# 服务端消息转储

独立工程，SprintBoot编写。

功能：部署在slave上，通过解析CommitLog分解出消息，并将消息存入MongoDB中。

原因：为后期对消息做细粒度的、带有业务字段的查询；出报表，多维度统计消息；历史消息消费和消息归档

架构：4大模块：文件监控、读消息线程、持久化线程、偏移量本地持久化县城。功能解耦，学习了源码中的服务线程的概念，继承ServiceThread。4者的关系是相互通知的。文件监控通知读线程CommitLog变更，读消息；读到消息，解析，封装后通知持久化线程转储；转储成功通知偏移量本地持久化线程，记录偏移量。线程之间均为异步，使用队列缓存请求。

文件监控使用了Guava的FileAlertationMonitor，当CommitLog发生增、删、改时触发不同的回调函数。

而读消息采用追赶者的策略，文件监控通知一次会一直向后读，直到读到最后一条消息。最后一条消息的判断方法是读到错误的魔数。

对读到的消息进行解析，消息包含properties，kv格式，将其展开，平铺，并动态的构造消息对象，因为不同类型消息properties不同。这里使用了Javassist，同时还会将构造的对象缓存起来，不必每条消息动态构造类。

持久化时走索引，插入效率高。插入时还会判重，依靠msgId和brokerName判重。判重的原因是本地存的偏移量落后于已经存储的位置，重启服务时从落后的位置开始入库。因此在msgId和brokerName创建联合索引。走了索引，插入耗时跟已经存入的消息数就无关。

优雅停机，线程间均为异步且服务中包含线程池，所以队列、线程池中可能还有一些请求没执行完毕，而且线程之间有顺序性。因此要按顺序，优雅关闭线程池，等待队列中请求完成。优雅启停试用了ApplicationLisnter。

经过测试，生产者发500w条消息，MongoDB中存储500w；生产者发100w消息，持久化进程启停20次，MongoDB中存储100w消息。具有良好的可靠性。

如果消息存储失败，这条消息会发给mq特定的topic，延迟消息，订阅后再次尝试插入，直到变为死信。

跟slave部署在一起的优势是可以利用操作系统共享内存的特性。slave和转储进程都会映射CommitLog，但在内存中只映射一次，slave的任何写操作，转储进程无需从磁盘上获取，内存中即可读到。