



# 支付宝的高可用与容灾架构演进

曾欢 @阿里巴巴

shanheng.zh@alibaba-inc.com

# Geekbang>

## 极客邦科技

整合全球最优质学习资源, 帮助技术人 and 企业成长  
Growing Technicians, Growing Companies

**InfoQ**<sup>ueue</sup>

专注中高端技术人员的技术媒体



**EGO** EXTRA GEEKS' ORGANIZATION  
NETWORKS

高端技术人员  
学习型社交网络



**StuQ**<sup>ueue</sup>

实践驱动的  
IT职业学习和服务平台



**GiT** GEEKBANG  
INTERNATIONAL  
TRAINING  
极客邦培训 东京 Tokyo

一线专家驱动的企业培训服务



旧金山 伦敦 北京 圣保罗 东京 纽约 上海  
San Francisco London Beijing Sao Paulo Tokyo New York Shanghai

# QCon

## 全球软件开发大会

2016年4月21-23日 | 北京·国际会议中心

主办方 **Geekbang** **InfoQ**  
极客邦科技

**7折** 优惠 (截至12月27日)  
现在报名, 节省2040元/张, 团购享受更多优惠

[www.qconbeijing.com](http://www.qconbeijing.com)



扫描获取更多大会信息



# 个人经历

2012年

加入技术保障部应用运维团队



2013-2014年

负责天猫、淘宝等相关核心系统的运维工作



2015年

负责蚂蚁金服运维，稳定性相关



连续两年『天猫双十一卫士』  
业务稳定性平台



蚂蚁异地多活项目





# 目录

01

OPTIONS

纯真-童年时期

02

OPTIONS

懵懂-少年时期

03

OPTIONS

成熟-青年时期

04

OPTIONS

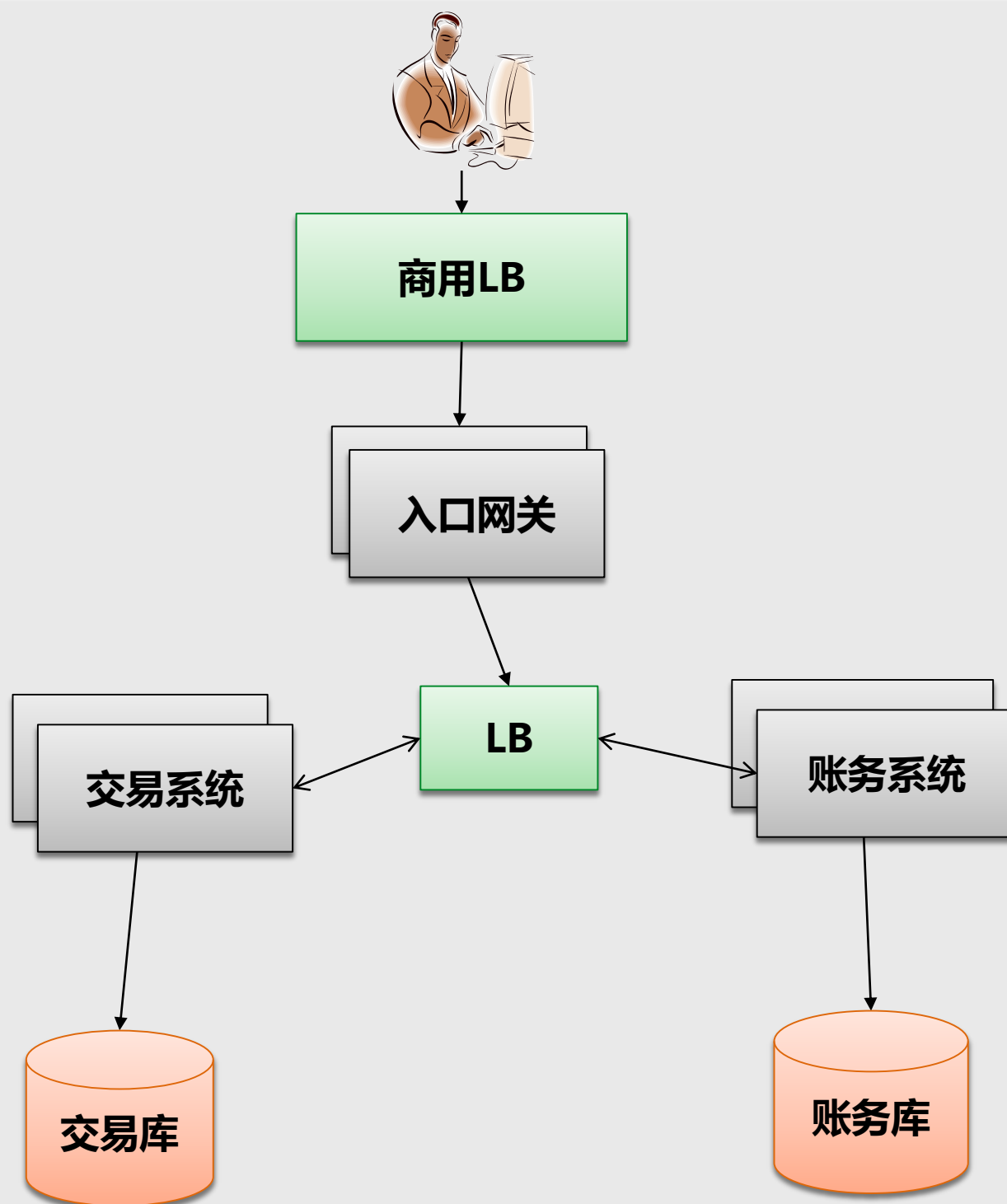
Q&A

# 纯真-童年时期





# 初期系统架构







# 『单机房』架构的背后

- 业务量

数十万级/天

- 应用系统

数十个

- 容灾能力低

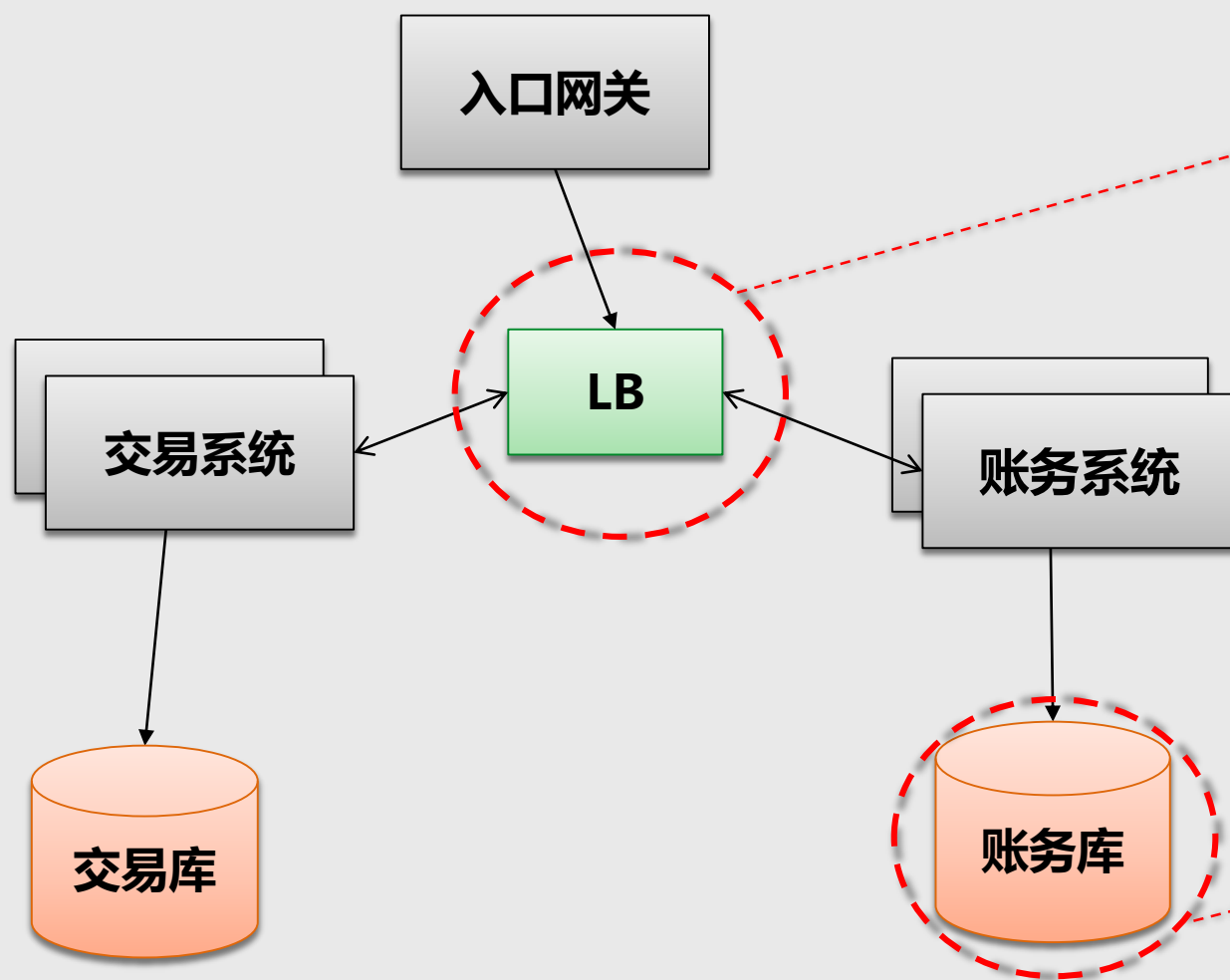
单应用出现问题无法切换  
主备切换业务中断  
停机维护







# 业务量增长带来的问题



- 商用LB瓶颈：VIP数量只增不减，LB宕机影响巨大

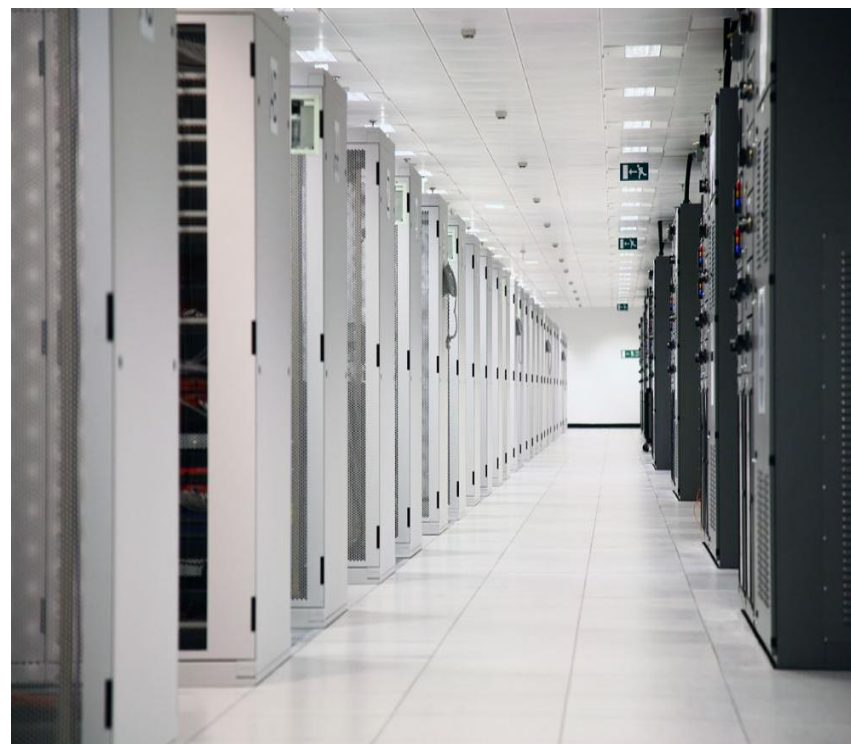
- 核心DB单库瓶颈



## 新架构要解决的问题：

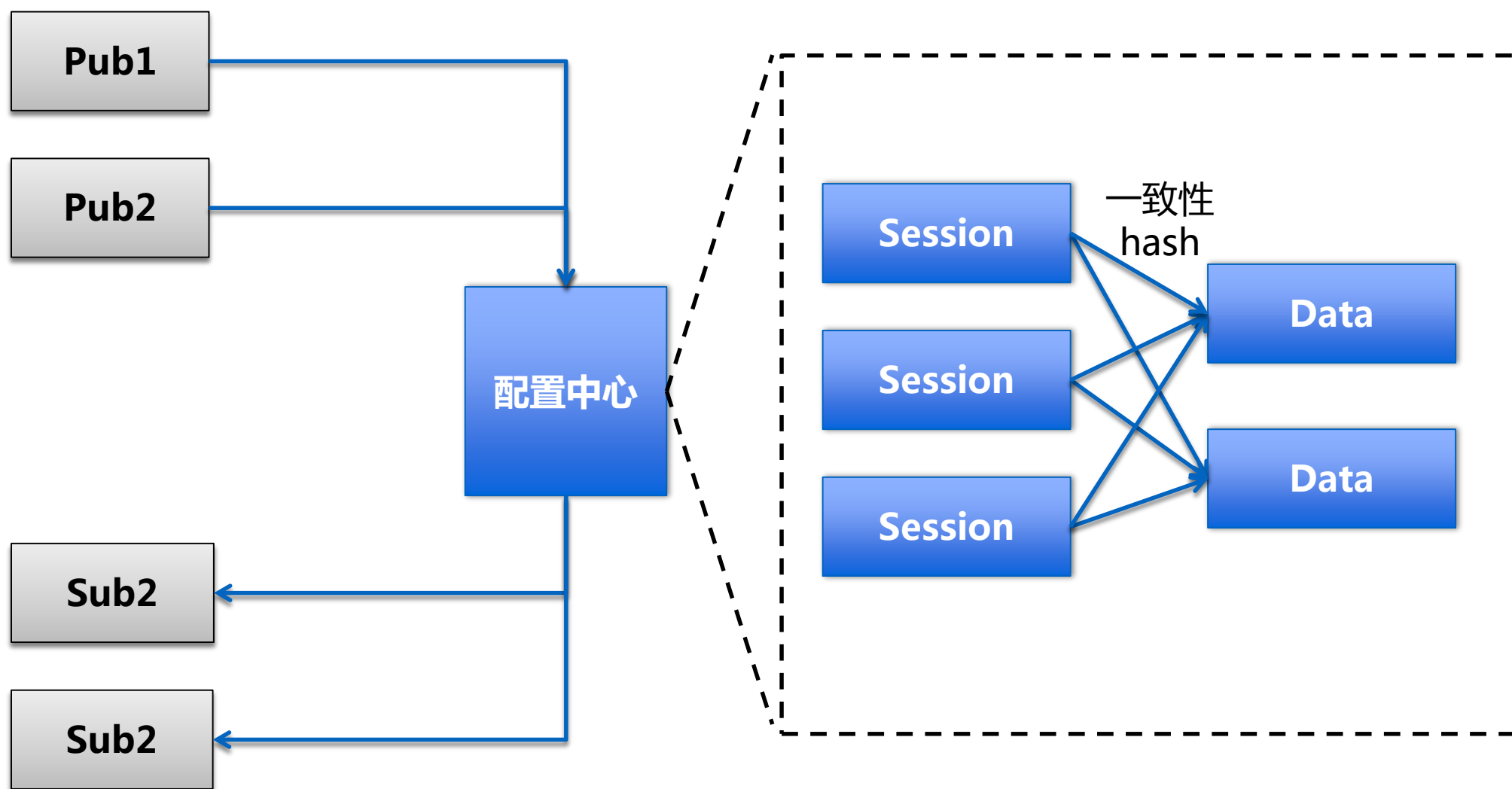
- 支撑日益增长的业务量：数百万级/天
- 解决容量问题：数百个系统
- 消除架构单点

