全面异步化：淘宝反应式架构升级探索

作者：徐川

 2019 年 1 月 15 日

2018 年初，淘宝开始尝试对整体架构进行升级，经过近一年的探索，实现了全面异步化，这一架构升级在部分应用中取得了 40% 以上的性能提升，同时也为后续的回压推进打下了基础。负责该项架构升级的是淘宝技术专家许泽彬，他在 2018 领域驱动设计中国峰会上做了《淘宝应用架构升级——反应式架构的探索与实践》的分享，InfoQ 也趁此机会对他进行了采访，了解了更多细节。

2018 领域驱动设计中国峰会 (DDD2018) 是 ThoughtWorks 发起的技术会议，旨在给国内的 DDD 实践者们提供一个互相交流、分享自己团队的成功经验的机会的平台。

许泽彬是淘宝技术专家，目前负责淘宝应用架构升级。曾参与淘宝用户增长设施与平台建设，负责过分布式调用链跟踪框架和系统"鹰眼"，曾经是阿里分布式数据库中美异地机房数据同步的核心开发，也是阿里旗下开源项目 otter 和 canal 的核心开发者。

淘宝此次架构升级，重点在于将同步的架构改为异步、面向流的开发，以便为后续的回压方案打下基础， 这里所说的回压方案要解决的问题是：应用在突发流量下的稳定性保证，以及最大化提升资源利用率。 许泽彬将其称为反应式架构。

经过近一年的推进，反应式架构已经在生产环境落地，在 2018 年双 11 万亿级大规模处理量下，架构升级对多个核心应用进行了验证落地，取得了超出预期的效果：

- 其中『猜你喜欢』应用上限 QPS 提升 96%，即只需一半的机器就能支撑现有业务；
- 而另一核心应用『我的淘宝』实际线上响应时间下降了 40% 以上，意味着用户可以更快得到响应。

