蚂蚁金服11.11：支付宝和蚂蚁花呗的技术架构及实践

贺岩 架构师 2016-01-20

每年“双11”都是一场电商盛会，消费者狂欢日。今年双11的意义尤为重大，它已经发展成为全世界电商和消费者都参与进来的盛宴。而对技术人员来说，双十一无疑已经成为一场大考，考量的角度是整体架构、基础中间件、运维工具、人员等。

一次成功的大促准备不光是针对活动本身对系统和架构做的优化措施，比如：流量控制，缓存策略，依赖管控，性能优化……更是与长时间的技术积累和打磨分不开。下面我将简单介绍支付宝的整体架构，让大家有个初步认识，然后会以本次在大促中大放异彩的“蚂蚁花呗”为例，大致介绍一个新业务是如何从头开始准备大促的。

因为涉及的内容要深入下去是足够写一个系列介绍，本文只能提纲挈领的让大家有个初步认识，后续可能会对大家感兴趣的专项内容进行深入分享。

架构

支付宝的架构设计上应该考虑到互联网金融业务的特殊性，比如要求更高的业务连续性，更好的高扩展性，更快速的支持新业务发展等特点。目前其架构如下：

整个平台被分成了三个层：

1. 运维平台（IAAS）：主要提供基础资源的可伸缩性，比如网络、存储、数据库、虚拟化、IDC等，保证底层系统平台的稳定性；

2. 技术平台（PAAS）：主要提供可伸缩、高可用的分布式事务处理和服务计算能力，能够做到弹性资源的分配和访问控制，提供一套基础的中间件运行环境，屏蔽底层资源的复杂性；
3. 业务平台（SAAS）：提供随时随地高可用的支付服务，并且提供一个安全易用的开放支付应用开发平台。