# 懵懂-少年时期



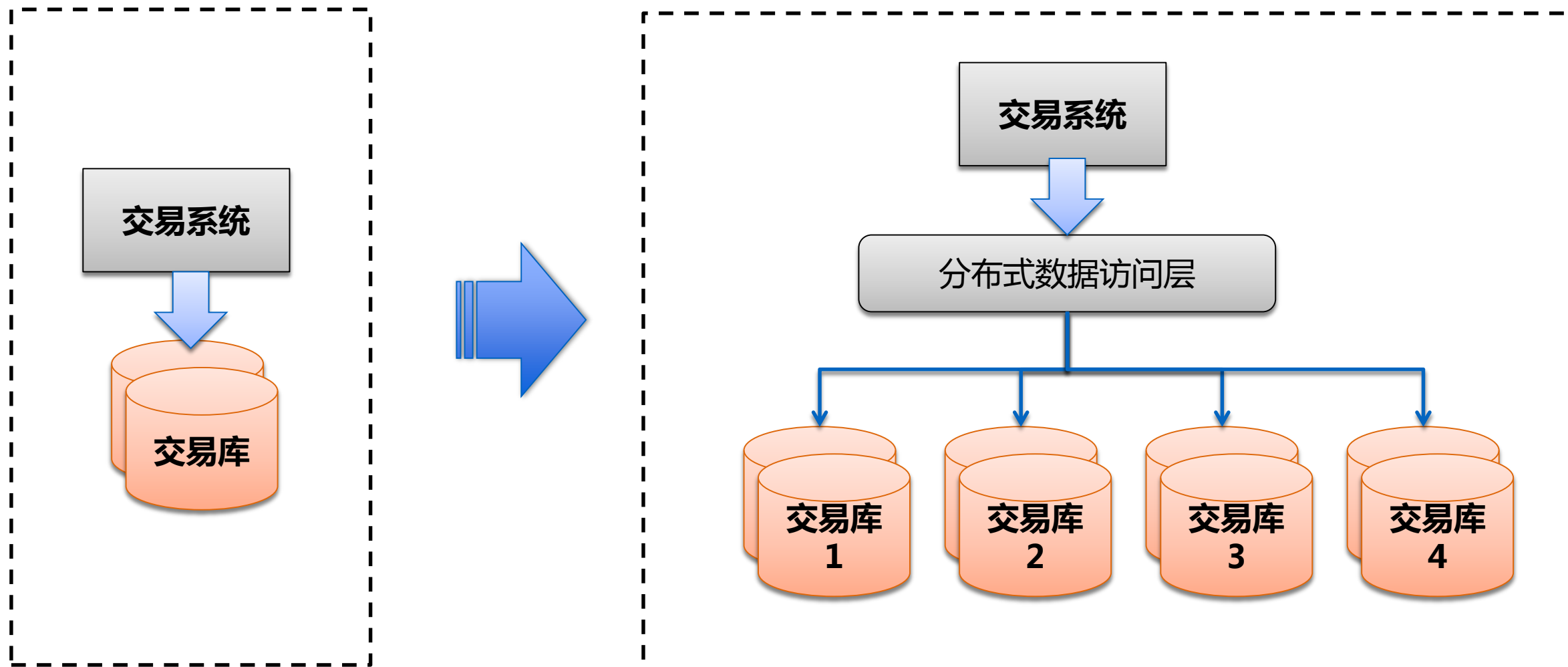


# 硬负载=>软负载：分布式服务调用





# 数据可扩展：UID水平拆分





GSLB

50%

50%

机房A

配置中心

交易系统

账务系统

交易库  
01

账务库  
01

交易  
备库2

账务  
备库2

机房B

配置中心

交易系统

账务系统

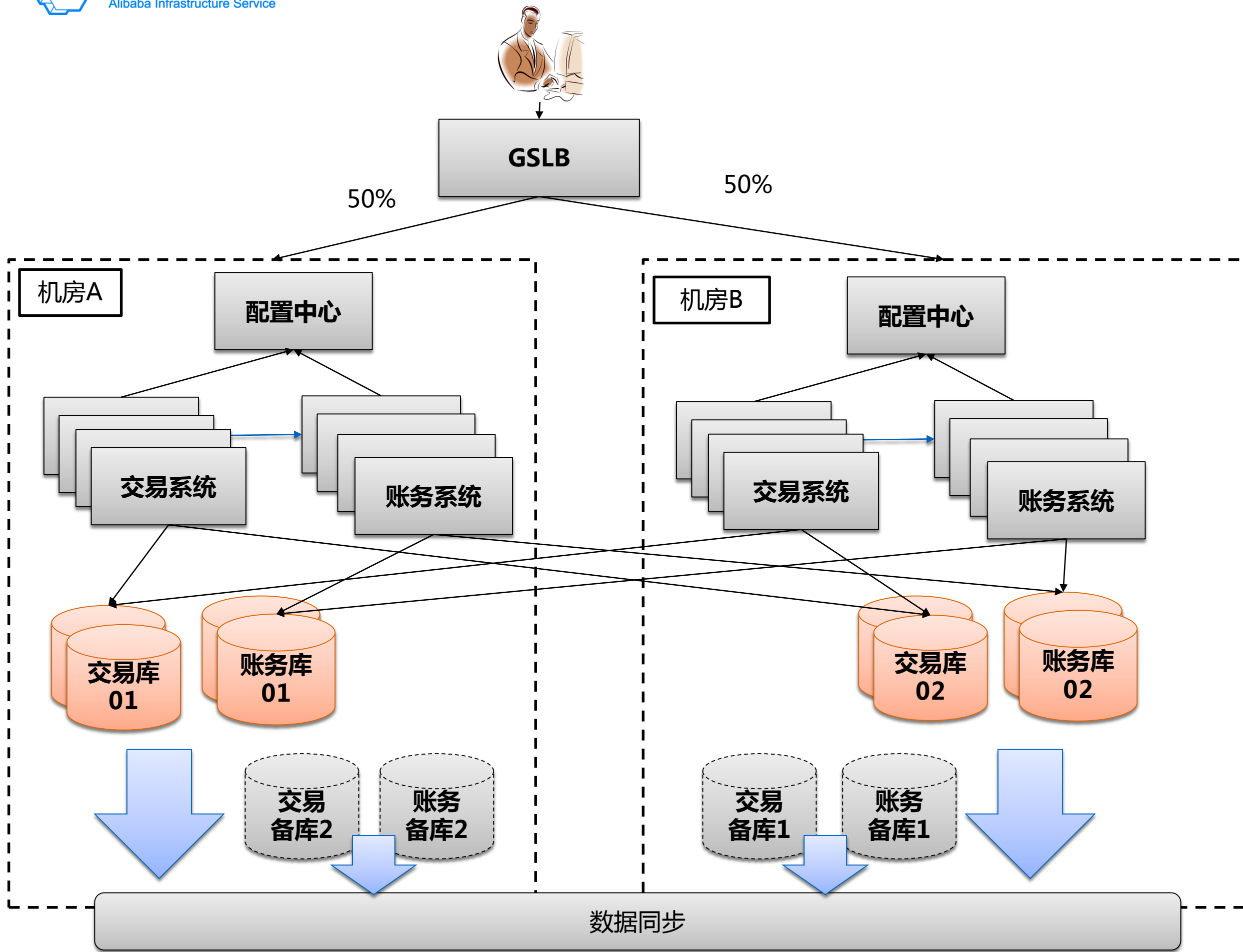
交易库  
02

账务库  
02

交易  
备库1

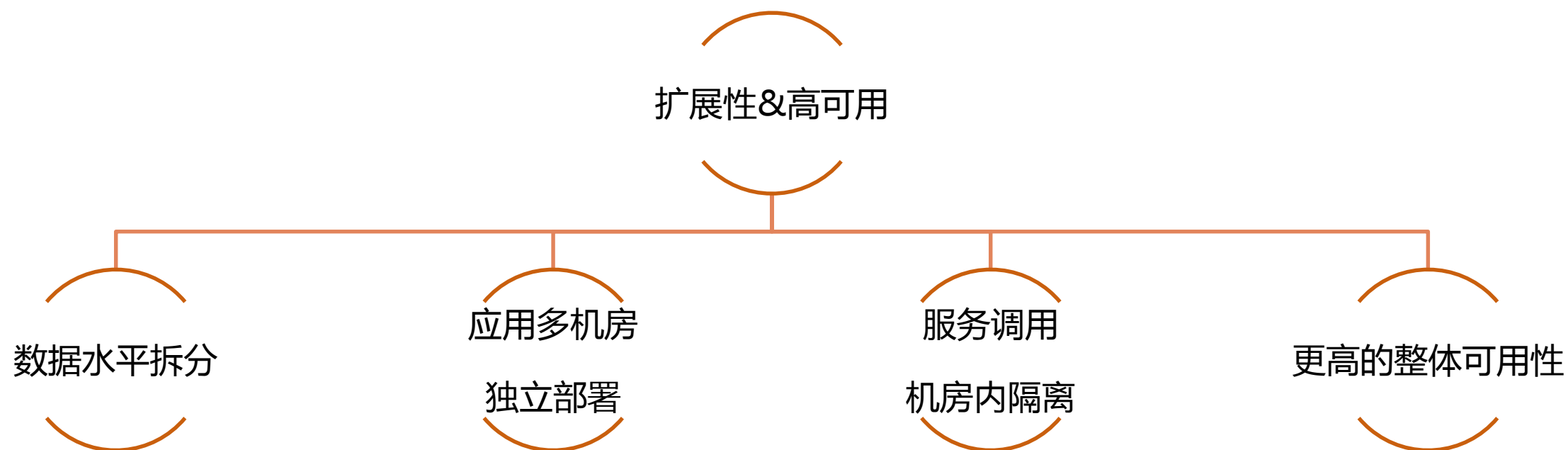
