个人信息

9 年工作经验 |本科 |32 | 男
微信号:
邮箱:

自我评价

近 9 年 Java 开发经验，其中 **3 年以上大型系统架构设计经验**，以及 **2 年以上的团队管理经验**，具有**分布式、高并发、高可用、大数据量的系统架构设计以及研发经验**，目前正**负责注册用户三千万，日活三百多万，日访问量 1 亿+，高峰期十万并发**的社区电商平台的架构设计与研发。同时拥有扎实的技术功底，**对 Zookeeper、Netty、Dubbo、Spring Cloud.Spring 等开源框架源码有过深入研究，并且有一定的框架定制开发经验**

教育经历

毕业学校:XXX 学历 ： XX

工作经历

2019.3-至今 XXX

2016.8-2019.03 XXX

2013.7-2016.07 XXX

个人技能

1、精通 Java、设计模式、网络编程(Netty)、并发编程、JM 内存管理及调优
2、精通 spring、springMVC、Mybatis，阅读过相关源码并根据需要扩展
3、精通 dubbo、spring cloud(Eureka、Ribbon、Feign、Hystrix、zuul)，阅读过相关源码
4、精通分布式事务，阅读过 2PC、TCC 相关组件的源码，设计可靠消息最终一致性方案最大努力通知方案、saga
5、精通 Mysql，具有 sql 优化、Mycat 分库分表、索引优化、性能调优、数据库灾备等丰富的实战经验
6、精通 Redis，具有集群搭建(Twemproxy、Codis、Redis Cluster)，冷热备份，性能调优、数据迁移等实战经验

7、消息中间件:掌握 rocketMQ 原理及集群布署
8、负载均衡:熟练使用 Nginx(Tengine、Openresty)、zookeeper 等负载均衡组件
9、自动化布署:Git、Jenkins.Gradle
10、容器化布署:docker，具有搭建 swarm、mesos、marathon、kubernetes 集群并运维经验
11、自动化运维:熟练使用 Saltstack，数据监控(zabbix)
12、建模工具:PowerDesigner、Rose、Visio、等 UML 建模工具
13、遵循华为测试规范，功能测试(单元测试、冒烟测试、集成测试、QA 测试)性能测试(Jmeter、LoadRunner)、自动化测试(selenium、QTP)

项目经验

项目名称:商品线架构拆分

项目描述:目前商品线运营后台功能还在基础工程中，而且商品线只有一工程，包含了所有功能，需要将同步 redis 定时任务和小程序接口进行工程拆分，同时在小程序接口工程中增加本地缓存逻辑

项目意义:商品线按运营后台，同步任务，小程序接口三个维度进行拆分工程，职责分明解耦，提高并发性能

项目职责:负责将活动管理模块拆分;负责同步事件的写入;负责品牌馆模块拆分和本地内存设计和实现

项目名称:商品资料专项

项目描述:现有商品模型没有统一，不具备扩展性和维护性，严重妨碍了物流侧的优化工作。需要将商品线业务进行重新梳理，建立完善可扩展性强的模型结构，重新定义表结构，整合业务，提高运行效率，降低业务和代码的复杂度。

项目意义:本标志着公司电商端、物流端商品基础资料库的统一，提高了公司商品编辑部门和物流分拣的工作效率:也为一品多规，统一库存单位打下了坚实的基础，也进一步为电商平台的人、货、场信息贯彻提供了数据基础，是意义非凡的项目成功

架构经验

(1)　分布式系统架构

1、分布式系统基于 dubbo 划分为商品线、交易线、资金线、用户线、支付线、履约线构成的电商系统架构，dubbo 配置关闭启动时检查、服务分组、多版本、延迟暴露等。基于 dubbo 如何做服务治理、服务降级以及重试
2、配置中心使用携程 Appollo 框架，注册中心采用 ZK 实现。
3、基于分布式数据库 MongoDB 解决消息中心、广告中心的大批量终端 ld 的信息更新 4、基于分布式搜索系统 ElasticSearch+Kafka 自研的 Trace 进行链路追踪和日志统计 5、分布式事务主要应用于交易中心的支付功能采用 TCC 事务，保障支付，风控，优惠券处理的一致性;支付网关回调采用可靠消息最终一致性设计，保证扣款，积分，抽奖的最终一致性;支付反馈消息采用最大努力通知型设计
6、分布式 session 使用 Tomcat-redis-session-manaqer 实现共享 7、采用 Redisson 实现的分布式锁方案解决分布式并发需要加锁的场景

(2) 高并发系统架构

1、SLB+Tengine 分散压力，优化并发连接数
2、Tomcat 采用非阻塞协议 Http11Nio2Protocol，启用压缩，优化并发连接数，JM 优化
3、用户中心、交易中心 采用 Mysql+ShardingJdbc+SLB 进行分库分表，读写分离
4、会员端小程序接口采用本地内存+Redis 多级缓存保存热点数据
5、个人中心、交易中心采用 RocketMQ 对低耦合业务进行异步处理
6、对热点数据预处理

(3) 高可用架构

1、各业务模块多点布署保证高可用
2、会员端小程序接口采用 Sentinal 对异常流量进行熔断降级和监控
3、用户信息、渠道信息等采用 Redis+Ehcache 多级缓存
4、MHA+Mysql+ShardingJdbc+SLB，应用分布式集群部署，Redis Cluster 集群保证存储高可用
5、RocketMg 高可用架构部署保证消息队列高可用
6、ElasticSearch+Zookeeper 集群保证搜索引擎高可用
7、Zabbix 自动化检测实时报警
8、核心接口异常和数据对比异常，发短信及时报警通知

(4) 高性能架构

1、采用构造器模式构建活动商品，采用状态模式进行商品的上下架，采用原型模式构商品 VO 转换

2、采用分布式定时任务轮巡保证数据强一致性
3、采用职责单一，层极分明，高聚合低耦合的原则设计代码结构
4、基于 AOP 机制进行异常的捕获，通过错误码的方式返回，便于快速定位解决问题
5、基于 Appollo 配置做新旧代码开关控制，线上发布无需回滚，直接更改配置即可

问题解决经验

解决会员端接口请求缓慢，带宽占比高，代码维护性差的问题
1、工程工程业务功能拆分，分别集群部署，提高单机的性能
2、建立本地缓存，所有会员端接口走本地内存方式
3、精简接口返回字段，不需要的字段不返回
4、重构代码，实现高聚合低耦合的原则

解决 Mysql 主库 CPU 使用率达到 46%
1、对历史数据进行归档
2、对查询频率最高前十个 sql 进行优化。
3、对大字段进行拆表处理
4、读写进行分离，读操作走从库，写走主库
5、经过优化，主库 CPU 使用率降低到 8%

优化慢查询 sql,解决 MySQL 读库 CPU 使用率 100%停止服务问题 1、优化导出实现方式，限制导出最大数目，限制最大导入数目
2、优化时间段过滤查询方式，走更好的索引
3、不写联合查询表的 sql,进来单表查询，逻辑处理在内存中进行处理
4、重构代码，减少不必要的查询

公共 Redis 迁移，解决商品线共用公共 redis 问题
1，将公共 Redis 中商品线所有用到的 Key 全部迁移到新的独立集群
2，使用阿里云 redis-shake 工具进行迁移
3，迁移后，需要对当天和明天的数据进行全量刷新缓存