QCon 全球软件开发大会
INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

BEIJING 2017

系统架构演进和最佳实践

付钱拉 · SPEAKER · 冯忠旗

主办方 Geekbang. 极客邦科技 InfoQ

分享者

SHARER

冯忠旗

付钱拉高级技术经理

付钱拉
FUQIAN.LA

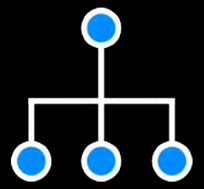
讲点儿什么

SPEAK SOMETHING

演进之路

THE PATH OF EVOLUTION

01 业务模型



02 架构演进过程

- 一无所有的初创期，单一架构轻装上阵
- 石器到工业的跃进，分布式架构保驾护航
- 化整为零，应对雪崩效应分而治之
- 从1到N的业务成长

03 最佳实践-如何早于用户发现问题

04 不忘初心 继续前进

BUSINESS MODEL

01 业务模型

BUSINESS MODEL

业务模型

BUSINESS MODEL

付钱拉 FUQIAN.LA

业务属性

BUSINESS ATTRIBUTES

聚合的复杂性、第三方依赖性

The complexity of aggregation, third-party dependency

金融系统

Financial system

安全性、中间账户

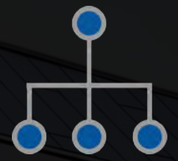
Security, intermediate account

实时性、一致性

Real-time, consistency

付钱拉 FUQIAN.LA

ARCHITECTURE
EVOLUTION
PROCESS



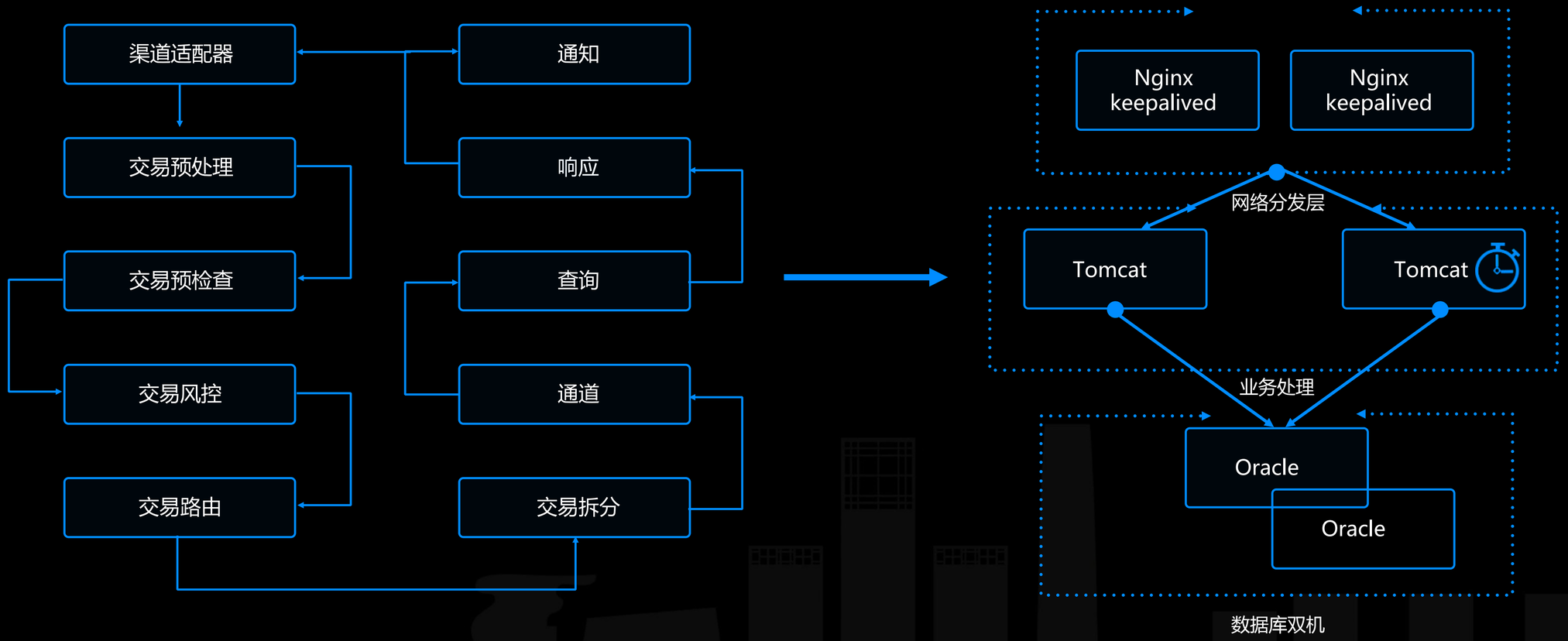
02 架构演进过程

ARCHITECTURE EVOLUTION PROCESS

- 一无所有的初创期，单一架构轻装上阵
- 石器到工业的跃进，分布式架构保驾护航
- 化整为零，应对雪崩效应分而治之
- 从1到N的业务成长

业务1.0 VS 单一架构轻装上阵

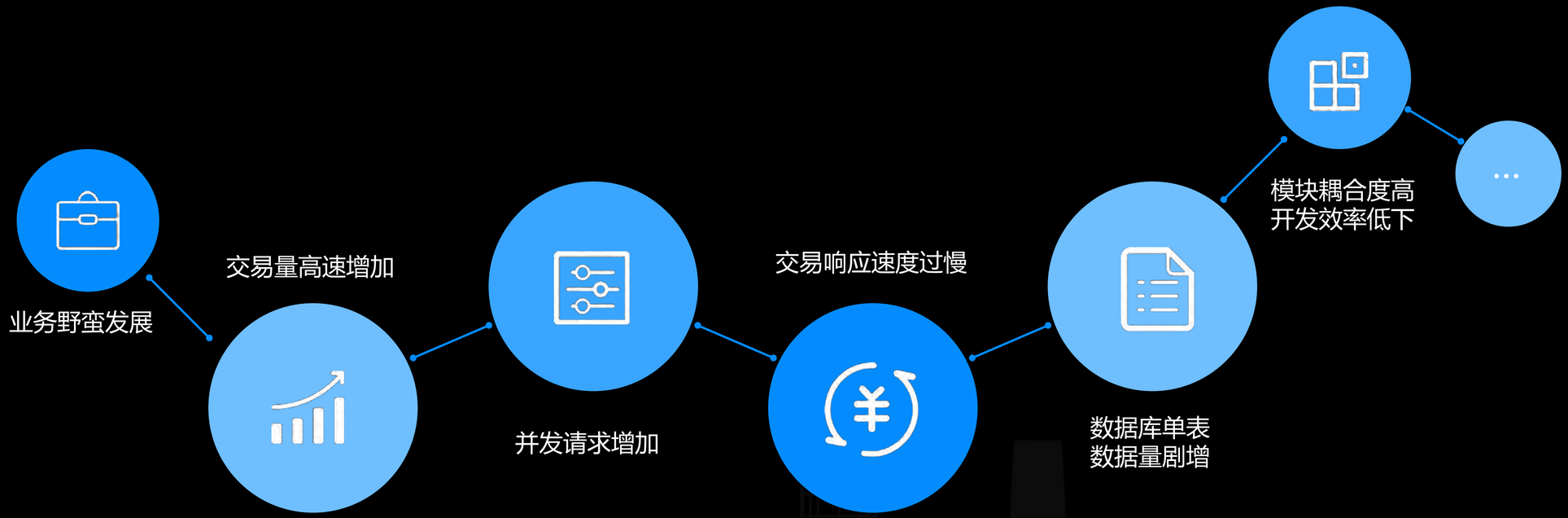
BUSINESS 1.0 VS A SINGLE STRUCTURE OF LIGHT INTO BATTLE



付钱拉
FUQIAN.LA

业务2.0痛点

BUSINESS 2.0 PAIN POINTS



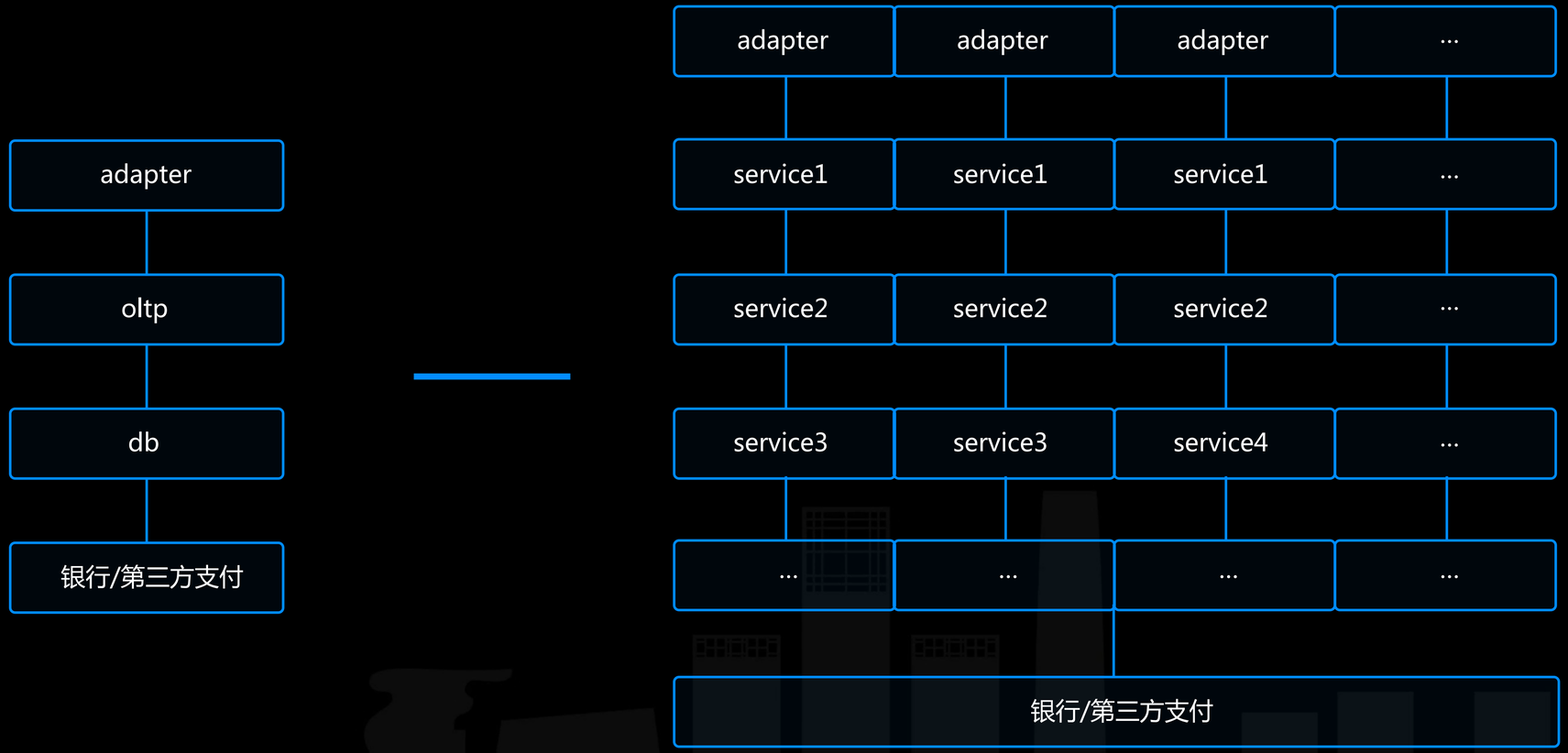
思考

THINKING

解耦	缓存	容灾
异步化	冷热数据隔离	限制
扩容	读写分离	池化
排队	分库分表	ACK
队列		...

