SGXOrders

SGX Quest DT Order Capture

Related Documents:

RJE:-

- SGXOrders Design.pdf
- SGXOrders Technical Overview.pdf

SGX:-

- SGX Quest DT Drop Copy Technical Information 1 1 _Final_.pdf
- OMexExternal_API_SGX_va522.pdf
- SGX+Quest+DT+Upgrade+Technical+Information+-+V3+6.pdf

These are the most relevant SGX documents but there are others.

Other Information:

Please consult the RJE Web Site www.rje.com.au for the latest information on RJE products.

This Document:

SGXOrders – SGX Quest ST Orders Capture – details all issues relating to installation and operation of SGXOrders.

Revision:

17/06/2014 – Added Mass Cancellations, and additional fields for compliance purposes.
29/05/2012 – Initial Document – SGXOrders Rev 01_01.

1. Overview:

1.0 Program Operation:

There is a separate MCclickSGX component, similar to MCTradesAT.

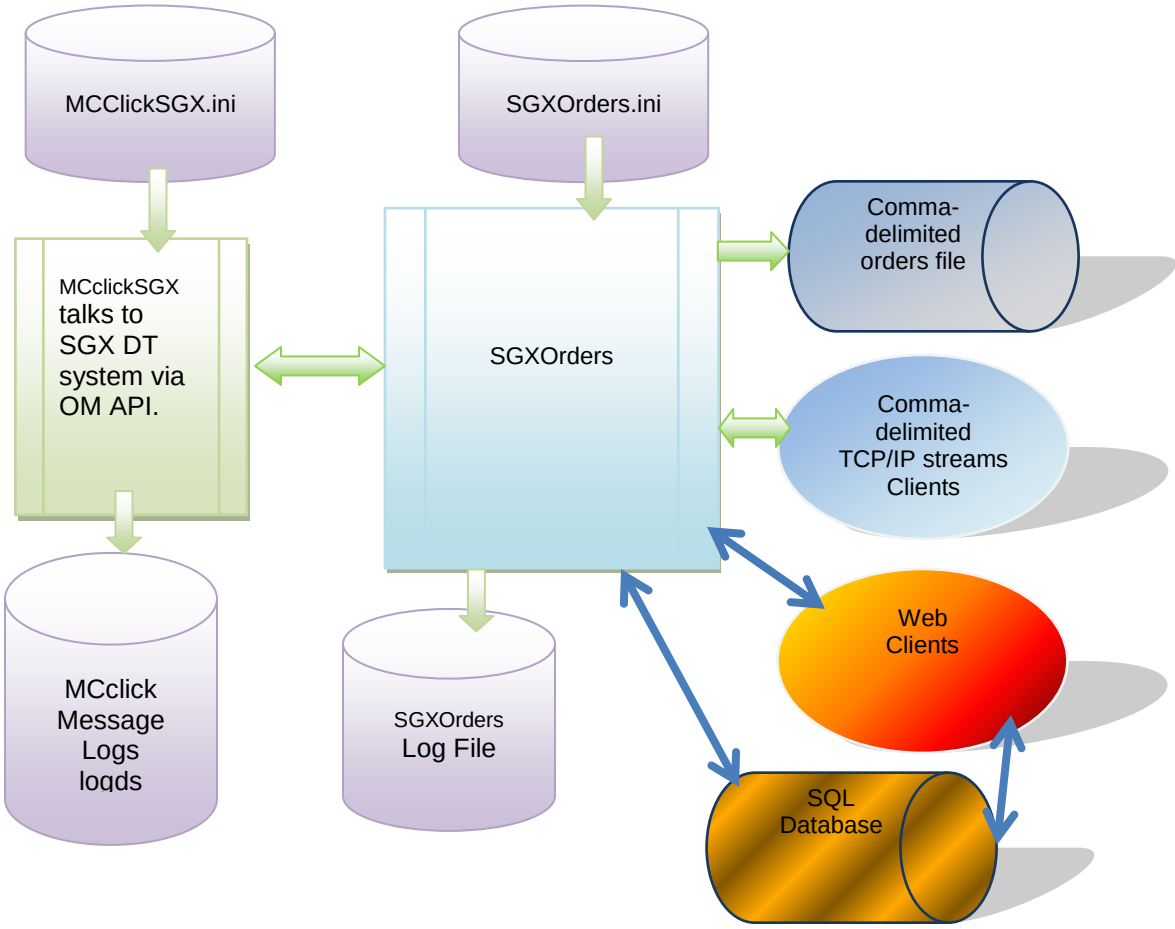


Figure 1. The SGXOrders Production System

1.1 SGXOrders:

The SGXOrders Program:-

- Connects to the Quest DT system via MCclickSGX
- Updates the SQL database with details of current orders.
- Performs order cancellations in the Quest DT system via MCclickSGX.
- Provides TCP/IP stream feeds of orders, transaction and trades information.

The configuration settings for the system are found in SGXOrders.ini

1.2 Order Cancellation:

SFEOrders supports cancellation of orders via:-

- Web Clients
- GUI Clients

More details can be found in:- 6. Command Clients – Order Cancellation:

1.3 SQL Database:

An optional SQL database can be configured to store orders and related information.

The SQL Database is required for Order Cancellation and for the Web Client.

More details can be found in :- 7. SQL Database:

1.4 Web Client:

A PHP based web client application can be used for view / cancel orders.
See :- 8. PHP – Web Client:

2. Daily Cycle:

