

阿里巴巴的上市不仅是商业和资本领域的一件大事，对于技术世界也是如此。以此迷你书感念诸位阿里技术达人在阿里巴巴成长过程中的努力，感念技术人的专注和可爱！



阿里巴巴上市背后的 技术力量

- 阿里巴巴国际站发展历程
- 淘宝交易系统演进之路
- 天猫推荐算法团队那些事儿
- 阿里云技术实现与难点
- 小微金服专家谈数据的力量



推荐序

我至今还记得2008年夏天一个阳光灿烂的午后，InfoQ中文站的创始人霍泰稳背着易拉宝满头大汗地来到支付宝大楼，和支付宝同事一起举办QClub。也是在那个午后，我紧张又兴奋地向社区的同学第一次分享了支付宝架构。6年多过去了，泰稳还在坚持做InfoQ，我还在坚持做支付宝平台，都在坚持自己的梦想并且乐此不疲。拥抱泰稳，也感谢InfoQ的作者和小编们，读你们充满匠心的文章，不仅收获技术，而且得到享受！

我相信这本“阿里技术”迷你书会给大家带来很多收获。支撑阿里繁荣商业生态的生产系统，是技术人员最严酷也是最好的老师。书中分享的阿里技术，都经历过每天海量的点击、交易、资金和数据的验证、承受过一次次“双十一”大促的极限考验、甚至源自于令人痛心疾首的系统故障，是难得的技术财富。书中分享的阿里技术背后，也体现了梦想与坚持。一直以来，阿里都坚持在能够促进未来商业变革的技术方向上投资，无论电商与金融技术平台、云计算、大数据、云OS、数据库……，从最初的梦想、到一路上历经挫折而永不放弃，到每一次成功后又迅速颠覆自己开创新格局，我觉得这种精神比技术本身更加宝贵。向阿里技术人致敬，并向同样拥有这种精神的所有技术人员致敬！

祝愿InfoQ技术社区越办越好，一起分享技术、分享感悟、创造更精彩的未来！

支付宝奠基人之一、小微金融服务集团CTO程立（鲁肃）

写在前面的话

阿里巴巴的上市，这不仅是商业和资本领域的一件大事，对于技术世界也是如此，在阿里巴巴的30位合伙人中，我们可以看到多位技术人员的影子，比如阿里巴巴集团CTO王坚，小微金融服务集团CTO程立（花名鲁肃），阿里巴巴集团副首席技术官姜鹏（花名三丰），阿里云高级研究员蔡景现（花名多隆），阿里巴巴集团技术保障、安全技术产品部副总裁刘振飞等。

自从人类发明了计算机，技术人的世界就是那么让人痴迷，有多少痴男怨女一朝踏入技术门，从此献身其中。这也是为什么InfoQ一直在强调，软件正在改变世界，技术人正在改变世界。在过去6年中，InfoQ中国对阿里巴巴、淘宝、支付宝、阿里云技术团队进行了全方位的报道，包括邀请其专家来[QCon全球软件开发大会](#)（10月16-18日，上海）和[ArchSummit全球架构师峰会](#)（12月19~20日，北京）做分享，其中程立（鲁肃）将在QCon上做题为“[大数据时代的金融技术变革](#)”的主题演讲。

在这本迷你书中，我们的编辑按照阿里巴巴技术、淘宝和天猫技术、支付宝技术、阿里云技术等四个维度，挑选[InfoQ中国上部分精彩内容](#)，以飨读者。也以此感念诸位阿里技术达人在阿里巴巴成长过程中的努力，感念我们技术人的专注和可爱。

InfoQ中国会一直和大家在一起，一起怀揣改变世界的梦想，继续努力！

InfoQ中国联合创始人兼CEO 霍泰稳

目录

阿里巴巴相关技术

阿里巴巴陶勇谈海量数据技术架构

潘磊谈阿里巴巴国际站发展历程

王晶昱：消息系统架构与变迁

周涛明：跨境网站的优化与挑战

张旭升谈ITIL及运维架构

淘宝和天猫相关技术

曾宪杰谈Java在淘宝的应用

王海亚：淘宝交易系统演进之路

张奇：天猫推荐算法团队的那些事儿

长仁：核心系统研发部的发展历程

阿里云相关技术

王坚博士：云计算就是服务

徐常亮谈阿里云ODPS的愿景、技术实现与难点

章文嵩：怎样做开源才有意义？

支付宝相关技术

小微金服前端技术专家伯约访谈：相信数据的力量

蔡学镛谈复杂事务处理（CEP）

程立谈大规模SOA系统



促进软件开发领域知识与创新的传播



中文 | 英文 | 日文 | 葡文 |



1

阿里巴巴相关技术

阿里巴巴陶勇谈海量数据技术架构

受访者 陶勇 作者 霍泰稳 发布于 2012年1月17日

个人简介 陶勇，2005年底加入阿里巴巴，先后从事过网站架构支持，PMO等工作；2010年成为技术部分布式数据库技术领域带头人，现负责海量数据领域相关技术产品的研发及推广，为解决网站多站点服务、数据强一致性等特性提供高可用的海量数据访问及数据消费等技术产品及架构支持，海量数据领域涵盖分布式数据库、分布式存储、数据实时计算等多个技术方向。

InfoQ：大家好，我现在在阿里巴巴园区采访阿里巴巴的资深技术专家陶勇。陶勇你好，请先向大家介绍一下你自己。

你好，我叫陶勇，05年加入阿里巴巴，现在在平台技术部负责海量数据这一块。就目前来讲，这一块分为了以下几个技术领域，包括分布式数据库、分布式存储，以及分布式计算。其实这部分在业界，都有一些成熟的原型及应用。在B2B，虽然具体的业务场景可能不太相同，但也会有大同小异的地方。所以大家有什么想了解的我可以做个介绍。

InfoQ：请先向大家介绍一下阿里巴巴网站，它目前的数据量大致在什么样的规模。从当初起步发展到现在，经历了哪些重要的阶段，每个阶段又有哪些特点？

阿里巴巴网站目前来讲分为中文站、国际站，还有一些国际交易和日本站之类的一些站点。我们核心表的数据量大概都是在亿和十亿级别，在这样的一个数量级下面，都会涉及到一些容量方面、性能方面，以及分布式方面的一些问题。整个网站发展到现在已经11年到12年的时间。我们的数据量也从最开始的很少一点，扩大到现在的规模，尤其是近几年更是呈爆发式的增长。数据库层面来讲，关键点是两个，一个是之前我们一直采用的Oracle数据库，不管是中文站还是国际站，Oracle数据库有非常多的优点，但

是随着数据量的飞速增长，无论是用小机也好，还是更高端的存储也好，都会带来瓶颈的问题。我们在07年末，08年初的时候意识到了问题，因为那时候，我们的数据量呈现急剧上升的趋势。当时我们考虑的是将数据库进行拆分，拆分现在对于大家来说都已经耳熟能详了，包括水平拆分和垂直拆分。如果从数据库角度来讲，就两个阶段，是99年到08年，然后08年到现在的这个阶段。

08年相当于是我们从单一的Oracle数据源跨越到多数据源支持的一个关键时间点。从08年开始，我们经历了大概一到两年的时间，将中文站最核心的表，以及和它关联的一些表拆分到了MySQL上，这时我们已经通过数据库的水平拆分，将数据均匀的拆分到了MySQL集群上面。不管是TPS (Transaction Per Second) 也好，容量也好，都极大的缓解了之前Oracle遇到过的问题，包括它的逻辑读写非常大，CPU也一度飙升到三四十之类的问题等。从目前来讲，阿里巴巴的数据库，相对于其它互联网的一些网站，比如说淘宝，比如说Twitter之类的，本身还是有一些特点的，目前来讲，还是会员服务的性质为主。对于会员来讲，数据的一致性以及可用性要求都是非常高的。与传统互联网企业相比，现在大多数新型的互联网企业在数据存储方面的要求可能有细微的差别。所以另外一点，数据要从集中式走向分布式。还有一点就是阿里巴巴网站相对于其他的站点而言，是少有的跨多个IDC，具备容灾备份的站点。不管是中文站还是国际站都有多个站点分布在不同的国内，甚至是国外的一些机房里面，这样的话，在包括数据一致性以及数据同步这一块，都会有一些特有的解决方案，大致就是这些特性。

InfoQ：针对如此大量的数据采取什么样的策略来保障，即能快速响应用户的请求，又能够及时应对业务的发展呢？

从数据库层面或者数据访问层面来讲无非就是两点，第一点，是压力如何分摊，分摊的目的就是为了把集中式变为分布式，这是其一。另外，采用多种的存储方案，不同的业务数据，它必然会有不同的特点，针对不同的特点，我们会采用例如Oracle来存储，或是MySQL来存，使用集中式的存储，还是分布式的存储，是采用传统的RDBMS，还是采用KV Store，或者是其他的一些存储方案。综合以上两点，从数据拆分来讲，我们目前在B2B更多的是水平拆分，我们把一张数据量非常大的表，上亿级别数据量的表，把它从Oracle里边搬出来，拆分到MySQL里边，这是一种水平拆分的方式。我们偶尔会采用一些垂直拆分的方案，但是垂直拆分方案相对而言，是一种过渡方案，最终还是为了达到水平拆分的目的。

其实水平拆分主要为了解决两个问题，一个是，底层存储的无关性。另外一个就是随着数据量以及访问请求包括TPS、QPS (Query Per Second) 的压力增长，我们能够通过线性的去增加机器就能够满足要求。此外还可以利用线性的这种增长方式，去支持压力和数据量的增长。至于多存储方案而言，其实这跟我们的业务是有紧密关系的，比如说我们非常核心的数据，目前还是选择存储在Oracle里面。像一些大数据量的，压力非常大的这种表，我们就选择把它拆分到MySQL数据库里面去。还有一种，其实很简单，比如说KV Store，跟目前的NoSQL的这种选型类似，就相当于把一些关系型不是特别明显的，例如网站的一些用户属性以及一些Property之类的数据，我们都把它放到KV Store里边去，大致，就是用这些技术手段来解决，应该说，最关键的一些技术解决方案，大致就是这些。

InfoQ: 刚才也提到其实阿里巴巴的数据，分布在不同的区域，对于这些不同地域的数据中心，同步是怎么处理的，采用什么样的方法，效果怎么样？

去年在QCon的时候，潘磊做过大致的介绍。当然现在我这边是有些更新了，因为这一块目前是我来负责，就目前来讲，其实有三个产品来密切配合解决这样的一些问题，这三个产品分别是Erosa、Eromanga和Otter。可能听上去会让人感觉有点摸不着头脑，不知道是干什么的，Erosa简单的说就是做MySQL的Bin-Log时时解析，我们会对MySQL也好，还有一些其他的KV Store也好，他的BIN-LOG进行时时解析，解析完了之后，会放到Eromanga，Eromanga是什么东西？其实就是增量数据的发布订阅的产品，Erosa产生了时时变更的数据发布到Eromanga。然后，这时候，不管是搜索引擎也好，还是数据仓库也好，还有一些关联的业务方也好，他就可以通过订阅的方式，把这些时时变更的数据时时的通过Push或者是Pull的方式拉到他的业务端，然后进行一些业务处理。

还有一块是Otter，其实更多的是它的定位，就是跨IDC的数据同步，之前是国际站，现在是中文站，中文站我们也有容灾站点，应该不光是说容灾站点，应该是AA站点，这时候，我们要求我们的数据能够及时的反映到两边去，我们是通过Otter来解决这方面的问题。Otter，其实就像是类似于业界，比如说现在SharePlex的这种方式的数据同步的方式，双向同步的数据解决方案，相比而言，不管是数据库的同步，还是说SharePlex的同步也好，我们做了一些定制化，包括对业务数据关联的一些文件系统同步，我们是按照事务的方式来将数据通过Otter重组应用到另外站点的数据库里面去，这是相对于目前业界的一些商业产品不同的地方。

另外一个，数据同步冲突这一块，其实阿里巴巴站点，虽然我们现在新的一些架构都极力避免同一条数据在两IDC机房里面的数据库中同时被修改，但是由于一些历史原因，的确存在有一些业务既在这个机房里变更，与此同时，他也会在另一个机房里面被变更，这时候怎么解决同步冲突。这时候我们其实这是这一块我们目前来讲，是最难解决，但是目前又还急需解决的，我们目前，当然现有的版本，暂时是一种比较简单的方式去解决问题的。其实就是以那个站点数据为优先，站点A，比如说A机房的站点的数据是优先的，不管怎么样，它就覆盖到B的，今后我们会有一些包括业务相关的策略，包括变更的时间，及冲突合并之类的方式来解决到这种双向同步，以及变更冲突这方面的一些问题。从时时效果来看，目前来讲，Erosa和Otter，目前承担了几乎B2B所有的，不管是跨IDC的还是说从跨站点的，比如说我们从中文站同步到DW，还是从国际站同步到DW，同步到其他的一些数据源。那目前来讲，基本上现在已经是遍地插满旗帜，大致情况是这样的。

InfoQ: 其实缓存在整个系统架构中的地位还是相当高的，像阿里巴巴这样的系统，是如何使用缓存的。另外，是怎样设计好系统的缓存结构？

缓存这个东西，首先在阿里巴巴来讲，它其实是分为几个级别的，从前到后，比如说从用户最开始发起HTTP请求，到最后访问数据库，目前来讲，是分了三级缓存的，包括图片，包括页面的缓存。然后直到最后面的包括数据的缓存、图片的缓存，以及静态页面的缓存，相对而言，都是采用热点数据方式，包括提高他的命中率，基本上我们的命中率都非常高，前端相对而言都比较高。对于后端比如说数据缓存这一块，目前来讲，分为Local Cache和Remote Cache，应该叫Distribute Cache。就相当于说，我们会把一些不怎么变化的比如说属性数据，用户属性，或是一些其他的属性数据，会在Server启动的时候，将其加载到Local Cache，这时，就不需要关心一致性的问题，反正就是启动之后就加载到Local Cache，这是一种缓存。

还有一种，就是Remote Cache，相当于是分布式的Cache。就B2B来讲，目前Cache的实现策略非常多，实现的底层不管是包括Berkeley DB也好，还是Memcache DB也好，实际有很多的。同时我们在前端也做了一层封装，无论底层的Cache是用哪一种存储实现，都能根据业务场景需要，提供出一些最优推荐。对于如何提高缓存命中率，如何进行缓存的设计，我说下B2B的一些选择。就前端图片和页面来讲，我们会把一些动态页面静态化。静态化这块我们有相应的产品来支持动态页面静态化。数据缓存这块，我们更多的是从对象缓存的角度来看，缓存命中率更关键的是把缓

存力度划分的越细，命中率相对会越高。另外一块，缓存跟业务是具有相关性的，在做架构设计时，需要确定哪些是需要缓存到里面的，因为缓存并不是无限大的。另外，缓存是有有效期的，不可能无限的让他存在那儿，要真是如此的话，他就不叫Cache，叫Store了，所以这块，大致就是这样一些机制。第一个是注意切分力度，什么样的业务需要保证什么样的力度，有些细力度，有些粗力度。还有一块，如何保证缓存的生命周期，他的有效期是多久，因为不可能把所有东西都缓存到Cache里面去。

InfoQ: 刚才也提到对数据库的拆分，那其实在数据量非常大的情况之下，数据库拆分是今后一定要去进行的一件事情，在拆分的过程中，是如何保证即不间断服务，又能进行平滑的过度？

其实在阿里巴巴，特别是在技术部，这边的团队有一个比较有意思的称谓，叫做拆迁大队。三国杀游戏里有个人物叫甘宁，就是专门负责拆牌的，这个团队其实就是把目前Oracle中大数据量的表拆成MySQL的。目前来讲，我们拆分，其实是有几个拆分策略的，第一个是按字段拆分，这是最细力度的。比如说一张表我按某个字段Company拆掉，我就按COMPANY_ID来拆，这是一种方式，还有一种是按表来拆，我把这张表拆到MySQL，那张表拆到MySQL集群，更类似于垂直拆分。还有一种是按Schema拆分，Schema拆分就相当于说，这是跟应用相关的。比如说我B2B这样应用，我会把它拆分到这个Schema机群里面来。另外W服务我会把它的数据库拆到另外MySQL机群。但是对外提供的还是这一组，就是前面我也提到了，我们说的Cobar，可能还没提到，就是说我们其实拆分负责数据拆分，我们有产品叫做Cobar，Cobar其实类似于MySQL Proxy，他解析了MySQL所有的协议，就相当于是说，我们是可以把它看成MySQL Server来访问的，它前端是挂了很多Cobar Server的。根据Schema拆也好，按照表拆也好，在Cobar上面会根据你的需要，去定义自己的拆分策略。拆分好后，就自动会路由到MySQL不同的服务器上面去。

还有一方面其实最关键的是，拆掉了之后，怎么样做数据的扩容，其实我们从08年发展到现在，只能说还在路上，因为有些自动化以及自动化的配置，其实跟流动计算是相关的，我怎么样要让正在运行的，让某MySQL机器人上来，然后，比如说这里有一千万数据，我把它对等的拆成各五百万，拆到集群上面，然后再把这台老的，MySQL上面的五百万老数据干掉，我们最终理想的状况是想自动化的完成，但目我们还是更多的是纯人工的方式来做，就相当于是说先把这500万搬过去，这时候其实有拆分原则的，但是我们内部，搬过去之后，然后再把这边的切掉，这是纯人工的。

就数据迁移来讲，我们内部，是有严格的五步走的策略，要把Oracle的一张表或者是数据库拆分到MySQL上面去，这时候我们有严格的五个步骤来做的，每个步骤做完了，第一步骤做完了才能做第二个，第二个步骤做完了，做第三个，简单来讲其实就是说第一个Oracle读写，MySQL写，就是慢慢的把一些数据写到MySQL里面去，这时候要做过程就是要把Oracle的数据要做一次数据搬迁，迁移到MySQL上面。后面，就是设一些阀值。比如说产生ID以上的一些数据，就写到MySQL，以下的数据写到Oracle。第二步是Oracle读写，然后MySQL读，再到后面的一步一步的走，都有不同的策略，这时候，当这五步完成了之后，整个数据就迁移完了，最开始的第一个是最大的表，我们才迁移拆分，花了漫长的将近两年时间，有一些原因是因为业务资源的配合问题，还有一块我们是在摸着石头过河，我们需要吃到一些亏，然后我们要去改进，吃到一些亏我们要去改进。但是到现在拆分基本上都是在两三个月就能搞定。我们没有像一些那种新兴公司，他们做数据迁移就是直接通知用户，我停一天，或者我停几个小时，然后我把数据迁过去，你可以继续登陆上来。基本上我们是不允许这么干的，所以我们通常来讲，要么就是晚上，很短的时间点，我们把数据做一次迁移。另外一块，我们要保证应用基本上不会受到影响，也就是说他可能需要配合我们做一些Restart，但是他的相关的一些操作，尽量不会影响到我们的网站用户，大致就是这么多。

InfoQ: 那如果说需要对线上的数据进行迁移，有没有什么比较特殊的策略？

线上的数据做迁移特殊策略。其实跟我们前面说得五步走是类似的。

InfoQ: 针对分库分表的情况有没有使用一些框架或者工具来为开发和部署提供便利？

分库分表都是在Cobar产品里面的，Cobar分为两类，分别是Cobar Client和Cobar Server，根据业务的需要进行选择，Cobar Server是一组独立的（Stand Alone）的Server集群，Cobar Client就是第三方的Java包，他就直接嵌入到应用上面去。然后他的拆分规则都是直接写到应用的配置文件里面的。这是我们自主开发的，没有用到外面的一些开源的工具，包括我们的SQL Parser，既然要做数据拆分，必然要解析SQL，SQL，我们之前也是自己写的。最近新的版本已经改用纯手写的，我们计划是提升大概四到六倍甚至十倍的性能，说到这点，我们的Cobar Client还是我们的SQL Parser，今年都有计划把它开源出去，相当于是说我们吃了很多我们要回馈一些出来。还有一些工具，就是我们有一组，不管是测试环境也好，还是线上的预发布环境也

好，都有一组服务器提供给业务去负责查询一些业务数据，或者说查询测试环境的一些数据，去检查它的路由规则是不是正确，大致就是这些东西。

InfoQ: 我们下面来谈一谈关于后台海量数据的处理上的一些问题，对于后台海量数据的处理，阿里巴巴的系统是如何设计，有没有使用到类似Hadoop这样的技术？

后台系统，目前后台系统通常B2B是有专门的部门叫数据仓库，就是DW部门的，他们会把我们的数据都拉过去，然后在那边进行一些分析。目前来讲，其实五花八门用的很多，比如说Oracle Rac，比如说Greenplum，当然也会用到我们的Cobar，还会用到前面提到的Erosa。其实之前来讲，首先后台基本上是一些离线数据，但是近几年，特别是从09年开始，我们对于后台的一些报表分析这一块，对于数据的时效性要求也越来越高。从去年我们时时数据中心上了Erosa以后，我们数据时效性，从一天已经缩短到了五分钟，甚至会更快。

另外一方面，Hadoop应用在B2B相对而言比较少的，因为我们更多的是采用了Greenplum这种方式来做，就是做一些BI分析也好，还是报表分析也好，复杂SQL都是通过Greenplum来做的。但是，随着我们的DW业务也在增长，所以他有一些时效性的查询需求，目前我们Cobar也会提供，他们也搭建了一套Cobar集群，来提供这样的在线数据服务，其实相当于现在DW我们已经搭建了一套相当于是分布式数据库的原型，只是我们是东拼西凑的把它拼起来的，但是效果不错，但是，总感觉不是很爽。所以接下来我们要做的，就是能够搭建一套适用于网站OLTP，也适用于DWOLAP的这样的一套或者是说你可以认为两个子领域的平台，这样的话，就能够做到技术可控，无论是数据访问也好，还是整体性能也好，都能够被满足，至少不会比现有的差，大致情况就这样。

InfoQ: 如果说将来有一天像阿里巴巴也要对外开放自己的数据，就好像现在淘宝一样，他也有自己的淘宝开放平台。那在您看来，会对系统带来什么样的挑战，应该如何应对？

其实我们已经走得很慢了，另外，B2B Open平台目前已经在做了，我们现在的技术储备相对而言都已经建立起来了。其实更多的是一些商务上面的，或者说我们对外的一些开发社区也好，还是说我们的应用一旦上来之后，包括流量监控，一些API的设计，这块东西，其实我们比淘宝也好，这块我们是走的相对慢的。但是我们已经从淘宝借鉴

了非常多的经验。从技术储备上来讲，我们的技术底层，包括服务平台，包括Open平台，已经有一个叫做Ocean的产品，就的专门做Open Platform。这一块，技术已经搭建起来了，我们现在也在摸索，在尝试着在几个业务方面，开放一些业务数据出去，然后提供给外部调用。整个这一块东西，因为实际上，我这块负责的不是特别多，但是我大致对整个技术部的技术领域大致都了解一些，所以基本的情况就是这样的。但是更细节，可能需要等我们的Open平台搭建出来了之后，整个开发社区也就会出来，大家也可以关注一下那方面的一些东西。

查看原文：[阿里巴巴陶勇谈海量数据技术架构](#)

潘磊谈阿里巴巴国际站发展历程

作者 丁雪丰 发布于 2010年11月1日

InfoQ：观众朋友大家好，我是来自InfoQ中文站的丁雪丰，现在正在QCon北京站的大会现场。在我身边这一位是来自阿里巴巴国际站的潘磊。潘磊能不能向观众朋友们介绍一下您自己，还有阿里巴巴国际站呢？