账务  
备库1

数据同步

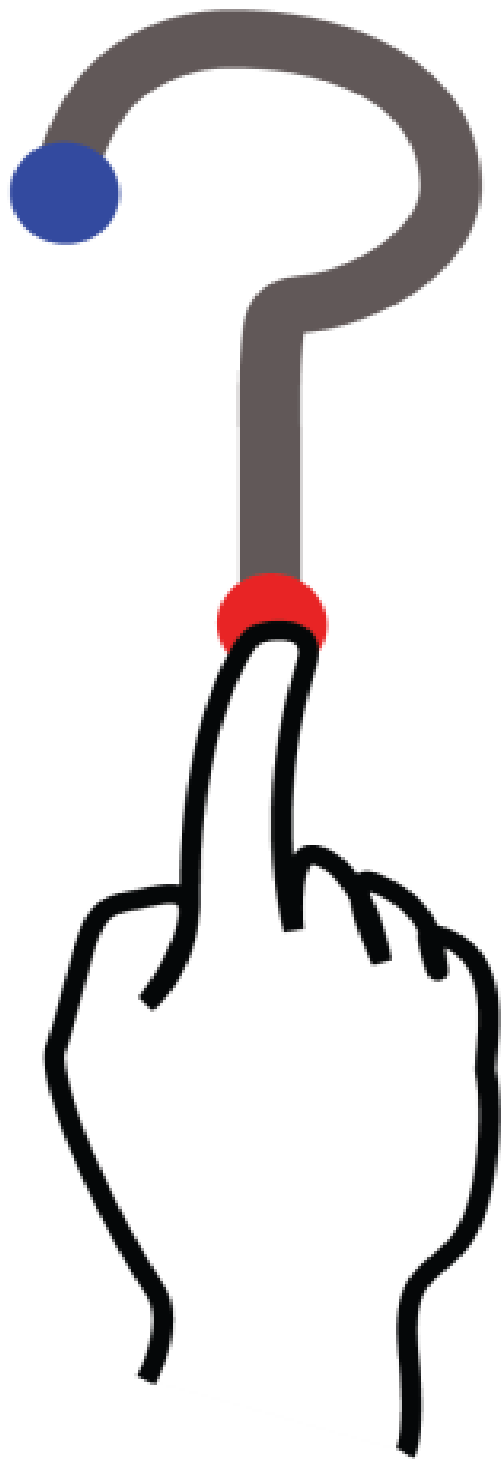




# 『多机房』架构优势







够用了吗？



# 容灾切换过程



Oracle异常情况下的主备切换  
10min



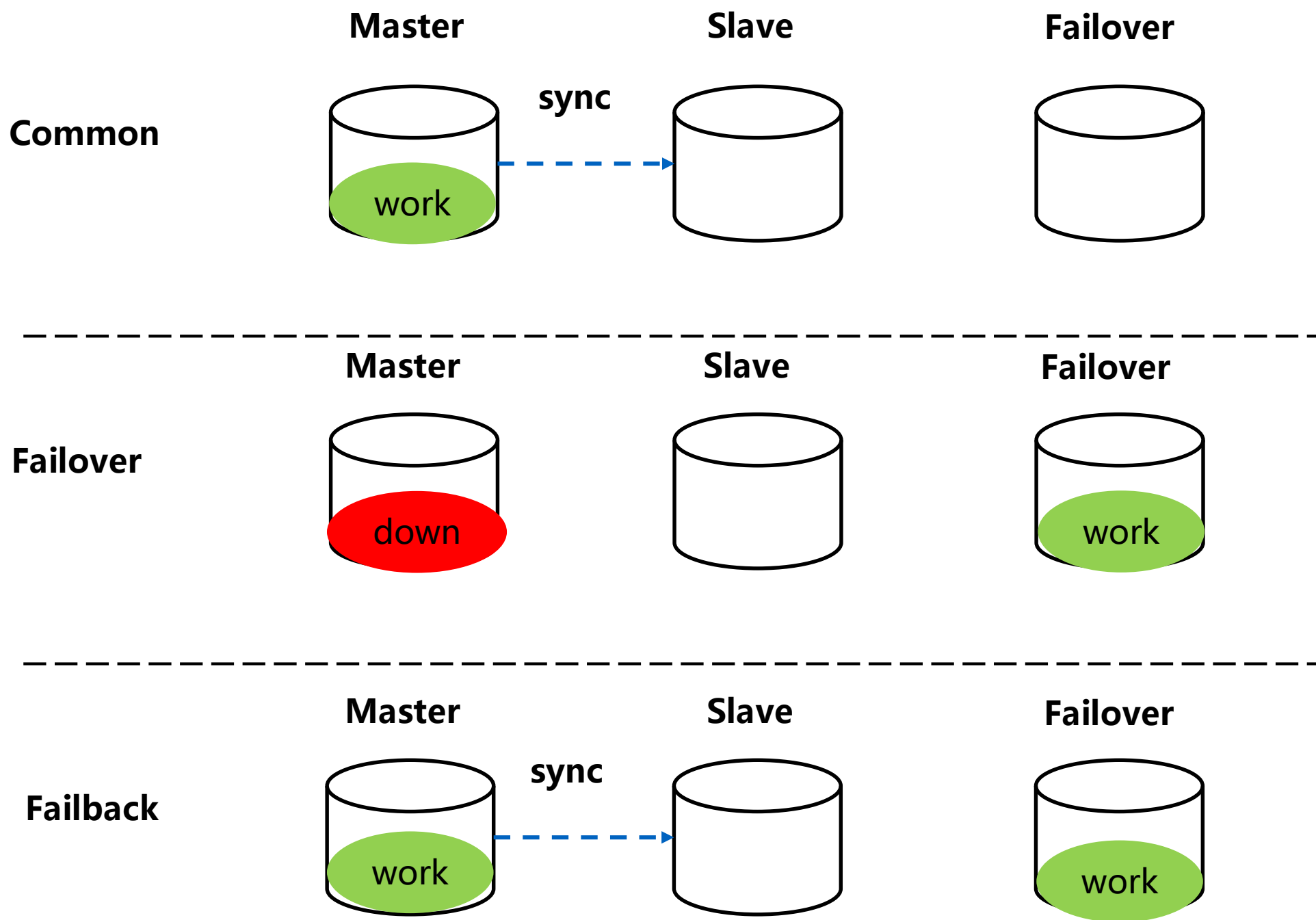
# 数据高可用



**FAILOVER方案**



# Failover方案





GSLB

50%

50%

机房A

配置中心

交易系统

账务系统

交易库  
01

账务库  
01

交易FO  
