**Falcon芝加哥商品交易所全球电子交易直接市场接入系统**

**Falcon CME Globex Ilink DMA System**

**API 使用说明**

|  |  |
| --- | --- |
| 日期 | 2018/4/22 |
| API版本 | 1.0.1 |
| 状态 | 初稿 |

1 文档变更日志 3

2 系统介绍 4

2.1. API架构 4

3 体系结构 5

3.1.系统结构 5

3.2.依赖库 5

4 开发接口 5

4.1. 系统参数设置 5

4.2. 系统实例化 7

4.3. start方法 7

4.4. stop方法 7

4.5. setCMESessionClient方法 8

4.6. registerOrderMgr方法 8

4.7. unregisterOrderMgr方法 8

4.8. isSessionLoggedOn方法 8

4.9. sendNewOrderSingle方法 9

4.10. sendOrderCancelRequest方法 11

4.11. sendOrderCancelReplaceRequest方法 12

4.12. sendOrderStatusRequest方法 14

4.13. sendOrderMassActionRequest方法 15

4.14. onMessage方法（回调函数） 16

4.15. onMessage方法（回调函数） 16

4.16. onMessage方法（回调函数） 17

4.17. onMessage方法（回调函数） 17

4.18. onMessage方法（回调函数） 17

5 开发示例 18

5.1. 系统参数设置 18

5.2. 系统实例化及连接 19

5.3. 下单 19

5.4. 订单管理模块（需完整继承CMEOrderInterface接口） 20

5.5. onMessage方法回调（需完整继承CMEOrderInterface接口） 20

# 文档变更日志

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| API版本 | 日期 | 作者 | 描述 | 备注 |
| 1.0.1 | 2017/11/29 |  | 初稿 |  |
| 1.0.1 | 2018/1/10 |  | 修改部分开发示例 |  |
| 1.0.1 | 2018/2/4 |  | 修改Oder Mass Action Request/Report相应部分 |  |
| 1.0.1 | 2018/2/10 |  | 修改部分接口和开发示例 |  |
| 1.0.1 | 2018/4/22 |  | 增加Secure Logon部分 |  |

# 系统介绍

## 2.1. API架构

Falcon芝加哥商品交易所全球电子交易直接市场接入系统（Falcon CME Globex ILink Direct Market Access System）是基于FIX协议的直接市场接入（DMA）系统。Falcon CME实现了与芝加哥商品交易所（CME）全球电子交易系统（Globex）的双向连接。

该系统 API 功能包括登录登出CME Globex 系统，支持对CME交易的期货和期货期权等金融衍生品进行下单，更改订单，撤销订单, 订单查询等操作。另外该系统还支持做市商报价（quote）功能如报价请求，大规模报价，取消报价，用户定义差价产品创建等功能。

该 API 包含以下文件:

|  |  |  |
| --- | --- | --- |
| 文件名 | 文件大小 | 描述 |
| include/CMEApplication.h | 14k | API头文件 |
| include/CMEGateWay.h | 6k | API头文件 |
| include/CMEOrderInterface.h | 1k | API头文件 |
| include/Log.h | 2k | API头文件 |
| include/Logger.h | 1k | API头文件 |
| libFalcon.so | 8k | API动态链接库 |
| config/CMEiLink.ini | 1k | CME Ilink quickfix 参数设置示例 |
| config/FIX42.xml | 13k | FIX4.2 XML定义 |

# 体系结构

## 3.1.系统结构

客户系统

Falcon CME Globex ILink

DMA API

CME Globex ILink Gateway

API call

FIX msg

## 3.2.依赖库

|  |  |
| --- | --- |
| 库名 | 最低版本 |
| libquickfix.so | 1.14.3 |
| libxml2.so | 2.9.1 |
| libz.so | 1.2.7 |
| libglibc.so | 2.17 |
| libpthread.so | 2.17 |
| libcryptopp.so | 6.1.0 |

