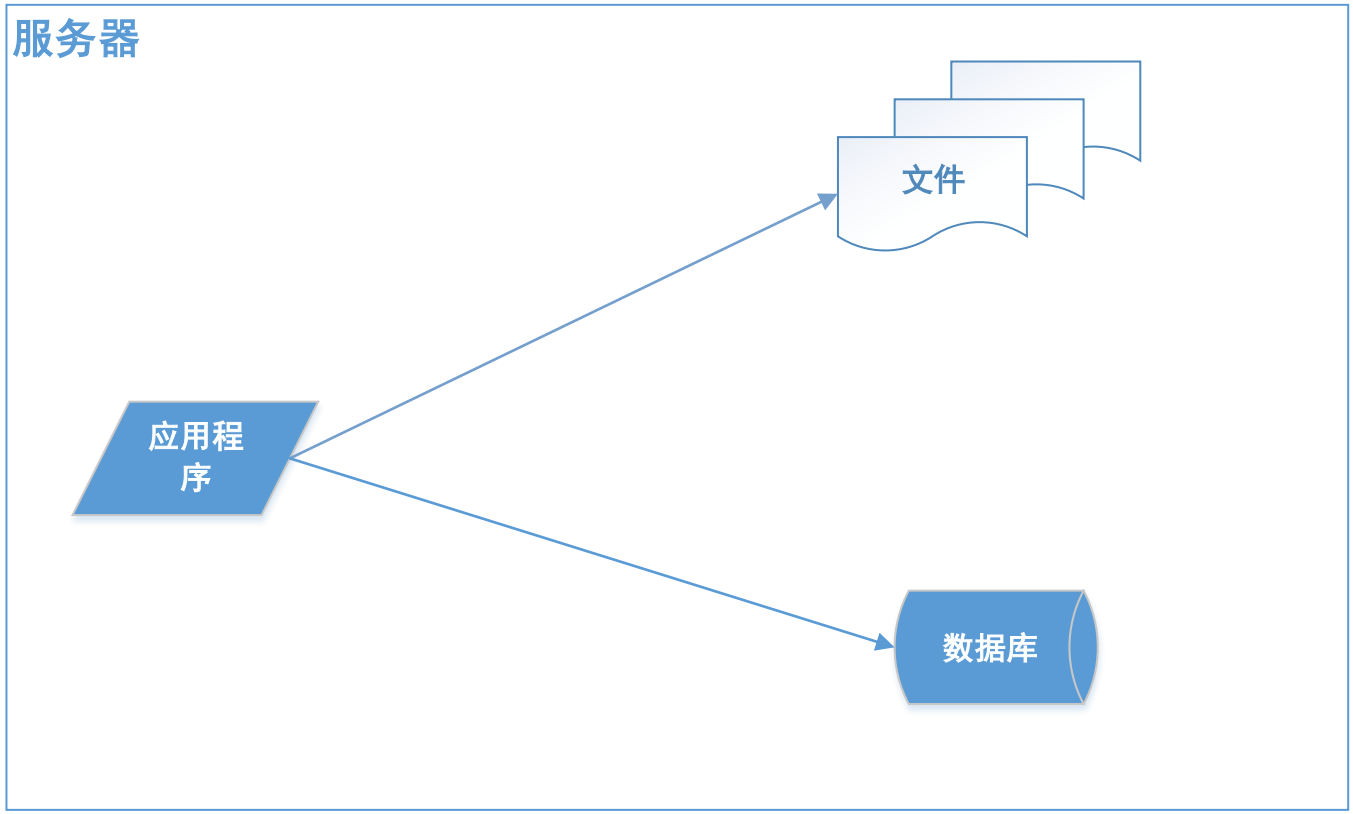
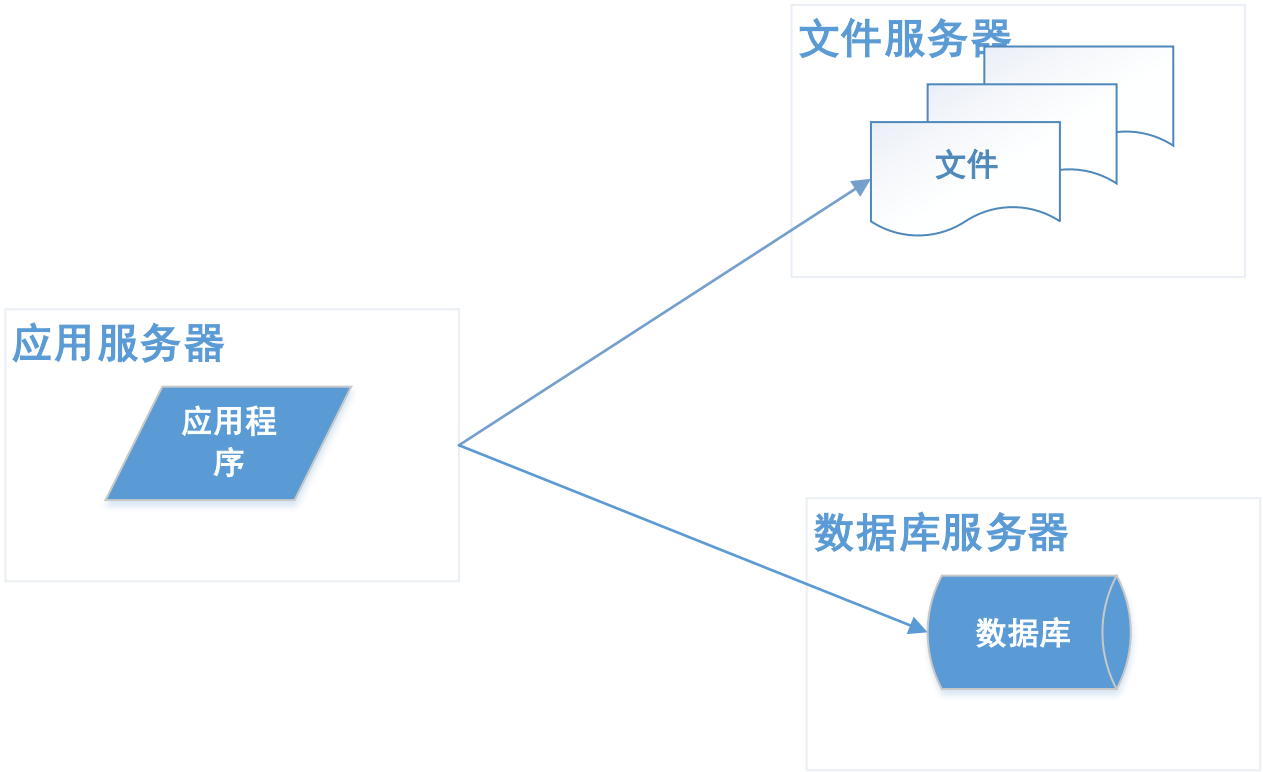


