JD.COM 京东
多 · 快 · 好 · 省

京东应用架构设计

吴博

目录 CONTENTS

1 架构愿景

2 业务架构

3 应用架构

4 数据架构

5 技术架构

6 618经验

架构目标

4. 多快好省

构建超大型电商交易平台，兼顾效率和性能，达到高人效、高时效和低成本的目标

3. 低成本

增加服务的重用性，提高开发效率，降低人力成本；利用成熟开源技术，降低软硬件成本；利用虚拟化技术，减少服务器成本

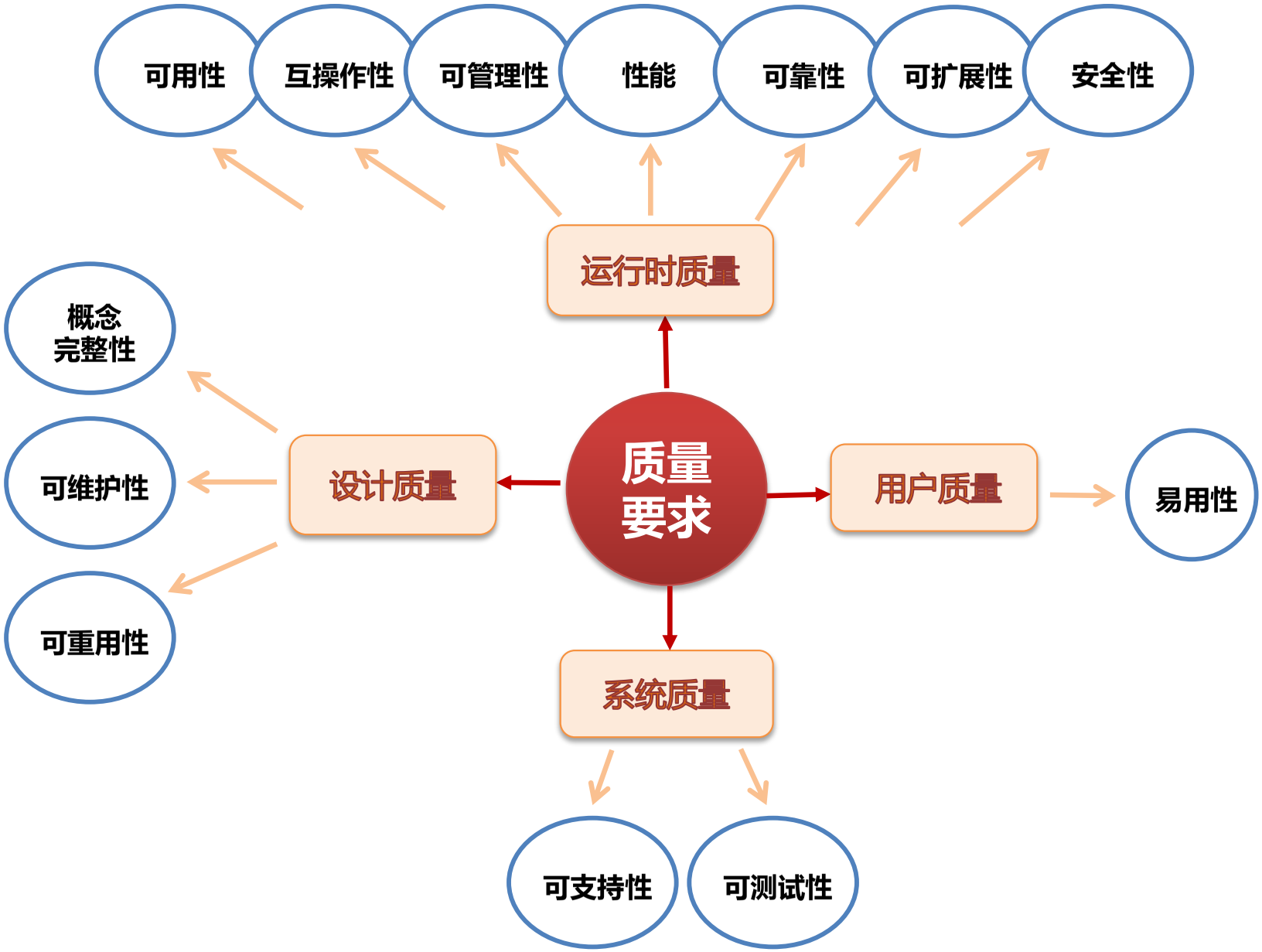
2. 高可扩展性

系统架构简单清晰，应用系统间耦合低，容易水平扩展，业务功能增改方便快捷

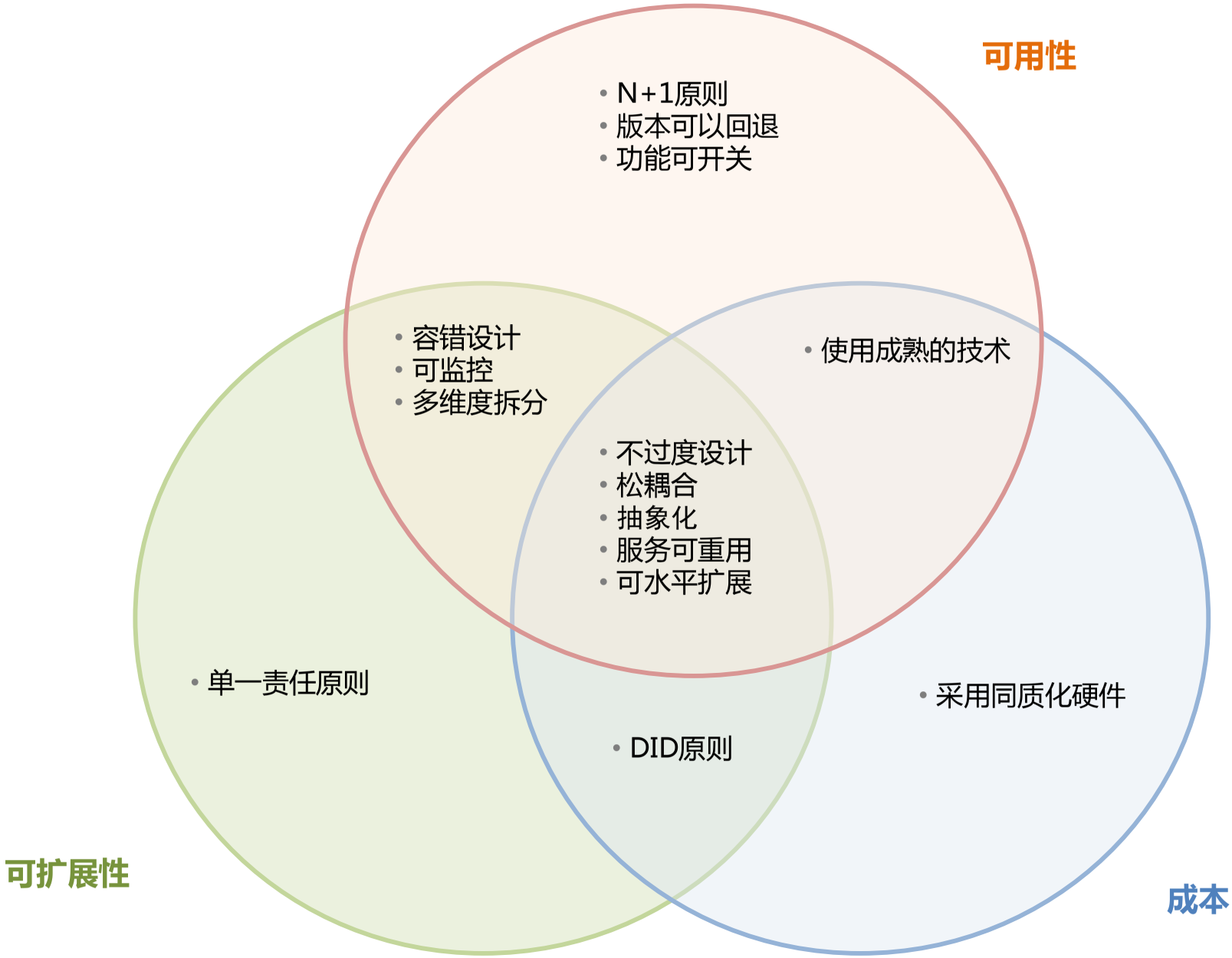
1. 高可用性

自动化运维。整体系统可用性99.99%，单个系统可用性99.999%。全年故障时间整个系统不超过50分钟，单个系统故障不超过5分钟

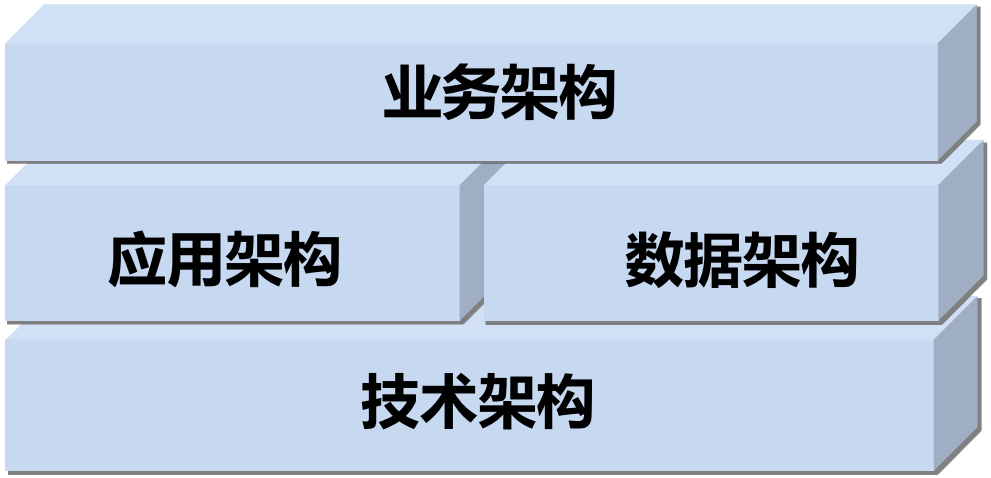
质量要求

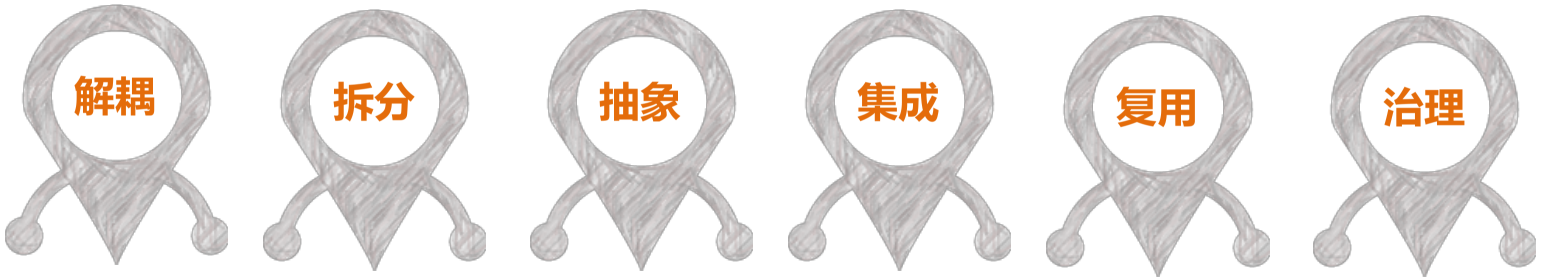

可用性
互操作性
可管理性
性能
可靠性
可扩展性
安全性
运行时质量
概念完整性
可维护性
可重用性
设计质量
质量要求
用户质量
易用性
系统质量
可支持性
可测试性

总体架构原则


可用性
• N+1原则
• 版本可以回退
• 功能可开关
• 容错设计
• 可监控
• 多维度拆分
• 使用成熟的技术
• 不过度设计
• 松耦合
• 抽象化
• 服务可重用
• 可水平扩展
• 单一责任原则
• 采用同质化硬件
• DID原则
可扩展性
成本

架构组成和关键点





目录 CONTENTS

1 架构愿景

2 **业务架构**

3 应用架构

4 数据架构

5 技术架构

6 618经验

业务架构设计原则

1. 业务平台化

- 业务平台化，相互独立。 如交易平台、仓储平台、物流平台、支付平台、广告平台等
- 基础业务下沉，可复用。如用户、商品、类目、促销、时效等

2. 核心业务、非核心业务分离

- 电商核心业务与非核心业务分离，核心业务精简（利于稳定），非核心业务多样化。如，主交易服务、通用交易服务

