

【解密】京东B2B业务架构演变

原创：李sir 京东技术 2018-11-02



来这里找志同道合的小伙伴！

B2B交易平台部

以分销、代销业务为核心的B2B交易平台，通过建立各级商家之间渠道通路的方式，赋能线上线下B用户，并为其提供一站式采购、售卖的解决方案。

B2B业务介绍和业务发展

B2B业务介绍和业务发展

01

业务介绍

京东 B2B 业务的定位是让各类型的企业都可以在京东的 B 平台上进行采购、建立采购关系。

京东 B2B 的用户群体主要分为 2 类，一类是大 B 用户、另一类是小 B 用户。比如联通、移动公司跟京东建立的采购关系，就是 B 平台的大 B 用户；如果有一家小超市需要在京东 B 平台上进行采购，那么它就是 B 平台的小 B 用户。

京东 B 平台需要支持各类型的用户群，因此必须要有自己的业务系统做支撑，比如订单、商品、价格、用户、权限、审核等系统。

02

业务发展



从上图可以看出，京东 B 平台的发展分为3个阶段：

1) 第一阶段（2014年）

B2B 浪潮开始兴起，京东在2014年与联通公司达成合作，意味着京东正式迈入B2B时代的大 B 行业。

2) 第二阶段（2015-2016年）

农村电商开始兴起，线下门店积极顺应互联网的发展趋势，将传统的零售搬到了线上；在这个阶段，京东成立新通路事业部开展此业务，从此京东正式迈入了小 B 行业。

3) 第三阶段（2017年至今）

在之前大、小 B 业务的基础上，京东的 B2B 业务在2017年得到快速发展，完美应对这个阶段产生的种种挑战，并发量、数据量均成几十倍的增长。

B2B技术架构演变
BSB技术架构演变

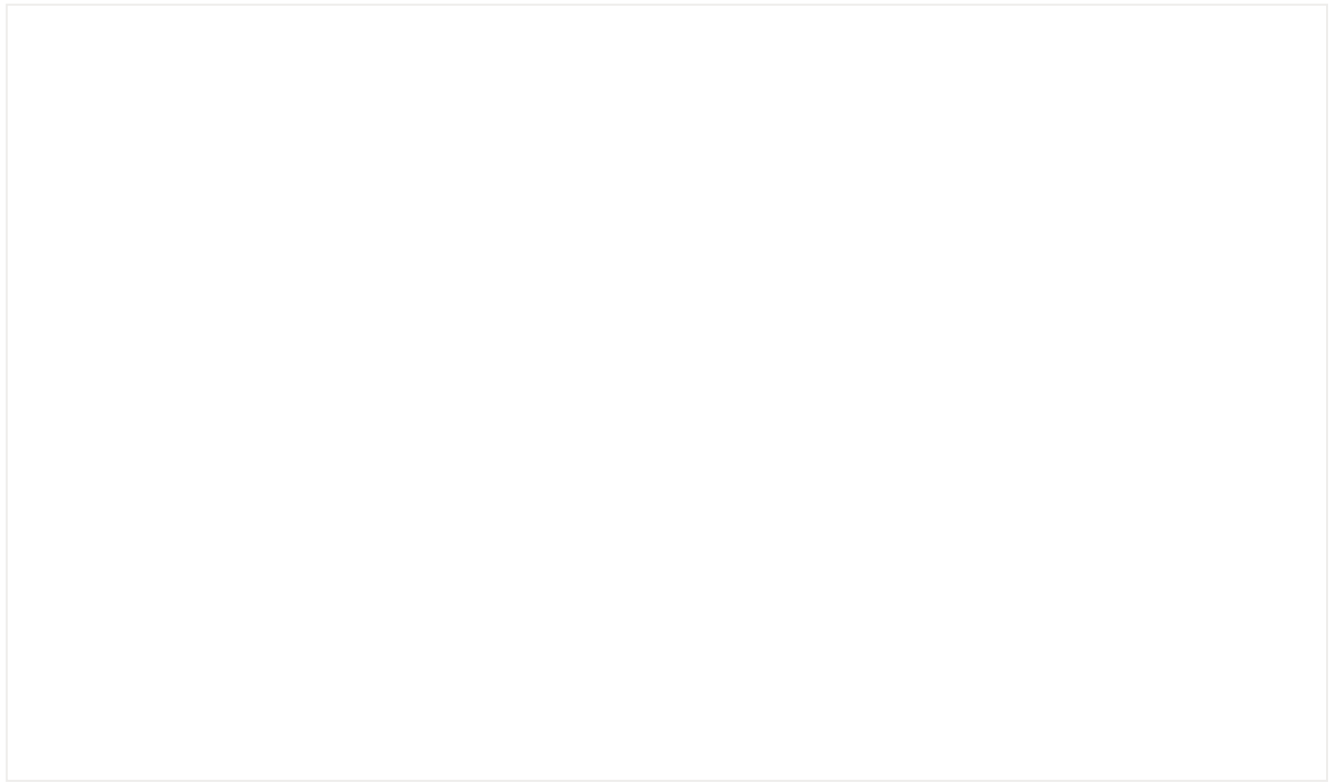