网站架构演进：初始阶段

1、应用、DB、文件等资源都在同一服务器上。（LAMP开源免费架构）

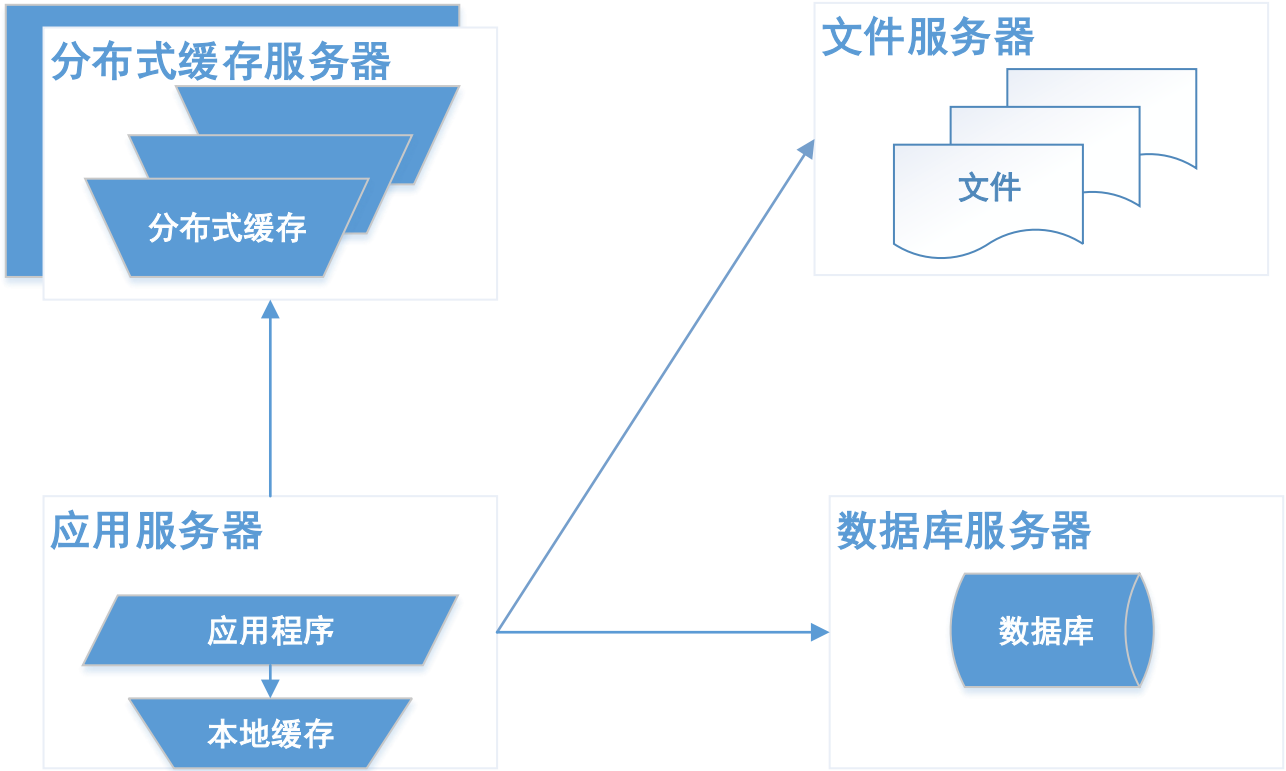
缺点：性能差、存储不足



网站架构演进：应用、数据分离

1、应用程序、数据库、文件存储分离，使用三台服务（缓解性能、存储不足）
2、应用服务器处理大量业务（CPU）
3、文件服务器存储文件（硬盘）
4、数据库服务器存储用户数据（硬盘、内存）