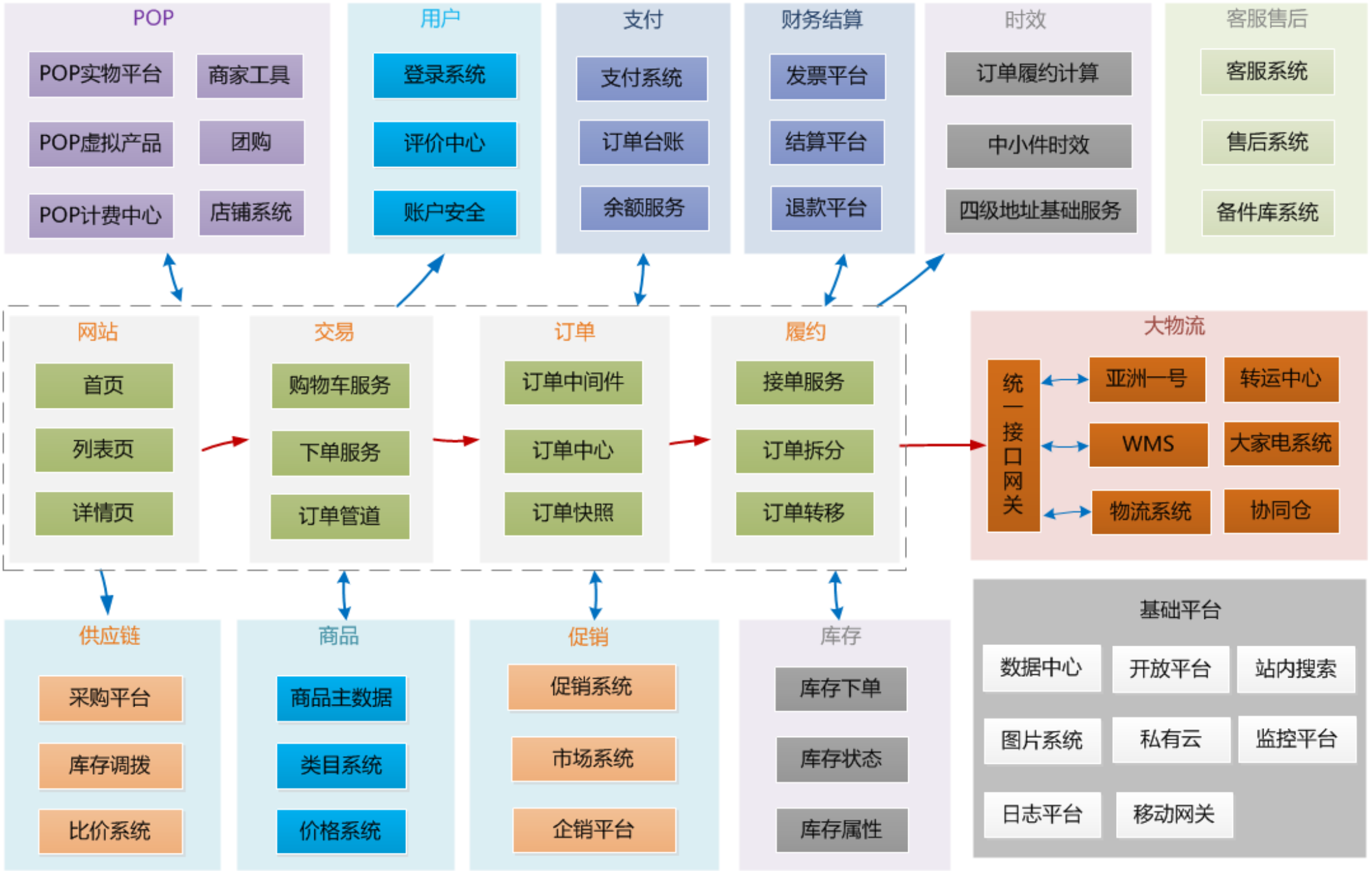
4. 区分主流程、辅流程

- 分清哪些是电商的主流程。运行时，优先保证主流程的顺利完成，辅流程可以采用后台异步的方式。避免辅流程的失败导致主流程的回滚。如，下单时，同步调用快照，异步通知台账、发票

3. 隔离不同类型的业务

- 交易业务是签订买家和卖家之间的交易合同，需要优先保证高可用性，让用户能快速下单
- 履约业务对可用性没有太高要求，可以优先保证一致性
- 闪购业务对高并发要求很高，应该跟普通业务隔离

业务架构图


POP
POP实物平台
商家工具
POP虚拟产品
团购
POP计费中心
店铺系统
用户
登录系统
评价中心
账户安全
支付
支付系统
订单台账
余额服务
财务结算
发票平台
结算平台
退款平台
时效
订单履约计算
中小件时效
四级地址基础服务
客服售后
客服系统
售后系统
备件库系统
网站
首页
列表页
详情页
交易
购物车服务
下单服务
订单管道
订单
订单中间件
订单中心
订单快照
履约
接单服务
订单拆分
订单转移
大物流
统一接口网关
亚洲一号
转运中心
WMS
大家电系统
物流系统
协同仓
供应链
采购平台
库存调拨
比价系统
商品
商品主数据
类目系统
价格系统
促销
促销系统
市场系统
企销平台
库存
库存下单
库存状态
库存属性
基础平台
数据中心
开放平台
站内搜索
图片系统
私有云
监控平台
日志平台
移动网关

业务架构实例：基础业务下沉

目录 CONTENTS

1 架构愿景

2 业务架构

3 **应用架构**

4 数据架构

5 技术架构

6 618经验

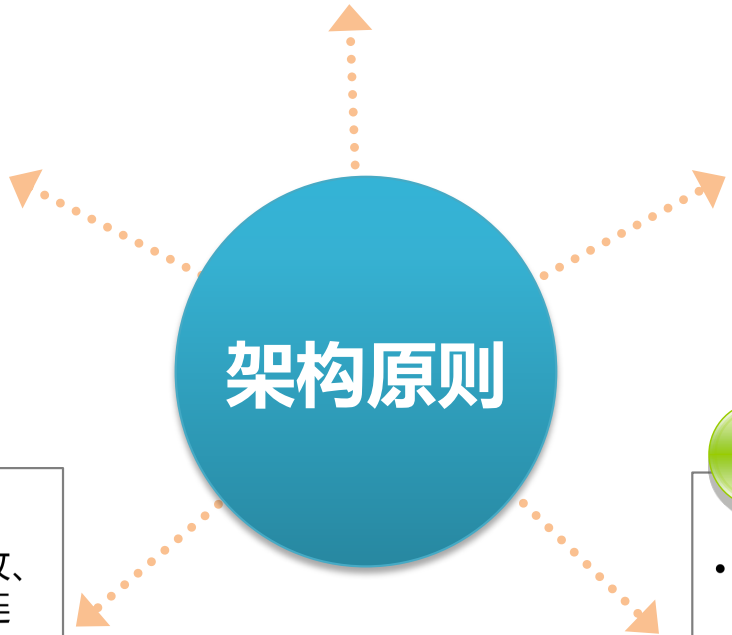
应用架构设计原则

1 稳定性原则

- 一切以稳定为中心
- 架构尽可能简单、清晰
- 不过度设计

2 解耦/拆分

- 稳定部分与易变部分分离
- 核心业务与非核心业务分离
- 电商主流程与辅流程分离
- 应用与数据分离
- 服务与实现细节分离



3 抽象化

- 应用抽象化：应用只依赖服务抽象，不依赖服务实现细节、位置
- 数据库抽象化：应用只依赖逻辑数据库，不需要关心物理库的位置和分片
- 服务器抽象化：应用虚拟化部署，不需要关心实体机配置，动态调配资源

