阿里Aliware十年微服务架构演进历程中的挑战与实践

2016-12-20 11:52:13 VIP006 BAT架构顾问

如今的阿里巴巴电商平台上，业务生态百花齐放，新的创新业务不断涌现，而这都得益于阿里底层的微服务架构高可扩展。而谁能想到，早在10年以前，偌大的淘宝网站点都是运行在单一的部署包内，往往对其中一个模块的改动都会牵一发而动全身。

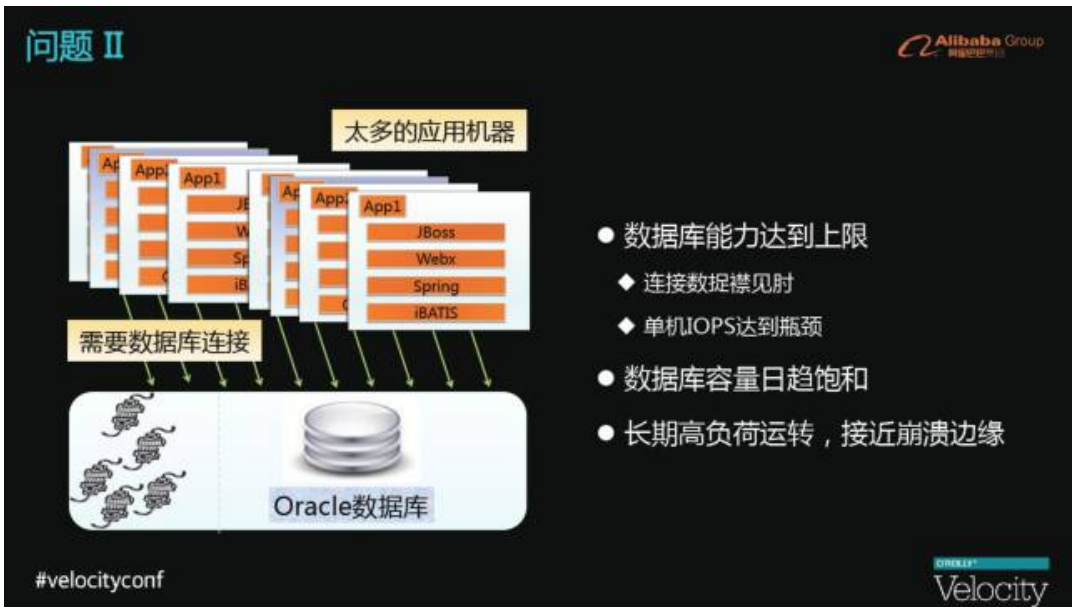
自从2007年以来，在这近10年时间里，阿里巴巴技术团队一直在微服务的道路上摸索前进着，其间伴随着互联网和移动互联网的盛行，海量的用户一次又一次的洗礼了各个机构的IT系统。

而在阿里，这种改变无疑更加频繁与剧烈——这些年下来，阿里中间件技术完成了从1.0到3.0时代的蜕变，并已经完成了将技术变成商业化产品，对于海量微服务的治理能力处于业界领先，形成了自主技术产品和品牌——Aliware。

