架构师

ARCHITECT

特刊

揭秘京东618背后的技术力量

SPECIAL ISSUE

June,2016

架构师特刊

Geekbang. 极客邦科技 | InfoQ

目录

卷首语

技术创造价值的几个层面

京东集团 CTO **张晨**

每个互联网公司都如此看重技术，但实际上技术在行业发展的每一个阶段，价值是完全不同的。以电商为例，最开始技术是一切业务的基础，也是业务背后的支撑。但如果仔细分析，你会发现这时候技术所做的事情主要是把传统的业务流程数字化、互联网化，用技术把人们从繁琐的工作中解放出来，让他们更多地从事决策性或是创造性的任务，换一句话说，这时技术扮演的主要是工具的角色。

而且对于京东这种体量和发展速度的企业来说，让技术满足业务成长需求已经是个相当大的挑战，需要从技术策略、架构到具体实施都缜密思考。同时，京东紧密跟踪行业最先进的技术和应用，因为每一个进步对于我们都可能有着巨大的价值。京东是技术应用的最佳土壤——这是我加入京东后经常对外说的一句话。确实，我们不仅在系统的体量、规模上非常独特，而且面临的是海量用户的体验，让技术能充分体现它的价值。

在这组分享京东 618 技术经验的文章中，大家可以看到京东在弹性云、多中心交易、高可用架构、平台化、风控系统等等领域的实践经验，在业务保障方面的技术进步让我们可以更加从容地应对 618 这样的大促。如果你有机会走进今年京东 618 作战指挥中心，看到数字不停变换的监控大屏和各个版块集中应战的京东技术人，你会惊讶地发现，从技术角度而言，我们度过的是一个近乎完美的，

一切尽在掌握的618，订单洪峰在经过千锤百炼的技术系统前被有条不紊地化解。

业务保障能力的完善和提高，让我们有能力，也必须用技术去创造更多的价值，让它不仅仅是人类改变现状的工具，更成为开启未来的桥梁。我们在这里主要谈的就是大数据。拥有了精准且丰富的数据，加上出色的数据挖掘技术，我们就可以完成许多人力无法企及的工作，例如针对每一个用户的个性化服务，京东称之为智能卖场。通过这个项目，每个用户都会看到不一样的产品展示、促销推荐、搜索结果，就像京东为每人配备了一个既了解用户消费习惯、潜在需求，又掌握商品知识和所有促销信息的专职电商管家。

这时，技术的价值就不仅仅是支撑和提高效率，用户会有完全不同的购物体验，他的购物路径会缩短，转化率会提升，用户和京东都因为这个项目收获了新的价值。而这种创造价值的能力会让技术在企业中扮演更为重要的角色，也让技术人成为描绘未来的艺术家。

最后，再次感谢 InfoQ 的专家们如此深入京东的技术体系，收集了大量一手资料并进行了很多细致的采访，让更多中国互联网行业的同仁们可以分享京东的技术经验。京东一直在实践——而不仅仅是思考技术能如何改变我们的生活，希望 6000 位京东技术人的不懈努力和创造会让我们的生活更加简单快乐。

京东 618：
15 万个 Docker 实例，所有业务全部容器化

作者 **穆寰**

在 2015 年的 618 大促中，京东大胆启用了基于 Docker 的容器技术来承载大促的关键业务（图片展现、单品页、团购页），当时基于 Docker 容器的弹性云项目已经有近万个 Docker 容器在线上环境运行，并且经受住了大流量的考验。而今年 618，弹性云项目更是担当重任，全部应用系统和大部分的 DB 服务都跑在 Docker 上。像 618 大促这样的流量高峰期，弹性云可以自动管理资源，做到弹性扩展，而在流量低谷期，又可以进行资源回收，在提升资源利用率的同时确保了运维系统的稳定性。据官方估计，本次大促活动中，京东线上将会启动近 15 万个 Docker 容器，从数量上来看，京东是全球范围内 Docker 的应用大户之一。为了了解相关的详情，InfoQ 记者采访了弹性云项目负责人鲍永成。

InfoQ：去年 618 的时候，我们就有聊过京东 Docker 的应用情况。您能对比去年 618，介绍下今年的规模、应用以及调整吗？

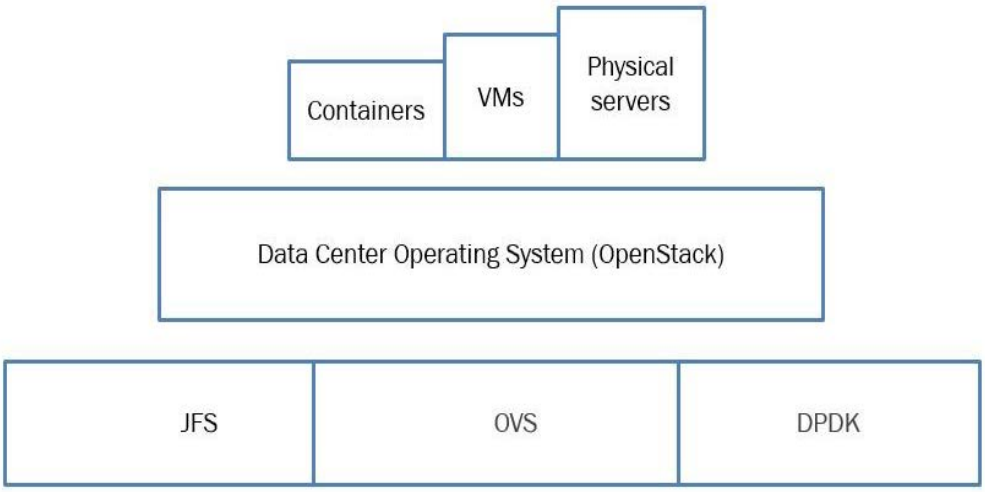
鲍永成：从数量上来讲，去年 618 线上容器应对峰值为 9 千个实例，今年截止 6 月 17 日线上容器突破 15 万实例；在整体布局上来看，对比去年 618，弹性云在规模上和业务全容器化实现战略落地。

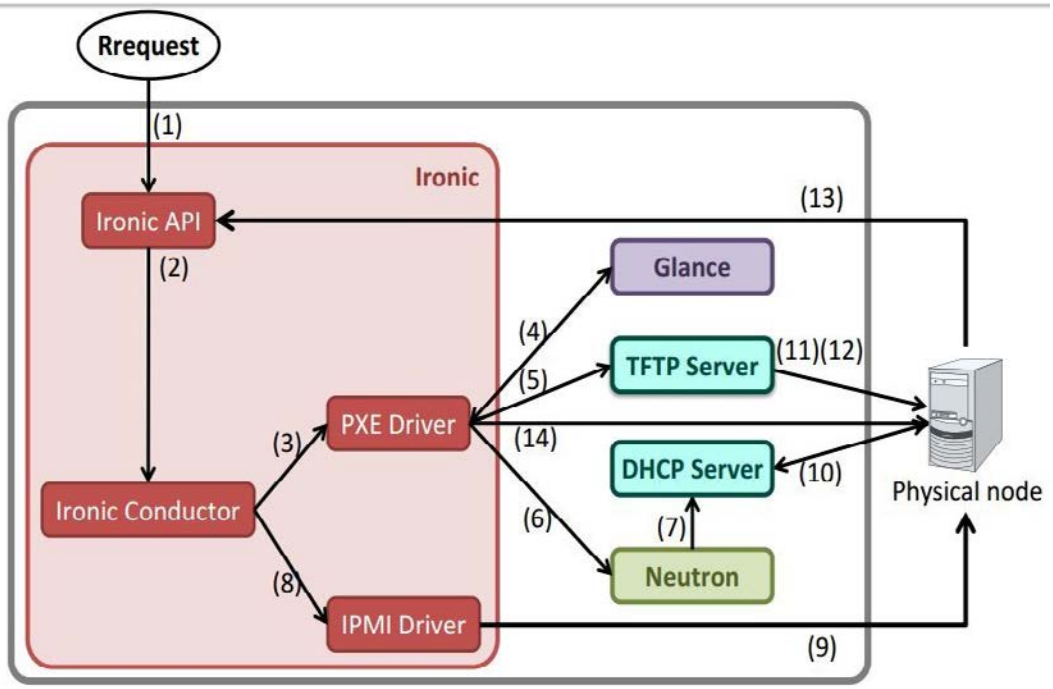
在应用层面，京东所有应用 100% 通过容器技术来发布和管理应用集群。值得指出一点：今年 618 有 5600 个容器实例支撑 DB 集群，对京东云数据库提供非

常便利的支持。

弹性云核心架构没有很大变化，依然简洁定义：弹性云 = 软件定义数据中心 + 业务容器集群调度。在此基础上，有两点增强：

- 单个容器的稳定性和性能方面做了很大提升，有效满足核心系统对计算和网络的巨大需求；
- 统一管理物理机、虚拟机和容器，加入统一集群进行调度，以适应不同业务对计算资源的不同需求。





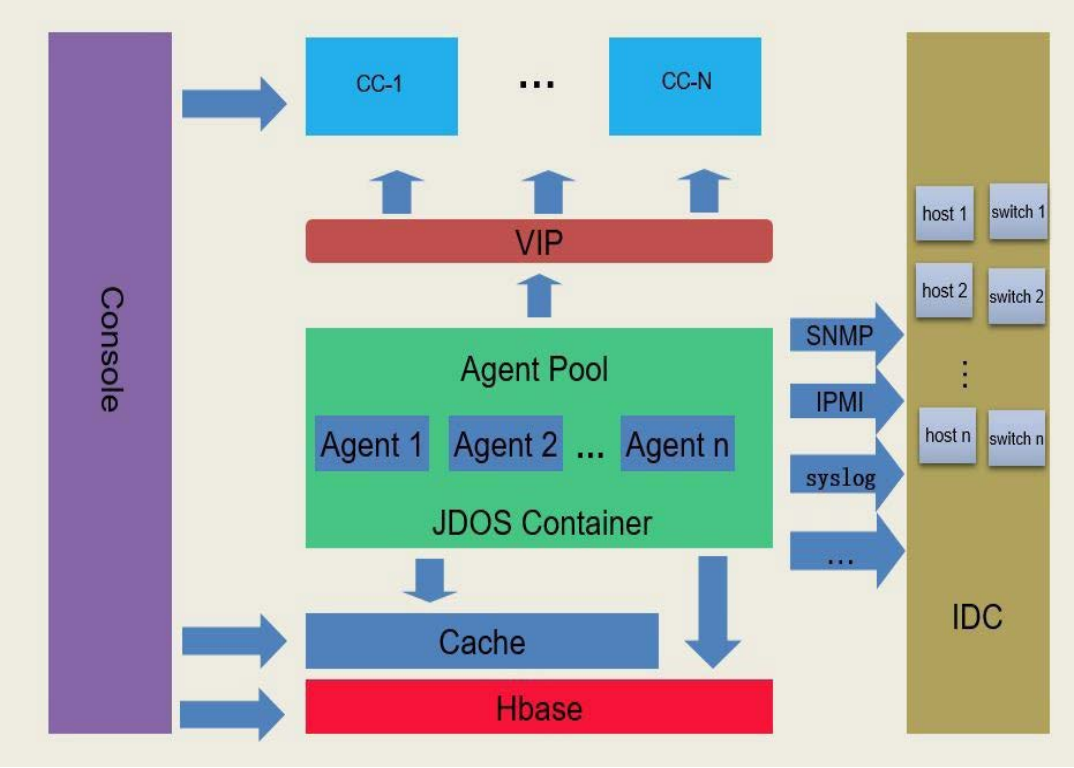
InfoQ：您能介绍下有哪些业务全量跑在 Docker 上吗？为了迎接这样的挑战，架构做了哪些调整？