4 松耦合

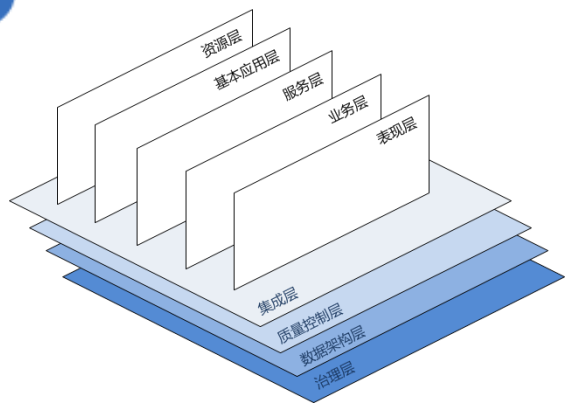
- 跨域调用异步化：不同业务域之间尽量异步解耦。
- 非核心业务尽量异步化：核心、非核心业务之间，尽量异步解耦
- 必须同步调用时，需要设置超时时间和任务队列长度

5 容错设计

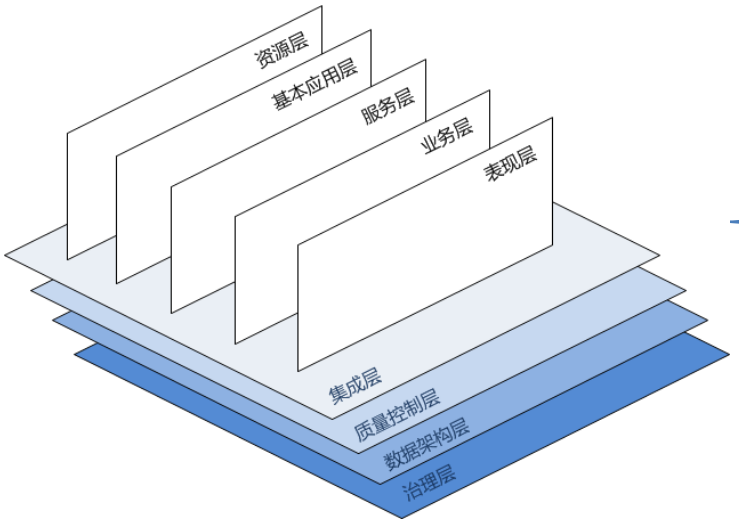
- 服务自治：服务能彼此独立修改、部署、发布和管理。避免引发连锁反应
- 集群容错：应用系统集群，避免单点
- 多机房容灾：多机房部署，多活

京东应用架构

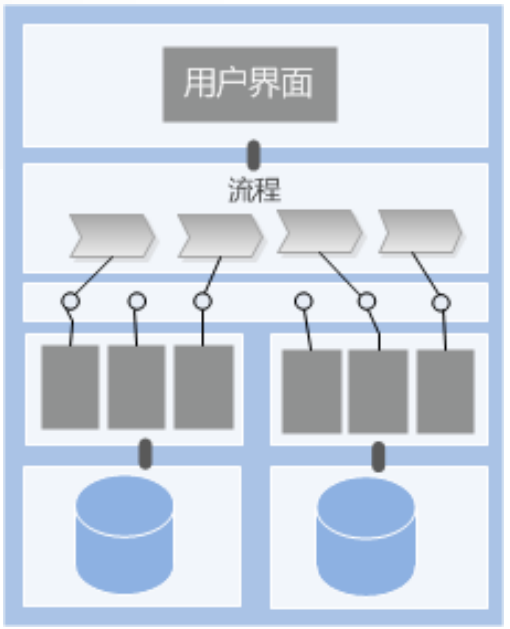

表现层
首页
列表页
详情页
业务流程层
商品系统
交易系统
订单系统
财务系统
物流系统
服务层
服务构件层
商品服务
交易服务
订单服务
财务服务
物流服务
资源层
集成层
服务质量层
数据架构层
治理层


资源层
基本应用层
服务层
业务层
表现层
集成层
质量控制层
数据架构层
治理层

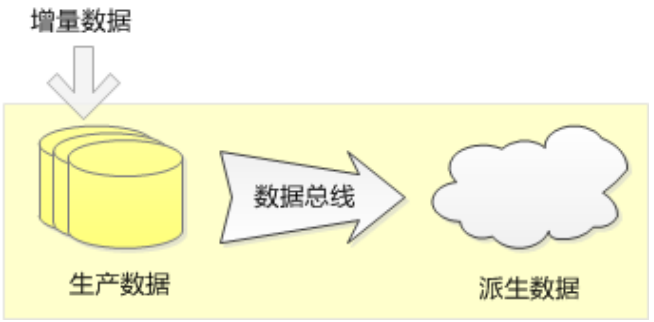
架构分析



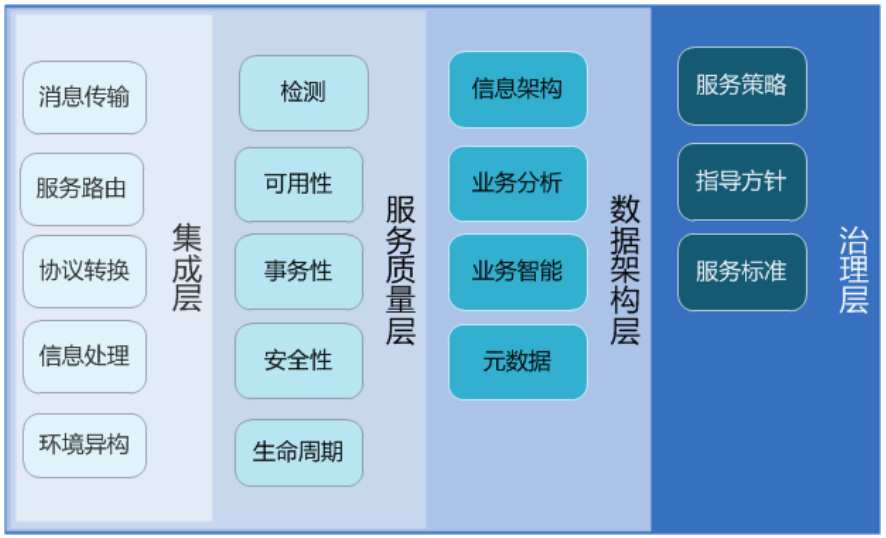
应用架构



数据架构



基础架构



架构分解原则

	应用系统	数据库	
1. 水平扩展（复制）	多机集群，提高并发能力	读写分离 如，商品读库，商品写库	高并发
2. 垂直拆分（不同业务拆分）	按业务域划分系统 如，商品系统 交易系统	按业务分库 如，商品库，订单库	
3. 业务分片（同业务分片）	按功能特点分开部署 如，秒杀系统	分库分表，提高数据容量 如，订单库按ID分库分表	大数据
4. 水平拆分（稳定与易变分离）	服务分层 功能与非功能分开	冷热数据分离，历史数据分离	