潘：大家好，我叫潘磊，来自于阿里巴巴国际站。我大约是04年加入阿里巴巴的，阿里巴巴国际站是一个B2B的电子商务网站，主要服务于全球用户，大概的情况就是这样。

InfoQ：我们知道阿里巴巴旗下的网站有淘宝、B2B国际站，还有支付宝等等。这些网站都有巨大的用户访问量，相信阿里巴巴能成长为现在这个规模，不是一日而成的，能否给大家介绍一下国际站的发展历程？

潘：阿里巴巴国际站可能是阿里系里面存在最久的一个站点，它建立于1999年，当时只有很少的几台服务器。发展至今已经整整十年了，这当中也经历好几次比较大的重构以及一些架构的变迁，才有了今天的访问量。当然在阿里系里面阿里巴巴国际站的访问量还是比较低的。

InfoQ：在整个发展过程当中，有没有让您觉得如履薄冰的时候？

潘：这个肯定有，印象最深的一件事情发生在早期，那时经常需要半夜起来做一些维护。阿里对外承诺7×24小时提供服务，维护过网站的朋友都知道，这就意味着我们要在很短的时间内，及时解决线上问题，而线上问题往往是稀奇古怪的，往往晚上一接到电话，大家神经就高度紧张，紧急成立一些团队，然后投入到紧张的处理故障过程当中。我觉得应该有无数个不眠之夜，让我印象非常深。

但这是很久以前的事了，最近还有一件事情，就是最近那次重构。我们那次重构前后历时有六个月，把阿里积累了五六年的代码和数据都重新梳理了一遍，因为时间比较紧，过程当中出了很多的问题，也有很大的困难。当时的项目组，几乎就是整个国际站技术部，整整做了大半年。每个人都加班加点，基本没有休息日，一直到发布完成之后，我们还紧张了好几个礼拜，我觉得那件事情对我来说是印象最深得一件事情。

InfoQ：您刚才说国际站经过了一次大的重构，刚刚上线，那能不能从技术的角度，给大家讲一讲国际站现在的架构？

潘：我觉得阿里系的这些网站的架构，基本都还是比较类似的，除了支付宝，它的后台是一个支付平台，可能会有点差异。其他网站的架构基本上都会是基于数据库+搜索引擎+存储，当然也会有一些Cache，一些集中式或者分布式的Cache系统，这样的一个解决方案。我们基本的做法是实现系统的小型化，以及系统的分离，经常让读系统依附于一些可线性扩展的Cache系统，或者KV Store这样的系统，而把写系统集中在Oracle或者MySQL这样的数据库上。

InfoQ：讲到数据库，在国际站现在的规模下，数据量应该是不断成指数级地往上增长，在这方面，国际站会对数据库的容量有一定的规划，您在这方面是怎么做的呢？

潘：其实就数据库来说，我觉得业界的发展方向是比较明确的，首先是拆分的问题，拆分其实分成水平和垂直拆分。其实在这之前，我们做的最重要的一件事情是提供更多、更丰富的持久化解决方案。除了数据库，我们会把一些并不是非常关系型的数据，都迁移到一些KV Store，或者说迁移到一些其他的持久化解决方案上。

在这之外，对于数据库这块，阿里过去一直比较依赖于Oracle，甚至比较依赖Oracle的Rack解决方案，我们将来的目标是，首先让数据库变得更简单，因为我们会有自己的数据库分布式解决方案，以及Cluster解决方案。首先要做到，让数据库，让我们的应用对数据库的类型不再敏感，也就意味着我们可能并不一定继续依附于Oracle。对我们来说，只要是一个标准的关系型数据库我们都可以使用，MySQL、Oracle都没有问题，甚至是其他的一些开源的数据库。

在这个基础上，我们还要解决一个数据容量的问题，数据拆分是一个必然的解决方案，我们的方案通常是水平和垂直拆分结合使用。目前，所谓水平就是把数据按照用户的维度进行一个水平的拆分。所谓垂直是把数据按照一些业务进行独立的部署。通过这种方

式，我们基本可以把数据的规模控制在一个比较稳定的范围内，甚至可以实现线性扩展。

InfoQ：我们都知在数据库性能达不到我们要求的时候，通常会考虑使用Cache。前面您的演讲中讲到了阿里巴巴国际站的一个Cache策略，能不能在这里更详细地给我们讲一下阿里巴巴国际站是怎么使用Cache的？

潘：其实阿里巴巴国际站在使用Cache跟使用其他技术时都遵循了同样的原则，即使用合适的技术去解决合适的问题，所以我们首先会对应用做一个划分，把我们的Cache定位到不同类型的应用场景。我们目前既有分布式的Cache解决方案，也有集中式的Cache解决方案，甚至还有内存Cache解决方案。我们会根据不同的应用场景分别应用这些方案，当然我们的Cache种类也比较多。当然我们还有一些阿里系特有的解决方案，把一些对性能要求比较高的数据载入到内存当中，形成内存Cache，同时这些数据有集中管理的要求，其实是一种可以集中管理的分布式内存Cache。大概是这样几种。

InfoQ：当网站发展到了一定规模的时候，肯定会涉及到负载均衡，还有多IDC部署，甚至是跨洲际的IDC部署。在跨IDC部署的时候，肯定会遇到数据同步的问题。在国际站，跨洲际IDC部署的数据同步是怎么做的呢？

潘：首先，跨洲际的IDC负载均衡还是一件相对比较简单的事情，其实就是通过DNS轮寻的方式，我们的轮寻方案也还是比较简单，根据用户所在的地区，加上一定的策略，去选择对他来说相对访问时间比较好的网站。

至于数据同步，现在多IDC之间，我们都有一个双向同步的通道，同时会针对不同的数据同步类型，设定不同类型的策略。国际站的数据同步总体而言，多个IDC之间所有的状态同步都是通过一套数据同步系统来实现的，无论是数据库，还是文件，还是一些Cache，它的实现原理基本上类似于一个完全消息驱动的系统，当然它比消息驱动要复杂的是在于我们需要在数据同步系统里面去处理一些异常检测，或者说业务数据的冲突检测等一些逻辑，所以这是一套比较庞大的系统。因为还涉及到一些比较具体的技术点，比如说一些Oracle的内置分析，或者说一些高可用的队列，所以总体而言，我们其实是用一套统一的数据同步的解决方案来解决多个IDC之间所有的数据同步，不仅仅是数据库的，也有文件的，也包含所有其他的状态数据。

InfoQ：现在很多网站都在追求高可用、高性能，对于一些刚刚起步的网站来说，您对他们有什么建议吗？

潘：我觉得，首先所有的网站设计都应该是从业务的角度出发，这是第一点，不能纯粹为了技术而去追求一个比较完美的一个目标；其次，很多问题在业界都有比较成熟的方法，作为一个刚起步的网站，应该尽量选择一些业界比较成熟的技术，去作为自己的基础架构。

当然，在此之外，我觉得作为架构师，还有一个很重要的素质，要解决所有碰到的问题，甚至直接去深入到一些细节，这样才能保证整个网站架构的可用性，或者说稳定性。这点我很想强调一下，不要放过任何一个碰到的疑难问题，往往一些小问题会对架构产生很大的影响。

InfoQ：您刚才提到了业务，其实在一个公司里，业务与技术应该是相辅相成的，作为一个架构师，不仅仅要着力于技术，更多的时候应该去做技术与业务的权衡，您作为国际站的架构师，有没有这方面的经验跟大家分享一下？

潘：首先，我觉得架构的事情，不能纯粹看成是一个技术的事情，因为一个好的系统，如果没有业务系统去配合，那它是很难长成或者说演化成一个相对完善的系统的，所以我觉得架构师首先要熟悉业务，甚至要擅长做一些系统业务的建模；其次，我觉得业务和技术的关系就是一个成本和收益的问题，业务就是我们的收益，技术就是我们的成本，所以不能完全从业务的角度，也不能完全从技术的角度出发去解决问题，应该充分评估我们的收益和成本，决定用什么样的技术去解决问题。我比较赞同一个观点，技术或者说架构其实是业务驱动的。

InfoQ：这次我们的QCon有幸邀请到了FaceBook和Twitter的成员，前段时间，Twitter开始由MySQL向NoSQL迁移，业界对NoSQL的讨论也非常热烈，我想请问一下您对NoSQL有什么看法？

潘：NoSQL最近讨论得很多，但是有一个观点是非常鲜明的，NoSQL并不意味着不要关心数据库，应该是Not Only SQL，或者是我们不仅仅需要关心数据库。对于阿里这样的一个网站而言，跟Twitter还有一定的区别，我们的业务类型更复杂，一些复杂的业务更适合用关系型的数据库去解决；但另一些业务则更适合用NoSQL的方案去解决，所以我觉得架构的问题，归结到底都是要选择合适的技术去解决合适的问题。

就目前的技术发展来说，我并不认为有任何一种技术可以解决所有的问题，我们可以看到很多业务，甚至占了我们很大比率访问量的业务都有希望迁移到NoSQL的架构上，

因为NoSQL的架构相对来说扩展性和分布式会做得更好。我觉得NoSQL是对关系型数据库的一个非常好的补充，但它不会成为其替代品，仅仅是个补充而已。

InfoQ：现在有一些NoSQL的产品，比较成熟的有MongoDB及Cassandra等等，作为一个架构师，肯定会涉及到技术的选型，如果让您选择技术框架的话，您会怎么样选择呢？

潘：我觉得要做选型，第一件事是要明确目标，要知道我要什么，这个目标会包含一些业务上的要求，也会包含从业务当中抽取出来的非功能性的要求。当然，非功能性的要求会有很多，通常我们会提到一些像可扩展性，或者说可用性。如果这时评估，从网站的角度来说，可用性和可扩展性，其实也很难说哪一点是最重要的，可能还是需要根据具体的场景去做一定的分析。甚至会出现这样的情况，我一直不排除阿里巴巴国际站会使用多个KV Store的解决方案。

仍然是那句话，合适的才是最好的。我可能会根据不同的业务目标，或者说根据不同的非功能性需求，去制订不同的技术的目标，然后去选择一个合适的一个产品。

InfoQ：现在您是一位架构师，没有人天生就能做架构师的，一定是一步一步成长上来的，我想很多朋友肯定都很想知道您是怎么成长起来的，对于那些希望能成长为架构师的朋友，您对他们有什么建议吗？

潘：我觉得要做架构师，首先要对技术有很强烈的意愿，你想做这件事，因为技术这件事，如果你不喜欢，它会变成一件很枯燥的事；其次，要善于学习，学习有两种，一种是从书本上，另一种是从和别人的交流里，你要去善于学习你自己不知道的东西，或者说你善于发现自己的短板，有针对性地做一些系统的学习。

要成为一个架构师，更重要的学习是来自于实践，曾经有人说过一句话，我觉得非常好，所谓一个行业的专家指的是什么？其实指的并不是他能力有多强，是指他碰到过了这个行业里面所有的问题，同时他解决了，他就能成为专家。同样道理，做架构师你就不要放过你碰到的任何技术问题，一定要钻研下去，知道里面究竟是什么，然后解决它。你就能在每一次解决问题的过程中得到足够的成长，所谓的架构师我觉得是一个水到渠成的事情。

这只是第一步，技术上我们要这样做，业务分析同样是架构师很重要的一个方面，要规划一个更好的系统，你必须要全面地看待这个系统。除了业务，在技术方面通常也可能会有一些误区，认为应用架构师只需要关心代码的实现。我觉得这是不够的，作为一个

架构师，他必须要关心技术的所有外延，比如说操作系统、网络、存储，甚至数据库，所有关联到的东西。作为一个架构师，应该去全面的掌握问题，当然深度你可以根据应用的场景，可能会因场景而异，或者因人而异。

InfoQ：非常感谢潘磊今天接受我们的采访，也恭喜您今天给我们带来一个非常漂亮的一个演讲，非常感谢！

查看原文：[潘磊谈阿里巴巴国际站发展历程](#)

王晶昱：消息系统架构与变迁

作者 水羽哲 发布于 2014年3月3日

对于大型的互联网业务来说，消息系统是必不可少的基础服务。子柳 在《淘宝技术这十年》中为大家展示了阿里消息系统架构的概貌，作为集团业务使用的核心基础服务，目前消息系统现在可以承受每天几百亿规模的请求，并在历年的双十一、双十二大促中承受住抗住了更加严峻的考验，消息系统背后的中间件团队还陆续开源了诸如 [MetaQ](#)、[RocketMQ](#) 等项目。近期，InfoQ 采访了阿里消息中间件团队消息和分布式数据层负责人王晶昱（花名：沈询），话题涉及案例中间件系统的选型、系统扩容与数据一致性、团队文化等内容。

InfoQ：对于阿里的消息中间件系统，大家所广泛了解的是@子柳 在《淘宝技术这十年》中介绍的 Notify，但是从最近的阿里的开源计划中，我们经常看到 [MetaQ](#)/[RocketMQ](#)，在阿里内部 Notify 和 MetaQ 是怎样的关系？我看到早期的MetaQ 是采用的Kafka的设计思路，那么可能大家就比较好奇“问为什么要重复造轮子”，能不能介绍这个方面的考虑以及所做的工作？

沈询：要讲明白这个问题，就需要从产品的实际需求角度入手开始做个介绍了。Notify 作为一个已经存在了5年多的消息产品，被广泛的应用在整个阿里巴巴集团的大部分消息通信领域。它的核心特性是： 提供事务支持、不保证消息顺序、消息可能会重复、推模型。

因为淘宝是个交易类网站，所以事务支持的特性能非常方便的让用户可以快速的依托于Notify完成他们自己的业务逻辑。

然而，一个产品不可能满足所有的场景，在当时我们就经常收到一些需要保证消息有序的发送和接收的需求，而这样的场景对于上来就定位于无序消息投递的 Notify 来说无异于釜底抽薪。

而正在我们讨论这类需求应该如何被满足的时候，正好赶上 LinkedIn 的 [KafKa](#) 队列开源，简单的文件队列，拉模型，保证顺序的特性一下就吸引了我们的目光，在对他的做了整体架构上的Review以后，我们认为这是个非常优雅的模型，因为他足够简单，简单就是最好的~！

然而里面也有一些特性不是我们所需要的，比如我们主要是面向内部用户，因此定期轮询去拉的方式就不适合我们的实际场景需求，并且因为 KafKa 的开发语言是 Scala，不大利于我们的后续的维护，因此我们借鉴了 Kafka 的核心思路，对其进行了重写并开源，当然我们还是向 LinkedIn 的 KafKa 做了致敬的，[MetaQ](#) 其实是 [Metamorphosis](#) 的意思，是 Kafka 的名作。

从上面的发展历程其实也能够比较清晰的找到两个消息队列产品的不同定位了：

- RocketQ(MetaQ) 主要面向消息有序的场景，能够提供更大的消息堆积能力
- Notify，主要面向需要更加安全可靠地交易类场景，无序，推模式。

InfoQ：您个人在分布式方面有比较多的经验，而消息系统中一个重要的考虑因素就是分布式的扩展能力，尤其是对于阿里、淘宝的业务，能不能介绍下目前中间件团队在分布式是如何做的？跨域、跨机房方面？

沈询：其实所谓分布式运算，核心的思路就是系统架构无单点，让整个系统可扩展。

如果要介绍分布式场景的实际经验，那么我就需要先引入一个概念：“有状态存储节点和无状态运算节点”。

无状态运算节点主要是部署 Web 应用、HSF 服务，消息队列等的机器有状态的存储节点主要是指部署数据库、缓存、配置服务器、NoSQL 等的机器。

那么针对无状态节点，因为不存储数据，请求分发可以采取很简单的随机算法或者是轮询的算法就可以了，如果需要增加机器，那只需要把对应的运算代码部署到一些机器上，然后启动起来，引导流量到那些机器上就可以实现动态的扩展了，所以一般来说在无状态的节点的扩展是相对的容易的，唯一需要做的事情就是在某个机器承担了某种角色以后，能够快速的广播给需要这个角色提供服务的人说：“我目前可以做这个活儿啦，你们有需要我做事儿的人，可以来找我。”

而针对有状态节点，扩容的难度就相对的大一些，因为每台 Server 中都有数据，所以请求分发的算法不能够用随机或者是轮询了，一般来说常见算法就是哈希或者是使用 Tree 来做一层映射，而如果需要增加机器，那么需要一个比较复杂的数据迁移的过程，而迁移数据本身所需的成本是非常高的，这也就直接导致有状态节点的扩容难度比无状态节点更大。

针对有状态节点的难题，我们提供了一套数据自动扩容和迁移的工具来满足用户的自动扩容缩容中所产生的数据迁移类的需求。于是，无论是有状态的数据节点的扩容，还是无状态的数据节点的自动扩容，我们都可以使用自动化工具来完成了。

在所有的节点都能够实现自动的扩容以后，整个体系就能够水平的进行扩展了。这种架构很好的支持了我们历年的双11大促，而且每年都有一些进步。然而，这套模式也不是万能药，在当下，杭州已经很难满足我们对机器的全部需求了。

为此，我们也在尝试进行异地数据中心的尝试，以期能够将一部分运算和存储能力搬运到异地机房进行。

提到异地数据中心，一般第一个会被想到的是 Google 的 [Spanner](#)，这套系统是一个跨数据中心的强一致数据库系统，然而我们在经过仔细的考量以后，认为在目前的情况下，使用这种方式并不能够解决一个大规模在线交易类网站对于高并发TPS的实际需求，因此我们选择了另外的方式来实现跨数据中心的事务模式。

其核心思想也很简单，即是将数据放置在距离用户最近的机房里面，并尽可能减少应用层的跨机房调用，以充分提高性能，降低延迟。

InfoQ：消息系统在保证数据的一致性方面做了哪些工作？

沈珣：一致性这是个说复杂，挺复杂；说简单，也挺简单的领域。要理解一致性，其实关键在一个“看”字，一致性约束的是一个用户写入并提交数据之后，其他用户去读这条记录的时候，要么看到的是事务开始之前的状态，要么就是事务结束后的状态，而在这两个状态之间的事务状态则不会被其他人看到。

我们以一个例子做说明：

李雷要给韩梅梅100元，那么，要么结果是韩梅梅有100元，要么是李雷有100元，而李雷减少了100元，但韩梅梅还没加上这100块的这个中间状态则不能够被其他人看到。

做到这个一致性的一般性做法就是把数据加锁，让某个数据只能被某个进程或线程访问就行了。但这样也有个代价就是锁住数据的时间越长，系统的并发程度越低，系统的tps也就越低了。尤其在分布式场景下，维持锁的延迟在加入了网络这个因素后，变得非常巨大，以至于很难接受。

因此在互联网行业中，大家普遍使用的方式是“最终一致”，也就是说，三种状态都有可能出现，但李雷减少了100元，韩梅梅却没加上100元这个状态，因为速度非常快，只有毫秒级，并且对用户没有太多的不良影响，所以就认为是允许了。

用户可见的状态，从原来的两个状态，变成了三个状态。

然而需要注意的是，最终一致并不意味着弱一致，也就是说，韩梅梅“最终”必须能够拿到这笔钱，能够拿到，就是“最终”一致，在异常状态下不能够拿到，那就是“弱”一致。

消息系统的作用，就在于能够将“弱”一致变成“最终”一致，保证多方数据的状态的最终正确性。

InfoQ：目前消息中间件的使用规模是怎样的？目前的吞吐量、负载如何？在中间件博客中提到了双12期间中间件在彩票活动中所起的总用，是否能够详细介绍下在两次大促期间，中间件团队所做的工作？遇到的问题以及应对方案？

沈询：基本上整个阿里集团都在使用我们的中间件所提供的服务，消息规模在几百亿每天，高峰期 load 在4~5左右。

在大促过程中，消息中间件主要起到了蓄水池的作用，投递消息的往往都是核心应用，机器会优先保证，处理能力强劲，而接受消息的应用则可能存在处理不过来的情况，因此业务请求会在消息中间件中做一次削峰的操作，从而可以保证后端系统不会因为前端流量过载而导致系统不可用。

InfoQ：目前所在的团队具体负责哪些工作？团队组成是怎样的？如何分工？业务扩展方向是怎样的？

沈询：阿里中间件团队，是国内为数不多的极具技术挑战性的团队之一，依托于全球规模最大的阿里巴巴电子商务平台所带来的巨大流量和海量数据，以及对于电子商务平台固有的稳定性要求，使得团队有机会去面对一个又一个技术难题，创造一个又一个技术奇迹。在刚刚过去的“2013双十一网购狂欢节”中，350亿元销售奇迹背后的每一笔交易订单都和阿里中间件团队的技术小二们息息相关。

中间件团队致力于成为中国第一、世界一流的Java技术团队。自主研发的一系列产品始于07年底开始的淘宝架构 2.0 到 3.0 的变迁过程中，使淘宝网从集中式的 Java 应用走向了分布式 Java 应用，涵盖了消息中间件、服务框架、数据层、应用服务器和大规模分布式稳定性平台等等。解决了淘宝网这个大型系统中的应用间以及应用到水平拆分后的数据库间的访问问题，通过消息中间件对应用进行了解耦并提供了最终一致性支持。目前广泛使用在大淘宝的各个Java应用中以及少部分的非Java应用中。而稳定性平台、性能优化平台是在淘宝系统分布式化后解决和稳定性、容量规划、降级管理、依赖告警以及性能丈量等方面的问题的利器。

中间件团队的同学是一群不安于现状且喜欢折腾的人，未必很资深但是很执着，充满热情。大家来自五湖四海，到这里一起解决技术难题，提升系统性能，完成业务突破，构建新的应用，玩儿转技术、业务、数据、无线。

如果你酷爱技术、喜欢钻研、愿意去帮助多个业务系统的发展，并且认为编程是别人不能剥夺的权利的话，欢迎加入我们，一起提升我们的技术产品，一起去支持业务更好的发展。

如果你希望在淘宝的土壤上，用我们的技术去折腾一些新应用，新玩儿法出来，也欢迎通过xiaoxie@taobao.com或者shenxun@taobao.com和我们联系

InfoQ：您个人的经历也是比较有意思，我看到在 **ADC 2012** 的分享中，您提到自己做过4年淘宝小二、参与淘宝所有去 O 项目、负责分布式数据库中间件团队、到现在负责消息中间件团队，能不能详细介绍下自己成长的历程？

沈询：我当年是被这团队的 Title “骗” 来的，当时中间件这个团队的名字非常唬人：“淘宝平台架构组”，不能不说平台架构这个名字对于刚毕业的学生来说确实是高大上啊~

其他的事情其实更多地还是在外部环境和自己的积累上。

能够赶上这样快速发展并且需求多种多样的应用场景，不能不说是一个程序员最幸福的事情，尽管也会偶尔为了解决一个线上问题在高度紧张下在线上操作到深夜等这类问题，不过当你回头看看，会发现过程本身其实就是一个积累的时刻，碰到并解决的坑越多，加上自己的一些思考与探索，自然就会有一定建树。

InfoQ：作为一个需要对阿里系的所有项目提供支撑服务的团队，在选择团队成员的时候一般都会从哪些方面考量？如何确保对的人作对的事？

沈询：首先，希望他是个技术人：毕竟这是个纯粹的技术团队，产品本身就是纯粹的技术产品，因此计算机体系内知识，编程技巧方面的要求是首要的要求。

第二，希望他是个品行端正的正人君子，能够准确的把握自己的实际需要，拥有自我管理的能力，并能与其他人进行合作，能够体谅其他人的难处。

第三，希望他是个能够独立解决未知问题的人、独立思考、不人云亦云、活用知识，能够解决实际问题。

在用人上面，我们比较关注每个人自己的主观能动性，毕竟兴趣是最好的老师，所以个人的需求会优先被考虑。当然，团队本身客观上会有一些不得不做的事情，不过这种情况下也会在充分尊重个人意愿的情况下做好沟通性工作的。