鲍永成：网站、交易、无线、微信手 Q 等全部应用系统和 DB 服务的大部分都跑在 Docker 上。

因为京东业务在多年前就开始微服务化治理，所以应用层架构调整很小。现如今微服务化比例已经很大，所以容器技术的融入比较顺利。弹性云平台，算是站在多年对各个业务系统微服务化治理的巨人肩膀上。

InfoQ：这么多容器，如何做到海量监控？采用了什么开源方案吗？有哪些注意的事项？

鲍永成：监控采取自主研发系统，该系统负责海量数据采集和存储工作，其构架如图。



对于 15W+ 容器监控，需要注意：指标采集传输一定要设计得非常高效，并且要做到采集过程对资源的消耗控制，建议使用加速告警和监控图表跟踪缓存状况。

InfoQ：在洪峰来临之时，弹性计算云平台是如何进行扩容的？可以从系统角度分享下其中的流程吗？

鲍永成：弹性云有两种模式：

- 自动模式会根据每个业务方自己的预设弹性扩展条件，自动触发扩容。扩容工作包括使用业务镜像spawn实例、自动DB授权、微服务框架注册、添加实例到LB负载。
- 手动模式的前提是有弹性事件被触发，垂直运维收到扩展确认消息，人工参与的只是这个消息的点击确认扩容环节，其后续流程与自动模式一致。

InfoQ：可以谈谈目前应用了哪些关键开源项目吗？各自在系统的哪些 level？对应的版本分别是什么？

鲍永成：主要是 OpenStack 和 Docker，具体如下：

- OpenStack-IceHouse 用于管理数据中心计算、网络、存储等资源，位于JD Data Center Operating System层；
- Docker-1.3用于spawn容器实例，加入很多自主研发的内容和功能。

InfoQ：这么多的容器，您觉得最大的挑战是什么？

鲍永成：

1. 集群规模：介于高效运维，弹性云集群走的是大集群架构的思路。单个 OpenStack 集群会建设得非常大，目前单集群规模控制在 6 千台计算节点左右。

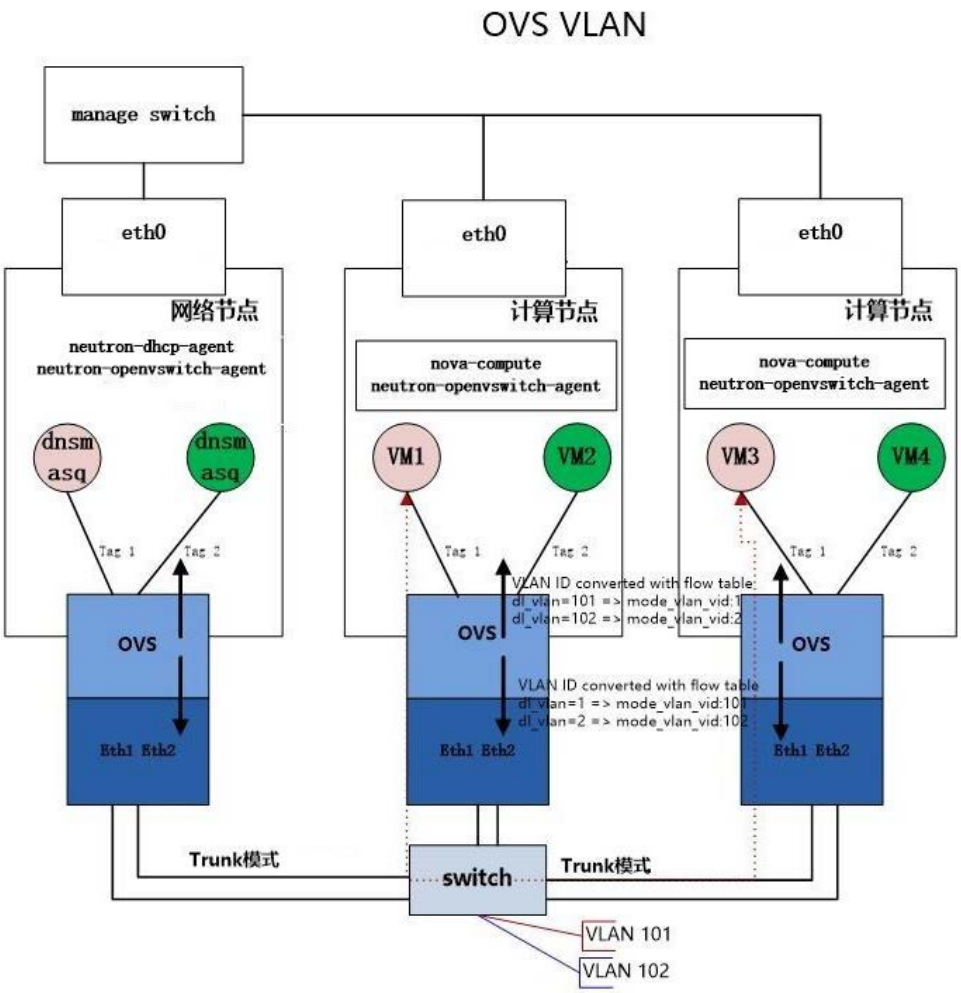
管理 6K 台计算节点集群，仅仅使用原生 OpenStack 是很困难的，因此我们对 OpenStack 依赖的数据库、MQ 等全部重新设计实现，设计经过测试可以支撑 1W 台计算节点。

2. 单实例性能调优：很多核心的业务对单个请求的响应时间有极其严格要求，这势必要求弹性云提供的每一个容器实例均要具有极好的性能，我们从 CPU 和网络两方面入手。

针对 CPU，我们采取 Scale-up 算法灵活调配 CPU 分配，使繁忙业务可以及时获得足够多 CPU 资源。

在网络这块，主攻方向为如何把万兆卡的性能发挥出来。我们对 OVS（Open

vSwitch）做了一些改进：减少了一些锁，优化 peer port 和网卡中断。这带来近乎物理机网卡的网络性能。



InfoQ：这么大规模的 Docker 应用，在国内外都是屈指可数的。在整个容器化的进程，你们得到了什么，失去了什么？

鲍永成：京东弹性计算云通过软件定义数据中心与大规模容器集群调度，实现海量计算资源的统一管理，并满足性能与效率方面的需求。

提升业务自助上线效率。应用部署密度大幅提升，资源使用率提升，节约大

量的硬件资源。

而面临的调整也很多，比如内核因为 bug 升级。全部容器化后，底层依赖系统的版本高度一致，很容易带来因为底层 bug 需要规模性升级，幸运的是我们在内核团队建设这块一直非常重视，已经有内核发布版本、内核热 patch 等方式，有效解决此类问题。

InfoQ: 在 CNUTCon 2016 全球容器技术大会上，你会重点分享哪些内容？

鲍永成：重点分享软件定义数据中心、大规模容器集群调度以及 Docker 容器性能调优。

鲍永成，京东弹性计算组项目负责人，带领弹性计算团队，深耕 IaaS 领域，致力于打造京东强大的虚拟化平台。2013 年初加入京东，重点在京东弹性云平台系统研发，运营多个中大规模 IaaS 集群，包括（京东弹性云、公有云、混合云等产品），在 OpenStack 研发 & 性能优化、自动化部署、KVM、Docker、分布式系统等方面有一定的实践经验。

京东 618：
三大系统防作弊，挑战直面用户的困难

作者 **韩婷**

京东的前端业务系统发展到今天，已经基本覆盖了京东交易环节的全流程。而今年的 618，除了流量上的考验，还增加了大量个性化数据、动态定价等诸多京东智能卖场所提出的新要求。那么，京东的前端是如何应对的呢？用到了哪些工具和技术呢？

InfoQ：晨总（京东集团 CTO 张晨）提到，今年 618 大促“PC 端大部分活动都是由京东大脑来支持的”，对于用户来说，很多数据都是个性化的。请您讲一下，对于首页来说，哪些数据是由京东大脑支持的，哪些是人工或半自动化设置的。对于不同类型的数据，处理的方式有何种不同？

尚鑫：首页对用户个性化概念是在 2014 年首页改版的时候提出的。随着近两年京东对大数据的运用以及京东大脑的成熟，今年 618 期间将由智能卖场为用户提供更加精准的品牌个性化推荐。

针对不同用户，在区域进行对首页各楼层下商品的分区域展示、广告的分区域投放、今日推荐、猜你喜欢，以及 618 期间的品牌盛典，都进行了针对用户粒度的个性化商品推荐以及活动推荐。

对于一些楼层活动入口以及频道列表入口目前多采用排期自动更换的半自动

化设置来满足业务人员对于首页运营的需求。

对于个性化数据的展示一般多采用前端异步实时调用的方式来获取个性化数据，做到了粒度为每个 PV 的实时商品运算。人工运营的部分以时间为单位由后端系统提前生成好展示给用户。

InfoQ：作为大型电商网站，京东的页面有很多种，如首页、单品页等。请您讲一下目前京东页面的种类，并举例说明针对不同页面的特点，分别采用了怎样的架构，遵循的原则是什么？

尚鑫：承载用户浏览、导购商品的页面主要为首页、频道页、列表页、单品页。它们各自的功能如下：

- 首页是京东的门面承载着整个京东所有业务体系以及重大活动（如 618）的流量入口。
- 频道页作为用户场景化、内容化导购的入口为用户展示商品以及活动。
- 列表搜索页则以类目和关键词为用户提供精准的单品入口导流。
- 单品页则作为用户购买商品前的最后一个商品展示环节为用户展示商品的所有信息细节。

从页面的流量特点上来说，首页和频道页的访问都较为集中，而列表搜索页和单品页则体现为长尾和分散。所以针对不同的流量和展示特点系统架构层面也存在着很大的差别。

首页系统作为门面对于页面的响应时间和首屏速度有着极高的要求，所以在进行设计的时候我们将首页进行了碎片化的处理，根据业务场景以及优先级将首页切分成了若干碎片并引入了 BigPipe 的概念来提升页面加载速度增强用户访问体验。对于可进行静态化的部分通过静态化的处理减少后端运算提升处理速度。页面上的 css/js、dns 预解析等方面也做了相应的优化来提升用户访问速度。

与首页流量特点形成鲜明对比的单品页访问分散并且长尾，在架构设计上考虑更多的是对于系统间依赖的解耦、对于差异较大的垂直业务的支持以及亿级商品数据的存储和展示。

所有的网站前台系统在设计上都会遵循数据闭环、动静分离、异步化、无状态、水平扩展几个原则进行设计保证系统的稳定性、扩展性以及对于业务瞬息万变的响应。

InfoQ：首页承载了大量二级页面的入口，那么首页的这些数据是采用怎样的方式加载的？采用了哪些技术来保证快速高效地处理这些数据？目前，京东的前后端是否已经完全分离？是否有中间层？

尚鑫：目前首页前后端已经完全分离，首页的数据维护以及各下级页面入口都是通过 CMS 系统后台来完成内容搭建维护的。各个业务系统数据更新通过消息的方式通知到中间层来实现数据更新、加工最后同步到前台展示层。像首页的非个性化数据一般是已经预先排期生成好的，所以除非遇到紧急的临时更新一般情况下首页的数据更新是非常高效实时的不会存在延迟的情况。

InfoQ：商品详情页的数据，尤其是价格方面的数据是非常重要的，对于数据的处理遵循哪些原则？采用了哪些技术来保证数据的准确性？另外，智能卖场提到此次 618 大促活动中会植入动态定价，前端是如何对此进行保障的？

