自我介绍

面试官你好，我叫周日良、本科毕业于郑州航空工业管理学院。最近一份工作是我现在就职的菜鸟网络技术有限公司。

在菜鸟网络我先后主导了菜鸟客服系统、申通协同平台、菜鸟资源中心的建设。积累了物流供应链电商相关的经验。

在这期间我从0到1建设过多个系统，在分布式、大数据、高并 发方面积累的较多开发和系统架构经验。

在此期间，我还多次担任过团队的双十一稳定性PM，以及SRE，对稳定性建设以及线上问题处理流程和应急机制有较多的实战经验。

谢谢，面试官，我介绍完了。

资源中心架构升级项目

1、如何实现大促200w+qps流量

首先采用了三层缓存架构，富客户端本地缓存、富客户端tair缓存、服务端缓存

富客户端本地缓存，是第一层缓存，用于缓存热点数据，可以抵挡90几以上的请求。调用方通过依赖资源中心的jar包，在调用方本地构建的缓存，可以通过配置修改管理超时时间、缓存数量。这里的缓存时间会设置的很小，一般是30s 2000个。

富客户端tair缓存，是第二层缓存，可以抵挡99.4以上的请求。调用方不经过服务端直接调用缓存服务器获取结果。 这一层用的是阿里mdb缓存，底层其实是memcache，用的双机房独立集群(每个机房都是master)，进行容灾。

服务端缓存，是第三层缓存，是持久化缓存。用的是阿里的ldb缓存，底层用的是leveldb，用的双机房主备集群（一个机房master，一个机房slave），用于机房容灾。用于抵抗一些未知的未接入到富客户端的请求。

其次在大促期间会对资源进行封网禁写，同时全量缓存数据

为什么使用leveldb不用redis

主要是3点leveldb有更高的持久化性能、只需要kv存储，不需要多样的数据结构，leveldb相对redis全缓存更便宜

2、本地缓存用的是什么技术

ConcurrentHashMap实现

3、租户功能

1、获取缓存、设置缓存加上对应的租户标识前缀

2、mysql切面拦截新增租户条件，查询对应租户的数据。 具体为TDDL实现sqlmonitor，ibatis实现 inteceptor

3、重路由 用于日常环境 解决没有单元环境的问题。

租户标如何获取

1、通过starter包的自动配置实例化config信息

2、TenantContext静态变量加载租户识别类缓存到mapCached

3、CompatibleTenantContext(继承了TenantContext)获取租户标，根据config的租户标识从mapCached中获取到对应的租户识别类，获取对应的租户标

4、获取缓存时会通过CompatibleTenantContext获取TenantId，拼装缓存key值，获取对应租户的缓存数据。

4、租户之间数据同步

1、通过精卫接收数据变更消息，调用其他使用租户(useTenantId)的同步服务接口 （修改服务调用的clusterServer）

2、发送其他租户所在单元的消息

5、不停机迁移怎么做的（数据+接口迁移）

同步双写 （资源变更消息发送只能发送一份）

1、离线全量+增量比对 经过一段时间的比对。

2、在线实时比对 (考虑性能) （降噪比如id）

3、根据接口特征开发灰度开关(用户id，资源编码，资源id) 在灰度数据上选择对业务小，调用量小的数据线进行灰度，灰度的比例慢慢扩大。

测试(功能测试+预发引流) -> 比对稳定 -> 测试数据灰度 -> 正式灰度(比例逐渐放大) -> 全量灰度

特殊情况：

批量接口一般是管理态页面使用，业务影响小，实时比对耗时太大，全量测试后，一刀切

6、难点：

难点1：【高并发】需要设计承接双十一200+QPS

三层缓存架构

难点2：【模型多，不兼容】模型升级设计到的模型多，有30多中资源类型。特别是模型不兼容问题。老数据中仓资源模型有版本概念、新模型里面没有版本概念。

共性资源用新模型主字段承接，比如资源所属CP编码、状态、名称、类型等，个性化信息由扩展大字段承接，对于有查询需求的建立二级索引表，管理页面使用宽表。

仓类型资源数据迁移

针对写，因为编辑页是统一管理页面，逻辑都在自己应用，去掉多版本相关逻辑，这块相对可控。

针对读，需要对依赖进行分析改造

1、对有依赖版本信息的接口进行分析。因为接口依赖方较多，通过开发比对工具，比对老接口跟新接口的返回结果，有差异的记录调用方记录，并分析改造。(比对逻辑异步跑)