嘉宾简介：沈询其实是个花名，原作里叫沈洵，是个40多岁的白衣大侠，真实的我就是个普通的程序员，平时喜欢看一些杂书，天文地理海洋气象历史政治经济都喜欢读读，在技术上主要还是关注数据库相关的业界最新进展。

查看原文：[阿里巴巴消息中间件团队消息和分布式数据层负责人王晶昱：消息系统架构与变迁](#)

周涛明：跨境网站的优化与挑战

作者 水羽哲 发布于 2013年9月26日

随着公司业务的扩展，越来越多的企业实现了服务的全球化，但是由于各个地区网络差异，跨境网站的可用性以及性能问题越来越凸显，在今年QCon上海的知名网站架构专场上，来自于阿里巴巴的周涛明将会以“[跨境网站性能优化挑战和思路](#)”为主题分享阿里巴巴的实践经验。

我们今天也有幸邀请到了阿里巴巴网站架构师周涛明，听他谈谈阿里的一些技术实践以及将要在QCon上所分享的内容。

InfoQ：简单的介绍一下自己目前负责的工作，以及在自己相关领域做过哪些方面的工作，关注过什么？

周涛明：我目前主要负责阿里巴巴国际站性能优化领域。性能优化是一个完整体系，也是一个生态体系。在生态体系中，有不同的子系统和角色，性能优化之后，随着需求的堆砌，原来性能优华后的成果一般都难以维持，所以性能优化之后一般需要有性能保障子系统，在国际站性能优化体系建设过程中，不仅仅进行了多项具体的性能优化工作，而且为了保障性能优化的成果，建立了性能基线系统，基线系统是对性能的关键指标，如DNS解析时间，StartRender、首屏加载时间、Full load时间、服务端响应时间、应用峰值QPS建立了基线，当超过基线的20%，系统会报警到相关页面的负责人这里，负责人会关注并修复问题。

同时在性能优化生态体系中，仅有基线是不够的，当系统帮助发现有问题，但是不知道问题出在哪里了，为了快速定位到问题，需要有性能诊断子系统。性能诊断前端部分，我们建立了前端性能服务中心，提供数据服务，模拟浏览器行为访问我们自己的站点，将影响性能的一些因素例如请求数、Dom节点数、html大小、图片大小和数量监控起来，当性能变差时，系统会自动给出可能影响性能的因素，列举出来，这样定位问题

的效率就简单得多，服务端我们将URI Profile的监控做起来了，一个典型的页面流程，调用了哪些方法，这些方法耗时趋势是什么样子的，当RT和QPS超过基线设定时，通过URI profile监控可以知道哪个方法出问题，通过这种方式，能够快速进行性能的定位和专门的优化。

InfoQ：你目前关注的重点是什么？

周涛明：我目前关注的重点是让性能优化生态体系能够高效运转起来。毋庸置疑，性能优化体系运转起来，首先要进行性能优化，所以性能优化本身是重中之重，对一个新网站来说，性能优化主要关注在业务发展、用户体验，最后才是成本问题。例如对一个跨境电子商务网站，面向的是全世界的买家，流量的引入主要依托各种搜索引擎的收录，通过引擎的推广和搜索，引入流量，爬虫爬取量的多少，在一定程度上影响流量的流入，所以对于搜索引擎比较关心的页面来说，该页面的后端响应时间尤其重要，所以RT从50ms变成40ms，节省10ms，对用户的真实体验其实是完全可以忽略不计的，但是对爬虫爬取量来说，可能就是20%的提取量提升，所以针对爬虫类的网站，优化RT实际上促进业务的上涨。

对于前端性能提升，其实并不复杂，用一些传统的方法实际上可以达到很好的效果，刚才提到了生态体系，其实人为主性能优化体系的核心角色，实际是起到非常关键的作用，还是相信那句话，只要去做了，总会有些收获和你意想不到的问题存在，性能优化本身还是有些规律可以遵循的，找到这些规律并实践非常重要，实践过程中会遇到很多问题。今年的Velocity大会上，淘宝、腾讯、百度分享关于性能优化的东西，其实和我今天谈到的性能优化体系是一样的，他们会也会遇到相同的问题，在解决问题过程中，会得到很多体会，而这种体会和实践是提高和深入是性能优化能力提升的根本。

InfoQ：感觉在过去一年，自己接触到的、关注的领域发生了什么变化？

周涛明：过去一年，业界技术创新应该不是特别多，也有一些变化：

1. Java重回编程语言第一宝座。Java8由于存在安全漏洞的问题，原本计划13年9月份发布，推延到明年3月，Java8开始支持Stream集合类，支持fork和join并行方式进行任务的分解和加速处理过程，非常值得期待。
2. Java启动新的项目[Nashorn](#)，开发人员可以在Java代码中写JavaScript。

3. 使用[Tengine](#)的网站达到全球网站数量的千分之一，Tengine的高并发处理能力和越来越丰富的模块支持，受到越来越多工程师的欢迎。当然Tengine是基于Ngnix改装的，Ngnix同样值得尊敬
4. [Taobao Tsar](#)工具开源，监控的多维度支持更加丰富
5. Google推出免费的cdn加速服务——Google's PageSpeed Service，
6. Google推出的[SPDY协议](#)，目前还处在实验阶段。

InfoQ：我们知道雅虎有著名的14条军规，是否可以谈下在阿里你们所总结的实战经验？

周涛明：前端性能优化，按照经常使用的频率和效果上来看，在前端性能优化上经常用到的点如下：

第一，减少http请求是非常有效的措施

减少http请求是非常有效的性能优化手段，例如常用的css sprite技术将背景图片进行合并来减少网络开销，阿里巴巴主页就用到这个手段，合并ajax请求，也是减少http请求的手段，请求的发起和接收，比一个请求的网络开销要大得多，从目前线上的效果可以看到，减少http请求数是非常有效果的方式。

第二，减少重定向

重定向意味着两次连接，用户在访问一个页面时，意味第二次需要重启发起连接（keepalive会重用连接），第二次请求由于存在网络开销，会出现短暂的白屏，在跨境电子商务网站来说，由于各个地区的网络差异非常大，所以网络的开销是非常大的，在阿里巴巴跨境网站业务，会针对新老用户，做不同的页面展现逻辑，重定向是非常普遍的，也很容易被开发忽视，在某种程度上来说，重定向编程相对容易，所以在下单关键流程上出现了非常多的重定向的跳转，在Google Analysis上面可以明显看到重定向时间达到1s以上，去年做了一次全面的整改，把重定向全部去掉，改成forward内部跳转，减少网络开销，成效十分明显，关键页面性能 onload 时间减少1s左右。

第三，预加载

预加载是预见性用户行为的性能优化方式，根据用户的操作行为，在CPU处于闲置状态时，进行预先的资源加载，当应用达到预测的页面时，部分资源已经预先加载，所以性

能就提升了。在阿里巴巴B2C网站Aliexpress.com的Detail页面进行性能优化时，就在用户搜索页面进行预先加载用户搜索的前6个商品，目前支持的浏览器有 Chrome 和 Firefox。

其实超级简单了，就是加一个类似的链接，`href`可以是任何资源，HTML，JavaScript，CSS，图片：

```
<link rel="prerender" href="" > (Chrome)  
<link rel="prefetch" href="" > (Firefox)
```

第四，尽量减少cookie的大小

大量的cookie会加重请求的发送网络开销，http规范对头是不压缩的，对于跨境网站来说，网络差异非常大，所以过大的cookie的网络latency是非常大的，当然为了满足业务需求，cookie的大小也没有更好的方法，所以只能尽量减少，毕竟满足业务需求前提下，才能满足性能。

第五，延迟加载

延迟加载，将资源延迟到需要的时候的加载，例如detail页面，相关产品推荐，当用户浏览更多的信息往下拉动滚动时，才进行加载，异步加载可以大幅减少对后端资源的使用，在需要的时候加载，是资源合理使用常用的方式，但是也带来一个问题，当往下拉才去加载，如果性能不够好，用户的体验其实是不好的，“菊花”转动的时间会比较长，同时异步加载对前端性能的作用也是非常明显的，渲染的节点数量大幅减少。

第六，Ajax异步化减少DOM的节点数，加快渲染时间，first byte时间

异步化通常是首屏加载优化，加快dom渲染速度（减少dom数量），异步化带来了firstbyte性能提升，用户感知页面有内容更快了，异步化同样也会带来用户体验的问题，大量的异步化，js的阻塞渲染（下载）的概率越大，所以适度的ajax很重要，用户体验是第一位的。

第七，CDN加速

Cdn加速其实是大型网站都要用的手段，cdn消除了地区间的用户访问差异，让用户就近访问，cdn需要注意的是需要尽量减少回源（不命中cdn），资源的过期时间尽量长，合

理的cdn架构也很重要，回源之后，文件处理过程也很重要，坚持一个原则，那就是尽量短的latency。

第八，延迟渲染

异步加载带来的问题是，需要发起请求到服务器拿数据，如果网络条件差，用户体验影响是非常大的，延迟渲染就是在需要的时候渲染dom，渲染html片段。例如搜索list页面，有32个搜索结果，前6个商品（首屏），是同步渲染出来的，其它的商品列表的html，用textarea进行包装，浏览器会把这种带有text area标签的html片断当做一个节点来看，所以渲染速度大幅提高，这种方式是提高首屏的渲染速度，达到了性能和用户体验的平衡。

第九，DNS预解析

Dns解析容易被人忽视，对于跨境网站，local dns位于世界各地，离机房的dns server是非常远的，解析成本实际上是非常高，特别是域名多的情况下，提前预加载dns，相当于在a页面提前把b页面的所有的域名提前加载进来，这样到达b页面时，dns几乎不需要解析了，dns解析的提前加载，提高了首屏full load、firstbyte，用户体验明显增加。

第十，js放在头部阻塞浏览器并行渲染

JS尽量不要放在上面，尽管现代的浏览器已经做了充分的优化-大部分情况下，已经不阻塞并行下载了，但是在很多情况下仍然会阻塞并行渲染

InfoQ：看到你对SPDY协议比较关注，如果畅想下这个技术得以普及，会对你现在负责的系统会有哪些影响？你怎么看待这个技术？

周涛明：Google的SPDY协议是HTTP的增强，它主要出发点是为了减少网络延迟，nginx已经支持，个人觉得它最大的几个优点：

- 多路IO复用 http协议的并行下载是利用客户端和服务端建立多个连接进行并行资源的下载，创建连接的开销和维护连接的成本是不低的，spdy运行一个连接，并行进行资源的下载，并且可以对请求划分优先级，对css等重要资源安排更高的优先级，在连接处理的效率应该说有比较大的增强。

- 对HTTP请求头和响应头的压缩 这个在cookie非常大的情况下，非常有用，压缩头部能够带来更小的网络延迟
- 冗余的请求头信息 在http协议的交互过程中，user-agent，还有其它的很多信息每次传输基本上不会变，这个细节的优化同样是非常好的。

但是spdy协议和http2.0协议并没有达成一致，spdy限制必须使用ssl，这个是个问题，可能这个是限制spdy协议发展的限制之一，毕竟不是所有的网站都需要那么高的安全标准。

Spdy目前还在实验阶段，加上和http2.0协议的不一致，所以对spdy协议还需要观察，tengine是基于nginx的，我们现在网站大多都是基于tengine的，一旦spdy准备就绪，我们随时需要准备着，这个是变革性意义的协议，也希望http2.0协议早日和spdy协议求同存异，互相汲取优点，让http真正快起来。

InfoQ：对于一个业务刚开始起步的系统来说，性能优化的时机和度怎么把握？从你的实际经验出发，推荐大家多从哪几个角度出发做优化？要尽量避开哪些方面（比如优化难度大，性价比不高的）？

周涛明：性能优化有一点非常重要，那就是顺势而为，这个势其实就是业务本身的发展趋势，例如需要提前对业务的发展数据作出预估，商业上一般都有这个数据，例如网站半年以后的增速大概是3倍，那就需要提前考虑3倍的访问量增加对用户体验对机器成本，对业务有没有可能的阻碍。业务发展初期，为了促进业务的发展，需要满足各种需求，例如提升用户购买率的需求，各种转化率的需求等等，基本上大型的网站的发展过程都是在满足功能的情况下，等网站发展到足够大的时候，才会花更多的精力去考虑性能的事情，例如为了减少机器成本，需要做单机峰值qps提升的性能优化，个人认为下列情况下需要做考虑性能优化：

1. 优化到什么情况是最合适的？

这个是根据网站的特点来，每个网站的转化率，流量和用户的特点是不一样的，在一定的性能情况下，性能的提升，一定会转化成对商业数据的影响上，这和理论上的数据还是有些差距，性能优化无极限，把商业数据拿出来看，可以度量我们什么时候才算是把性能优化的事情可以告一段落，例如我们曾经做过测试，将首屏时间优化到3s以内，发现转化率和流量处于停滞状态，这个时候性能上发力其实是无济于事的。

2. 优化从哪几个角度优化？

网站首要的考虑的是对用户体验的影响，影响网站的体验，性能只是一个方面，纯从技术上来说，个人认为可以从以下几个角度：

2.1. 确定重点

优化对用户体验影响大的点，例如，detail页面，需要确定哪块是重点，例如首屏，关键数据区的优化。重点关注用户关键流程上的点，对于电子商务型网站来说，购买流程中的页面时关键的。在时间有限的情况下，理清重点很重要。

2.2 确定相关重点页面的关键性能指标

1. 首要考虑用户能够感知的指标，例如首屏加载时间，如果没有首屏，需要考虑分屏。
2. 关注用户操作过程中关键点的性能指标，例如跨境网站的物流运费计算，一定要足够快，用户最好是能够很快看到内容，而且是关键内容。

2.3 需要常态化优化的跟踪机制

性能优化不是一次性的行为，一定要有常态化的机制去保障，例如我们是把关键页面的性能建立了基线，优化了之后，把优化之后的结果作为基线，超过基线的多少，需要有机制保障触发再次优化。所以性能优化一般需要有强有力的监控系统，对于前端来说，可以结合前端性能监控系统，可以自己做，也可以借助第三方。对于后端，可以结合后端的关键性能指标例如RT，QPS，对这两个最基本的关键指标建立基线，超过多少，触发优化。

2.4 需要有完善的配套的性能诊断体系

个人还是认为，性能优化是一个体系，性能优化不是一次性的东西，是一个完整的生态体系，这个体系包含，做性能优化的人，性能基线体系-保障性能优化成果，提升性能优化效率的性能诊断系统，和流程机制保障。性能超出了基线设定，需要快速告诉开发人员那里出了问题，例如后端一个关键页面的RT从100ms变到200ms，仅有整个页面的RT不能高效低排查问题，所以最好是有方法级别的监控，告诉开发人员，哪里的方法有衰减。对于前端，可以把影响前端性能的关键数据例如http请求数，页面大小，图片大小和数量等监控起来，进行数据分析，建立基本的诊断系统，非常有必要。

3. 应该避免哪方面的优化？

一， 需要防止伪性能优化

伪性能优化指的是不考虑用户体验，只注重看似重要的性能指标而做的性能优化，这是掩耳盗铃式的优化，例如关键页面的首屏，用户看到的关键数据需要等待很长时间，看起来首屏是很快，但是关键数据需要等待，其实对用户体验提升完全是没用的。所以ajax的异步优化要特别注意这一点，非常容易造成用户体验不佳。

二， 需要防止过度优化

过度优化的特征是投入产出已经不高，而且效果非常不明显，其实性能的调整和提升，会造成一些负面的问题，性能提升，往往需要通过架构上的调整，一旦架构上调整、排查问题、系统的可维护性都会变差，看投入和产出：为了用户体验，把负责留给网站自己来解决是权衡；为了不必要的性能，引起架构的可维护性变复杂，是没有必要的。下列情况属于明显的过度优化：

1. 用户体验无明显提升，用户没有感知
2. 投入大，收效小，例如系统第一次优化qps从100提高到了400，第二次优化从400提高到402，节省机器2台，投入30人日，这个属于明显的不合理的性能优化。

需要防止过度优化！

嘉宾简介：周涛明，阿里巴巴网站架构师（@周涛明），11年加入阿里巴巴，在阿里巴巴之前，曾在思科工作5年，目前负责国际事业部性能领域，专注国际事业部B2C国际技术部的性能优化工作，网站性能优化领域专家，在大型网站性能架构有丰富的经验，致力于性能生态系统和跨境电商网站性能解决方案的研究，对性能相关的知识体系如操作系统内核，网络，服务器性能调优，架构优化，前端相关性能知识体系有浓厚的兴趣。未来两年的愿望是想把性能方面的经验和实践，汇成一本对一线工程师有实际帮助的书籍。业余时间他喜欢羽毛球运动，坚持5年每周两次羽毛球运动从没有间断过。

查看原文：[阿里巴巴网站架构师周涛明：跨境网站的优化与挑战](#)

张旭升谈ITIL及运维架构

受访者 张旭升 作者 霍泰稳 发布于 2012年1月4日

个人简介 张旭升（Peter Zhang），双CCIE，资深网络架构师，从事十余年网络基础平台建设和运维工作，先后在电信运营商、系统集成商工作，曾负责国内多个省份的电信运营商骨干网和城域网的设计和实施工作，具备丰富的骨干网设计、实施和项目管理经验。

InfoQ：大家好，我现在是在阿里巴巴园区采访阿里巴巴的网络技术专家张旭升。旭升你好，请先做下自我介绍？

你好，我叫张旭升，08年加入阿里巴巴公司。在加入阿里巴巴之前呢，曾在国内的电信运营商以及一些系统集成商中工作，主要负责一些省内的省级的骨干网的一些设计和实施工作。在加入阿里巴巴之后，主要负责网络架构方面的工作，主要是推行网络的标准化和自动化方面的工作，大致的情况就是这样。

InfoQ：在你们目前的运维流程当中，是否引入了ITIL或者类似的处理机制。请给大家简单介绍一下ITIL这个概念。在你们目前的运维流程当中，是否引入了ITIL，或者类似的处理机制？

是的，我们现在在用ITIL这套管理体系。那么我认为ITIL这个东西，是根据每个企业自身的需要来决定的。那么当一个企业在发展初期时人员比较少，可能通过人和人之间的沟通就能解决一些问题。那么当一个公司发展到一定程度的时候，特别是在人员，规模比较庞大的时候，靠人和人之间的沟通就会产生一些问题。只有靠系统来维护这样的一个体系。这时，我们开始引入ITIL流程管理的概念。由于每个公司的管理体制可能都不一样，那可能会根据每个公司自身的需要来定制一些东西。我们也是从这方面去考虑的，根据我们自己运维的一些需要。开发了一套这样的系统。

InfoQ: 那是否有基于ITIL对运维团队的考核体系，如果有的话，考核的重点是什么？

是的，我们ITIL的考核管理体系，大概是从几个方面进行考核的。阿里巴巴作为一个上市公司，是全球的B2B电子商务的领先者，那么作为我们的运维部，保证完整的可用性是首当其冲的。我们的ITIL考核体系里面，把完整的可用性是放在第一位的。除此之外，对变更的一些管理的质量，还有事件处理的质量，监控的质量，以及问题管理的质量，从这几个方面进行考核，其中可用性是重中之重。

InfoQ: 那运维团队与公司的其他部门，比如说研发，市场，客服这些部门之间的关系是如何处理的？

这个可能跟每个公司的这个组织结构有关系，从我们公司来讲，我们的运维部主要面对的是我们的开发团队，像我们监控团队会面对一些客服团队。那么我们开发团队也会面对我们的产品部门，产品部门面向我们的销售部门，销售部门面向我们外部的客户，是这样一级一级推上去的。我们部门面对更多的是一些开发部门。所以说像您所说的这个问题，可能会产生一些资源上的一些竞争，可能这样相对来说比较少，如果有这样的竞争的话，一般我们会根据项目的优先级，包括项目的重要程度和项目的紧急程度来进行取舍，或者是决策。一般来讲，我们是自己能够做出一些判断，如果我们判断不了呢，就会由我们团队的Leader再到我们的总监，再到VP一直到我们的CTO为止，来最终做一个决定。一般来讲我们绝大多数的事情，总监级别以下基本上都能够处理了的。

InfoQ: 那我们回忆一下，在你的作为运维这个架构师的一个过往的发展当中，有哪些比较值得回忆的时候，或者是比较骄傲的成绩呢？

这个是比较的，因为我是从08年加入阿里巴巴的。那么在我加入公司的这个过程中，我也曾经负责了一些比较大的项目。那么从我个人来讲，我印象比较深刻的有那么几件。

第一个是在09年的时候，我们的ABTN (Alibaba Backbone Transmission Network) 网络重构的时候，在这个项目里，其中有一个月的时间，大概需要有十几次的重大变更，这个重大变更意味着如果发生意外的话，可能会导致我们整个阿里巴巴网站，甚至整个集团的网站全部瘫痪，重要性是非常高的。我们把这样的变更一般都放在凌晨。那么连续

十几个重大变更放在集中一个月的时间里面，也就意味着每个礼拜大概需要有两到三天的时间，那么每个变更大概需要五到六个小时。那么在那一个月里面就基本上是要经常的通宵的，那这个对我来讲可能会记忆比较深刻，那虽然是很辛苦，但是我觉得还是蛮值得的，因为最后项目也取得了成功，我觉得这个是记忆比较深刻的。

第二个事情，我觉得印象比较深刻的，是在去年的时候，一个双休日，我刚好在家里面休息，在半夜的时候，大家都休息的时候，突然接到监控部门的通知说我们的网站被攻击了。这是毫无疑问的，我们作为运维人员肯定是要冲在第一线，马上去处理这次的事件。最后我们通过一个晚上的处理，把一些流量清洗掉了，维护了我们网站的可用性，这个我觉得印象也是非常深刻的。

第三个事情，因为在运维的过程中，这种事情是很多的，我相信在每个公司来讲，都有它的运维部，或多或少都会有这样的一些事情，那么我们常常讲运维人员一般都像是手术师一样，每天都在做不同的手术，有些是大手术，有些是小手术，都是在生产环境上做一些这样的手术。那么每一个手术带来的是对网站来讲都是非常重要的，而且如果做得不好，都会产生重大的影响，那么我觉得整个工作运维的工作可能就是这样的。那么如果说有成就感的话，那我觉得几年来，我觉得从我个人的体会来讲，那么阿里巴巴作为一个承载着数千网民企业客户，中小企业客户这样的一个大型的网站，服务了这么多的公司，那么作为这样一家公司的运维人员我觉得是一件比较骄傲的事情。从其他方面来讲，我觉得骄傲的事情还是蛮多的。

从公司99年成立到现在为止，我们整个运维团队已经有70多位成员，为我们整个阿里体系，其他的分公司以及我们的业界培养了非常多的相关领域的技术专家，这个我觉得也是一个比较有成就感的事情。第三个有成就感的事情，我觉得作为我们网络团队，因为我自己在网络团队，这方面可能更有切身体会一些，我们牵头设置和建设了国内第一家运营商级别的电子商务骨干网，叫ABTN，阿里巴巴骨干传输网，以及国内第一家全球级的电子商务骨干网，叫AGN (Alibaba Global Network)，阿里巴巴全球骨干网，为我们全球的客户提供了更加优质的服务，更提高了我们网站的可用性，这个从我们的运维角度上来讲，我觉得也是一件比较骄傲的事情。