尚鑫：商品详情页的数据来自于多个业务系统。数据的闭环、消息化、异步化三个原则是保证商品详情页稳定高效的基石。

每分钟几十万次的商品信息操作更新通过消息的方式通知到商品详情页系统再由系统将页面展示所需的数据异构到本地进行闭环处理最后展示给用户。

对于价格、库存这种重要实时性要求高而且敏感的数据采用异步的方式实时获取从而保证展示的准确性。

InfoQ：为了备战今年的 618 大促，京东针对 618 的压测方案有哪些？压测的数据是怎样设定的？具体采用了哪些技术？是如何进行压测的？

尚鑫：由于网站系统读大于写的特点，一般压测都是为了找到系统的读性能瓶颈。

之前采取的是通过 TCPCopy 的方式进行线上压测，但由于 TCPCopy 的压测方式具有一定的风险性，所以后来放弃了这种方式。现在采用的是从线上 nginx 日

志中离线采集访问请求然后处理成压测请求脚本后使用 jmeter 对目标机进行压测。压测一般是看正常情况下的系统访问承载能力的极限、异常请求下的系统容错能力和系统承载能力以及自我恢复能力。

压测的数据根据以往系统日常流量和峰值流量的访问来进行压测基数的设定，然后在此基础上进行成倍的增加请求，并通过性能监控系统辅助，来找到系统的性能拐点以及承载的极限点。而异常压测则是将系统依赖的服务进行干扰模拟接口服务不稳定或不可用的情况下系统容错能力和稳定性。

压测后根据两种场景的结果来对系统进行优化提升性能和对缺陷进行改造提升稳定性。

InfoQ：为了实时了解 618 大促的情况，前端是否有相应的监控平台？具体的功能是什么？采用了哪些技术实现的？

尚鑫：监控我们从两方面来做。服务端的监控，京东有统一的监控平台，可以通过它看到系统在服务端的性能表现。而对于在浏览器端展现的页面层面，我们内部有个前端监控系统，主要从以下两个角度对页面进行实时监控：

- 主动监控，针对页面上的一些异步服务进行延迟、错误主动信息上报，提前发现有问题的服务并采取降级等相关措施。
- 被动监控，通过抓取设定的样本页面并模拟浏览器的访问来看页面上dom元素内容加载情况、页面乱码、接口内容获取失败等问题并进行报警。这种和前端用户访问直接相关的风险点在后端是很难做监控的尤其在页面启用cdn后，前端页面的展示和后端完全分离开来这时候就需要一些被动抓取式的监控策略来保证出现。

目前我们对京东流量 Top10000 的页面及其各种临时的促销活动页面进行快速自动抓取。这类页面如果产生错误状态码（404,50x）、页面空白、跳错误页等，就会立即触发报警系统，后台主要使用 NodeJS 异步任务队列定时执行。而针对页面内容的检测通过系统的规则引擎对页面 dom 信息和预期规则进行匹配如果验证错误就会记录错误信息，保存现场快照，然后触发报警机制进行邮件或者短信

通知。

InfoQ：在 618 大促过程中，前端是如何对用户的数据进行统计分析的？使用了哪些工具？如何快速筛选出非正常用户并对其进行快速隔离？

尚鑫：我们将从用户可以直接感知的前端业务风控系统进行剖析。

交易订单风控系统

交易订单风控系统主要致力于控制下单环节的各种恶意行为。该系统根据用户注册手机，收货地址等基本信息结合当前下单行为、历史购买记录等多种维度，对机器刷单、人工批量下单以及异常大额订单等多种非正常订单进行实时判别并实施拦截。

目前该系统针对图书、日用百货、3C 产品、服饰家居等不同类型的商品制定了不同的识别规则，经过多轮的迭代优化，识别准确率已超过 99%。对于系统无法精准判别的嫌疑订单，系统会自动将他们推送到后台风控运营团队进行人工审核，运营团队将根据账户的历史订单信息并结合当前订单，判定是否为恶意订单。从系统自动识别到背后人工识别辅助，能够最大限度地保障订单交易的真实有效性。

爆品抢购风控系统

在京东这样的电商平台，每天都会有定期推出的秒杀商品，这些商品多数来自一线品牌商家在京东平台上进行产品首发或是爆品抢购，因此秒杀商品的价格会相对市场价格有很大的优惠力度。但这同时也给黄牛带来了巨大的利益诱惑，他们会采用批量机器注册账号，机器抢购软件等多种形式来抢购秒杀商品，数量有限的秒杀商品往往在一瞬间被一抢而空，一般消费者却很难享受到秒杀商品的实惠。针对这样的业务场景，秒杀风控系统这把利剑也就顺势而出。

在实际的秒杀场景中，其特点是瞬间流量巨大。即便如此，“爆品抢购风控系统”这把利剑对这种高并发、高流量的机器抢购行为显示出无穷的威力。目前，京东的集群运算能力能够到达每分钟上亿次并发请求处理和毫秒级实时计算的识别引擎能力，在秒杀行为中，可以阻拦 98% 以上的黄牛生成订单，最大限度地为

京东的正常用户提供公平的抢购机会。

商家反刷单系统

随着电商行业的不断发展，很多不轨商家尝试采用刷单、刷评价的方式来提升自己的搜索排名进而提高自家的商品销量。随着第三方卖家平台在京东的引入，一些商家也试图钻这个空子，京东很早就提出了对此类行为的“零容忍”原则，为了达到这个目标，商家反刷单系统也就应运而生。

商家反刷单系统利用京东自建的大数据平台，从订单、商品、用户、物流等多个维度进行分析，分别计算每个维度下面的不同特征值。通过发现商品的历史价格和订单实际价格的差异、商品 SKU 销量异常、物流配送异常、评价异常、用户购买品类异常等上百个特性，结合贝叶斯学习、数据挖掘、神经网络等多种智能算法进行精准定位。而被系统识别到的疑似刷单行为，系统会通过后台离线算法，结合订单和用户的信息调用存储在大数据集市中的数据进行离线的深度挖掘和计算，继续进行识别，让其无所遁形。而对于这些被识别到的刷单行为，商家反刷单系统将直接把关联商家信息告知运营方做出严厉惩罚，以保证京东消费者良好的用户体验。

前端业务系统发展到今天，已经基本覆盖了京东交易环节的全流程，从各个维度打击各种侵害消费者利益的恶意行为，几年时间，成绩斐然。

InfoQ：为了保障 618 大促的顺利进行，前端做了哪些预案？在何种情况下回启动这些预案？启动的方式是怎样的？

尚鑫：由于前端系统调用了众多业务系统的接口服务，所以我们的预案大多会集中在对于前端调用接口的治理上。

我们开发了前端使用的“三峡系统”来为前端系统提供接口的监控、降级、切换一站式服务。“三峡”，顾名思义就是作为一个总闸来控制所有前端系统对接口的调用和内容展示。从接口的性能、可用性、数据正确性几个方面进行监控，并提供便捷的降级和切换功能。这样可以保证一旦监控发现依赖的接口服务性能或可用性下降，我们就可以一键式从后台对接口服务进行降级、切换处理保证前

台页面展示。

InfoQ：京东的前端团队是如何分工的？在 618 大促期间是如何保障团队内部的高效沟通与合作的？与其他部门又是如何协作的？

尚鑫：目前京东前端团队主要负责京东商城的网站前台系统建设。团队主要根据业务线进行垂直划分。但团队的内部沟通并不会影响团队的沟通效率和合作。用户的每一次访问、每一个点击，都会将每个团队串在了一起。

我们的内部沟通不拘泥于任何形式，只要有问题出现大家就会聚在一起进行讨论，并形成一个可执行的方案进行推进。

对于外部，由于我们是处在系统链的最前端，所以我们会建立主动沟通机制，和各个团队进行随时随地的沟通，来预防和解决任何可能出现影响用户体验的问题。同时，由于我们部门所负责的各个系统全部面向客户，有任何的小问题都会严重影响到客户体验。因此，我们也建立了问题快速响应的专职团队，负责及时

尚鑫，京东商城网站移动研发部负责人。2007 年加入京东，见证了京东 IT 系统演进的全过程，先后负责京东多个核心系统，熟悉大流量高并发系统架构设计。

京东 618：
从演习、监控到预案，京东无线全面备战

作者 **韩婷**

在京东上季度的财报中，无线端（包括移动端和微信等其他无线平台）占比已经超过 72%，这也给京东无线业务部带来了巨大的压力。今年，京东 618 主会场首次全面采用个性化策略，同时，618 期间的一系列促销活动，预计将为后端带来超出日常 20 倍左右的流量洪峰，这都给无线业务部带来了更大的挑战。为了迎接挑战，防止突发情况的发生，无线技术团队从演习、监控到预案，制定了全方位的备战计划。

InfoQ：为了备战今年的 618 大促，京东做了很多演习。请具体介绍一下移动端演习的内容、分类和流程，以及演习时的模拟数据是怎样设定的。

赵云霄：大促前我们都会有一系列的演习，一般情况下，无线端会在大促前至少半个月开始演习，具体的时间由演习的不同性质来决定。当然我们的演习不可能只是一轮，一般是多轮的演习，具体几轮视情况而定。

做演习的目的，一方面是测试系统所能承受流量的能力，另一方面是，测试在某些环节出现问题时，我们切换预案的执行效率如何。这两方面同时也是我们在做演习时的主要内容，所以演习相应地分为压力测试和预案切换两类。

压力测试是基于工具的，用来测试我们对流量的承受能力，包括单机和系统整个集群的承受能力。压力测试一般分为两种，一种是工具测试，一种是线上流

解决我们发起线上的流量引到其他体验的数的降低到最测试单机的性能。

预案切换类的测试，就是对预案进行演练。预案演练是指，有一些预案虽然已经定好了在出现问题怎样去执行，但在现实的执行过程中仍有可能会出问题，所以我们需要提前去对预案进行演练。这类演习一般都是提前一个月或者一个多月先做一轮，先暴露出一些问题，然后根据暴露的问题来分析是否有必要再做一轮。如果再做了一轮还有问题，那么我们还会调整，再演习。但是一般情况下，不超过三轮我们就会发现很多问题，包括大数据要解决的问题。

演习的数据有两种，一种是线上流量，一种是造数。

- 线上流量，一般就是使用线上的数据，例如将一个集群中的十台机器摘掉九台，看看一台能不能支撑，以此来测试单机的性能。
- 造数是指由我们专门的测试团队去造测试数据。但是这样一来，一些写操作，会引入一脏些数据。在造测试数据时，我们会尽量把SKU分散开，预估热点SKU的数量，来做更真实的模拟。同时，我们会在压测的请求URL中增加一个标识，来区分正常业务和压测数据，以便后期清洗这些脏数据。

陈保安：我重点从下单之前和下单之后这两个环节说一下模拟数据。

一是下单之前的模拟数据。我们通过测试，把线上的日志数据抓下来之后做回放，尽可能模拟真实用户的请求。然后将它放大几倍或者几十倍去做压测。这类数据所进行的操作分为两种，一种是读，另一种就是刚才提到的写操作。读操作的数据可以无限放大，因为这种东西不会对数据产生干扰。而写操作的数据需要在最后清理出来，以免对真实数据造成影响。因此会设置标识来便于后期清洗脏数据。

二是下单之后的环节的模拟数据。在有些场景下，比如在我们整个京东的“军演”过程中，使用的是线上的数据。它模拟的是我们下单之后，后端所有的支付、配送等所有的系统的压力。我们会在“军演”时“憋单”，使得管道里面存 20 万单或者更多，然后一下把订单下传下去，模拟出给整个后端环节的压力。

InfoQ：在 618 的期间的数据跟平时可能会有些不同，例如某些热销产品，数据变化较大，针对这种情况是不是会做一些特殊的处理？

陈保安：是的。618 期间有一些数据会过热，尤其是一些促销力度过大的商品相关的数据。我们在演习的时也会去模拟这种数据。例如，拿线上 50 万或 300 万的数据去进行模拟时，我们会将其中 50 到 100 个商品的流量放得更大一些，来模拟部分商品热度较高的情况。我们还会测试在过热的情况下，例如一个商品受到了极限的访问，甚至达到整体的访问量那么大以后，我们的系统会不会有更好的处理。