在淘宝内部，这仅仅是回压——突发流量下的稳定性以及最大化提升资源利用率方案 —— 刚刚迈出的第一步。

泽彬认为，这里将其称为反应式更为准确，响应式更多用于前端的界面中，对应的英文是 Responsive。

反应式架构与一般架构相比，其反应体现在：

第一个，对用户有反应，对用户有反应我们才说响应，一般我们说的响应，基本上都说得针对跟用户来交互。

第二个，要对失败有反应，应用失败了系统不能无动于衷，等着它挂掉，要有反应。

第三个，要对容量和压力变化有所反应，比如说淘宝的秒杀，系统需要反应来保证对用户的响应性，再如那个当流量降下来，将系统缩容，可以节约成本，这也是一种反应。

第四个，对输入有反应，响应系统的输入，也可以叫做消息驱动。

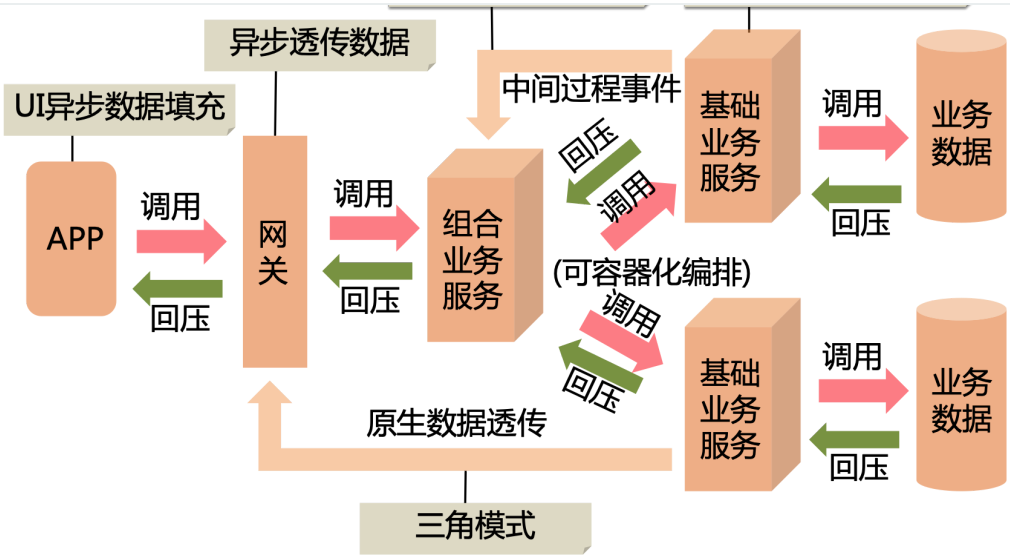
要做到反应式，需要做到三点：

- 适应性，也就是发生失败能恢复回来，无论是系统、网络、代码出现了问题都能恢复。
- 弹性，这点主要是应对流量的变化，弹性的前提是做到可伸缩性 Scalability，从软件设计上，要做到去中心化；同时，在运行时，要感知节点当前的系统负载，将压力往上游进行反馈，做到系统可以感知链路级别的节点压力，使得系统可以针对整体压力进行有目的地扩容缩容。这样才能够做到真正的弹性，根据系统负载进行扩容或缩容，这也是淘宝的回压方案在后续所需要支撑的场景。
- 消息驱动，有了消息驱动才能比较好的做到上面两个点。在反应式架构里，以前这点叫做事件驱动，后来改为消息驱动，消息驱动强调无阻塞、无 callback，所以不会有线程挂在那里，不会有持续的资源消耗。同时，事件驱动或消息驱动都是异步化，而异步化会将操作系统中的队列情况显式地提升到了应用层，使得应用层可以显式根据队列的情况来进行压力负载的感知，这对于淘宝后续的回压方案非常重要，而要做到这点，就需要异步了。

反应式架构中的核心概念是“流”，流就是面向数据的顺序串行执行的一系列操作组合，它同传统的编程相比，将业务逻辑导致数据改变，变成了操作改变数据，反过来影响业务逻辑的改变。面向流编程就是面向数据编程。

面向流的开发的优势主要体现在：

- 提供大量强大的操作符，包括创建、过滤、转换等。声明式表达比过程式编程更加完备、高级、快捷。
- 在并发控制方面，不同的流之间无依赖，通过切换 Scheduler 就可以自动多流并发，而业务按照语义编写，可以更友好的并发控制，更优的维护性和性能。
- 更高的资源利用率。通过更少的上下文切换、更低的竞争，可以降低负载，提升资源的有效利用率。
- 流可以加强分布式架构的治理能力。流引用可被远程化，从而实现系统级的流式贯通，而流提供的更好的回压、三角模式透传，以及天然的截面编程能力等，可以给架构治理提供更好的帮助。目前淘宝正在推进回压的方案，就是为了给系统在稳定性和资源利用率上提供更高级的治理手段。



淘宝反应式架构实践

淘宝之所以要做架构升级，是因为现有架构存在一系列问题：

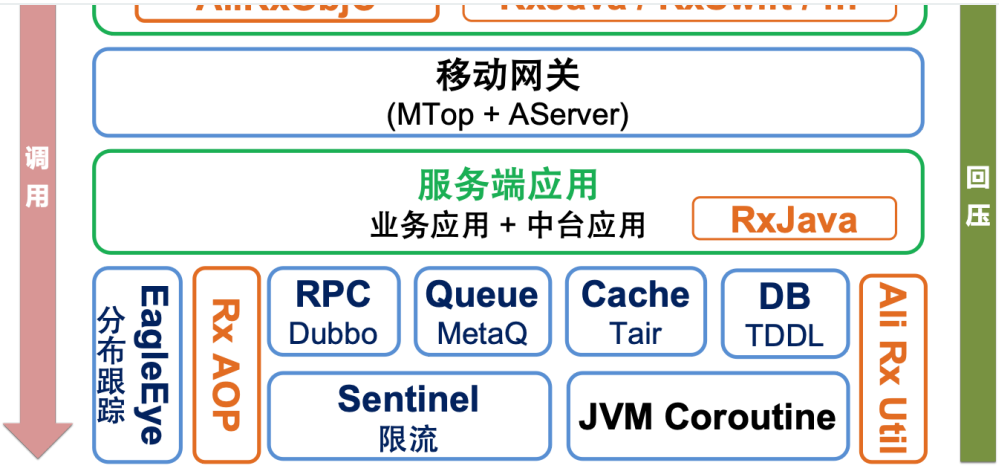
- 同步等待造成资源浪费，现有的同步模型线程多负载高，导致资源利用率较低。
- 架构的并行度有限，无法实现纯业务依赖并发，微服务化让问题更加凸显，服务增加造成响应时间累积。
- 而响应时间累积又带来一些连锁反应，包括为了降低响应时间而过早的引入 cache，每个服务都需要设置超时来解决长时间无响应问题，而这些带来维护成本的提升，也提高了业务实现的复杂度。
- 同时，在应用系统无事先准备的情况下，面对突发大流量时，很容易被打挂，造成稳定性问题，导致用户体验严重下降。