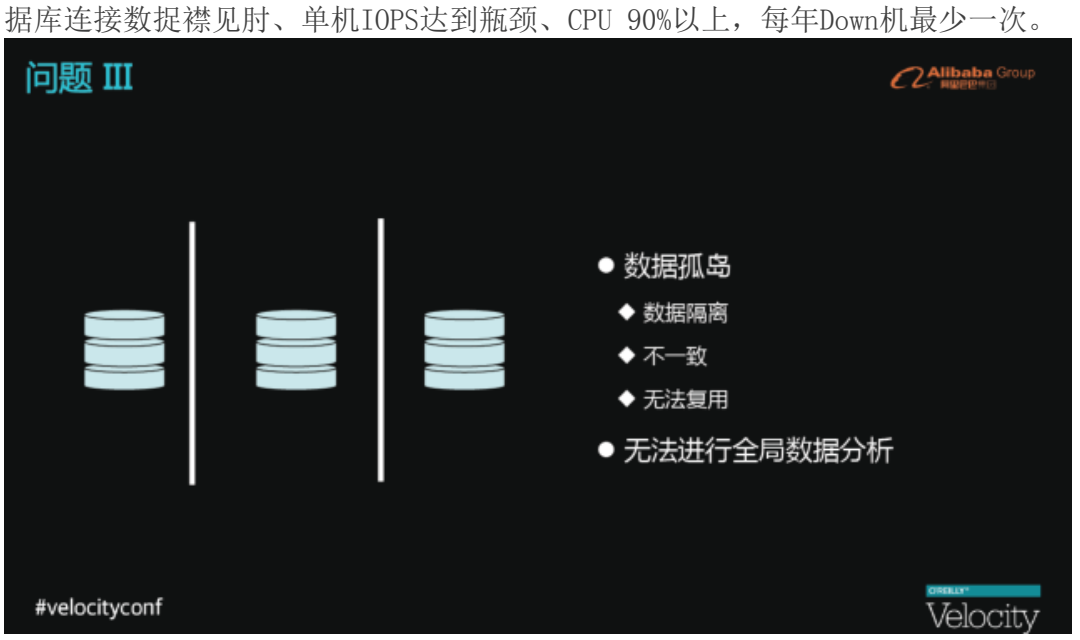
服务化缘起

在2007年的时候，阿里技术团队规模大概是500人左右，当时的主要业务站点淘宝网，都在一个单一的WAR包进行部署，基于传统Java EE应用开发架构，使用的是Oracle数据库和JBoss服务器——而与此同时，淘宝网的业务每年翻倍增长。

在那个阶段，我们面临着非常多的挑战。第一个挑战是系统的研发成本非常高。上百人维护一个核心工程会碰到很多问题，包括源代码冲突严重、协同成本非常高等；同时，项目发布周期太长；所有的逻辑都是耦合的，错误难以隔离，对淘宝网整个工程里的某个模块、某个系统功能进行一些改动时，整个系统都会面临非常大的技术风险。



第二个挑战是数据库能力达到上限。淘宝早期用Oracle数据库，单机的Oracle数据库连接数捉襟见肘、单机IOPS达到瓶颈、CPU 90%以上，每年Down机最少一次。