SGXOrders can be run for multiple days; it shuts down and wakes up at a certain scheduled time each day.

Refer:- 3.2.6 Daily Cycle Parameters:

WAKE_TIME = time when program wakes up each morning, SHUT_TIME = time when the program shutdown (hibernation) occurs. This area functions as per existing RJE products.

3. Installation/Configuration:

3.1 Installation:

3.1.1 SGXOrders:

Simply install SGXorders as follows :-

<Install Directory> :- SGXOrders.exe, SGEOrders.ini
<Install Directory>:-au.com.mcx.dll, Npgsql.dll,
<Install Directory>/logs :- make a subdirectory for log files.

To run the program simply run SFEOrders.exe, provided the configuration in the .ini file is correct no other information is needed.

You must set the following parameters correctly:-

- 3.2.1 MCclickSGX Connection Parameters
- 3.2.2 Quest DT Logon Parameters
- 3.2.3 Broker List Parameter.

Note: If you which to run the program without a GUI refer:-3.2.11 Other parameters:

3.1.2 MCclickSGX:

Typically, this would be in a separate directory but could be on the same machine.

Simply copy all files as follows;
<Install Directory> :- MCclickSGX.exe, MCclickSGX.ini
<Install Directory>:- libeay32.dll, oapimtdll.dll, ssleay32.dll, zlib1.dll
<Install Directory>/logs :- make a subdirectory for log files.
<Install Directory>:- plogDTSGX.exe – utility for converting log file extracts to text.

3.2 SGXOrders Configuration:

All configuration parameters are stored in SGXOrders.ini

3.2.1 MCclickSGX Connection Parameters:

MC_SERVER_HOST=myhost
MC_SERVER_PORT=7001

MC_SERVER_HOST = Name of Server running MCclickSGX.
e.g. FIX_SERVER_HOST= myhost

MC_SERVER_PORT = Port to connect for MCclickSGX client connections.
e.g. FIX_SERVER_PORT=7001

This corresponds to a port setting in MCclickSGX.ini:-
CLIENTS_PORT =7001

Note: SGXOrders always connects to the Quest DT system via MCclickSGX.

