**NSignal System**

对来自NxCore的Quote数据、Trade数据以及Category的OI数据进行处理并存储。

1. **NxCoreReader**

从NxCore tape文件中提取出所有的期权交易（Option Trade）记录，并为每条记录保存5秒前且离5秒最近的Ask Quote和Bid Quote，以及上一条交易的成交价。

**1.1 用法（Usage）**

NxCoreReader [-t |Tape file path] [-r | Output file path]

* -t：Real-time，从NxCoreAccess接受实时数据。不需要指定[Tape file path]。
* Tape file path：需要处理的NxCore Tape文件的文件名。当使用-t参数时不要指定该参数。
* -r：Remote，输出结果至远程接受服务器（NSignalReceiver Server）。不需要指定[Output file path]，需要配置文件（nsignal.properties）。
* Output file path：保存输出的所有期权交易记录的结果的文件名。当使用-t参数时不要指定该参数。当不使用-t参数且省略该参数的，将保存结果到文件名为Tape file path加.trade后缀的文件；如使用-t参数，则该文件名为以当前日期为时间戳按照NxCore Tape文件名规则生成的文件名。

**1.2 配置文件（Property）**

当前目录下名为nsignal.properties的文件。配置项格式为key=value，配置项间使用换行分隔。

**配置项：**

* server.url：远程接受服务器的URL，即NsignalReceiver服务器的URL。
* server.port：远程接受服务器的端口号，即NsignalReceiver服务器的端口号。

**1.3 模块结构（Module Structure）**

* OptionTradeInfo：Model，保存了一种Option Symbol的所有数据。在NxCore读取Tape文件记录过程中，以当前读取的记录的发生时间（比如Trade成交的时间、Quote报价发出的时间）为目标时间，以当前记录的期权为目标期权，该类保存了所有目标时间前5秒内和一条离前5秒最近的一条5秒外的关于目标期权的Ask Quote记录和Bid Quote记录，以及目标期权的上一条Trade记录。
* NxCoreOutputStream：结果文件的输出流。两个实现类：
* NxCoreOutputFileStream Class：将结果文件保存到本地的输出流。
* NxCoreOutputTCPStream Class: 将结果文件发送至NSignalReceiver服务器。需要从配置文件中读取服务器的URL和Port。
* NxCoreReader：处理每条从NxCore Tape文件中读取的记录。该接口有7个方法，分别对应7种NxCoreMessage类型。目前只实现了两个方法，分别处理NxMSG\_TRADE记录和NxMSG\_EXGQUOTE记录：
* ProcessTradeMsg Method：处理Trade记录。
* ProcessQuoteMsg Method：处理Quote记录。

**1.5 输出格式**

CSV格式。

首项为记录类型：

1：option trade

2:：open interest(The day before yesterday and yesterday oi)

1. **NSignalReceiver**

读取NxCoreReader处理的结果数据并再次对Trade数据记录进行交易方向判断（Inferring Trade Direction）、交易合并（Conbine Trade Leg）、大交易标志位设置（Big Trade Flag）处理，最终将处理结果写到数据库。

多线程处理，一个Receiver可以接受多个Reader的输入。

接受完一个Reader的输入后，再统一将一天的记录存入数据库中。

每次存数据库时会检查表是否存在，若不存在则创建表。

**2.1 用法（Usage）**

NSignalReceiver [-r| Trade File Path]

* -r: Remote。从远程接受Trade记录，即打开NSignalReceiver Server。需从nsignal.properties配置文件中读取Server要绑定的端口Port。
* Trade File Path：Trade记录文件路径。

**2.2 配置文件（Property）**

当前目录下名为nsignal.properties文件和log4j.properties文件。配置项格式为key=value，配置项间使用换行分隔。

* nsignal.properties配置项：
  + server.port：服务器的端口号，即NsignalReceiver服务器的端口号。
  + db.driverClassName：数据库驱动类名。
  + db.url：数据库的URL。
  + db.username：数据库用户名。
  + db.password：数据库密码。
* log4j.properties配置项：按照log4j标准配置。

**2.3 开发环境部署（Deploy Development Environment）**

在项目根目录下输入gradle eclipse命令或gradlew eclipse命令（服务器未安装gradle情况下），等待命令执行完成，即完成在eclipse下的开发环境部署。完成后用eclipse打开文件夹即可。

**2.4 运行部署（Deploy Runtime Environment）**

在项目根目录下输入gradle shadowJar命令或gradlew shadowJar命令（服务器未安装gradle情况下），等待命令执行完成，即完成在eclipse下的运行环境部署。独立的JAR包将生成在build\libs文件夹下。

**2.5 模块结构（Module Structure）**

* OptionTradeModel：保存Option Trade记录并提供解析功能。
* OptionTradeHandlerContext：Option Trade记录的处理环境，在该Context对Option Trade进行所有处理。每一个输入流对应一个Context。
* NSignalReceiver：完成对Option Trade记录的接受、处理以及保存至数据库。有两个实现类：
  + NSignalFileReceiver：从文件读取Option Trade记录并处理。需提供Option Trade File Path参数。
  + NsignalRemoteReceiver：启动Receiver Server，接受来自客户端的Option Trade记录并处理。需提供-r参数。
* OptionTradeService：提供OptionTradeModel相关的数据与数据库的交互服务。比如CRUD。
* OptionTradeMapper：Mybatis的Mapper文件，完成直接与数据库进行的操作。通常被Service调用。