库2

交易  
备库2

账务  
备库2

机房B

配置中心

交易系统

账务系统

交易FO  
库1

交易库  
02

账务库  
02

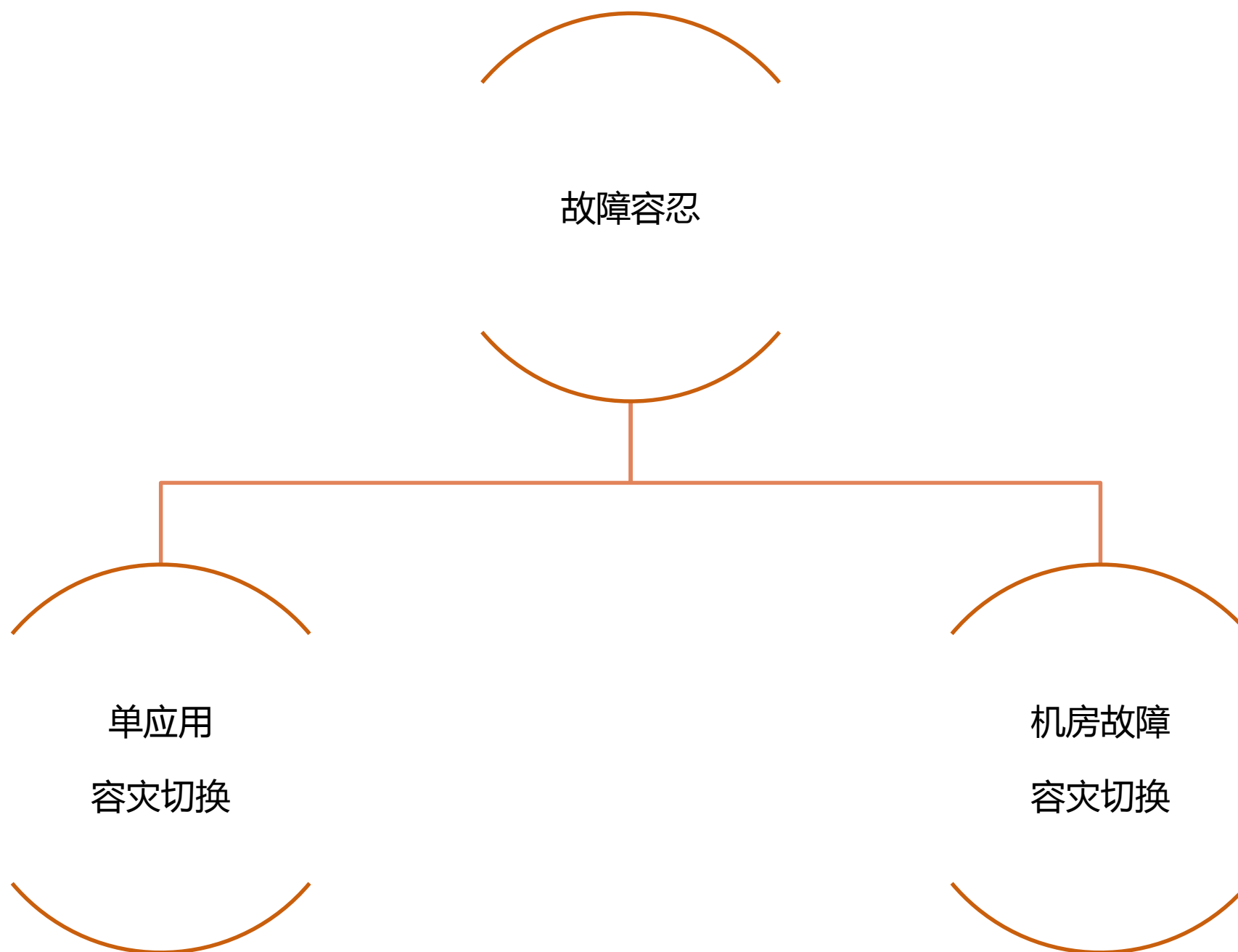
交易  
备库1

账务  
备库1

数据同步



# 『多机房多活』架构的容灾能力



# 成熟-青年时期







# 严峻的新问题

DB连接数



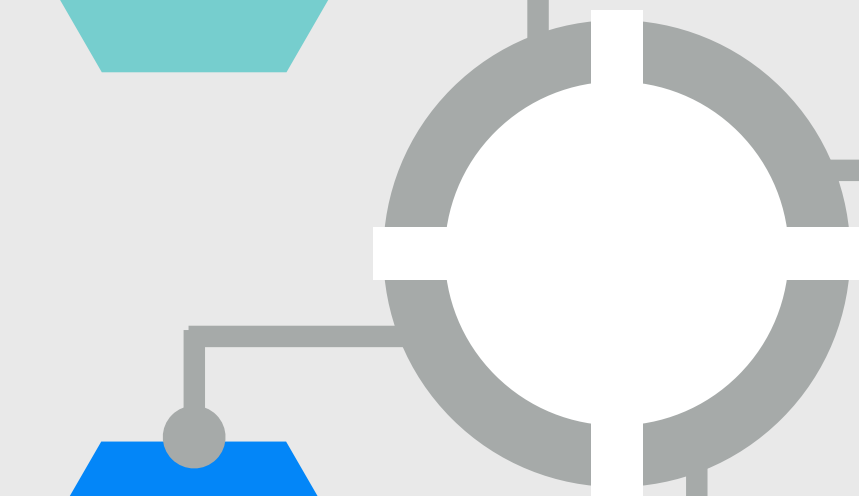
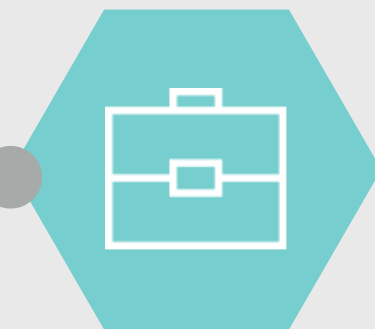
城市IDC资源限制



