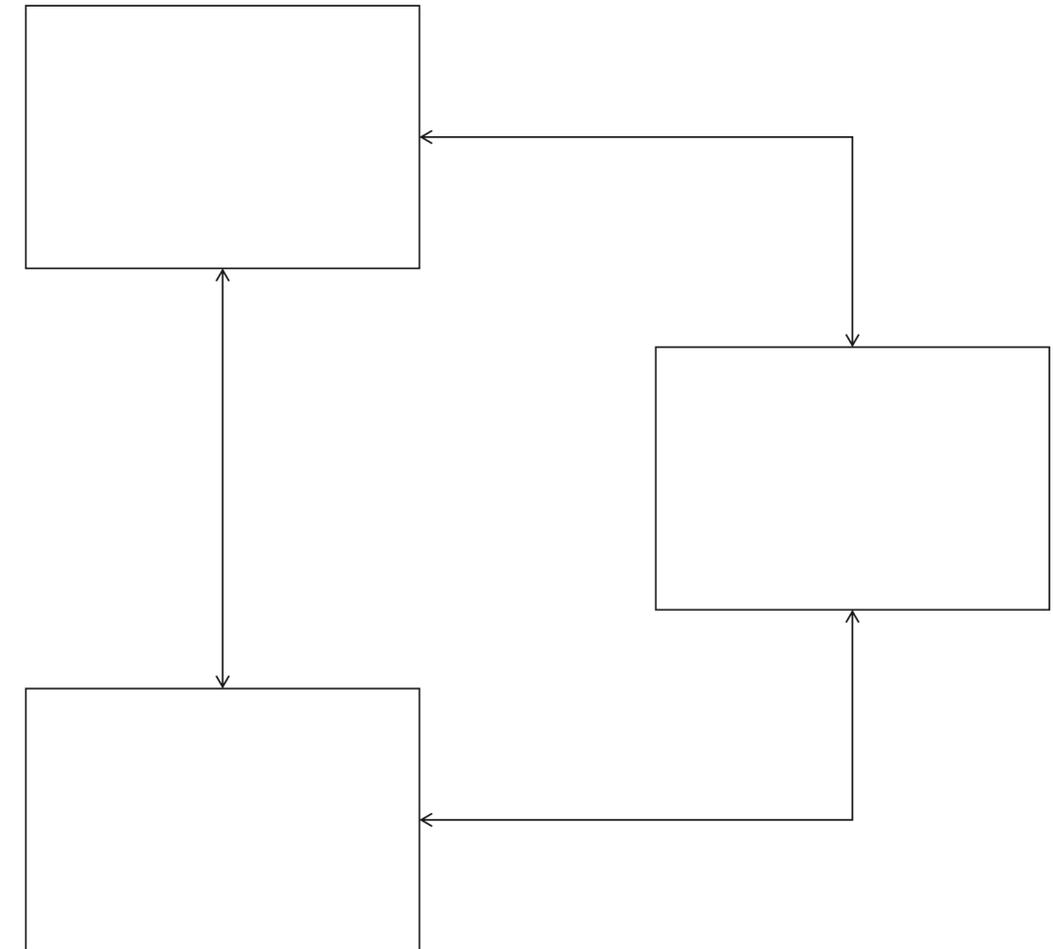
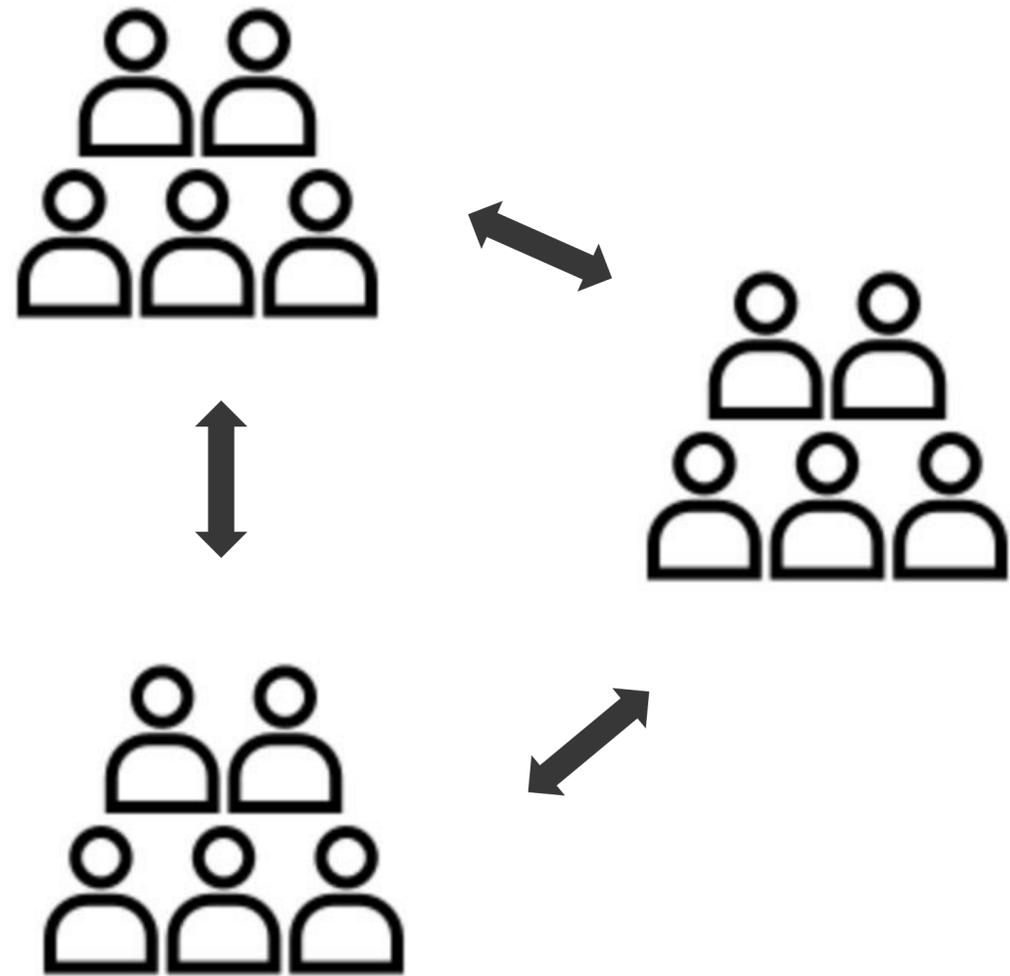
# 功能维度：横向