赵云霄：因为过热的商品会造成访问数据集中在某一片缓存上，这有可能会暴露出系统设计上的问题。我们会通过技术的手段解决这种问题，然后将测试数据集中在上面，来看测试效果。

InfoQ：那方便说一下具体采用了哪些技术手段吗？

陈保安：我们会对热点数据做更多的本地化的缓存。例如单品页，它整个核心流程访问量最大，所以我们对单品页设有多级缓存来扛单品页的访问。通过 CDN 来扛外网流量的访问，主要适用 APP 的访问，另外 nginx 的共享缓存也会增加热点数据的缓存，因为有一些内网的调用不走 CDN，第三方面在 java 进程堆内存中增加热点缓存，这一层的缓存时间较长一些，透传过来的流量会对上游基础服务做一个保护，所以一级一级这么去保护，避免某些爆款商品拖垮整体服务。

赵云霄：总的来说就是，设置多级缓存，同时尽量让第三方缓存的请求变得更散。

InfoQ：网络的复杂情况给移动端带来了很多问题。为了保障用户的最佳体验，京东做了很多适配工作，并在接口层面做了做了相应的设计。请您具体讲解一下适配的标准，以及接口的设计根据 618 大促的需求做了哪些改变。

陈保安：因为网络的复杂情况而带来的这些问题主要是由于手机端与 PC 有着不一样的特性。

首先，从网络流量和耗电量来看，图片加载对手机的消耗比较大。而从进到

首页到后端搜索，再到商品详情，我们展现给用户更多的就是图片。所以，我们对图片会重点处理：根据不同的分辨率、不同的网络类型、不同的手机系统版本（安卓 4.X 版本和之前的版本区别较大），指定了相应的标准。这里是有一个比较大的 escape 标准东西，就是说有一个成熟的表单，它是由产品人员做了很多次测试得出的，在什么样的位置上，图片适配采用什么样的大小尺寸都有标准。

第二是，PC 端关于价格、商家信息、状态跟踪，包括一些库存有一些信息，很多都是异步加载出来的。PC 端的页面呈现给用户的信息可以在第一时间加载出来。然而手机端不一样。因为网络环境比较复杂，有可能这一秒有网，下一秒走到角落里可能就变成了弱网，甚至没有网了。这种情况下去创建更多的连接，成本是比较高的。而京东的单品页因为垂直的频道比较多，所以会更加复杂，虽然仅仅是一个单品页，但是它聚合了我们后端近 50 个基础服务。针对手机端，我们不会在单品页把数据异步下发，而是由后端聚合之后一次下发给客户端，这样就减少了建立链接的成本。

这一次 618 我们做的比较大的改动，就是规则的识别下发给了客户端。比如，刚刚提到的图片的识别规则原来是在后端去处理的，要根据手机的分辨率、网络类型、机器的版本去做适配，这样一来就涉及到静态化的数据，使得利用率变得非常非常低了。所以这一次我把这规则下放给了客户端，客户端只需要告诉后端是安卓的哪个版本，后台会给它下发统一标准的数据。而规则是动态下发给客户端的，比如说打开客户端以后，后端就把规则下发给了客户端。这样一来，后端就不需要再去根据分辨率、网络类型等去做适配，而是由客户端去做适配工作。

具体的下发的规则是统一的，包括在 2G、3G 等网络情况下分别对图片有怎样的要求。这一点是根据分辨率来固定相应的范围的，不是所有的范围，否则规则会太多。

赵云霄：总的来说是三点：

- 根据图片不同的位置，采取不同的适配标准。例如单品页商品展示的图片会比较高清。

- Sever端的接口设计异于PC端的ajax异步加载。尽可能通过后端把数据聚合下发，减少客户端和后端频繁创建链接的过程。
- 单品页图片适配规则可动态变化。这是无线端针对618大促做的比较大的改动，单品页图片适配规则下发到客户端来执行，规则是动态可变化的。这使得后端不需要依赖手机的网络类型、系统版本等信息，可以极大地提高后端的Cache命中率。

InfoQ：京东曾提到，在系统升级的过程中，会尽量减少强依赖，将强依赖尽量转化成弱依赖，并不是直接依赖于服务。能否举例说明减少了哪些强依赖，是如何将强依赖尽量转化成弱依赖的。

陈保安：强依赖转变成这弱依赖，或者说尽量不依赖，就是说我们对这于实时性要求不高的，像商品的信息，包括图片、特殊属性、详情、商家等信息，以前可能是完全依赖于基础服务的，现在会把它的一些数据异构并存储下来。在发生变化时，通过管道 MQ 去通知做出一些变更，达到最终不会实质依赖基础服务去做商品聚合展示的目的。所以现在，其实是间接依赖。因为这虽然不是实时，但是是通过异步的方式给我的。而像价格、库存等变化非常频繁的数据，依旧是实时依赖。

还有比如说在下单之后，或者是在支付环节，会涉及到DB。而DB相关的数据，我们现在也是通过异步的方式落地的。因为这个说白了实质性要求没那么高，无线收银台可以不完全依赖 DB，更多地依赖实时用户会话的缓存来于第三方支付机构进行交互。这也是因为在去年双十一的过程中，DB 遇到了一些麻烦，主要原因是连接数比较多，所以这从强依赖变成了弱依赖。

赵云霄：其实像京东的系统，是一个完备的包括所有业务的系统，是一个完全的服务化的系统。哪些是必须看成强依赖，哪些必须看出弱依赖，在我看来这个东西没法去区分。

京东有几百上千个服务，我们要去梳理这些东西，比如说哪些东西没有必要直接同步的去调用它，让它影响我们，我们会去把这些东西梳理出来。现在的套路，

要不是异步化，要不就像保安说的有一些地方我可以用空间去换取更好的交互，所以要么是采用中间件，要么是异构一份数据到缓存中。那么所谓减少依赖都是在正常服务不可用的极端情况下可以去考虑的，在正常的服务化系统之间的交互，我们还是尽量保证专门的系统去做其专门的事。例如，在平时，我们会去依赖A、依赖B、依赖C，而且还会调度它们，但是我们还会有一个异步方案，在特殊情况下，比如说在流量非常大的时候，去切换到这个方案上，这个方案对后端压力比较小。

所以我认为没有必要刻意总结哪些强依赖、哪些是弱依赖。在一个比较庞大复杂的服务系统中，你没有太多办法要求别人的系统去做一个什么什么事，而我们的系统也要配合别人的系统去工作，所以这个复杂就会更高。所以我们只能想办法，各自做一些在极端情况下能保护后端的方式，这就是我们减少依赖的目的，也是遵循了前端保护后端的通用原则。

InfoQ: 那么也就是说在618期间流量比较大时，可能会做出一些取舍，比如用空间去换取交互效果。那么在大促结束之后，会不会换一套方案把之前做的牺牲再换回来？

赵云霄：这是有可能的。因为在严谨的系统设计中都会有备选，在特殊情况下必须用空间去换取想要的效果，但是日常的流量情况不需要牺牲额外的空间。既要节省成本，又要保证正常的交互。另外，就像刚才提到的很多弱依赖的场景其实是个别情况，在大部分交易流程中，很多依赖还是实时的。把需要实时反馈的数据转换成弱依赖，实际上是我们做出的妥协。所有的这种妥协，在回归正常情况时，我们都有可能收回。

陈保安：做出妥协是非常必要的，领导曾说，我们虽然要依赖一些基础服务，但是基础服务真的挂了的时候，自己还要保证能活，这就是我们的最终目标，就是说要在基础服务出现问题的时候，我们能够有办法快速地去解决问题，而不是处于死等状态。

赵云霄：任何一个大型在线系统，它的主干要是最强壮的。我们可以减少分支上无关紧要的东西，来保护最重要的部分免受大流量的破坏，让它处于安全的

保护之内，能提供正常服务。同时，由于用户体验非常重要，又涉及到京东很多用户，所有我们会多考虑出 20%，比如考虑流量达到 120% 的时候要做出什么方案来保护它。

InfoQ: 这些预案都是来处理特殊情况的，而在实际的大促过程中，为了保证万无一失，京东还做了很多监控工具和监控平台，那么京东都是从哪些纬度来进行监控的？移动端的监控平台有哪些？具体采用了什么技术？

赵云霄：事实上，在刚起步时，我们也缺少监控，经常半夜被叫醒处理技术问题。后来我们逐渐意识到，监控系统跟核心系统是同等重要，应当把监控系统看做正常业务系统的一部分。比如说，要新上一个业务系统，那么监控系统就占 50%，没有监控系统就不可以上线。

从无线端来看，监控系统是多维度的。从底层的操作系统的各项指标，到各个层面的业务系统，再到网络都需要监控。

这些监控都是非常必要的，比如业务监控，有时候业务表面上是存活的，但是业务不正常，有可能出现很多逻辑上的问题，所以移动端还有对订单量、购物车、搜索等方面的监控，这些监控是业务指标级别的，也是应用层面的。我们有对应用主动调用的检测，不同的 VIP 进来，这就是从应用层面的存活激活，还有一些其他。各个团队如果感觉这应用不够，还会自己添加一些小的应用系统跑在机器上去监控。

另外还有对运维、云平台和网络层面的监控，这些监控一起形成了一个比较立体的监控系统，从底层到最上层都有监控。

目前，移动端涉及到的监控平台有以下几个：

- 核心业务监控（cpm平台）。主要有三个核心监控模块，依赖基础服务的返回码错误监控，依赖基础服务的异常监控和收银台全方位（机房、渠道、接入方、银行等维度）的数据监控。
- monitor.m.jd.com：承担无线运维主要的监控入口，包括一些核心业务指标以及服务器基础监控等。分钟级别的监控，第一时间对业务以及可

用率抖动进行告警。

- 网关、httpdns探测系统——prism.m.jd.com：主要承担网关、httpdns的服务器存活监控和网关连接池实时监控。监控系统每分钟探测自动化部署中这两个应用的所有服务器状态并以图形化的界面展示出来，只要连续两次探测不存活就会报警，如果没有及时修复，会持续报警，直到正常。实时监控网关连接池，根据它可随时调整连接池参数。
- 爱马仕监控平台——hermes.m.jd.com，承担网关全量接口的可用率、性能、调用次数监控及短信报警。同时，提供机房内外网、服务器IP、省份、运营商、客户端版本等各种维度按需组合的监控。
- ZKCloud——zk.m.jd.com,为接入ZooKeeper用户提供可视化操作与详细操作日志，提供zk节点存活监控、各项指标性能监控，报警维度有短信、邮件、咚咚。

建设监控系统也有其相应的监控原则，例如监控一定要报警，允许偶尔误报，但不能不报。报警是多通道的，报警方式可以是短信、微信、邮件等。而且，报警至少要有两个人以上的接受人。

所用的监控技术，业内用的都差不多。我们主要就是埋点数据的升级，比如说在最基本的操作系统层面的监控会去采集利用相关文件，查看它的CPU、内存，在应用层面也会埋很多的点去上报这些数据。在获得数据后，有专门的团队来进行实时的数据分析，用最快的速度分析出结果，需要报警的话再走报警通道。对于重要的URL或者重要的请求，有探索工具来适配我们所希望的结果，如果探索不到这些结果，那么系统可能就有问题。总的来说，所用的技术就是数据实时分析和主动探测。

InfoQ:刚才提到，京东的监控做得非常立体，有很多团队去做。那么有没有统一的管理，或者说一些原则？有没有专门的人去对监控系统做审查?