InfoQ: 这确实是非常令人骄傲的，在整个的一个过程中有没有现在让你回想起来还比较尴尬，或者是比较痛苦的那个地方？

这个比较尴尬和比较痛苦的，我觉得作为运维人员可能都会有这样的问题。因为对我们的运维人员来讲，首先第一条就是要确保24小时开机，无论在何时何地，碰到了问题，接到故障，就必须第一时间去处理。比如像我们在双休日在家里休息，陪着老婆看着电视，或者是在休息睡觉的时候突然接到电话说网站出故障了，这时毫无疑问，必须起来去处理，有些甚至要到机房去，我觉得是比较痛苦的，但是这也是必须的。

InfoQ：我们问一个和技术相关的就是运维团队在面对很多相关的工具要做选择的时候，你们是基于什么样的原则去选择这些运维工具的？

一般来讲，我们可能会看这个工具是不是足够的灵活、方便，是否适合我们来使用，还有管理方面是不是适合后期的维护，另外还有一个，我觉得要看，如果使用这个工具我们需要投入多少的成本，会从这几个方面去考虑。从我们现在过往的使用的一些经验来看，我们可能也会遵循一些互联网的精神，比如像开放的精神，分享的精神，我们尽可能选择基于开源的软件来使用，包括像我们系统管理层面。早期的时候我们会使用kickstart12: 21这种开源的系统来做一些自动装机的工作。后期我们随着我们工作量的提高，或者是网站规模的增大，我们会有需要自行开发一些跟自己比较相适应的一些软件。像系统管理层面，目前还是在用CFengine这套安全系统。在网络管理层面呢，我们也在基于自行开发的基础上做了一些自动化的工具，同时也会选择一些开源的自动化软件，如果不是很适合的话，我们也会选择一些商业的产品，基于SMP，或者是Netflow13: 12这样一些商业化的软件来协助我们做网络的管理，大概是这样的情况。

InfoQ：像你们自行开发的产品或者软件，有没有计划去开源。另外，就是能不能给我们其他的运维人员提供一些建议，关于一些通用性的工具方面，比如有没有可以推荐的工具？

在这一块我们其实一直以来都在考虑这个问题，因为我们公司本身的精神就是要开放、分享。我觉得在合适的时期，我们也会考虑去把我们开发出来的一些工具炫耀出来。那如果说要推荐一些通用性的工具，从我们过往使用的一些经验来看，像装机系统，包括像kickstart、CFengine一些系统管理工具，包括像MRTG（Multi Router Traffic Grapher）的网络管理工具，这些工具我觉得都是比较值得大家去使用的。

InfoQ: 运维人员他需要自己去编写自动化脚本来完成很多运维的任务，你们实施这种脚本语言有没有去写这种语言？

我们现在是从规划来做这些事情，那么自动化的系统，我们现在用的比较多的是Python，当然还有Perl这种脚本语言。那我觉得还是像Python的特点是支持的数据类型丰富，容易开发，具有比较好的扩展性，不管是发一些小的脚本，或者是发一些应用我觉得它都比较合适。那么像Shell的话，相当于把Unix的命令进行一些堆积做一个批量的处理。但是呢，他有一个问题就是他可能支持的数据类型不是那么的丰富，那这样的话，在做一些简短的脚本的时候，可能会比较合适去用这样的脚本语言。但是如果要开发系统的话，可能他会去，缺少一些这方面的一些功能。那么像Perl的话，也是跟Shell有点类似的，支持的数据类型相对会比较少一点。它的语法上可读性会差一点，后期的维护上会稍微困难一些，Perl的话，我们会用的相对还是比较少，Python相对来说用的还是比较多的。大概情况是这样。

InfoQ: 那在性能优化方面，在网络操作系统级的优化方面有没有比较好的通用的经验和大家分享？

说到性能优化的话，我觉得其实是一个比较大的话题，性能优化是一个长期的过程，是在工作的过程中不断地积累经验的一个过程。那么在我们过往呢，性能优化的这个过程中也做了非常多的事情。比如说像网络层面，我们通常会从架构上，从流量上，从监控上，以及管理上，这几个方面会去做一些优化。那么我们做得比较多的一些事情。比如说像网络的标准化，还有像我们的大二层（L2MP、OTV），还有去二层的环路，还有双机的集群提高网站可用性方面，在这方面下了很大的工夫。然后在一个自动化的工具开发上。在提高整个网络可用性的基础之上，提高大家的工作效率，这个是我们做的一些优化的工作。

从操作系统以及服务器的优化方面来讲，因为目前大多数网站，基于Java的还是比较多，那么Java它有一个比较好的一个特点，就是它前端展示比较好，开发起来，相对也会比较容易一些。但是它有一个缺陷，就是他很难充分发挥CPU的利用率。我们知道现在新型的服务器都是四核以上的，很多都是高性能的服务器，但是它不能充分的发挥这些服务器的性能。通常的解决方案就是说做一些虚拟化的一些工作。原来在一台服务器，比如说只提供一个服务的，那么我们如果把它虚拟成四台，我们就可以同时提供四

个服务，那这样的话，我们可以把服务器的性能给充分发挥出来，这个是我们在服务器的性能优化上所做的一些工作。

另外一个方面，我们也会去结合我们的平台部门，还有开发部门去做一些网站的代码的一些优化，那么去年我们网站平台也做了一个非常大的项目，就把我们网站的一些代码做了一个大的优化，据我了解可以降低1/3的服务器的利用率，这个也是对整个服务器的利用率的一个性能的优化做了很大的贡献。

InfoQ: 那像那种涉及到一些代码方面的优化，是运维团队去推动还是研发团队自己去推动？

可能都会有，我们团队也会，如果发现有这样类似的问题，比如说像我们数据库部门，他们也会发现一些，有些部门可能对数据库的使用不是很恰当，那他们也会提出一些自己的建议。当然我们那个平台部门可能是主导这个事情的发展，最后会有我们的平台和开发最终来落实到这个方面去优化。

InfoQ: 刚才你也提到了虚拟化，现在云计算也是比较热的一个话题，那对于云计算它能够给运维团队带来的影响，你是怎么看待这样的问题的？

云计算的话，现在是大势所趋，大家都在研究云计算。那我觉得云计算最大的一个特点，它就是弹性的计算和按需索取我们的网络资源也好，其他的一些服务资源也好，这是一个整体的方向。那么我们另外有一家兄弟公司叫阿里云计算公司，从09年开始成立的，他们主要就是研究这方面的一些技术。到现在为止，一些服务已经在用了，包括我们自己B2B公司，还有我们其他的兄弟公司已经在使用他们一部分的云服务了。如果说云计算对我们运维团队的影响来讲，我个人觉得可能会有两个方面，第一个方面，我觉得作为运维人员，保证网站的可用性当然是第一位的，当然我觉得可能最重要的要去更多的去学习新的技术，要有创新的精神，然后不能固步自封，要提高自己的技术性能，技术能力。否则的话，随着一些新技术的发展，就会被这个时代慢慢淘汰，就像我所说的“逆水行舟，不进则退”。第二个方面的影响我觉得，可能我们也会去考虑，作为我们自己一家公司来说，我们是不是也可以把我们的网站设计成一种稍微相对来说规模小点的云提供给我们的客户来使用。把我们这朵小的云结合到我们阿里云计算那朵大的云上去，共同来提供服务，这样会不会，有更好的效果，这个是我们可能以后去研究的一些东西。

InfoQ: 现有的运维人员是如何培养起来的，是不是有专门的针对运维人员的一些培养的规划。然后另外的话，就是一个好的运维人员需要具备哪些重要的特点和技能，能不能举出三个或者五个的一个这样的例子？

我们现在来讲，我们以前的一些培养的经历，最主要的手段或者是方法，一般都是通过一个老人带一个新人这样一对一的模式去带动他的。一般会通过一些具体的项目，然后通过一些平时的一些工作上的习惯上，包括你自己跟团队之间合作的一些方法上面，还有一些技能的掌握上，和工具的掌握上，从这几个方面去培养我们的新人。

从长远来讲呢，我觉得，当然我们公司有一个具体的一些人才的培养计划，那么主要会有三个方面，第一个就是说，我们的新人的培养，这个是着眼于短期的。那么新人的培养主要是我刚才讲过的，通过一个老人带一个新人，我们叫师傅和徒弟的关系，一对一的这种手把手的去教导他，去指导他去做一些日常的一些工作，提高他一些满足我们工作所需的一些技能，包括对我们团队的一些文化的理解，对一些责任的一些理解，那这个是一个方面。

从中期来讲，我们会组织一些团队内部的一些分享，还有一些跨团队的，甚至是跨公司的一些技术分析，我们公司也会相对出一些激励措施来激励大家多分享自己的一些技术上的东西，在分享的过程中提高了自己，同时也提高了别人，这是一个比较好的良性循环的过程。从长远来讲，我们公司也在做一些事情，包括人员培养的一些计划，我们具体做的一些工作主要是在一些领域的研究方面，每个团队会有一些技术专家来规划他们相应团队的一些技术研究的方向，制定出一些领域研究计划，最后会选拔出相应的该领域的一些领域带头人。由这个领域带头人来设计该领域的一些具体的研究的目标，研究的计划，和具体的一些实施的步骤。在这个过程之中，每个团队的人员也会根据自己的兴趣爱好，去加入一些领域的研究。在这个过程之中，大家研究的过程之中提高了大家的技术性，技术能力，这是一个长远的一个计划。

InfoQ: 好的运维人员他应该具备那些最重要的特点和技能，比如列出三个到五个？

我觉得作为一个运维人员有三个，如果说要列的话，我觉得三个最重要的，首先是责任。第二个是技术能力，第三个是团队意识。为什么这么说呢？特别是运维人员，保障我们网站的可用性是第一位的。如果说网站都不可用了，后面的一切其实都是空谈。那么怎样才能保证这个网站的可用性？我觉得除了你工作的方式方法，除了你的技能，最

重要的是你要有一份责任心。你在做具体的项目的时候，或者是做具体的一些变更的时候，要真真切切的去考虑这个问题对我们网站带来的影响，这个是责任心的事了，还有一些比如说当我们网站，我们设备出了故障的时候，你是不是能够积极的承担起责任来，积极的去做一些维护的工作，尽快的恢复网站的可用性，这个责任是第一位的。第二个就是工作的一些技能，包括要不停的学习，因为IT行业是一个不断发展的一个行业，而且更新换代，技术的更新换代非常快，需要不断的去学习，不断的创新，要跟得上时代发展的需要，那么技能我觉得也是非常重要的。第三个我觉得就是团队意识，要有团队的合作精神，要善于和你的同事们一起工作，包括去帮助同事们成长。还有一个就是不能只顾自己眼前的一些利益，只保证自己的这一块一亩三分地，要尽可能去帮助你的同事，去帮助解决他的问题，不能把这些划得太清，这个是一个团队合作的精神，我觉得这三点对运维人员应该说是最最重要的。

查看原文：[阿里巴巴张旭升谈ITIL及运维架构](#)



精彩早知道

一个程序员的创业寻梦坎坷之路

前JavaEye网站创始人 范凯



开放式移动端平台架构设计

阿里巴巴-研究员 岑文初

程序员与黑客

知道创宇技术副总裁 余弦



开放式移动端平台架构设计

IBM Multi-tenant JVM项目技术负责人
李三红

Netty架构剖析和行业应用

华为平台架构师 李林锋



技术人该做什么样的产品

上海泰尼网络科技有限公司创始人 郝培强

支付安全的实践和思考

小微金服（支付宝）安全专家 郑歆炜



Spark应用案例分析

Databricks软件工程师 连城



网购狂欢节
11·11



Tmall.com

2

淘宝和天猫相关技术

曾宪杰谈JAVA在淘宝的应用

作者 丁雪丰 发布于 2011年7月14日

在7月10日举行的[淘宝技术嘉年华之淘宝技术专场](#)上，来自淘宝产品技术部[中间件团队](#)的[曾宪杰](#)（花名华黎）为大家介绍了近几年Java在淘宝的应用情况——[《Java@Taobao》](#)。

演讲之初，曾宪杰先分四个阶段介绍了淘宝的建构变迁：

- 2003年5月至2004年5月，小而快的简单架构，基于LAMP，符合当时的需求。
- 2004年2月至2008年3月，一个懵懂的阶段，开始分为多个层次。这个阶段需要一个能够支撑百万到千万用户级的架构，必须容易扩展；系统从WebLogic迁移至了JBoss；开发了大量软件，例如[TFS](#)、iSearch、TDBM和CDN。
- 2007年10月至2009年11月，开始有前瞻性，走向产品化及服务化。能够支持大型团队的并行开发，系统逐步模组化、中心化，可快速扩容，可用性大大提升，基础软件也开始走产品化道路。非核心的数据由Oracle迁移到了MySQL上，构建起了消息系统和服务框架，[淘宝开放平台](#)（TOP）正式上线。
- 2009年8月至今的淘宝则更系统化、智能化及专业化，这是发展的必然方向。知识经验慢慢融入工具之中，降低了门槛，减少误操作几率；操作从人工处理逐步转为系统自主决策；在稳定性和性能方面有长足发展（在2010年以前eBay的稳定性要好于淘宝）。

曾宪杰指出所谓大型网站就是要同时满足高访问量和搞数据量的要求，核心是通过分布式系统解决数据的处理、存储及访问问题。随后，他根据淘宝网站的结构示意图，分别介绍了其中涉及到的Java基础技术产品。

首先是在阿里集团内被广泛使用的Turbine风格的MVC框架——[WebX](#)，其核心代码由阿里巴巴18创始人之一的周悦虹（宝宝）编写，具有很好的层次化、模块化特点，高度可扩展，WebX还对Velocity模板做了编译优化，在一般情况下能有20%至50%的性能提升。

其次是中间件中的服务框架，负责服务的发布、查询、调用和治理。该服务框架使用上简单透明，支持软负载（没有中间层，服务使用方直接连接到服务提供方，使用服务注册查找中心进行管理），灵活可控，方便扩展，为保证稳定性提供了有力支持。

接下来是实现系统松耦合的消息中间件Notify，这是一个高性能、高可靠、可扩展组件，支持最终一致性和订阅者集群。所谓订阅者集群，即将订阅消息的客户端分为多个集群，集群之间采用Topic方式，让每个集群都能收到消息，集群之中再按照Queue的方式，仅由一个客户端来处理消息。

在数据层上，为了更好地支持分库分表以及读写分离，也做了一定的封装，对上层应用而言还是在操作JDBC，实际则是在使用淘宝分布式数据层（TDDL），它能实现SQL解析、规则路由、数据合并；既可以用jar的方式在客户端直接连接数据库，也可以让客户端通过DBProxy服务器访问数据库；还支持非对称数据复制。如果将众多数据源看成一个矩阵，横向是同一数据库的主库与从库，纵向是不同的数据库，TDDL将数据源分为三个层次——TAtomDataSource，封装单个数据源，将配置进行集中管理；TGroupDataSource，封装横向的多个数据源，支持权重和节点的增减；TDataSource，管理整个数据源矩阵。在具体的场景中，系统可以选择配置的粒度。

在分布式存储方面，淘宝基于[HBase](#) 0.90.2做了一些扩展，主要是提供了一个运维页面，修正了Master节点恢复时间过长以及备用Master无法自动接管的问题，并做了一些优化。[林昊](#)在下午的*DataForum*上就HBase做了更详细的[分享](#)。截止两周前的使用情况：

- [一淘](#)拥有100台服务器，已使用60T。
- [数据魔方](#)拥有10台服务器，已使用500G。
- 交易日志拥有12台服务器，已使用360G。
- UDC拥有8台服务器，使用600G。
- 此外还有众多应用，例如历史库搜索等。

搜索方面，基于[Lucene](#)和[Solr](#)开发了终搜，可进行中心化的配置管理，容易接入，支持多种Dump机制。淘宝还拥有国内最大的[Hadoop](#)集群（云梯I系统），总容量超过1400台服务器，约30PB，利用率55.2%。主要做了JobTracker的异步化，NameNode优化，存储优化以及小作业优化。为了提升稳定性，除了哈勃（使用[Cassandra](#)存储数据，由云梯处理数据）对全网服务器基础数据进行采集分析以外，淘宝还有CSP监控，它主要监控核心应用，协助进行容量规划，以及服务的依赖于降级。目前CSP一共有15台服务器，15个库，对200多个系统进行监控，每日采集数据超过5000万行。淘宝如此大规模的系统的运维也是由系统完成，目前在系统的部署、发布、监控方面已经做了很多工作，在运营管理方面稍有欠缺。演讲最后，曾宪杰列举了众多各有特色的业务系统，分析了它们的架构，说明了各基础技术产品在这些系统中的应用。

查看原文：[曾宪杰谈Java在淘宝的应用](#)

王海亚：淘宝交易系统演进之路

作者 郭蕾 发布于 2014年6月25日

淘宝的交易系统承载了购物车、下单、订单管理等多项淘宝的重要业务，随着淘宝业务量的不断上升，交易系统也随之几经改造。InfoQ此次专访了阿里巴巴架构师王海亚，分享淘宝交易平台的架构演变及并行化实践。

InfoQ：淘宝的交易系统，主要承载了哪些业务？

王海亚：从用户视角来看，交易承担了购物车、下单、订单管理这些功能；从功能视角来看，交易系统包括做业务规则和服务整合的交易平台以及支撑交易功能的底层服务，服务包括商品、优惠、库存、订单、物流等，业务规则主要是确定在不同的购买场景如何使用底层服务的不同功能，比如什么样的购买要走付款减库存，什么样的购买要走预售模式的价格体系和支付体系，什么样的购买要限制使用积分等等。

InfoQ：请您简单介绍下交易系统的架构演进过程。

王海亚：个人把交易系统的演化分成三个阶段，第一个阶段是单应用阶段，那个阶段淘宝的业务比较简单，业务量也比较少，交易系统相对还是很简单的，当时的交易系统只有一个应用，囊括了商品、优惠、物流、订单等功能，一个小的开发团队就能够完成交易系统的开发维护工作。

随着承载的业务越来越多，交易系统变得越来越复杂，需要投入越来越多的开发来维护这个系统。人员规模上去之后，内部沟通的成本、代码管理的越来越高，大家都针对一个代码库做更新，很难保证非常好的功能模块设计，经过持续的代码累积，满足业务需求并行开发越来越难。后来提出了服务化改造，从原来的单一应用中逐步做功能剥离，拆分出多个系统，每个子系统负责不同的功能，由不同的团队维护。交易系统的架构进入第二个阶段，交易系统采用分布式架构，存在多个后台服务系统，多个前端应用。商

品、优惠、库存等功能逐步沉淀成纯粹的后台服务系统，原来的交易系统作为前端应用存在。由于组织结构的原因，当时是存在几个交易前端应用，比如说当时的商城也就是后来的天猫，无线团队等等，因为前端内容的差异，都有自己独立的前端应用。

服务化改造之后，从某种程度上缓解了第一代架构的缺陷，但是新的架构在落地过程中还是存在一些问题，比较突出的有两个问题，一是由于功能是从原有单核应用中剥离出来时，缺乏一个明确的功能边界定义，对业务规则和功能的识别没有统一的标准，导致业务规则和功能存在紧耦合，同一个业务的业务规则一部分在前端应用中，一部分在后端服务系统中，业务规则变化又比较快，任何一次规则变化都要涉及到多个团队做配合实施，沟通协调的成本比较大；另外一个问题是由于多个前端应用，前端应用做的比较重，业务逻辑都要在多个前端应用中重复实现，除了开发的人力浪费之外，也引发了很多由于落地节奏不一致导致的业务问题，即使耗费大量精力做沟通协调也无法完全避免此类问题。

在新的系统架构中，对业务规则和功能做了明确的识别，通过集中式的业务规则控制以及功能定制化机制，实现业务规则和功能的完全解耦，多数情况下业务规则变更不会引发后台服务系统的变更，后台服务系统的功能增加也能做到对前端应用的透明。另外在前端应用中引入新的编程框架，通过框架做纯粹的服务整合就能组织出要给用户披露的信息。前端做轻之后，针对不同终端提供统一的业务服务，或者针对不同终端根据单页需要披露的内容来做不同服务的整合，成本都降低很多。

InfoQ：没有改造之前的交易系统，大概遇到了哪些问题？

王海亚：问题主要可以分成三类，第一类问题是业务快速落地很难，由于交易的业务规则变更很快，通过casebycase代码修改，系统中功能与功能之间的耦合越来越多，系统维护成本越来越高。另外业务规则散布在多个前端应用和后端服务系统中，经常由于各个系统的规则不一致或者上线节奏不一致导致业务故障。第二个问题是系统接入的成本很高，由于业务的需要，会引入越来越多新的服务系统提供对应的功能，这些服务都是在一个前端系统中做服务的整合，一个点的不稳定或者性能瓶颈，都会影响整个系统的稳定及用户体验，想通过小成本的尝试来做业务创新的验证成本非常高。第三类问题是前端应用比较重，业务逻辑主要存在于前端应用中，前端应用的页面逻辑和后台逻辑职责边界又不是特别清晰，导致前端应用的开发成本非常大，另外又存在多个前端应用，业务逻辑的一致性很难保证。

InfoQ：新的交易系统是如何解决老系统的问题的？

王海亚：总结来说，可以从7个点来讲。

第一，通过服务治理，把业务规则从原有功能中剥离，明确各个服务所提供功能的完备性及独立性，从系统边界上确保功能之间无耦合。

第二，通过集中式的业务规则管控，保证交易全链路的业务规则统一及业务规则灵活可配置。

第三，通过标准化的交易框架及组件化设计，保证服务在交易平台快速接入。

第四，通过异步并行及容错，提升系统响应速度，减少单点服务故障导致的整体不稳定。

第五，通过开关及预案机制，保证代码的兼容性发布，业务降级容错。

第六，通过流量管控，防止雪崩，保证在正常情况或者某个服务集群能力波动时，整个交易系统不被压垮。

第七，通过在前端应用中做前后端解耦，通过约定的数据模型，前端开发只负责页面展示和交互，实现前端、后端的并行开发，也使得不同终端的自适应可以以较低成本实现。

InfoQ：新的交易系统是如何应对交易洪峰的？

王海亚：由于交易的调用链路比较长，从Web服务器到执行链路中的各个服务系统，包括缓存、DB等存储系统，请求都是以队列的方式被处理。各个系统如果没有处理好容量控制，就会导致请求的堆积，对外呈现的就是响应时长增加。

在处理链路比较长的情况下，这种异常会被放大，链路后面的一个点发生堆积，前面的各个点也会逐步出现堆积。页面应用又是一个用户强交互类型的应用，如果用户发现慢就会刷屏，而后面的系统无法感知接下来要处理的请求是否是有效的请求，这种情况下可能会导致整个网站的瘫痪。我们目前是在前后端应用中都引入流量管控机制，主动拒绝超出自己处理能力之外的请求，保证每个应用都不会被压垮，有多少能力就能把多少能力提供出来。

另外在前端应用处理流控时，主动引导用户到低成本/静态化的页面上，增强用户体验，也能提高网站其它部分的曝光率。整个流量管控体系，除了应用框架的能力之外，还有一些配套的机制，比如自动化压测系统、容量计算公式、自动化扩容/下线等，整个一套体系，确保我们轻松应对交易洪峰。

查看原文：[王海亚：淘宝交易系统演进之路](#)

张奇：天猫推荐算法团队的那些事儿

作者 杨赛 发布于 2014年4月1日

天猫技术部算法组是一个相对比较新的团队，刚刚成立一年，目前有10个算法工程师和5个开发工程师。这个团队所负责的内容是天猫上的数十个推荐产品，这些推荐产品帮助消费者找到他们喜欢的东西，将用户跟商品匹配的路径缩短。当然对天猫平台来说，推荐算法的价值在于提高转化率。从去年的双十一开始，天猫技术部推荐算法组第一次将推荐产品引入到了双十一大促活动当中。