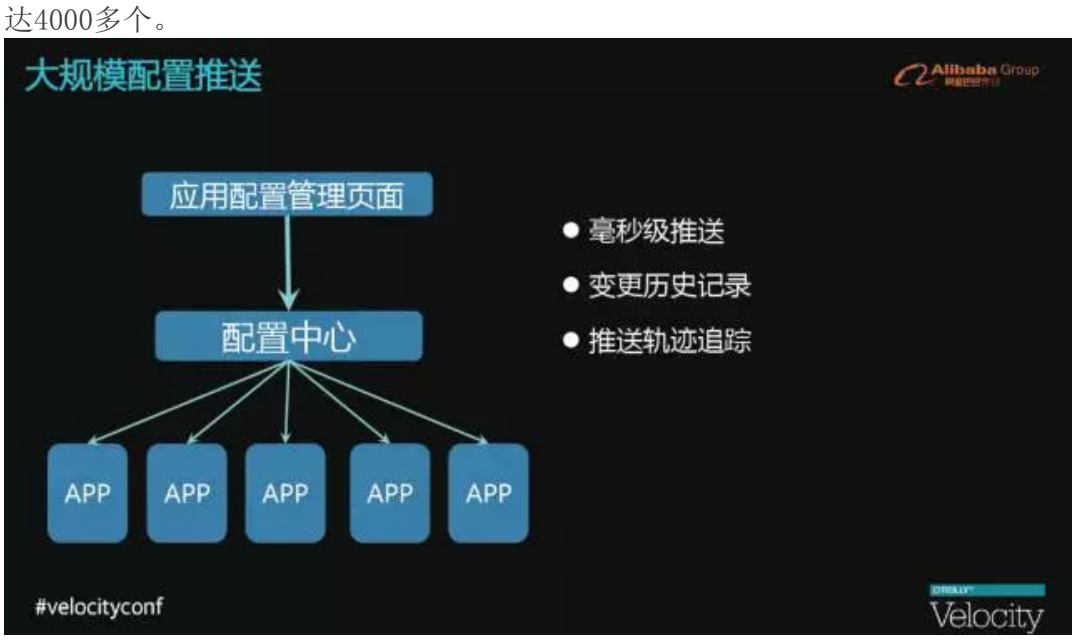


第三个挑战是数据孤岛。数据隔离，重复建设，数据不一致；无法进行大数据分析。

微服务架构的形成

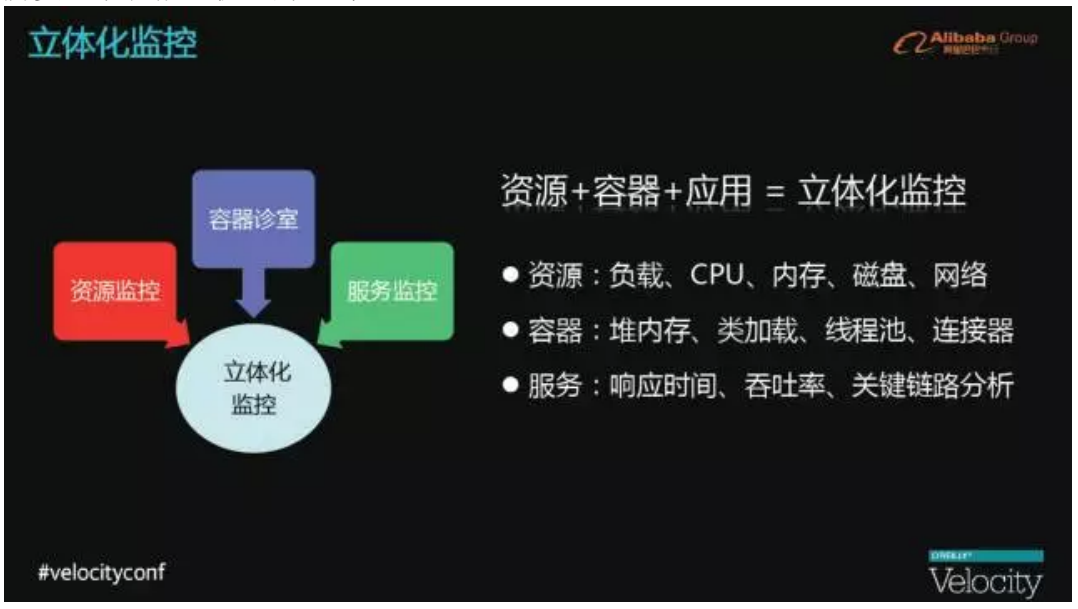
整个微服务架构落实到技术层面，就是把原本集中式的模块分散到分布式里不同的机制上运行，并且希望有这样一个框架能够将不同机器、不同机房、不同模块之间的服务化调用能够顺畅的构建起来，能够帮助组织服务发布、服务注册以及服务发现等过程。

目前阿里生产环境使用的是第三代RPC框架：EDAS-HSF，在整个平台90%以上应用当中使用，历经8次双11大促大流量和高并发考验，支持分布式事务。而我们也将第一代RPC框架EDAS-Dubbo开源，目前已经成为国内最活跃开源软件之一、开源分支达4000多个。