2、对直接数据表依赖应用进行分析，有影响的进行改造并验证

3、对仓资源变更消息进行分析改造，去除版本号依赖逻辑，并验证

难点3：【不停机切换】数据源切换、接口切换要求不能停机

写数据迁移

1、开发ETL逻辑，从老库迁移部分数据到新库。通过读接口做数据正确性验证（基本测试，引流比对，实时比对）

2、开启双写逻辑（先写新库再写老库，失败回滚）

3、开启存量数据的迁移，迁移到开启双写时间之后的一个时间(10s buffer)

如果碰到数据不准确问题，关闭读接口，删除脏数据，修改验证后重新执行数据迁移

难点4：【接口多，依赖方多】涉及71个接口的切换，200多个依赖方

针对不同的接口采用不同的灰度切流策略。

\* 根据业务维度灰度切流。比如根据userId，资源编码，资源类型进行灰度

\* 按业务影响程度从小到大切流

\* 按流量比率从小到大切流

\* 根据依赖方切流，从不重要的到重要的切

难点5：dump数据深分页问题

滚动查询

7、国际化开发经验： 一套代码多地部署+国际化开发经验 TODO

1、多租户框架 spring的autoconfigration condition初始化不同环境的变量值

2、插件化 支持不同中间件

3、数据库utf8mb4 代码编码utf8 时间Locale 规则校验，电话号码校验， 地址库

8、业务

虚仓

虚仓的存在是为了让货品更加高效、灵活的进行售卖，并且进行商品在各个仓库之间的调拨

在途虚仓：解决售卖在途货品库存问题

预售虚仓：在没有实仓库存的情况下，要通过负卖业务进行销售，小二需要发布负卖虚仓，负卖虚仓会继承其对应的唯一的菜鸟实仓的覆盖范围，但发货优先级低于菜鸟实仓。

JVM内存优化

我有个资源的老应用在每天凌晨0-4点之间会接收多个燎原包来构建本地缓存。这个燎原包，可以简单理解为一个数据文件，每天凌晨这段时间生产端的应用会基于最新数据重新生成文件，然后通知给消费应用拉取，拉取到本地磁盘后生成为新的缓存对象替换老的缓存。

应用在替换缓存期间会发生fullgc，时间会达到1s多，碰到流量大的时候可能导致服务超时。

机器用的是jdk1.8 CMS垃圾收集器，4C16G XMS XMX 12G yonggc也很长

内存笔记大所以考虑先降低JVM内存大小，所以做了两个动作

1、把最大内存调整为了10G，

2、每次加载一份缓存后主动调用System.gc尽量触发JVM进行gc 以减少gc时间 但是实际效果一般

然后又把触发fullgc垃圾回收的阈值调低

3、降低触发fullgc阈值，CMSInitiatingOccupancyFraction从80%降到70%，效果也一般

然后考虑升级到G1试试，因为G1在大内存垃圾回收方面能够比较好的控制时间，正好公司今年也开始推送jdk1.8到11的升级

所以就

4、升级jdk8到jdk11，使用G1垃圾收集器 XMX设置为10G 最大GC暂定时间设置为 MaxGCPauseMillis=200ms fullgc基本没有了

但是yong gc时间稍微有点长 考虑缓存会加载很多大对象 加快回收速度

5、调整堆区域大小(G1HeapRegionSize)设置成16M默认是1M yong gc时间提升3倍多

解释：G1增量回收，控制停顿时间，对对象回收优先级进行标记，优先回收

协同平台项目

申通客服工单项目

项目简介：

原申通客服系统不能很好的满足当下快速业务发展，同时在技术栈转向java的大背景下需要替换掉原.net系统，亟需新建一个能快速接入业务，适应灵活变动的业务的新客服系统。