变化

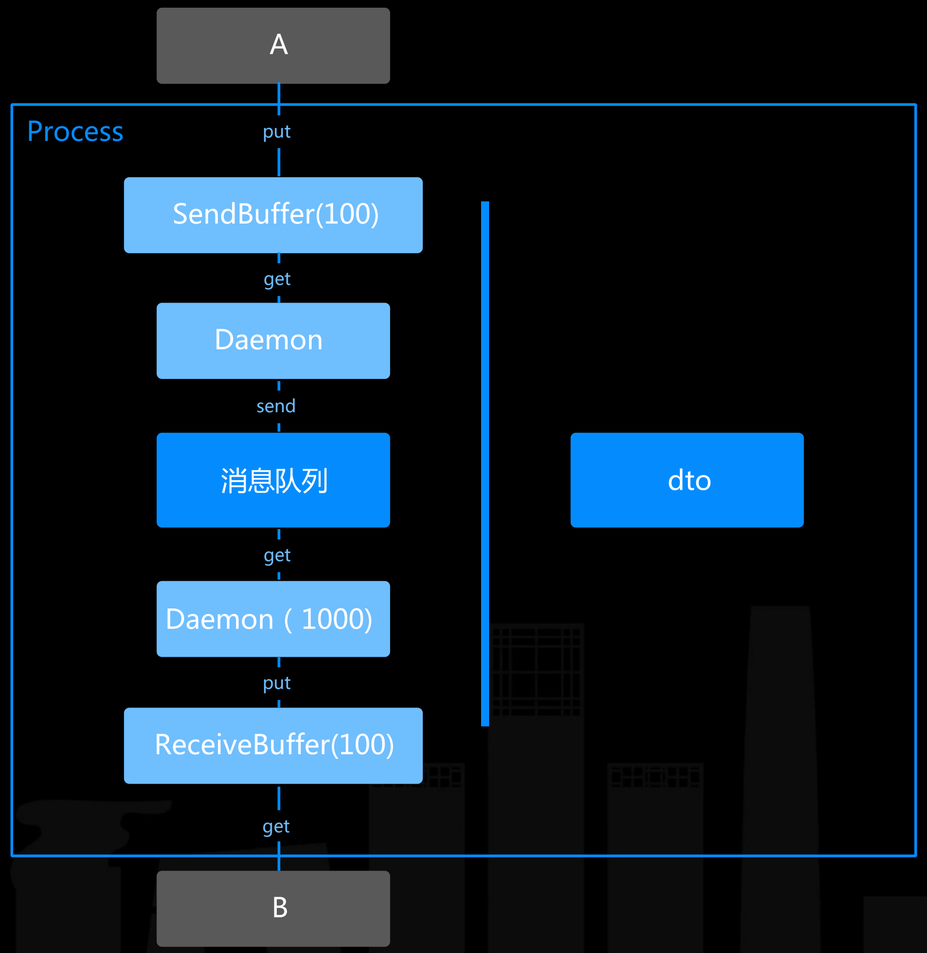
VARIETY



付钱拉
FUQIAN.LA

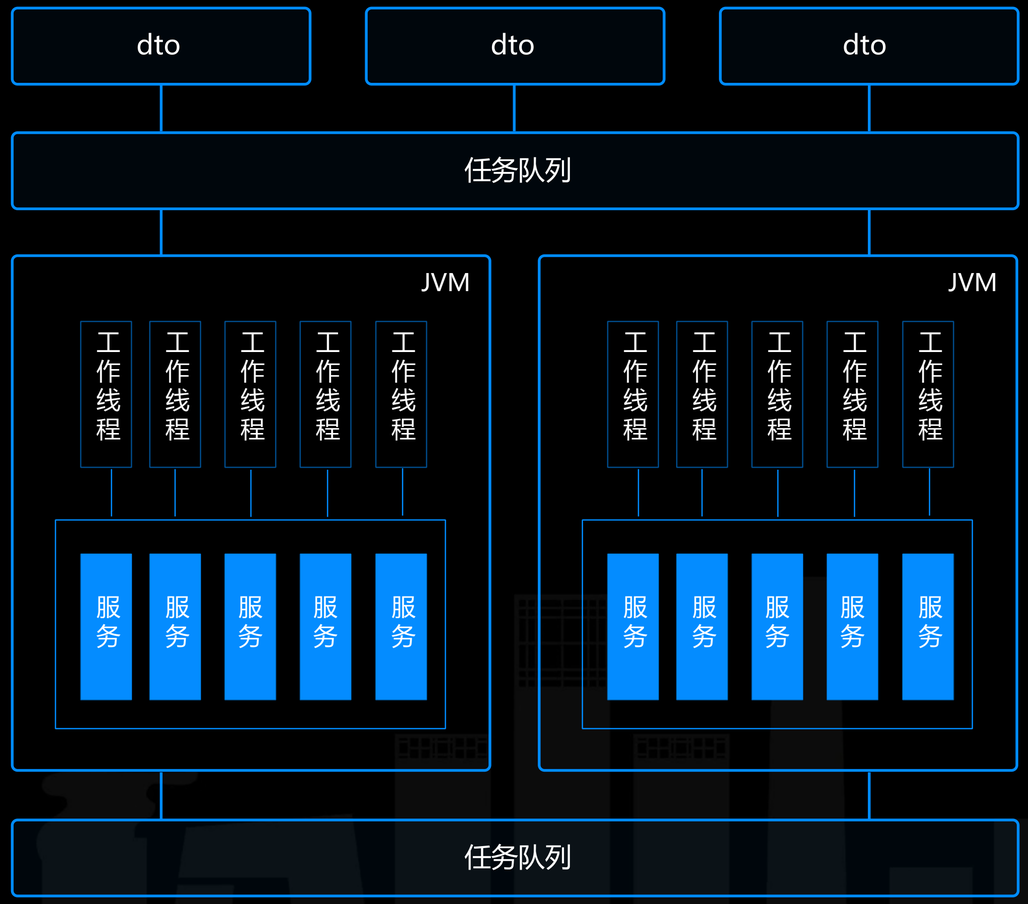
构思

IDEAS



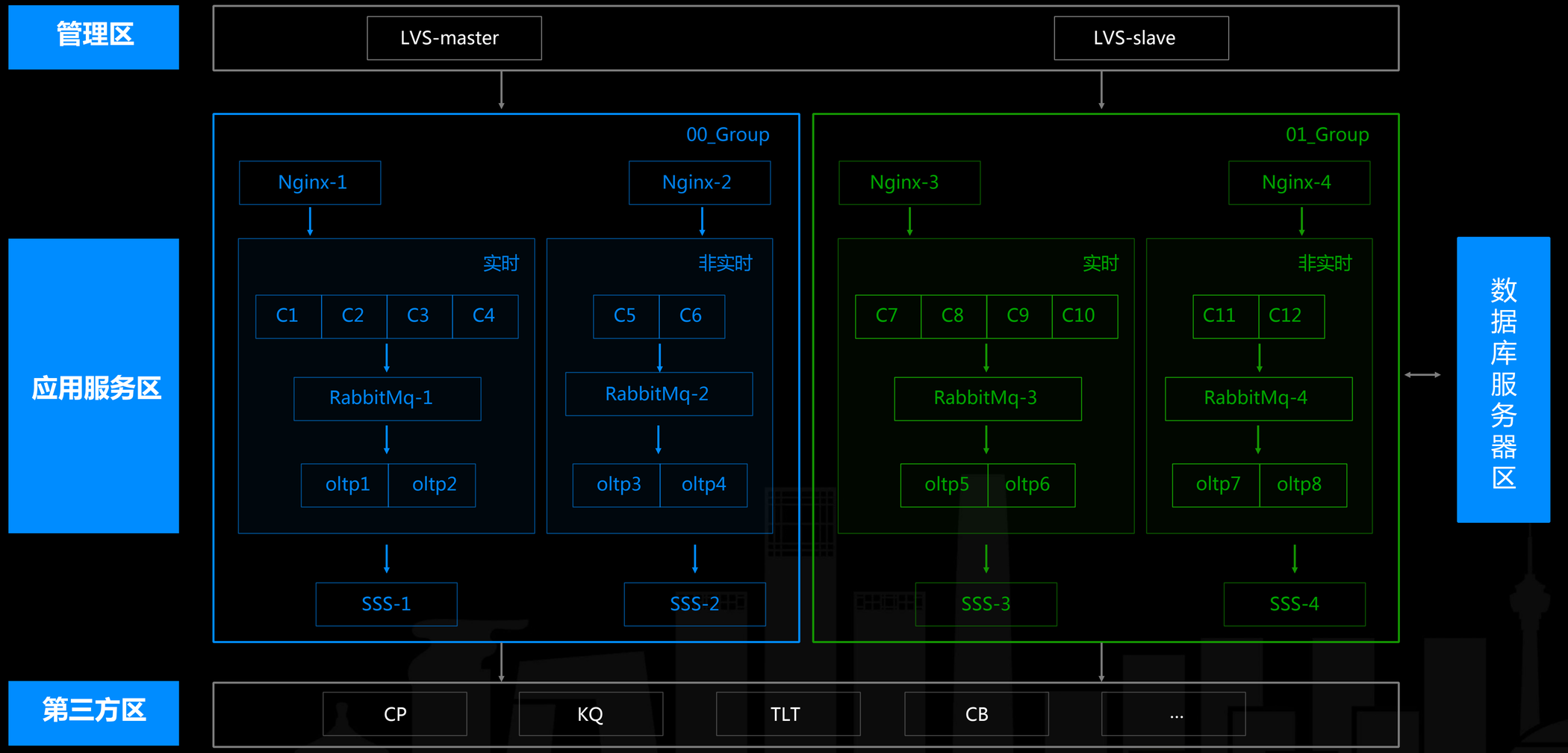
设计

DESIGN



基于消息的分布式架构

MESSAGE - BASED DISTRIBUTED ARCHITECTURE



PAIN POINTS

业务3.0痛点