依赖原则

2. 跨域弱依赖

- 跨业务域调用时，尽可能异步弱依赖

1. 依赖稳定部分

- 稳定部分不依赖易变部分
- 易变部分可以依赖稳定部分
- 要求：避免循环依赖

3. 基本服务依赖

- 基本服务不能向上依赖流程服务
- 组合服务、流程服务可以向下依赖基本服务
- 条件：基本服务稳定

6. 核心服务依赖

- 核心服务不依赖非核心服务
- 非核心服务可依赖核心服务
- 条件：核心服务稳定

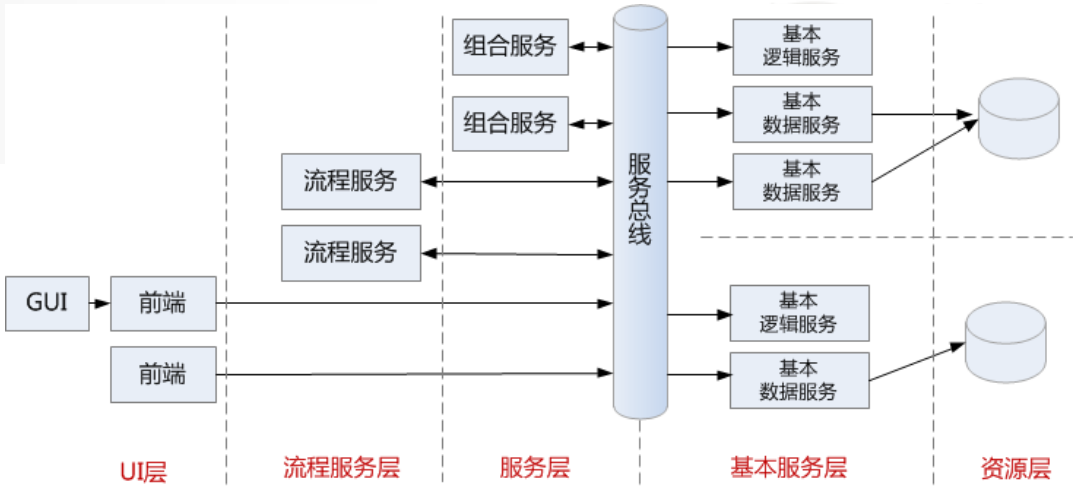
4. 非功能性服务依赖

- 非功能性服务不依赖功能性服务
- 功能性服务可依赖非功能性服务
- 条件：非功能性服务稳定

5. 平台服务依赖

- 平台服务不依赖上层应用
- 上层应用可依赖平台服务
- 条件：平台服务稳定

服务设计原则



无状态

- 尽量不要把状态数据保存在本机
- 接口调用幂等性

可复用

- 复用粒度是有业务逻辑的抽象服务，不是服务实现细节
- 服务引用只依赖于服务抽象

松耦合

- 跨业务域调用，尽可能异步解耦
- 必须同步调用时，设置超时和队列大小
- 相对稳定的基本服务与易变流程服务分层

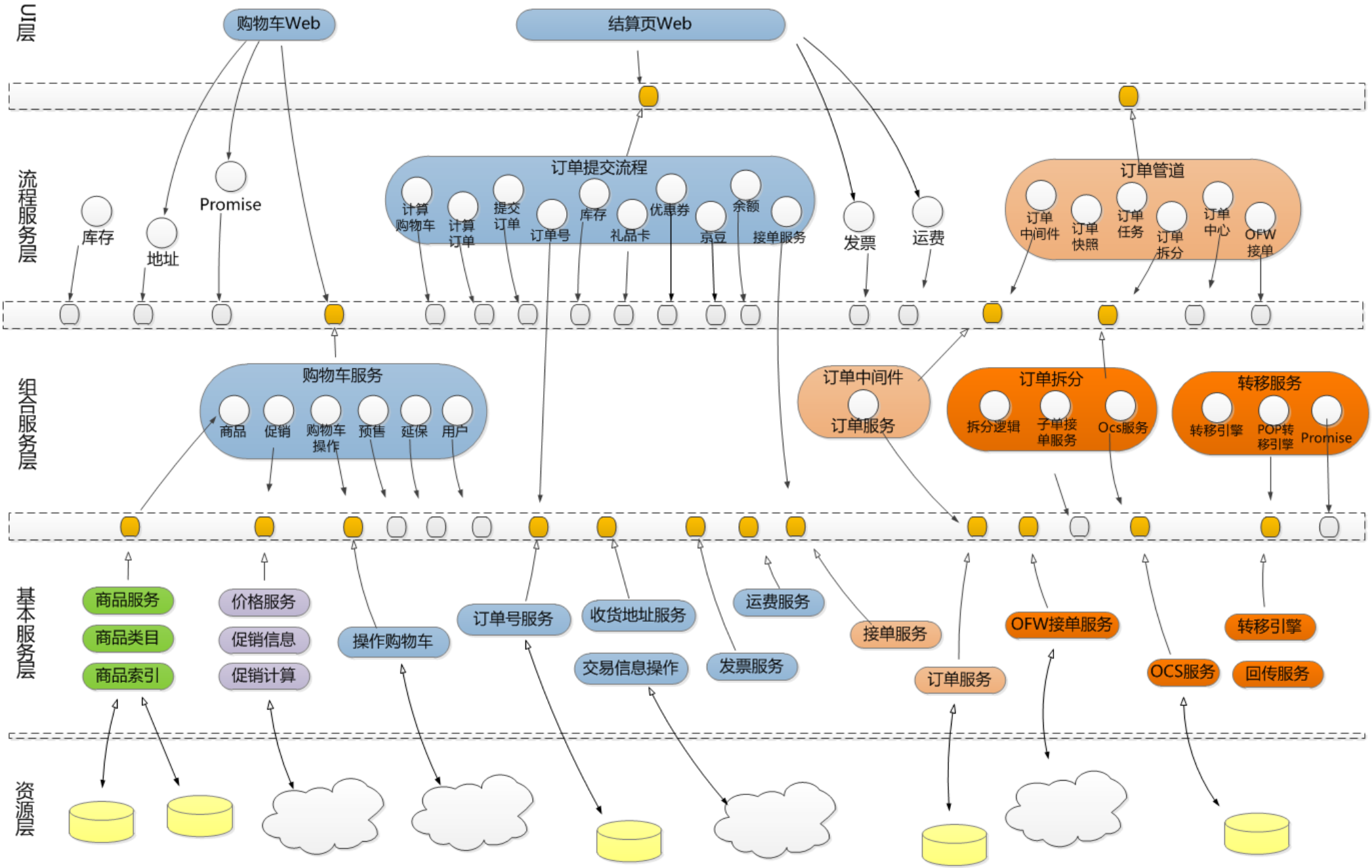
可治理

- 制订服务契约
- 服务可降级
- 服务可限流
- 服务可开关
- 服务可监控
- 白名单机制

基本服务

- 基础服务下沉、可复用。如时效、库存、价格计算等
- 基础服务自治，相互独立
- 基础服务的实现，要求精简、可水平扩展
- 基础服务实现物理隔离，包括基础服务相关的数据

应用架构实例：交易订单


UI层
购物车Web
结算页Web
流程服务层
库存
地址
Promise
订单提交流程
计算购物车
计算订单
提交订单
订单号
库存
礼品卡
优惠券
京豆
余额
接单服务
发票
运费
订单管道
订单中间件
订单快照
订单任务
订单拆分
订单中心
OFW接单
组合服务层
购物车服务
商品
促销
购物车操作
预售
延保
用户
订单中间件
订单服务
订单拆分
拆分逻辑
子单接单服务
Ocs服务
转移服务
转移引擎
POP转移引擎
Promise
基本服务层
商品服务
商品类目
商品索引
价格服务
促销信息
促销计算
操作购物车
订单号服务
收货地址服务
交易信息操作
运费服务
发票服务
接单服务
订单服务
OFW接单服务
转移引擎
OCS服务
回传服务
资源层