3.2.2 Quest DT Logon Parameters:

SGX_USER_ID= Quest DT Logon user supplied by the SGX.
e.g. SGX_USER_ID=Z1389A1

SGX_PASSWORD= Quest DT user password initially supplied by the SGX.
e.g. SGX_PASSWORD=ABCDEF1205G463HIJKLM

SGX Quest DT user passwords can expire and currently must be changed every 60 days. SGXOrders will detect that a password has expired and will automatically change the password to one that is compliant with the SGX rules. The .ini file is updated with the new password; this is a full automated process and should not need manual intervention.

The following parameters can be set for automatic password handling:-
AUTO_PASSWORD_BASE=ABCDEF
CHANGE_PASSWORD_DAYS=10
CHANGE_PASSWORD_DAYS=n – change the password ‘n days’ before expiry.

3.2.3 Broker List Parameter:

You must define a list of broker numbers to be monitored by the SGXOrders application in this parameter. This setting ensures the SGXOrders program does MQn orders data queries for each broker in the list.

As well, the Quest DT system must be configured to correctly route orders and trades data for those brokers to the Drop Copy user id.

This should always be tested, both querying existing orders via Broker List at start up and receiving details for any new orders via BO5 broadcasts.

```
*****************************************************
* BROKER_LIST Data Processed for brokers in this list*
*****************************************************
BROKER_LIST=AB389,XX123
```

3.2.4 Order Types to Query:

* (Normal ACTIVE orders always processed)

Normal Active orders are always queried, it is possible to configure the system so it does not query other types of orders on start up. However, generally you should get all types of orders data as per the settings below.

```
INACTIVE_ORDERS=YES
STOP_ORDERS=YES
SESSION_ORDERS=YES
```

3.2.5 Trades Data:

ABN is already getting trades data from the SGXClear environment via MCTradesSGX.

Initially, we recommend turning of the gathering of trades data as per the setting below:-
TRADES=NO

This is because there will be an issue with the dated .trades file if the program is stopped/restarted between 12pm and 2am SGX time. (We will address this in the next version.)

Some more development and testing may be required to adequately ensure we can robustly handle all aspects of 24x7 operation while gathering trades.

Once this is resolved SGXOrders should be able to provide a reliable feed of T+1 trades data.

Some consideration was given to matching orders and trades to determine if an order has traded out while the program is offline. However, this issue is more complex than is first apparent, detailed consideration is required before attempting this type of matching.

3.2.6 TCP/IP Feed Parameters:

This is the TCP/IP port that applications can connect to receive a feed of orders data.

ORDERS_PORT = TCP/IP port for all Orders.
e.g. ORDERS_PORT=12008

This is the primary feed produced by this application, but it also produces a feed of BO5 transactions and trades, this can be configured as follows:-

e.g. ORD_TRANS_PORT=12012
e.g. TRADES_PORT=12014

3.2.7 Command Client Parameters – Order Cancellation.

e.g. COMMAND_PORT= TCP port command clients must connect to issue cancel requests.
COMMAND_PORT=12010

3.2.8 SQL Database Parameters

The use of an SQL Database is optional but the Web Client facility will not work properly if it is not enabled.

All SQL Database access occurs in a separate the thread it should not affect the performance of the rest of the SGXOrders application.

If we detect that database updating cannot keep up with the rate the SGX is sending data, the program can be enhanced with multiple database updating threads.

SQL_DATABASE_NAME=Name of the database to access. The presence of this parameter turns on database processing. All tables and functions mentioned in SQL_DATABASE_NAME=sfe

SQL_DATABASE_SERVER=The machine which is the PostgreSQL database server.
SQL_DATABASE_SERVER=rjelinuxlap

SQL_DATABASE_PORT=Port for the PostgreSQL database.
SQL_DATABASE_PORT=5432

SQL_USER_ID=PostgreSQL database user.
SQL_USER_ID=sfe

SQL_PASSWORD= PostgreSQL database user password*
SQL_PASSWORD=rjexxxxxx

* There may be a better way of controlling access to the PostgreSQL database. We have chosen the user/password model to simplify the initial development.