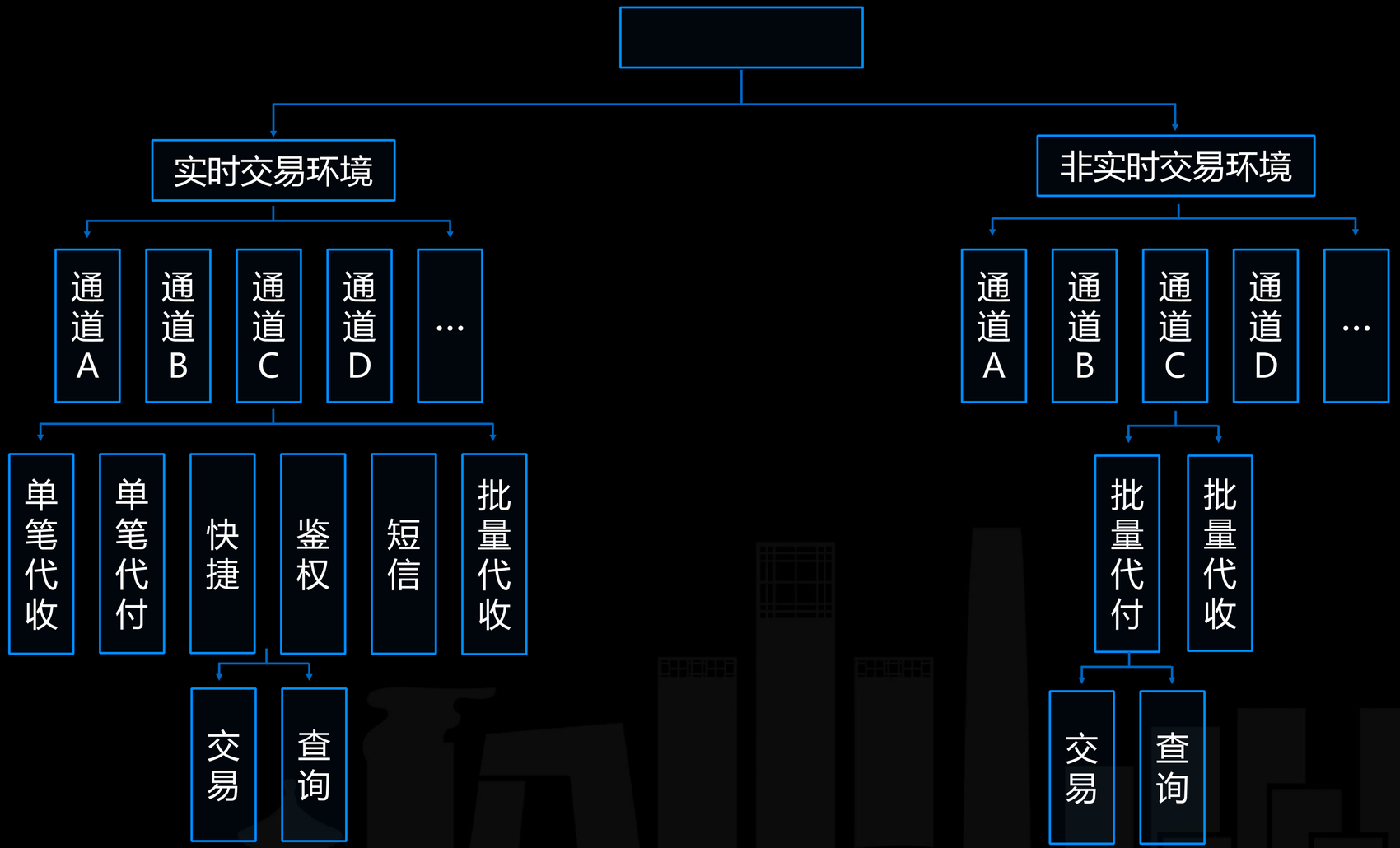
BUSINESS 3.0 PAIN POINTS

业务3.0远景

BUSINESS 3.0 VISION

分而治之

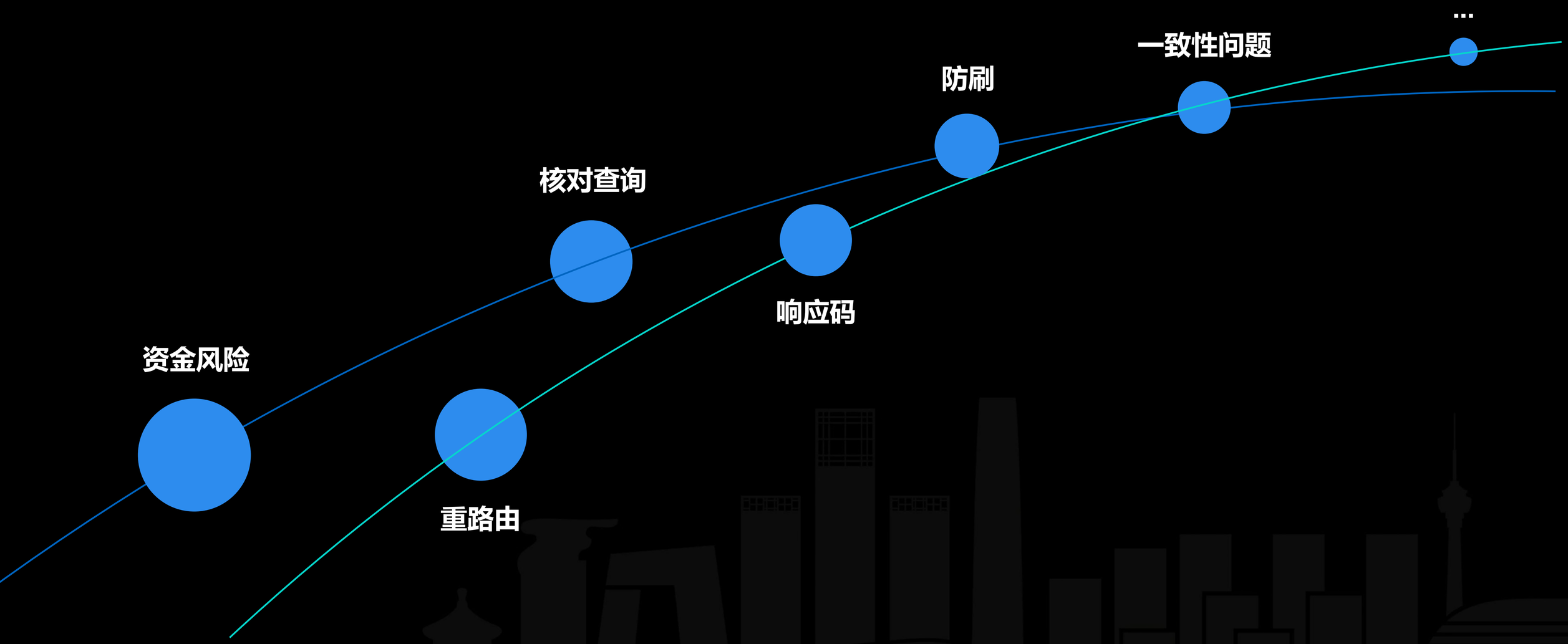
DIVIDE AND RULE IT



付钱拉
FUQIAN.LA

从1到N的业务成长

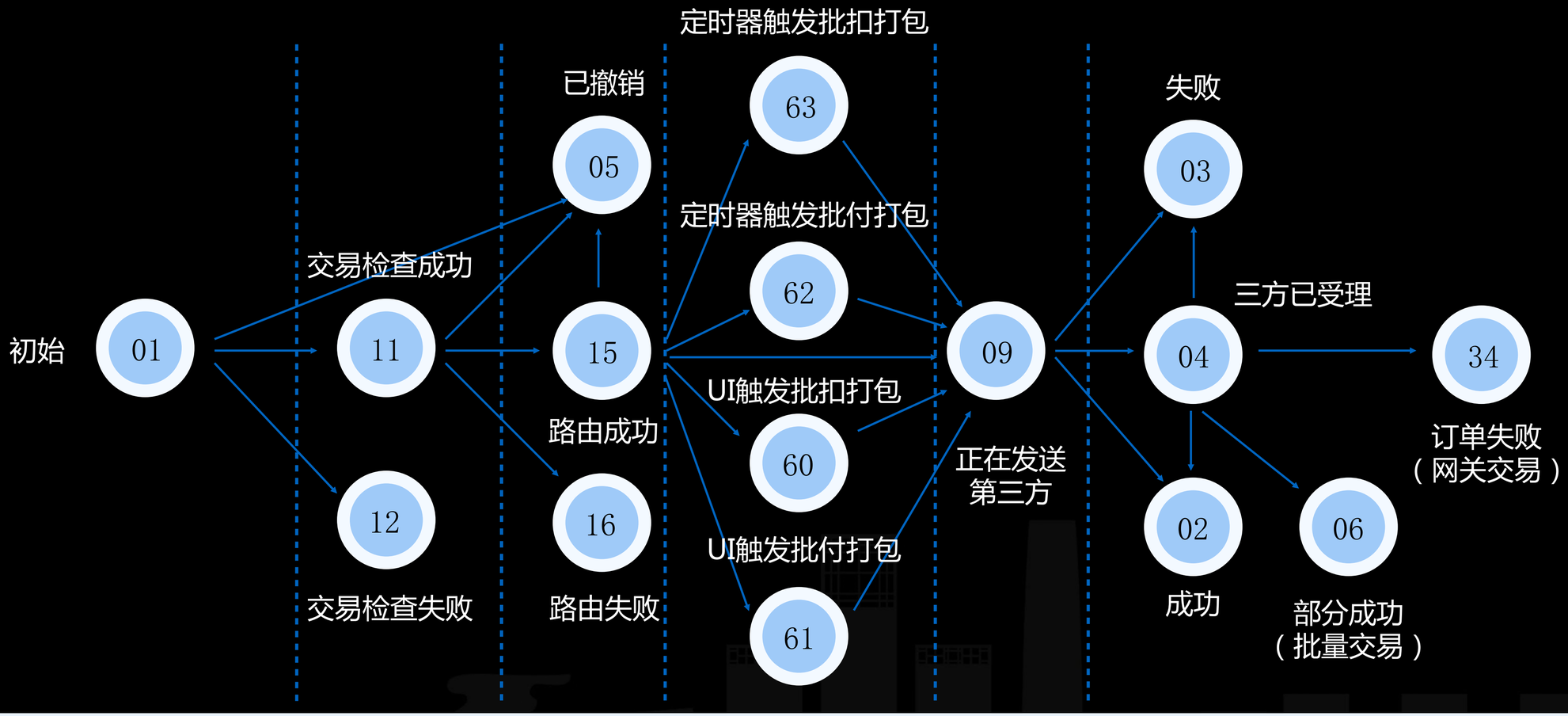
FROM 1 TO N BUSINESS GROWTH



