第三方与行情服务器通信接口说明

2014.12.31

## 说明

本接口是在期货、期权行情的基础上扩展的支持证券行情，证券和期权合约代码等有不同的表示，在阅读时请留意。

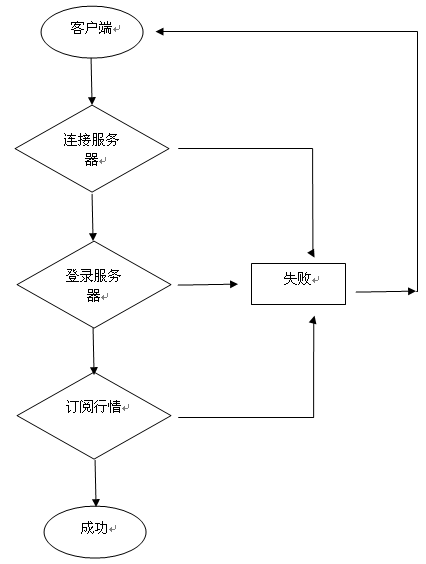
## 网络通信模块设计

服务端和第三方模块基于socket 套接字的TCP/IP网络通信方式。

## 通信思想

流程：

1. 行情客户端连接行情服务器。
2. 行情客户端发送登录请求。
3. 行情客户端发送验证用户订阅行情事件请求。



# 数据包结构

1. 所有数据包遵循统一格式：包头 + 数据 + 包尾
2. 数据包结构按字节对齐

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 数据项 | | 数据类型 | 字节长度 | 备注 |
| 包头 | PkgHead | headid | unsigned int | 4 | 恒为0x7cab5caa |
| len | int | 4 | 内容长度，不含包头包尾 |
| compressflag | int | 4 | 包体压缩标识 |
| prioritylevel | int | 4 | 优先级 |
| cmdid | unsigned int | 4 | 命令字 |
| moduleid | unsigned int | 4 | 创建者模块标志（未使用） |
| seq | unsigned int | 4 | 请求包的序号 |
| subseq | unsigned int | 4 | 子序列号（未使用） |
| userdata1 | unsigned int | 4 | 请求成功或错误状态 |
| Userdata2 | unsigned int | 4 | 当前包序号 |
| Userdata3 | unsigned int | 4 | 总包数 |
| Userdata4 | unsigned int | 4 | 业务层按需使用（未使用） |
| crc32 | unsigned int | 4 | CRC校验字 |
|  |  |  | 包内容，可以没有 |
| 包体 | PkgBody |  | unsigned int | 4 | 恒为0xba55ca81 |
| 包尾 | PkgTail |  |  |  |  |

# 业务通讯协议

## 心跳

### 心跳请求

|  |  |  |
| --- | --- | --- |
| PkgHead | len | 0 |
| cmdid | CMDID\_HEARTBEAT |
| moduleid | 0 |
| PkgBody |  | NULL |
| PkgTail |  |  |

### 心跳请求返回

|  |  |  |
| --- | --- | --- |
| PkgHead | len | 0 |
| cmdid | CMDID\_HEARTBEATRSP |
| moduleid | 0 |
| PkgBody |  | NULL |
| PkgTail |  |  |

## 3.2登录相关接口

### 3.2.1登录请求

|  |  |  |
| --- | --- | --- |
| PkgHead | len | sizeof(sLoginReq) |
| cmdid | Cmd\_QuotSvr\_Login\_Req |
| moduleid | 0 |
| PkgBody |  | sLoginReq数据 |
| PkgTail |  |  |

### 3.2.2登录返回

|  |  |  |
| --- | --- | --- |
| PkgHead | len | 成功：0  失败：错误信息长度 |
| cmdid | Cmd\_QuotSvr\_Login\_Rsp |
| moduleid | 0 |
| userdata1 | 成功：0  失败：错误信息 |
| PkgBody |  | 成功：NULL  失败：NULL |
| PkgTail |  |  |

## 3.3 行情相关接口

### 3.3.1行情订阅请求

|  |  |  |
| --- | --- | --- |
| PkgHead | len | sizeof(合约ID) |
| cmdid | Cmd\_SubscribeQuotEvent2\_Req |
| moduleid | 0 |
| PkgBody |  | 合约ID （证券 交易所+合约代码 如上交所的000342表示为sse000342，深交所的表示为szse000342） |
| PkgTail |  |  |