1) 架构



从上图可以看出，业务架构 1.0 分为 3 层：

- 业务层：主要是 B 平台的所有业务线
- 服务层：包含订单、价格、商品、用户等 SOA 服务系统
- 存储层：使用 mysql 数据库进行存储

2) 问题



在2014年初期，为了响应业务的发展，我们需要快速上线业务系统，采取的是纵向按业务线进行划分，每一条业务线都有自己的业务层、服务层、存储层，当有新的业务线接入的时候，还是以同样的模式进行开发。

当来到第二阶段的时候，这种架构面对了极大的挑战，主要有以下几个表现：

- 开发周期长，无法快速满足业务要求
- 服务之间的相互影响，订单和商品在一个数据库，一个出问题，会影响别的服务
- 系统之间耦合度大

上述的表现反应出业务架构存在以下几点问题：

- **服务问题**

服务之间零复用性，各个业务线的服务没有抽取共享的服务，其实有80%都是相同的模式，没有抽象。

- **系统耦合**

系统之间的相互调用采用的 jsf 服务，全部是同步调用，调用链路很长，一个环节出问题会导致整个系统都出问题，没有核心服务和非核心服务、没有异步化服务。

- **公用数据库问题**

由于前期是按业务线进行划分，在一个业务线上，核心服务的数据都存储在一个数据库上面。

3) 服务问题改进



- **第一步拆分**，将各个业务线的的服务拆分成小服务。
- **第二步拆分**，组合第一步抽取的服务，形成公共服务，以后所有业务线都请求这些公共服务，提高服务的复用性。

4) 测试方案



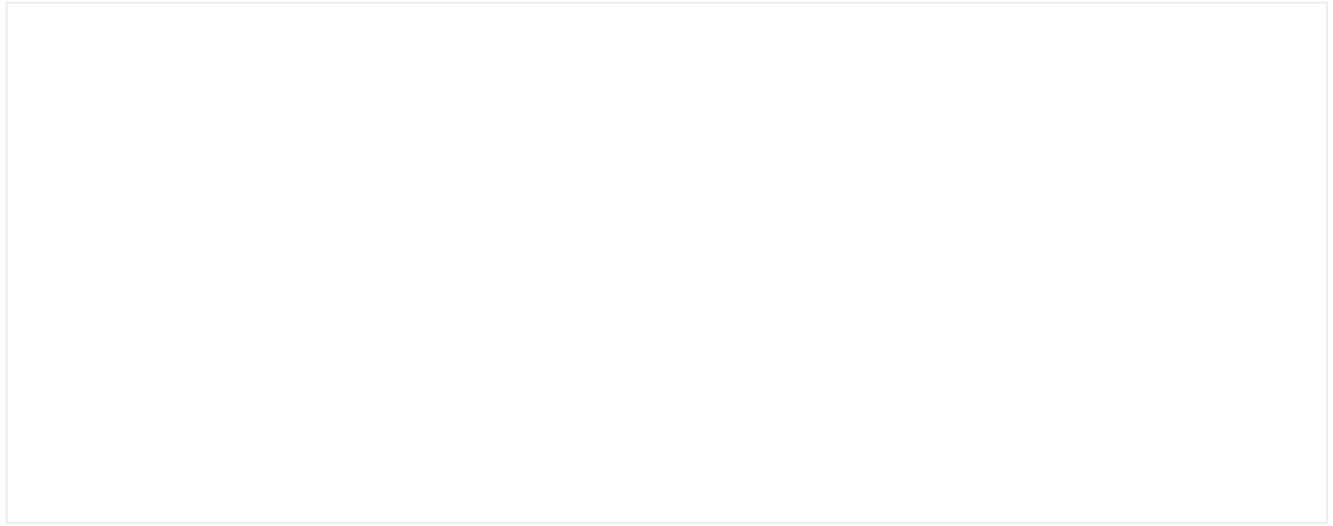
- 配置（人工）
- 采集数据（自动）
- 分发请求（自动）
- 数据对比（自动）
- 核对（人工）