跨IDC网络延时



机房间流量





# 对新一代架构的需求

01

彻底解决DB连接数问题

02

彻底解决IDC资源限制

03

保证业务连续性

06

**单元化**

05

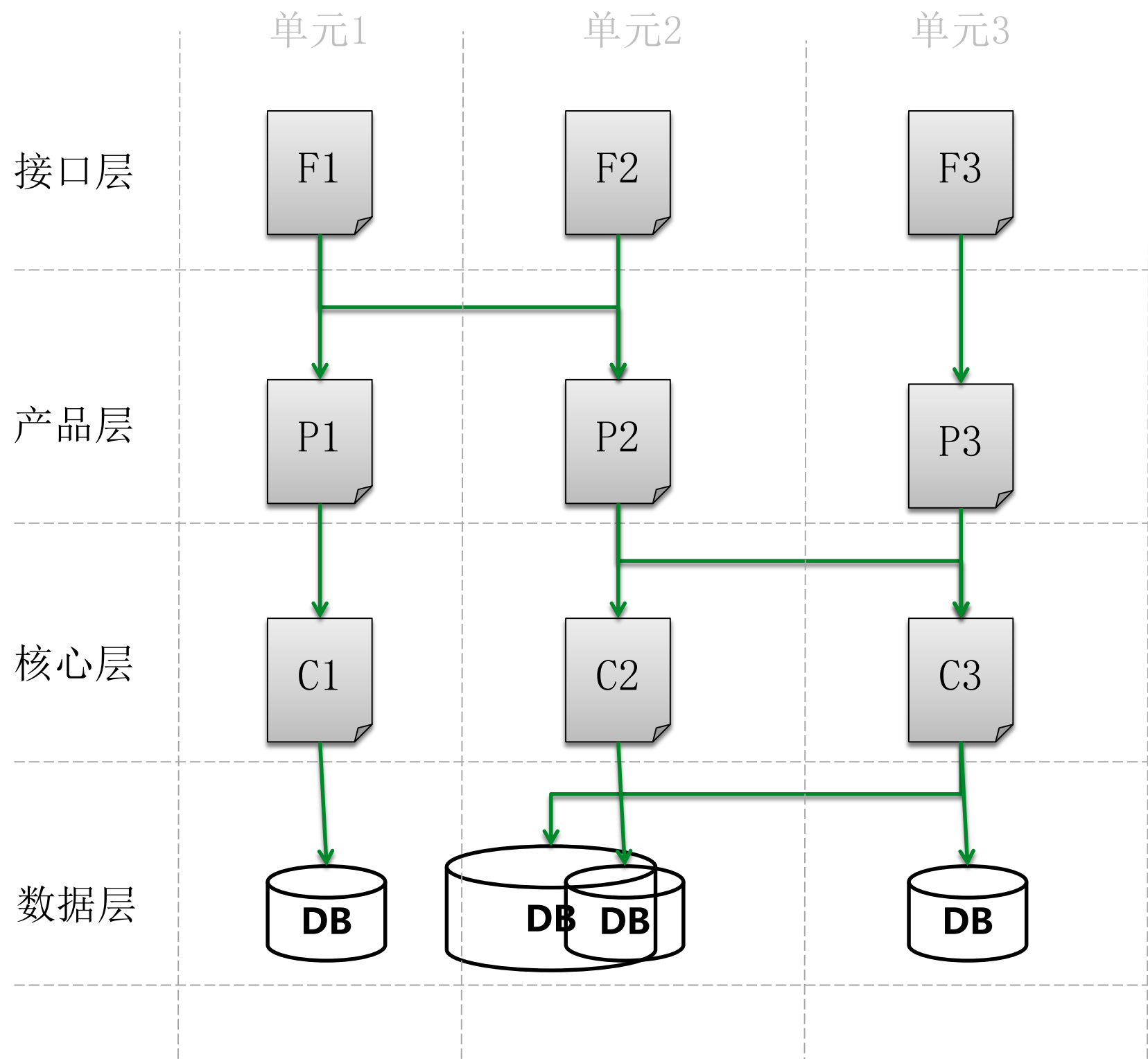
高可用-异地多活

04

蓝绿发布



# 单元化的实质





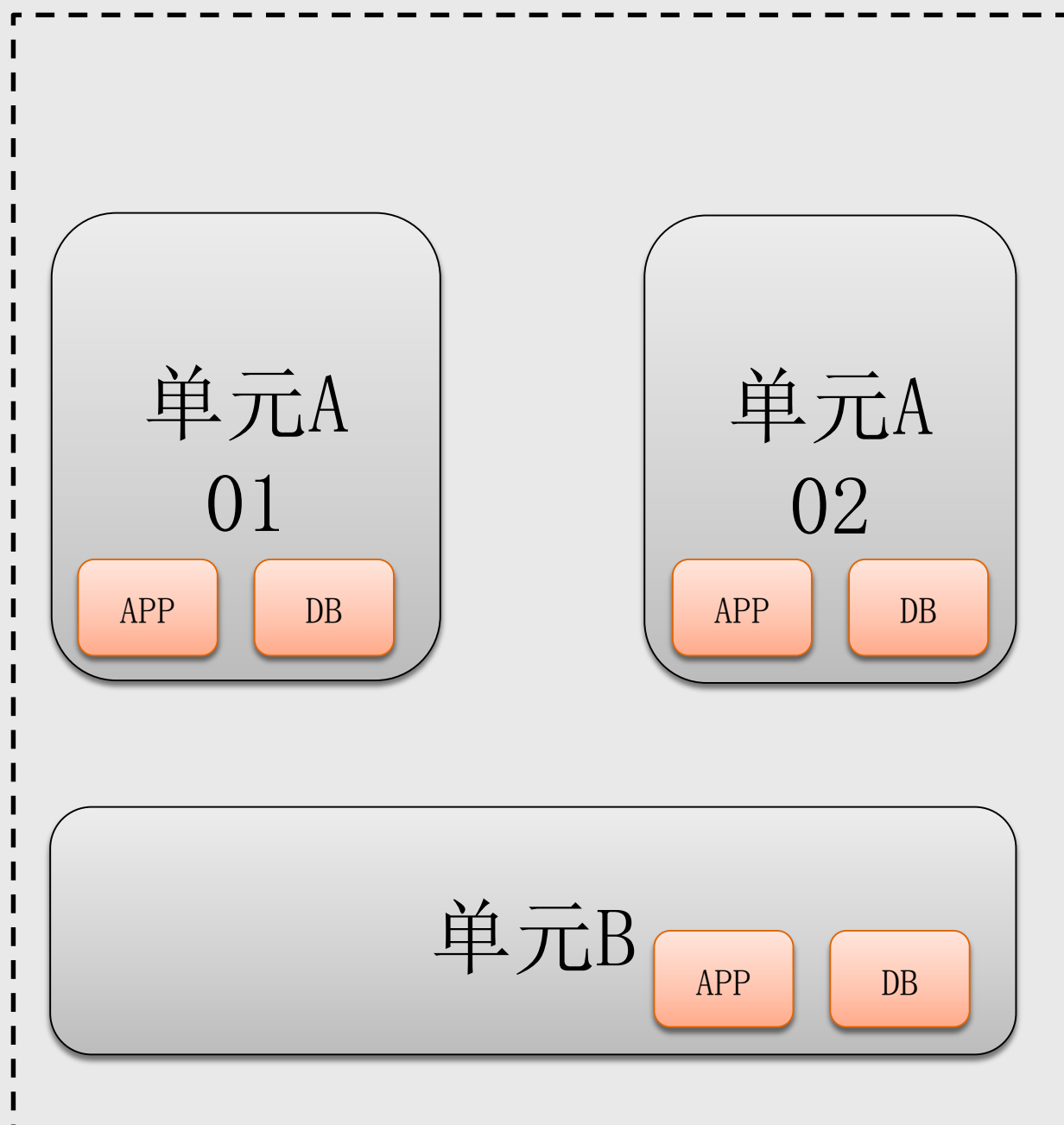
# 单元化核心思想



- 核心业务
- 数据按照UserID拆分，多机房部署
- 调用封闭
- 部分数据，不共享

- 非核心业务
- 不能按照UID拆分
- 核心不依赖长尾

# 支付宝单元化架构实现

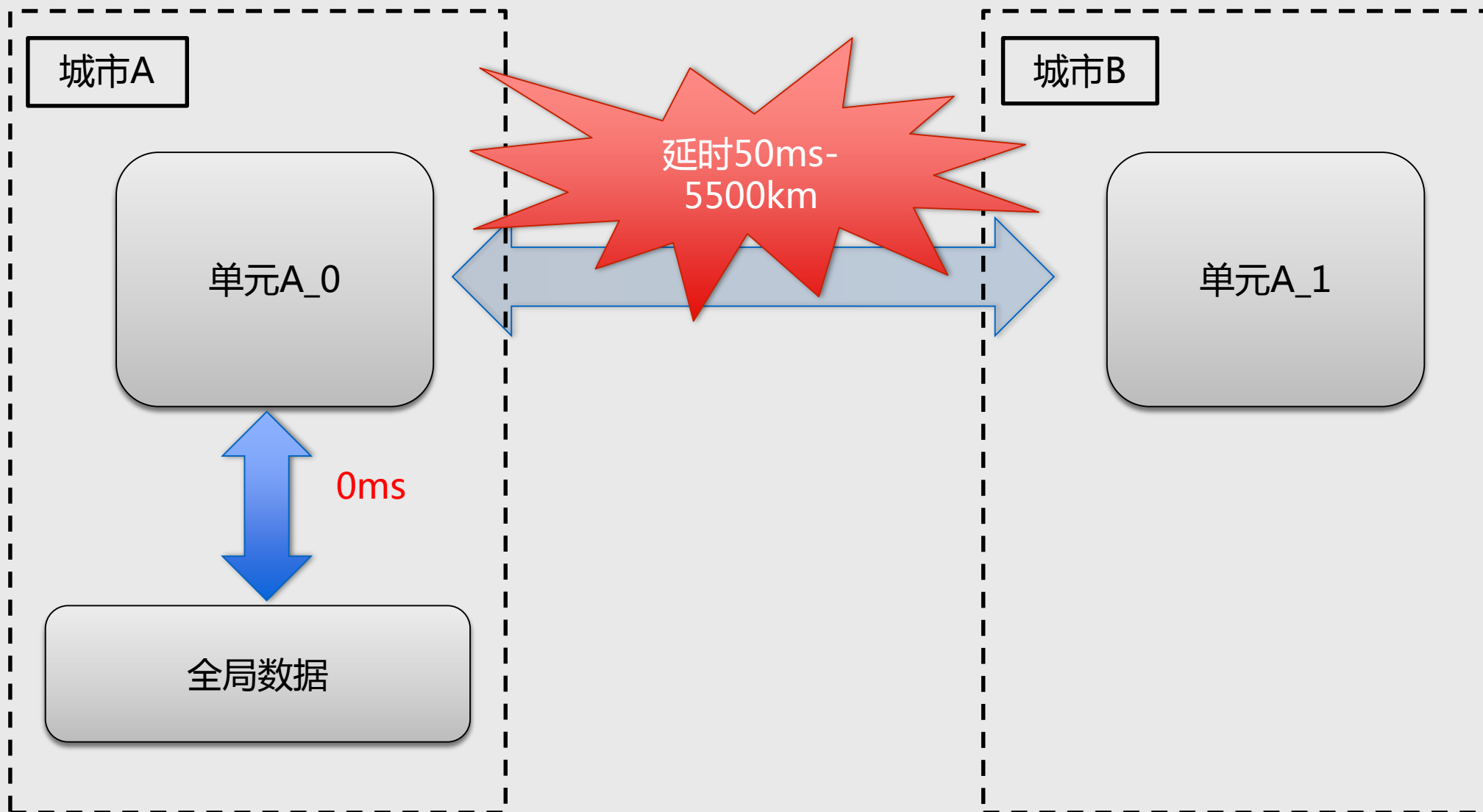


实现思路：

- 水平拆  