3.2.9 Logging Parameters:

APP_LOG_FILE = file base for application log, a new log is taken each run; the application log includes the current date and time.
e.g. APP_LOG_FILE= SFEOrders
filename= SFEOrders_20080429_150113.log.

APP_LOG_DIRECTORY=Directory where the application log is stored.
APP_LOG_DIRECTORY=logs

LOGGING_LEVEL= Set the level of application message logging; can turn on additional diagnostic messages.
LOGGING_LEVEL=9

3.2.10 Daily Cycle Parameters:

Refer:- 2. Daily Cycle:

WAKE_TIME = time when program wakes up each morning (hour:min), default 07:00.
e.g. WAKE_TIME=07:30

SHUT_TIME = time when the program shutdown (hibernation) occurs (hour:min) default 23:30.
e.g. SHUT_TIME=21:00

3.2.11 Other parameters:

NO_GUI=YES – Specify this value to run without a GUI, e.g. as a Windows NT service.

3.3 MCclickSGX Configuration:

These are similar to parameter settings for similar components e.g. MCclickASX.ini, MCSecurSGX.ini.

3.3.1 Client (SGXOrders) Connection Parameters:

As mentioned earlier SGXOrders must make a TCP/IP connection to MCclickSGX which in turn talks to the Quest DT system via OM API over a TCP/IP transport.

The corresponding settings for SGXOrders are:- 3.2.1 MCclickSGX Connection Parameters:

CLIENTS_PORT = TCP/IP port that clients (in this case SGXOrders) connect to.
CLIENTS_PORT =7001

3.3.2 Quest DT Connection Parameters:

SGX_CLICK_GATEWAY = Quest DT Gateway to connect to – supplied by the SGX.
e.g. SGX_CLICK_GATEWAY =questDT
SGX_CLICK_PORT =Port to use on Quest DT Gateway – supplied by the SGX.
SGX_CLICK_PORT =21024
*questDT=10.37.253.177

3.3.3 Quest DT Connection Options:

These control the use of encryption or compression on the OM API link.

The SGX will advise if Compression or Encryption is to be used and our settings must match theirs.

e.g. OMNIAPI_COMPRESS =NO (YES)
e.g. OMNIAPI_ENCRYPT =NO (YES)

3.3.4 Logging Options:

These control the amount of information being logged.

DIAGNOSTIC_LEVEL =1 - controls how much information is logged in text diagnostics messages. Higher number mean more information is logged. Just use the default value unless otherwise instructed by RJE support personnel.

LOG_MESSAGES =A - controls amount of info logged
A=All, C=Client, X=Exchange, T=Text, W=Warning, E=Error, N=None
Can specific a single type or multiples (e.g. C+T)
A=C+X+T
T -> All Text messages includes warnings & errors.
W - > Warnings includes errors
Lowest setting is E -> Error messages only.
Error messages are always logged if logging is enabled.
N -> Turns logging off
Log files can get big quickly but logged info gives us the ability to diagnose problems.

LOG_FILTER - further control on amount of info logged for Exchange & MC API messages
D = log deals
5 = log BO5's
When the filter is set no other query response/broadcast message types are logged.
By default the filter is not set and all message types are logged.
Example – the recommended settings for MCTrades are:-

- LOG_MESSAGES =C+W
- LOG_FILTER =D5

3.3.5 Other Configuration Options:

TCPIP_CONNECTIONS =n Allow ‘n’ concurrent TCP/IP connect attempts (backlog), Default = 5.

BCAST_POLL_RATE=n - Broadcast Poll Rate - Polls per second (default = 10)
The ASX may instruct users to set the poll rate to a particular value.

BCAST_HBEAT_POLL=n – Special poll rate for order entry apps not subscribed to any broadcasts (default = 1).

QUIT_DELAY=’n’ milliseconds - time to wait before closing client socket after sending quit response. Default = 200 M/S. (You should not need to use this.)