在进行服务化拆分之后，需要将每一个服务使用的配置进行集中式管理。因此，我们研发了可靠的配置推送服务，能够在毫秒级时间内完成配置推送，同时支持变更

历史记录和推送轨迹的查询。

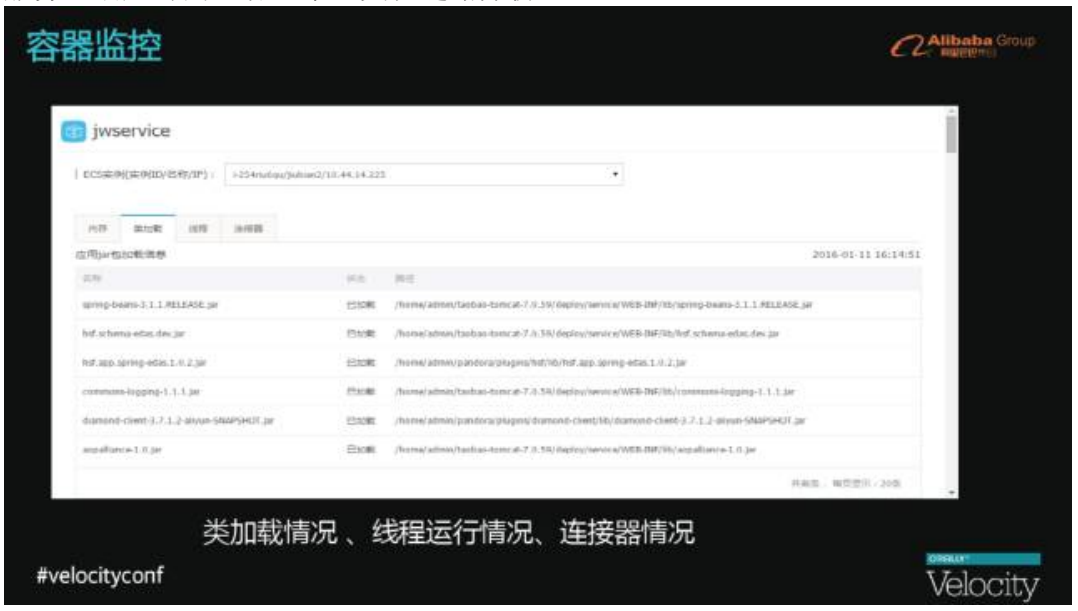


监控是我们非常关注的事情，对于系统整体的性能指标也非常重要，所以，我们会尝试从不同层面收集信息，实现对应用立体化的监控，包括资源、容器和应用，具体包括以下三大方面：

系统资源：负载，CPU、内存、磁盘、网络

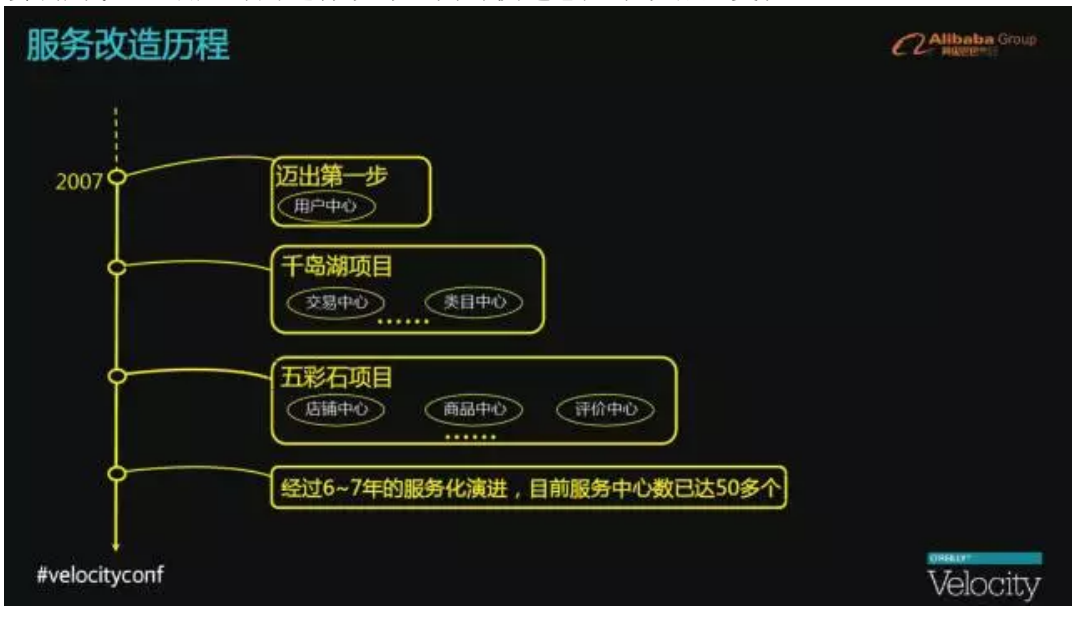
容器：堆内存、类加载、线程池、连接器

服务：响应时间、吞吐率、关键链路分析



阿里巴巴目前是架构在Java平台上，作为包含Java运行环境的Java容器，是监控的重点，我们会监控堆内存与非堆内存使用情况、线程运行情况（提前将线程情况全部显示出来）、连接器情况，类加载情况尤复杂，很多时候一个类进行初始化时，层层依赖其它类，对此，我们在应用启动时跟踪加载的类。