赵云霄：监控点的加入是渗透在系统的设计中的。例如，系统的初期就应该对业务指标的数据进行监控，这是一个上线之前的规则。我们会定期做考察，如果有些该设置埋点的地方没有设，那么QA人员会做处理。

埋点的原则主要有以下四点：

- 资源调用处（接口、缓存、DB）必须埋点
- 埋点必须可动态关闭（通过配置下发）
- 数据收集类的埋点必须通过多层的代码review
- 埋点数据收集不能影响系统性能，对埋点的SDK要有性能测试报告才可以使用

InfoQ: 也就是说开发人员也要负责运维人员的监控环节吗？开发人员需要跟踪一段时间，来交接监控系统吗？

赵云霄：是的，而且我们的运维有研发能力，这是对互联网研发人员一个最基本的要求。

陈保安：各个团队做的职责不一样。比如说各个机房的流量分配运维更偏向于网络层、包括服务器层这些基础组件的监控。网关层面更注重的是流量的监控，所以业务团队除了一些基本业务指标监控之外，也会做一些业务细节的监控，比如可用率、性能、异常量、错误码等等。比如说某个点的单量突然跌了，需要尽快定位到具体是什么原因引起，所以业务侧的监控更多是能一眼就能聚焦到引起问题的某个点上，通过预案快速修复问题。

而且我们上线、发布新的业务，都是要有灰度的。比如上线一个业务之后，只是其中的一部分流量走新业务。此时，我们通过埋点监控，去对比新老业务，看用户的反应，分析其影响，然后再做决策。

InfoQ: 刚才提到埋点，那么大促过程中移动端设置的埋点跟平时有没有不同？大促结束后会不会关闭某些埋点？

赵云霄：埋点的作用分两种，一种是用来收集业务数据的埋点，另一种是监测系统指标的埋点。大促之前，我们会针对不同促销会增加一些的收集业务数据的埋点。京东的业务非常复杂，跟单纯做平台其实是不一样的。市场有时会做一些新业务，这时就可能要根据业务再判断如何增加埋点。用来监测系统指标的这些埋点基本不会变。

埋点基本上都是可以去关闭的。埋点要么是通过现在的技术，通过 APP 内网去申报，要么就是写在磁盘日志上。这两种埋点对网卡和磁盘都会造成比较大的损耗，所以没有用的埋点，我们会想办法动态关闭。

InfoQ: 京东做了一套立体的监控，又设置了很多埋点，那么肯定是可以监控到非正常用户的。具体是怎样筛选出非正常用户的？移动端在管控这些非正常用户方面做了哪些工作？怎样保证快速隔离非正常的用户？

陈保安：对于非正常用户，我们会对接到京东集团的风控系统。风控系统利用对用户的积累，对用户做了很细致的分类，可以根据用户的行为采用模型算法对其进行识别。例如，有的用户每天就只有刷单操作，没有搜索等行为；有的虽然模拟用户进行多步操作再下单，但是下单的频率比较频繁，一秒钟下单很多次；还有些用户下单的商品全是无货商品，也就是说在活动开始之前，脚本已经启动了，而这段时间商品是无货的。利用最简单的这些特征，就能分析出来这些用户是非正常用户。

同时，在安全方面客户端和服务端之间也有一些基本的接口调动时候的安全限制。比如说签名，我会给你分配到签名，你要拿到对应的签名才能去访问相应的服务，这里也会有几种策略下发给客户端去用。但是非正常用户为什么可以刷单，就是因为破解了个别签名。这也是为什么还需要后端不断地去对用户行为分析。

具体的隔离方式：针对非正常用户，在关键时刻我们会做非常严格的限制，否则会对系统和商品销量造成很大的冲击。例如，在 618 和双十一零点时，会对识别出的用户做特殊处理。表面上会给他们提供一些服务，但实际上会把他们引流到一个具体的集群中，性能、稳定化不做保障，如果他们挂掉了，我们也不负责，这样才可以对正常用户做到保护。而即使是引流，他们最终访问的流量还是会透传到各个基础服务，在特殊紧急情况下，也会把这些用户掐掉。这些不同级别的限流都已经经过了演习并报备，目的都是为了保证正常用户。

InfoQ: 相当于把非正常用户加入黑名单，那么这份黑名单是怎么得到的呢？

赵云霄：得到这份黑名单非常不容易。京东有非常多的数据，也希望能利用这些数据对用户提供更好的服务。京东有一个专门的团队，利用大数据分析，通过数据模型不断地调整、优化，再利用从前端到后端的多级防控，才从海量的数据中分析得出用户的画像。无线团队也会做一些比较简单的用户分析，也会要求他们为我们提出安全点。大家始终在做这种配合，然后根据用户的不同级别，做一些反馈。比如，像刚才提到的，在特殊情况下掐掉的非正常用户，都是我们真正有把握的非常用户，而不是模棱两可的。正常用户去抢购商品是不会被加入黑名单的，这样才能真正包含用户的利益。

同时，我们队黑名单也会做出动态调整。因为不同业务的业务模型是不一样的，对用户的分类是有差别的。所以需要跟专门的团队去沟通，让他们根据业务模型分析非正常用户。实际上，非正常用户的占比也是非常小的，这也让我们非常欣慰。

陈保安：这模型建立固定好后，通过埋点实时地埋过去，就可以算出真正的非正常用户。在大促时，经常遇到有人临时去注册一批用户的情况，但是只要他开始刷单，我们很快就能够识别出来。

InfoQ: 据我所知，无线端对整个购物流程都做了非常细致的预案，能不能具体地说一下细致到了什么程度？开启的方式是自动化吗？在什么样的情况下会启动预案？

陈保安：整个京东针对 618 做了上千种预案，而无线部门也做了几百种预案，这个力度是都非常非常细的。大的预案例如机房的切换，而小的预案也有很多。

比如，京东 App 上搜索现在 70% 搜索的结果都是个性化的，这样一来缓存命中率会降低，服务器的压力就会增大。如果 CPU 使用率达到了 40%，那么我们会通过降级个性化服务来保证用户能够快速访问。再比如现在支付环节要依赖京东台账和订单中心，如果服务出问题，我要通过降级保障用户能正常支付。

开启的方式有自动化的，也有人工的。比如 httpDNS 服务出现问题，客户端会自动降级到普通的 DNS 服务解析，这个影响范围比较大，那么就不能等人工去

切换，要使用自动化方式。

关于何时启动预案，虽然往年我们为了保证最基本的购物体验，我们在零点时会做一些处理降级。但今年晨总要求了两点，一是技术保障，最基本的要求，二是业务驱动指标，就是说不能随便降级，要保障业务。所以，包括之前提到的个性化，以往在零点去做一些降级，但是今年不会去做降级。但是一切大前提就是保障基础购物流程的体验，关键时候系统自我的保护就会启用一些预案，比如说单品页下面的个性推荐，只要它达到基础性能的某个比例后，我们就会采取一些紧急措施来保证整体商品的展示。

赵云霄：事实上每年都会做很多预案，在大促前都会有一个严格的评审。细致到什么程度呢？比如后端依赖某一个具体的接口，针对每一个接口都要有一个预案，包括资源的依赖都会有原。我们是尽可能地去考虑可能出现的问题。

预案启动的方式是自动还是人工需要看情况。

- 系统压力过大，CPU达到了40%，有这样的具体标准，那么可以采用自动启动。在自动启动预案时，都会向相关人员发送通知。
- 大促期间，有些情况比较复杂，是多种原因造成的，这时候就需要优秀的技术人员去处理，那么此时就需要人工启动。相应级别的人会有判断和执行的权利，但是在事前和事后都要周知一下。预案的执行有流程，也有反馈，这我们预案执行的大原则。

有了之前提到的强大监控系统来提供一些判断依据，预案的执行效果还是比较好的。

事实上，预案有专门的分归类，不同模块的人会去管理自己的预案。在执行时也会遵循保证用户正常体验的大原则。

InfoQ: 京东的无线端一步步发展至今，两位想必有很多感慨吧？

陈保安：对，我跟云霄就是从无线成立到现在，一步一步走过来的。在四五年前，无线只是试水，那时我们还叫手机移动端。上了一万单时，大家很开心，还专门庆祝了一番。现在，我们的访问人次和订单量在当时都是难以想象的，比

如，过去一个月的日均活跃用户数已经接近 4000 万，技术团队由最初的两三个人发展到现在的过百人，有了比较完善的架构。

赵云霄：我们整个技术团队也是随着流量的不断增加而慢慢成长起来的。开始，我们会参照别人的技术，羡慕别人的技术。后来，我们有了很多问题要去解决，在这个过程中也积累了很多经验。现在，我们不需要再去参照别人了，而是要将自己的经验分享给大家。因为我们发现，很多人一路走来，面临的问题都是相似的，只是处于不同的阶段。

京东给了移动端快速成长的机会，让我们不断进步。我们也一直保持着对技术的旺盛求知。网上有一些技术贴，说京东的技术如何如何，我们没人去霹谣，因为谣言太多了。你可以说京东只是流量很大，但我会说京东的系统复杂度是最大的。我们经历了这么一个体系的成长，现在也有了一个足可以让我们自己去仰望的平台。

我记得我在 QCon 做分享的时候，还有人跟我说，我是第一个京东无线部门出来分享的人，当时徐总也说这是五年磨一剑。我希望我们无线部门是京东一个非常好的代表，希望每一次分享都是一个非常经典的干货，而不是同一个主题在各个大会上去讲。

赵云霄，京东商城无线业务部首席架构师。2011年加入京东进入无线团队，负责服务端的开发与架构设计工作。见证并参与了京东无线从小到大，由弱到强的整个过程。五年间，带领团队完成了无线服务端的两次重大架构升级，无线服务端从一个简单的Web应用进化成为支撑每天几十亿级访问的分布式系统。目前负责京东无线网关系统的研发工作。

陈保安，京东商城无线业务部研发高级经理。从事研发工作十年，2011年加入京东。目前主要负责无线核心业务的后端研发工作，多次带领团队进行系统架构升级和多次大促经历磨练，为京东无线核心系统后端服务稳定性打下坚实基础，支撑无线千万级的活跃用户和京东过半的订单占比。

京东 618：
多中心交易平台系统高压下的高可用性

作者 **穆寰**

对于京东来说，每年的618大促都是一年中最重要也是挑战最大的技术考试。去年 618 京东平台单日成交订单量逾千万，同比增长超一倍。京东推测今年下单量将会继续成倍增长，并为此加强了系统处理能力。这千万数量级的成交订单背后，一定是需要够硬的技术支撑。在过去的一年，京东都做了哪些技术突破？如何应对本次 618 大促？ InfoQ 记者采访了京东资深构架师吴博，分享京东自主研发的多中心交易平台系统。

InfoQ：本次 618，对于交易平台，您认为最大的挑战是什么？为了应对这样的挑战，京东都做了哪些准备？

吴博：本次 618 对于交易平台，最大挑战仍然是如何保障大流量下的系统高可用性。为了保障高可用性，618 前我们上线了多中心交易平台系统二期，保证应用系统和数据库的高性能以及多机房多活。而且，今年 5 月份还上线了诺亚方舟二期，全部的应用系统都运行在万兆网络环境的容器上，保证流量超预期时，系统能快速水平扩展。通过各系统配合线上压测不断调整和优化性能，在保证合理的延时指标时，最大化提高系统吞吐能力，有效利用每个容器的资源。