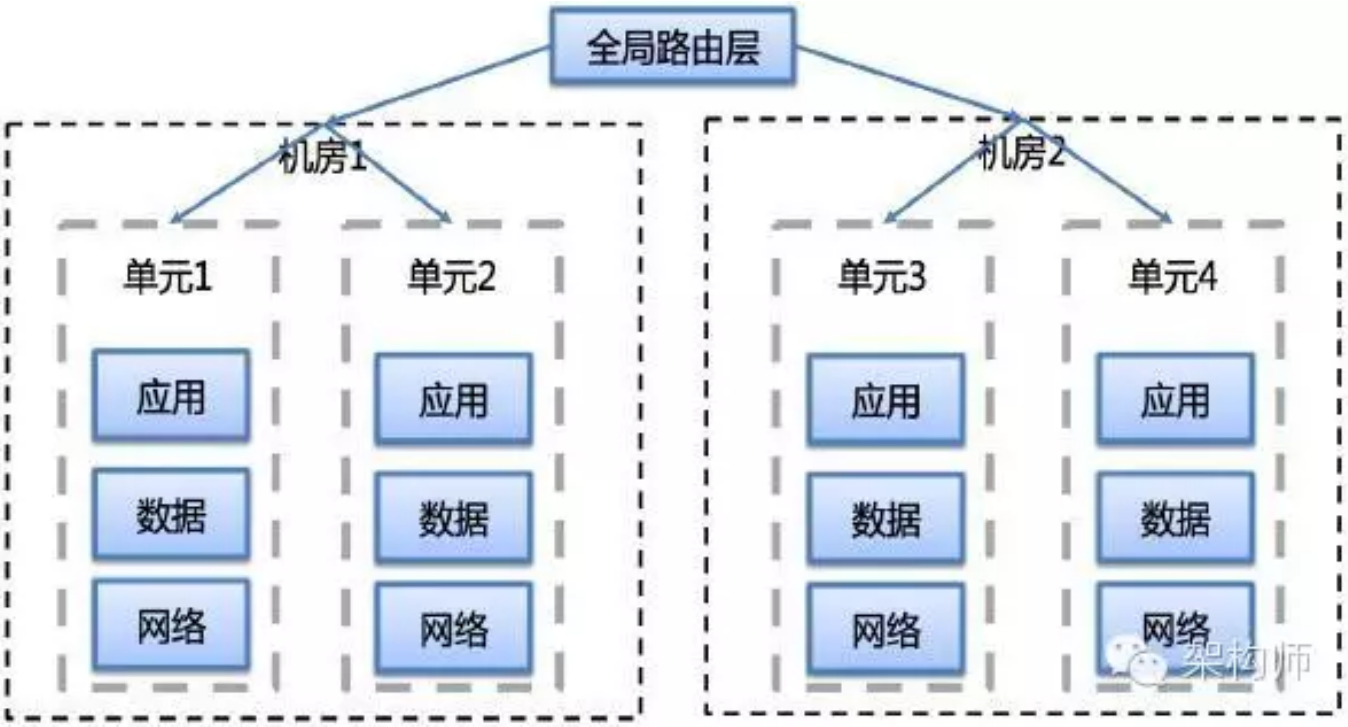
架构特性

逻辑数据中心架构

在双十一大促当天业务量年年翻番的情况下，支付宝面临的考验也越来越大：系统的容量越来越大，服务器、网络、数据库、机房都随之扩展，这带来了一些比较大的问题，比如系统规模越来越大，系统的复杂度越来越高，以前按照点的伸缩性架构无法满足要求，需要我们有一套整体性的可伸缩方案，可以按照一个单元的维度进行扩展。能够提供支持异地伸缩的能力，提供N+1的灾备方案，提供整体性的故障恢复体系。基于以上几个需求，我们提出了逻辑数据中心架构，核心思想是把数据水平拆分的思路向上层提到接入层、终端， 从接入层开始把系统分成多个单元，单元有几个特性：

1. 每个单元对外是封闭的，包括系统间交换各类存储的访问;
2. 每个单元的实时数据是独立的，不共享。而会员或配置类对延时性要求不高的数据可共享;
3. 单元之间的通信统一管控，尽量走异步化消息。同步消息走单元代理方案；

下面是支付宝逻辑机房架构的概念图：



这套架构解决了几个关键问题：

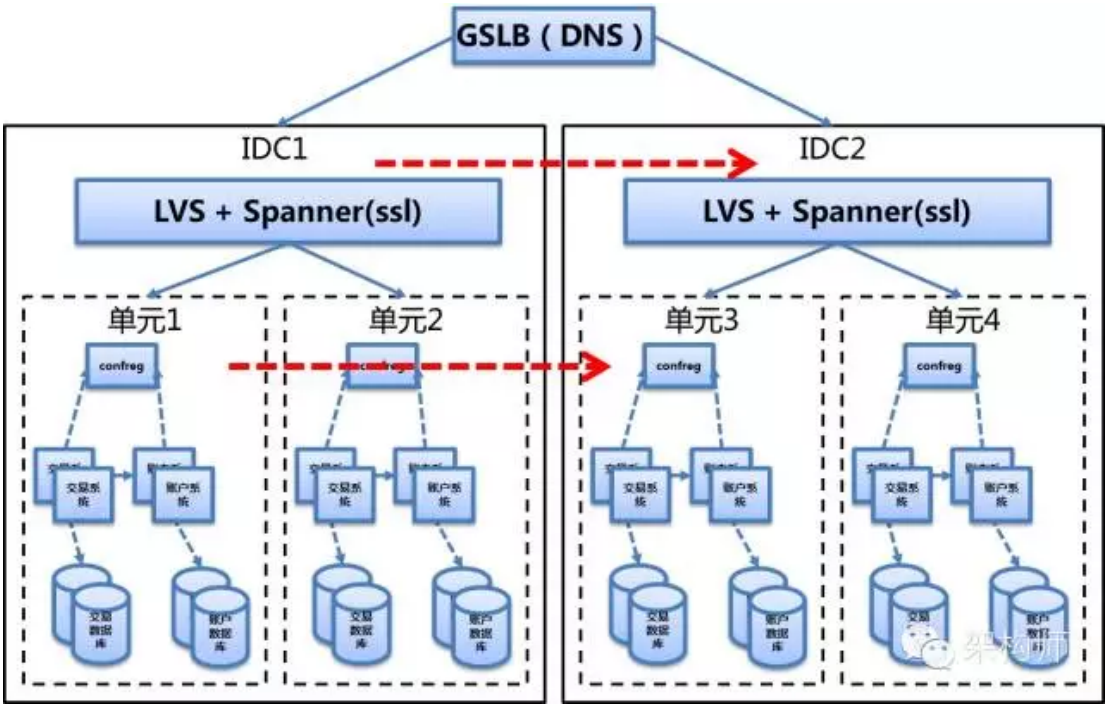
1. 由于尽量减少了跨单元交互和使用异步化，使得异地部署成为可能。整个系统的水平可伸缩性大大提高，不再依赖同城IDC；
2. 可以实现N+1的异地灾备策略，大大缩减灾备成本，同时确保灾备设施真实可用；