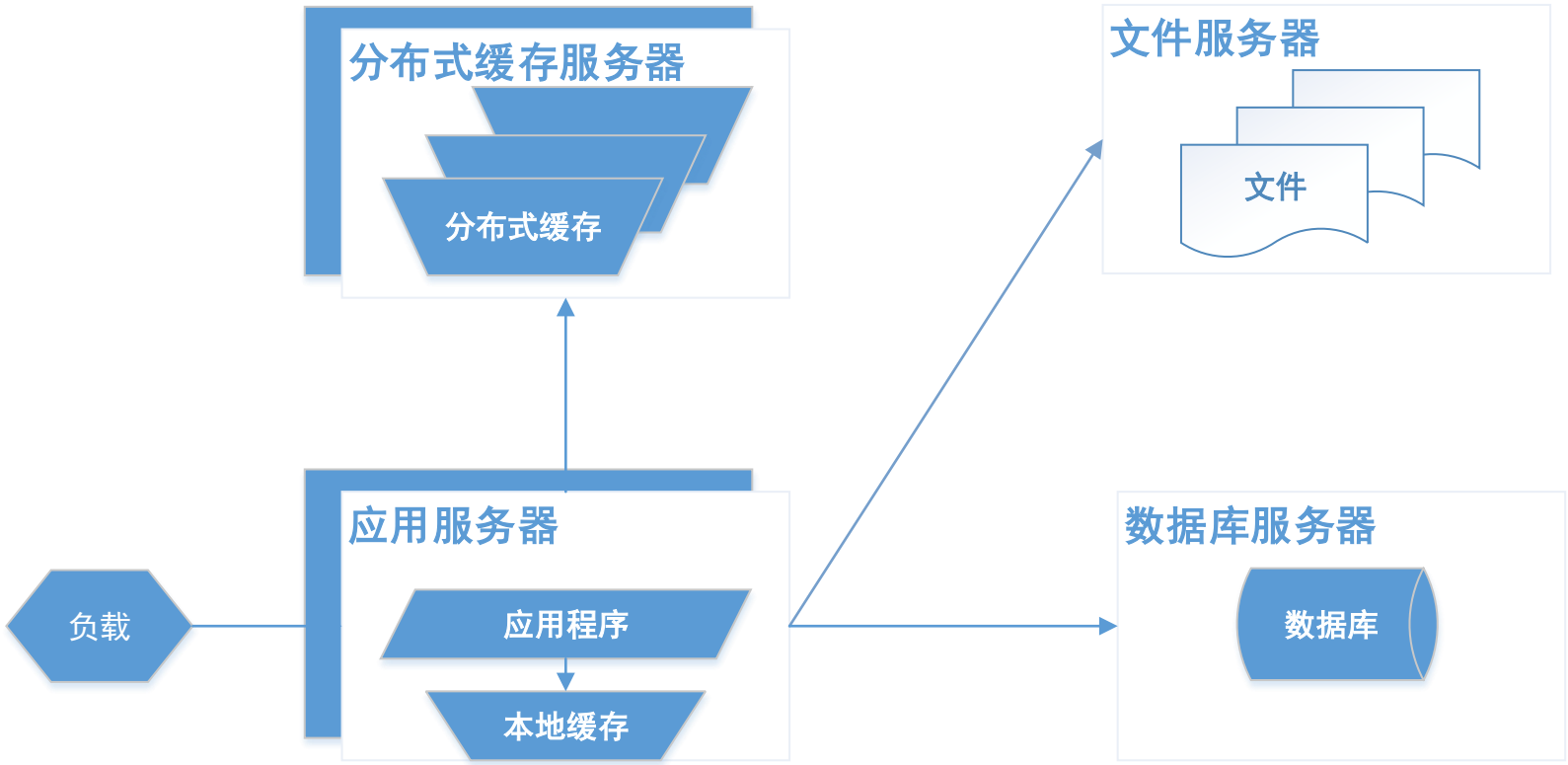
缺点：数据库压力太大，访问延迟



网站架构演进：应用缓存治理

1、使用本地、远程分布式缓存服务（缓解数据库访问压力）
2、本地缓存速度快但内存受限
3、分布式缓存解决内存受限瓶颈

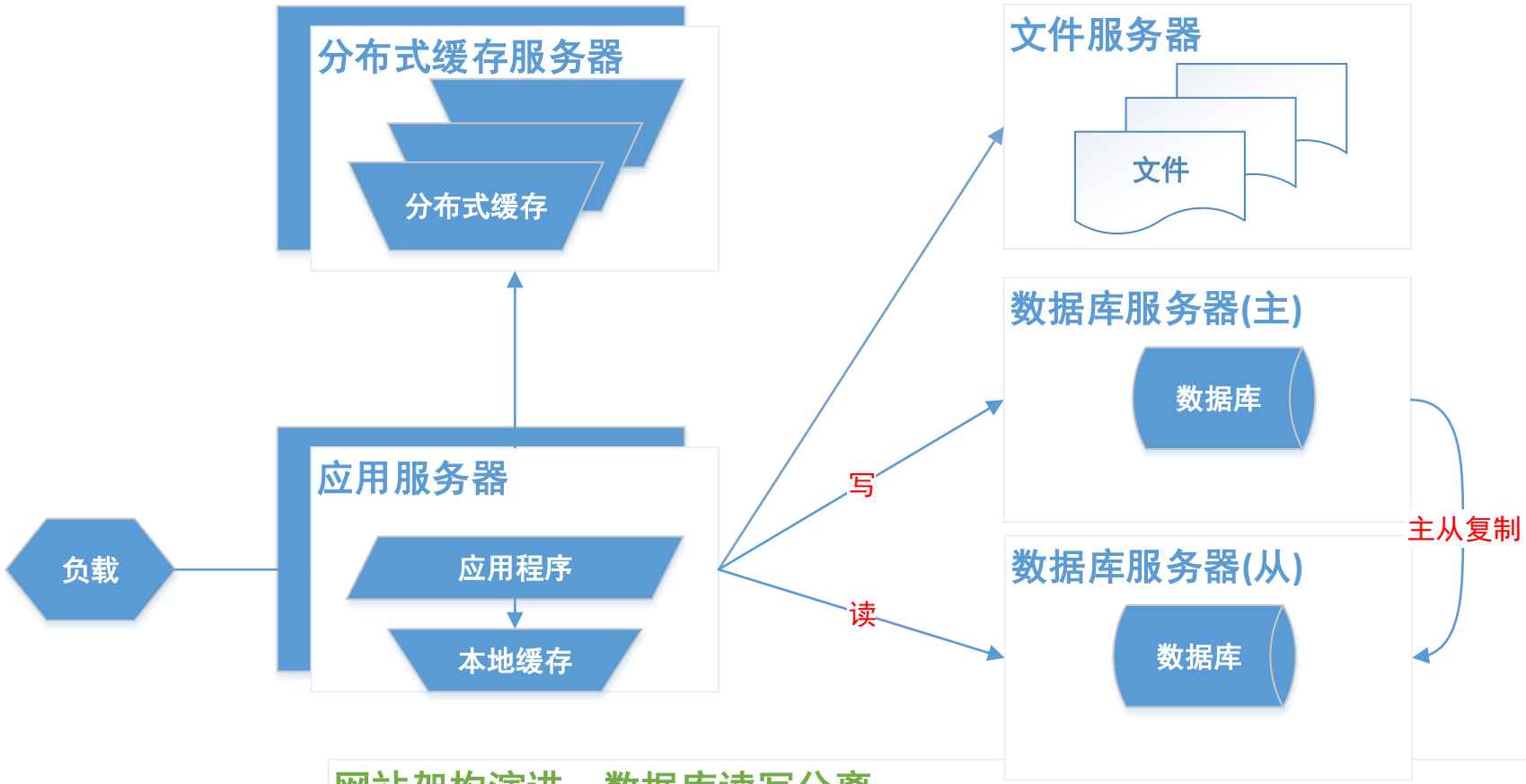
缺点：高并发请求连接受限，单一应用服务器访问瓶颈



网站架构演进：应用服务集群

1、应用服务采用集群方式，集群服务通过负载提供相同服务（缓解应用服务访问压力）
2、部署相同的应用服务到不同虚拟机/物理机中
3、通过软硬件提供负载均衡服务，指向应用服务集群；访问请求均衡分发到应用服务集群