# 开发接口

## 4.1. 系统参数设置

基于quickfix的CMEILink.ini参数设置示例，具体可进一步参考quickfix参数配置文档：http://www.quickfixengine.org/quickfix/doc/html/configuration.html

|  |  |  |
| --- | --- | --- |
| 参数名 | 值 | 所在session |
| ConnectionType | initiator | [DEFAULT] |
| ReconnectInterval |  | [DEFAULT] |
| SenderCompID |  | [DEFAULT] |
| FileLogPath |  | [DEFAULT] |
| FileLogBackupPath |  | [DEFAULT] |
| BeginString | FIX.4.2 | [SESSION] |
| TargetCompID | CME | [SESSION] |
| StartTime |  | [SESSION] |
| EndTime |  | [SESSION] |
| HeartBtInt | 30~60 | [SESSION] |
| SocketConnectPort |  | [SESSION] |
| SocketConnectHost |  | [SESSION] |
| DataDictionary |  | [SESSION] |
| SenderSubID |  | [SESSION] |
| TargetSubID |  | [SESSION] |
| SenderLocationID | CN | [SESSION] |
| ApplicationSystemName |  | [SESSION] |
| RawData | password(使用Secure Logon 则不需要此项) | [SESSION] |
| AccessKeyID | Secure Logon访问码 | [SESSION] |
| SecretKey | Secure Logon私钥 | [SESSION] |
| TradingSystemVersion |  | [SESSION] |
| ApplicationSystemVendor |  | [SESSION] |
| UseDataDictionary | N | [SESSION] |
| ValidateUserDefinedFields | N | [SESSION] |
| FileStorePath |  | [SESSION] |

## 4.2. 系统实例化

实例化Falcon CME ILink DMA API是通过参数配置文件来实例化的。具体参见开发示例代码。

从2018年4月底开始，CME将将强制使用Secure Logon方法进行Ilink的logon。在Ilink配置文件CMEILink.ini则需要设置相应的密码对，即Access Key ID 和Secret Key，而不在需要设置原始密码（RawData）。该密码对通过crypto++库进行加密。

|  |  |
| --- | --- |
| AccessKeyID | Secure Logon访问码 |
| SecretKey | Secure Logon私钥 |

## 4.3. start方法

|  |  |
| --- | --- |
| 函数原型 | bool start() |
| 参数 | 无 |
| 返回值 | bool值，连接发起成功返回true，否则返回false |

## 4.4. stop方法

|  |  |
| --- | --- |
| 函数原型 | bool stop() |
| 参数 | 无 |
| 返回值 | bool值，断开连接发起成功返回true，否则返回false |

## 4.5. setCMESessionClient方法

|  |  |
| --- | --- |
| 函数原型 | bool setCMESessionClient(CMEApplication\* cmeSessionClient, std::string targetCompID); |
| 参数 | cmeSessionClient：CME会话客户端实例指针  targetCompID： 连接对应方ID |
| 返回值 | bool值，设置成功则返回true，否则返回false |

## 4.6. registerOrderMgr方法

|  |  |
| --- | --- |
| 函数原型 | bool registerOrderMgr(CMEOrderInterface\* orderMgr) |
| 参数 | orderMgr：订单管理器（实现了CME订单接口） |
| 返回值 | bool值，注册成功则返回true，否则返回false |

## 4.7. unregisterOrderMgr方法

|  |  |
| --- | --- |
| 函数原型 | bool unregisterOrderMgr(CMEOrderInterface\* orderMgr) |
| 参数 | orderMgr：订单管理器（实现了CME订单接口） |
| 返回值 | bool值，取消成功则返回true，否则返回false |

## 4.8. isSessionLoggedOn方法

|  |  |
| --- | --- |
| 函数原型 | bool isSessionLoggedOn(); |
| 参数 | 无 |
| 返回值 | bool值，会话已经连接则返回true，否则返回false |

## 4.9. sendNewOrderSingle方法

|  |  |
| --- | --- |
| 函数原型 | bool sendNewOrderSingle( std::string account,  std::string clOrdID,  std::string custOrderHandlingInst,  int32\_t orderQty,  char ordType,  double price,  char side,  char timeInForce,  double stopPx,  std::string securityDesc,  int32\_t minQty,  std::string securityType,  int32\_t customerOrFirm,  int32\_t maxShow,  std::string expireDate,  bool manualOrderIndicator,  std::string giveUpFirm,  std::string cmtaGiveupCD,  std::string allocAccount ); |
| 参数 | account： 客户账户  clOrdID： 客户指定单号  custOrderHandlingInst：落单来源，详见CME ILink解释  orderQty： 订单数量  ordType：订单类型  price：订单价格（限价单）  side：订单买入卖出方向  timeInForce： 订单委托模式  stopPx：止损价格（仅限止损单，其他设0）  securityDesc：合约代码  minQty：最小成交量  securityType：合约类型  customerOrFirm：订单类型：客户/公司  maxShow：最大显示数量（Iceberg Order）  expireDate：订单有效期（仅限GTD单有效）  manualOrderIndicator：是否手动落单  giveUpFirm：Giveup公司  cmtaGiveupCD：Giveup类型，详见CME Ilink解释  allocAccount：Giveup账号 |
| 返回值 | 订单发送成功返回true，否则返回false |