条件：新老服务的对外接口参数不变

5) 系统耦合改进

系统耦合的问题，通过引入 jmq 消息中间件进行解决。消息无序的问题，采用乐观锁进行解决，主要是依靠数据的版本对比来解决。

6) 数据库改进



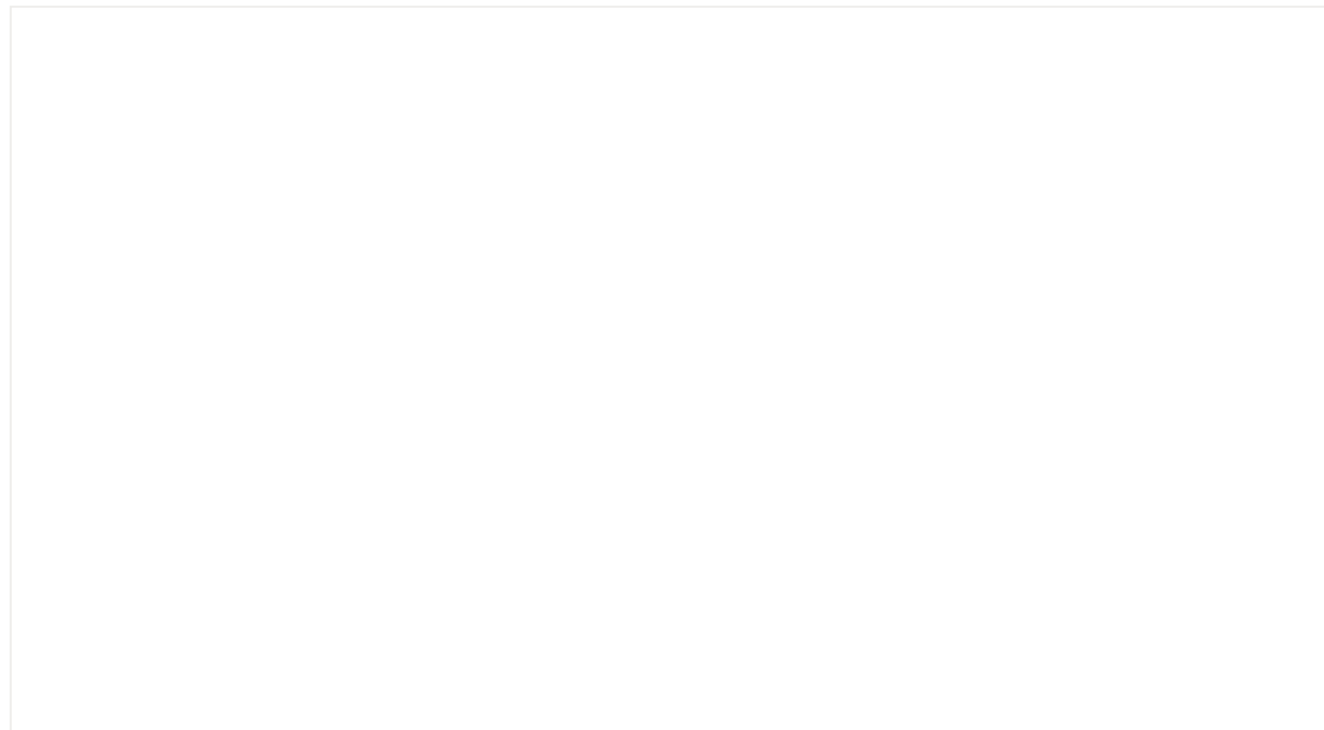
数据公用的问题，解决方案还是进行拆分：

- **第一步**，将各个业务系统 SOA 服务的数据，单独存储在自己的数据库，订单有订单专门的数据库、商品有商品专门的数据库，服务之间互相不受影响。
- **第二步**，在第一个步拆分后，有的业务数据量单表数量还是很大，需要对表进行拆分，我们采用 jproxy（不支持分表）进行分库，按业务的相关主键 id，进行 $\text{hash}(\text{id})\% \text{count}$ （分库数量），支持水平扩展。