3. 整个系统已无单点存在，大大提升了整体的高可用性；同城和异地部署的多个单元可用作互备的容灾设施，通过运维管控平台进行快速切换，有机会实现100%的持续可用率；
4. 该架构下业务级别的流量入口和出口形成了统一的可管控、可路由的控制点，整体系统的可管控能力得到很大提升。基于该架构，线上压测、流量管控、灰度发布等以前难以实现的运维管控模式，现在能够十分轻松地实现。

目前新架构的同城主体框架在2013年已经完成，并且顺利的面对了双十一的考验，让整套架构的落地工作得到了很好的证明。

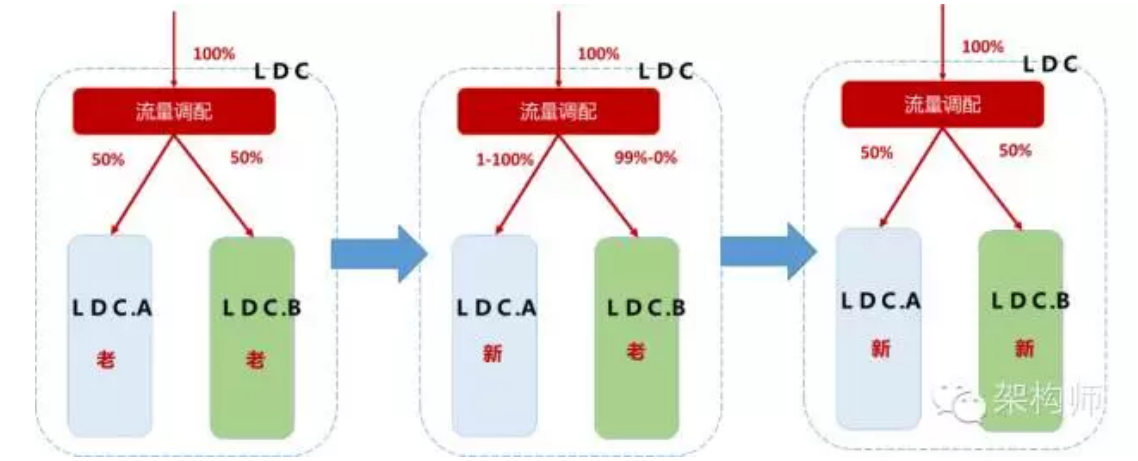
在2015年完成了基于逻辑机房，异地部署的“异地多活”的架构落地。“异地多活”架构是指，基于逻辑机房扩展能力，在不同的地域IDC部署逻辑机房，并且每个逻辑机房都是“活”的，真正承接线上业务，在发生故障的时候可以快速进行逻辑机房之间的快速切换。这比传统的“两地三中心”架构有更好的业务连续性保障。在“异地多活”的架构下，一个IDC对应的故障容灾IDC是一个“活”的IDC，平时就承接着正常线上业务，保证其稳定性和业务的正确性是一直被确保的。

以下是支付宝“异地多活”架构示意图：



除了更好的故障应急能力之外，基于逻辑机房我们又具备的“蓝绿发布”或者说“灰度发布”的验证能力。我们把单个逻辑机房（后续简称LDC）内部又分成A、B两个逻辑机房，A、B机房在功能上完全对等。日常情况下，调用请求按照对等概率随机路由到A或B 。当开启蓝绿模式时，上层路由组件会调整路由计算策略，隔离A与B之间的调用， A组内应用只能相互访问，而不会访问B组。