### 3.3.2行情订阅返回

|  |  |  |
| --- | --- | --- |
| PkgHead | len | 成功：0  失败：错误信息长度 |
| cmdid | Cmd\_SubscribeQuotEvent2\_Rsp |
| moduleid | 0 |
| userdata1 | 成功：0  失败：错误信息 |
| PkgBody |  | 成功：CThostFtdcSpecificInstrumentField  失败：NULL |
| PkgTail |  |  |

### 3.3.3行情退订请求

|  |  |  |
| --- | --- | --- |
| PkgHead | len | sizeof(合约ID) |
| cmdid | Cmd\_UnSubscribeQuotEvent2\_Req |
| moduleid | 0 |
| PkgBody |  | 合约ID |
| PkgTail |  |  |

### 3.3.4行情退订返回

|  |  |  |
| --- | --- | --- |
| PkgHead | len | 成功：0  失败：错误信息长度 |
| cmdid | Cmd\_UnSubscribeQuotEvent2\_Rsp |
| moduleid | 0 |
| userdata1 | 成功：0  失败：int，错误信息 |
| PkgBody |  | 成功：CThostFtdcSpecificInstrumentField  失败：NULL |
| PkgTail |  |  |

### 3.3.5行情推送

|  |  |  |
| --- | --- | --- |
| PkgHead | len | sizeof(PlatformStru\_DepthMarketData) |
| cmdid | Cmd\_QuotEvent\_Push |
| moduleid | 0 |
| userdata1 | 0 |
| PkgBody |  | PlatformStru\_DepthMarketData |
| PkgTail |  |  |
|  |  |  |

## K线接口

### K线数据请求

|  |  |  |
| --- | --- | --- |
| PkgHead | len | sizeof(KLineQuery) |
| cmdid | Cmd\_QueryQuotKLineData\_Req |
| moduleid | 0 |
| PkgBody |  | KLineQuery数据 |
| PkgTail |  |  |

### K线数据返回

|  |  |  |
| --- | --- | --- |
| PkgHead | len | 成功：N\*sizeof(KLineBase)  失败：错误信息长度 |
| cmdid | Cmd\_QueryQuotKLineData\_Rsp |
| moduleid | 0 |
| userdata1 | 成功：0  失败：int，错误信息 |
| PkgBody |  | 成功：N\*sizeof(KLineBase)  失败：NULL |
| PkgTail |  |  |

### 分时线数据请求

|  |  |  |
| --- | --- | --- |
| PkgHead | len | sizeof(KLineQuery) |
| cmdid | Cmd\_QueryQuotTimeLineData\_Req |
| moduleid | 0 |
| PkgBody |  | KLineQuery数据 |
| PkgTail |  |  |

### 分时线数据返回

|  |  |  |
| --- | --- | --- |
| PkgHead | len | 成功： N\*sizeof(TLineBase)  失败：错误信息长度  maxQueryCount：来自配置文件Account.xml |
| cmdid | Cmd\_QueryQuotTimeLineData\_Rsp |
| moduleid | 0 |
| userdata1 | 成功：0  失败：int，错误信息 |
| PkgBody |  | 成功：N\*sizeof(TLineBase)  失败：NULL |
| PkgTail |  |  |

## 期权接口命令字

### 波动率请求

|  |  |  |
| --- | --- | --- |
| PkgHead | len | sizeof(Volatility) |
| cmdid | Cmd\_QueryVolatilityData\_Req |
| moduleid | 0 |
| PkgBody |  | Volatility数据 |
| PkgTail |  |  |

### 波动率数据返回

|  |  |  |
| --- | --- | --- |
| PkgHead | len | 成功：sizeof(Volatility)  失败：错误信息长度 |
| cmdid | Cmd\_QueryVolatilityData\_Rsp |
| moduleid | 0 |
| userdata1 | 成功：0  失败：int，错误信息 |
| PkgBody |  | 成功：Volatility  失败：NULL |
| PkgTail |  |  |