2014年，阿里巴巴集团举办了[阿里巴巴大数据竞赛](#)，大赛的规则、题目、比赛数据、评价标准与评审，都是由算法组负责。最近，InfoQ中文站采访了这个团队的负责人张奇（得福），了解天猫推荐算法组的团队情况与工作内容。

嘉宾简介：张奇，花名得福，2010年毕业于中国科学技术大学计算机系，信息检索方向博士。2010年7月加入阿里云计算，搜索广告组，从事搜索广告算法研究，参与Yahoo中国搜索中搜索广告的排序算法设计，负责了国内最大规模之一的，每天近40亿网页浏览记录的挖掘、用户行为分析和User Profile 建模。2012年3月加入天猫产品技术部，推荐算法组，负责天猫推荐算法的改进和数十个推荐业务的优化，包括PC推荐业务、无线推荐业务，建立起一套基于机器学习的推荐算法流程。

InfoQ：简单介绍一下天猫技术部推荐算法组的职责和团队分工？

得福：推荐算法和搜索、广告系统类似，很多模块之间都有上下游关系，如分析用户偏好兴趣，采集用户点击、购买行为，计算用户现在想买什么，比如想买连衣裙还是t-shirt，连衣裙是日韩风格还是欧美风格。还有的组做排序，比如有一堆的商品，连衣裙，天猫有几十万款连衣裙，给用户看什么？这就需要先做品牌、价位的筛选，再有个性化排序，给每个商品计算分数，通过预测点击率、购买率，来给候选商品做重排序，这样选出10个、20个商品给用户看。另外还有一些固定的物理属性，如用户的性别、收

入水平等。每个模块都有上下游关系，不同的人做不同的系统。每个推荐流程都有用户行为分析、商品检索、个性化排序、返回前台的过程。

产品上来说有不同分工，我们这边也是PM制度：比如一共有40~50个推荐产品，如果有新的产品要开发，就需要PM来协调上下游的模块的同事一起去完成。PM跟运营或PD沟通需求是什么，需要我们怎样支持，需求怎样拆解成现有的推荐算法能够支持的模块，然后安排各个模块的同学去开发，还要监控上线之后产品的效果，效果好或不好，有数据报表，不好有改进意见。每个产品都要有同学去负责。

整个推荐分商品推荐、品牌推荐、活动这三种，如果一个同事对某种类型比较熟悉了，跟对应的业务方也非常熟悉了，可能他就专门把这个类型的产品负责好。商品推荐比较广泛，推荐比较多，就要分给不同的同学。总体来说，这一块在系统上是横向的划分，业务上是纵向的划分。

InfoQ：你以前做的是阿里云搜索。搜索和推荐产品上从算法和产品的角度来看，差别大么？

得福：算法和产品都不太一样。算法方面，搜索是处理带有明确意图的query，很多工作都基于对query的理解上面，包括query怎么分词，如何抽取有用的信息，包括品牌、商品型号、类目等，就是通过query的关键词抽取来反映用户意图，然后做查询、排序。推荐的话，用户不会输入query，他们用自己的行为表达自己的兴趣。对我们来说这更困难，你要从非常多的行为里把用户的兴趣抽象出来，我们需要做很多用户意图理解的模块。这是两者在算法上最大的区别。

产品方面，搜索是一个通用系统，固定的产品形式就是有一个框。推荐的产品可以有很多变种，不同行业也不一样，比如服装可能是基于风格、搭配来推荐，书籍、音乐则是另外一种推荐方式，比如书籍要考虑话题、类别、知识水平，所以比搜索来说，产品形式会更加复杂。

InfoQ：验证新算法会有AB测试吧？怎么做的？

得福：我们的测试有两种，一种离线测试（线下测试），一种线上测试。一开始我们都做线上AB test，你拿5%或10%的流量供测试，95%或90%的流量作为基准，然后你可以做参数变动或算法调整。这是以前。后来我们考虑到，因为同时会有很多算法想要做测试，一方面流量有限，而且对天猫来说流量是非常值钱的，每个用户可能只有20分钟或

者30分钟，所以一个算法如果未经验证就到线上做AB，其实是很浪费流量的，对用户的体验也不好，5%或10%的用户可能会使用一个非常差的版本。

所以渐渐就开发了一套离线评测方法，不需要线上流量就能测试：它用以前的日志去模拟现在的行为，相当于把日志作为模型的输入，看不同版本的算法在日志上跑的结果怎么样，来判断这个东西放到线上后大概的效果。这种系统不会100%准确，离线结果A比B好10%并不表示在线上也是10%，但一般如果离线显示A比B好，那到线上A一般也会比B好，好多少不一定。我们用离线方法就可以淘汰很多不太靠谱的算法——A如果比B明显要差，我们就可以去掉。只有A比B好，我们才拿到线上测，保证流量能够被充分利用。另一个好处是，离线测试十分快速，因为处理日志跑起来是很快的，可能十几分钟就跑完了很大的用户量，而线上一方面流量少——不可能切换太多过来，同时你需要看很多天，效果才能稳定下来，可能需要一到两周才能看出效果。

所以这两个方法现在我们同时在用，互相补充。

InfoQ：天猫算法跟产品相关性比较大，会有很多跟产品的沟通。而你们也在产品开发更下一个层面。跟PM、PD、运营的沟通模式是怎样的？

得福：PD是我们的需求方。PD冲在最前面，跟业务靠的比较近，甚至会跟业务绑在一起。PD去了解业务的需求，然后提前跟我们交流需求是否合理，如果合理且我们能接受开发量，就回去写PRD、MRD这种文档。这可能会涉及很多个开发团队，就要把开发团队的工作量、时间都安排好，最终确定一个上线时间。

一般PD会先写MRD，证明该产品需求是合理的。再写PRD，对产品进行详细的功能描述，分哪些模块，谁来开发，都要涉及。最后会有PRD评测，所有相关方都会到，如果有意见会让PD去改。如果大家觉得OK，PD需要指定PM和最终上线时间，如每个团队在哪些时间点要提供哪些接口出来，测试时间等等。

InfoQ：你们在去年完成了多少项目？

得福：我们不是按照项目的个数来算工作量。我们总体的评价是，通过推荐帮天猫带来多少成交，这是我们最重要的指标。这些指标的实现过程有很多小项目累加，包括新项目的开发和老项目的优化，目的都是提高转化率，不会拆的那么细。

其他具体的方向也有一些，比如无线的产品创新做的如何，或者大赛的支持等等。

InfoQ：你在去年最值得分享的成就是什么？

得福：双十一。去年是我们第一次在双十一尝试这种千人千面的推荐方式，而不是纯卖场的方式，我们是第一次做这种尝试，也突破了很多阻力。当时开发非常辛苦，也不是很高科技，很多跟运营一起的工作，数据都是excel来传递，自动化系统都没有准备好，因为系统没有打通，所以很多人肉的工作，非常痛苦。那段时间加班很多，容易出错，很多时间用来做协调的工作，不过最后效果还是很好的，转化率比原来那种卖场方式能提高10%~20%左右，给业务带来比较多的价值。

InfoQ：转化率是对整个团队的考评。对于每个人的考评，你们是怎么做的？

得福：我们对工程师个人的考评主要分三个部分：1、业务推动，比如承接了哪些新产品开发，在过程中起了什么作用，是PM、模块开发还是数据报表开发等。如果作为PM，对推动业务做了什么，比如老业务上去可能效果并不好，你是否通过一些主动的行为——如改变产品形式——来把它的效果提升。这是很大一块。2、技术工作，如你在这段时间有没有算法方面的创新？数据结果有没有做产品化、工具化，把之前做的工作总结、凝练、抽象出来，让别的同学也可以去用？3、团队贡献，如你是否做了一些对其他同学支持、分享的工作。这个跟前面两个也有关系，相当于是从前两个部分中抽取出跟团队支持相关的内容来进行评估。

InfoQ：任务的分配是指派的，还是大家可以自己选择感兴趣的方向？

得福：一开始肯定是指派的任务。到你熟悉整个业务了，或者对某个方向掌握非常好后，就会放手让这个同学去做某一块的业务，不需要team leader去管。一般可能需要半年时间。

InfoQ：你们的团队招聘主要是什么渠道？

得福：我们的社招不多，应届生和转岗比较多。对应届生，如果是研究生，我们要求他要么是工程能力很好，项目的深度、广度掌握的比较好，跟我们的方向比较一致，要么是研究做的比较好，论文质量高。社招的话，因为他有经验了，我们主要要求他的经验跟我们的方向比较一致，同时他在之前公司做的结果比较好，做成功过一些项目，对项目细节和全局都能掌握的比较好，算法能力也比较好。我们对社招的要求比较高，名额也紧张，通过率比较低。

InfoQ：对这次的天猫天池算法大赛有什么期待？

得福：其实我们以前没想过这个比赛会做的这么大，原来我们只是觉得好玩就行了，在小圈子里面有同学有兴趣的同学来一起玩，加强算法交流，也没有想限制在学生的范围，而是希望其他公司的同学也可以来一起玩，通过这个比赛加强交流，也认识认识其他公司的团队。

后来这个比赛做大了，做到集团层面了，整个集团层面可能希望有一些校招的效果，对我们来说也是更好的事情。如果有很好的学生愿意过来，我们当然非常高兴。不过我们更希望大家更纯粹的在这个平台上进行比赛。

查看原文：[天猫推荐算法团队的那些事儿](#)

长仁：核心系统研发部的发展历程

作者 杨赛 发布于 2013年8月15日

在2013年7月的阿里技术嘉年华上，阿里巴巴核心系统研发部的王琤（长仁）分享了他们团队在计算密集型应用优化上的一些经验（[观看视频](#), [PPT下载](#)）。会后，InfoQ中文站跟长仁进行了一次交流，了解他们在需求处理、团队组成与分工、与其他部门的合作方式等方面的一些情况。

嘉宾简介：王琤/长仁（微博：[@王王争](#)），阿里巴巴核心系统部专用计算组负责人。在阿里巴巴致力于专用计算的推广及实践，以追求最佳性能及性耗比为目的为特定应用针对特定硬件架构进行算法及实现优化。在图像搜索、机器学习等领域，针对特定CPU微架构，GPGPU等不同体系结构进行关键算法及实现进行优化并获得了指数级的性能提升。同时负责淘宝JVM的开发工作，基于OpenJDK VM为淘宝定制、优化更加贴近应用需求的专用JVM。

InfoQ：长仁你好，感谢接受我们的采访！您刚才的分享涉及到很多底层的优化，那现在，优化团队是怎么去和业务部门交互的？

长仁：现在流程不太一样。早期的时候，是我们去到处找需求。自己去找是很难的。

后来有一个突破，是因为我们核心系统有一个图像团队，他们意识到出了性能问题，主动要求我们来帮忙做一做优化。那么我们一做，就有了十倍的性能提升。

现在，需求方面有很大改观。一方面是由于我们有很多优化成果的积累，有了一定的口碑。另一方面，我们接触到越来越多计算密集型的应用，我们的工作就有了需求。由于我们有各种各样的成功案例，开始有越来越多的、阿里的其他的团队主动来找我们。

现在的流程是，兄弟团队找我们，和我们讲，应用是什么样子，现在有什么样的性能问题，然后我们就会跟着应用团队一起做profiling，帮他们进行优化，上线测试。

未来的流程会不一样：我们现在正在做一个系统，System-wide profiling，也就是一个系统层面的直接，持续profiling，我们会把这些agent点布到每一台机器上。Google有一篇[有关google wide profiling的论文](#)。我们的方案也类似：自动感知哪些程序跑的慢，哪里跑得慢。我们现在正在做这个，已经有原型出来了，马上就要布到我们的MPI集群上。部署完了以后，将来就是谁也不用找谁了，系统一看就知道，这个程序这跑的比较慢，我们来去帮你优化。

InfoQ：这个系统方面的profiling是不是有点像应用层面的监控？

长仁：不是应用层面，不用应用改任何代码。我们在底层自动进行profiling，应用跑在上面，性能数据我们就能大体知道了，比如说每一个时钟周期内平均执行多少条指令，跟理想值比跑得是不是慢了，程序里头具体哪个函数占的时间最长，我们都能知道。所以未来的流程就是，我们主动通过系统一看，哪里有问题，我们来帮你修哪里。

InfoQ：你现在在阿里这边做性能优化，跟你以前在甲骨文做的事情有什么不一样？

长仁：甲骨文完全不一样。我在甲骨文是在Application Product Group，应用产品的部门。我们在做的是一个自己的产品，叫SDE，Solution Development Environment，它是一个Solution的开发平台。这个平台跟企业级应用靠的更近，跟互联网应用没关系，跟密集计算更没关系，所以跟现在完全不一样，但是有很多是相通的。相通的就是，你了解的越底层、越深，对于开阔你的思路，解决问题的方法是更有好处的。

InfoQ：你们团队的组成是怎样的？

长仁：我们团队很有特点：绝大部分都是应届生。真正有工作经验加入到团队的，目前为止，算上我，只有两个人，剩下的六个人全是应届生培养出来的。为什么？因为对于我们来说，社招生跟应届生区别不大。你想，我们优化JVM，全国有多少人做JVM的？第一，很难找；第二，应届生培养也是培养，社招来了没准也需要培养，而我们的应届生同学都是非常优秀的，经过千挑万选来的，他们的学习能力很强，经过几年的培养，有的已经成为国内的首屈一指的。举个例子，你们都知道莫枢（注：莫枢在QCon上做

过JVM定制改进的分享），JVM在国内是顶尖的，那是我们团队的；还有洪熙，现在在GC这方面可能国内没有人比他更熟悉。

InfoQ：团队的分工大概是什么样的状态？

长仁：我们组就两大部分，专用计算跟JVM。JVM部分四位同学，剩下的都做负责计算方面的优化工作。分工还是很明确的，因为这两个领域差别比较大。我们是一个小团队，人不多，关键是干练。

InfoQ：所以，当一个需求过来的时候，你们跟应用开发团队是如何配合的？

长仁：流程刚才介绍过。客户来了，可能是各种各样的应用，如果是Java Application，就是JVM组去处理；如果是计算密集型的应用需要优化，就由计算组去支持。

InfoQ：万一各种各样的应用都来找你们，你们不是得搞懂所有的应用？

长仁：是的，需要有很强的学习能力，同时，如果我们的客户多到那个程度就太好了。

InfoQ：你们团队的工作效果是如何评估的？性能提升了多少倍这样？

长仁：这个问题特别好。为什么？一般的人会觉得，你们的评估标准是不是提升了十倍，几十倍这些，但恰恰不是这样。对于我来说，我们提升几倍十几倍几十倍都是家常便饭，但是这些从来都不是我们的KPI。

真正的成绩是什么？成绩就是用户真正的用上，在线上使上，产生了好的结果。为什么要这样？因为在互联网企业，做一个工作必须要有结果；而且互联网企业节奏是非常快的，没有时间让你做纯研究性的东西，你做的东西必须有人用上，带来好的效果，这是最重要的。

所以我们的要求是很严格的：第一，技术有难度；第二，还必须要有好的结果。其实回顾一下我们的时间都用在哪儿，对于我来说就是，1/3的时间做优化，1/3时间推广上线，还有一个1/3是培养新人。对于我们组的员工来说就是基本上一半一半，一半的时间在写代码做调优，一半的时间在找应用来推广，或者出去做广告。所以，很多外面的同学会觉得，你们做得东西真好，有那么好的效果；但是你真正来了以后就知道，优化本身不是最难的东西，最难的是让用户真正的使上，产生效果。有时候由于各种原因，你做的东西可能最后用不上。

InfoQ：万一有的同学，他就是不擅长推广，但是他特别擅长做这个东西，这样也不行么？

长仁：有这样的同学。但我们标准就是这样，看结果。学会推广是必要的技能，必须要做，所以我们的同学其实确实挺苦。有的同学性能虽然提升很多，因为推不上去，没有用上，没有结果，最终年底Review就会很低。

InfoQ：底层这块是相对枯燥的领域，你们每年要如何吸引新鲜血液的加入？

长仁：这个领域确实是比较小众的，每年的应届生里边就几个对口专业的有一些人才，这里面还不是所有的人都想来。

这个领域不像其他领域，比如出了一种新语言，或者Web编程一种新的设计模式，或者新的架构，大家都在聊。这些东西确实太底层，真正对它有兴趣的人很少，有机会来做这个的人也很少，两者叠加确实人很难找。

所以我们现在就是应届生培养，来了以后先“洗脑”，告诉他这个东西很重要——越没有先前观念的，他越能接受；另外一部分人，确实奇葩中的奇葩，天生就爱这个，那就最好。我们组三年，现在算上我才有八个人，每年转换的应届生就一到两个人。

吸引点有几个，阿里巴巴公司品牌是一个，同时，我们做的工作对于对这个领域有兴趣的同学会很有吸引力，因为在企业里专门从事这个领域的机会的确不多。

InfoQ：您自己是属于对这方面有特别爱的？

长仁：我目前看搞这块，就三类同学：第一，自发的有兴趣，就是喜欢。这个就属于天生爱，你让我改行，我也不适应。

第二类同学是被逼的。这类同学，实验室导师也好，或者工作领导，就是让他干这个，为了生计他也就干了。他可能不喜欢，但是他干了以后也有成长。

第三类是被启发的。他可能以前没意识到，然后当自己技术发展到一个瓶颈的时候，比如说一个程序员发展到架构师了，以后不知道哪些方面是发展的方向，比较迷茫，技术上也得不到突破。很多人会遇到这样一个技术的瓶颈期。然后他可能偶然的，比如偶然接触到一些硬件的知识，软件究竟在硬件上是怎么跑的，我们怎么能提升那么多倍性

能，可能他就觉得这像打开了一扇大门，进入了一个新的领域，他觉得很好奇，开始在这个领域里探索摸索，这是被启发的。

像今天我的分享，我相信里面很多人可能听不懂，但是会给他一个启发，会让他觉得其实这个领域应该关注关注。

查看原文：[从找需求到自动感知：阿里巴巴核心系统研发部的发展历程](#)

ArchSummit

全球架构师峰会2014

2014.12.19-20 北京国际会议中心



互联网金融
研发体系构建
云计算解决方案专场
电商，不是搭个平台就能赢

- 转型中的SNS
- 智能硬件，更懂你
- 移动互联网，随时随地
- 云计算与大数据，从技术选型说起



3

阿里云相关技术

王坚博士：云计算就是服务

作者 霍泰稳 发布于 2012年11月15日

又是一个双十一，很疯狂的季节，列出几个数据大体就可以了解何谓“疯狂”：去年双十一，淘宝交易额是33.6亿，已经让人咂舌，在同样的时间段里，今年双十一期间淘宝的交易额已经是191亿，出单量也达上亿次。在国外的一些技术交流网站上，很多技术人员也在好奇地问，这么短的时间，这么大的交易量，这家中国的公司是如何做到的？我们国家的12306火车票售票网站，每天的出票量不过数百万，还经常瘫痪，可想而知支撑双十一背后的技术储备需要多深厚。再想想阿里巴巴从几年前就开始的去IBM小型机，去Oracle数据库，去EMC高端存储设备等，并从2010年就开始的双十一购物狂欢活动，每一个改变都让业界小有震动。而这一系列技术动作都和自2008年加入阿里巴巴，并任首席架构师的王坚博士有着密切的关系。

相对于这些隐形的、可能并不为太多人所熟知的故事，也许技术圈内大家更关注的还是阿里云所做的事情，而这也正是带给王坚博士非议最多的地方。按照王坚自己的说法，阿里云已经不知道死过多少次，一代又一代人坚持不下来而倒下，一代又一代新生力量站起而坚持，这最终也慢慢让人看到了萌芽的诞生。在今年双十一的上亿次交易中，就有超过20%的运算是进行在阿里云平台上的。除此之外，数据量更大的雅虎邮箱数据，也将在年底前全部完成从专有存储设备到阿里云平台上的迁移。

和大多数热爱技术的人一样，王坚博士也有着自己的技术情节，愿意为开发人员的事情而奔走，这也是为什么他坚持从去年开始，每年举办“阿里云开发者大会和大赛”的主要原因之一。在阿里云开发者大会期间，和王坚博士的这次深度交流中，我们谈到阿里巴巴对技术人员外出分享的看法，对当前技术趋势的理解，对阿里云飞天平台的架构解释，以及飞天开放平台的起因和背景等。他也提到通过阿里云开发者大会，让他度过了平台和应用的纠结阶段，虽然应用为王，但是作为平台提供商，自己要不要去做应用是一个非常困难的选择，这也是一个鸡生蛋和蛋生鸡的问题。因为平台做得不好，大家就

不会去用，而没人用，平台也就一定做不好。更多的开发者在飞天平台上做应用，可以帮助阿里云更专注于平台的建设，根据用户的反馈不断地去优化，而不是将精力放在和开发者相竞争的应用开发上。这样抉择的结果，也让阿里云逐渐走上了良性的发展道路。

InfoQ：阿里巴巴的技术比较开放，敢于让技术人员出去分享，作为阿里巴巴集团的CTO，您如何看待这样的事情？

王坚博士：首先阿里巴巴是一家服务公司，不是软件公司。虽然不同的部门对技术分享这个事情理解还不太一样，开放程度也不同，但是整体上，对技术分享我们是非常支持的。一方面，阿里巴巴的业务大多数是平台性质，拿出去讲也没有什么关系，即使现在我们将飞天的源代码给你，你也不一定能用，在这方面，我们还是很有信心的；另外，这和企业的传统文化也有很大的关系，阿里巴巴是一家好打抱不平的公司，愿意帮助弱小者成长，可以说我们是有分享的基因在里面的。

InfoQ：在09年的时候我就看过您关于大数据分析的一些分享，当时很多人还不是很理解，三年过后，这已经是任何一个大公司发展过程中绕不开的话题，您的这些前瞻性的观点从何而来？

王坚博士：我认为我还是蛮有运气的。2006年我在微软工作的时候，就成立了一个研究数据智能的小组，因为这个原因，当时看了很多和数据相关的东西，这些资料也帮助我去思考。我记得那时候微软的Office产品第一次有了收集用户反馈意见的功能，就是弹出一个框，询问你是否参与微软的用户体验改进计划。这也让我第一次认识到互联网对数据的影响，也明白了云计算的基本道理：云计算就是为数据处理而生的，而存放数据（Hosting）只是其中的一小部分。看一家公司的变化，只需要看Hosting在其业务中扮演的角色就知道了，我认为50%是一个临界点，而阿里巴巴目前已经远超过这个数值。

你可以理解，现在阿里云是一个数据处理公司，不是以计算为中心的，而是以数据为中心的，阿里云也有志成为一家真正的云计算公司。开始的时候很多同事不是很理解数据和处理之间的关系，经常被我搞疯掉，辛辛苦苦做的一个系统，可能很快就被否决掉，后来到了盘古这一层才算通了。所以说，云计算的真正意义还是在数据业务，数据沉淀在哪里就应该在哪里处理，而不是将数据和处理这两件事情分开。

InfoQ：飞天项目的愿景也非常大，这个云计算的平台上包括云OS，包括内核，也包括存储服务。那么这个平台除了为阿里巴巴内部使用外，对外还提供什么服务？

王坚博士：飞天的架构还是非常大的，这在当时我们开始做飞天的时候，也引起了非常大的争议。很多人说我们做的东西太多了，太大了，不符合现实。这让我当时也非常纠结，但云计算就是一个综合服务，你让客户在这家公司买Hosting，在那家公司买CDN，然后再找一家公司做数据处理，他们肯定搞不定。中国网络电视台/央视网CNTV就是一个很好的例子，就是大多数情况下，你不能只提供一项服务，而是要帮助客户做完整的生意。其实不论是基础设施，还是平台，还是存储，都是天然整合在一起的，云计算的鼻祖亚马逊现在也是沿着这条路走的。