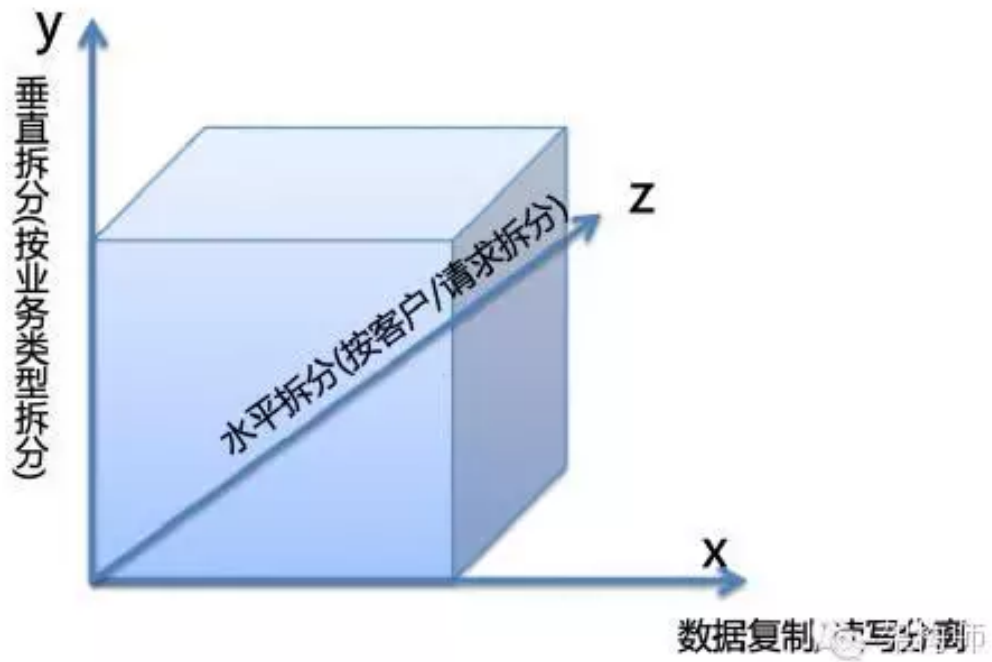
然后进行蓝绿发布流程大致如下：



Step1. 发布前，将“蓝”流量调至0%，对“蓝”的所有应用整体无序分2组发布。
Step2. “蓝”引流1%观察，如无异常，逐步上调分流比例至100%。
Step3. “绿”流量为0%，对“绿”所有应用整体无序分2组发布。
Step4. 恢复日常运行状态，蓝、绿单元各承担线上50%的业务流量。

分布式数据架构

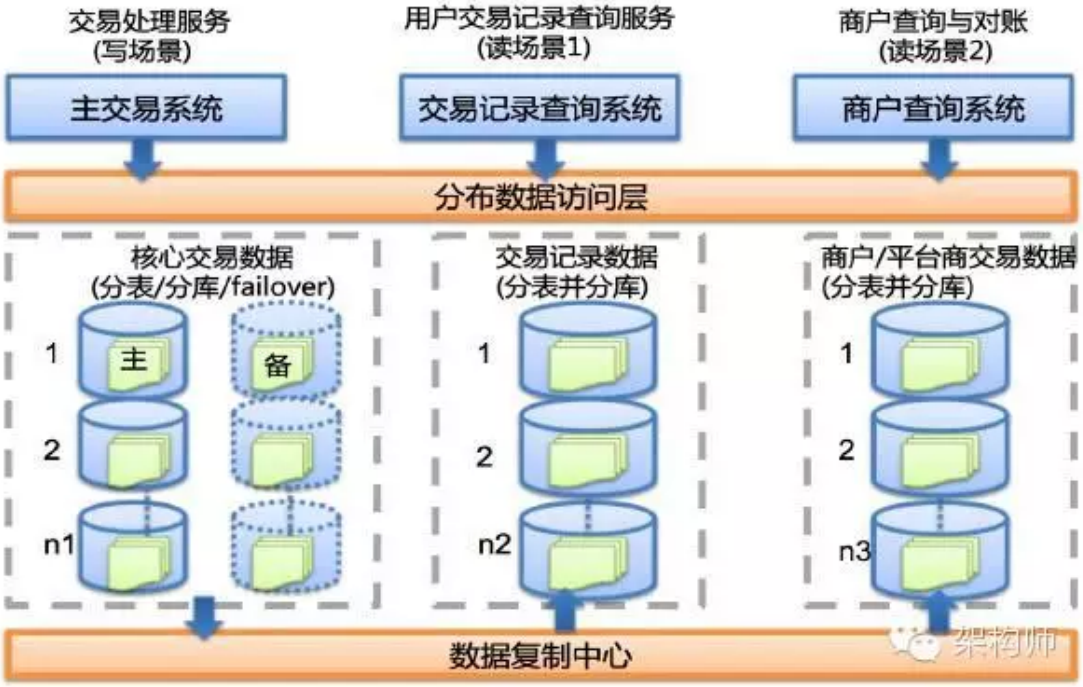
支付宝在2015年双十一当天的高峰期间处理支付峰值8.59万笔/秒，已经是国际第一大系统支付。支付宝已经是全球最大的OLTP处理者之一，对事务的敏感使支付宝的数据架构有别于其他的互联网公司，却继承了互联网公司特有的巨大用户量，最主要的是支付宝对交易的成本比传统金融公司更敏感，所以支付宝数据架构发展，就是一部低成本、线性可伸缩、分布式的数据架构演变史。
现在支付宝的数据架构已经从集中式的小型机和高端存储升级到了分布式PC服务解决方案，整体数据架构的解决方案尽量做到无厂商依赖，并且标准化。
支付宝分布式数据架构可伸缩策略主要分为三个维度：