Performance Statistics:

BCAST_STATS=n - Output Broadcast Stats every 'n' seconds - zero default = no stats

4. Password Changing:

SGX Quest DT user passwords can expire and currently must be changed every 60 days. SGXOrders will detect that a password has expired and will automatically change the password to one that is compliant with the SGX rules. The .ini file is updated with the new password; this is a full automated process and should not need manual intervention.

See 3.2.2 Quest DT Logon Parameters: for more details.

5. Recovery Strategy:

Each order transaction has a unique key:- OrderID+Symbol+Side

This allows for OMX block order transactions, although these are not in current use in the Quest DT system. There are block transactions for Market Maker quotes, but quotes are currently not stored in the SQL database.

When the link to Quest DT drops out, SGXOrders performs the following recovery sequence:-

After some consideration the easiest way for us to handle this is:-

1. Wipe all order data in memory.
2. Close all client connections (kicking off, connected applications).
3. Market all order in the database as not current.
4. Open a new.orders file.
5. Recover the link then re-download all order data from Quest DT.

So each session starts with an orders snapshot which is then updated via BO5 broadcasts.

In other words the only difference between a between the Quest DT link dropping out and a clean restart is that the program keeps running and attempts to recover.

The other advantage of this scenario is that each restart is a clean restart if things seem to be going wrong you can just restart.

If SGXOrders detects any serious problem, it will just stop running.

6. Command Clients – Order Cancellation:

6.1 Order Cancellation Overview:

SFEOrders supports cancellation of orders via:-

- Web Clients – The final production system.
- GUI Clients – TestClient.exe – temporary fallback option (allows cancellation of groups of orders).

Configuration settings are found in:- 3.2.7 Command Client Parameters – Order Cancellation.

6.2 Supported Cancellation Types:

Cancel requests and sent to the commands port us plain text strings.

```
CxlType->CxlActive==A|CxlInactive==I|CxlStop==S
```

Delete a single order:-

```
CANCEL_REQUEST|CxlType={cancel type)|[OrderID={order id)
                   [Security={security}]|Side={bid/ask}|
                     [Country={country}]
                  |Firm={firm id}|[Trader={traderid}] <- trading code.
```

Delete a group of orders:-

```
CANCEL_REQUEST|CxlType={cancel type)|
        [Security={security}]/Series*(1)|Side={bid/ask/*}
        [Country={country}]
        |Firm={firm id}|[Trader={trader id}]|  <- whose/trading code.
         [Account={account}] <- whose
        [Customer={cust info}]|[Exchange={exchange info}
```

Note: The absence of an order_id indicates a group cancel request.

There is also a simplified type of cancellation intended for use by the web client.