模块包括工单处理、查询&搜索、流程绘制、接入配置、报表等。

工作内容

1、负责客服系统的建模，以及整体技术方案；引入流程引擎、搜索、分库分表等技术栈解决面临的场景问题。

2、指导产品同学进行老业务流程的梳理，新系统的宣导、培训。

3、给技术同学进行技术指导，难点攻关等。带领团队同学完成项目开发

业务场景：

虚假签收 已派未签 超时未揽收 有称无重 信息分

任务网点升级总部 按照组织分库分表 查询就支持不了

指标：出港及时率、出港量 本周不达标天数

管理页面： ads

实操页面: drds

20年3月开始做taskcenter应用的分离

icp\_task\_config icp\_role\_config icp\_channel\_config(消息通道)

20年8月开始工单客服项目 接入了审批工单、备案

同步opensearch

int 会给默认值0 改成 LITERAL

opensearch这边有个坑，如果字段设计成int的话，那么如果新增没有设置值的时候，会默认给个0

LITERAL的话就不会

难点：opensearch同步问题

背景：业务上需要根据任务(task)维度，表单(form)维度进行多维查询，查询的信息回同时涉及到任务表、表单表两个表。

方案是构建两张搜索(opensearch)表， 然后这个过程碰到比较多的问题。

一开始想通过异步消费binlog日志变化来更新搜索数据。使用了dts(canal)中间件，但是发现搜索中的数据跟数据库中的数据不同步。

排查最后发现dts(canal) 一个物理库对应一个Dstore，消费时不能保证顺序，在高并发情况下会导致任务(task)更新的最新数据被form覆盖。

然后就改成mybatis intercepter 拦截后发送局部有序的rocketmq，接收到消息后查询最新数据去更新opensearch。

mq集群故障，发送RT超时，导致数据库连接池被耗尽从而导致服务超时 改成了异步，重试3次，异步失败告警，人工介入处理。

获取连接 -> 执行intercepter(包含了mq发送) ->释放连接 mq发送3s超时导致连接快速耗尽

大电&家装自营项目 21年1-4月

项目介绍：从0-1打造菜鸟大电大件自营仓配网络体系，提升服务体验，形成有竞对优势。系统上沉淀以配驱仓模式。完成仓配，纯配正逆向链路建设，提供仓&配预约、拦截、上门取件等能力。

工作内容

1、作为OMS域技术PM完成总体技术方案编写，同时协同其他域一起保障项目交付

2、逆向链路的开发工作

大电：仓配， 家装：仓配、纯配

模式差异：

仓配一体 -》仓配解耦

CP分单 -》 菜鸟分单

CP系统 -》 菜鸟系统（以配驱仓）

CP管控的车队 -》 菜鸟管控到车辆

弱控 -》 强控

仓作业：

仓接单 -> 预处理-》 仓汇单 -》拣选 -》集货 -》 打包&复核 -》 仓配交接

预处理：切分成包裹，计算体积，需要的包材，业务规则

汇单：占库存、生成拣选单、取面单

改约、增值能力

hold、制定旅行计划、合单、延误重算

cnfcp: 相当于协调者，协同仓配等

cnfdp: 相当于大脑

物流链路特点：物流的链路非常的长，只要其中一部失败都会导致卡单。 相对电商前台断链要求的成功率不同，小概率失败可以通过界面直接重试。

合理切分流程，失败重试，异步执行；卡单监控

决策hold逻辑 决策 hodedecision

下发仓hold单根据 决策返回的计划时间

仓内包裹预处理： 拆分包裹、获取电子面单

cnfcp 履行平台

通过识别业务身份、订单类型匹配到流程，串联能力，差异点用扩展点满足

核心设计，如何支撑不同行业、不同业务、不同模式的物流履约？

履行协同平台抽象了四层抽象：

身份抽象：可隔离，提供差异隔离的基础(个性化的基础)