1. 按照业务类型进行垂直拆分
2. 按照客户请求进行水平拆分（也就是常说的数据的sharding策略）

3. 对于读远远大于写的数据进行读写分离和数据复制处理

下图是支付宝内部交易数据的可伸缩性设计：



交易系统的数据主要分为三个大数据库集群：

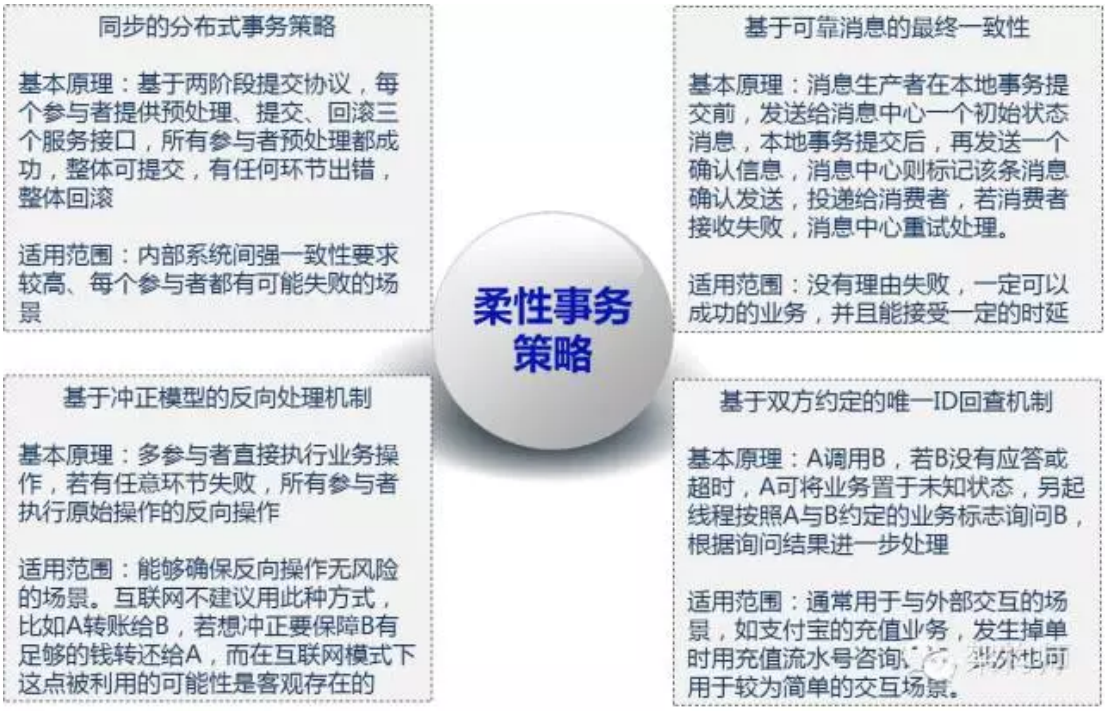
1. 主交易数据库集群，每一笔交易创建和状态的修改首先在这里完成，产生的变更再通过可靠数据复制中心复制到其他两个数据库集群：消费记录数据库集群、商户查询数据库集群。该数据库集群的数据被水平拆分成多份，为了同时保证可伸缩性和高可靠性，每一个节点都会有与之对应的备用节点和failover节点，在出现故障的时候可以在秒级内切换到failover节点。
2. 消费记录数据库集群，提供消费者更好的用户体验和需求；
3. 商户查询数据库集群，提供商户更好的用户体验和需求；