交易、支付、账务等，每个单元只有部分数据。
- 上层单元化改造  
从DB层往上延伸水平拆分概念，包括应用层到入口层。



# 走向异地的挑战

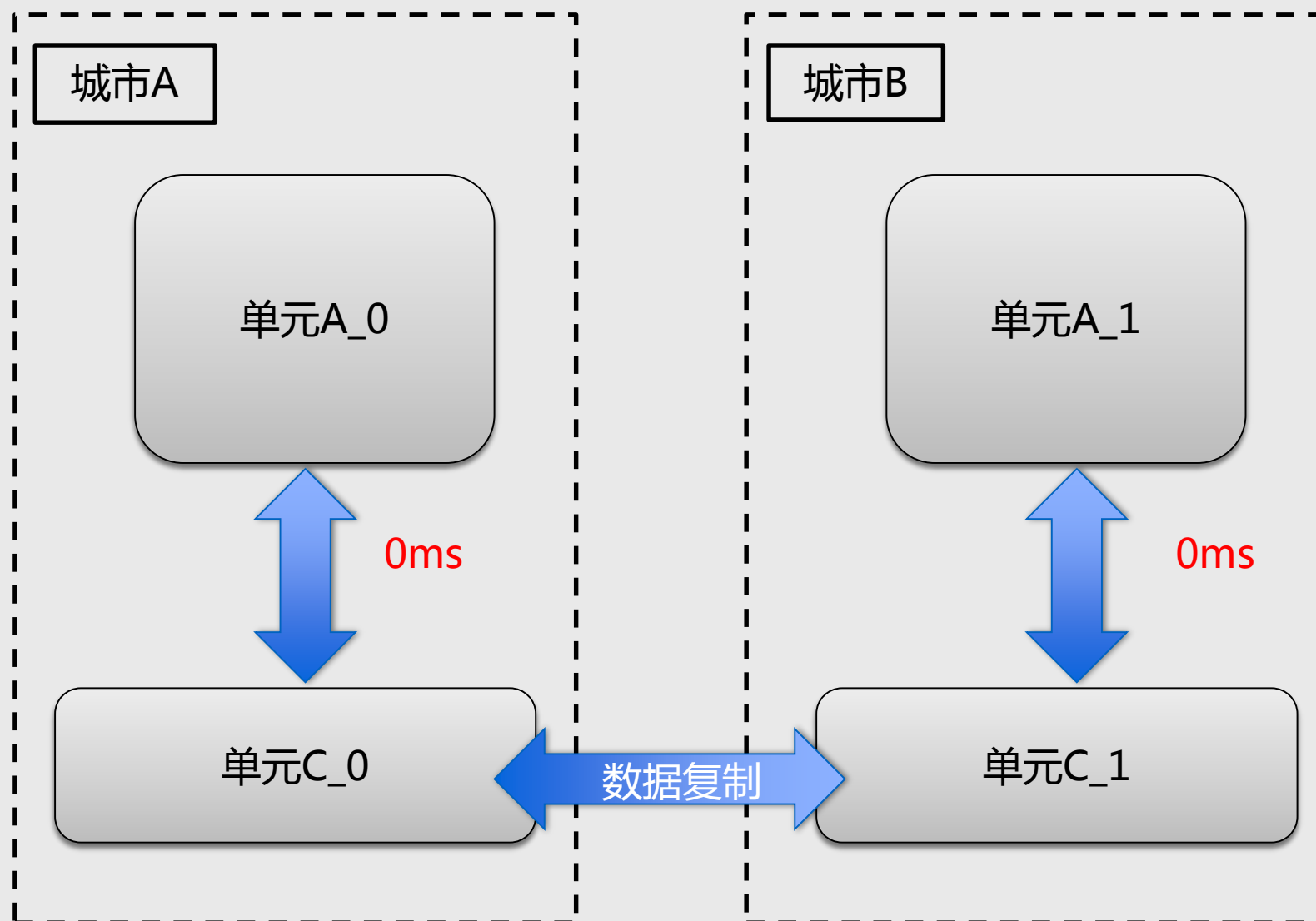




# 解决延时问题

## 引入单元C

无法拆分的全量数据，全局复制。  
支撑核心业务本地调用  
读写分离





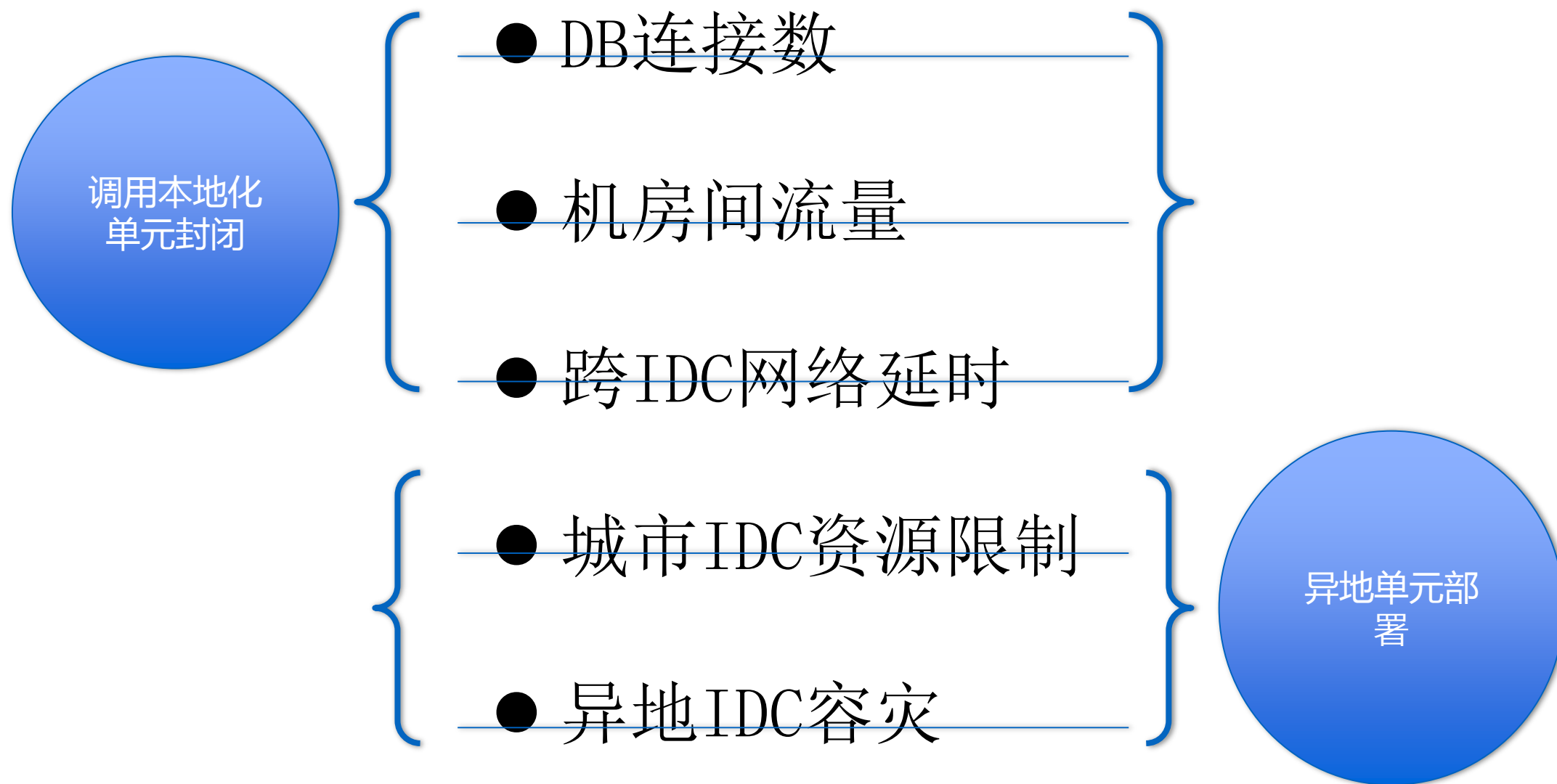


## 解决数据一致性和时效性问题

- 基于DB同步的数据复制：  
延时非敏感业务的异地复制方案；部分业务数据，可忍受3s时效性延迟（比如大部分的配置数据）。
- 基于消息系统的数据复制：  
对于延时非常敏感的业务，更低延时的实现方案；上层基于应用进行复制，减少延时。底层DB主备同步同时进行。