缺点：业务规模增长，数据库部分读写出现瓶颈（部分读取操作、全部写操作负载压力瓶颈）



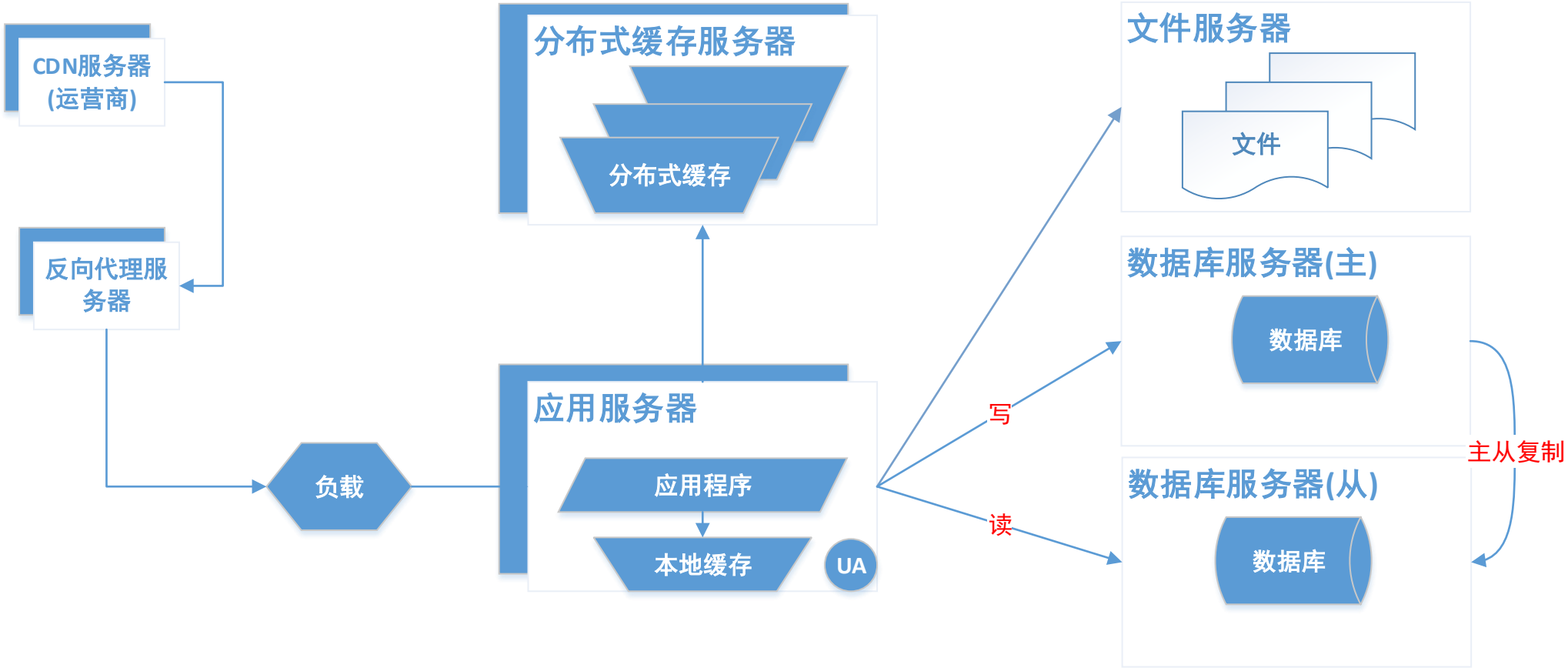
网站架构演进：数据库读写分离

1、利用数据库主从复制，分离数据读写服务（缓解数据库负载压力）

2、Master数据库提供读写功能，Slave数据库提供只读服务

3、Slave数据通过数据库主从复制功能实现数据同步

缺点：所有请求均被应用服务端接收，提供的服务不能快速响应或接收新的请求，用户体验差；



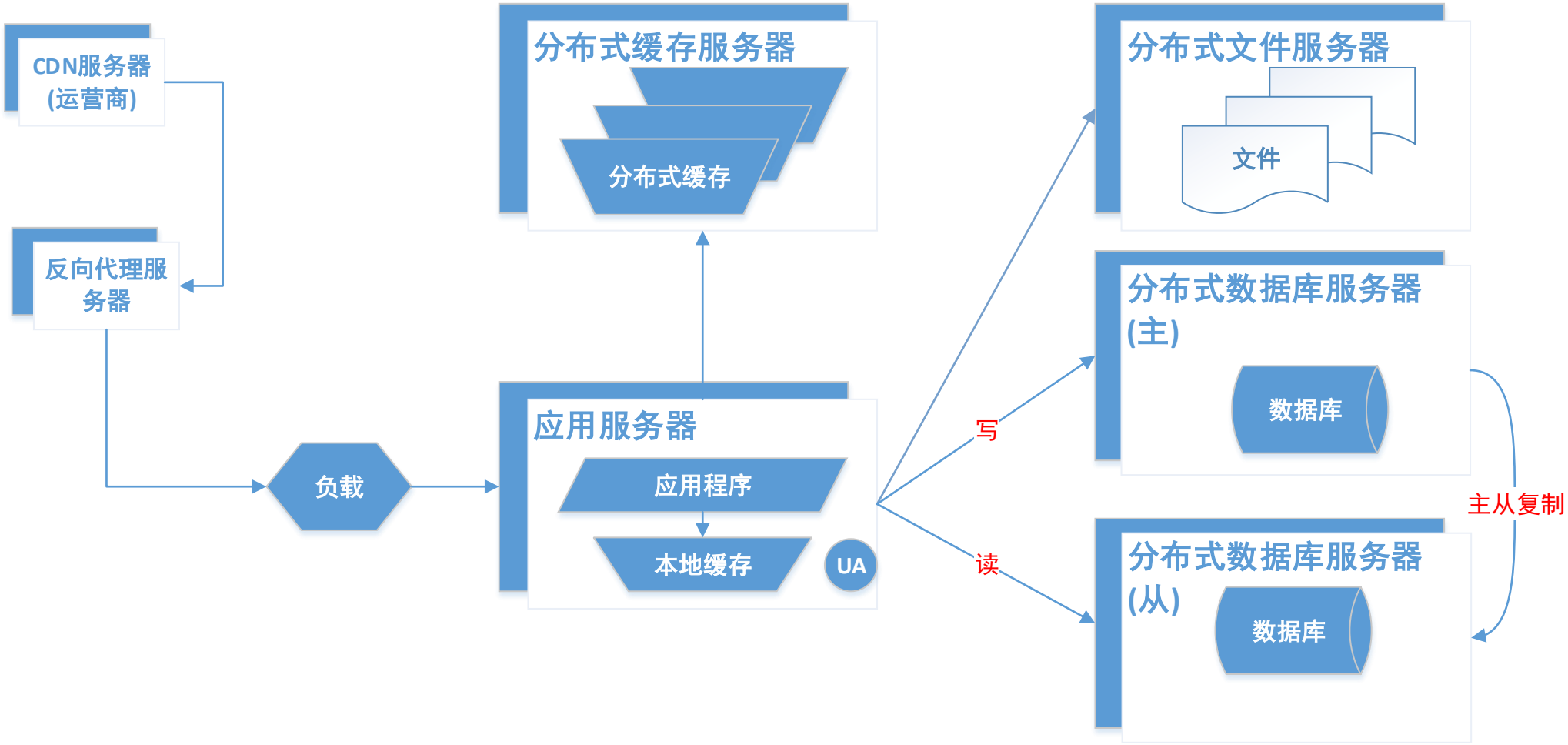
网站架构演进：CDN、反向代理缓存加速网站访问速度

1、利用CDN、反向代理缓存，尽早返回数据给用户，加快用户访问速度，减轻后端服务器负载压力（加快部分服务返回时间，提升用户体验）

2、CDN部署在运营商机房，从用户最近服务器返回数据给用户

3、反向代理部署在平台网络机房，接收到服务后，首先访问到反向代理服务器，缓存有用户数据直接返回，没有则继续向应用负载转发。

缺点：单一文件、数据库服务器不能支撑业务增长



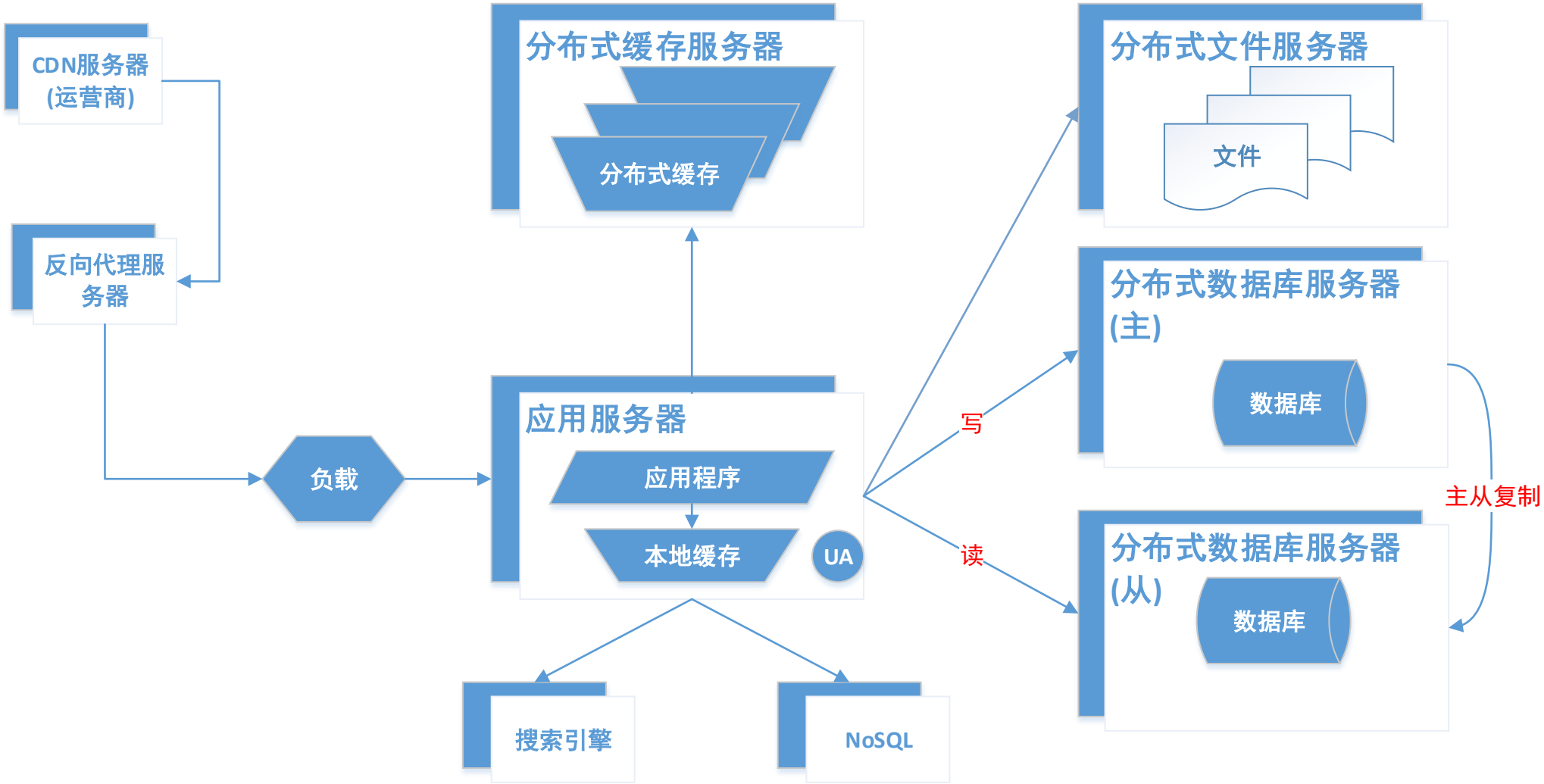
网站架构演进：分布式数据库、文件系统