流程抽象：可复用，快速搭建新的业务流程

能力抽象：可复用，封网各域业务特性

扩展抽象：可隔离，差异隔离的具体实现

不同的身份映射到解决方案(每个解决方案映射了一个流程)，流程串联了各个能力，能力执行

BefEngine(qlExpress) 通过单据信息得到 =》业务身份

业务身份 + 场景(delivery 发货) =》 业务流程

流程引擎回调Activity(TocOmgSendToStoreActivity toc发货) -> ability(tocOmgSendToStoreAbility)

-> 核心逻辑 + 扩展点逻辑(TocSendToStoreExtPoint)

nbp -> handler(sendStoreHandler) 驱动流程 ->

难度：

刚转岗到国内供应链团队，对国内供应链的架构、系统还不够熟悉；

项目设计的域较多，项目涉及OMS、TMS、WMS、裹配三个大域，我作为OMS域PM是链路中枢需要协调其他两个域；

同时涉及十几个系统的开发，技术栈复杂，我自己开发的一条链路上关流程组件就有3个，过度的异步消息，链路梳理费劲4次对象转换

问题：冷热OTS设计？

冷热数据同时查询场景

挑战性项目讲解，或者挑一个项目讲解：

资源中心技术架构升级

1、项目背景介绍 (比如资源中心架构升级项目)

2、项目难点列举

3、挑一个进行详细讲解，如何解决的（比如 数据、接口的不停机切换怎么做的）

同事冲突怎么解决

我觉得碰到冲突时候首先要让双方冷静下来，以开诚布公方式进行交流，把冲突的根因找到，探讨分析可行的解决方案，争取找到双方都可以接受的解决方案。如果还是无法解决，可以升级到部门老板进来一起协调解决。

我这里举一个例子：

在一次项目代码CR的时候，我对其中一块的代码提出可以使用策略模式以提高扩展性，但项目同学认为没有必要，这增加了代码复杂度。我仔细跟同事进行讨论，其实他是觉得一来修改会导致开发时间拉长，因为本来项目时间就紧张，二来代码维护成本会增加。我了解到之后，跟项目同学进行说明，这块业务目前是一期，很快后续会有新的业务进来，从长期看这部分的修改是很有必要的。最后他同意用设计模式，但是考虑到代码变动带来的回归测试，我同意在一些地方做了简化处理。最后问题得到解决。

另外一个例子：

在一个项目技术方案评审的时候，对于前后端交互的设计我跟一位同事产生了不同的意见。我的设计思路是，前端只做展示，不做涉及带有逻辑的功能。同事的意思是想尽量的复用后端接口，前端差异化的部分通过前端逻辑来处理，以减少后端的开发。我肯定了同事的想法是很好的，然后说明了前端同学流动性大，后端同学看不到业务全貌，后期的维护成本会很高。同事同意了我的观点。最后我们达成了一个折中的方案，对于逻辑较重的放在后端处理，对于一些简单的比如值转换等交给前端处理。

如何做团队管理

目标管理（团队目标+成员目标），团队建设，流程制定（开发流程、线上问题处理流程等）、团队氛围的营造、激励、风险管理

我觉得应该包含以下几个方面

1、目标的管理

团队目标的制定：

按一定周期制定团队中长期目标，要跟公司、老板的目标保持一致。并根据变化及时的调整，同时让每个团队同学清晰了解团队的目标

目标拆解&成员目标制定：

根据团队目标，以及成员分工情况对目标进行拆解，帮助每个成员制定目标，做到对自己的目标清晰

2、团队建设

根据团队情况，设计一个高效的团队结构。保证任务、责任和角色分配清晰，同时考虑一定的灵活性以应对变化

做好梯队建设，确定关键岗位，并做好备份。识别有潜力成员，进行针对性的培养

3、流程制定

制定优化开发管理流程，提高效率。为团队提供先进的工具和技术，简化日常工作。（比如需求开发流程、上线流程、问题排查流程等）

4、氛围营造

为团队提供一个包容、尊重和助人为乐、有技术氛围工作环境。 （比如团建、生日会、各类分享会、品码会）

5、激励

明确绩效目标，及时认可和奖励团队成员的表现和努力，提高团队士气

6、风险管理

及时识别项目风险，制定应对计划化解风险

具体的一些例子可以借助项目讲解：

模块化管理：对整个协同平台进行模块划分，根据功能和业务逻辑的不同将平台拆分成多个独立且可协作的模块，以确保专注性与高效率的开发工作。

任务分派：负责将不同的模块开发和维护任务分配给团队成员，确保每个成员明确自己的责任范围和达成目标，同时根据成员的技能和职业发展目标优化分配方案。

梯队培养：通过制定个性化的成长计划和提供专业培训，致力于培养团队内部的人才梯队。我注重为团队成员提供技术提升和职业发展的机会，帮助他们在技术深度和管理广度上实现成长。