扩展：

- 还有那些分布算法方式：ID 区间算法、时间区间
- 热点数据：二次 hash
- 事务：弱化强一致性，采用消息的机制进行交互，实现最终一致性。
- 垮库查询：分布式查询

7) 分布式搜索



分布式查询，采用的是 elasticSearch 搜索进行支持，简称es。

- 消息同步才用 jmq 和 binlog
- 如果 es 集群有问题，采用的方式是备份集群支持服务（数据有延迟风险）

02 业务架构2.0

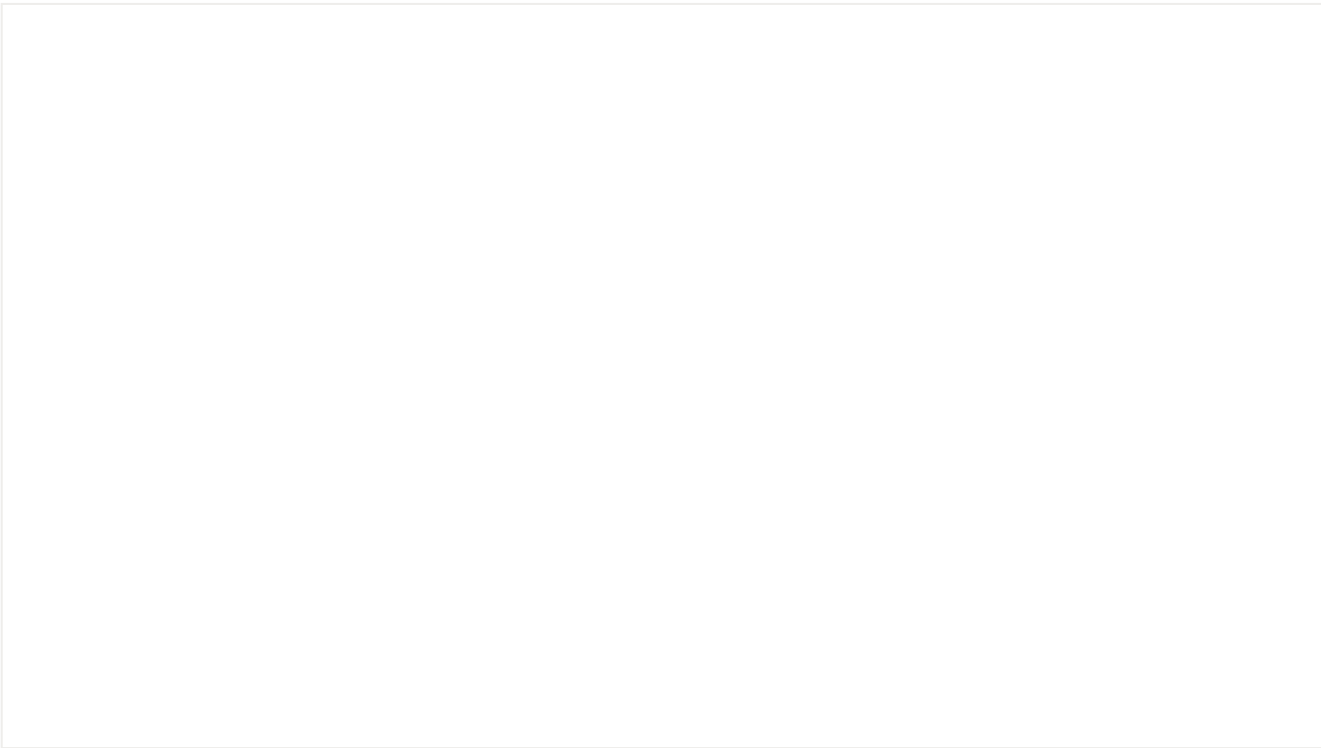
1) 架构



在1.0的基础上，做如下三点调整：

- 服务层抽取公共服务
