数字货币交易所源码开发火币、币安、OKEX等产品系统架构设计

产品名称	数字货币交易所源码开发火币、币安、OKEX等产品系统架构设计
公司名称	河南漫云科技有限公司
价格	1000.00/件
规格参数	漫云科技:数字货币量化平台websocket开发
公司地址	郑东新区升龙广场3号楼A座3202
联系电话	13103827627 13103827627

产品详情

部门领导让我研究数字货币交易所的kaifa技术，好不容易把码云（Gitee）上开源数字货币交易所CoinExchange的代码编译成功并搭建出来了，在排查问题的过程中，不断的查看代码以及使用到的技术，因此对系统的整个架构有了一定的初步认识，这里做个笔记记录一下。

系统整体架构

这是开源项目CoinExchange上作者放的一张逻辑架构图，猛一看其实没什么感觉，但是当我通过代码层级的阅读，以及各种软件的安装，对这个图的认识更加深刻了。

数据存储

首先，我们从数据存储这一块看，该项目使用了三种数据存储方式，另外还有一个数据存储是内存，在撮合交易引擎中，直接通过Java的并发链表存储的。

MySQL主要存储业务数据，一共有65张表。

MongoDB主要存储委托成交明细和K线数据（成交量、成交价），如下所示是K线数据，按照一定的时间周期统计所得。

消息通讯

因为项目是基于SpringCloud微服务架构kaifa，所以这套交易系统有很多的服务，这些服务之间的通信本可以通过Euruka服务注册中心调用相应的服务，但是这套系统使用了Kafka，Kafka是一种高吞吐量的分布式发布订阅消息系统，它可以处理消费者在网站中的所有动作流数据。大概是因为交易系统中对委托的处理需要非常迅速的处理能力和稳定性，所以这个项目使用了Kafka。

通过代码层面，可以看到，用户通过Exchange-api服务下单以后，它会将委托订单保存到数据库，同时会把委托订单发送到Kafka，通过Kafka传送给消费者Exchange（撮合引擎），当Exchange（撮合

引擎）完成撮合以后，它会将委托成交明细发送给Kafka，接着由消费者Market(行情引擎）完成数据的存储（持久化）。

这样的设计可以让Exchange（撮合引擎）只需要专心处理撮合就可以，也能够充分发挥它的性能。

前后端分离

整套系统的前端与后端完全分离开，这是比较主流的kaifa方式，可以让后端kaifa人员与前端kaifa人员各自专注于自己的业务实现。目前可以看到前端主要有四个：用户PC端、用户Android端、用户IOS端、管理员PC端。它们都是通过Api与服务对接，传输数据是通过Json。

作为饱受上面代码的受害者，前后端分离的设计可以说是解放了我们这些程序员。

qukuailian钱包接口

项目中对每个币种的RPC接口做了一层抽象，作为抽象层的wallet项目，屏蔽了不同币种的对接问题，qukuailian钱包节点的RPC调用方式千奇百怪，项目中通过wallet把生成地址、扫块、充值监控、余额归集等操作抽象出来，当我们想接入新的币种的时候，只需要对Wallet-RPC-XXX项目进行复制粘贴就可以了。

前端技术实现

前端作者也用了比较流行的vue框架，对前端项目实现了很好的MVC解耦，kaifa人员不用频繁的操作html元素，只需要专注于对数据的处理就可以了，让前端的kaifa可以变得很优雅。比如下面，我截取了一段代码：

交易机器人

从作者的架构图可以看出来，交易机器人通过同步获取到了各大交易所的交易数据，进而在自身交易所种绘制相应的K线，在我跟作者的沟通过程中，我了解到了机器人的大概设计原理，尤其是其中有很多参数的设计，非常关键，可以让盘面表现出跟大型交易所一样的行情展示效果。