对于分拆出来的各个数据节点，为了保证对上层应用系统的透明，我们研发一套数据中间产品来保证交易数据做到弹性扩容。

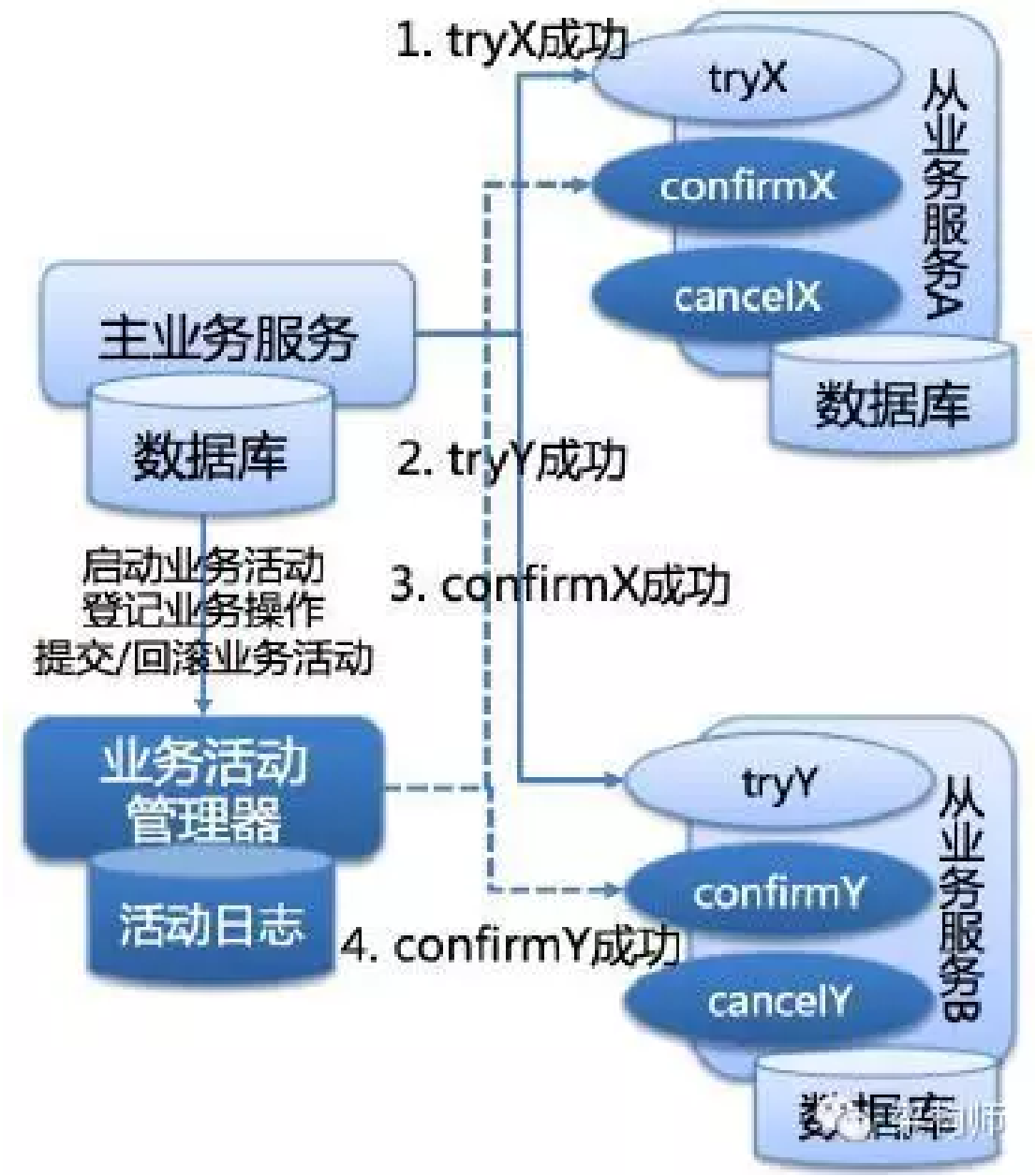
数据的可靠性

分布式数据架构下，在保证事务原有的ACID（原子性、一致性、隔离性、持久性）特性的基础上，还要保证高可用和可伸缩性，挑战非常大。试想你同时支付了两笔资金，这两笔资金的事务如果在分布式环境下相互影响，在其中一笔交易资金回滚的情况下，还会影响另外一笔是多么不能接受的情况。

根据CAP和BASE原则，再结合支付宝系统的特点，我们设计了一套基于服务层面的分布式事务框架，他支持两阶段提交协议，但是做了很多的优化，在保证事务的ACID原则的前提下，确保事务的最终一致性 。我们叫做“柔性事物”策略。原理如下：