- 引入基础层的工具
- 存储层分库分表

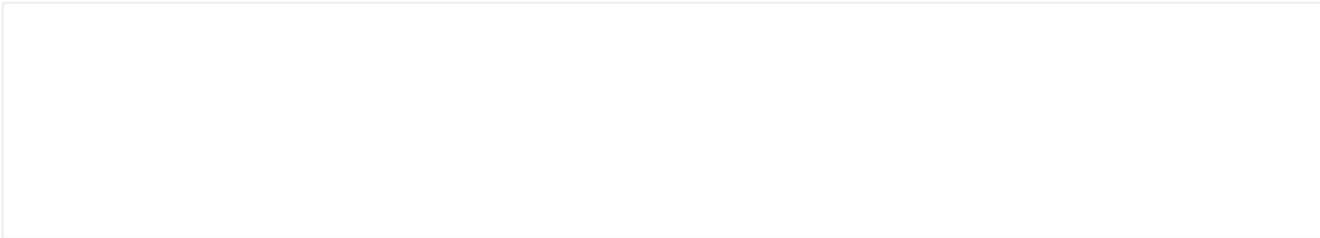
2) 问题



2.0系统在2017年进入第三阶段后，开始面临以下两个问题：

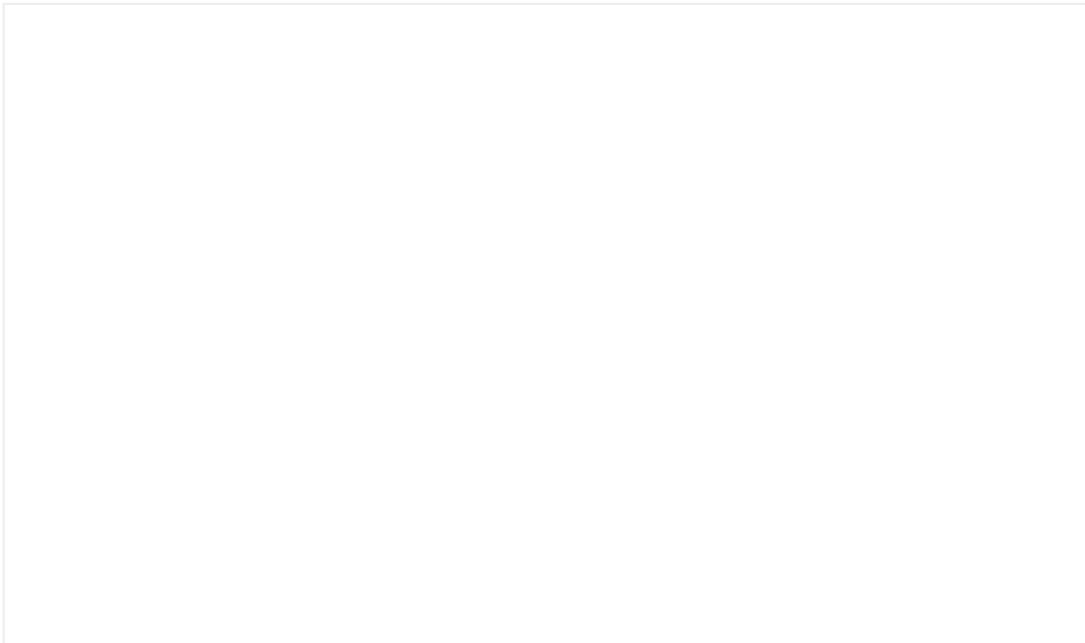
- 平台化服务业务代码臃肿
- 个性化需求的重复开发

3) 思考



举一个例子，我们现在有「订单」、「价格」、「商品」服务，A 用户需要订单服务，B 用户需要订单、价格服务，C 用户需要订单、价格、商品服务。现在的做法是开发3套业务接口服务，底层的
服务是通用的，但是业务层的代码有重复开发、业务代码臃肿的问题。