1、分布式文件系统替代单一文件系统，提升读写性能

2、分布式数据库替代单一数据库系统，同时沿用Master、Slave模式，解决大数据存取（按业务拆分，DB最后手段）

3、

缺点：传统关系型数据库不能满足大数据查询、检索、统计

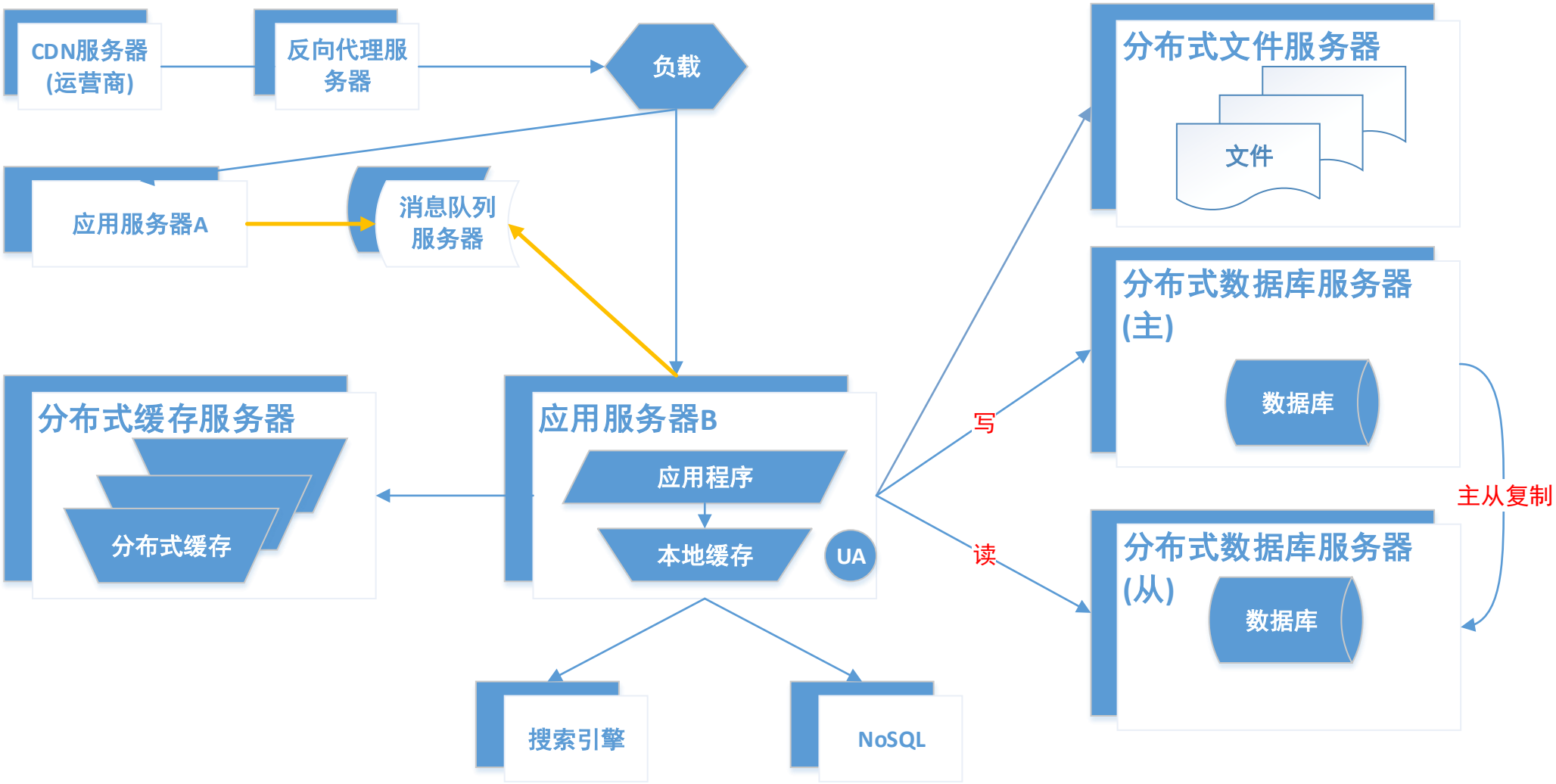


网站架构演进：NoSQL、搜索引擎

1、集成NoSQL解决大数据存取，提升系统可伸缩性、读取性能

2、集成搜索引擎，提升大数据搜索性能及效率

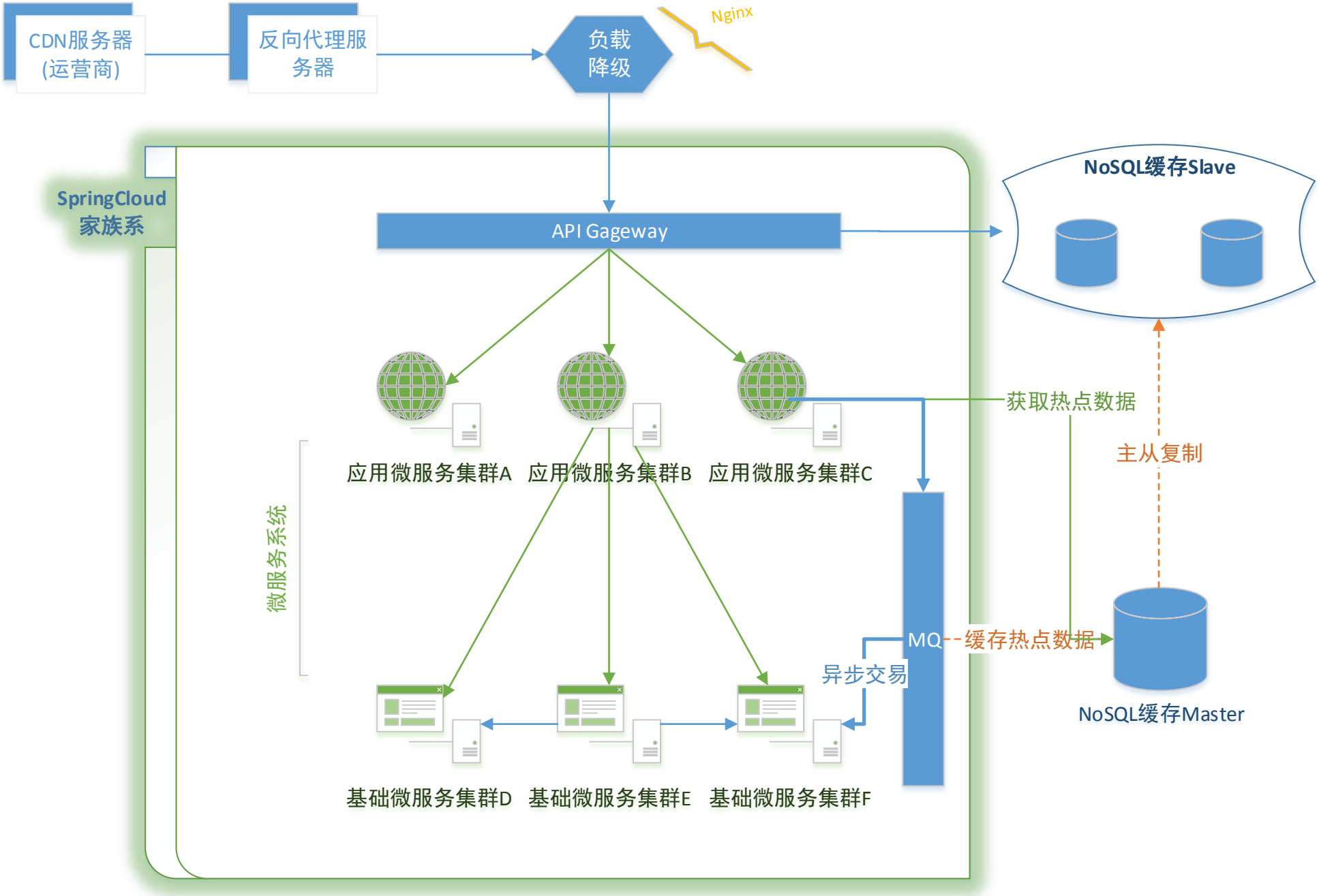
缺点：业务臃肿、事物强一致性、同步交易性能低下

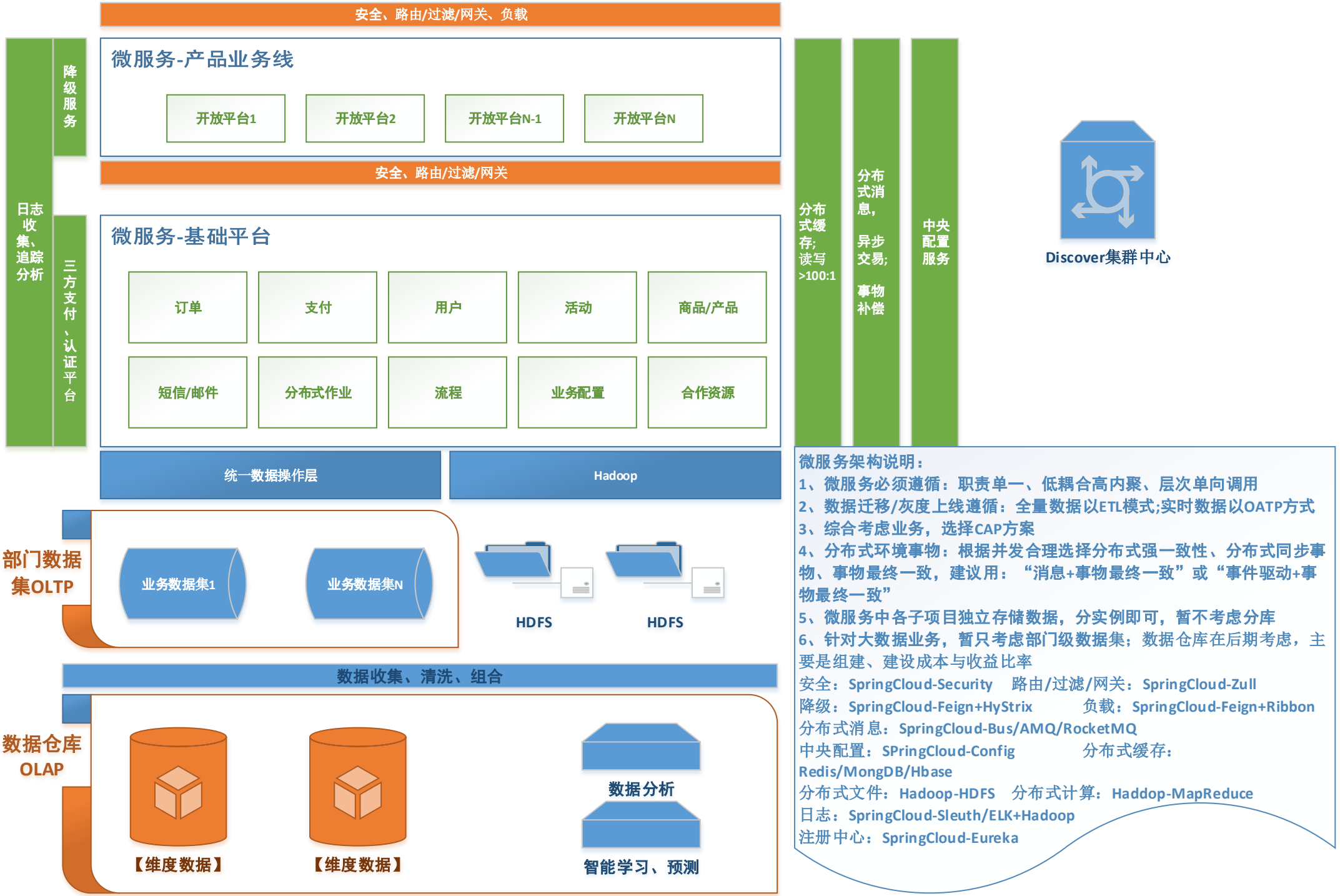


网站架构演进：业务拆分、消息异步

1、按战略策略进行业务纵向拆分、系统横向拆分（业务功能单一化，提升系统性能）
2、业务与业务系统通讯异步化（能延迟做的事情都可以异步，支持高并发、稳定性）
3、事物最终一致性（把传统事物强一致性转化为事物最终一致性，注意：按业务场景考虑）

缺点：性能、服务端压力问题


CDN服务器
(运营商)
反向代理服务器
负载
降级
Nginx
SpringCloud
家族系
API Gateway
NoSQL缓存Slave