## 4.10. sendOrderCancelRequest方法

|  |  |
| --- | --- |
| 函数原型 | bool sendOrderCancelRequest( std::string account,  std::string clOrdID,  std::string orderID,  std::string origClOrdID,  char side,  std::string securityDesc,  std::string securityType,  std::string correlationClOrdID,  bool manualOrderIndicator ); |
| 参数 | account： 客户账户  clOrdID： 客户指定单号  orderID：CME单号  origClOrdID：客户原单号  side：订单买入卖出方向  securityDesc：合约代码  securityType：合约类型  correlationClOrdID：客户上次更新单号  manualOrderIndicator：是否手动落单 |
| 返回值 | 订单取消发起成功返回true，否则返回false |

## 4.11. sendOrderCancelReplaceRequest方法

|  |  |
| --- | --- |
| 函数原型 | bool sendOrderCancelReplaceRequest(std::string account,  std::string clOrdID,  std::string orderID,  int32\_t orderQty,  std::string custOrderHandlingInst,  char ordType,  std::string origClOrdID,  double price,  char side,  char timeInForce,  bool manualOrderIndicator,  double stopPx,  std::string securityDesc,  int32\_t minQty,  std::string securityType,  int32\_t customerOrFirm,  int32\_t maxShow,  std::string expireDate,  std::string correlationClOrdID,  char IFMFlag,  std::string giveUpFirm,  std::string cmtaGiveupCD,  std::string allocAccount ); |
| 参数 | account： 客户账户  clOrdID： 客户指定单号  orderID：CME单号  orderQty： 订单数量  custOrderHandlingInst：落单来源，详见CME ILink解释  ordType：订单类型  origClOrdID：客户原单号  price：订单价格（限价单）  side：订单买入卖出方向  timeInForce： 订单委托模式  manualOrderIndicator：是否手动落单  stopPx：止损价格（仅限止损单，其他设0）  securityDesc：合约代码  minQty：最小成交量  securityType：合约类型  customerOrFirm：订单类型：客户/公司  maxShow：最大显示数量（Iceberg Order）  expireDate：订单有效期（仅限GTD单有效）  correlationClOrdID：客户上次更新单号  IFMFlag：In flight mitigation 指示，详见CME ILink解释  giveUpFirm：Giveup公司  cmtaGiveupCD：Giveup类型，详见CME Ilink解释  allocAccount：Giveup账号 |
| 返回值 | 订单更改发起成功返回true，否则返回false |

## 4.12. sendOrderStatusRequest方法

|  |  |
| --- | --- |
| 函数原型 | bool sendOrderStatusRequest( std::string clOrdID,  std::string orderID,  char side,  std::string securityDesc,  std::string securityType,  std::string correlationClOrdID,  bool manualOrderIndicator  ); |
| 参数 | clOrdID： 客户指定单号  orderID：CME单号  side：订单买入卖出方向  securityDesc：合约代码  securityType：合约类型  correlationClOrdID：客户上次更新单号  manualOrderIndicator：是否手动落单 |
| 返回值 | 订单状态请求发起成功返回true，否则返回false |

## 4.13. sendOrderMassActionRequest方法

|  |  |
| --- | --- |
| 函数原型 | bool sendOrderMassActionRequest(std::string clOrdID,  int32\_t massActionScope,  int32\_t marketSegmentID,  std::string symbol,  std::string securityDesc,  int32\_t massCancelRequestType,  std::string account,  char side,  char ordType,  char timeInForce,  bool manualOrderIndicator); |
| 参数 | clOrdID： 客户指定单号  massActionScope：批量行动范围  marketSegmentID：市场分段ID  symbol：产品代码  securityDesc：合约代码  massCancelRequestType：批量取消请求类型  account：账户  side：订单买入卖出方向  ordType：订单类型  timeInForce：订单委托模式  manualOrderIndicator：是否手动落单 |
| 返回值 | 订单状态请求发起成功返回true，否则返回false |

## 4.14. onMessage方法（回调函数）

|  |  |
| --- | --- |
| 函数原型 | void onMessage (const FIX42::BusinessMessageReject& businessMessageReject) |
| 参数 | businessMessageReject：业务消息拒绝 |
| 返回值 | 无 |

## 4.15. onMessage方法（回调函数）

|  |  |
| --- | --- |
| 函数原型 | void onMessage(const FIX42::Reject& reject); |
| 参数 | reject：消息拒绝 |
| 返回值 | 无 |