另外，阿里云飞天项目还有一点是需要说明的，就是不论是对外，还是对像天猫、支付宝这样的内部公司，阿里云所提供的基础服务都是一样的，我们并没有为天猫、支付宝单独定制什么功能，从版本升级，到系统兼容性，对内、对外都是一样的。在这次双十一购物狂欢节中，将有超过20%的订单是在我们的弹性计算上进行的，这可是每天1000万的PV量。换句话说，我们能够支撑双十一这样的业务，那么其他的互联网服务我们也是可以做的。

InfoQ：目前有没有一些已经接到飞天平台上的案例，他们是如何使用的，或者说飞天是如何帮助他们的？

王坚博士：虽然我们的平台可以支持Hosting、CDN和数据处理，但是坦白来说，目前我们的客户主要用的还是Hosting的服务，稍大一点的客户会用到负载均衡，但像InfoQ这样的传媒行业，可能会用到CDN等，能使用到综合性服务的客户并不是很多，但一定是一个趋势。这儿需要解释一下的是，在阿里云的产品里，并没有CDN这一项产品，但是我们是支持这个服务的。

另外，现在我们考虑的是如何让大数据也变成一个服务，这一点也是我们的客户，很多传统的客户教会我们的。基本上来看，Hosting是一头，数据处理是另外一头，现在Hosting占得比例比较大，但总有一天会是大数据。比如我们和CNTV的合作，从ODS（开放架构化数据服务）、RDS（关系型数据库服务）到CDN等能用的功能他们全用上了。

还有一个例子，我们有一个客户，做中间件的，开始的时候只是想把中间件从专有的存储设备迁移到我们的云计算平台上，最后发现不仅中间件，其实其工作平台也是可以用在云上的，员工甚至在家里就可以办公了，从而引起了其现有组织的变革，给公司的发展带来的新的机遇。这并不是说我们想做一个什么生态链，而是像福特生产线一样，很

多结果都是自然而然到来的。就像电气化一样，电的出现改变了社会，但电网公司顺带也挣了点钱。阿里云也是一样，我们在改变整个生态，顺带也可能挣了点钱。

InfoQ：我看了飞天开放平台的一些介绍，也听一些朋友在谈，都认为这是阿里云的一个大动作，请帮助介绍一下飞天开放平台的起因、背景，以及能够解决的问题。

王坚博士：如果说阿里云的开放平台有什么不一样之处，那么这个开放平台的特点就是只关注云计算。整体来说，云计算是分两种的，一种寄生于业务，另外一种相对独立，和业务关系不大。阿里云属于后一种，不寄生于任何业务，也就是任何业务都可以在阿里云上进行。我们希望能够让小公司来做大公司的事情，比如Teamcola（一家提供工作日志存储的公司），让大公司也可以做小公司的事情，提高效率，比如中软国际等。

对于飞天开放平台，现在只是开始，希望能将能力开放给更多的开发者和公司。其实，在互联网时代，IT运维是如此地重要，像Teamcola这样的小公司，你很难想象如果不将数据放在云平台上，他们怎么成长？要知道现在单单是招聘运维人员就是很困难的，有时候业务的发展就是受自然法则的约束。另外就是很多公司，老的应用太多了，凭自己的技术能力，很难完成迁移，我们计划扶持一些公司来做类似的事情，帮助这些公司完成转变。

再者，飞天开放平台很坚持的一点就是，我们不自己做应用，而是鼓励其他人和公司来做，比如网盘，这样的应用对于一家云计算公司或者开放平台来说是很重要的，因为可以很容易地做出来并形成规模，但阿里云不做。我想这也是飞天开放平台和其他很多云平台不一样的地方吧。

InfoQ：对阿里云下一步有什么动作，您心中的规划或者期望是什么？对云计算在国内的发展，您认为下一阶段会有什么新的变化？

王坚博士：中国小而美的公司还是很多的，可能比美国还要多，我想如果将来有一半运行在阿里云上，就差不多了。目前阿里云的成本还是比较高，还不是很符合国内用户的使用习惯，但将来一定是可以降下去的。到那个时候，有更多的小公司起来，自然而然，创业的氛围、互联网的氛围也会越来越好。

从技术上来说，阿里云的系统已经做得很靠前了。我没有做过精确的统计，但是阿里云应该是世界上第二家有能力做多Master的公司，这个技术还是很有难度的。所以，近期不会对系统做大的调整，希望能够就系统的易用性、稳定性做些优化。

懂更深的角度来看，要做一个组件或者功能比较容易，难的是将整个系统立起来。这些都是技术问题，不是商业问题。比如让云计算的成本下降，很多人可能认为这是一个商业问题，其实不是，这是要从技术上解决的，如何让系统更省电，更节能等。

云计算在中国一定会起来的，以我们的经验来看，这是一件很难的事情，单是“提供云计算服务”这句话就有很多种定义，而且很多能够提供云计算能力的公司在技术上还有许多的缺陷，需要继续积累。但前景看好。

编辑后记：行文至此，猛然想到在谈论视频业务在国内的发展时，优酷创始人兼CEO古永锵说“没有一亿美元，就不要玩视频”。也许这句话套用在云计算平台领域也适用，虽然有些小公司可以提供某一方面的云计算能力，但是要为企业提供全方位的服务并不是那么容易，也许最终只有像阿里巴巴、腾讯、百度、盛大这样的公司才能做到。但对于中国互联网的发展，我们还是乐于看到更多关注云计算的大小公司能够冒出来，提供各种各样的服务，小公司可以专其所长，为客户提供刀片式的核心服务，而大公司可以利用大的优势，为客户提供全套解决方案。正如王坚博士在访谈中所提到的，更多小而美的公司起来的，那么我们的创业氛围、互联网氛围才会越来越好。

查看原文：[云计算就是服务：阿里云总裁王坚博士访谈录](#)

徐常亮谈阿里云ODPS的愿景、

技术实现与难点

作者 杨赛 发布于 2014年4月7日

2014年1月，阿里云将其[ODPS](#)服务开放公测。2014年4月，[阿里巴巴大数据竞赛](#)的所有参赛者将在ODPS平台上进行算法的调试、测试；同月，ODPS也将开放更高级的功能进入公测。

InfoQ中文站近日跟ODPS平台的技术负责人徐常亮进行了采访，交流了有关ODPS的愿景、技术实现、实现难点等话题。

InfoQ：先介绍一下ODPS现在的情况吧。这个产品能做什么？

徐常亮：ODPS是2011年正式有的名称，全称叫做Open Data Processing Service，简单来说就是数据处理的服务。它的定位是在飞天之上，提供数据仓库、数据挖掘和其他数据应用等功能。

2011年的时候我们尝试对外提供ODPS，当时有一些小试点，但是后来发现各方面条件没有完全成熟，不管是外部对云的了解还是内部对ODPS未来的预期都不是很清晰，所以一直到2012、2013年，它发展的节奏都比较慢。去年大概6、7月的时候有一些变化，因为飞天到了5K的里程碑，在技术能力方面的顾虑已经小了很多。因为飞天是分布式操作系统，它提供最基本的存储、CPU调度能力、内存使用、网络等功能，是最基本的资源包装整合，相当于是一台计算机，而我们是在它上面开发的应用，相当于是一个分布的数据仓库，让用户可以在上面做基本的ETL处理、SQL查询、数据导入导出等，还有一些MATLAB、统计软件的功能。

除了这些基本功能之外，我们还提供了一整套数据挖掘算法Xlib，让用户可以建模、做高级的数据分析。另外，我们还可能提供一些编程框架，让用户自己可以编写程序进行数据处理，比如单机上有Python、Java，我们就提供MapReduce编程框架、或者专门为

了解解决迭代问题的Graph编程框架（也叫做BSP，跟Google Pregel模型很类似）。我们会逐渐加入各种内容，凡是涉及数据处理的工具和编程框架我们都会想办法加进去，让开发者和用户可以对数据进行各方面的操作。

总而言之，ODPS就是基于飞天分布式系统提供的一套关于数据处理各方面工具和框架的服务。

InfoQ：对应AWS的话，相当于是[RedShift](#)和[EMR](#)吧？

徐常亮：可以这么去对应。纯粹从功能来讲，我们会提供类似EMR和RedShift的功能。但我们不仅于此，我们还有建模的库、机器学习的库，从编程框架的丰富性上面也不仅仅是MapReduce，还有迭代框架。暂时来看我们做的可能更多，当然AWS也在逐步提供更多的功能。

另外有一个很大的不同是，ODPS是作为一个有机的整体来提供这些服务的，不同的功能是服务的不同层面，而不是单卖的功能。比如在我同一个体系里面，我数据仓库类型的一个SQL处理好了，我紧接着一个MapReduce的作业就可以很好地关联起来，他们的物理存储数据，以及描述这些数据的元数据，都是在同一个体系里面。不像RedShift和EMR，它们是在一边处理完了之后，要把数据导出到另一套系统里面去处理，它们的元数据描述不是互相共享的，要有一个第三方来做对应，比如RedShift表结构是怎样的，EMR的结构要怎样相应的去设计。ODPS希望让对象都在一起，让要处理的对象和元数据都在一个ODPS体系里。在此之上，你要做授权也好，管理维护也好，都是同一个界面，对用户而言就是在一套系统里面做不同的处理，用户觉得我就是在一台机器里面，只不过在不同的文件夹。AWS的话，用户会感知这是两台计算机。

另一个区别是，ODPS希望做成服务：open data processing service，我们希望看到用户把数据往里灌，相当于是公有云的用法，总之数据都放在同一个系统里面。如果以后用户之间希望他们的数据之间发生一些作用，则能够非常容易的做到，只需做一些互相授权就可以。而AWS的RedShift和EMR对各自用户而言都相当于是私有云，它自己处理的东西只在它自己的空间里，如果要跟外部交互，可能必须要借助S3等外界方式。当然了，可能它原本的设计目标就是这样，这个也谈不上优劣，只是目标不同。

在我们这个体系里，因为用户的东西都在一个平台上，所以我们其实也可以像苹果那样开一个应用市场，用户把数据挖掘算法或者清理流程当做一个应用来发布，别人如果想

用可以来买。当然这个可能之后有一系列的如何算钱之类的问题要处理，但平台在这儿，如果商业、产品愿意多考虑，这个事情也是水到渠成的。这个和整个阿里巴巴的愿景是相关的：阿里巴巴想成为数据分享的第一平台。这就真的要有这么大的一个地方做存储，要有那么大的计算能力，让用户有能力来处理大数据，还要保证安全。其实安全也是这次大赛我们比较紧张的地方：我们既要允许用户的代码跑上来，又要保障用户的数据安全，这是个非常大的挑战。

InfoQ：你们组是怎样分工的？

徐常亮：大致有三个方向：数据仓库场景，数据挖掘场景，编程框架场景。其中编程框架不仅是SDK，还会有一些重新定义，会引入一些新的框架。比如Hadoop上有Hive这样用SQL做的，还有Yahoo的Pig——完全是另外一种语言，还有现在很火的Spark，虽说是基于Scala，但数据处理那一层又是抽象了一层出来，提供了groupby、filter这些算子。我们也会提供类似的东西，或者让用户根据我们提供的基础编程能力来定义自己的框架。也可以说我们今后可能会自己再造一套分布式系统的处理语言，或者让用户来创造也有可能。

面向数据仓库的SQL，和数据挖掘有一个Xlab让用户能像写R或者Madlib、MATLAB一样建模，这些是基本算法的包装，都是用户可见的。还有很多模块，大家不一定看得见比如SQL的执行引擎怎么做，数据的存储怎么做。去年我们做了一个比较大的事情，我觉得跟飞天5K可以媲美：飞天5K是单集群5000台，但今天5000台当然也是不够的，你需要有多个5000台。ODPS就有一套系统能够管理多个集群，同时让用户觉得自己只是面对一个集群。这里涉及很多策略，决定你的计算到底在哪个集群上跑，数据在哪个集群的哪台机器上存放，是否在多个集群上都有存放，多个集群间数据的平衡复制怎么做等等。这个东西管的事情是比较的，我们对外希望做到比较透明化。

InfoQ：你个人主要关注的方向是什么？

徐常亮：和计算相关的我都关注。比如数据仓库SQL这块，从解析到执行计划到执行引擎、存储，我都会看。这块是我直接负责。另外还有编程框架这块也是我这边会看的。这两块的同学直接汇报给我。另外，整体ODPS的架构怎么做、上面的控制集群怎么做的，我也会参与。

InfoQ：你们做过的这些东西当中，你觉得最值得跟我们分享的一件事是什么？

徐常亮：可能有一点是比较有难度的，就是怎么做开放。今天我们看Hadoop社区，因为它是开源的，大家对它各方面都有所了解，所以可以基于它的架构出很多新的东西。新东西都是有迹可循的，不是突然就冒出来的，Hadoop上的很多新东西都是基于单机时代的理论，比如数据库上的理论，这些东西都是有一定基础的，可能今天是有人把它们应用到了分布式环境下就成了新东西。

ODPS是开放数据处理服务，而开放不一定是开源，目前飞天和ODPS的代码都还没有公开的计划，即使现在公开出去你也用不了，因为要依赖很多配套设施。所以，在不开源的情况下做开放，这里面需要很好的平衡。

开放，意味着让用户自由、方便的使用我们的计算能力，充分挖掘数据的价值。对ODPS而言，要做到开放，让用户的想象力充分激发，取决于我们能把编程框架做得多漂亮。编程框架很重要。SQL、算法库这些可能更多面向BI的人员，他们可以拿相对现成的东西来用；开放数据处理服务在编程框架上做的事情更多是面向开发者，让他们根据我们开放的引擎、构造通过接口暴露出去，让他们能够用，又不至于把下面的运作模式都暴露出来，既要让用户有很高的定制权，又不违背我们的安全原则和我们对分布式和单机的平衡选择。

MapReduce就是一个很好的方式，因为有人给我们领过这条路，大家觉得这个方法比多线程处理锁的关系要容易很多，仿佛在写一个单机程序，只不过步骤不同。所以，我们提供的MapReduce可以照抄已有的东西。但是今后很多东西可能不是两个步骤就能处理完的，我们想用DAG——就是有向无环图，用比如现在的YARN或者MapReduce 2.0来支持这样的理念，像Hortonworks那个Tez框架就能支持一连串的、若干个task相继的关系，只要你不要成环，能描述依赖关系为有向无环图，我们都把它分解出来，让用户在各个阶段做什么操作，这样来定制。这个东西我们会拿出来给用户用。当然对开发者来说，DAG就比MapReduce复杂一些，但它的处理能力和自由度更高。

我们还在想一些能帮用户做包装的东西，比如写一个wordcount：可能用户写一个MapReduce也很简单，但如果在SQL里面写就只要一个select和groupby就完成了，一句话就覆盖了wordcount的东西。所以，我们能不能给用户再包装一些语义？我们提供一个groupby的算子，用户就可以用。SQL虽然也被称为一门编程语言，但是它毕竟不像我们一般语言的逻辑，你可以写for循环，if之类的，控制能力很强，而SQL就感觉你只能表述一下自己想干什么，后面的细节很难控制，所以开发人员会感觉受局限。提供类

类似于SQL的基本算子——groupby、filter这种想法，在Spark里面也有类似的体现，我们可能也会做类似的事情。我们会考虑是否有一些东西能沉的更底层一些，或者有些东西可以拔高一些，以此来做一些设计或权衡。

当然这个思路可能有很多，我只是提出几个点，如MapReduce、DAG、结合SQL算子来提供高层功能，让用户跟写程序一样。我觉得写一个SQL可能还不是写程序，写程序还是有变量赋值、关系等更多操作。今后我也不知道会不会有别的，但这几个地方我们会下很大的力气，希望在整个大体系下做到安全并提供关键功能，在里面能做迭代、广播等MapReduce不提供的东西，让这些都能通过编程框架放出去，外面的人就能更好的控制分布式系统所具有的能力。如果真能做到的话，我觉得就能把开放做的很好。

InfoQ：所以从某种程度上而言，Hadoop下面出来这么多子项目，也是因为MapReduce的局限性？

徐常亮：某些方面是这样。你看Hadoop 2.0，或者说YARN调度器的出现，很大的原因就是Hadoop 1.0的job tracker只支持MapReduce和map only这两种简单的调度模型。在YARN上你就可以做MPI或者迭代等各方面事情，Spark也可以在YARN上跑，各方面的事情都相对容易。对ODPS而言，因为基于飞天，而飞天的调度——伏羲——从第一天开始就支持YARN今天能支持的模式。从这点也可以看到飞天的发展历程，一开始很多想法还是比Hadoop好的。

InfoQ：如果想提供这些比较丰富的模式，也是可以直接复制现成的子项目的吧？

徐常亮：这是一个做法。比如SQL，因为有标准的定义，我们就可以很容易的复制，只要你写出这个SQL，我的解析器就能按你想的那样解析它，你也想不出别的花招来。这方面已有的理论和体系都比较成熟了。但是Spark你拿出来，虽然这套东西我觉得也很不错，但是毕竟还没有像数据库理论那么定型，或者说自成体系，它有一些缺点。

我们是拿来主义，它好的地方我们就拿过来。Spark基于Scala写的，对于很多同学还是比较陌生的，如果我们把它移植成Java或者Python，这两个语言的社区更大，可能会更容易做。其实Spark这个东西在今天的ODPS上也可以跑起来，但我们这上面跑起来的Spark可能执行体系是完全不一样的。这块也是开放编程框架未来的一个方向，以后比如你可以把Pig也搬上来，都是有可能的。Spark有十几二十个算子，现在已经差不多都能在我们上面跑起来了。

今天我们做飞天也好，ODPS也好，我们做这些自主研发的东西并不意味着我们在闭门造车，我们一定会看外面好的东西，有些东西我们会结合我们自己的场景做整合或者微创新、创新。

InfoQ：对于ODPS，目前业务部门来提需求的多吗？

徐常亮：有一些业务部门的需求很明确，比如业务部门可能做一些数据分析，说我想更快，或者要处理更大的数据，以前支持TB级，现在可能要PB级。有些需求很明确，这些我们就想办法去解决，而且这些在分布式系统下，数据量变大本身就是线性扩展必须解决的问题，否则分布式系统就没有意义了。而处理速度更快这方面，我们也在做一些自己的探索，比如刚才我提到我们在里面做迭代很容易，有一些数据不落地，在实时化处理上，今天我们内部的SQL跑的速度非常快了，比Hive这些都要快。今后感兴趣的话我们可以公布一些benchmark的数据。

另外一些方面，比如编程接口，这些用户都是开发人员，他们的品味都会不一样，所以这就是为什么我们希望把底层包装好、放出来，让开发人员可以自己定制。这样每个人都会高兴。当然今天可能就是只能有一部分人高兴，毕竟把Java、C、PHP、Python的同学放在一起肯定是会意见不同的，我们希望还是把底层的算子、我们定义的一些东西拿出来，这样以后定制能力更高。如果每个人的需求都一个一个去搞，我觉得很难实现。

InfoQ：你觉得实现过的最有挑战的东西是什么？

徐常亮：我觉得那些学术性的、理论性的东西其实都解掉了，也看得到别人已经做好的产品，这方面没什么特别的难题。一路走来，我觉得还是工程问题居多。比如分布式系统里面本来是小概率的事件变成常态，而且因为不断地交互会放大，解决这些小概率事件变成了挺难的事情，因为这些问题往往在你的防范之外，你要怎么定位、解决，是非常有挑战的事情。

再一方面就是早期，不管飞天还是ODPS在人员配备上，人数和工作进度的压力都很大，有一些工程、项目管理上的问题。当然这个不是技术上的挑战了。挑战都是有的，但是一定会解决。

最常见的小概率事件就是设备坏掉。硬盘坏掉大家听到很多，另外网卡也会坏掉。虽然理论上盘古团队会处理硬盘坏掉的问题，但早期不管是调度还是存储都是坐在一起的，所以大家一起处理，更何况我们这儿有真实的场景，有大数据量，可以发现很多问题。

我们之前碰到一个网卡的问题：一台机器大概有千分之一的几率网卡坏了，它坏了又不是全坏，大概是万分之五的机会会把一个数据传错，一个bit会翻转——比如1变成0。总的来说是将近亿分之一的机会出一个错误。但是因为交互的数据量大，就给撞上了。

这个问题怎么发现的呢，刚才提到ODPS的几个特点，其实有一点很重要但是我没说，就是正确性。我们对正确性的要求很高，因为我们的第一个正式的商业客户是小微金服，就是阿里小贷。他们的业务关系到钱，直接关系到你能否把这个钱贷出去，所以我们要对他们的坏账率负责。在这个层面上，我们对准确性要求很高，所以每一次发布之前，我们都会做全批量的验证。这个过程我们需要比对各版本的数据，确保他们都是对的。这个过程，因为我们有数据做对比，所以发现有这么一个问题。这个用户都不一定能发现，他可能某一次跑发现某个数据不能解释，但是跑下一次又ok了，这个事情可能就过去了，因为亿分之一的几率几乎肯定不会再次发生在他头上，可能就换一个人。

发现问题后跟飞天的同学沟通，飞天网络层的同学可能会觉得是不是你们上层逻辑写错了，造成这种随机性，我们就要想办法证明我们上层逻辑没错。后来我们专门做了一个端到端的数据校验checksum。之前我们可能就是像HDFS那样对存储的数据做一个checksum，网络传输过程中做因为会带来一些额外的开销所以以前是没有全做的，但因为发生了这件事就不得不做了。所以我们必须对自己每一次的版本发布做一个很严谨的回归，有任何错误都不能放它过去。这也是我们的一大特色。

InfoQ：最后谈谈你对天池算法竞赛的期待吧？

徐常亮：ODPS是今年1月24日开始商用，开始邀请一些用户进来。我们希望在这次大赛开始的时候，也就是4月底，将整个ODPS正式对外商用。到时候我觉得外面的用户也会反馈很多，借助天池大赛也是看一看我们的竞赛选手对ODPS的反馈。

首先，我们毕竟是做平台出身，在用户体验方面可能做的不太好。我们在平台底层投入很大，但是对交互式的使用、API可能并不是定义的很好。这方面用户如果有反馈，对我们来说是很大的帮助。商业化以后，我们要对外部的方方面面要投入更多。所以希望借着大赛做出相应的改进。

另外，我们这次提供给用户的东西还是比较多。1月我们只有SQL，4月我们会开放Xlib机器算法平台帮助用户建模，这个我们认为还是很有威力的。去年我们内部做了一次大赛，类似这次的，得奖的前几名基本都用了这个超级武器，这个在今天同类产品里面基

本上是没有的。我们也是希望借这次大赛把招牌打响，当然也是看看用户的反馈，让它不仅是威力很大，也要让用户的整个建模流程比较流畅。

另外，我们会把一些用户可自定义编程的东西放出来。当然我们也不希望一次开放太多，前期有MapReduce框架和结合SQL的udf，让用户可以自定义一些函数。这块我们也希望看一下用户的体验。这一块4月不会商用，但是会开放出来做一个测试，可能就是以大赛的用户为主。

最后，我们也在探索这个“数据分享第一平台”该怎么做。今天天猫把数据分享出来让大家建模，如果能达到很好的推荐效果，我们阿里巴巴也会受益很大。因为有几千支队伍，大家会有不同的想法，也许也会有新的东西。在我看来我们要做数据分享，就是要让大家能看到数据的价值。这就要看大家的想象力了。