付钱拉
FUQIAN.LA

有限状态机

FINITE STATE MACHINE

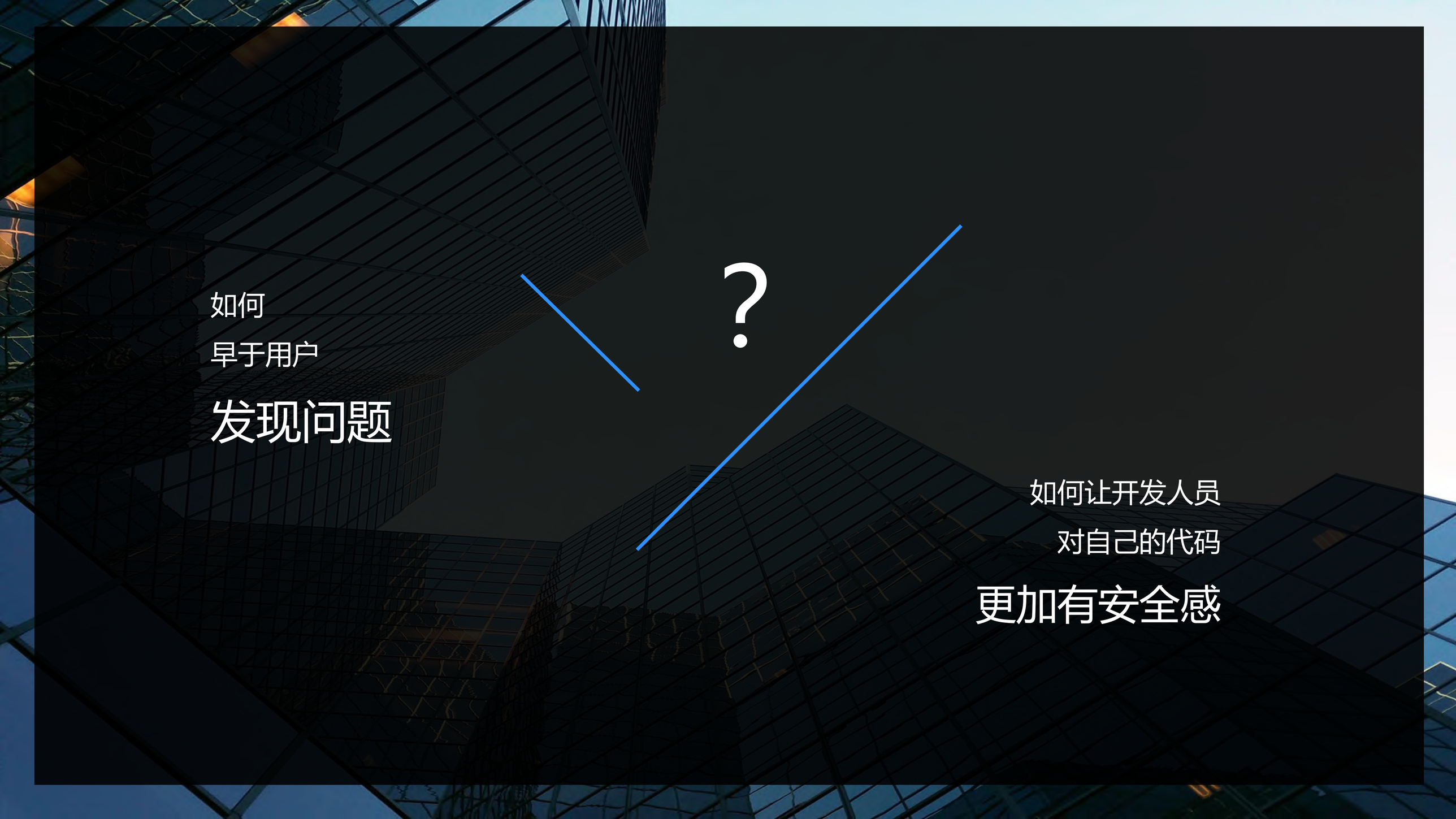


接入网关	交易检查	服务路由	批量任务	通道适配	响应处理

BEST
PRACTICE

03 最佳实践-如何早于用户发现问题

BEST PRACTICE - HOW TO FIND THE PROBLEM
EARLIER THAN THE USER


如何
早于用户
发现问题
?
如何让开发人员
对自己的代码
更加有安全感

有哪些困惑

WHAT IS THE CONFUSION

这些困惑的体现是什么呢？

- 开发人员如何提高代码质量，减少频繁迭代产生的bug？
- 线上环境突发事故，第一时间如何决策减少事故影响范围？
- 开发人员排查问题速度过慢？
- 随着业务的增长，问题越来越多，第一优先级需要解决什么？
- 系统突然CPU、内存利用率暴增，如何定位代码？
- 数据库连接数被耗尽，怎么办？
- 各种OOM如何预防？
- 随着系统交易量的增加，高可用系统的设计点很多，如何快速抓住建设要点？