目录 CONTENTS

1 架构愿景

2 业务架构

3 应用架构

4 **数据架构**

5 技术架构

6 618经验

数据架构设计原则

1 统一数据视图

- 保证数据的及时性、一致性、准确性、完整性

2 数据、应用分离

- 应用系统只依赖逻辑数据库
- 应用系统不直接访问其它宿主的数据库，只能通过服务访问

3 数据异构

- 源数据和目标数据内容相同时，做索引异构。如商品库不同维度
- 内容不同时，做数据库异构。如订单买家库和卖家库。

4 数据读写分离

- 访问量大的数据库做读写分离
- 数据量大的数据库做分库分表
- 不同业务域数据库做分区隔离
- 重要数据配置备库；

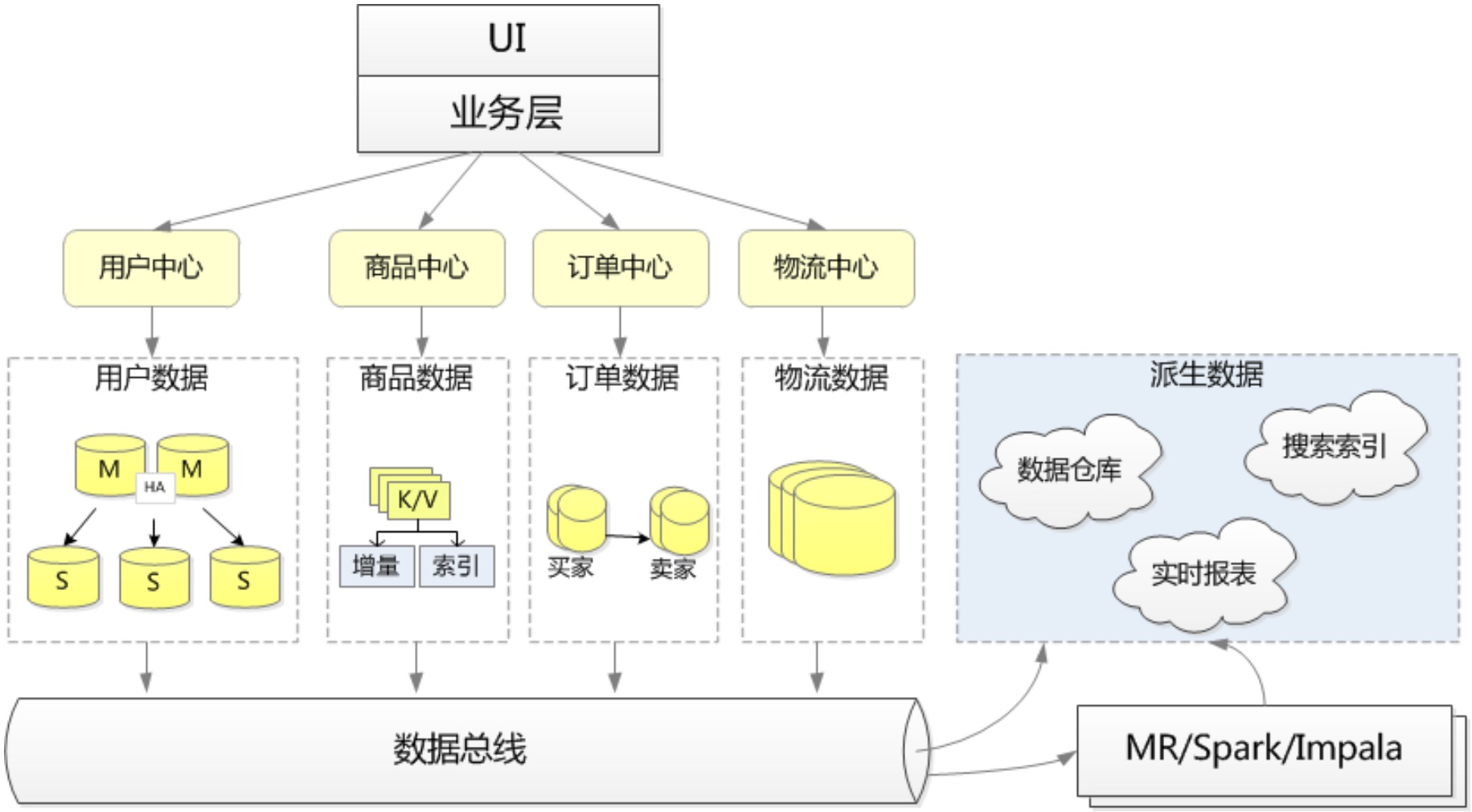
5 用Mysql数据库

- 除成本因素外，Mysql的数据库扩展性和支持高并发的能力较强，公司研发和运维在这方面积累了大量经验

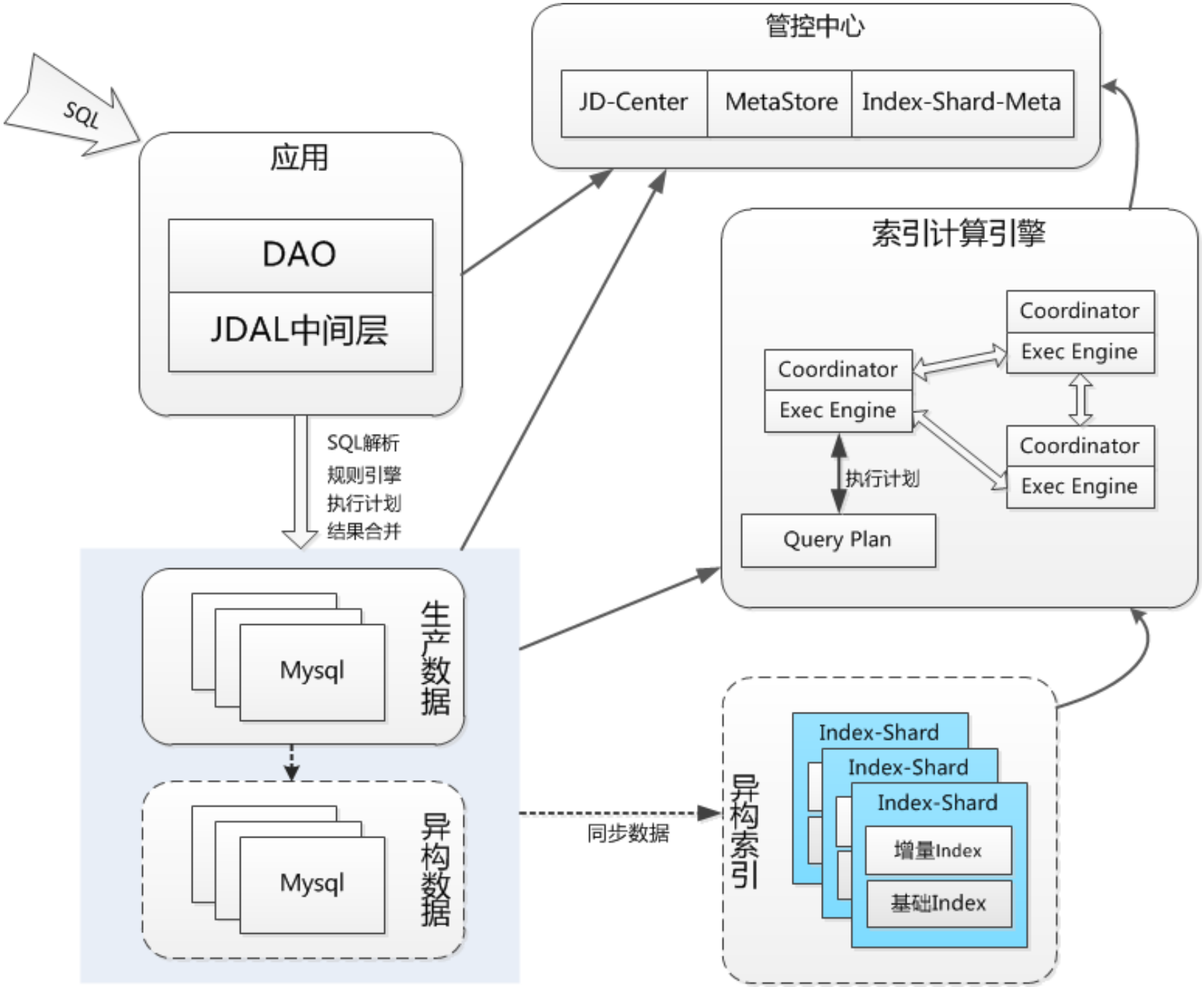
6 合理使用缓存

- 数据库有能力支撑时，尽量不要引入缓存
- 合理利用缓存做容灾

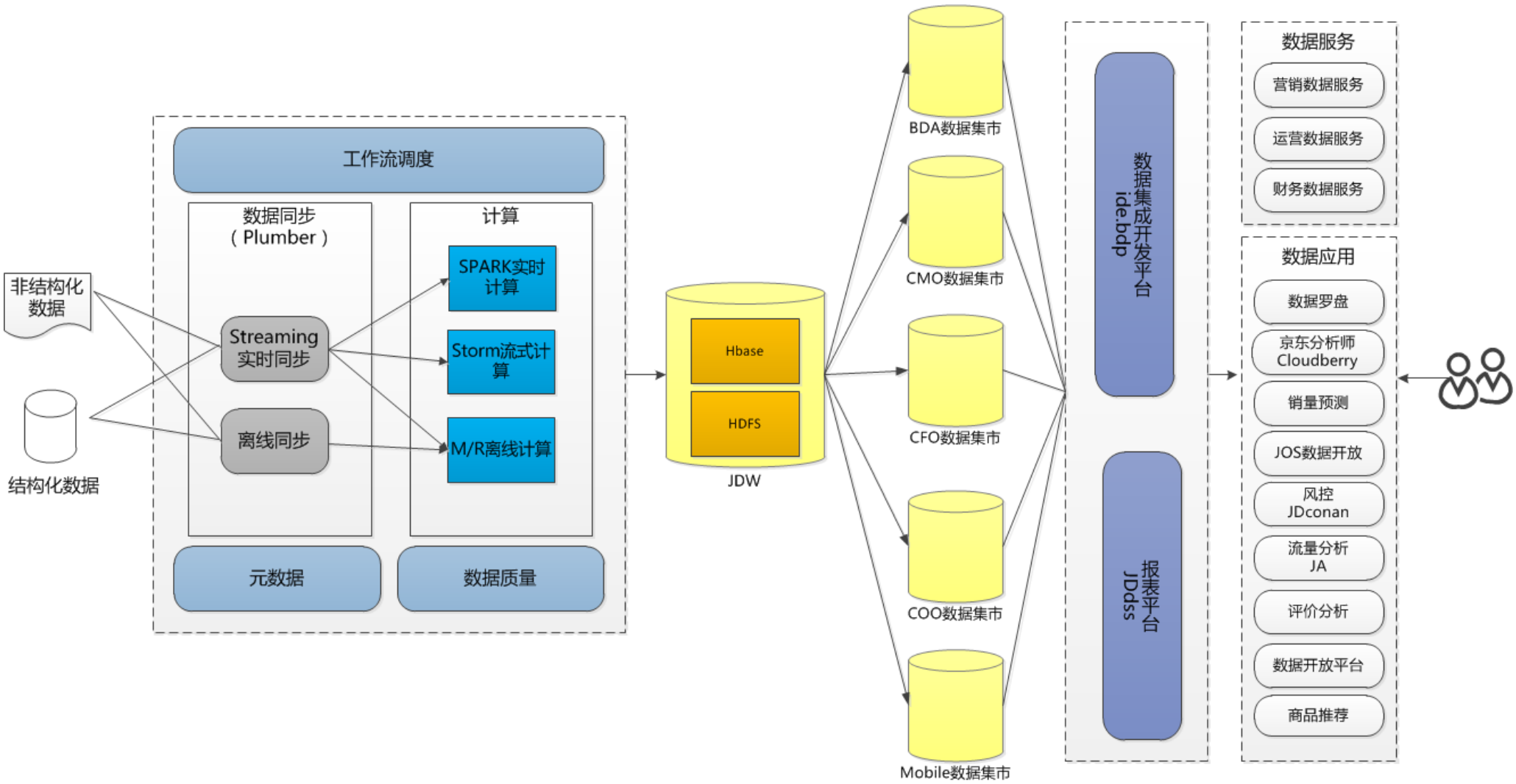
数据架构


UI
业务层
用户中心
商品中心
订单中心
物流中心
用户数据
M
M
HA
S
S
S
商品数据
K/V
增量
索引
订单数据
买家
卖家
物流数据
派生数据
数据仓库
搜索索引
实时报表
数据总线
MR/Spark/Impala

数据架构实例：分布式索引系统



数据架构实例：数据平台


工作流调度
数据同步
(Plumber)
计算
SPARK实时计算
Storm流式计算
M/R离线计算
Streaming 实时同步
离线同步
非结构化数据
结构化数据
元数据
数据质量
Hbase
HDFS
JDW
BDA数据集市
CMO数据集市
CFO数据集市
COO数据集市
Mobile数据集市
数据集成开发平台 ide.bdp
报表平台 JDdss
数据服务
营销数据服务
运营数据服务
财务数据服务
数据应用
数据罗盘
京东分析师 Cloudberry
销量预测
JOS数据开放
风控 JDconan
流量分析 JA
评价分析
数据开放平台
商品推荐