# 定义

#define CMDID\_HEARTBEAT     0x7fff0001     //心跳命令  
#define CMDID\_HEARTBEATRSP     0x7fff0002     //心跳答复

#define CMDID\_BASE\_HIWORD\_QUOT 0x000F //行情高位CMDID

#define Cmd\_QuotSvr\_Login\_Req MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x000C) //登录行情服务器请求

#define Cmd\_QuotSvr\_Login\_Rsp MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x000D) //登录行情服务器返回

#define Cmd\_SubscribeQuotEvent2\_Req MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x000E) //验证用户订阅行情事件请求

#define Cmd\_SubscribeQuotEvent2\_Rsp MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x000F) //验证用户订阅行情事件返回

#define Cmd\_UnSubscribeQuotEvent2\_Req MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x0014) //验证用户退订行情事件请求

#define Cmd\_UnSubscribeQuotEvent2\_Rsp

MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x0015) //验证用户退订行情事件返回

#define Cmd\_QueryQuotKLineData\_Req MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x0010) // 查询K线事件请求

#define Cmd\_QueryQuotKLineData\_Rsp MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x0011) //查询 K线事件返回

#define Cmd\_QuotEvent\_Push MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x0005) //行情事件推送

#define Cmd\_QueryQuotTimeLineData\_Req

MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x0012) //查询行情分时线事件请求

#define Cmd\_QueryQuotTimeLineData\_Rsp

MAKE\_CMDID(CMDID\_BASE\_HIWORD\_QUOT,0x0013) //查询行情分时线事件返回

# 数据结构

**5.1 用户登录请求结构**

struct sLoginReq

{

\_USERSTRING msUserName; //用户名

\_PASSWORDSTRING msPassWord; //用户密码

UserType meUserType; //用户类型 填 2（交易员）

eClientType meClientType; //客户端类型 填 1

};

typedef char \_USERSTRING[21];

typedef char \_PASSWORDSTRING[33];

enum eClientType

{

CLIENT\_TYPE\_ADMIN = 0, //管理客户端

CLIENT\_TYPE\_TRADE, //交易客户端

CLIENT\_TYPE\_RISK, //风控客户端

CLIENT\_TYPE\_SHOW //展现客户端

};

enum UserType

{

USER\_TYPE\_MANAGE, //管理员

USER\_TYPE\_RISK\_CONTROL, //风控员

USER\_TYPE\_TRADER, //交易员

USER\_TYPE\_VIEWER, //展现员

USER\_TYPE\_SLAVESERVER, //从服务器用户

USER\_TYPE\_MASTERSERVER, //主服务器

USER\_TYPE\_INVALID //无效的用户类型

};

**5.2 行情信息**

struct PlatformStru\_DepthMarketData2

{

///交易日

TThostFtdcDateType TradingDay;

///合约代码

InstrumentIDType InstrumentID;

///最新价

TThostFtdcPriceType LastPrice;

///今开盘

TThostFtdcPriceType OpenPrice;

///最高价

TThostFtdcPriceType HighestPrice;

///最低价

TThostFtdcPriceType LowestPrice;

///今收盘

TThostFtdcPriceType ClosePrice;

///成交金额

TThostFtdcMoneyType Turnover;

///持仓量

TThostFtdcLargeVolumeType OpenInterest;

///上次结算价

TThostFtdcPriceType PreSettlementPrice;

///数量

TThostFtdcVolumeType Volume;

///最后修改时间

TThostFtdcTimeType UpdateTime;

///申买价一

TThostFtdcPriceType BidPrice1;

///申买量一

TThostFtdcVolumeType BidVolume1;

///申卖价一

TThostFtdcPriceType AskPrice1;

///申卖量一

TThostFtdcVolumeType AskVolume1;

};

///指定的合约

struct CThostFtdcSpecificInstrumentField

{

///合约代码

TThostFtdcInstrumentIDType InstrumentID;

};

typedef char TThostFtdcInstrumentIDType[31];

typedef char TThostFtdcDateType[9];

typedef int TThostFtdcVolumeType;

typedef double TThostFtdcPriceType;

typedef double TThostFtdcLargeVolumeType;

typedef double TThostFtdcMoneyType;

typedef char TThostFtdcDateType[9];

typedef char InstrumentIDType[64];

typedef char TThostFtdcExchangeIDType[9];

typedef char TThostFtdcExchangeInstIDType[31];

typedef int TThostFtdcMillisecType;

typedef double TThostFtdcRatioType;

typedef char TThostFtdcTimeType[9];