# 质量维度



# 团队维度



康威定律：Communication dictates design (组织沟通方式决定系统设计)

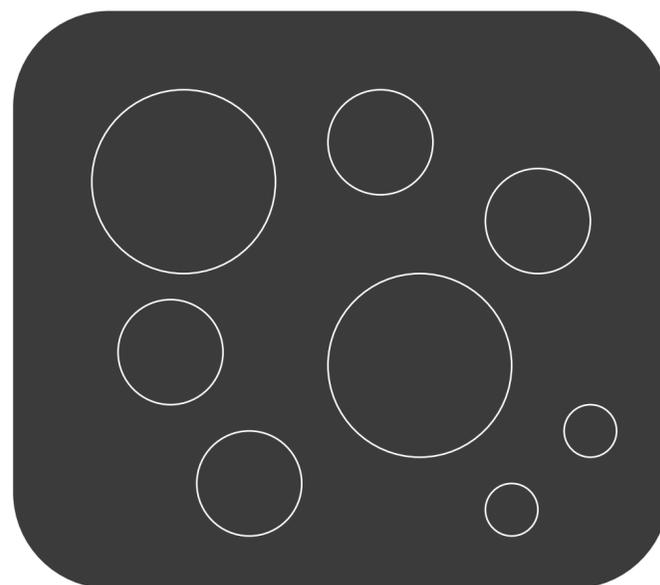
# 总结

服务拆分，需要综合考虑业务功能、质量属性、团队多方面因素，权衡利弊，做出的合理选择。

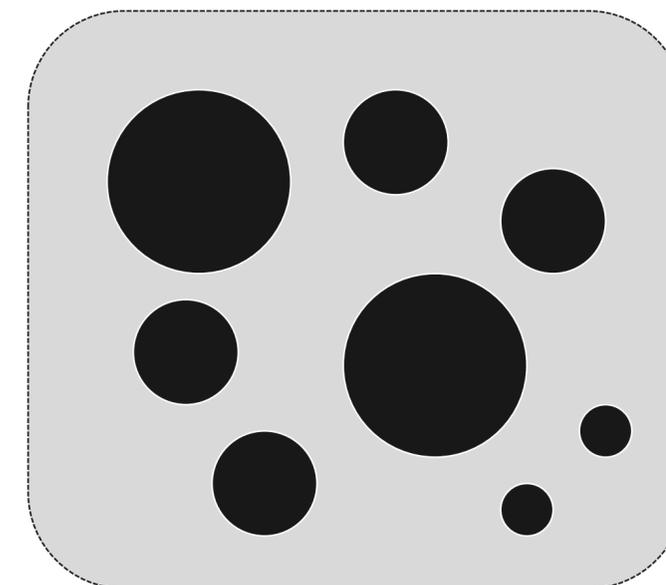
单体



组件化

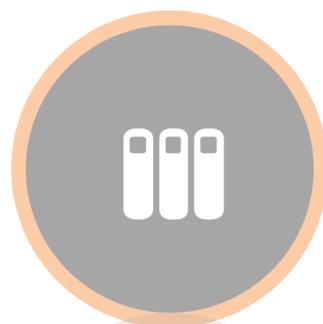


服务化



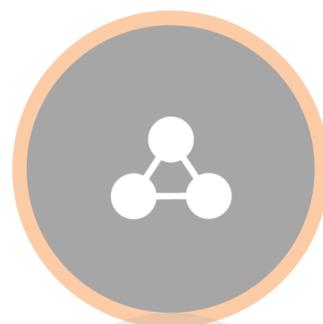
Simon Brown

如果你不能正确地构建大型单体应用，那么微服务也帮不了你。



## 微服务

什么是微服务?



## 服务拆分

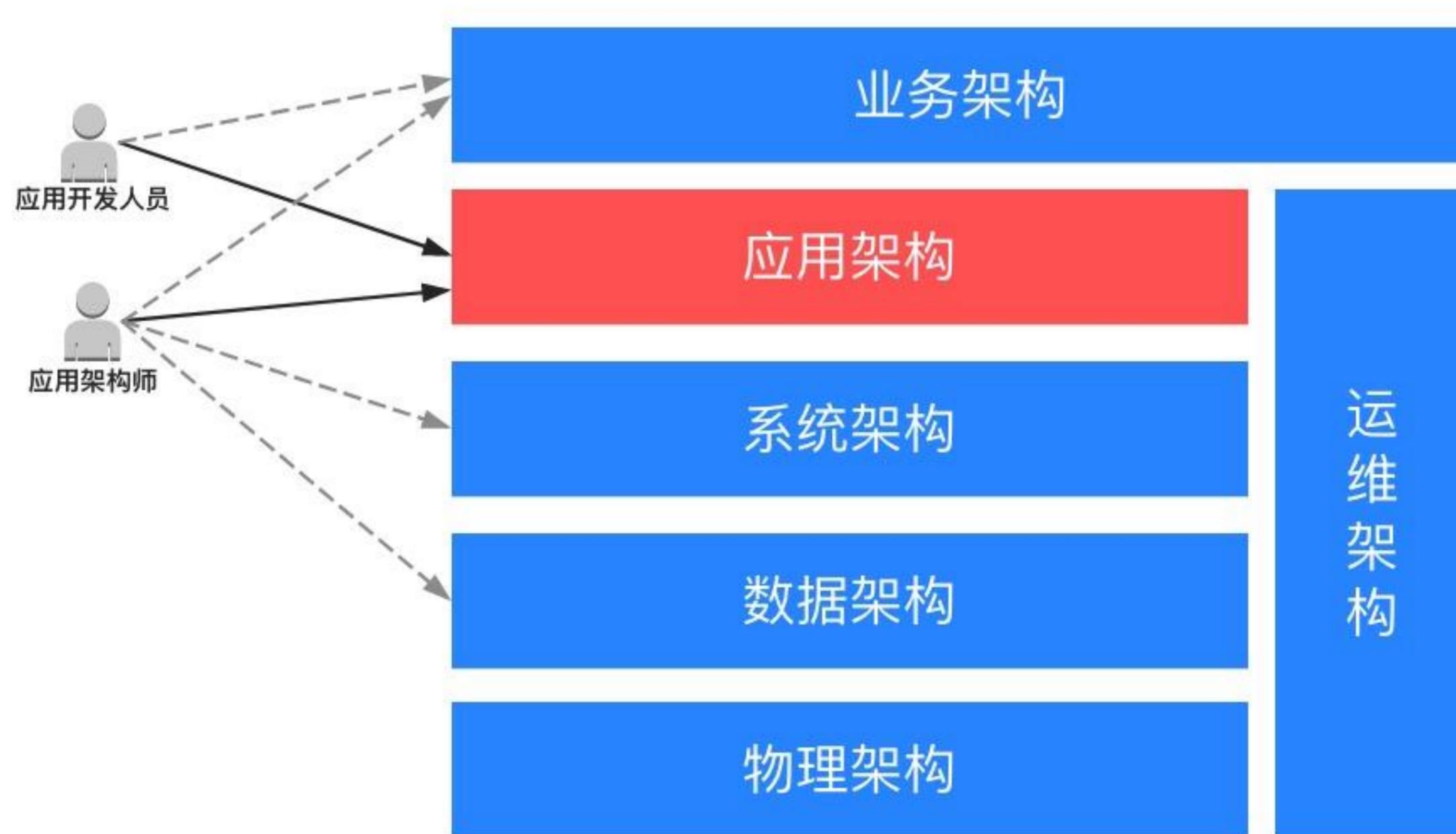
服务如何拆分?