## 4.16. onMessage方法（回调函数）

|  |  |
| --- | --- |
| 函数原型 | void onMessage(const FIX42::ExecutionReport& executionReport) |
| 参数 | executionReport：执行报告消息 |
| 返回值 | 无 |

## 4.17. onMessage方法（回调函数）

|  |  |
| --- | --- |
| 函数原型 | void onMessage(const FIX42::OrderCancelReject& orderCancelReject) |
| 参数 | orderCancelReject：订单取消拒绝消息 |
| 返回值 | 无 |

## 4.18. onMessage方法（回调函数）

|  |  |
| --- | --- |
| 函数原型 | void onMessage(FIX50SP2::OrderMassActionReport& orderMassActionReport) |
| 参数 | orderMassActionReport：批量订单取消报告 |
| 返回值 | 无 |

# 开发示例

## 5.1. 系统参数设置

|  |
| --- |
| *## http://www.quickfixengine.org/quickfix/doc/html/configuration.html # default settings for sessions* **[DEFAULT] ConnectionType**=**initiator ReconnectInterval**=**60 SenderCompID**=**4V5004N** *#file log* **FileLogPath**=**./fixlog FileLogBackupPath**=**./fixlog** *# session definition* **[SESSION]** *# inherit ConnectionType, ReconnectInterval and SenderCompID from default* **BeginString**=**FIX.4.2 TargetCompID**=**CME StartTime**=**06:30:00 EndTime**=**23:30:00 HeartBtInt**=**30 SocketConnectPort**=**26272 SocketConnectHost**=**69.50.112.143 DataDictionary**=**../config/FIX42.xml** *#CME header specific* **SenderSubID**=**Falcon001 TargetSubID**=**G SenderLocationID**=**CN** *#Logon specific* **ApplicationSystemName**=**SunX RawData**=**Password TradingSystemVersion**=**1.0 ApplicationSystemVendor**=**SunX UseDataDictionary**=**Y ValidateUserDefinedFields**=**N** *#STORE* **FileStorePath**=**./sessionStore** |

## 5.2. 系统实例化及连接

|  |
| --- |
| #include **"include/CMEGateWay.h"** #include **"include/CMEApplication.h"**  **using namespace** falcon::cme; CMEApplication cmeApplication(**"../config/CMEiLink.ini"**); CMEGateWay cmeGateWay; cmeGateWay.setCMESessionClient(&cmeApplication, **"CME"**); cmeGateWay.start(); cmeGateWay.stop(); std::cout << **"Hello, World! 88"** << std::endl; |

## 5.3. 下单

|  |
| --- |
| **auto** res = cmeGateWay.sendNewOrderSingle(  account,  clOrdID,  custOrderHandlingInst,  orderQty,  ordType,  price,  side,  timeInForce,  stopPx,  securityDesc,  minQty,  securityType,  customerOrFirm,  maxShow,  expireDate,  manualOrderIndicator,  giveUpFirm,  cmtaGiveupCD,  allocAccount  ); |

## 5.4. 订单管理模块（需完整继承CMEOrderInterface接口）

|  |
| --- |
| CMEGateWay cmeGateWay;  MyOrderMgr myOrderMgr;//Implements CMEOrderInterface  if( cmeGateWay.registerOrderMgr(&myOrderMgr) ){  std::cout << “**Registering Order Manager successful”** << std::endl;  }  else {  std::cout << “**Registering Order Manager failed”** << std::endl;  }  if( cmeGateWay.unregisterOrderMgr(&myOrderMgr) ){  std::cout << “**UnRegistering Order Manager successful”** << std::endl;  }  else {  std::cout << “**UnRegistering Order Manager failed”** << std::endl;  } |

## 5.5. onMessage方法回调（需完整继承CMEOrderInterface接口）

|  |
| --- |
| **void** onMessage(**const** FIX42::ExecutionReport& executionReport){  **auto** execType = executionReport.getField(FIX::FIELD::ExecType);  if(execType == “8”) { // Execution Report - Fill Notice (tag 35-MsgType=8, tag 39-OrdStatus=1 or 2) message is sent upon fill or partial fill of client order.  **auto** cumQty = executionReport.getField(FIX:: FIELD::CumQty);  **auto** clOrdID = executionReport.getField(FIX::FIELD::ClOrdID);  std::cout << **"Total traded qty on "** << clOrdID << **" is "** << cumQty << std::endl;  }  }  **void** onMessage(**const** FIX42::BusinessMessageReject& businessMessageReject) {  **LOG\_ERR**(businessMessageReject.toString()); }  **void** onMessage(**const** FIX42::Reject& reject) {  **LOG\_ERR**(reject.toString()); } |