31 | 极端业务场景下，我们应该如何做好稳定性保障?

2018-02-28 赵成

讲述：黄洲君
时长 10:38 大小 4.87M

从今天开始，和你分享我对微服务和分布式架构下的稳定性保障的理解。

稳定性保障需要一定的架构设计能力，又是微服务架构比较核心的部分。在陈皓老师的“左耳听风”专栏，以及杨波老师的“微服务架构核心 20 讲”专栏都有非常详细的介绍。所以在我的专栏里，我会结合特定的场景，并着重从运维和技术运营的角度来分享。

我们所面对的极端业务场景

首先，看一下我们当前所面对的极端业务场景，我把它大致分为两类。

1.**可预测性场景**

什么是可预测性？简单来说，就像电商每年的大促，如 618、双 11、双 12 等等。这类业务场景是可预测的，业务峰值和系统压力峰值会出现在某几个固定的时间点，我们所做的所有准备工作和稳定性应对措施，都是针对这些固定时间点的，比如零点时刻。

峰值压力可预测，就意味着可以提前评估用户访问模型，并根据模型进行压测调优。发现系统中的瓶颈就调优或者扩容，调整完成之后，继续压测，继续调整，直至系统容量达到原来设定的目标。由此可见，在可预测的场景下，与后面的不可预测场景相对比，从准备过程上来说会更加从容。

但是，我们的优化或扩容是有限度的，也就是不会无限度地投入成本，来满足零点这个峰值时刻，让所有用户都能够正常访问。从成本和收益角度来说，这样做是不现实的。

所以，在峰值那个时间点上，当用户流量远远大于系统容量时，我们所采取的措施绝不是再去扩容或优化，因为无论是从时效性、系统稳定性还是成本收益上看，这样做都已经无法满足要求了。

那我们该采取什么策略呢？这里我们采取的策略是在系统承诺容量内，保证系统的核心功能能够正常运行。以电商为例，就是要确保整个交易链路是正常的，用户可以正常登陆，访问商品，下单并最终支付。对于非核心功能，就会通过预案执行功能降级。对于超出系统承诺容量的部分进行流量限流，并确保在某些异常状况下能够熔断或旁路，比如缓存故障，就要马上限流并将请求降级到数据库上。

所以，我们在 618，双 11 和双 12 的零点峰值时刻去访问各大电商网站，很大概率上都会提示系统正忙，请稍后再试，短则 2~3 分钟，长则 5~10 分钟，再去访问，网站功能就一切正常了。这并不代表各大电商网站宕机了，而是其在瞬时超大流量访问压力下采取的一种保护措施，这一点反而说明这些电商网站的大促预案非常完善。

2.**不可预测性场景**

我刚刚提到的电商大促场景，其实已经非常复杂了，没有一定的整体技术能力，想做好从容应对也并非易事。我们这两年做大促模拟压测，动辄上百号人通宵投入，说到底还是在这方面的经验以及各类工具平台的积累不够，体系的完善需要一定的周期和过程。

那不可预测的场景就更为复杂。社交类业务就具有这种明显的特征，比如微博、朋友圈、空间等等。以微博为例，我们知道之前鹿晗公布恋情，王宝强以及乔任梁等的突发事件等等，

这些事情什么时候发生，对于平台来说事先可能完全不知道，而且极有可能是大 V 的即兴发挥。当然，现在因为商业合作上的原因，某些大 V 的部分营销活动也会与各类社交业务平台提前沟通，确保活动正常执行，但是即使是提前沟通，周期也会非常短。

对于不可预测性的场景，因为不知道什么时候会出现突发热点事件，所以就无法像电商大促一样提前做好准备。社交类业务没法提前准备，就只能随时准备着，我认为这个挑战还是非常大的。

我们要迎接的技术挑战

说完了场景，我们来看看这给技术带来了哪些挑战。

1.**运维自动化**

这个不难理解，应对极端场景下的系统压力，一定要有资源支持，但是如何才能将这些资源快速扩容上去，以提供业务服务，这一点是需要深入思考的。结合前面我们讲过的内容，**标准化覆盖面是否足够广泛，应用体系是否完善，持续交付流水线是否高效，云上资源获得是否足够迅速，这些都是运维自动化的基础。**特别是对于不可预测的场景，考验的就是自动化的程度。

2.**容量评估和压测**

我们要时刻对系统容量水位做到心中有数，特别是核心链路，比如电商的交易支付链路。我们只有对系统容量十分清楚，才能针对特定场景判断出哪些应用和部件需要扩容，扩容多少，扩容顺序如何。同时，系统容量的获取，需要有比较完善的自动化压测系统，针对单接口、单应用、单链路以及全链路进行日常和极端场景下的模拟压测。

3.**限流降级**

我们前面提到了电商大促的例子，业务在峰值时刻，系统是无论如何也抵御不住全部流量的。这个时候，我们要做的就是保证在承诺容量范围内，系统可用；对于超出容量的请求进行限流，请用户耐心等待一下。如何判断是否需要限流呢？这时我们要看系统的各项指标，常见的指标有 CPU、Load、QPS、连接数等等。