## 应用架构

应用架构如何设计?

# 什么是应用架构



# Clean Object-oriented & Layered Architecture

## 整洁面向对象分层架构

分层设计

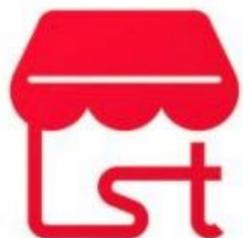
扩展设计

规范设计

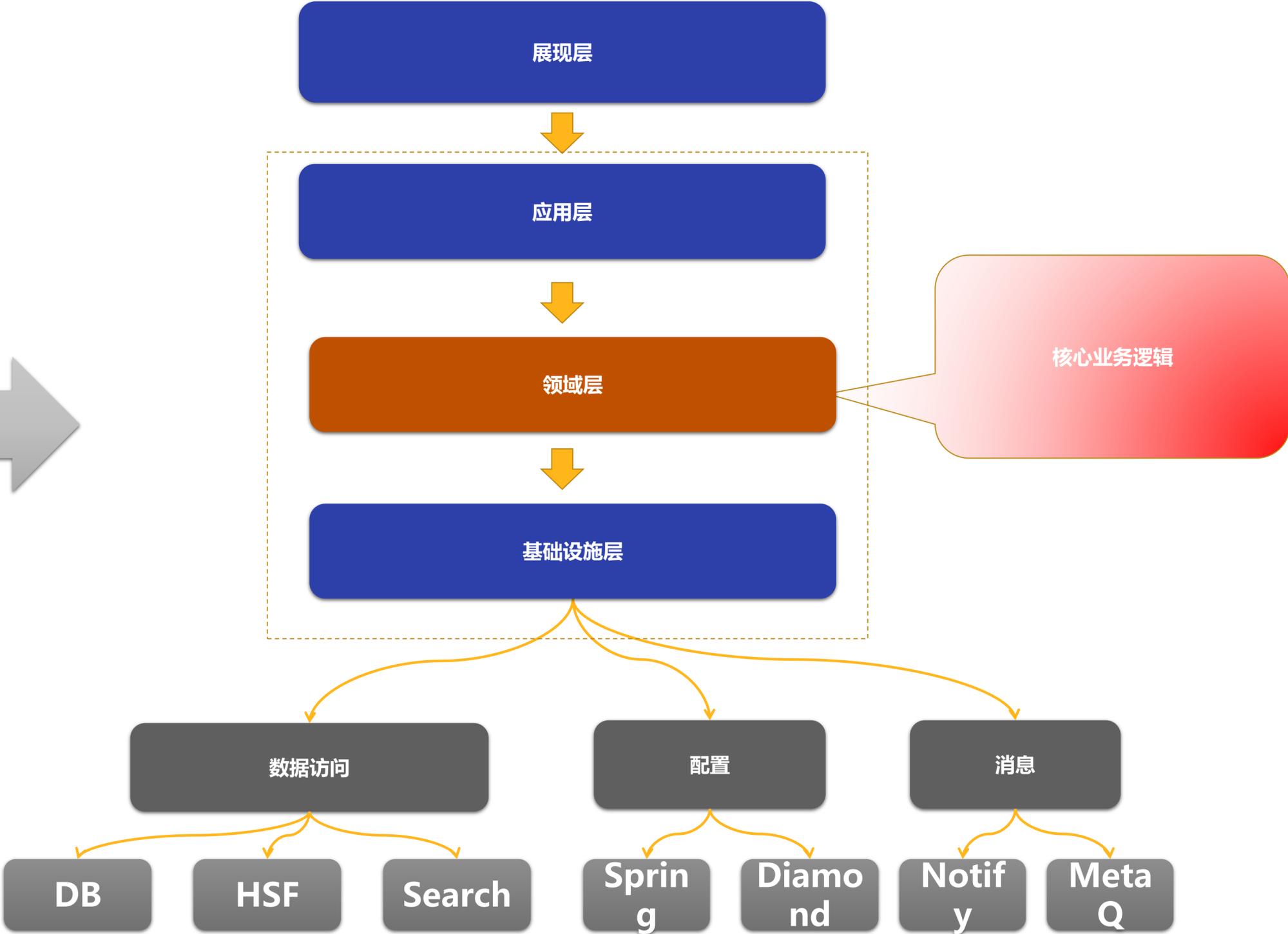
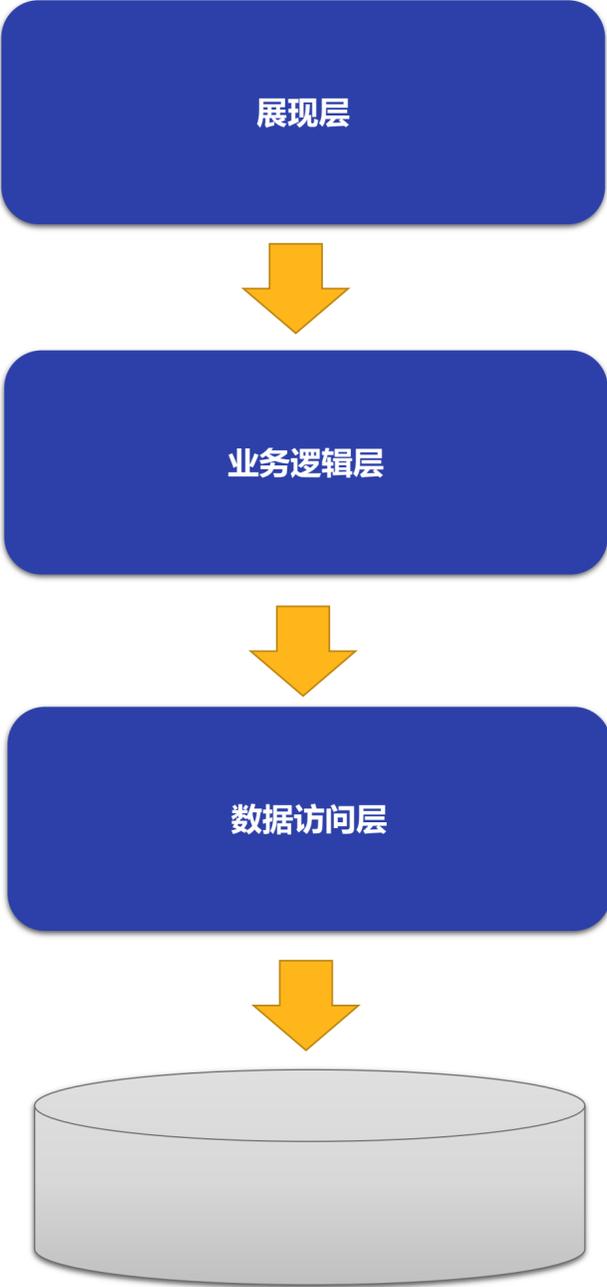
方法论指引