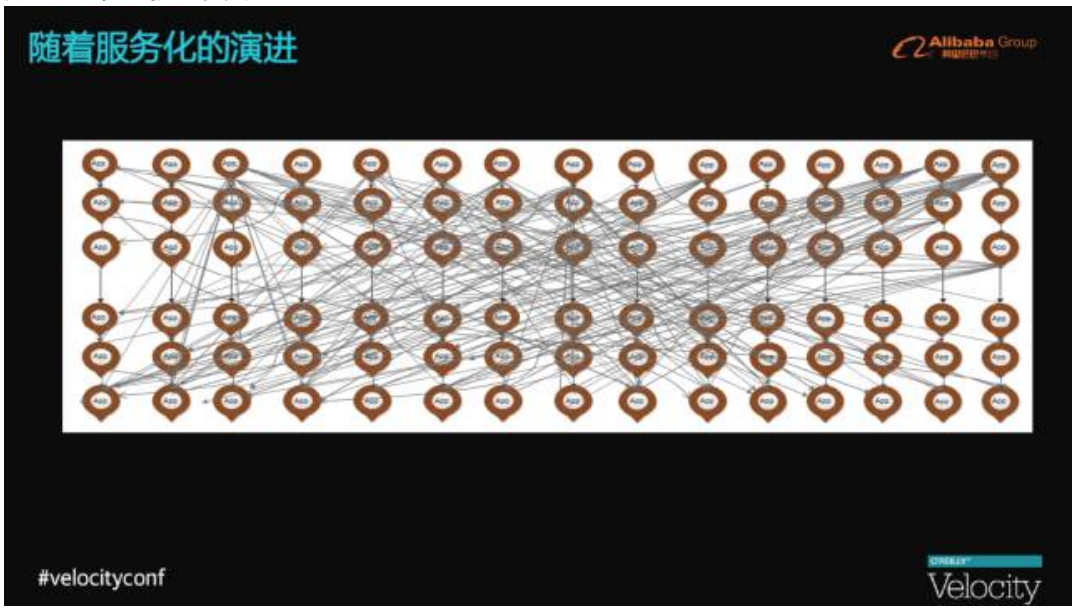
当原本在集中式的系统架构里面，每个页面会贯穿非常多的模块，每个模块都耦合在一个系统中，最终监控出的是表象，无法知道页面打开慢是哪个模块哪个功能逻辑上慢。现在，我们会对每一个服务接口、方法的实时调用情况进行监控，我们还会调用QPS、响应时间进行统计，同时快速感知系统流量变化。



淘宝网围绕EDAS技术体系进行了一整套的服务化改造，在这个改造过程中，我们首先将数据复用度最高的数据进行拆分，剥离出用户中心这样的共享型的服务层，对上层所有业务进行用户相关的所有逻辑，接下来又陆续有千岛湖项目、五彩石项目，这些项目的背后都是一系列的服务化中心拆分出来的产物，后来经过6～7年的服务化演进，目前服务中心数已达50多个。

图为阿里巴巴核心服务化架构。自主创新走出技术困境，沉淀一大批成熟中间件技术，最底层为共享型中间件和组件，以及阿里云沉淀下来的技术支撑型产品；共享服务体系打破应用“烟囱式”建设方式，支撑业务快速创新；云化基础架构高效支撑业务增长，灵活的弹性伸缩带来巨大的成本节约。