应用微服务集群A
应用微服务集群B
应用微服务集群C
获取热点数据
主从复制
微服务系统
MQ
缓存热点数据
异步交易
NoSQL缓存Master
基础微服务集群D
基础微服务集群E
基础微服务集群F


安全、路由/过滤/网关、负载
日志收集、追踪分析
降级服务
三方支付、认证平台
微服务-产品业务线
开放平台1
开放平台2
开放平台N-1
开放平台N
安全、路由/过滤/网关
微服务-基础平台
订单
支付
用户
活动
商品/产品
短信/邮件
分布式作业
流程
业务配置
合作资源
统一数据操作层
Hadoop
分布式缓存;读写>100:1
分布式消息,异步交易;事物补偿
中央配置服务
Discover集群中心
部门数据集OLTP
业务数据集1
业务数据集N
HDFS
HDFS
数据收集、清洗、组合
数据仓库OLAP
【维度数据】
【维度数据】
数据分析
智能学习、预测
微服务架构说明：
1、微服务必须遵循：职责单一、低耦合高内聚、层次单向调用
2、数据迁移/灰度上线遵循：全量数据以ETL模式;实时数据以OATP方式
3、综合考虑业务，选择CAP方案
4、分布式环境事物：根据并发合理选择分布式强一致性、分布式同步事物、事物最终一致，建议用：“消息+事物最终一致”或“事件驱动+事物最终一致”
5、微服务中各子项目独立存储数据，分实例即可，暂不考虑分库