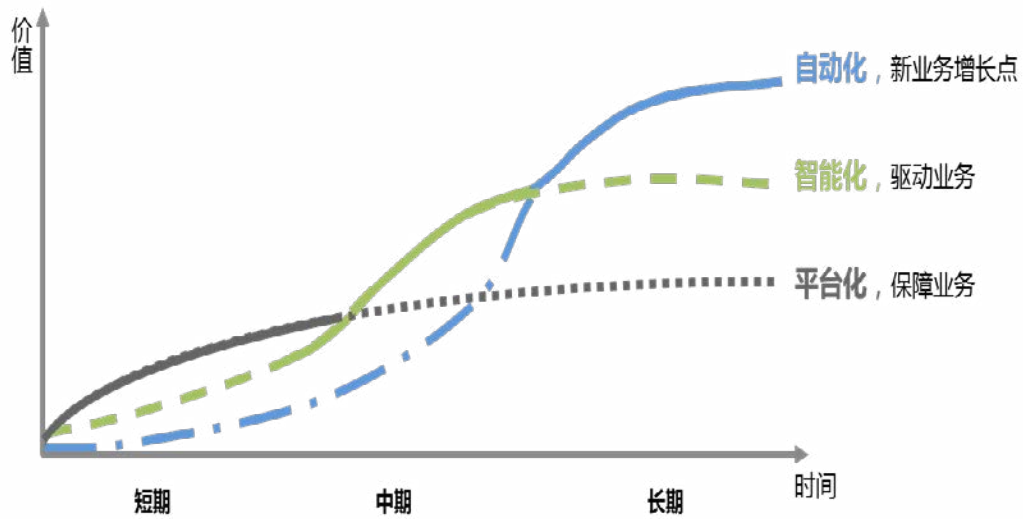
为了进一步提升交易平台的高可用性，我们针对 REST、RPC 两类服务在

nginx、JSF 框架层做了限流保护措施。对于 JSF，实现了基于频次和 CPU 利用率阈值的熔断机制。

对于核心的交易流程链路上的各个服务，交易平台统一了原来分散在各处的流量可视化和故障切换平台，以便提高故障定位能力和切换速度。

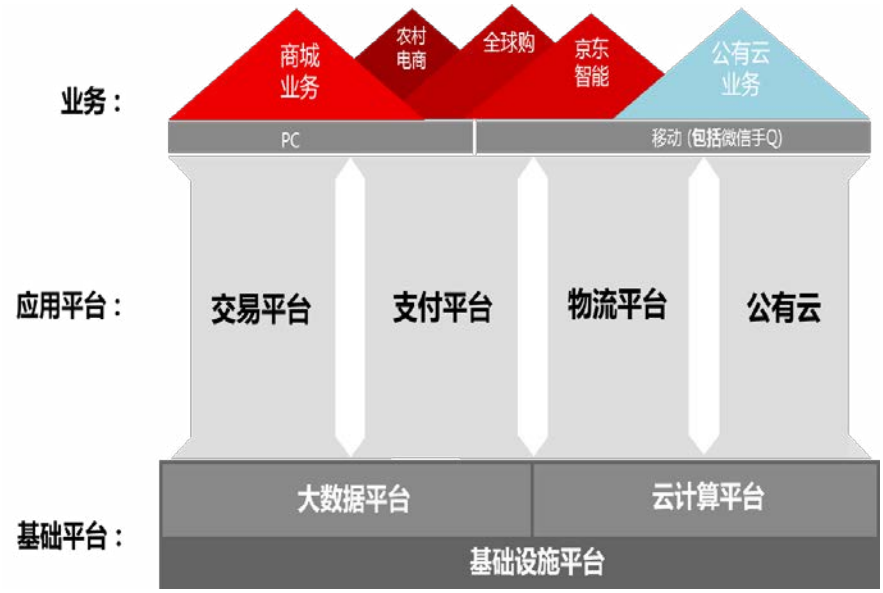
InfoQ：您认为在本次 618 准备过程中，京东主要的技术突破、亮点和创新都有什么？

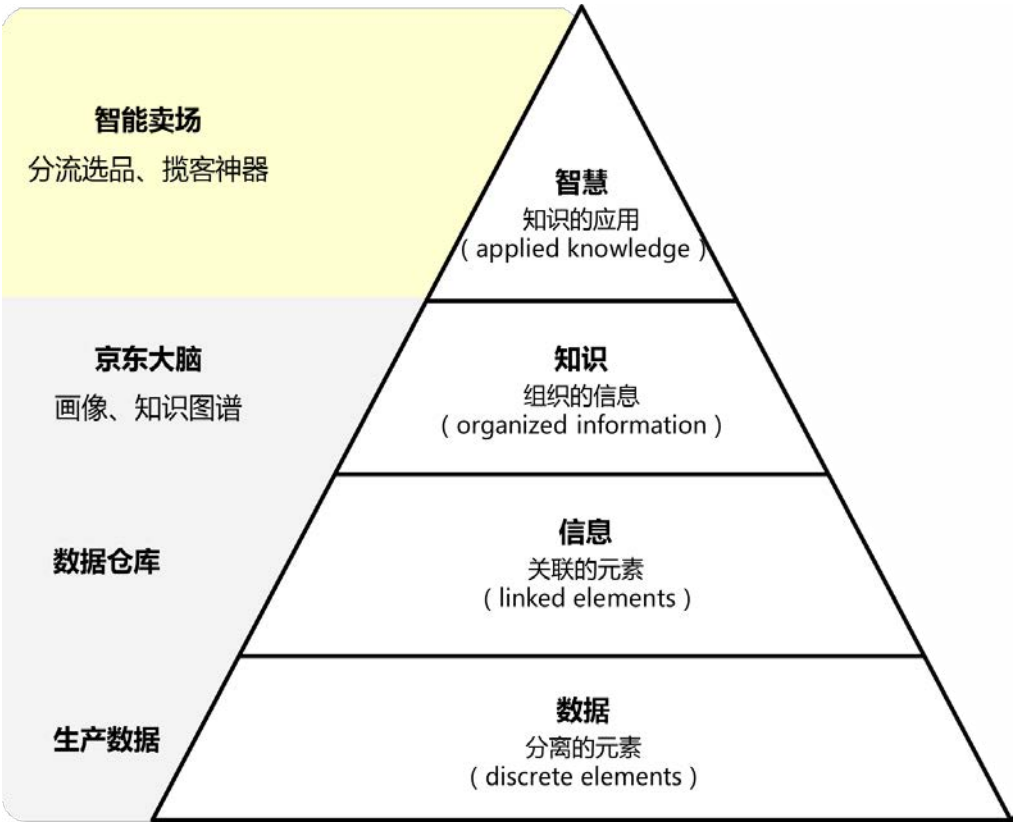
吴博：本次 618 准备，重点在 3 个方向：推进平台化、开展智能化、尝试自动化。平台化目的是保障京东业务变化灵活多样，提升核心系统高可用性；智能化目的是实现技术驱动业务的发展，比如利用大数据分析，指导应用系统和业务流程的优化；自动化是在智能化的基础上，尝试让机器做决策，减少人工参与，提高人效时效。



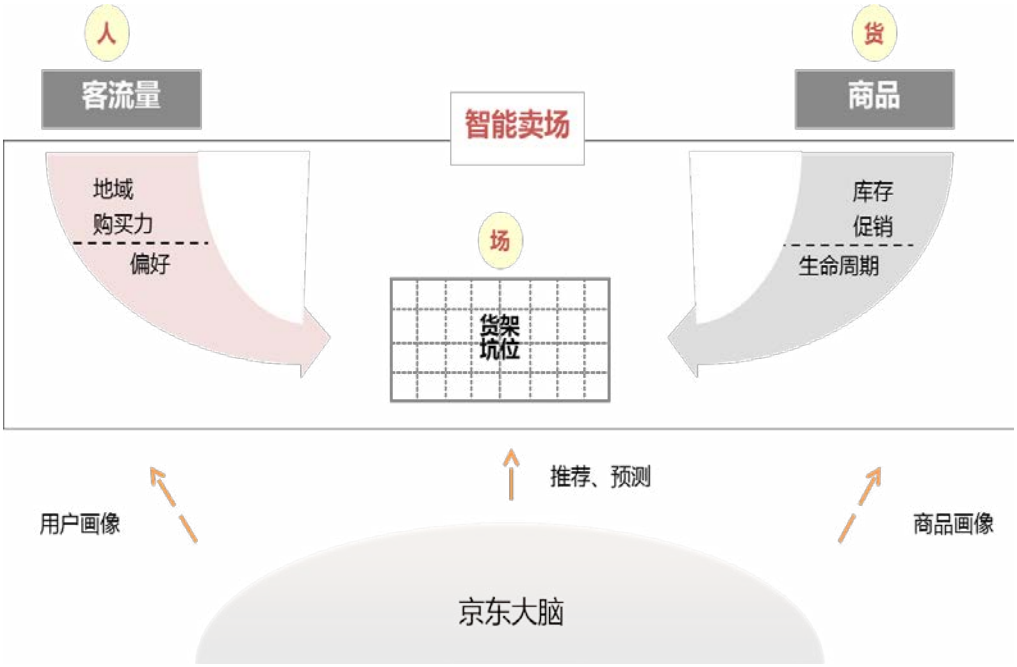
平台化方向：京东在这次 618 前，完成了多中心交易系统 2 期，保障应用系统和数据多机房多活；顺利实施了诺亚方舟计划，完成 2 个大型数据中心的建设，京东的全部应用系统和部分数据库系统部署在弹性云上，实现资源动态按需分配；深化了仓储平台化，搭建 EDI 平台，理顺与厂商信息交换壁垒。

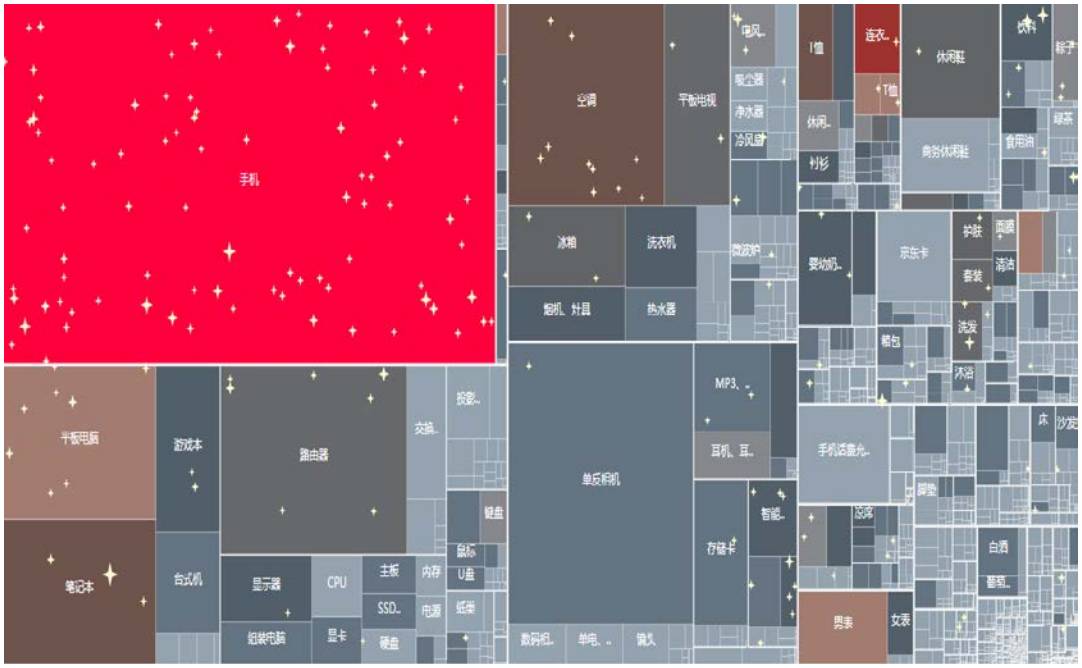
智能化方向：智能化是这次 618 的一个亮点，重点是用大数据指挥 618 大促，基于京东大脑的画像和知识图谱，搭建智能卖场，让大促更智能。