同时，对于非核心的功能，在峰值时刻进行降级，以降低系统压力。这里有两种方式，分别是**主动降级**和**被动降级**。主动降级就是在峰值时刻，主动把功能关掉，如商品评论和推荐功

能等等；我们前面介绍到的静态化，也是一种降级策略。对于被动降级，也就是我们常听到的**熔断**。某个应用或部件故障，我们要有手段将故障隔离，同时又能够保证业务可用，所以会涉及故障判断和各类流量调度策略。

4.**开关预案**

上面介绍到的限流降级，也是一类开关，属于**业务功能开关**；还有一类是**系统功能开关**，比如当缓存故障时，我们就需要将请求转发到数据库上，目的也只有一个，让系统可用。但是问题来了，数据库的访问效率没有缓存高，所以缓存可以支撑的流量，数据库肯定是支撑不了的，怎么办呢？这时，就要与限流策略结合起来，先限流，限到数据库能够支撑的容量，再做降级。这样几个策略组合在一起，就是应急预案的执行了。当然，预案里面还会有业务预案。

5.**故障模拟**

上述预案，需要在日常，甚至是从经历过的故障中提炼出场景，制定好策略，然后不断进行模拟演练。只有这样，等到真正出现问题时，我们的预案才可以高效执行。我们知道 Netflix 的 Chaos Engineering，其中的 Chaos Monkey，就是专门搞线上破坏，模拟各种故障场景，以此来看各种预案执行是否到位，是否还有可以改进的地方。

所以，类似 Chaos Engineering 的**故障模拟系统**，也需要建设起来。我们需要模拟出一些场景，比如最常见的 CPU 异常，RT 响应异常，QPS 异常等等，看我们的预案是否能够快速执行，能够保持系统或将系统快速恢复到正常状态。

6.**监控体系**

最后，我再提一下监控。通过我们前面介绍的内容，监控的重要性就不言而喻了，因为所有的指标采集和统计，异常判断，都需要监控体系的支持。监控体系和前面介绍的运维自动化一样，都是最为基础的支撑平台。

极端业务场景下的不确定因素

上面我们讨论了极端业务场景给技术层面带来的挑战。但是**对于稳定性保障而言，我认为最困难的部分，不在技术层面，而是在业务层面，也就是用户的业务访问模型**。从技术层面来说，我们还有一些确定的套路可以去遵循，但是访问模型就是个极不确定的因素了。

我们这里还是以电商来举例说明，比如大促时用户下单这个逻辑。一个用户在购物车勾选商品去结算这个场景，用户的访问模型或业务场景就可能会有很多变化，比如是下 1 个商品的订单，还是同时 5 个商品的订单？每个商品的购买数量是 1 个、2 个还是 3 个？商品的购买数量有没有限制？这些商品涉及 1 个卖家，还是多个卖家？不同卖家又会有不同的优惠折扣，是买二送一，还是满 100 送 20？满一定额度之后是否包邮？全站促销是否有全站优惠，是否有时间段限制？优惠之间是否有优先级和互斥逻辑？支付方式是优先使用支付宝，还是微信，亦或是银行卡等等。

上面这些还只是简单描述，并且是针对单个用户的。当用户数量达到几十万，上百万之后，用户行为可能还有访问首页、详情页以及搜索的情况等等，场景更复杂，整个业务模型中的变量因素就会非常多，且不确定。

往往某个因素的变化，就可能带来容量模型的改变。假设 1 个商品，1 个卖家，1 个优惠策略，对 DB 产生的 QPS 是 20，TPS 是 10，但是这其中的因素稍微一变化，产生的 QPS 和 TPS 就是翻倍的。如果这一点评估不到位，大促时实际的用户场景跟预估的偏差过大，系统可能就会挂掉。

所以，**对于稳定性而言，用户访问模型才是关键，这个摸不准，只有技术是没用的，这就更需要我们能够深入业务，理解业务**。

我们经常听到的“脱离业务谈架构，谈技术，都是不负责任的”，原因就在于此。希望今天的内容能够让你在学习到知识和技能的同时，也有所启发，切忌脱离业务，空谈技术。

今天我们分享了极端业务场景下，如何做好稳定性保障。关于稳定性保障，你还有哪些问题？有过怎样的经验？欢迎你留言与我讨论。

如果今天的内容对你有帮助，也欢迎你分享给身边的朋友，我们下期见！

上一篇　30 | 云计算时代，我们所说的弹性伸缩，弹的到底是什么？

下一篇　32 | 稳定性实践：容量规划之业务场景分析

精选留言 (1)

写留言

feie22
2018-02-28

3

对于熔断和降级实现方式，应该是研发主导吧，能举个具体的例子说明下吗

展开

作者回复: 技术实现上研发主导，但是场景梳理，产品设计上，还是会依赖运维，运维离线上环境是最近的，这个优势并不是所有开发都具备的。

后面文章还会介绍到，可以关注下