开源地址: <https://github.com/alibaba/COLA>

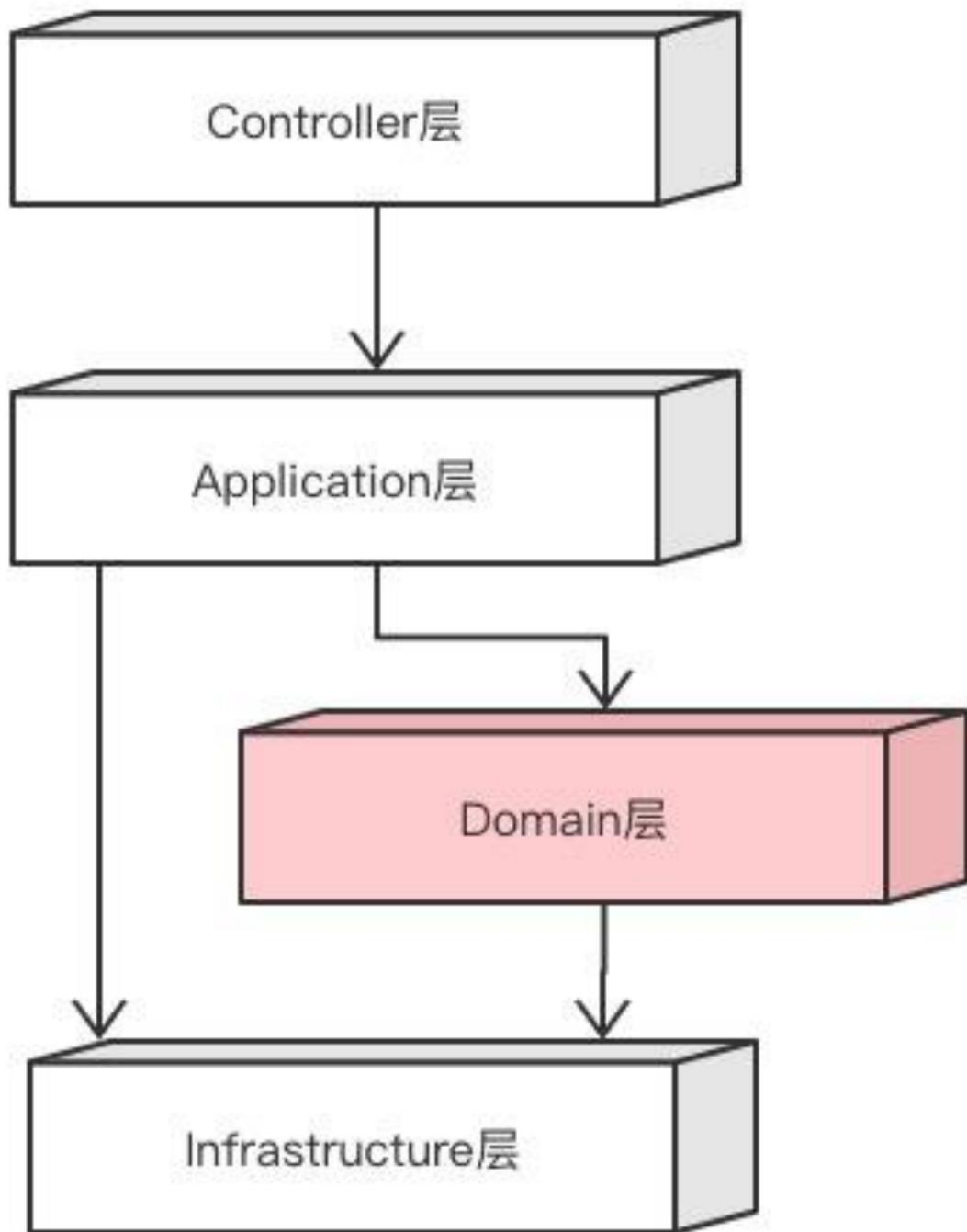
阿里云应用生成: <https://start.aliyun.com/>



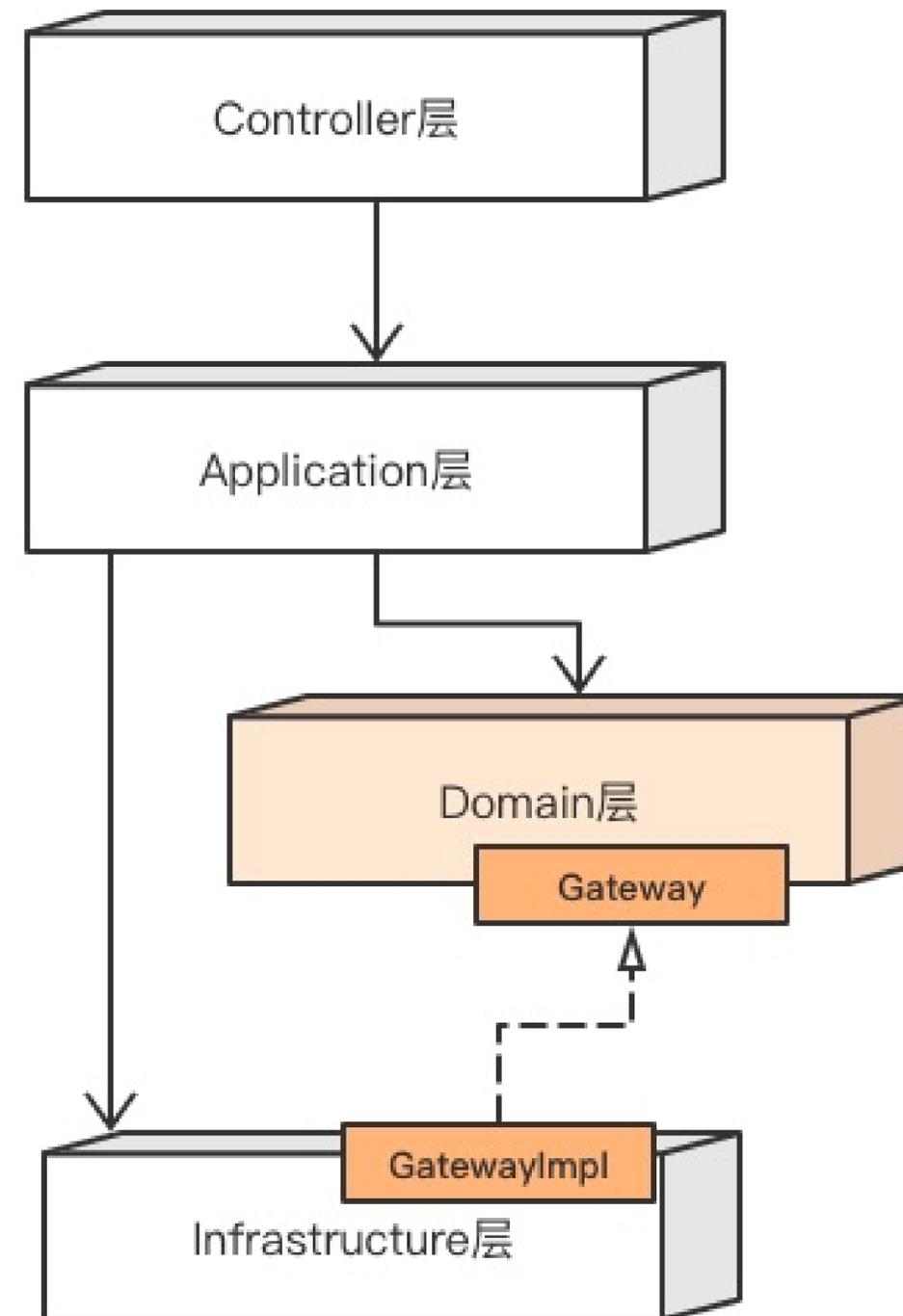