**5.3 K线结构**

struct KLineBase //k线数据结构，分钟到月线都是这个数据结构

{

time\_t dwTime; //时间，从年月日开始的秒数

double fOpenPrice; // 开盘价

double fHighPrice; // 最高价

double fLowPrice; // 最低价

double fClosePrice; // 收盘价

double fTurnover; //成交金额

DWORD dwVolume; //成交量

DWORD dwOpenInterest; //持仓量

KLineBase& operator=(const KLineBase& other)

{

dwTime = other.dwTime;

fOpenPrice = other.fOpenPrice;

fHighPrice = other.fHighPrice;

fLowPrice = other.fLowPrice;

fClosePrice = other.fClosePrice;

fTurnover = other.fTurnover;

dwVolume = other.dwVolume;

dwOpenInterest = other.dwOpenInterest;

return \*this;

}

bool operator<(const KLineBase& src) const

{//从大到小排序；

return dwTime > src.dwTime;

}

} ;

**5.4 分时线**

struct TLineBase //分时数据结构

{

time\_t dwTime; //时间，从年月日开始的秒数

double fLastPrice; // 收盘价

double fTurnover; //成交金额

DWORD dwVolume; //成交量

DWORD dwOpenInterest; //持仓量

TLineBase& operator=(const TLineBase& other)

{

dwTime = other.dwTime;

fLastPrice = other.fLastPrice;

fTurnover = other.fTurnover;

dwVolume = other.dwVolume;

dwOpenInterest = other.dwOpenInterest;

return \*this;

}

bool operator<(const TLineBase& src) const

{//从大到小排序；

return dwTime > src.dwTime;

}

};

**5.5 查询K线、分时线结构**

enum EnumPhrase

{

PHRASE\_1MIN = 1,

PHRASE\_5MIN = 5,

PHRASE\_15MIN = 15,

PHRASE\_30MIN = 30,

PHRASE\_60MIN = 60,

PHRASE\_DAY = 24\*60,

PHRASE\_WEEK = 7\*24\*60,

PHRASE\_MONTH = 31\*24\*60,

PHRASE\_USERDEFINE = 0

};

struct KLineQuery

{

time\_t szStartTime; //

time\_t szEndTime; //

\_InstrumentID InstrumentID; //合约代码

EnumPhrase nCycle; //K线周期

int nDataType; //1-K线数据，-分钟线数据

};

**5.6波动率数据结构**

struct Volatility

{

///交易日

TThostFtdcDateType TradingDay;

///合约代码

InstrumentIDType InstrumentID;

///20天波动率

TThostFtdcPriceType VolatilityTwenty;

///60天波动率

TThostFtdcPriceType VolatilitySixty;

};

**6错误类型**

#define CF\_ERROR\_COMMON\_BASE 0x0000 //公用故障

#define CF\_ERROR\_LOGIN\_BASE 0x0050 //登陆错误

#define CF\_ERROR\_INSTRUMENT\_BASE 0x0060 //合约错误

#define CF\_ERROR\_COMMON\_INPUT\_PARAM MAKE\_ERROR\_CODE(CF\_ERROR\_COMMON\_BASE,0x0001) //输入错误

#define CF\_ERROR\_LOGIN\_PASS\_INVALIDATE MAKE\_ERROR\_CODE(CF\_ERROR\_LOGIN\_BASE,0x0001) //登陆是否密码错误

#define CF\_ERROR\_LOGIN\_USER\_NOT\_EXIST MAKE\_ERROR\_CODE(CF\_ERROR\_LOGIN\_BASE,0x0003) //不存在该用户0

#define CF\_ERROR\_LOGIN\_QUOTSERVER\_ERROR MAKE\_ERROR\_CODE(CF\_ERROR\_LOGIN\_BASE,0x0010) //用户未登录行情服务器

#define CF\_ERROR\_NO\_INSTRUMENTS MAKE\_ERROR\_CODE(CF\_ERROR\_INSTRUMENT\_BASE,0x0001) //没有合约