6、针对大数据业务，暂只考虑部门级数据集；数据仓库在后期考虑，主要是组建、建设成本与收益比率
安全：SpringCloud-Security 路由/过滤/网关：SpringCloud-Zull
降级：SpringCloud-Feign+HyStrix 负载：SpringCloud-Feign+Ribbon
分布式消息：SpringCloud-Bus/AMQ/RocketMQ
中央配置：SPringCloud-Config 分布式缓存：Redis/MongDB/Hbase
分布式文件：Hadoop-HDFS 分布式计算：Haddop-MapReduce
日志：SpringCloud-Sleuth/ELK+Hadoop
注册中心：SpringCloud-Eureka

架构、技术选型原则:

1、战略方向决策业务

2、业务决定架构

3、架构选型技术

架构是演变而来的，不是精心设计的，不能盲目模仿/复制大公司企业网架构。

架构面临的挑战：

1、性能

2、可用性

3、扩展性

4、伸缩性

5、安全

微服务架构特性：

1、微服务不是目的

2、业务纵向、系统横向拆分

3、敏捷开发、部署

4、高内聚、低耦合、Rest风格

5、系统维度复杂化、运维成本大（需考虑服务治理，利用成熟中间件实现自动化部署、测试、流量切换、服务管控，资源自动降级、扩容）

6、分布式事物处理难（根据业务，合理选择事物方式）

7、数据库拆分难（按业务合理拆分，热点数据缓存和本地存储选取）

架构设计原则：

1、各层级限流机制，热点数据直接返回，减少核心服务压力

2、热点数据缓存/本地存储，[事件/队列+MQ|复制]实现数据同步

3、公共基础服务功能单一、服务幂等、本地事物化

4、交易异步化，提升并发、性能

5、直线微服务层级调用不易超过3层，超过3层考虑拆分

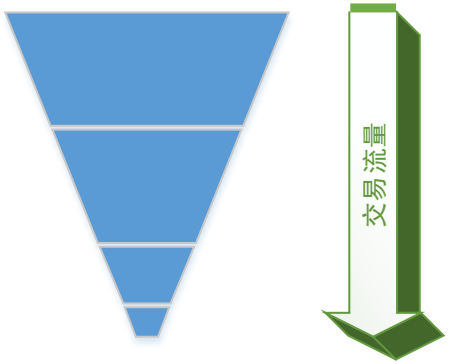
6、后端微服务不直接开放给调用端，需通过API Gateway实现交易聚合、隔离等机制

7、同一实例、数据库，尽量只被1个服务或中间件使用，减轻数据库服务资源及压力

8、数据库读写分离，提升读写速度

9、采用哈希一致算法实现分布式缓存集群，减轻缓存失效带来的雪崩效应

10、分布式事物拆分为本地事物，合理选择CAP（可用、一致、分区容忍）



1、书籍推荐

《软件设计精要与模式》

《恰如其分的软件架构》

《重构：改善既有代码的设计》

《代码整洁之道》

《重构与模式》

《设计模式》GOF

《TestDriven Development》测试驱动开发