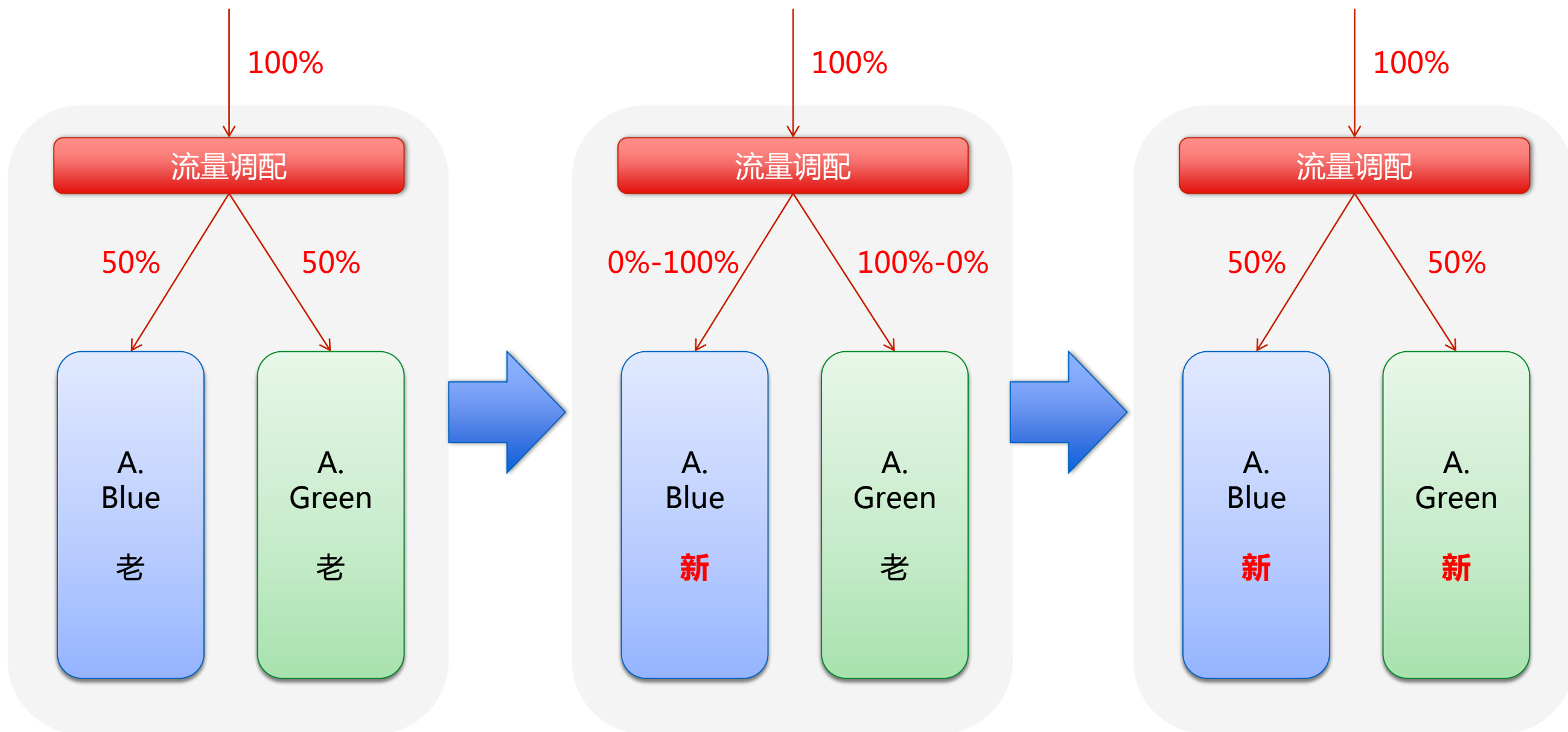


## 回顾核心问题



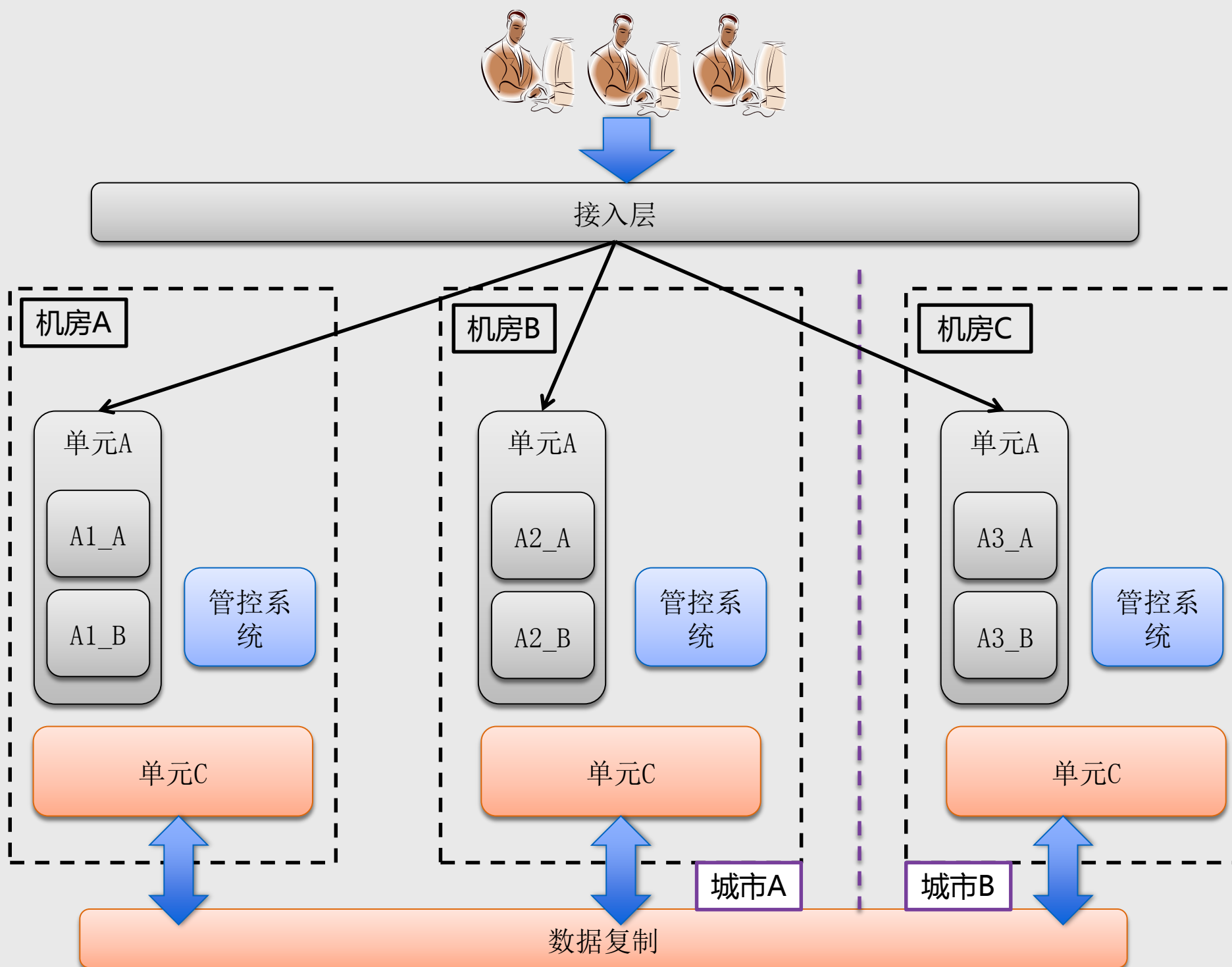


# 蓝绿发布



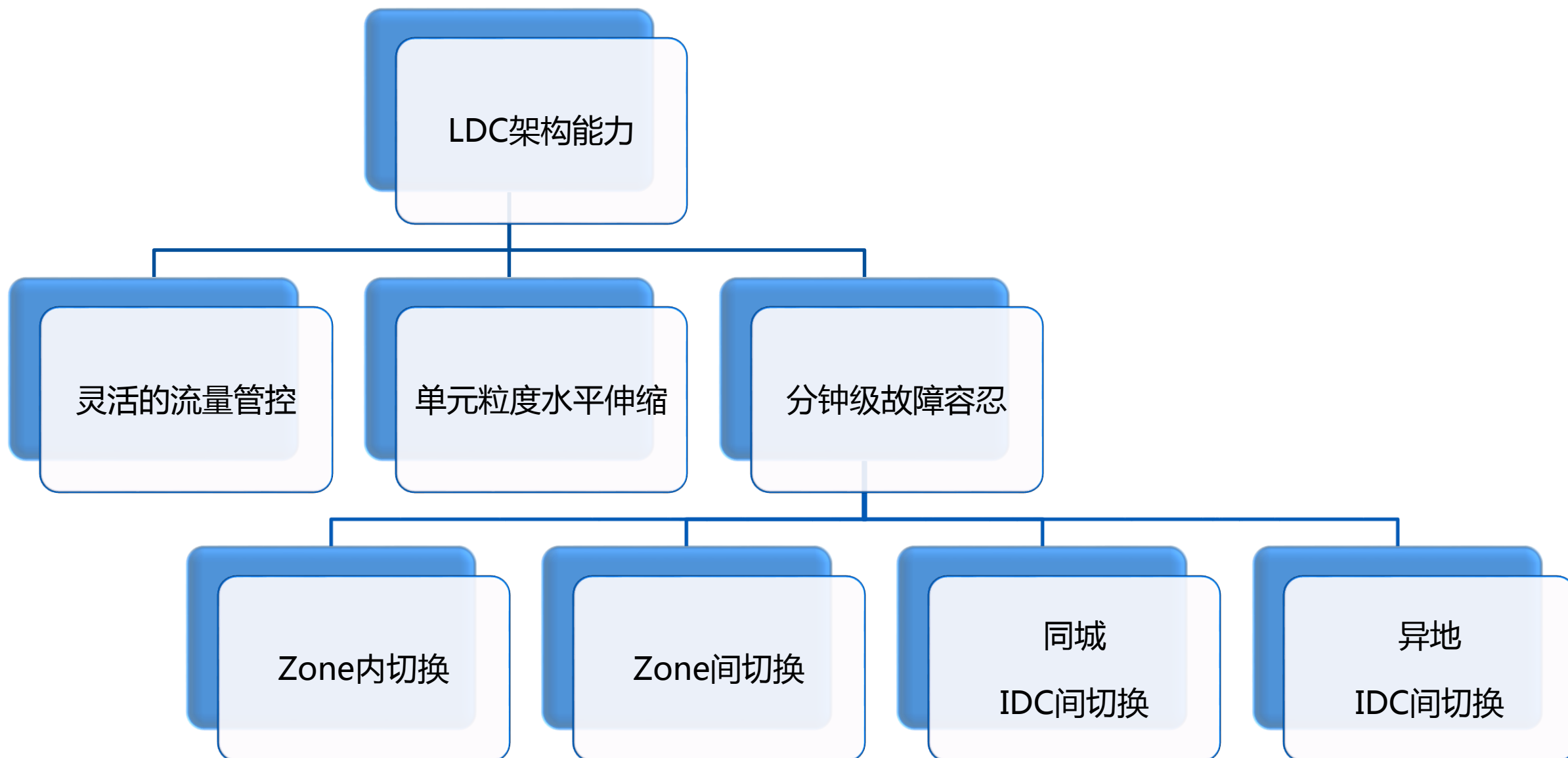


# 支付宝单元化全局架构





# 单元化下的高可用及容灾能力



# Thank you

Q&A



@阿里技术保障



@阿里技术保障