而经过调研后，淘宝架构团队认为使用反应式架构是当前可行的一个方案。原因包括，Java 8 已经逐渐普及，因为它包含对 Lambda 的支持，这让开发者对 Lambda 的接受度大大提高；同时 Reactive 相关的业务框架在业界已有成熟的实现，RxJava 已经广泛在大小公司中应用；最后，包括 Java 9（引入 Reactive Sreams 规范 API）、Spring 5（引入 Reactor/WebFlux）、Spring Boot 2 都开始拥抱 Reactive，说明反应式编程的确是趋势。

整个方案对业务架构的升级主要包括编程框架、中间件，以及业务方的升级。中间件的升级，包括服务框架（RPC）、网关、缓存、消息（MQ）、DB（JDBC）、限流组件、分布式跟踪系统、移动端 Rx 框架。

这其中值得注意的包括，对服务框架的升级，流式实现将在 Dubbo 3 中放出；DB 中的异步集成使用 Ali JVM 协程或用线程池实现；移动端为了支撑已有的 iOS 应用，淘宝开发了 AliRxObjc 并即将开源。

最后，改造后的架构如图：



实施反应式架构的难点

反应式架构在各个模块上基本都有成熟的方案，除了个别领域如数据库，基本没有特别的瓶颈。实施反应式架构的难点主要在于工程师的思维转换，因为之前工程师主要使用同步式的思维写程序，突然要换成以流的方式编写，思维必须要做转换。

因此，要做到全面异步化，组织必须从上到下全力支持。淘宝的做法是，成立虚拟小组，在每个业务线里挑选能力比较强的同学统一进行异步式的培训和指导，之后由他们在团队内部推广。

同时，要让业务方有动力去做异步化的改造，需要让他们认识到这么做的好处，因此异步化改造首先要做出一些标杆性的成绩出来。这中间的策略包括选择面临瓶颈的地方，业务逻辑简单的， 以及业务压力不大的应用来进行试点，一旦做出成绩，就可以给其它团队以信心和动力。

淘宝应用架构升级后续规划

目前，淘宝的异步化改造在技术上已经大部分完成，后续的规划主要包括：

- 回压的实现
  - 分布式上下游的联动回压，解决高并发压力下的响应时间、有效请求数、系统自恢复、链路短板自发现等问题，解决系统在突发流量下的稳定性问题，以及提高资源利用率。
  - 自适应回压解决静态配置无法应对系统波动变化问题
- 实现全异步 / 流式为核心的服务框架，让业务方做到异步优先；
- 考虑引入 Kotlin 协程，它的异步设计符合过程式的编程习惯。

淘宝的回压方案，目前正在推进和实施的过程当中，后续他们也将会有更多的经验分享出来，欢迎关注。

架构 治理 方法论

最新评论

银狐 2019 年 01 月 18 日 09:34 0 回复
playframework一直在践行反应式开发，虽然是小框架，但是思路跟楼主是一样的。

海迷 2019 年 01 月 16 日 10:45 0 回复
好文章，收藏了

景b 2019 年 01 月 15 日 20:29 0 回复
mark

风云0312 2019 年 01 月 15 日 11:00 0 回复
这篇文章回答了什么是反应式架构的概念。又举了很多淘宝的例子。真是一篇不可多得的文章。 反应式编程是未来的趋势。作为Java技术栈的朋友们，这是下一个金矿。 希望能放出Slides或PPT，了解细节。