我们该怎么做？



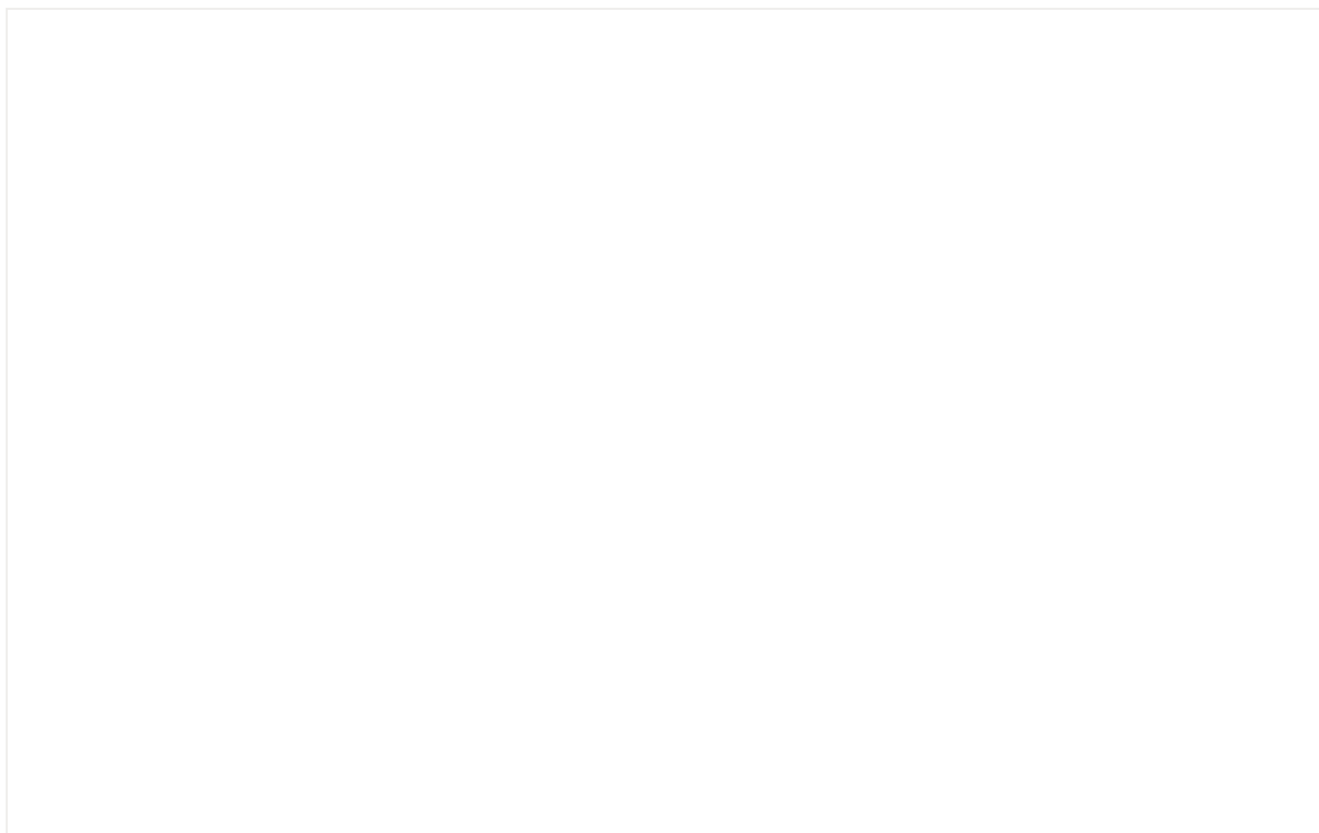
引入配置中心

- 对服务进行配置

- 对页面进行配置
- 可以自定义插件服务

4) 组件

通过以上的思考，我们有了组件的思想，以后对外的服务都是可配置的。就像一个加工厂，对服务加工，就可以对外部业务进行使用。



03

业务架构3.0



3.0的版本主要是修改，包含服务层的抽取、业务和 SOA 分离，同时引入业务和工具组件。

总 结

资 源

B2B业务架构经过3次变迁与升级，我们总结到以下经验：




-----END-----

下面的内容同样精彩

点击图片即可阅读

▼



京东技术
---关注技术的公众号

长按识别二维码关注