思路

THINKING

需求设计阶段- 首次拦截

开发实现阶段-尽可能避免故障

- 预见运行期、所想即所得、限制和保护
- 设计可容错的系统（快速失败、超时、自动重路由）
- 设计具备自我保护的系统（拆分、限制、优雅停止）
- 制定合适的开发规范

系统运行阶段-及时发现故障

- 撒网
- 实时监控（傻瓜式、开发不用查日志）
- 可视化运营

系统运行阶段-出现故障快速解决故障

撒网与实时监控

REAL-TIME MONITORING

- 网络监控
- 主机监控
- 服务监控
  - 中间件、接口探测、日志抓取
- 业务监控
  - 状态类（响应码、交易状态、退款状态、商户状态）
  - 耗时类（交易总时长、分段时长、SQL耗时、代码耗时）
  - 统计类（订单量异常预警、非法IP预警、交易额过大）
  - 网络异常（单通道和多通道、不同的分布场景）

可视化运营

VISUAL OPERATION

```
2016-07-22 18:15:00.512||pool-73-thread-4||通道适配器||通道适配器-发三方后
||CEX16XXXXXXX5751||16201XXXX337||||||04||9000|| 【平台消息】处理中
||0000105||98XX543210||GHT||03||11||2016-07-22 18:15:00.512|| 张 张
||||01||tunnelQuery||true||||Pending||||10.100.140.101||8cff785d-0d01-4ed4-b771
-cb0b1faa7f95||10.999.140.101||0001||||0.01||||||||http://10.100.444.59:8080/re
gression/notice||||240||2016-07-20 19:06:13.000xxxxxxx
||2016-07-22 18:15:00.170||2016-07-22 18:15:00.496xxxxxxxxxxxxxxxxxxxx
||2016-07-2019:06:13.000||||||||01||0103||111xxxxxxxxxxxxxxxxxxxxxxxxxx
||8fb64154bbea060afec5cd2bb0c36a752be734f3e9424ba7xxxxxxxxxxxxxxxxxxxx
||622xxxxxxxxxxxxxxxx||9bc195a59dd35a47||f2ba5254f9e22914824881c242d211
||||||||||||||||||6xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx010||||||||||
```

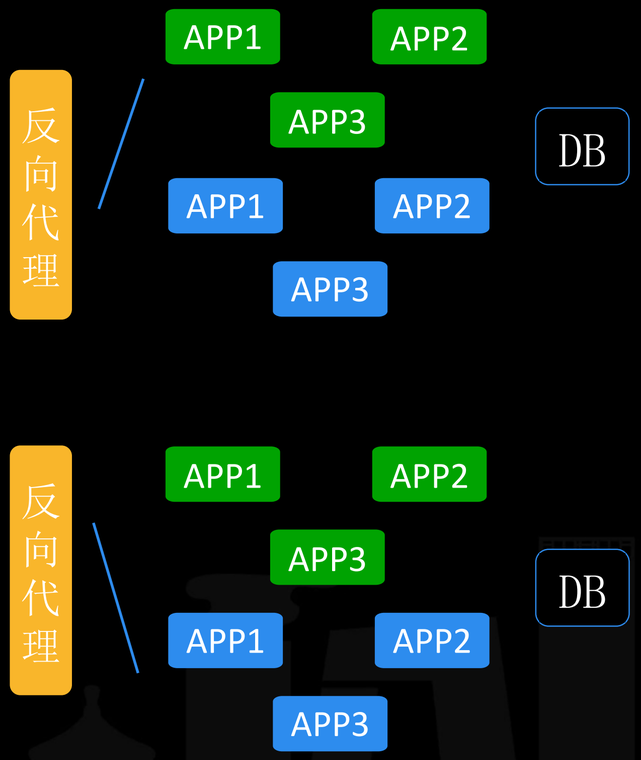
	顺序ID	模块操作时间	模块名称	模块描述	操作者	系统单号	通道名称	通道商户号	支付类型	交易状态
1	6162046607794634753	2016-07-22 08:16:44.044	渠道适配器	渠道适配器发起	channleAdapter				单笔代扣	
2	6162046607874326529	2016-07-22 08:16:44.343	交易预处理	预处理业务检查通...	tradePreCheck				单笔代扣	处理中
3	6162046608067264513	2016-07-22 08:16:44.389	交易检查	交易检查处理完毕	tradeCheck				单笔代扣	处理中
4	6162046608130179073	2016-07-22 08:16:44.404	风控检查	交易风控处理完毕	tradeRisk				单笔代扣	处理中
5	6162046608285368321	2016-07-22 08:16:44.441	路由	路由处理完毕	tradeRouter		FY		单笔代扣	等待发送
6	6162046608386031617	2016-07-22 08:16:44.465	通道适配器	通道适配器-发三...	tunnelTrade		FY		单笔代扣	已发送
7	6162046622755717121	2016-07-22 08:16:47.892	通道适配器	通道适配器-三方...	tunnelTrade		FY		单笔代扣	交易失败
8	6162046622806048769	2016-07-22 08:16:47.904	单笔响应处理	更新交易状态	singleResponse		FY		单笔代扣	交易失败
9	6162046622877351937	2016-07-22 08:16:47.047	渠道适配器	渠道适配器应答	channleAdapter				单笔代扣	交易失败
10	6162047690633904130	2016-07-22 08:21:02.497	通道适配器	通道适配器-发三...	tunnelQuery		FY		单笔代扣	交易失败
11	6162047690709401602	2016-07-22 08:21:02.515	单笔响应处理	查询后更新交易状态	singleResponse		FY		单笔代扣	交易失败