这些工作不仅优化了项目管理流程，并且培育了高效率的团队协作氛围，同时也确立了团队成员之间的互帮互助和知识共享文化，有利于建设一个可持续发展且具备自我迭代能力的团队。

如何做项目管理

按项目前、中、后顺序讲

定义和规划项目：明确项目目标、范围、价值主张和预算。 包括确定项目的关键里程碑和交付物。

组织团队：组建项目团队(圈资源)，明确每个团队成员的角色和职责。

详细计划：制定和管理详细计划时间，确保各项任务按时完成。

风险管理：识别项目中潜在的风险，制定风险缓解和应对策略。

质量保障：确保项目成果符合预定的质量标准和用户需求。 做好CR， 测试用例评审、测试(功能测试、性能测试)

沟通管理：与团队成员、利益相关者以及客户保持持续的沟通，确保信息准确传递，并管理期望。

进度跟踪和报告：监测项目进展，向管理层和客户提供定期的状态更新和进度报告。 日报、周报、每日站会、周会

解决问题和决策：解决项目过程中出现的问题和挑战，作出关键决策，确保项目保持正确方向。

项目总结：分析项目完成后的反馈和数据，以逐步改善未来项目的管理流程。

碰到资源瓶颈问题怎么办

首先我会先向资源缺失方（比如相关开发主管、测试主管）说明项目情况，请求新增资源，或者协调更熟悉的开发同学进行调换。

如果不行，第二，会项目成员负荷安排申请加班是否能够解决，如果可以申请加班

如果不行，第三，对项目事项优先级进行排序，是否能够按照迭代方式发布。 需要对需求 举例 冷链多仓项目 导入功能、导出报表功能迭代二三发布

如果不行，第四，跟需求方、产品经理进行讨论，是否可以砍掉不重要的功能

如果不行，第五，继续升级问题到大老板进行决策。

如何做项目里程碑

MRD评审 - PRD评审 - 技术方案评审 - 测试评审 - 提交测试 - 演示 - 发布 - 实施 - 整体交付

架构师的职责是什么

我觉得架构师应该包括以下几个方面

1、架构设计: 领域划分、领域交互方式、部署模式

2、技术决策: 技术选型 框架、中间件、数据库选项等

3、风险管理：识别风险，并制定策略 自建 or 开源 or 采购

4、团队领导与指导： 提供技术指导，对技术人员进行培训提高他们技术能力

5、制定规范: 制定编码规范，符合团队的开发规约

要做好以上方面我觉得架构师需要具备以下能力

1、要深入理解业务，比如业务逻辑、业务规模、业务发展 才能设计出符合业务，以及具有前瞻性的架构

2、对各类框架、中间件等熟悉其使用和原理 以及应用场景

3、有较好的协同沟通能力，使得各个领域能够配合进行按照规划架构落地、升级优化

4、扎实的编码能力，能够指导一线技术同学实施架构

秒杀设计

https://ata.atatech.org/articles/11000253590?spm=ata.23639746.0.0.3549706bq14ld6#YJhUh7ON

业务特点：限时、限价、限量 活动预热 售卖持续时间短

技术特点：瞬时流量突增 读多写少 数据一致性要求高

架构原则：

1、跟普通应用跟里

2、尽量将请求拦截在系统上游

3、秒杀详情页动静分离

flink 精确一致通过

1、checkpoint barrier

2、两阶段提交

https://blog.csdn.net/u013411339/article/details/101730043

flink面试题

https://blog.csdn.net/weixin\_40035038/article/details/135521590