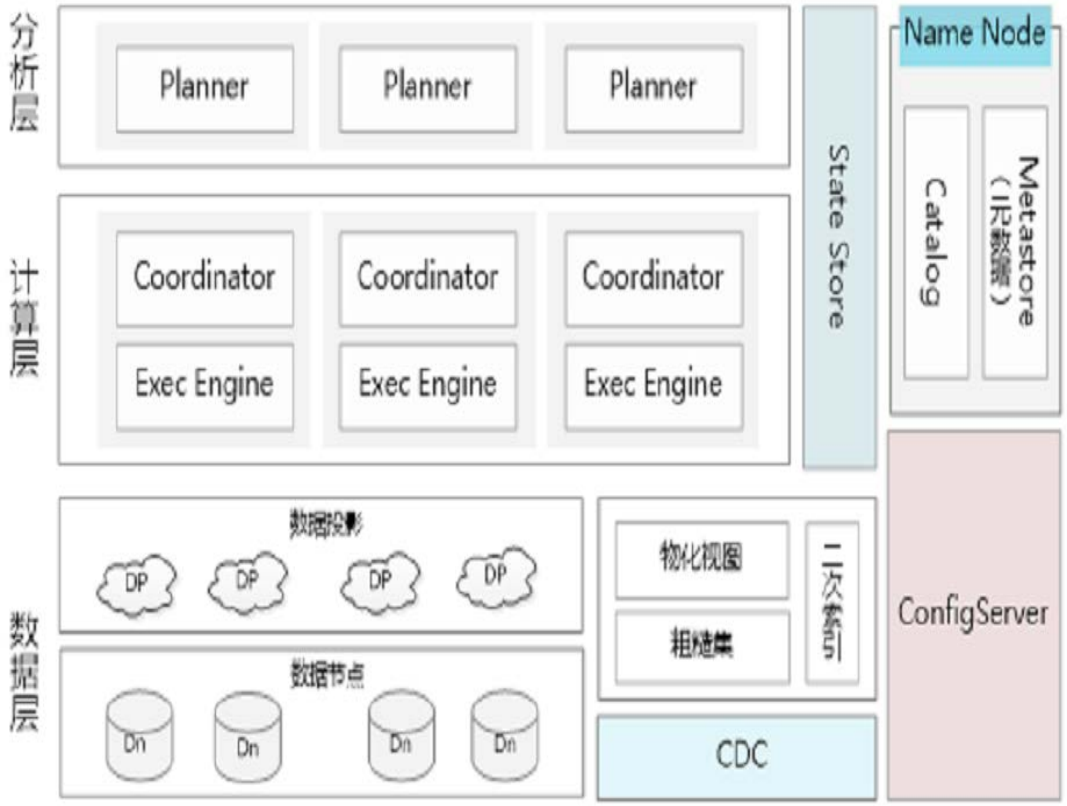
例如，智能卖场的应用，可视化展示大促中的人货场，实时分析顾客的构成和顾客购物路径：他们从哪里来？想买什么？正在围观什么商品？正在往购物车中放什么？结算的转换率？库销比等等，并在此基础上做矩阵分析。





为了解决智能卖场实时分析难题，我们设计了 JingoBase 实时计算平台，较好地解决了百亿级点击流、十亿级商品、用户和订单数据的实时关联分析问题。数据存储采用定制化的 kudu 列存储方案，解决点击流数据快速读写问题；计算框架采用定制化的 Dremel 框架，做关联分析；利用内存平台做基于 SQL 的主数

据投影，进行数据裁剪（例如，商品主数据十亿级，每天动销商品千万级，利用数据投影和主数据更新慢的特点，能大幅提高系统分析能力）。



自动化方向：京东研发一直致力于“仓配客”的自动化、无人化研发。今年618的最大创新点是，京东将率先在全国多个省市用自己研发的无人机送货，这将大大提高配送时效，京东的配送无人车、仓储机器人也正在紧锣密鼓的研发中。

客服机器人JIMI也取得了突破性的进展，研发团队将深度神经网络嵌入到DeepQA框架中，利用深度神经网络的复杂模型表示能力，改善性能。同时，研发全新的场景引擎，通过意图识别模型与记忆模型的整合，使JIMI能更深刻的理解业务，并通过和用户的多轮交互收集意图信息，提升服务质量。在本次618中，会将该技术拓展到更多的售前售后业务上。

InfoQ：能介绍下你们的压力测试方案吗？

吴博：每年618我们会提前预估一个容量指导值，各部门按指导值进行压测，合理配置资源。618压测主要分为：线下压测、线上压测和全流程压测。

线下压测在测试环境进行，主要是单实例、单机、单集群压测，目的是找到系统的基本性能问题，进行优化改进。并依据压测结果，预估线上系统的容量，并利用线上压测验证。

线上压测是对线上系统按比例隔离，模拟真实访问场景，进行线上的多机房集群压测，找出集群的瓶颈点进行优化，得出线上系统能承受的量，作为扩容和容灾方案的依据。

全流程压测是每年 618 大促必做的功课，观察多系统在峰值流量下的相互影响。

InfoQ：在这么大规模的集群中，京东是如何保证数据一致性的？

吴博：大规模分布式系统的一致性问题一直都是业界关注和讨论的焦点，也是系统设计上的一个难题，如何保证数据水平拆分，冗余多份存储的同时不引入数据不一致的问题，各厂均有自己的解决方案和经验总结。

京东在这个问题上，从存储层和业务层都解决了这个问题，同时保证数据严格的一致性。

1）在存储层，我们同时使用主流的开源数据库 MySQL 和我们自主研发的分布式存储系统，来保证海量数据的可靠存储和高性能访问。针对 MySQL，我们在半同步复制基础上做了一些优化和定制，很好的保证了主从切换以及数据访问的一致性问题。针对京东自研的分布式存储平台，我们引入了诸如 Paxos 等主流强一致算法来保证多副本之间的一致性问题。

2）在业务层，不同业务域对数据的一致性要求不同，例如交易环节，对系统的可用性要求高，需要优先保证用户能快速下单，数据要求最终一致性；履约环节的可用性要求没有交易环节高，但对订单状态数据的一致性有较高要求。针对数据一致性要求很高的业务会自己再做一层对账机制，来严格校验和补齐数据，保证数据一致性问题，部分极端情况，也有业务会牺牲掉可用性问题来换取一致性，比如单机 Scale Up 的方案，这样也可以天然屏蔽一致性问题。

InfoQ：为了保证用户体验，在资源有限的条件下，我们必须保证关键系统的稳定性，这也就引入了响应的服务降级方案，能谈谈你们这块是怎么做的吗？

吴博：每年 618 大促的一项重要工作是，研发各部门的预案起草、演练和评审。各部门会针对大促中可能出现的各种问题，提出扩容、限流、降级和故障转移等解决方案，并逐条演练和评审。目的是保障商城业务黄金流程的稳定性，出现故障时，能有条不紊地快速按预案执行，减少故障影响时间。

我们先将系统分层：业务层、应用层、数据层和基础层。其中，业务层上的系统比较轻、靠前；应用层上主要是一些平台级系统，如交易系统、商品系统、用户系统、履约系统等；数据层上主要是生产库相关的系统；基础层上是一些缓存、消息中间件和存储基础件等系统。并制定各层间的依赖原则：上层可以依赖下层，下层不能依赖上层，同层之间尽量异步解耦，避免循环依赖等。

再将系统分级：按业务的重要程度以及故障的影响范围进行分级，分成 0 级系统、1 级系统、2 级系统等，并区分系统中的核心服务与非核心服务，从服务治理的角度进行规划，对服务进行分级分层治理，重点保障 0 级系统核心服务的稳定性。同时制定系统间的依赖原则：0 级系统不依赖 1 级系统，核心服务不依

赖非核心服务等。

经过以上的治理措施后，主业务链上的核心服务发生故障的可能性和影响范围已大大降低。但是在资源有限条件下，还是可能会有影响稳定性的事件发生。这就需要有依赖关系的各服务之间约定好 SLA，超过约定时或者出现特殊事件时，可以进行报警和降级处理。

在制定降级预案时，解除服务依赖并不是唯一的选择，而是根据业务特点、系统特点和具体场景制定多级降级策略。客户端可以采取的降级策略有：解除对非核心的服务依赖，降低服务调用频次，优先处理高优先级和紧急的业务，使用内置简易逻辑处理简单数据。总原则是保证主流程顺畅，降低事件影响范围。服务端可以采取的降级策略有：对相对次要的流量进行限制，对整体流量进行限制，启用备用服务，总的原则是保障服务可用。

实际降级措施后，可能会引起数据不一致现象，所以需要事先准备好数据一致性维护机制，借助日志、状态机等工具，找到不一致的数据，进行补偿，达到数据最终一致。

InfoQ：能谈谈你们的峰值系统的监控架构和方案吗？

吴博：监控系统是技术人员的眼睛，好的监控系统就像一台 CT 机一样，能够透察系统的运转情况。京东的业务具有两个显著特点，一个是业务链条长，一个是促销带来系统的动态性大，因此京东从很早开始构建全方位的监控系统，来实现对系统的洞察力，保障大促和业务的日常运转。

京东的监控系统分为三个层面：业务、系统、基础设施。

1）业务层面的监控，主要侧重在公司的核心业务指标及其多维度的细分，比如公司的实时订单量，以及订单在渠道、省份、运营商、机房、品类、活动等各个维度进行监控，从而在及时发现核心业务指标变化的同时，能够快速定位到问题可能的位置。

2)系统层面的监控,主要是实现系统间各个调用关系及核心处理过程的监控，比如对于两个系统间典型的交互过程，会给出从双方角度看出的调用次数、各分位值的 Latency、成功率等，特别是，对于一个长链条的复杂调用关系，能够从前到后实现贯穿，从而实现在系统角度的快速问题定位。

3）基础设施的监控，主要是机器和网络层面的监控，这是所有业务运行的基础环境，我们会从交换机、服务器、容器上采用黑盒和白盒两种方式来收集对应的指标数据，黑盒数据用于效果的监控，白盒数据用于细化的问题追查，双管齐下，快速协助业务确定基础设施是否正常，以及问题在什么位置。

从监控系统的设计和实现角度，可分为采集、传输、存储与查询、异常检测、Dashboard、报警收敛等各个层面，京东在传统监控系统基础上，结合京东的业务特点进行定制和针对扩展性的研发。比如，我们在京东的 RPC 框架中都植入了统一的 SDK， 能够自动统计 RPC 的调用次数、成功率、时延等指标，在业务层面，我们基本上也按照一套统一的基础设施，多套业务自定义的 Dashboard、数据关联等业务逻辑的方式进行监控系统的收敛和扩展，保证灵活性的同时，避免重复造轮子。

InfoQ：对于超卖，目前京东是如何解决的？这个方案是最优的吗？

吴博：京东有独立的库存系统，分别从业务规则和技术两方面解决超卖问题：

1）从业务规则的上进行防超卖。先根据商品编码查询库存主数据；再计算各个库存项，包括现货、预售、在途等等，按照一些设定的业务规则，进行扣减前校验；然后做库存扣减，库存扣减后校验（回滚）。

2）从技术上防超卖。随着订单量的增长，传统单数据库已经无法满足我们需求，我们将库存扣减数据都放到 redis 进行处理，利用 redis 只能顺序执行命令的特性，进行订单号防重提交处理；然后利用单物理机进行内部数据流转、关键数据闭环处理来保证数据准确性；同时将各阶段的关键数据落地，统计已产生的各项数据并汇总，进行差异比对和数据核查。

这个方案也不一定是最优的，但能满足京东目前的需求。根据目前比对情况来看，库存差异很小，超卖情况基本没有。一些微小的差异，基本也是由于一些新老业务冲突问题或程序更新的 bug 导致的。

InfoQ：您是如何看待微服务架构的？可以介绍下京东商城的微服务化落地流程和方案吗？对于电商平台的微服务化，您有什么可以传递给读者的经验吗？

吴博：微服务是一种较实用架构思想，早在“微服务”提出之前，一些大公司就有不少类似的架构实践，比如腾讯的“大系统小做，分而治之”做法。大的

互联网公司很少照搬传统 SOA 架构中的 ESB 企业服务总线，而是结合自己公司的实际情况做一些改进。

京东商城的架构并没有强调微服务化，平时的架构规划中或多或少地有这方面的体现。目前的京东架构的重点是平台化：保证底层的基础平台基础稳固，中层的应用平台高可用，上层的业务轻薄敏捷。每层服务的要求不太一样，对于中层和底层服务，更重视基础服务的隔离、解耦和高可用。