海量微服务的挑战和实践

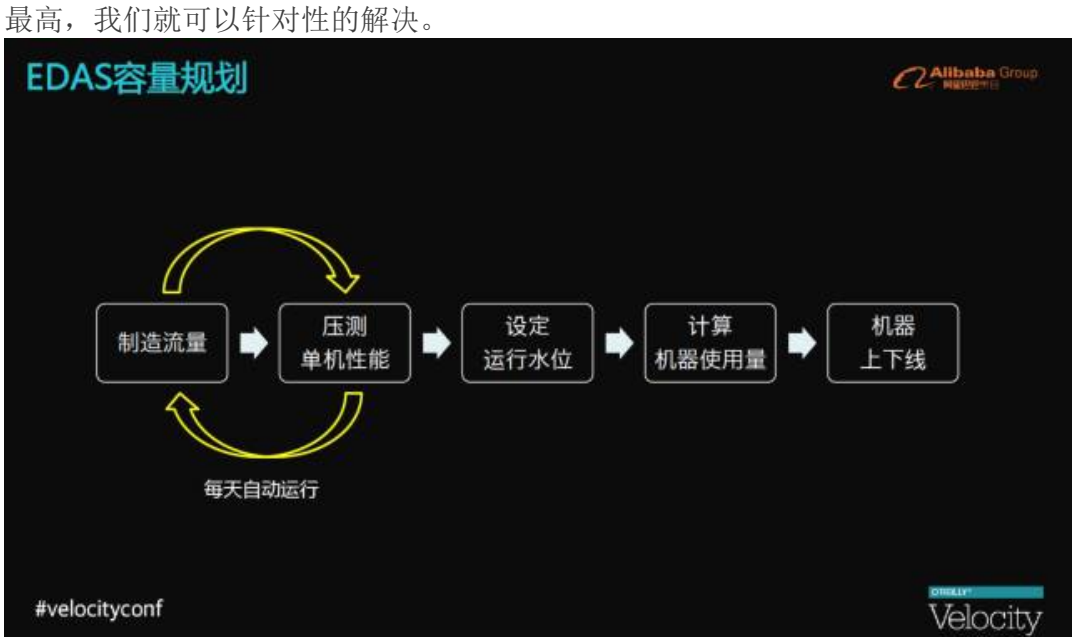


随着服务化的拆分，所有的系统会变得越来越多，箭头指向就是底层的服务化中心，上层调用过来就是前端的业务系统。很多系统调用很多的服务中心，这时已经没有能够人为的架构师帮助我们进行服务依赖和架构梳理。

于是，阿里内部进行了EDAS鹰眼的研发。图中从上至下，包括从页面打开到页面完整响应所经历的分布式各层系统调用。阿里内部就是依靠一整套的链路跟踪系统，能够在系统出现打开失败时，可以非常清晰的故障根源在哪。

当我们把类似的请求调用链路全部汇总起来进行分析后，就可以在很短时间内进行数据采集，并且有数据化的运营出来。峰值的QPS是指今天在某一个业务高峰时某一个业务的服务在分钟级别的服务化的调用过程中达到的最大的QPS，如图中标记可以看出，即使页面暴露在最前端，但不一定是压力最大的，这就算数据可视化带给我们的价值所在。我们还要对数据进行决策上的帮助，数据最大的价值在于可以精准化的通知我们最大压力点。

某个页面打开经过一系列的系统调用时，总会在某一个点出现问题，称之为易故障点。我们可以直观的看到在过去的一天里，到底所有的请求在哪一个组件的出错率最高，我们就可以针对性的解决。



在过去的6～7年时间，沉淀了一整套的容量规划模型。首先我们希望在第一步将线上真实流量进行引流，通过真实流量压测部分单机性能，然后根据设定的运行水位计算系统承载的最高容量，从而到最后可以实现机器按需的上线和下线，把这些系统融会贯通在一起，就是整体的容量规划提供的功能。所有的压测在单机上都会定一些指标,当我们进行集群中把一半机器流量全部引到另一半时候，所有流量的QPS就会翻倍，当单机性能如果没有达到运行水位时，就会继续引流，直到达到指标为止。

为了在大促时保证系统更稳定的运行，采用了限流和降级的手段。根据不同服务的优先级，不同服务的重要程度来执行限流和降级的措施。限流降级是阿里最有特色的功能之一，我们会面对非常强大的挑战就是双十一网购狂欢节，我们需要在成本和体验中选择一个好的平衡点，要利用这个平衡点我们必须要保证系统的可用性，不能因为用户多导致系统无法服务，就像排队买票一样，我们需要对自己的系统进行优化，具体表现在以下两方面：

限流：针对非核心服务调用者

降级：针对系统的非核心服务依赖

本文转载自微信公众号阿里技术（ali_tech），已经获得作者授权。

作者介绍

倪超，阿里花名银时，阿里巴巴企业互联网架构平台产品经理、国家认证系统分析师、IT畅销书作者，著有《从Paxos到ZooKeeper》一书，2015年国内新书畅销榜Top10。2010年，以实习生身份加入阿里，入职中间件技术团队，经历了阿里中间件技术从1.0到3.0的变革，目前负责商用软件EDAS。

阅读 2 0 举报

打赏

评论