以下是分布式事务框架的流程图：



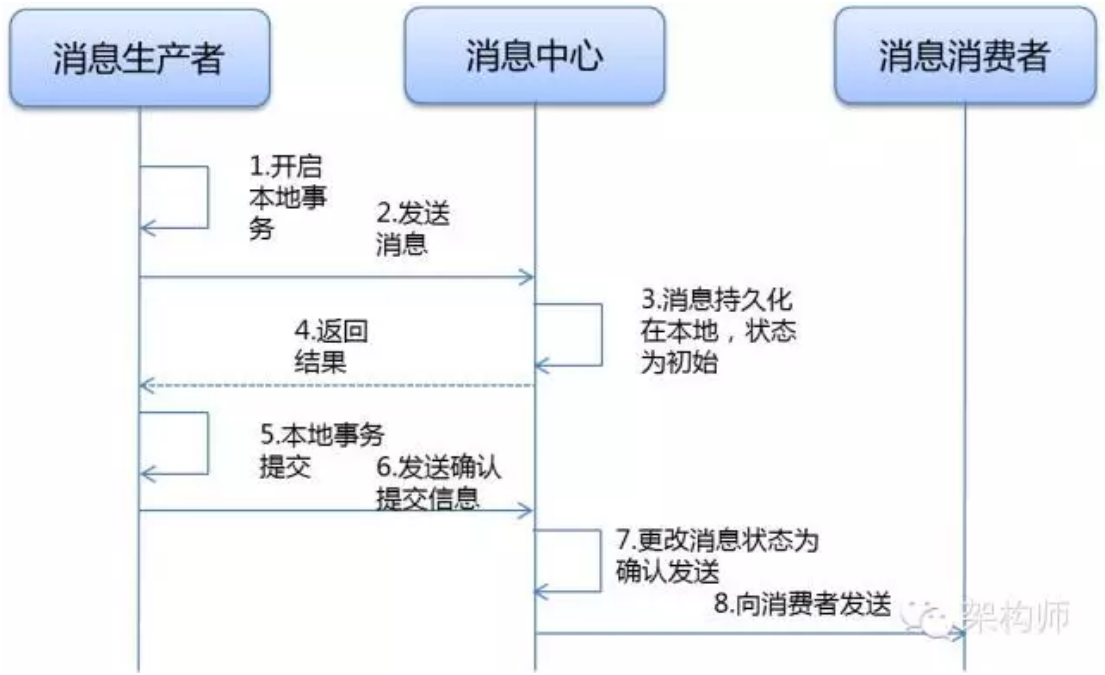
实现：

1. 一个完整的业务活动由一个主业务服务与若干从业务服务组成。
2. 主业务服务负责发起并完成整个业务活动。
3. 从业务服务提供TCC型业务操作。
4. 业务活动管理器控制业务活动的一致性，它登记业务活动中的操作，并在活动提交时确认所有的两阶段事务的confirm操作，在业务活动取消时调用所有两阶段事务的cancel操作。”

与2PC协议比较：

1. 没有单独的Prepare阶段，降低协议成本
2. 系统故障容忍度高，恢复简单

其中关键组件异步可靠消息策略如下：



其中一些关键设计点：

1. 若在第2、3、4步出现故障，业务系统自行决定回滚还是另起补偿机制；若在第6、7步出现异常，消息中心需要回查生产者；若在第8步出现异常，消息中心需要重试。第6步的确认消息由消息中心组件封装，应用系统无需感知。
2. 此套机制保障了消息数据的完整性，进而保障了与通过异步可靠消息通讯的系统数据最终一致性。
3. 某些业务的前置检查，需要消息中心提供指定条件回查机制。

蚂蚁花呗

蚂蚁花呗是今年增加的一个新支付工具，“确认收货后、下月还”的支付体验受到了越来越多的消费者信赖。跟余额和余额宝一样，蚂蚁花呗避开了银行间的交易链路，最大限度避免支付时的拥堵。据官方数据披露，在今天的双十一大促中，蚂蚁花呗支付成功率达到99.99%、平均每笔支付耗时0.035秒，和各大银行渠道一起确保了支付的顺畅。

蚂蚁花呗距今发展不到一年，但发展速度非常快。从上线初期的10笔/秒的支付量发展到双十一当天峰值2.1w笔/秒。支撑蚂蚁花呗业务发展的技术体系经过不断演进、已经完全

依托于蚂蚁金服的金融云架构。

在2014年12月，蚂蚁花呗团队完成业务系统优化，按照标准将系统架设到了金融云上，依次对接了渠道层、业务层、核心平台层、数据层，使得用户对蚂蚁花呗在营销、下单和支付整个过程中体验统一。