# COLA 分层设计



# COLA 分层设计

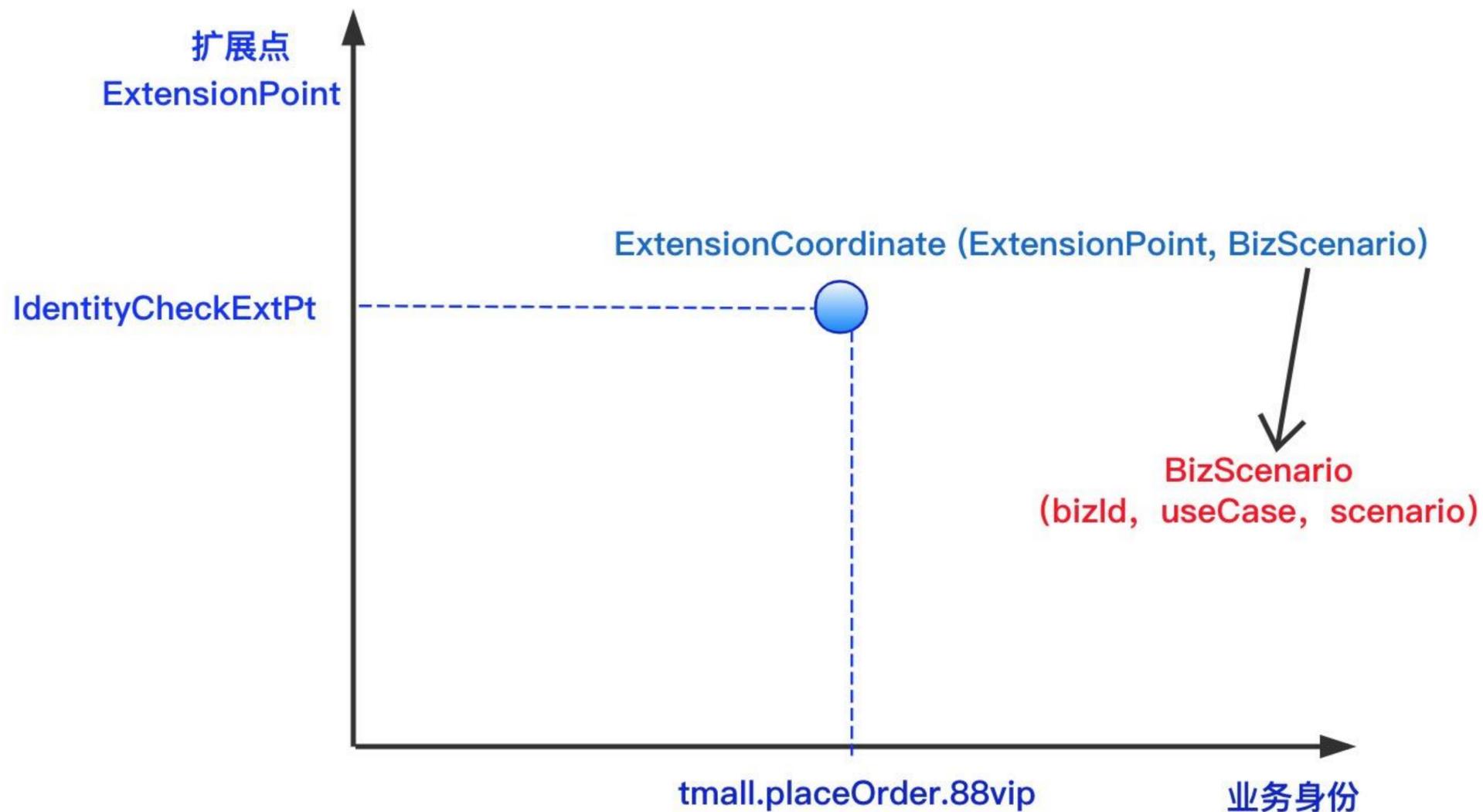


**DIP** →



# COLA 扩展设计

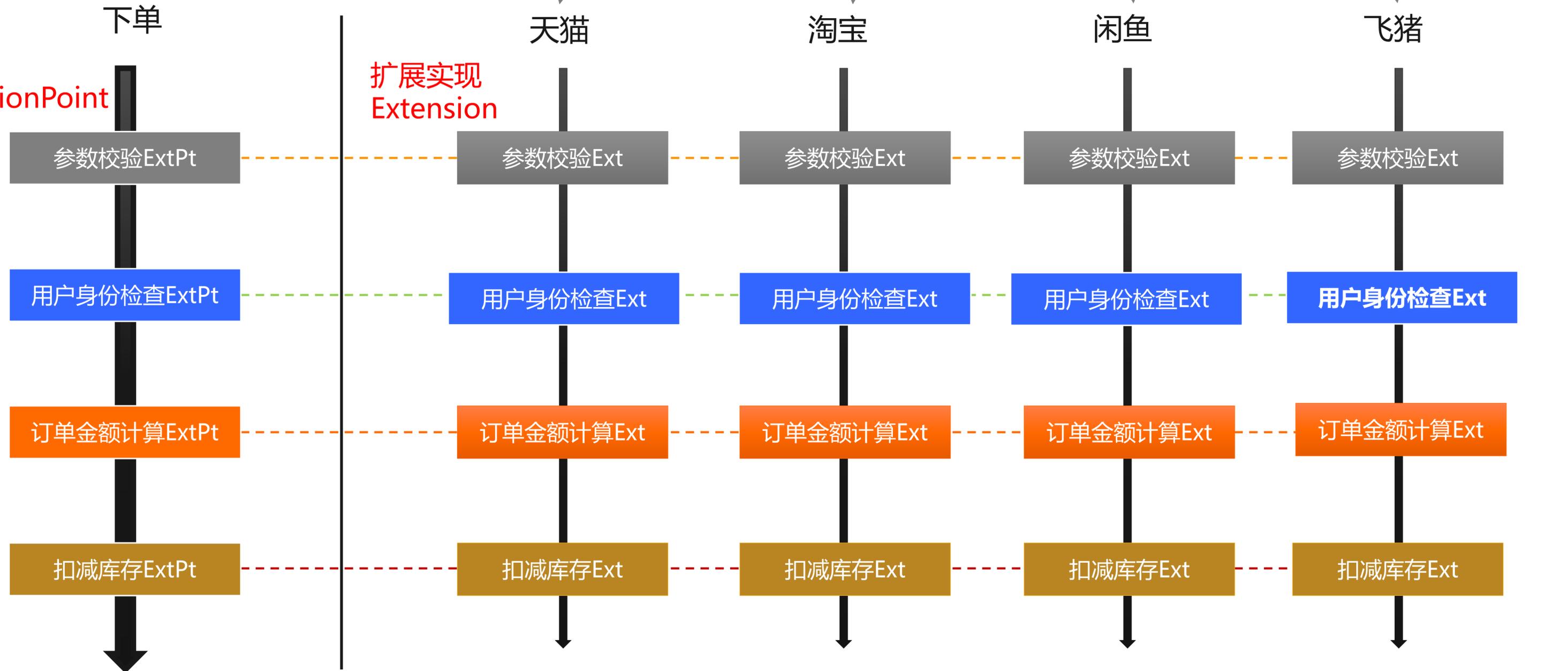
扩展坐标 = 业务身份 + 扩展点



```
@Extension(bizId = "tmall", useCase = "placeOrder", scenario = "88vip")  
public class IdentityCheck88VipExt implements IdentityCheckExtPt{  
  
}
```

