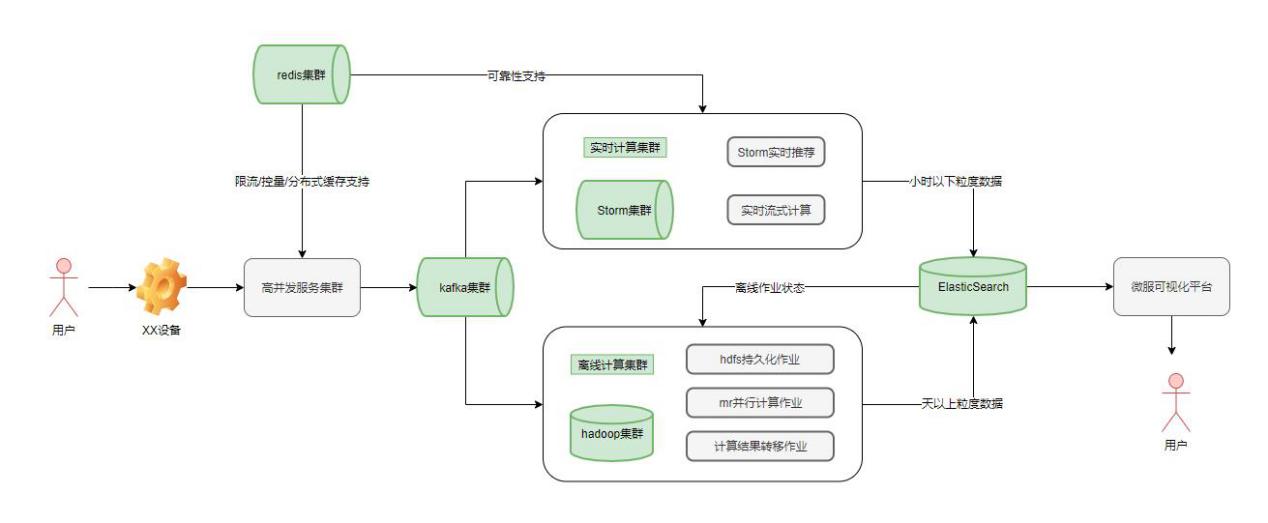
**一、场景和背景**

主要针对手机APP、穿戴设备等采集的数据进行分析；本套Demo属于笔者休假期间根据过去工作所学、所思有感而设计；主要利用当前掌握技术对系统进行一些改进，也加深对技术点的熟悉度。

**二、数据流向图**



关于基础组件的版本，普遍选择地域最高版本下面的几个版本，避免去踩最高版本的坑，避免使用很低的版本存在各种问题，后面又需要升级。1)高并发服务器采用Netty设计，版本是4.1.42.Final，Netty在4.0以上版本属于大版本优化；2)Redis版本4.0.3；3)kafka版本2.10-0.10.1.0，这个版本消费者比较稳定；4) Storm版本1.2.3；5) Hadoop版本2.8.5，一般选择较高版本，目前低版本会扫描出很多漏洞；6) Elasticsearch是7.3.2，Elasticsearch的高版本后期逐渐引进high-level-client，原先的Client已经废弃，这里API编写查询会有出入；7)可视化平台后台使用Springboot 1.5.9；8)JDK 1.8.0\_231

关于系统的业务线，1)以{设备ID，设备地区，APP渠道，制式，时间戳}格式统计不同维度和时段的设备数;2)以{书籍CODE，书籍类型，书籍名称}格式做实时推荐

**三、分布式并发服务**

1) 高并发服务设计

属性：处理定向URL的HTTP请求

缓存：分布式二级缓存，本地Cache做一级缓存，Redis做二级缓存，分布式缓存同步利用Redis发布订阅完成统一

设备限流：Redis Bitmaps实现，Bitmap根据一致性hash原则分布到各个节点

设备计量：Redis HyperLogLog

数据去重：请求头带UUID，在CRC64 位算法hash后入Redis，短时间精确去重

可靠性：kafka异常状态下写本地磁盘，Redis提交命令Pipeline设置提交尺寸

**四、分布式计算服务**

1) 离线批计算设计

属性：分布式并行聚合长时间数据，目前代码中是聚合小时

数据流程：Kafka-hdfs-mr-hdfs-es

可靠性：定时批量消费kafka，kafka位移按照分区和offset提交，kafka位移的提交和hdfs文件写入维持事务；在数据流程上设计状态码，任何流程中断，状态更新也随之中断，系统自动从状态码开始重新执行

分布式一致性：Redis setnx做分布式锁

2) 实时流计算设计

属性：实时推荐计数等

可靠性：kafka位移异步提交，异步Bolt中失败情况入Redis和磁盘，定义失败信息处理Spout，即正常流程一个Spout，异常流程一个Spout

设备状态判定：Redis 的Bitmaps实现，32位hash的量在20+亿，这里hash只给32位，存在少量hash冲突的可能性

1. **分布式存储选型**
2. Druid

优势，具备预聚合功能，在大数据量的摄入和聚合方面优势明显，成本低；劣势，随着数据量增大Druid维护难度大，查询功能难用不完善，查询命中率低精度不稳定，也不支持修改数据，一旦成型的数据无法修改，上游的数据开发者压力可想而知，存储数据主要支持结构化数据，作为时序数据库也要求数据结构中必须带有时间字段

1. Elasticsearch

劣势，数据摄入的查询所消耗的资源大，属于实时聚合；优势，查询功能非常完善，命中率高，搜索能力强大，对各类数据支持，增删改操作完善，维护成本低

综上，Druid和Elasticsearch最大的差异在于预聚合的能力，本文所引入的离线计算目的就是弥补ES在预聚合方面的缺陷，实时聚合在海量查询时必然会遇到瓶颈，最终数据量会达到算法和各项调优都无济于事的地步，因此充分利用闲时服务器资源进行预先处理是一条有效出路