2015年4月，蚂蚁花呗系统同步金融云的单元化的建设，即LDC，使得数据和应用走向异地成为了现实，具备了较好的扩展性和流量管控能力。在可用性方面，与金融云账务体系深度结合，借用账务系统的failover能力，使得蚂蚁花呗通过低成本改造就具备了同城灾备、异地灾备等高可用能力。任何一个单元的数据库出了问题、能够快速进行容灾切换、不会影响这个单元的用户进行蚂蚁花呗支付。在稳定性方面，借助于云客户平台的高稳定性的能力，将蚂蚁花呗客户签约形成的合约数据迁移进去，并预先写入云客户平台的缓存中，在大促高峰期缓存的命中率达到100%。同时，结合全链路压测平台，对蚂蚁花呗进行了能力摸高和持续的稳定性测试，发现系统的性能点反复进行优化，使得大促当天系统平稳运行。在之前的架构中，系统的秒级处理能力无法有效衡量，通过简单的引流压测无法得到更加准确、可信的数据。立足于金融云，系统很快通过全链路压测得到了每秒处理4w笔支付的稳定能力。

蚂蚁花呗业务中最为关键的一环在于买家授信和支付风险的控制。从买家下单的那一刻开始，后台便开始对虚假交易、限额限次、套现、支用风险等风险模型进行并行计算，这些模型最终将在20ms以内完成对仅百亿数据的计算和判定，能够在用户到达收银台前确定这笔交易是否存在潜在风险。

为了保证蚂蚁花呗双11期间的授信资金充足，在金融云体系下搭建了机构资产中心，对接支付清算平台，将表内的信贷资产打包形成一个一定期限的资产池，并以这个资产池为基础，发行可交易证券进行融资，即通过资产转让的方式获得充足资金，通过这一创新确保了用户能够通过花呗服务顺利完成交易，并分流对银行渠道的压力。通过资产证券化运作，不仅帮助100多万小微企业实现融资，也支撑了蚂蚁花呗用户的消费信贷需求。蚂蚁小贷的资产证券化业务平台可达到每小时过亿笔、总规模数十亿元级别的资产转让。

总结

经过这么多年的高可用架构和大促的准备工作，蚂蚁金融技术团队可以做到“先胜而后求战”，主要分为三方面技术积累：“谋”，“器”，“将”。

“谋”就是整体的架构设计方案和策略；

“器”就是支持技术工作的各种基础中间件和基础组件；

“将”就是通过实践锻炼成长起来的技术人员。

纵观现在各种架构分享，大家喜欢谈“谋”的方面较多，各种架构设计方案优化策略分享，但实际最后多是两种情况：“吹的牛X根本没被证实过”（各种框架能力根本没经过实际考验，只是一纸空谈），“吹过的牛X一经实际考验就破了”（说的设计理念很好，但是一遇到实际的大业务的冲击系统就挂了），最后能成功的少之又少。这些说明虽然架构上的“心灵鸡汤”和“成功学”技术人员都已经熟的不行，但是发现一到实践其实根本不是那么回事。从此可以看出，其实最后起决定作用的不是 “谋”方面的理论层面的分析设计，最重要的是落地“器”和“将”的层面。有过硬高稳定性的各种基础设施工具的和身经百战被“虐

了千百次”的技术人员的支撑才是最后取胜的关键。而这个两个层面的问题是不能通过分享学到的，是要通过日积月累的，无数流血流泪趟雷中招锻炼出来的，没有近路可抄。

而目前从业务和市场的发展形势来看，往往就是需要技术在某个特定时间有个质的能力的提升和飞跃，不会给你太多的准备技术架构提升的时间，在技术积累和人员储备都不足的时候，如何构建平台能力，把更多的精力放在业务相关的开发任务中，是每个技术团队的希望得到的能力 。

过去我们是通过某个开源或者商业组件来实现技术共享得到快速解决谋发展技术的能力的，但是随着业务复杂性，专业性，规模的逐步变大，这种方式的缺点也是显而易见的：1、很多组件根本无法满足大并发场景下的各种技术指标；2、随着业务的复杂和专业性的提高，没有可以直接使用的开源组件；3、“人”本身的经验和能力是无法传递的。

所以现在我们通过“云”分享的技术和业务的能力的方式也发展的越来越快，这就我们刚才介绍的“蚂蚁花呗”技术用几个月的时间快速的成功的达到“从上线初期的10笔/秒的支付量发展到双十一当天峰值2.1w笔/秒，快速走完了别人走了几年都可能达不到的能力。类似的例子还有大家熟知的“余额宝”系统。

这些都是建立在原来蚂蚁金服用了10年打磨的基础组件和技术人员经验的云服务上的，通过目前基于这种能力，我们目前可以快速给内部和外部的客户组建，高可用、安全、高效、合规的金融云服务架构下的系统。

来源：InfoQ

原文：http://www.infoq.com/cn/articles/technical-architecture-of-alipay-and-ant-check-later

·END·

架构师

我们都是架构师！