目录 CONTENTS

1 架构愿景

2 业务架构

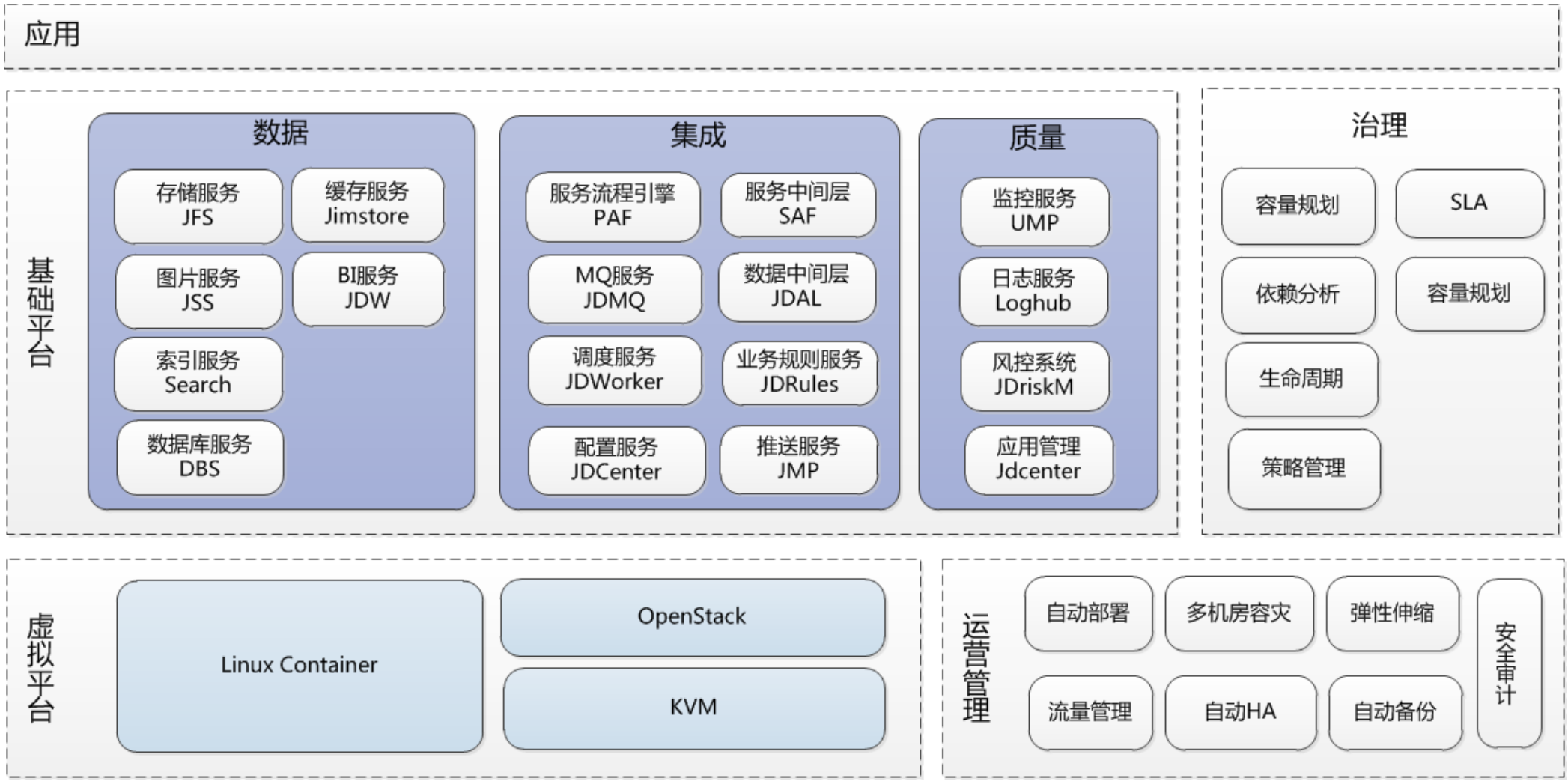
3 应用架构

4 数据架构

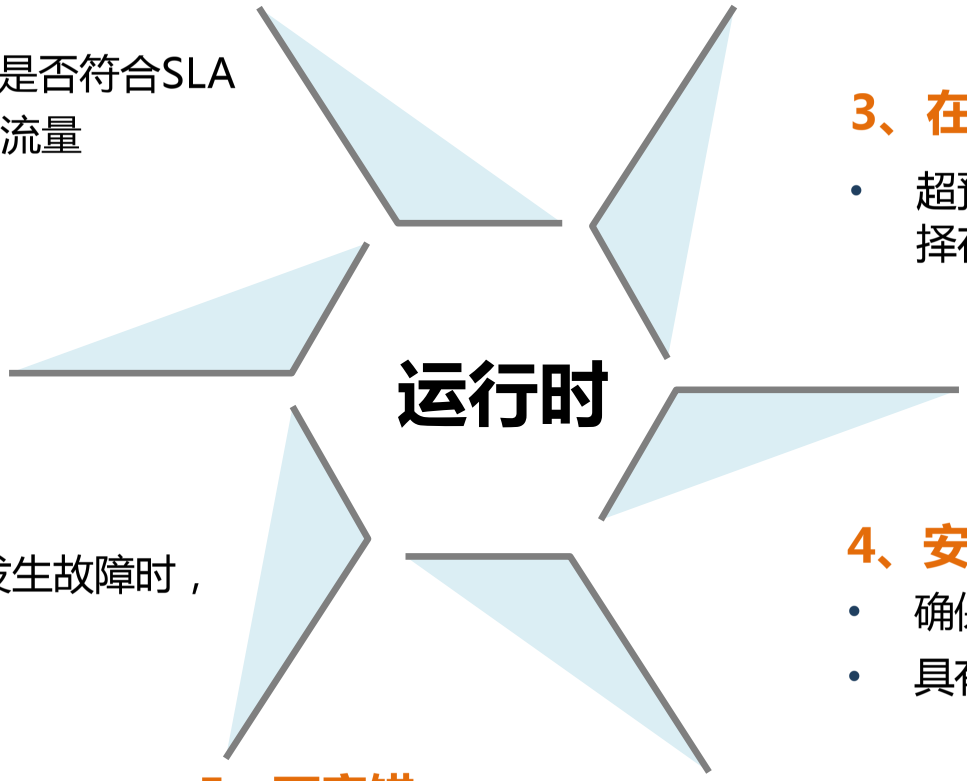
5 **技术架构**

6 618经验

基础架构


应用
基础平台
数据
存储服务 JFS
缓存服务 Jimstore
图片服务 JSS
BI服务 JDW
索引服务 Search
数据库服务 DBS
集成
服务流程引擎 PAF
服务中间层 SAF
MQ服务 JDMQ
数据中间层 JDAL
调度服务 JDWorker
业务规则服务 JDRules
配置服务 JDCenter
推送服务 JMP
质量
监控服务 UMP
日志服务 Loghub
风控系统 JDriskM
应用管理 Jdcenter
治理
容量规划
SLA
依赖分析
容量规划
生命周期
策略管理
虚拟平台
Linux Container
OpenStack
KVM
运营管理
自动部署
多机房容灾
弹性伸缩
安全审计
流量管理
自动HA
自动备份

