**服务器端第一阶段接口需求文档**

1. 系统架构

本服务器采用主动服务型推送式，根据用户的需求一旦数据有变化主动给用户推送数据，提供查询接口，大体分为以下几个模块；

1. 前端接口模块(负责处理用户的请求)，主要是查询接口，并负责将数据打包成json格式
2. 第三方API模块，负责和第三方API交互 实时获取第三方的行情和交易数据，并将数据缓存到服务器中。
3. 数据库模块，负责存储和提供数据查询
4. 数据推送模块，根据用户的需求能够在数据发生变化时及时向用户推送和更新最新的行情和交易数据
5. 日志模块，记录用户的活动情况
6. 登录、注册、认证模块。未登录用户可以通过Token（128位哈希码）访问合约和行情信息。获得交易信息需要用户注册并登录。采用JWT协议保存用户的登录会话。

初步设计为如上的系统架构，在具体实现中可能会稍作修改，服务器完成后再具体完善文档。

1. 数据库设计

使用MongoDB，提高数据库查询速度

通过第三方接口接收服务器信息，并保存至本地数据库

通过websocket协议，为用户提供查询服务，并发量5000+

1. 用户查询接口
2. 请求

{

"type" : "type of request"

"data" : [parameters about "exCode", "proType", "conCode" or "kType"]

}

**请求类型type:**

HET：心跳包。客户端必须定期向服务器传心跳包，否则会被服务器踢掉，具体心跳时间待定。

CON: 查询合约

MAR: 查询行情

EMR: 根据交易所查询行情

KLE: 查询K线

SUB: 订阅合约

**产品类型** proType**:**

比如原油，国企指数等(在第一次合约请求中会返回给客户端)

**K线类型kType:**

"ONE","THR","FIV","TEN","HAF","SIT","FOH","DAY","WEK","MON";

分别代表1 3 5 10 30 60 4hours 1day 1week 1month

**交易所代码exCode：**

例如“CME”

“ALL”表示所有交易所

**合约代码conCode：**

例如“CL1712”

“ALL”表示所有合约

具体的请求格式和返回格式如下(第一阶段采用轮询式访问)

原则: type类型必须有, data字段根据type的请求视情况传递：

1. **心跳包请求**

{

“type”: “HET”,

“data”: []

}

返回:

{

"state" : 1, //1表示请求成功，0表示请求失败

"errorCode" : error\_code,

"errorMsg" : "error\_massage",

}

1. "type"=="CON"(合约代码查询)

{

“type”: “CON”,

“data”:[]

}

返回数据:

{

"state" : 1, //1表示请求成功，0表示请求失败

"errorCode" : error\_code,

"errorMsg" : "error\_massage",

"data" : {

"交易市场" : {

"交易品种" : [“合约名称”,"品种合约", "品种合约", …],

"交易品种" : [“合约名称”,"品种合约", "品种合约", …],

……

},

"交易市场" : {

"交易品种" : [“合约名称”,"品种合约", "品种合约", …],

"交易品种" : [“合约名称”,"品种合约", "品种合约", …],

……

},

……

}

}

1. **"type"=="MAR"（市场行情查询 可以查询多个合约）**

请求数据

{

“type”: “MAR”,

"data":[

[ "exCode1", "conCode1"],

["exCode2", "conCode2"],

……

]

}

返回数据：(标红的表示api中无法获取到的数据 暂时没管)

{

"state" : 1, //1表示请求成功，0表示请求失败

"errorCode" : error\_code,

"errorMsg" : "error\_massage",

"data" : [

[交易日,合约代码,交易所代码,最新价,现手,持仓量,上次结算价,本次结算价,涨跌,涨跌幅,当日均价,买量,卖量,申买价一,申买量一,申买价二,申买量二,申买价三,申买量三,申买价四,申买量四,申买价五,申买量五,申卖价一,申卖量一,申卖价二,申卖量二,申卖价三,申卖量三,申卖价四,申卖量四,申卖价五,申卖量五],

[交易日,合约代码,交易所代码,合约在交易所的代码,最新价,现手,持仓量,上次结算价,本次结算价,涨跌,涨跌幅,当日均价,买量,卖量,申买价一,申买量一,申买价二,申买量二,申买价三,申买量三,申买价四,申买量四,申买价五,申买量五,申卖价一,申卖量一,申卖价二,申卖量二,申卖价三,申卖量三,申卖价四,申卖量四,申卖价五,申卖量五],

……

]

}

1. **“type”=”EMR” (合约的查询 查询涨跌和持有量 新增接口 根据交易所查询该交易所下所有合约的行情)**

请求数据

{

“type”: “EMR”,

"data":[ “exCode” ]

}

返回数据

{

"state" : 1, //1表示请求成功，0表示请求失败

"errorCode" : error\_code,

"errorMsg" : "error\_massage",

"data" :[

[合约代码，最新价，涨跌，涨跌幅，成交量，持仓量]

[合约代码，最新价，涨跌，涨跌幅，成交量，持仓量]

……

]

}

1. **“type”=“SUB”（查询订阅合约的信息）**

请求数据

{

“type”: “SUB”,

"data":[

[ "exCode1", "conCode1"],

["exCode2", "conCode2"],

……

]

}

返回数据

{

"state" : 1, //1表示请求成功，0表示请求失败

"errorCode" : error\_code,

"errorMsg" : "error\_massage",

"data" :[

[合约代码，最新价，涨跌，涨跌幅，成交量，持仓量]

[合约代码，最新价，涨跌，涨跌幅，成交量，持仓量]

……

]

}

1. "type"="KLE"(单个合约的K线)

请求数据

{

“type”: “KLE”,

"data":[ “exCode”,”conCode”,”kType” ]

}

返回数据

{

"state" : 1, //1表示请求成功，0表示请求失败

"errorCode" : error\_code,

"errorMsg" : "error\_massage",

"data" : [ //30个点

[时间,开盘,最高,最低,收盘,涨跌,涨跌幅,成交量]

……

]

}