互联网公司看重的是架构“落地能力”，找到适合自己公司特点的架构，并让架构能在公司业务的快速发展中不断完善，没有一种架构能适用所有公司短中长期发展。不建议过多强调架构的先进性。

“三流的点子加一流的执行力，永远比一流的点子加三流的执行力更好。”

吴博，应用架构师，京东架构委员会成员。2013年加入京东，负责应用架构设计和治理工作。有10多年的互联网工作经验，参与过大型B2C电商平台、搜索引擎、Hadoop云平台等系统的方案设计。对分布式系统、海量数据分析、高性能系统方向比较感兴趣。

京东 618：
揭秘京东历经多年的 618 架构核心

作者 **李东辉**

618 作为京东一年最重要的大促之一，每年 6 月 18 日京东将遭遇记录历史级别的流量挑战。如何成功保障交易平台高并发高性能已经成为包括京东在内的众多电商念念不忘的念想，而京东作为国内电商领军企业之一，在架构积累上成就了如何领先的技术底蕴？现在我们就来采访李尊敬老师，让我们看看京东给这个架构世界带来了怎样的惊喜吧？

InfoQ：能否简单介绍京东商城交易平台的业务，其中您的主要工作是什么？

李尊敬：交易平台为京东商城提供交易平台化的服务，涉及到购物车、结算页、订单中心、促销价格、商品、库存等一系列平台化的服务，PC、APP、微信手 Q 都依赖交易平台提供的服务。我主要负责订单交易相关环节的技术保障、架构升级和性能优化等工作。

InfoQ：您在大流量高并发系统性能瓶颈诊断方面有丰富的经验，能否简单谈谈一般系统内部存在几种瓶颈，哪种瓶颈是“木桶短板”？能否举例阐述判断系统瓶颈的过程？

李尊敬：我接触过的系统中瓶颈点大致有以下几类：

1. CPU瓶颈：序列化和反序列化、高频日志输出、大量反射的应用是CPU飙

高的主要原因。

2. 网络带宽瓶颈：在大流量高并发系统中，网络带宽也会成为瓶颈点，交易平台这边解决带宽瓶颈的方式主要有压缩输入输出内容、使用双网卡和升级万兆网卡等方式。
3. 磁盘IO瓶颈：App中的磁盘IO飙升主要是高频和大量日志输出导致的，可以通过增加日志缓冲区、精简日志来优化。数据库类IO瓶颈主要是通过优化查询、使用Fusion-IO、水平Sharding 分库分表等方式优化。
4. 数据库连接数瓶颈：随着流量的暴增，应用服务器也在不断增加，这时数据库、Redis等连接数就会出现瓶颈，导致前端应用拿不到连接数。
5. 木桶短板：CPU、网络、磁盘IO瓶颈通常都可以通过单纯增加资源、升级设备解决，但是数据库连接数容易形成木桶的短板。目前我们主要通过服务隔离和解耦、数据库垂直拆分、分布式数据库集群来解决连接数问题。

目前我们是通过全链路压力 + 全链路监控来实现快速发现系统瓶颈的，全监控链路实现对各层应用瓶颈点监控。

InfoQ：京东交易系统历经多次重构和改造，能否举例说明每次重构是如何突破上次重构的瓶颈的？能否分享重构中过程中的经验与教训？

李尊敬：2014 完成独立秒杀系统，将秒杀和交易主流程彻底解耦，解决了秒杀影响交易主流程的问题；2015 开始诺亚方舟计划和多中心交易项目，分别实现了突发流量下系统弹性扩容和交易系统同城多活。

多中心交易项目核心功能需要实现 MySQL 数据库、Redis 的多机房写的问题。在解决多写的问题上我们关注了开源社区的实现，例如 Linkedin 的 Databus、阿里的 Otter，在这些基础上自主实现了适合京东交易系统的组件。

架构升级改造经验和教训：架构升级最大的风险是业务风险，通常单纯的架构升级是不改变原有业务流程的，但是架构升级势必涉及到代码变更，因此架构升级后的系统功能性测试非常重要。交易平台多个系统架构升级采用基于用户流量比对测试方法，完成系统升级安全切换。

InfoQ：能否谈谈你们订单中间件系统去 IOE 的影响，为什么要去 IOE，

目前使用了什么技术代替，效果如何？

李尊敬：订单中间件系统是订单生产环节核心系统，早期存储用的Oracle，随着订单量的突飞猛进，Oracle 数据库连接数和 IO 出现明显瓶颈。

我们去 IOE 的方式是将 Oracle 等商业存储设备换成 MySQL 集群方式。去 IOE 在架构上最大的工作量是 SQL 转换，Oracle 和 MySQL 的 SQL 语法有差异，而且 Oracle 单机性能远好用单台 MySQL，之前在 Oracle 跑的很好的 SQL 到 MySQL 会很慢。解决这个问题的方法是优化和拆分 SQL，将压力分摊到客户端。

大量变更 SQL 业务风险很大，因此需要全方位的功能和性能测试。在这个项目中我们首次尝试将用户流量完整 copy 到升级后的集群，将订单生命周期全过程回放到新集群，然后通过比对新老集群响应结果进行性能和功能测试，很大程度上缩减了去 IOE 工作量。

目前交易核心业务都实现去 IOE，立足点是满足业务的需求，同时节约成本，发挥最大的效率。这也是京东一直保持业务与技术双向驱动的态度：业务发展推动企业规模，进而规模的扩展推动技术成长；另一方面，技术创新的成果能够更有效地保证业务发展，甚至引领业务的发展，这一直是京东所遵从的原则。

InfoQ：去年 618 当天访问峰值达到了什么级别，同时当天暴露了哪些架构短板？在平时的大促中如何保证数据一致性？大促当天性能如何在原来基础上再度提升？

李尊敬：去年 618 当天京东商城的订单量突破 1500 万单，实时价格、商品等核心接口峰值调用量达到上千万 / 分钟。去年 618 按照 10 倍流量标准备战，经过多次军演和预案演练，当天系统表现稳定。订单交易系统对数据一致性要求极高，像订单号、下单核心系统都是采用不带缓存的无状态化架构设计，采用水平扩展的方式对抗线上的流量。

对于订单中心等应用采用最终一致性原则来保障数据一致性。大促当天主要工作就是预案的执行，通过执行清洗恶意流量、分流和限流预案，降低系统负载，保障整个交易系统的高可用。

InfoQ: 什么是无状态化的架构设计？与一般的架构设计有什么区别和优势，实现上有什么挑战？

李尊敬：无状态化架构设计是指各服务之间完全独立，地位均等，不存在状态化数据交互和同步。无状态化最典型的例子就是 WEB 服务器。

无状态化的架构优点：天然高可用，可以水平扩展，底层依赖的数据不需要相互同步，无状态的架构是系统高扩展性的基石。交易平台订单号服务，接单服务等都是采用无状态化架构设计。

以接单服务集群为例，各个接单服务独立运行，数据完全独立，一台服务或者数据库宕机后会从服务中自动下掉，整个服务保持高可用。

无状态具体实现上的挑战是需要根据业务将系统拆解成底层数据相互独立的单元。要求 SOA 化的时候服务的力度要足够细，另外在有些场景下服务必须是有状态的，因此并不是所有系统都可以做到无状态。

InfoQ：每年 618 一般在什么时候启动技术准备？在架构上具体需要准备什么？今年 618 与前几次大促相比在技术准备上存在哪些不同？如何通过数据预测今年 618 的情况以及承受压力？

李尊敬：每年 618 在春节后就会启动技术备战，通过完整的预案和演练来应对 618 的压力。架构层面我们准备的工作分成以下几点：

- 交易相关系统架构梳理，风险点评估；
- 系统机房部署情况核对，是否遵循流量闭环原则，有无跨机房调用情况存在；
- 检查各系统和渠道服务器资源分配情况，资源利用率是否合理；
- 交易核心流程调用链梳理，各渠道终端的用户整个请求链耗时情况分析。

今年 618 和去年大促技术上准备有很大的不同，本次 618 交易系统全面接入到弹性云 docker 平台，诺亚方舟机房全面启用。接入弹性云后系统扩容和缩容变得更加简单和便捷。

为了保障从物理机到弹性云安全切换，我们在弹性云环境做了大量压力测试，针对弹性云环境做了参数调优和性能优化工作。今年也会启用多中心交易二期项目，多机房热备，同时承担用户流量。

在预测方面，我们通过最近单量的分布情况，以及促销力度和数量，结合机

器学习算法实现对 618 当天单量以及核心系统调用量和性能预测。

InfoQ：你们预测 618 当天会给你们带来哪些方面的挑战？能否介绍你们预先准备的解决方案？是否存在可用的“四两拨千斤”技巧？

李尊敬： 今年 618 交易系统面临的挑战有以下两点：

- 成倍增长的访问量对交易系统的挑战

交易平台核心交易系统按照 20 倍日常流量的规模来备战，通过全链路压测模拟在 20 倍流量的压力下系统的表现，事先找出系统薄弱点，提前做好系统的优化和扩容准备工作。对来自非正常用户的恶意流量将建立风险分级策略。

- 弹性云

今年 618 交易系统首次全面接入弹性云，为了保障交易系统在弹性云环境下安全稳定运行，我们在弹性云环境做了多轮全面压力测试，对于较难实现的部分通过写流量压测、采用憋单演练的泄洪的方式，用真实的用户订单来考验弹性云下的订单交易系统。

对应大促期间的洪峰访问，有几个基本技巧：

- 分流，将核心系统和非核心系统流量分离，按照用户访问特征分流。
- 限流，按照用户安全等级分级限流，清洗恶意流量。
- 异步化，非交易核心功能尽量异步化处理，削弱访问量峰值。

InfoQ：2015 年 10 月上线的多中心交易系统成功通过了双 11 的考验，能否简单介绍多中心交易系统，它有怎样的优势，实现起来有什么难点？在这次 618 会承担怎样的角色？

李尊敬：多中心交易系统设计类似实体商超：多个交易中心按用户分流，每一个交易中心建在不同地区，用户直接访问本地区的交易中心。多中心交易系统不仅响应速度快，用户体验更好，并且每个中心服务一定数量的用户，用户则不用跨地域访问数据中心，水平扩展性好，进一步降低单个中心的数据压力。多中心交易系统能支撑更大的交易规模，可支持异地容灾，进而降低灾难的影响和风险，更加安全。

数据一致性是实现多中心交易系统最大的挑战，数据写错了无法恢复。其次要保障用户从进入京东商城到浏览商品，到访问数据库，该全链路的路由规则都

是完全一致的。

今年多中心架构在性能、可视化等多方面进行了较大的改进，而今年多中心交易主要有三个目标：

- 多数据中心支持交易流程；
- 异地容灾，实现无缝切换；
- 用户体验提升，保障就近访问和交易。

李尊敬，京东商城交易平台架构师，负责京东商城购物车、结算页、订单中心等核心系统架构设计和重构工作。先后负责和参与过订单中间件系统去IOE、订单中心架构重构、多中心交易系统中订单交易系统设计和改造工作。对大流量高并发系统性能瓶颈诊断、架构重构和高可用方面有丰富的经验。

版权声明

InfoQ 中文站出品

出版：极客邦控股（北京）有限公司

北京市朝阳区洛娃大厦 C 座 1607

欢迎共同参与 InfoQ 中文站的内容建设工作，包括原创投稿和翻译，请联系 editors@cn.infoq.com。

网 址：www.infoq.com.cn