系统运行时原则



1、可监控

- 服务的TPS和RT是否符合SLA
- 是否出现超预期流量

2、应用可回滚，功能可降级

- 应用出现问题时，要求能回滚到上一版本，或做功能开关或降级

3、在线扩容

- 超预期流量时，应用系统可选择在线水平扩展

4、安全保证

- 确保系统的保密性和完整性
- 具有足够的防攻击能力

5、可容错

- 核心应用要求多活，避免单点设计，并且自身有容错和修复能力。故障时间TTR小

6、可故障转移

- 多机房部署，发生故障时，能即时切换

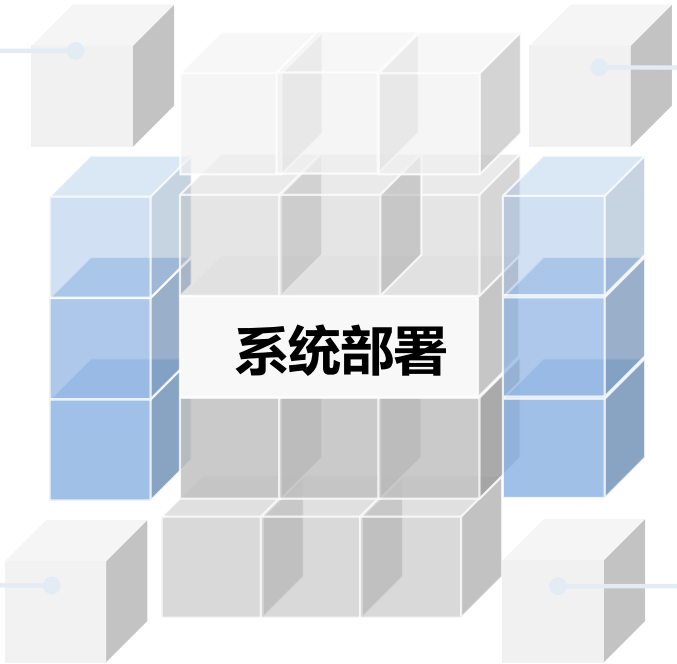
系统部署原则

2 D-I-D原则

- 设计20倍的容量（Design）
- 实现 3 倍的容量（Implement）
- 部署1.5倍的容量（Deploy）

1 N+1原则

- 确保为故障多搭建一套系统，避免单点问题。例如，多机房部署、应用系统集群、数据库主备等
- 功能开发与运维分开。系统开发完成后，交给专业的运维团队管理和运营。



3 支持灰度发布

- 系统新上线，要求支持“灰度”发布，分步切流量，故障回滚

5 业务子网

- 机房部署以业务域划分：基本服务和数据库，相同业务域的服务器部署在一起；不同业务域的服务器物理隔离

4 虚拟化部署

- 虚机部署：二级系统、三级系统采用虚拟机部署，节省资源和管理成本
- 虚拟化部署：一级系统应用服务器，采用虚拟化部署

目录 CONTENTS

1 架构愿景

2 业务架构

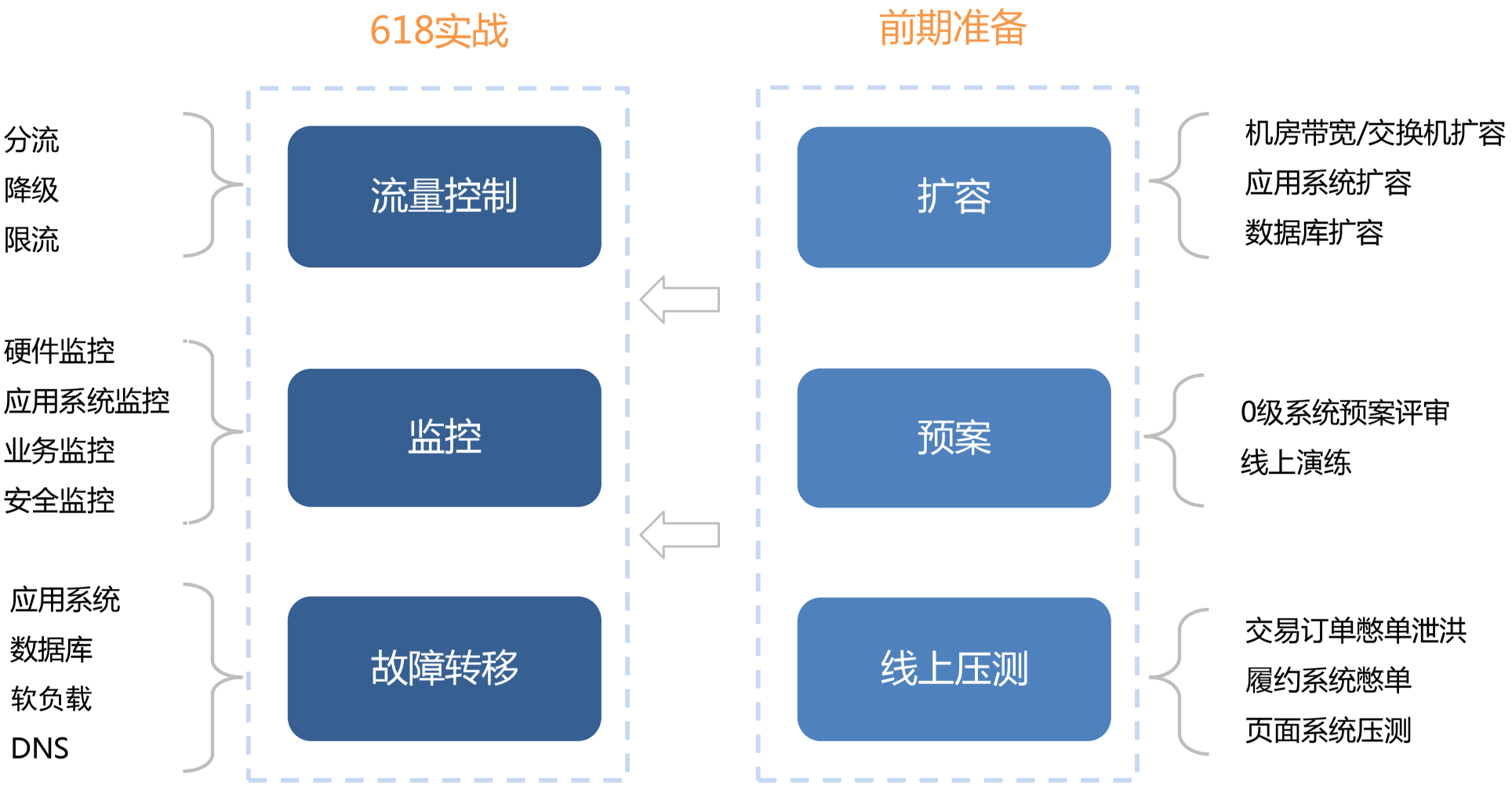
3 应用架构

4 数据架构

5 技术架构

6 **618经验**

618经验


618实战
前期准备
分流
降级
限流
流量控制
扩容
机房带宽/交换机扩容
应用系统扩容
数据库扩容
硬件监控
应用系统监控
业务监控
安全监控
监控
预案
0级系统预案评审
线上演练
应用系统
数据库
软负载
DNS
故障转移
线上压测
交易订单憋单泄洪
履约系统憋单
页面系统压测

流量控制

1. 分流

水平扩展	应用：集群，无状态，提高访问量 数据：读写分离，提高性能	商品读库，商品写库
业务分区	应用：按业务域划分成不同子系统 数据：数据分区	商品库、交易库
分片	应用：不同业务类型分片 数据：分库分表，提高数据容量	秒杀系统从交易系统中分离；非核心业务分离
动静分离	应用：分层，功能与非功能分开 数据：冷热数据分离	业务流程层、应用层

无法缓解大流量

2. 降级

页面降级	1. 动态页面降级到静态 2. 整体降级到其他页面 3. 页面部分内容
业务功能降级	1. 舍弃一些非关键业务，如购物车库存状态
应用系统降级	1. 降级一些下游系统，如一次拆分暂停
数据降级	1. 远程服务降机到本地缓存，如运费

无法缓解大流量

3. 限流

Nginx前端限流

京东研发的业务路由，规则包括账户，IP，系统调用逻辑等

应用系统限流

客户端限流

服务端限流

数据库限流

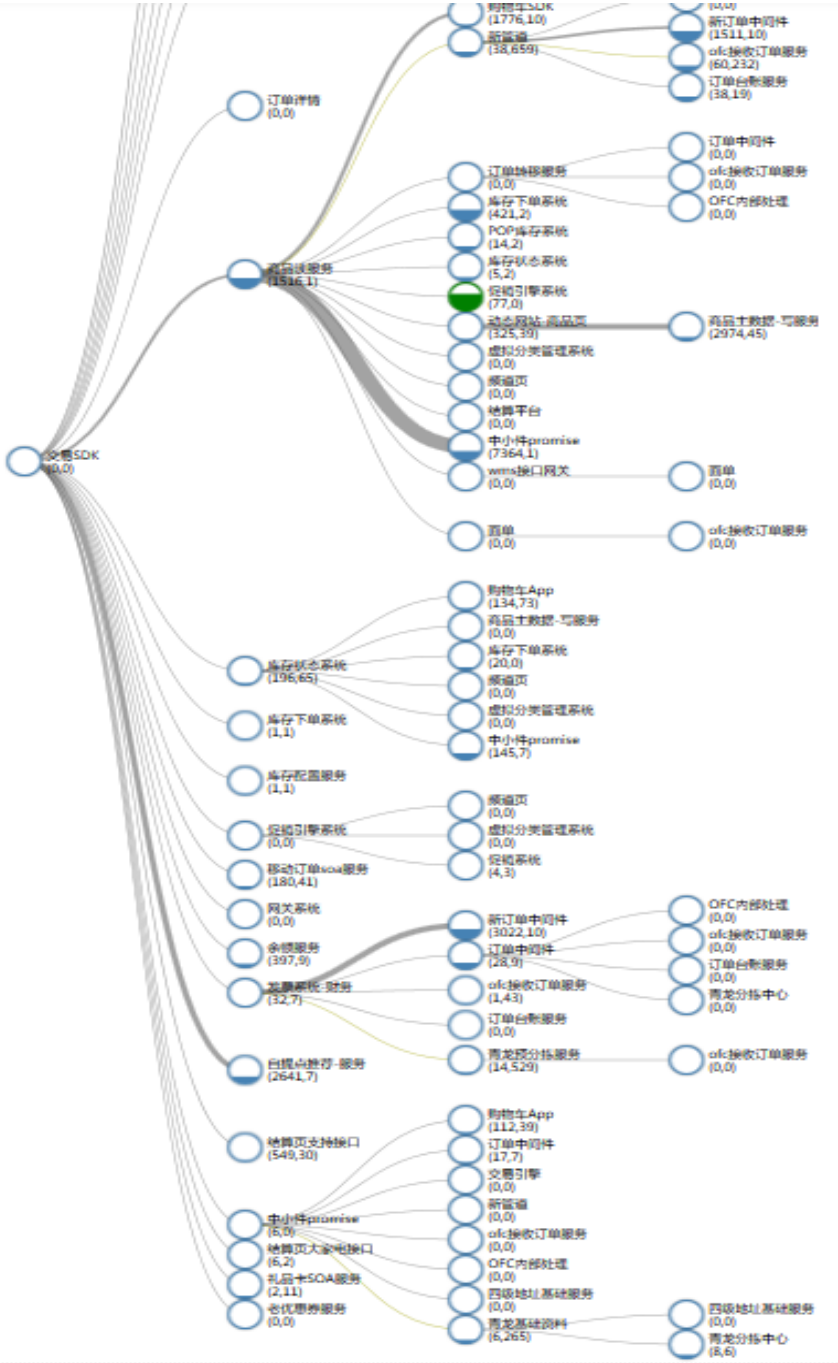
红线区，力保数据库

架构运行状态分析

目的：故障预测，故障隔离

功能：

- 显示应用之间的依赖关系
- 分析应用和服务的血缘和影响
- 根据依赖关系，分析应用的入出流量分配。超预期流量时，方便定位问题
- 根据应用系统运行情况，计算应用风险值
- 根据服务sla、tps、rt和依赖关系，评估服务风险值
- 全局风险评估，并动态更新，即时发现可能的问题



关系	应用名称	风险	TPS	RT(ms)	CPU%	LOAD	MEM%	带宽(M)	连接数	机器数	分级	类型	业务域
血缘 | 影响	任务引擎（含推送体系）(exadeliver)	30	67986	938	17	2.9	47	14	2500	15	1级	web	商品系统
血缘 | 影响	库存状态系统(stock-status)	63	33175	178	12	0.88	76	21	5767	27	0级	sdk	库存系统
血缘 | 影响	京东显示库存系统(webstock)	29	31645	178	21	0.68	66	21	4957	65	1级	sdk	网站
血缘 | 影响	商品中心系统 (pop-ware-ic)	38	15348	3384	12	3.09	57	33	1628	11	1级	sdk	POP平台
血缘 | 影响	网关系统(mobilegw-server)	25	10559	160	15	9.26	80	30	2029	64	1级	web	移动
血缘 | 影响	库存后端管理系统(stockadmin)	9	8233	178						1	2级	web	库存系统
血缘 | 影响	库存ERP系统(stockerp)	10	8233	177							2级	web	库存系统
血缘 | 影响	价格管理服务(skuPrice)	23	7101	17						4	1级	sdk	促销系统
血缘 | 影响	JSHOP商家装修平台(jshop-pop)	18	5889	33	9	1.19	76	14	3448	12	2级	web	POP平台
血缘 | 影响	评价中间件(sns-club-soa)	22	4156	579	13	0.35	81	56	5336	19	2级	web	用户中心
血缘 | 影响	商品读服务(pbia)	81	3775	19	11	0.85	53	4	13073	104	0级	sdk	商品系统
血缘 | 影响	自提点推荐-服务(srs)	20	2795	3	5	0.58	38	9	2072	61	2级	web	时效服务
血缘 | 影响	中小件promise(middle-promise)	35	2623	85							0级	sdk	时效服务
血缘 | 影响	移动商品soa服务(mobile-soa-ware)	28	2104	2242	31	1.48	51	113	2742	9	1级	web	移动
血缘 | 影响	销量分析系统(analysis)	21	1533	1	6	0.59	62	7	1578	9	2级	web	网站
血缘 | 影响	售后开放服务(afs)	16	1453	568	11	3.24	45	92	1758	5	2级	web	售后系统
血缘 | 影响	售后服务平台(afs-all)	16	1453	225	15	15.49	78	100	1522	19	2级	web	售后系统
血缘 | 影响	新订单中间件(order-middleware)	42	1442	289	3	0.52	31	7	676	14	0级	sdk	订单系统
血缘 | 影响	商品主数据-基础中间件(jd-item)	28	1228	8	4	0.11	21	20	2680	8	1级	sdk	商品系统
血缘 | 影响	商品主数据-写服务(exaitem)	68	1130	1078	11	2.33	61	33	3023	16	0级	sdk	商品系统
血缘 | 影响	购物车SDK(cart-sdk)	42	1127	143	6	1.13	46	9	4519	61	0级	sdk	交易系统