嘉宾介绍：徐常亮（[@常亮姓徐](#)），北京大学双学士（主修化学，转入IT行业纯属兴趣），普林斯顿大学博士（计算化学方向），曾在纽约时报网络部任职搜索组组长，开发、维护自主开发的搜索引擎，最早期的Amazon ec2、s3和Hadoop用户。2009年加入阿里云，曾负责阿里云分布式平台--飞天--底层基础维护，现在主要负责ODPS平台的架构和开发，产品主要满足数据仓库、分布式编程框架、数据交互等各种场景。

查看原文：[阿里云ODPS的愿景、技术实现与难点](#)

章文嵩：怎样做开源才有意义？

受访者 章文嵩 作者 杨赛

嘉宾介绍：章文嵩，阿里云技术副总裁，核心系统负责人，主要负责基础核心软件研发、推进网络软硬件方面的性能优化、搭建下一代高可扩展低碳低成本的淘宝电子商务基础设施。他也是Linux内核的开发者，著名的Linux集群项目—LVS（Linux Virtual Server）的创始人和主要开发人员，LVS集群代码已在Linux 2.4和2.6的官方内核中，并得到广泛的应用。在架构大型系统、系统软件开发、Linux操作系统、网络和软件开发管理上有着丰富的经验。他一直在自由软件的开发上花费时间，并积极推动开源活动在中国的发展。

InfoQ：大家好，我是InfoQ的主持人，现在在架构师峰会现场。今天我们很高兴邀请到阿里云的技术负责人章文嵩博士来接受我们的采访。第一个问题是有关淘宝-阿里系的开源进程，我们从外面看起来似乎是有三个比较明显的阶段：第一个是说我引入开源的方案来替换掉商业的方案；第二个是我们把开源的方案改进，然后形成一些淘宝自己的T项目，再通过淘蝌蚪等平台把这些项目对外开源；第三个阶段好像是现在在用自研的方案去替换一些开源的方案。您自己是怎么看待这个过程的？

章文嵩：这应该是我们发展的不同阶段。那一开始当然，拿淘宝来说，早年业务的发展很迅速，我关注怎么样快速的能交付，那时候用了很多商用的产品。但是很多商用的产品，随着规模慢慢增大了，商用的产品是不能支撑的，因为商用的产品对我们来说是黑盒子，碰到问题解决的时间周期就特别长，而我们在线上出任何问题，都是需要第一时间去修复的，所以这个黑盒子基本上对我们来说，很多年以前，2009年以前就已经不能接受了。

比如说淘宝的图片，最早存在NetApp上面，NetApp到2006年就支撑不了。它比较好笑的问题是，我们用NetApp最高端的设备，存储容量还够，但是文件数已经放不下了，

因为我们存的图片都是小文件。但是NetApp我们一台放不下，他们就建议我们一台不够换两台，两台不够换四台，那时候就逼得我们不得不自己去研发一个更低成本的，更扩展式的一个分布式存储系统。

我们就自己做，2007年上线，项目名字叫TFS，就是淘宝File System，在2010年9月份我们把它开源了。我们现在的TFS目前存了四十多P的内容，开源的版本就是我们自己用的版本，而且在TFS上我们实际上做了很多架构的取舍，针对这种图片存储已经优化的相对比较极致。所以TFS在性能、价格、稳定性、规模这些方面还很有优势的，我相信在未来很多年会继续存在。

你刚才说第三个阶段是拿自研取代开源的，这个就要看了。我们如果自研的有一些优势，像TFS是自研的也是开源的，但我们每回替换的过程中总是要有收获，比如是不是这东西做得更好了，成本是不是更低，性能更好。我们还是需要拿数据来说话，所以这个可能性都是存在的。

比如WebServer，过去最早阿里是用Apache，然后到2010年我们逐步换成Nginx。Nginx的社区相对来说比较封闭，我们提供的patch，收录的速度很慢，我们做了很多的功能，Nginx上游接受的速度很慢。可能因为它本身背后有一个商业公司，做得Feature跟我们做的Feature很类似。后来我们就不得不Fork了一个项目Tengine，就你所说的T项目，Tengine目前也是完全开源的，开源的版本跟我们用的版本一样，现在是第九大最流行的WebServer之一，目前国内很多公司在用，海外也有。这种可能性都存在。像TFS新浪微博在用，新浪微博的很多图片也存在上面。原来我一个早先的同事正好在那边做存储项目，然后我当然给他建议，用TFS会解决很多问题，我们已经开源了。然后他就把它架起来，我们也提供一定的帮助。这在阿里投资新浪微博前就发生了。

我们一方面自己研发的有可能会开源出来，另一方面我们在开源的项目上再做了一个开源项目。

InfoQ：阿里总体来说还是一个业务导向的公司，研发过程肯定是产品导向，而不是说先有spec，基于spec去做，一切目的是快速完成产品经理的需求，这个是首要的任务。开源社区基本上正好过程是反过来的：他们先去讨论，我有这么个想法，这么做对不对。然后大家讨论说，你的想法好，按照你的做法去做吧，然后这样才出来实现。所以他

有个矛盾，就比如说Google做Android，Android代码一开始它是被内核社区曾经驱逐过。就是他们先做出来了，然后放进去，但是之前没有讨论，所以社区的人就不接受。我听说阿里也有因为类似的原因，没有被上游认可的Patch。从业务的角度，“产品优先”才是正确的；但是从工程的角度，可能是“思路验证优先”是更正确的。您怎么去做这个平衡？

章文嵩：我觉得要分开来看这个问题。我们开源的东西往往是有共用的一些价值，可以拿到别的应用场景复用的，这样开源才有意义。你刚刚说那个业务上产品导向也没错，比如淘宝跟天猫的平台上面，我们有四千多个应用，应用开发本身我们要业务导向，要做的很快，那没错。但是我们里面沉淀了一些东西，下面的中间件或者底层的研发平台，这种底层的支撑平台、基础平台这方面来说实际上有共通性，那这方面可能更容易开源。这方面的变化节奏，就不像上面的业务会来的那么快，而且本身我们做一个底层或者中间件，肯定都会考虑不光是一个应用要用，很多其他的应用也会用。所以这里面赋予我们开发的时间，包括架构的设计时间会更长。

往往开源的项目是属于那些有共用的，相对偏底层的一些偏基础的一些东西，所以这两者我觉得如果分开来看不会那么多矛盾了。我们当然响应业务的需求，怎么把上面的应用、产品做得更快，那我们的开源的大部分是有一些共用价值的东西，可以有更多的时间来做，而且要好好规划的，不光是为一个产品来做，要为更多的产品来做。

InfoQ：第三个问题是有关自研的选择。一般来说团队去选择自研方案，首先可能是因为有一些开源的实现，但是不够成熟，但是业务又马上需要。要么另一种原因就是说，我觉得我有信心，做的比现在的那些方案都好，才去这么做。但是如果业务不是马上需要的情况，其实也可以等他们过两年，然后开源项目就成熟，就变成可用的了。但是这个时机可能也是会比较早的，就是先进去做有可能就成炮灰了，所以到这种情况就有三个选择：第一个选择就是我立刻把所有兵力都投进去，然后搞自研，拼一把；第二个选择是我找一个看起来还不错的开源项目，我去投人进去，然后把它搞起来，以后如果这个项目发展起来，我就有主导权；第三个选择是我先观望两年再说，如果是你来选择的话，如果有一个领域你觉得挺有前景的，但是现在处在这个时间段，你会怎么选择？

章文嵩：我会选择第二种。我会先看开源的解决方案，有哪些可能的方案存在。我们会去评估一下这些开源方案，它目前的所处的状态、成熟度怎么样，有哪些功能是我们期望拿到的，可能有哪一些目前还没有做到，也会评估我们能不能在这个基础上，在上面

加东西。如果整个框架可以很容易去扩展，里面加东西也比较灵活方便的话，那至少跟我们定义想做的东西是匹配的。长远来说架构上面如果没什么大的冲突，那我们在上面发展，实际上是非常好的事情，一方面可以节约人力投入，自己可能少走一些弯路，将来还有机会可能参与到整个社区里面。社区的话，不光是我们一家公司在做，有可能很多家公司都在做，大家如果形成一个生态系统，每家公司的投入相对都会少。

关键是这个开源的东西是不是服务我们的业务，如果服务我们的业务很好，我们投入的人少，那何乐而不为。我会倾向你所说的第二种；除非是评估了一圈没有合适的，但我想做的东西又可能不远的将来马上要要，那我们有可能会选择自己去先投入去看一看，先尝试一下，这也是有可能的。

InfoQ: 不会担心以后对项目失去掌控权？

章文嵩：我觉得这个不用担心。我们不说掌控权，大部分说影响力，这个影响力是看我们对这个开源项目的贡献来定的。大家也知道，因为在社区贡献，大家做的贡献多，影响力自然就会大起来。

即便不行，碰到情况，假设原来的社区碰到问题，我们也可以在这个基础上fork出来做另一个开源项目，我觉得未来这个路径都是通的，我觉得不存在太大的问题。

InfoQ: 第四个问题，现在有一种说法是，开源的发展促进了IT行业人才的流动。比如说这个人在一个公司搞内核，或者是搞OpenStack，或者是Ceph、Hadoop这样的，那么这个东西是通的。假设说他想去换一个公司，到了新公司如果也用一样的架构，他就可经立刻复用了。但是如果到了新公司，他的技术体系是另外一套，他不管有多少经验，先得把这套东西再学一遍，学个几星期、几个月，然后才能有真正的贡献。但是现在国内的情况，几个互联网巨头，包括像阿里也是有很多私有的项目或者分支，会造成上述的问题。你对这个问题是怎么看的？是觉得它不可避免，还是可以改进？

章文嵩：你问的是很好的一个问题。我觉得要看这个产品上面获得的竞争力对整个公司的业务的重要程度来定。拿淘宝、天猫来说，我们真正是建一个大规模的交易平台，在上面的很多业务做的好坏就是靠数据，比如说信用模型，用户的交易记录，导致每一个商家都有相应的信用的数据，然后消费者也有信用的数据。实际上对整个淘宝、天猫来说，这个数据是最关键的，数据是日积月累，拿不走。所以数据是我们最关键的竞争。

软件当然也是我们的一种能力。我们能实现这么大规模的一个平台的能力，我们把它开源出去，对我们核心的业务价值不会有损失，别人拿这个软件再建另一套淘宝、天猫，也很难与此竞争的。因为如果建一个软件的平台，要找到合适的人，大概花一些时间也能建出来，但建出来在这个平台上没有任何数据，那试问一个消费者，他愿意在淘宝、天猫的平台上做交易，还是跑到一个空空如也的平台上做交易？那显然应该是前者。当然阿里本身也很开放，所以淘宝、天猫的很多技术平台我们已经对外开源了。

那是不是我们有产品是没开源的？当然有了。比如说我们目前的飞天平台是没对外开源的。飞天能处理五千台或者未来上万台的一个规模，是大规模的数据处理、存储，基本上是分布式的一个操作系统平台。对阿里来说，这也是非常大的一个竞争力，尤其在云计算上面。如果要保持这个竞争力，当然目前的状态，目前是没有开源。除非有一天，我自己个人的一个想法就是，如果将来我们的云计算的规模、优势已经非常大了，我们已经不再担心别人拿我们的软件再去搭一套平台跟我们竞争，我们竞争优势已经在别的方面了，那开源也是有可能的。所以我觉得这个是不是用开源不开源，是用不同的状态来看待这个问题的。

InfoQ：最后一个问题关于开源本身。开源模式的价值，其实它最大价值是说我在全球范围内去找到用户和贡献者，不分任何的国界。其实包括LVS项目的成功，其实很大一部分原因也是，它一开始就是以国际化的项目去做的，所以你认为有必要纠结于国产的开源项目，或者是由国人发起来的开源项目这种说法？开源项目是否都以国际化的思路去做会比较好？

章文嵩：我觉得关键是看自己面对的客户群。如果我们的客户群都在国内，那没必要，如果整个网页都写成中文的，用户会更接近，更容易使用，这是最关键的。如果我们的客户群是全世界的，当然要让全世界的用户更容易的了解我们的项目，就要用英文来做。我觉得不存在国产的或者国际的开源项目，关键看我们的定位。

1998年我做LVS项目的时候，说那时候也没有太多选择，因为那时候很多开源项目的邮件列表都是在海外，那肯定做了一样东西，要让全世界更多的人来了解，所以那时候一上来LVS的网站，一开始都是用英文写，做了哪个版本发布，哪些功能，都是跑到邮件列表里去发邮件，用英文发。回过头来，还是我们这个开源项目的客户群来决定的。

查看原文：[章文嵩：怎样做开源才有意义？](#)

架构师

www.infoq.com/cn/architect

每月8号出版





4

支付宝相关技术

小微金服前端技术专家伯约访谈：

相信数据的力量

作者 水羽哲 发布于 2013年7月19日

7月13日，第三届[阿里技术嘉年华](#)在杭州召开，大会为期两天，涵盖“前端技术”、“业务架构&后端技术”、“搜索”、“大数据应用”、“无线技术”以及“测试”等专场，有将近3000人参与。在“前端技术”分论坛上，来自小微金服的[王磊](#)（花名：伯约）以“用户行为监控”为主题分享了小微金服的用户体验验证体系、Session&Tracker用户行为监控解决方案和用户行为动画还原方案等，InfoQ在现场对他进行了采访。

InfoQ：首先请您做一下我介绍？

伯约：我是伯约（王磊），来自支付宝，主要是负责用户行为监控体系的建设，也会兼顾去做核心产品的一些用户体验工作。

InfoQ：您主要是做通过数据监控来指导产品的改进工作，那么现在如果支付宝有新的产品设计方案时，需要在监控方面提前做哪些工作？

伯约：现在支付宝的产品设计环节，我们会考虑后期的数据需求，设计师会告诉前端在设计中在哪里进行埋点、布控。前端根据设计师所提供的信息，将对应的代码部署上线，这是一个人肉的方式。另外，我们还有一套自动化布点的方案，它会通过一个脚本，根据页面元素的一些特点，例如元素的父对象、URL的域名前缀等规则，自动拼接出当前元素的名称，实现自动埋点。

InfoQ：对于数据采集，如果早期可能采集了三个点，那么后期我才发现这三个点不够，如何来补数据？

伯约：就像刚才说的，我们有自动化埋点，可以通过这套自动化埋点的方案去补全数据，但是现在有一个问题是自动化埋点的可读性并不是那么好，因为要考虑到设计师，

包括产品经理可能是非技术出身的，如果说那个点不是他自己预设的，那么他们在后期自己去看的时候会非常困难。

InfoQ：所以现在你们想到的、想不到的元素都会有埋点，会不会采集的数据量太大了？

伯约：这个会考虑，但是目前来看，支付宝的数据存储能力应该是非常强大的，我们现在对数据也非常关注，比如说刚才分享中提到的案例数据，我们会对数据做一些分级，保留对以后数据挖掘有用的数据，剩下的数据则可能存一、两个月就定期的去做一些清理。

InfoQ：在数据分析方面，你们是和数据仓库团队来进行合作？

伯约：是的，和数据仓库团队合作。

InfoQ：产品的设计可以通过后期的数据来做论证，那是否可以在产品上线之前就建立数据模型提供预警机制呢？

伯约：这个是可以的，我们在产品上对应的地方去做埋点，通过一个类似于产品魔方的平台，去做这个事情，进而去监控用户在操作过程中所遇到的问题。我们可以通过通用的全局数据，比如说在一段时间的走势、访问量波动等来判断产品是否达到预期。也可以通过细节的数据，比如预设的产品用户操作路径分析具体的产品体验问题。

InfoQ：之前您提到的那个案例最后得出结论是：对于一个具体的问题，并不是说在一个页面，或者两个页面就能够看出问题所在的，因为支付宝毕竟是流程中的一环，那么可能在他之前别的网站时遇到了一些问题了，这时该怎么分析？

伯约：这个可能就需要其他网站授权了，说实话是挺难的，现在我们看到公司也有一些动作，把外围相关的数据打包进来，这样我们就可以看到用户在整个购物链路里面的体验问题了。我觉得这可能跟我们在数据上面的一些思考是有些关系的，有了这些外围的数据之后，整个链路真的就能打通了！

InfoQ：刚才分享你提到了一个Session&Tracker的解决方案能够实现用户状态的还原动画，同时您还提到这个方案还存在一些问题？

伯约：主要是数据权限控制的问题，因为如果通过这种方式会涉及到用户的敏感信息，因此需要确定哪些人可以看、哪些人不可以看，这真的是一个非常大的系统工程，如果没有办法去界定哪些人可以看，万一真的用户的隐私受到侵犯，这是坚决不允许的。

InfoQ：Google或者Facebook他们应该有一套方案，你们是否了解他们的一些做法？

伯约：市面上现在比较现成的一些行为监控的方法，我们都有去看过，但其实很多是开放给所有的个人甚至企业的，这一套监控体系可能跟我们想要的还有一点不一样，他们可能更关注全局面数据的走势，但我们其实很多时候除了这部分的数据之外，还希望知道用户的操作细节，通过结合这些具体的信息才能更好的去做一些体验的优化。

InfoQ：我看到你的介绍里提到你十分相信数据的力量，能否谈一下您对数据的看法？

伯约：我觉得有些时候数据只是一种习惯跟信仰，可能长期以来，我无论做什么都会去做一些数据的跟踪和采集工作，比如说我前段时间在看一些人做的同步并行下载，国外其实已经有很多大牛在做这个事情了，但我自己觉得任何东西都是需要亲自去验证，所以我会做对应的一些数据测试，因为大牛们的文章里是没有具体操作的，所以还需要有一些具体的数据去验证，那么验证之后我们再看到这个信息时，可以通过数据更直观的把结果展现出来，跟大家单纯看文章的收获是不一样的。

我觉得无论做什么，特别是用户体验，其实还是偏感性的工作，如果说没有数据，也是不科学的。有数据指标，我们就可以去做相对的比较，而不是单纯去看它的绝对值，对于体验也更可衡量。

InfoQ：是的，感性的设计需要理性的支撑，那最后一个问题，作为一个前端的大牛，希望能给这些前端的同学提一些建议如何提升自己的技能？

伯约：我觉得其实还是习惯，平时遇到问题是学习跟成长的一个最好的机会。比如，在我们团队中就会发现有很多同学很忙、压力很大，在项目过程里面会遇到非常多的问题，但是可能解决完问题就接着忙了。很多时候去解决一个问题是很简单的，但是知道所解决问题背后的原理是什么，才是成长中最重要的！如果你一直忙，一直在重复做你以前在做的事情，而没有去问为什么，是不会有成长的！但如果说有这种积累的习惯，不用多久，自然而然就会觉得自己变成大牛了，当然我不觉得自己是大牛。

查看原文：[小微金服前端技术专家伯约访谈：相信数据的力量](#)

蔡学镛谈复杂事务处理（CEP）

作者 胡键 发布于 2010年10月13日

蔡学镛，台湾清华大学硕士，曾任程序员、技术经理、技术总监。除了将技术知识用于软件开发相关的工作之外，他也担任过培训班讲师、研讨会讲师、技术图书翻译与编辑、技术专栏作家等工作。他撰写的《Java夜未眠》一书，至今仍常被IT圈提及。蔡学镛于1995年就开始玩Java，在2001年进入.NET领域，近年则倾向于使用动态语言。因为工作关系，他接触到CEP（Complex Event Processing），并完成了一套相应的系统。InfoQ编辑在QCon全球企业开发大会（北京站）期间，就CEP相关的技术，企业采用CEP的价值等话题采访了蔡学镛。

InfoQ：按惯例，请您先做一下自我介绍。然后，再向我们的读者介绍一下您在这次大会上讨论的主题？

蔡学镛（以下简称学镛）：大家好，我是支付宝技术部的蔡学镛（编辑注：现已离职），这次到QCON上演讲的主题是CEP，刚好也算是我在支付宝的一个项目的心得体会。基本上，CEP系统能够让我们把很多来自四面八方，各个系统的一些事件搜集起来，从中萃取出一些更大、更有意义、更有用的信息，这些信息不管是对制订公司、运营和营销的决策，还是对发现系统潜在的问题、漏洞和威胁都有帮助。这种系统就是CEP系统，它是准实时的，跟其他一些系统，比方说数据挖掘，又不太一样。那些系统是在事后才对过去的历史数据进行挖掘。这次跟大家分享的就是CEP这个技术。

InfoQ：在未来的企业中，CEP会充当一个什么样的角色，它如何跟我们企业中其他的系统进行交互，完成企业的业务目标？

学镛：大家之所以会对CEP比较陌生的原因，是因为CEP过去主要用在银行业、金融业。例如，他们可以通过CEP系统来判断某一些持卡人的信用卡操作是有问题的，可能是这个卡掉了，或者可能因为卡所处的状态而就把卡给锁住了。但是这样就太局限了，

我认为CEP的用途非常广，它可以用在任何企业里面。任何一家企业里面都有许多系统，系统之间交互和系统内部都会产生大量信息，如果我们能把这些信息善加分析、处理和使用的话，会对企业经营有很大帮助。至于怎样应用，就很难讲了，必须要看企业是哪种类型。但我相信，不管是什么样的企业都可以使用到CEP。

InfoQ：在选择CEP产品和解决方案的时候，我们要考虑哪些因素？市面上主要的CEP产品有哪些，各有什么优缺点？

学镛：市面上提供CEP产品的厂商，基本上就是做数据库的那几家，我就不特别提及了。另外一些较小的厂商，如TIBCO，也不能说它小，它其实是CEP最大的厂商，号称有40%的市场占有率，而且我看他们的CEP确实也不错。还有一些小厂商，像StreamBase；另外一些开源的，像EsperTech，都可以在产品选型时考虑。

那么，要考虑哪些因素呢？我觉得有两大重点。第一，所提供的EPL语言。EPL语言可以让我们描述哪些事件，在什么状况发生了我们感兴趣的符合条件的事件。这个描述语言本身如果不灵活，表达能力不够好的话，那势必这个系统买回来对你的作用也不大，甚至可能会造成很严重的误报警率。误报警率一高，这个系统就不堪使用，所以我觉得EPL是最重要的一一个考量点。除了EPL之外，我认为引擎本身的效果也非常的重要，因为CEP面对的是海量事件、海量数据，如果没有一个很强有力的算法和引擎的话，整个系统效率绝对会是一个非常大的问题。

InfoQ：既然市面上已经有了这么多CEP的产品，为什么支付宝还选择自行开发？这个自行开发的解决方案，跟这些产品相比，有什么独到之处？

学镛：其实，早期我们评估过是否采用开源CEP框架。但我的评估结果是它的能力基本上达不到我们的要求，原因就是刚才我提到的第一点，EPL的描述能力。所以，我认为可能还是自行开发会好一些。我们采用了状态机的设计方案，后来确实证明这个方案完全符合支付宝现金流的需求，我们现在不管是担保交易，还是更复杂的业务，再复杂的规则，即便整个状态图大如地图，这个引擎都能够把它描述出来。

确实，现在回想起来，当时决定自行开发引擎的做法是对的。但我不建议所有企业都自行开发，还是得量力而为。因为，如果你做的东西是别人已经做好的话，你可能得评

估，自己有没有办法做得更好，或者是自己花一点钱就买得到的东西，是不是得自己开发？

InfoQ：这个CEP解决方案目前在支付宝内部的应用情况怎样，目前处于什么样的阶段，大概主要应用在哪些业务中？

学镛：支付宝内部其实很多部门都在用CEP了，虽然我们不会特别强调CEP这个词，但其实都已经在用这样的概念了。采用的解决方案有很多，有买来的，有自行开发的。例如，支付宝里面有一个非常有名、非常经典的例子，CTU。CTU大家应该都看过，这是一个24小时反恐的政府组织，他们做的事情就是要针对恐怖分子去做一些应对措施。支付宝因为牵涉到金融，所以我们也对这部分特别谨慎，这一直是支付宝非常核心的技术，大量地在使用CEP。而且每隔一段时间就会发布新的供给模型，让支付宝会更加完善。这是一个例子，它们是用来防止网络攻击以及洗钱等这些行为。另外，支付宝还有一些比较小的CEP应用。比如，在营销、协助公司决定生产或判断趋势方面都有应用。至于我开发的这个总督系统，其实是比较新的，它的目的就是用来侦测我们各个系统之间的信息交互有没有问题，属于比较动态的系统校验。这些都是支付宝里面用到的CEP。