# COLA 扩展设计

扩展点  
ExtensionPoint



# COLA 规范设计：认知成本



随心所欲

从混乱到有序

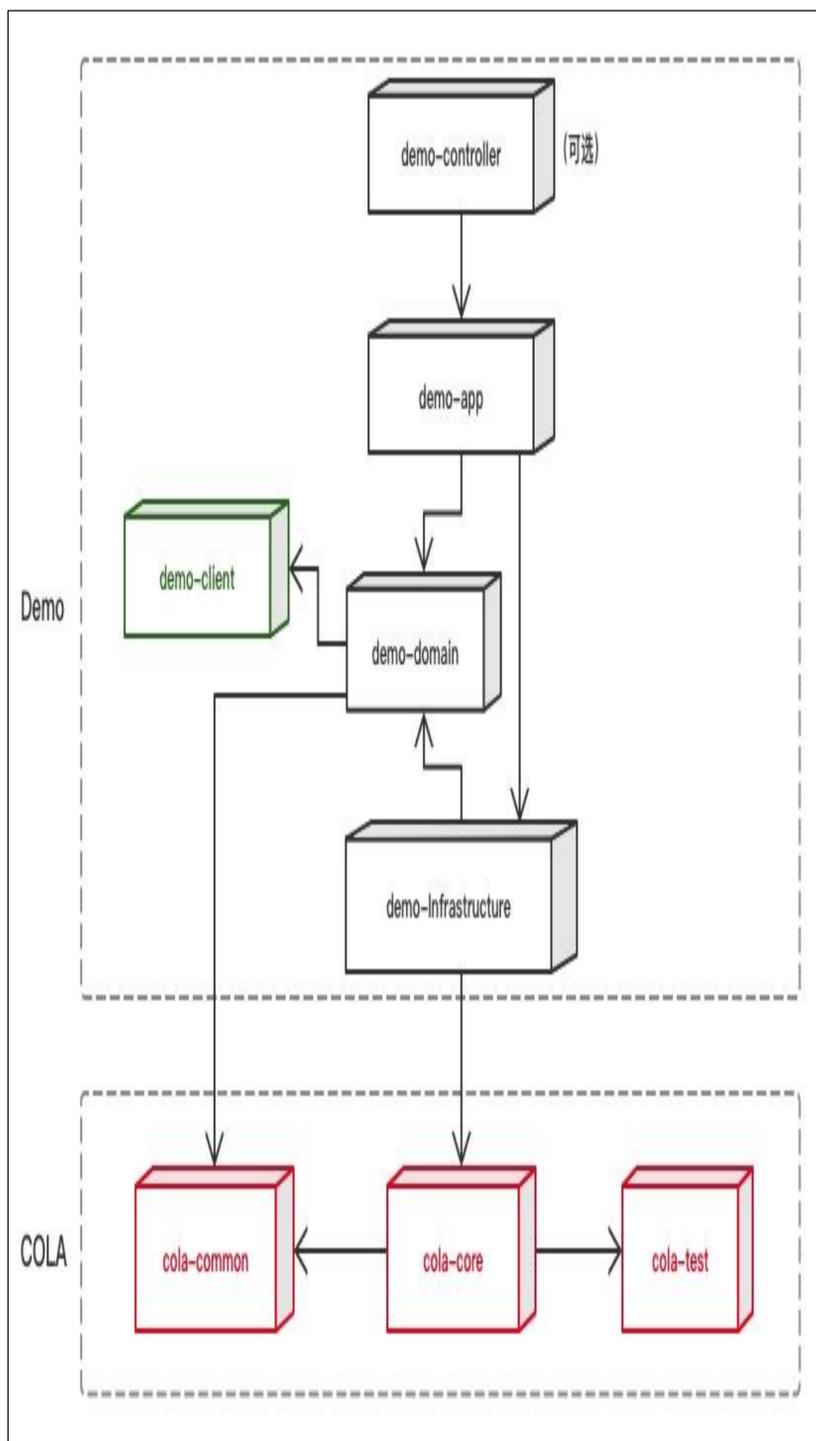


遵守规范

你是喜欢阅读**左边**的代码还是**右边**的代码呢？

# COLA 规范设计：物以类聚

## Module规范



## Package规范



## 命名规范

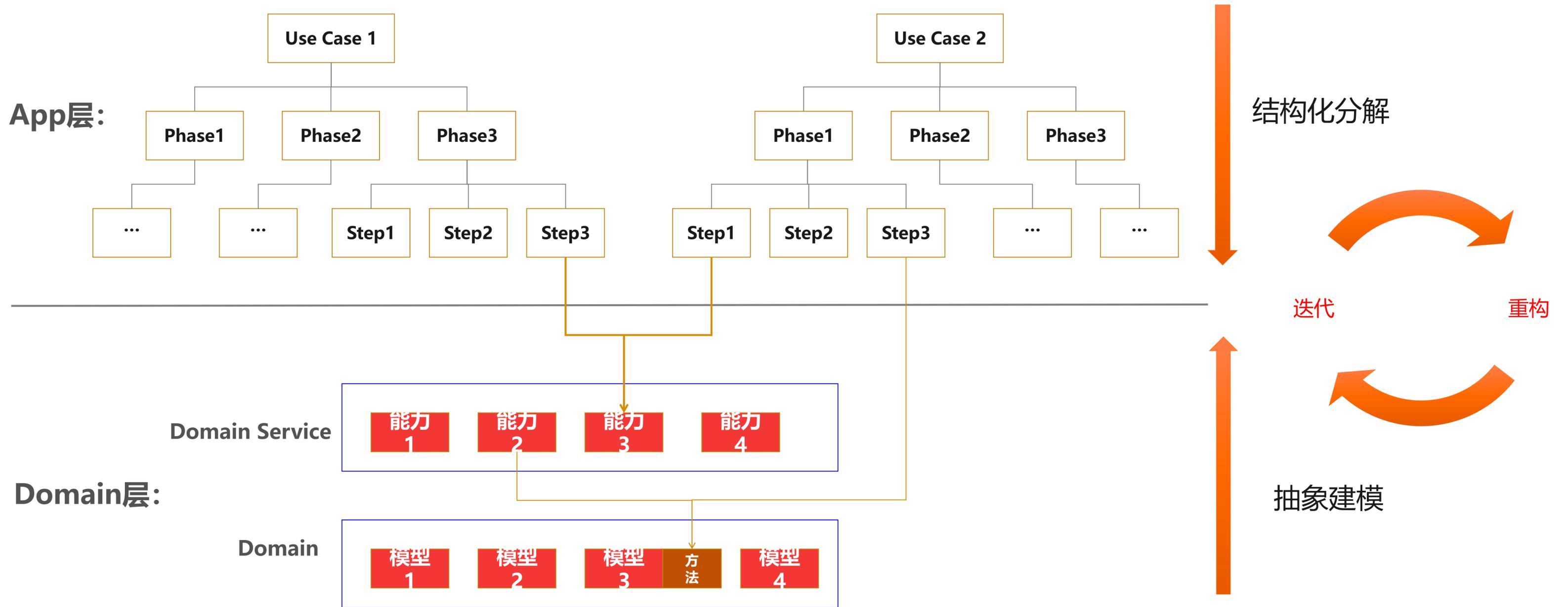
### API Classes

Client Request	↔	XXXCmd.java
Client Object	↔	XXXCO.java
API Service	↔	XXXServiceI.java