付钱拉 FUQIAN.LA

快速响应故障

QUICK RESPONSE FAILURE

优先恢复　蓝绿部署　提前准备　服务降级

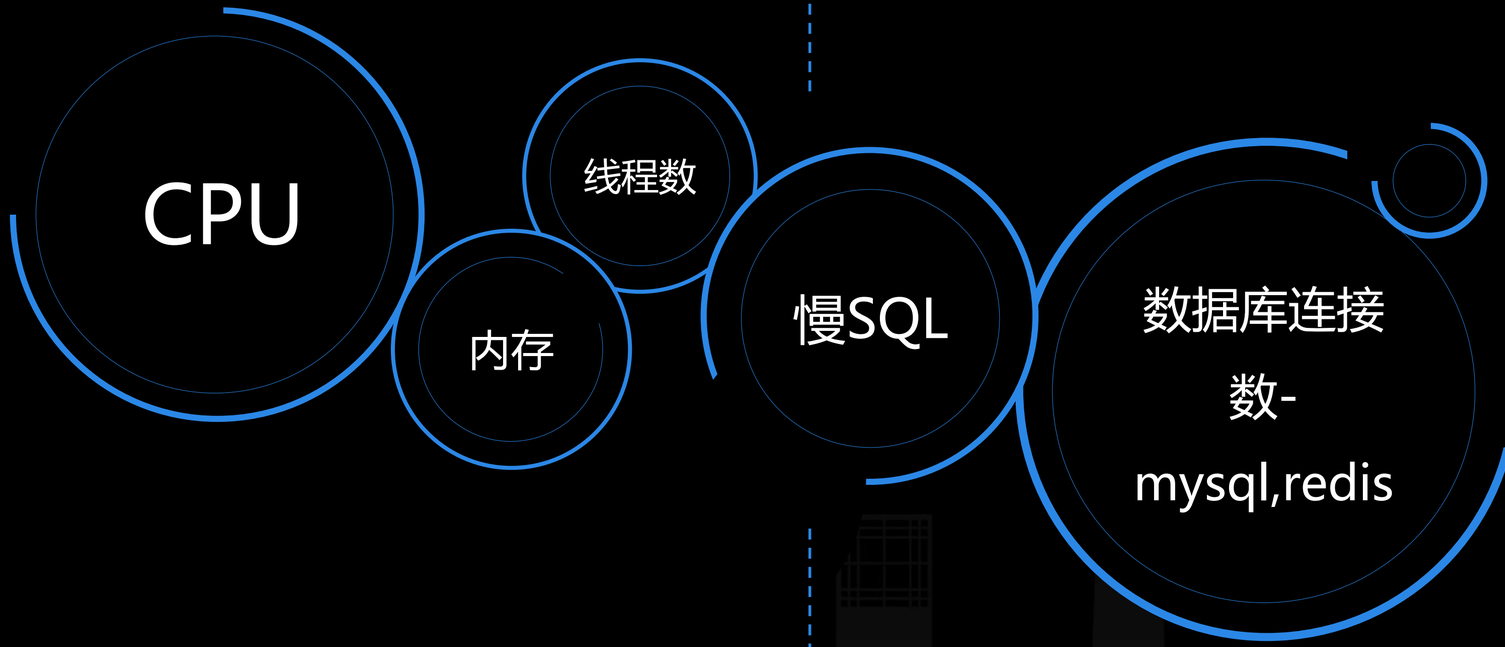


kill 慢SQL、摘节点、线程栈和内存堆现场保留

Java系统常见问题

JAVA SYSTEM FAQ

我们解决了以下系统自身问题



最后的问题是什么？

DO NOT
FORGET THE
BEGINNING

04 不忘初心，继续前进

DO NOT FORGET THE BEGINNING, CONTINUE TO MOVE FORWARD

付钱拉
FUQIAN.LA

继续前行

MOVE ON

当前架构的适用范围

中小规模

硬件规模不过百

技术债积累期

团队规模不过百

数据规模日交易量不过千万

微服务迁移

配置繁琐

调用混乱

组装困难

协议单一

统一API网关

规范

SPECIFICATION

一 编程规范

1.业务代码中所有SQL耗时打印耗时

2.业务代码中关键方法打印耗时

3.和第三方接口交互，需要设置连接超时和读取超时时间，避免同步线程阻塞

4.和第三方接口交互，需要考虑是否需要通过代理出网

5.和第三方接口交互，需要考虑是否要相互添加白名单

6.和第三方接口交互，需要考虑设置合适的work线程符合第三方并发数量限制

二 安全规范

1.页面请求参数严格限制或者校验处理，防止SQL注入

2.页面URL请求做细粒度的权限拦截，防止访问权限过大

3.部署在公网的应用做好防止XSS攻击的防范措施

4.和第三方系统交互需要互加白名单确保安全

5.系统全站提供HTTPS服务

6.和第三方系统交互报文需要加密传输

7.用户敏感数据做数据脱敏

8.预防页面被频繁请求，占用系统资源

9.预防API被频繁请求，占用系统资源

规范（续）

SPECIFICATION (CONTINUED)

三 性能规范

1.常见OOM预防

2.禁止应用中显式创建线程，避免不可控出现unable to create new native thread

3.控制select/update/delete/insert的数据级和可变集合的size，避免随着业务增加内存数据量不可控

4.页面查询不推荐全表查询，查询通过查询条件限制查询条数

5.页面下载条数和下载次数做限制，避免请求过多导致OOM

6.SQL优化目标必须满足range、ref或者consts，不可以是all类型，避免慢SQL导致连接数耗尽影响业务功能

7.代码书写中考虑MySQL中共享锁和排它锁场景，预防产生死锁

8.代码中不建议使用@Transactional，因为一般业务场景中用不到，它影响数据库性能并且多个操作可能在并发下导致数据库死锁

9.数据库单表达到一定数据量级需要做分库分表或者冷热数据隔离，避免业务增加带来的性能问题

10.尽量避免使用全局变量防止并发出现线程安全问题，从而影响业务

11.定时器问题预防

定时器浪打浪情况下，任务重复处理会导致资金风险，建议使用redis避免

定时器浪打浪情况下，启动多个定时器即默认启动多个线程，影响系统性能

定时器浪打浪情况下，如果定时任务处理过慢会导致内存耗尽

12.避免系统中出现单点故障，包括中间件和应用程序等所有的节点

13.能异步处理的别同步处理，异步可以释放线程资源，避免阻塞，提高响应效率

14.随着业务量的增加，考虑功能拆分和数据库表拆分，除此支付系统建议按照通道拆分，不同的通道指定独立的work线程，分而治之，避免相互之间影响；提高并发的一个思路就是拆分，拆分后通过异步提高并发效率

付钱拉

让 金 融 更 简 单

金融系统最大的成功是部署了一套可靠的、稳定的解决方案，

而不是什么新奇的东西。

付钱拉

让 金 融 更 简 单

付钱拉官网

付钱拉微信公众号

关注QCon微信公众号，
获得更多干货！

Thanks!

INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

主办方 Geekbang. 极客邦科技 InfoQ