InfoQ：您在演讲时候也提到准确性的问题。您开发的这个CEP解决方案，在性能和准确性方面表现如何？有没有相应的数据或者案例来支撑您的说法？

学镛：由于目前这个产品还属于比较前期的阶段，所以还没有特别去针对它收集相关数据，这个数据我给不出来。但是它的效率的确非常好，因为我们现在的所有的计算，都是在内存里面而不是在数据库中进行的，它的效率绝对会比数据库方式来得快。我们现在就是几台服务器，处理支付宝目前的那些系统已经足够了，因此也没有特别针对项目进行调整。当前的效率都还可以，但我没办法给出比较确切的数据。

InfoQ：那准确性呢？

学镛：关于它，我倒是可以给出些数据。我们做过一些像担保交易、即时到帐等这类支付宝比较经典的应用，基本上所有的支付宝应用都是担保交易和即时到帐，它们大概一直占了百分之八九十，所以已经算是涵盖支付宝大部分的应用。监控这些业务，之前的误报率是3%，已经很低了。而且现在，一旦发生误报警，我们就会检查为什么会误报警，看看规则里面少了哪些判断，然后再去补充规则，所以现在又降低了，可能是

1%多点，而且又在不断地持续降低，这是我们这个总督系统的现状。至于其他像CTU的方式，那边的数据我就不清楚了，那边数据的效率可能会高一些。

InfoQ：您曾经说过，在您开发的CEP解决方案中有一个规则DSL。DSL现在相当热门，您能不能对此详细介绍一下？

学镛：我认为状态机的描述，最好能够让它非常简单，这样就可以让更多人能够去写规则，放到我们的这个平台上，这个CEP就能应用到更多场合。如何让这个规则可以很简单、很容易地写呢？DSL是关键。DSL是第一种方式，另一种方式就是利用易用的绘图工具，让我们可以去做一些表单设置或者是状态图的绘制。我们先是采取DSL的方式，未来当人力更充足时，我们就会去做这个图形工具，我相信这两个阶段都达到之后，使用我们的系统或者是在上线之后再修改规则，都会更容易、更快。

InfoQ：您刚才说过支付宝的CEP解决方案，山头林立，未来是否有可能出现一个统一的CEP平台？

学镛：有可能，这就是我的野心了。我期望我做的这个平台，能够尽量多的发挥它的价值。我认为，因为这个平台有非常大的潜力，所有希望借目前的这些应用，慢慢扩展到未来也能够去做网络攻击、洗钱的侦测。当系统做到更好、更稳定的时候，能够把它推荐到阿里巴巴的其他子公司，像淘宝或B2B，让他们也能够用。当然，这是我的目标啦，都还是未知数，现在言之过早。

InfoQ：祝您早日完全心愿。如果我想说服一家企业去采用CEP的话，您有何建议？回顾当初支付宝内部决定采用CEP解决方案的过程，有没有一些经验可以跟我们分享？

学镛：企业如果想采用CEP，不应该是为了用CEP而用CEP，应该是确切发现公司里面的某些地方能够通过CEP带来价值。不管是营销，还是其他方面，先要找到应用点。找到之后，然后再去评估这个应用点值不值得花大钱去做CEP。因为提供CEP系统的厂商都是大厂商，东西都挺贵，这个可能要考量的第一点。如决定采用，那这个应用值不值得投入这么多钱去买一个CEP，或者是投入这么多人力开发一个CEP，说到底还是应用。你刚才的第二个问题是？

InfoQ：因为支付宝现在已经有经验了，有什么可供借鉴的？

学镛：支付宝倒是没什么问题，在支付宝内部要推行CEP非常容易，因为公司本身的属性虽说是网络业，但有一大部分是金融业，所以大家的安全意识非常非常的强。就以CTU来说，在支付宝里面算是最重要、最大的CEP了。其实，由于支付宝一成立就立刻有了这个CTU，它等于是伴随着支付宝公司成长的，在支付宝没有推广不下的可能。

InfoQ：对于想要了解和学习CEP的开发者，您有什么建议，能否给出一些参考资料和学习方法？

学镛：我觉得可以先从各个CEP厂商的技术白皮书着手。技术白皮书一般都较为简短，可能会比较好读。像TIBCO、StreamBase、Oracle，这几家的CEP，都可以从这里着手。另外，有本书其实挺好的，去年刚出版，国内可能没有引进，或许可以通过英文站点从国外购买，书名是《Event Processing: Designing IT System for Agile Companies》，McGraw Hill出版社出版，作者名字不记得了。这本书非常棒。而且这本书里面提到的不只是CEP，基本上CEP只占一章，其他讲的都是事件处理，重点是怎么结合起来成为一个整体，我觉得CEP不是单独存在的，它会搭配很多其他的事件处理的概念，所以这本书整个对它进行了一个完整的说明。另外，今年开始也会陆续有一些面向程序员，不是着重架构的，让程序员看的CEP书籍。像是Manning出版社今年就会出一本，里面应该有CEP，这些都是不错的参考资料。相对其他领域，如SOA，大家对它可以说是耳熟能详了，说知道CEP的确实是少得可怜。

查看原文：[采访：蔡学镛谈复杂事务处理 \(CEP\)](#)

程立谈大规模SOA系统

受访者 程立 作者 李明 (nasi) 发布于 2009年12月1日

嘉宾简介：程立，支付宝公司的首席架构师。他从2004年起参与支付宝与淘宝网的建设，2005年正式成为支付宝人，随着支付宝的业务与技术的成长，从开发工程师、架构师到首席架构师一路走来，全身心投入并见证了支付宝业务与系统发展的完整过程。在加入支付宝之前，他在上海交通大学攻读计算机专业博士，同时作为Sun公司的兼职顾问，参加过电信、制造、互联网、教育等各个行业Java企业系统的咨询与建设。

InfoQ: 大家好，这里是首届QCon Beijing的现场，现在坐在我的旁边是的支付宝的首席架构师程立。先给大家介绍一下，支付宝架构发展到今天，经历哪些时期，都有哪些里程碑？

我回忆一下，支付宝系统架构发展大概有这么几点。我本人大概是2004年下半年参与支付宝系统建设的。当时的目标，支付宝系统是面向整个互联网，而不是淘宝网内部的一个产品。那应该说是支付宝系统的一个起点，那当时非常的简单，就是一个应用程序，提供了我们所有的功能。功能也不多，有我们基本的支付功能，还有清算的功能，基本的会员管理功能，包括后台管理功能，这是支付宝的第一个里程碑，从无到有的过程。

第二个阶段是我正式加入支付宝，2005年2月份，进来之后，我们做了一件事情，当时作为支付宝三期，这期项目实际上是一个真正意义上的支付宝，因为支付宝不仅仅是一个交易系统，支持各种各样的业务。我们希望它能够支持各种各样的交易流程，包括担保型的支付宝交易、即时到账的交易等等都会在里面。但是当时支付宝主体的系统还是在一个应用程序里面——我们面向前端用户的系统，包括我们最后交易、支付的、会员管理的，都在这么一个系统里，这是第二阶段。在这之后呢，我印象很深刻，就是在2005年上半年，这个系统里面有20个子工程，到2005年的下半年，不到半年时间翻了一倍。当然我们也尽可能把一些业务做成独立的系统，但是发现支付宝大量的业务它的耦

合度还是挺高的，所以我们不能把它做到一个系统里面。那这应该是支付宝的第二个阶段。就是我们开始在一个核心的主体应用里面，不断去堆积我们的业务功能，发展很快。那在2006年初的时候，我们觉得这个不是长久之计，于是我们开始探索怎么样用SOA的思路去解决这样的问题，找一个切入点的话，我们找的是用ESB，把一些可以异步处理的，耦合度不高的业务拆开，做成单独的业务服务。那这个阶段，大概持续了有半年左右。这个时候我们发现，虽然我们把一些相对来说比较边缘的业务拆开，但是对我们核心业务，我们的交易、支付、处理、会员，他们之间耦合的非常紧，基于ESB，基于消息，我们很难把他们拆开。

去年的时候，我们在讨论这样一个问题，我们不能在持续一个应用中维护这么多业务，当时我们支付宝，开发规模已经上百人，十几个并行项目，每周一到两次发布，工作在这么一个codebase上面，肯定是不可接受的。那我们当时选择交易作为一个点，前面所说的原因，交易系统当时响应业务的变化已经很吃力了，一个应用程序里面，这么复杂的商业逻辑。我们既要响应前端的快速处理，又要保证交易逻辑不出任何错误，那么选择交易服务，针对这个服务我们也从技术上做一些重要的构思，我们建立了自己的服务框架。我们之前也做了很多调研，把一些我们不需要的功能拿掉，再把我们特别关注的一些功能，放上去，那这个项目大概持续了有四个月左右。这是我们支付宝的第一组服务，它是可以支持我们核心业务的。

在这个基础上，我们就去拿支付宝最重要的功能去改造去提升了——支付宝的帐务会计体系，还有支付宝的客户体系。关于这两个体系的建设，我们的思考，从几年前就开始了。那么支付宝最初上线的时候，业务的范围非常小，但是随着支付宝整个业务领域的拓展，原有的帐务会计体系不论是灵活性还有专业化的程度都不能满足需要。客户的模型，也不能满足很多业务的需要。首先是在这两个业务模型的基础上，做一个很大的提升。然后我们SOA整个一套业务的思想，把它分装成真正可以支持业务发展的两套服务——会计帐务服务以及客户信息服务。这两个服务无论从业务和技术上都有挑战。业务上挑战是什么？就是我们业务服务模型是不是能够支撑支付宝未来数年里持续的发展。还有一个挑战就是技术上的挑战。帐务处理一旦拿出来后，那我们所有的业务，和我们的帐务处理之间全部都是分布的，很显然这个事务就是很关键的问题，还有它未来的伸缩性，当时我们的会员数可能接近1个亿。我们的交易笔数也是每年翻3番。那这个我们也做了考虑，解决了一些技术上的问题，同时也对交易服务框架做了一个提升。

那么这样的服务全部做完了的话，大概大半年的时间，这个时间是比较慢的，慢的主要瓶颈是，我觉得前面我们讲的业务服务，怎么去识别服务，怎么去搭建一个可以支撑业

务持续发展的模型，这块我们觉得难度非常大，需要有非常强的业务架构思维的人来做这样的事情，那么在支付宝的话，我们这样的人相对比较稀缺，所以说这样的事情我们只能一件一件去做。可能这两个基础的服务体系搭建完了之后，支付宝的基础服务便形成了一个三角：帐务会计、会员，还有交易，在这上面，我们发现，有了他们这个基础，我们又产生了很多服务。这时，支付宝应该是一个全面铺开的阶段。SOA对像支付宝这样的企业，既要响应互联网快速变化，又要保证核心业务处理绝对安全可靠，可持续稳定服务。这是必须的。我们不能想象，支付宝现在数百位开发人员，在一个单体应用程序上，怎么去做。但现在，我们不但知道如何去做，而且做的更好，我们现在支付宝上海也有研发中心，都能够一起协作，这是SOA给我们带来的价值。

InfoQ: 您基本上把支付宝发展的历程完整地说了一下。您刚才也谈到了业务，我觉得在支付中业务需求的变化应当是非常快的，支付宝对于这个业务变化是做出了哪些改变，这个是不是支付宝引入SOA的一个主要的原因？

之前提到了，引入SOA，首先要让所有人达成一个共识。不光是技术人员，不光是业务人员，还有我们的高层，要达成一个共识。实际上当时支付宝所有人都看到了，存在一个挑战，就是前面提到的前端快速响应用户需求、后端持续稳定服务，这个快和稳的矛盾。我们认为首先SOA会帮我们把快和稳切开，这样的话，才能够让前端真正快起来，SOA在这方面确实给我们提供了价值。但是SOA的话，让我们实现真正的业务敏捷的话，它有更深的意义，这是我们当时没有看到的，那就是SOA是真正能够让我们的系统和我们的业务架构对齐。因为它统一了业务和技术，通过服务的概念统一起来了。这样我就能让业务不但符合当前的需要，而且能够符合长远发展的需要，能够符合整个架构，甚至和公司的整个使命、甚至是愿景对齐。这些方面，其实现在我们也在探索。

InfoQ: 现在支付宝的这个架构是非常优秀的，您觉得对于一个企业来说，一个优秀的架构意味着什么呢？

架构的话，我个人觉得，可以从很多层面去理解。我刚到支付宝的话，我看到的架构很狭隘，我就看到了，比如说我写一个框架或者一个公共基础类的，这就是架构，这是我的理解。后来我发现这个架构仅仅局限于技术的话，真的是太小了，我们会把业务考虑进去。我们希望整个架构，既包含业务又包含技术。双方的基础是统一的，只不过大家面向的客户是不一样的。大家可能采用的技术手段是不一样的。但是从概念层上里看是

统一的。再后来，我发现，光有业务、技术架构，以用户的需求去驱动，也不够。当我们一次、两次、三次满足用户需求之后，突然发现用户的某个需求可能跟我们原来的设计模型和基础设施是不匹配的，这样如果要快速响应，我们很难去做到。所以说我们这时候，认识到架构它不是直接面对业务的需求，还有一个输入是企业的战略，架构要与企业的战略对齐，这样的话，才能支持企业长期的发展。战略会规划企业未来几年的方向，可能的规模，重点的任务。如果架构能够跟它对齐，那架构将会有更强的生命力。

InfoQ: 那，我的理解就是说一个企业要想看这个架构适不适合它，就是看它是否与企业的战略对齐呢？

是的。因为说战略，可能有些虚。如果是实了，就是说架构不但要支持当前的需求，对业务需求要有一个持续稳定的支持能力。而且这个持续稳定，我们很难去估计未来的需求，那这个时候就只能靠战略，如果是和战略对齐的，我们会发现战略里面考虑的未来重点方向，都在架构中有落实，无论是业务上，还是技术上，这样我们就更加信心，架构是能够支持企业长期发展的。

InfoQ: 下面问一些技术方面的问题。支付宝的系统在SOA化以后，觉得面对的一个很大的挑战应该是分布式事务的问题，因为事务对于一个交易平台来说应该是非常重要的，那么在这个方面有什么经验给大家讲一下？

我其实第一次遇到事务问题，是在建设交易系统的时候，其实交易一旦拿出来，我们就已经遇到事务的问题了。交易系统作为一个单独的系统，帐务系统在另外一个系统里面，那怎么保证他们之间的事务一致性。当时，由于交易系统本身业务的特殊性，我们当时采用的就是重置的方式，在交易的过程中，可能请求某些帐务，这时帐务处理可能会失败，这个时候我们保证通过恰当安排交易系统的处理顺序，如果是由于业务原因造成帐务处理失败，那交易系统是能够正常地把业务回滚的。那如果由于是系统的原因，那交易系统会处于一个等待系统重置处理的状态，这个系统会安全地通过重置的方式，让最后的交易与帐务处理达成一致。交易处理是幂等的，帐务处理也是幂等的，这是当时的一个处理方案。

这个方案比较简单，但是能满足帐户和交易之间是业务的需要。后来提到支付宝把整个帐户的体系进行SOA化的时候，那帐户系统面对的不只是交易系统，和所有系统都有这种分布式处理的交易。我们需要提升这个框架，满足多系统，多层次的服务之间做分布

式事务的需要，所以当时我们想了很多的方案，我们也是看了一些标准，那个时候正好是2007年的上半年左右，正好WS Transaction作为标准推出来了，我们看了看WS-AtomicTransaction等，而且我们实际做了相应测试的环境，去做一个概念验证，最后发现在实在是成本太高了。完成一个事务，20多条的消息的交互，其中只有1条是业务消息，其他都是系统之间的协议消息。那对于支付宝互联网应用，必须在很短时间做出响应的交互，我们无法支付这样的成本，那我们只能去考虑一些山寨的解决方案，当时对我们最有启发的一个是Ebay，关于系统设计，他又一些最佳实践。其中有一条，就是关于一致性的。我们在满足业务需求的情况下，允许有不一致的情况出现。这对业务是允许的，这是一个很重要的思路，虽然他没告诉你怎么做。另外呢，亚马逊关于这些方面也有一些很好的阐述，比如说亚马逊有一位架构师，他关于这方面写了一篇文章，当然他也是面向他的场景，比如他们的分布式存储如何保持一致性，这些对我们也有一些启发，那关于事务我们需要的ACID到底哪些是必须保证的，哪些是可以放宽的，那atomic原子性这个是必须要保证的，还有isolation，对事物的处理过程中，对事物所涉及到的资源中，我们一定要控制起来，否则如果你再用这个资源做事务中的某些事情的时候，就可能被人用掉了。如果要是一笔钱的话，用户可能把钱用到了其他的场景，没法在做业务上的处理，那这样的话，就把一些没有隔离性的补偿方案排除掉了。那还Consistency有一致性我们是必须需要的，但我们要的是最终一致性，我不一定在某一个瞬间完全一致。那我们就认为一致性可以放宽。

那最后决定，在ACID四个属性中，我们放宽一致性的需求，同时我们也是在考虑到未来系统可伸缩的需要，我们一定不能用分布式的事务，标准的基于XA的分布式事务。在这两个基础上，我们就设计了一套支付宝分布式事务方案，在数据库操作这个层次我们建立transaction，我们的帐务处理是一种TCC的模式，任何一个帐务处理，你可以Try它，最后你可以Confirm它，也可以Cancel它。那在这个过程中，有TCC的基础支持的话，结合中央事务协调器，我们就可以做分布式服务，而且可以多层次的，任何一个服务，都可以做在事务里面。这个也会有成本，像设计一个支持TCC操作的业务服务。当然这方面，我们也探索出一些模式，可以做得比较好。

InfoQ：现在支付宝每天的交易量是非常惊人的，在这么大交易量的情况下，在网站的伸缩性设计上，有哪些考量？

我们觉得做SOA很大的好处就是他把很多复杂操作的伸缩性，简化成单个服务的伸缩性。比如对支付宝来说，我们有这么多的业务，对伸缩性最有挑战的，还是帐务处理和会员业务。比如帐务处理，那帐户处理就是相对来说比较单一的一个应用，那对这样的应用，我们有一些比较好的伸缩性的方案，那比如说应用处理，它对SOA来说是比较简单的，因为每个服务都是无状态的，是自治的，是可以很方便的通过水平扩展的方式去处理。其次是数据，那数据的话，取决于你的使用场景，如果你的场景支持数据很好的水平分割，那它的伸缩性也不会有什么问题，那从目前来看，因为我们把服务拆开了，对于每一种服务我们都可以选择最适合它的一种方案。比如会员数据，我们可以很方便的根据会员把他们分开。我们帐务处理其实也是一样的模式。那交易可能会复杂一些，但它也有它相应的模式，但我们并没有立刻去做这样的事情，因为目前来看，底层的基础还能够做相应的支持，支持我们现在事务的处理，但是我们相应的方案已经准备好了。如果当业务量达到去做分割的时候，我们就会去做。毕竟分拆之后，对运维，对成本都会有一些相应的要求。我么可以在适当的时候支付这样的成本。

InfoQ: 还有一个问题，其实也是大家都想问的问题。如果一个公司，它想在系统里面引入SOA的话，您作为一个在这方面的实施有非常丰富经验的人，能给他们提一些建议吗？

实施方面，关于SOA怎么实施，其实有很多指导。那我们支付宝实施SOA的话，前面也说到了，就是大概先去学习，对这个概念的认识，基本知识的掌握，获得所有人对走上这条路的共识，因为只有这个共识，大家才愿意投入这样的资源。因为SOA既有风险的，又要投入很多的资源，而且它不一定会马上看到效果，所以这肯定是第一步。

完成这一步之后呢，需要选择一个切入点，每家企业都有自己最痛/痒的地方，第一个投入这个项目的时候，要找准切入点，在找的过程中，要选择一个既不要太难，又不要太简单，然后做之后能产生效果的项目，因为太难的话，可能SOA这个项目就会做失败，那对SOA迁移就会产生非常大的打击。如果太简单的话，又不能起到锻炼技术、锻炼队伍、积累经验的目的。最后它要看到结果，如果看不到结果的话，那后续的资源投入都会遇到问题。

然后下一步，我感觉支付宝是比较大胆的，我们立刻拿我们最核心的业务去动刀了。这一块我觉得要动，早动会比晚动好。我们在2007年去动这块业务，当时我们敢动。现在如果动的话，我相信会有更多的挑战、更大的难度。

那整个基础打好之后，这个时候我想，包括你的基础设施，包括你的主要的SOA团队的建设，包括前面讲到基础，不光是技术基础，也是业务基础，包括整个一套的方法论，无论是已经正式成形的，还是大家口头认识上的，都应当有一定基础了。这个时候，可以让SOA交付它的价值了。但是我们支付宝，有一方面我们觉得做得还不够，还需要改进。就是治理引入太晚了，对于治理的问题，我们并没有非常早的意识到，一路的高歌猛进，但是后来发现，SOA会引入一些问题。使系统在某些方面变得更加复杂，更加难以控制等等这样一些问题，包括怎么让SOA真正交付价值，这些都需要治理来支持。

今天我才看到，有一篇应用SOA治理的报道，非常有意思，它里面就提到，如果你的系统服务个数达到50个以上，治理的话，不但要有，而且是正式的，得到了充分资源的保证的流程，实际上治理的话，不能说没有，一开始我们是一种人治的方式，通过这么多SOA系统建设的话，我们有一个很好团队，通过他们的口头的方式治理，但是随着整个服务体系全面铺开，靠人是很不靠谱的。我们希望如果有其他的企业想要走在SOA这条路的话，希望能够把治理尽量提前。

InfoQ: 其实对于企业来说，它从业务角度上来考虑SOA松耦合、多重复用等特性，您认为从架构的策略上面会有哪些原则可以遵循来实现这样的一个业务上的问题？

在架构方面的话，我想，首先SOA有一个很高层次的架构方面的指导，就是对一个企业来说，它的服务并不是所有服务都是遵循平等的，每个服务都有各自的安排。比如说对支付宝来说，我们有的服务是我们核心业务服务，有的是合作伙伴的服务，有的服务是我们产品的服务，有些服务是我们后端管理的服务。有了这么一个安排之后，要有规划的、有计划去做好每一个服务组，然后在这大的规划之后，才能有系统的去识别，在我这个业务里面我需要怎么样的服务。而不是，看到一个需求，就做一个服务。这样的话，整个系统就不成架构了。这块儿的话，支付宝是一直努力去做，加深大家的理解，让这个服务的识别能够更加匹配这样的架构。

InfoQ: 整个SOA改进的过程中，有没有遇到一些平台转型的问题？

平台转型的话，应该是感谢Java企业开发技术，越来越轻量化，侵入性越来越小了。无论是开源产品还是商业产品，它都能做到这样。所以平台转型方面，虽然我们有考虑，但是我们也没有在这方面做太多的工作，实际上我们大量SOA基础设施都是我们自己开

发的，因为考虑到我们只需要这样一些功能，我们不需要其他的功能。而且我们这些功能又有很多个性化的需求，所以我们倾向自己开发。

查看原文：[程立谈大规模SOA系统](#)