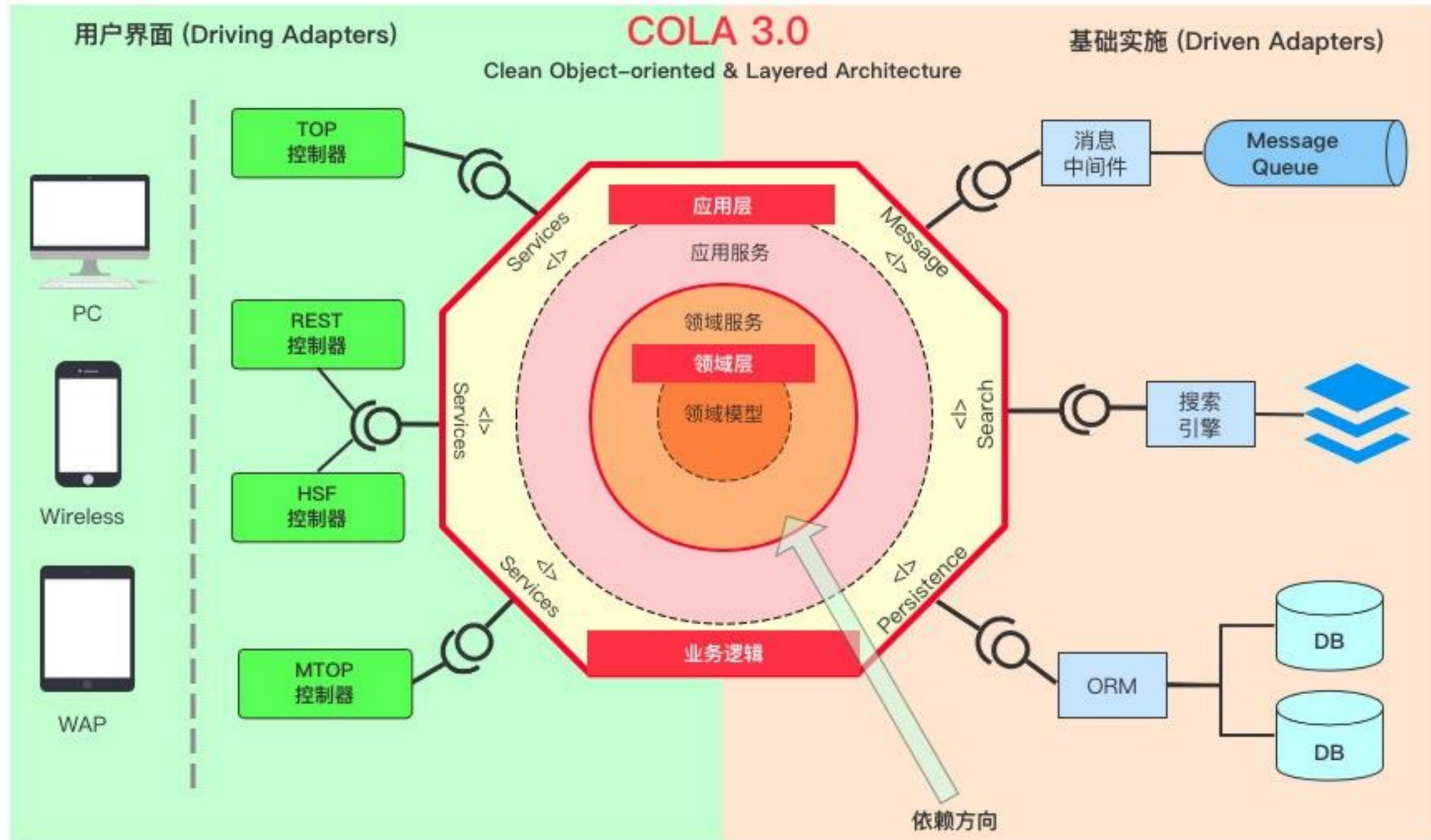
### Domain Classes

Domain Object	↔	XXXE.java
Value Object	↔	XXXV.java
Domain Factory	↔	XXXFactory.java
Repository	↔	XXXRepository.java
Business Rule	↔	XXXRule.java
Domain Service	↔	XXXDomainService

# COLA 应用开发方法论



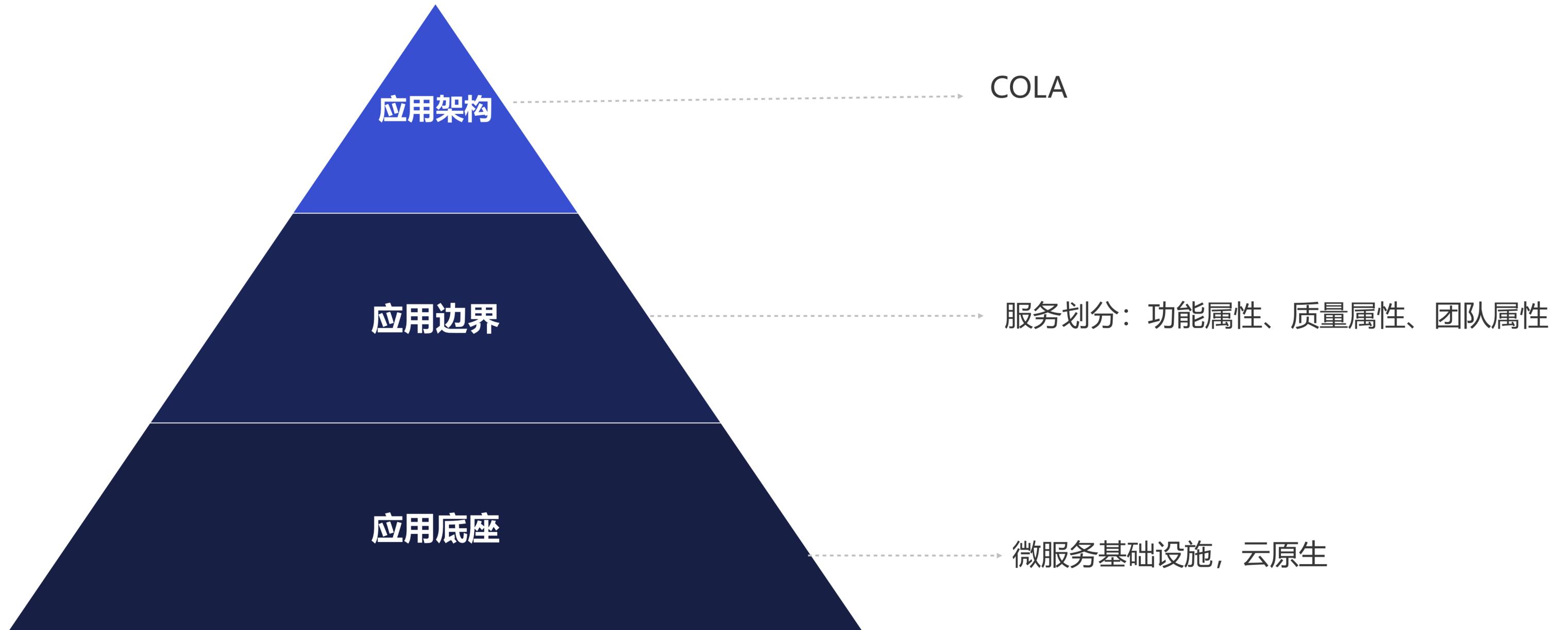
# COLA架构：洋葱圈、六边形、整洁架构、中台架构、DDD



# 分享总结

开源地址: <https://github.com/alibaba/COLA>

阿里云应用生成: <https://start.aliyun.com/>



# THANKS



阿里云 × 2020 云原生微服务大会  
Microservices Summit

关注“阿里巴巴云原生”公众号  
获取第一手技术资料