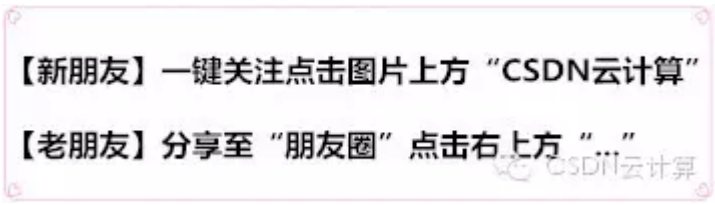
履单流程的弹性架构——麦包包峰值架构实践

CSDN云计算 2014-11-22



OMS（订单管理系统）是电商ERP系统中的核心模块，其中的订单履行流程（履单流程）是消费者购物过程中有直接感知的最后一段，关系到用户体验，其正确性和时效性必须得到保证。同时履单流程也是电商系统中直接面对销售高峰带来的短时间内剧增的数据量的子系统之一，如何在流量骤增10倍甚至更多的情况下保证OMS的正常服务，是每一家电商密切关注和不断改进的重点。

与前端接单流程不同，履单流程无需提供秒级实时响应，但它要执行的工作不单是生成订单保存入库，而要协调ERP中诸多子系统、外部系统及人工处理，共同完成一个订单的出库发货，因此面临的问题和需要的解决方案都有所区别。

在麦包包，我们经历了一小时接单峰值从千级到万级再到十万级的高速增长，虽然OMS中有中间表、分页批处理等抗冲击设计，但在一年比一年高的流量峰值冲击下，仍不时出现阻塞甚至崩溃的迹象，让人胆战心惊。为解决这个问题，需要对整个履单流程所有环节进行分析，找出瓶颈和浪涌威胁，制定有针对性的解决方案，进行评估和测试，明确系统的峰值处理能力，保证在各种情况下的正确处理。

履单流程的流量治理同洪水治理道理相似，要找出高水位或者决堤的位置进行加固。加固的方法可以分为两个方向：

- 疏通和拓宽河道——优化算法、合理并行；
- 蓄水限流——异步、缓冲、限量。

本文将结合麦包包的履单流程面对流量暴增问题时，需要研究的流程节点分析、执行效率优化、入口隔离和异步化、固定流控和弹性架构等方面进行讨论。

关注点：一致性、可用性和峰值处理能力。

最低目标：流量不高于每小时10万单情况下，能够保证每个无须人工确认订单30分钟内提交拣货；流量不高于每小时100万单情况下，能保证每个订单最慢在6小时内提交拣货。

高级目标：流量高于每小时10万单时能动态提升吞吐能力，在流量不高于15万单情况下保持每个订单30分钟内提交拣货。
在讨论治理方案之前，我们需要将流程抽象和分解，进行深入分析。

流程节点分析

要保证系统在流量暴增的情况下正常提供服务，整个流程不能有任意一点产生阻塞。所谓阻塞点是直接或间接造成某个共享计算资源超负荷的点。电商OMS中，CPU和内存的密集计算一般不多，这个超负荷的计算资源多为DB Server。到底是哪一种类型的计算资源并不是最重要的，我们需要先找到引发或可能引发超负荷的原因，再找到其所在程序位置去解决。

履单流程从业务逻辑上一般分为前后衔接的多个步骤，我们将其称为处理节点，我们要做的就是分析流程中各个处理节点，特别是以往产生过阻塞的节点。分析时除了一般性的代码、数据结构的静态分析和单元测试，对产生过阻塞的节点还可以从现场日志入手去分析。例如，对于DB Server超负荷的情况，要分析阻塞现场的SQL队列，找到引起DB Server资源超载的主要几条SQL，再顺藤摸瓜找出触发这些SQL的代码位置和其所属的处理节点。

麦包包的履单主流程包括订单审核、支付确认、出库单生成、快递匹配等节点，简化的流程如图1所示。

图1 履单逻辑流程

通过代码和执行日志分析，我们发现订单人工审核、物流匹配耗时较长，订单的自动审核缺乏流量控制等问题。流程中节点耗时过长不仅会对整个流程的执行时间有直接影响，更重要的是当大量订单涌入时可能堆积过多的数据库慢查询，造成数据库并发过高、超负荷。人工订单审核虽然不是自动流程的一环，但由于和自动流程访问相同的数据库，在面临大量订单时同样也会拖慢数据库性能，造成大量的锁。

提交拣货到提交发货这一段属于WMS的范围，也需要持续优化，但这超出了本文的主题不在这里讨论。

这些问题需要逐一解决，首先我们需要对单节点的性能进行优化。

基本处理能力优化（单节点优化）

流程由一个个处理节点组成，如果各节点本身还有优化的余地，那么整体分析就难以把握最优方向。因此在流程级优化之前应该先对各个节点进行优化，确保计算处理本身已足够

高效，也包括资源的合理使用和每个节点在架构上的可伸缩性。

流程整体调优前，首先要消除各个节点的执行效率瓶颈，做算法优化，提高并行效率，减轻或合理分布计算资源的占用。

以针对SOM（销售订单管理）模块进行的优化为例，优化前SOM模块的平均请求响应时间为1.89秒，优化后降低到了0.09秒。优化后单服务器每分钟可响应请求667次，而优化前只有32次左右，在系统架构和硬件资源不变的前提下，提高了节点的处理能力，降低了资源的占用。

做单节点的优化，主要侧重于逻辑优化、算法优化和代码优化等。这个话题已被讨论很多，基本原则如下。

- 优化算法，选择合适高效的算法，降低不必要的递归、循环、多层循环嵌套等计算。用简单的算法完成大部分情况，不要为少数特例而将算法复杂化。特例由特殊的分支处理。
- 避免申请过多不必要的内存开销。
- 及时释放资源，降低资源占用时间，包括内存、I/O、网络和数据库等。
- 善用缓存：缓存常用的、不易变化的；偶有变化，可以考虑缓存依赖机制。
- 慎用数据库锁。
- 恰当地使用事务，事务要细粒度。
- 选择适当的通信方式：Socket、Remoting、Web Services（REST和SOAP）、WCF、 Named Pipes等，要特别注意长连接和短连接的恰当使用。
- 计算并行化。
- 降低系统或模块之间的通信次数，例如工作流服务和数据库服务。
- 降低系统或模块之间的传输数据量，不必要传输的不传或少传。
- 异步计算，降低等待时间。
- 考虑延迟加载和提前加载两种方式。
- 分离原则：分离业务模块，如分离大I/O模块、分离高耗内存模块和分离高耗宽带模块。
- 统筹使用计算资源，如寻求内存计算、数据库计算和网络开销三者之间的最佳平衡。

在单节点优化中，以上方法都可能用到。而对于履单流程来说，计算并行化和节点伸缩性也会起到比较重要的作用。

……

作者周海冰，麦包包技术总监，中国第一波互联网大潮的见证者。目前致力于打造智能化的电商交易和运营平台。

杨鹏，麦包包高级架构师，较早从事电商和云相关工作的架构师，原亿达云计算研究所骨干之一，打造了国内第一个线缆行业的垂直电商平台，并多次作为云专家参与国家级科技项目规划。

本文节选自《程序员》11月刊，欢迎订阅《程序员》杂志iPad/Android版。

阅读原文