风险评估

风险评估：利用应用之间的关系，评估每个应用可能的风险大小。

计算方法：

一、风险指数：R = Rp * Rs * Ra

其中，Rp发生故障可能性， Rs故障影响严重程度，Ra发现和解决故障的能力，初始值为3。

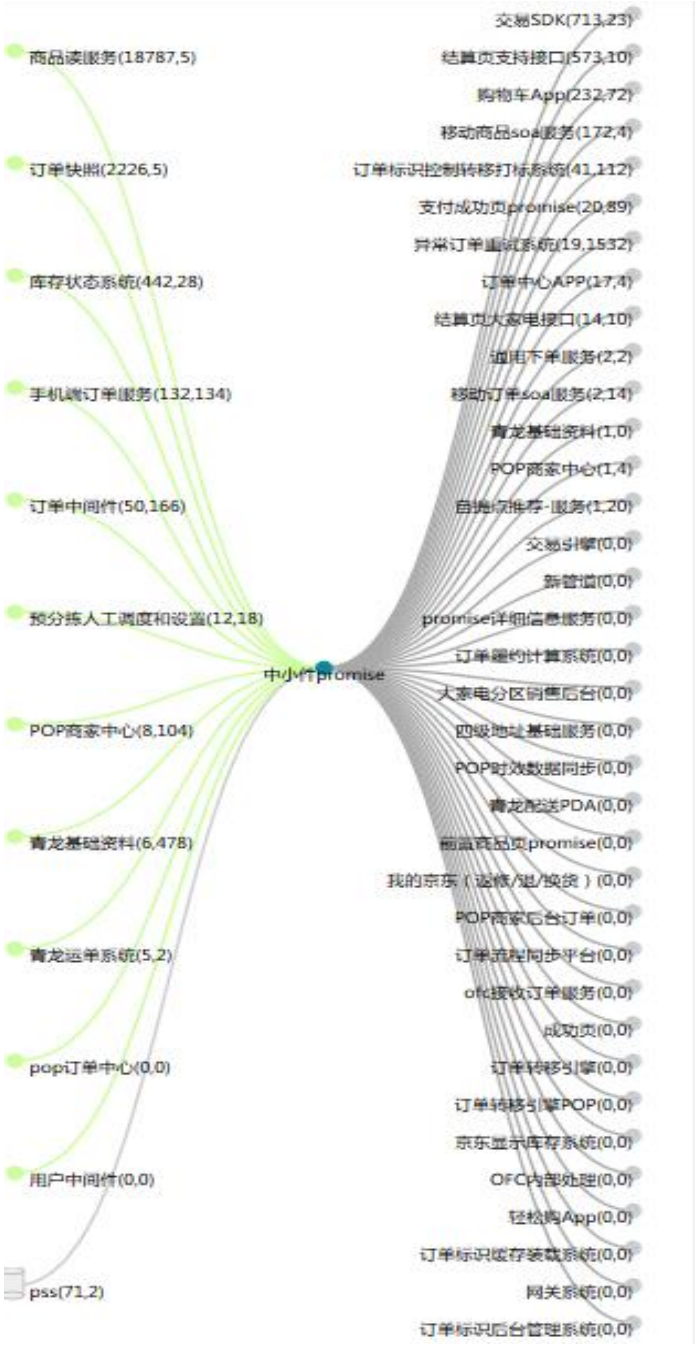
1、Rp计算：Rp = p0 + p(血缘关系)
其中，p0 = x0 * 10
p(血缘关系) = x1*w1 + x2*w2 + ... + xn*wn
x = f (mem , cpu , tps , rt)

2、Rs计算：Rs = s0 + s(影响关系)
其中，s0 = s0 * 10
s(影响关系) = y1*b1 + y2*b2 + ... + ym*bm
y = f (系统分级)

二、修正后的风险指数：C = Cp * Rs * Ca

Cp: 修正后发生故障可能性。根据618预案评估
Ca: 修正后发现和解决故障能力。根据618预案评估

三、根据修正值，迭代计算风险指数



容量规划

- 容量指标选择
- 压测

1. 容量评测

2. SLA

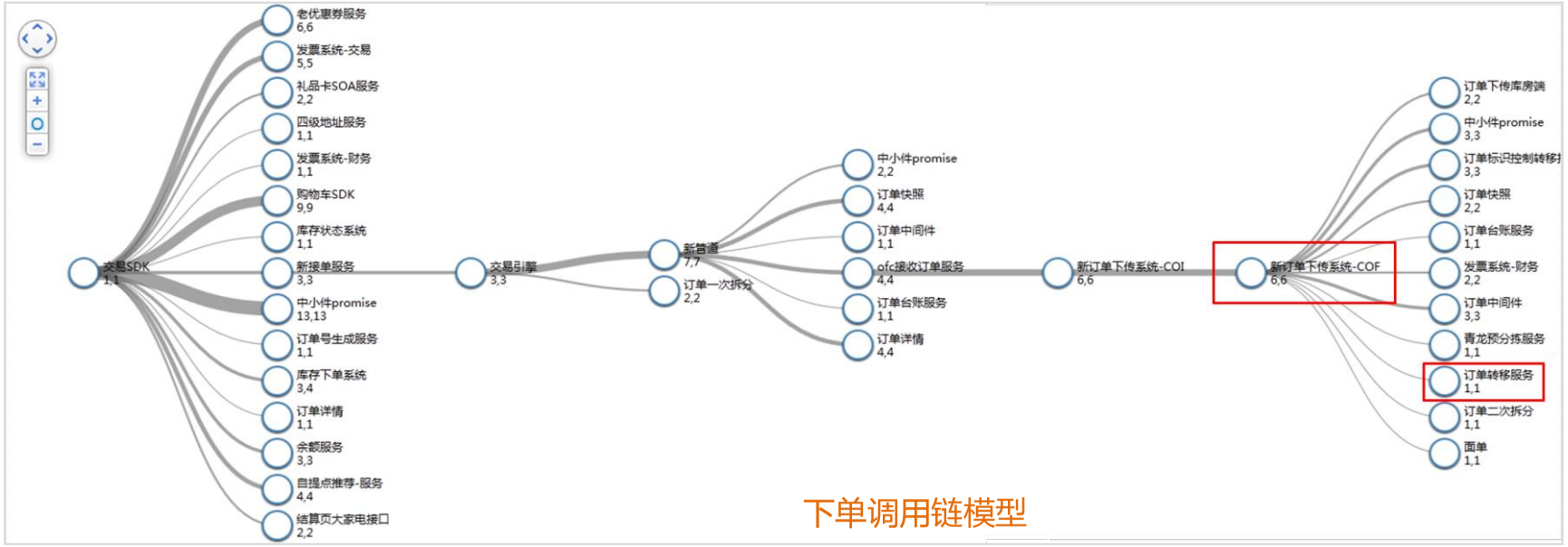
- 服务关系分析
- SLA制定
- 依赖治理

- 监测容量指标数据
- 为扩容、降级、降级提供依据

4. 容量管理

3. 趋势预测

- 下单调用链分析：每笔交易对应tps
- 单量预测：根据历史数据，预测618单量
- 根据调用链模型，估算各节点业务峰值



下单调用链模型

架构总结

	解耦/拆分	抽象	集成	复用	治理
业务	1. 电商业务域 2. 核心、非核心业务 3. 主流程、辅流程 4. 业务规则分离		1. 跨业务域调用异步 2. 非核心业务异步	1. 基础业务下沉，可复用	1. 厘清业务边界、作用域
应用	1. 应用集群水平扩展 2. 按业务域分离应用 3. 按功能分离应用 4. 按稳定性分离应用	1. 服务抽象，服务调用不依赖实现细节 2. 应用集群抽象，应用位置透明	1. 易变依赖稳定 2. 流程服务依赖基础服务 3. 非核心应用依赖核心应用	1.复用粒度是有业务逻辑的抽象服务	1. 服务自治 2. SLA 3. 可水平扩展 4. 可限流 5. 服务可降级 6. 容错设计 7. 服务白名单
数据	1. 读写分离 2. 按业务域分库 3. 分库分表 4. 冷热数据分离	1. 数据库抽象。应用只依赖逻辑数据库	1 数据库只能通过服务访问 2. 统一的元数据管理 3. 统一的主数据管理		1. 重要数据做主备 2. 合理利用缓存容灾 3. 双写要做补偿
技术	1. 功能开发与运维分离 2. 业务子网 3. 分离功能、非功能型需求	1. 服务器资源抽象。应用只依赖虚拟化资源	1. 同步调用时，设置超时和任务队列长度 2. 利用回调异步化 3. 利用MQ、缓存、中间件异步化	1. 代码提共通，可复用 2. 非功能性服务，可复用 3. 基础配置、基础软件复用	1. N+1设计 2. 灰度部署 3. 版本可回滚 4. 可监控 5. 可容灾

谢谢！
Thank you!

北京市朝阳区北辰西路8号北辰世纪中心A座6层
6F Building A, North-Star Century Center, 8 Beichen West Street,
Chaoyang District, Beijing 100101
T. 010-5895 1234 F. 010-5895 1234
E. xingming@jd.com www.jd.com

JD.京东.COM