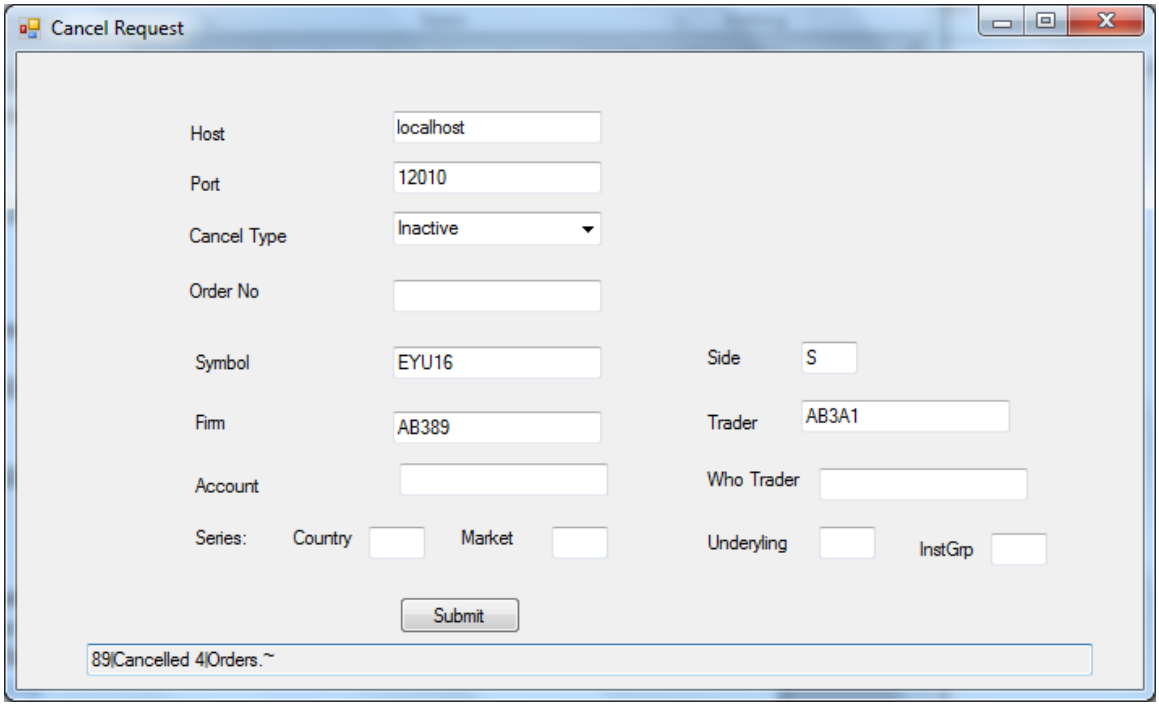
```
CANCEL_REQUEST| OrderID={order id)
```

In that mode SGXOrders will look up the order in memory and fill in any other fields required by the Quest DT system.

6.3 Canceling Groups of Orders:

It is possible to cancel groups of orders by following the rules of the MO4 transaction as documented in the relevant Quest DT manual.

As example of how to do this type of cancellation can be seen on the screen below.



SGXOrders now has specific enhancements to automate the Mass Cancellation of all firm orders as detailed in:- 10. Mass Cancellation:

7. SQL Database:

The database is common to a number of order pulling applications which gather orders information for use by the web client application.

The SGXOrders program uses the “npgsql” .net data provider for PostgreSQL.

The SGXOrders program typically calls ProgreSQL Functions (Stored Procedures) for database access and updating. This approach was chose as we believe it will deliver the best performance.

7.1 SQL Database Tables:

7.1.1 Table - system

Standard Web Client table – used by SGXOrders.

7.1.2 Table – system_state

Standard Web Client table – used by SGXOrders.

7.1.3 Table – sgx_orders

This is the main table of interest to Web Clients and other application which wish to display data.

When the field order_active=’Y’ the order is an active order which is a candidate for cancellation.

When the field order_active=’I’ the order is an inactive order which is a candidate for cancellation.

As orders trade out or are cancelled order_active is set to ‘N’.

On a restart/recovery order_active is set to ‘U’. After recovery any orders which remains as order_active=’U’ must have been deleted or traded out while the system was offline.

We are keeping orders information as it may be useful, if we continue with this approach are archiving process must be developed.

Function :- func_sgx_update_order() – Updates the orders table for each execution report transaction. Note: It is likely that this function will be specific to the SFE other systems will use the same database table but may have a slightly different update function.

Function :- func_sgx_wipe_current_orders() – Sets order_active to 'U' on restart/recovery. Can be changed to delete orders if required,

```
CREATE TABLE sgx_orders
(
  id bigserial NOT NULL,
  system_id uuid NOT NULL,
  order_id character varying(50) NOT NULL,
  message_no integer,
  order_active character(1),
  order_status character varying(4),
  order_type character varying(4),
  order_book_class character varying(4),
  ord_change_reason character varying(4),
  order_ref character varying(50),
....
```

7.1.4 Table – sgx_order_cancel_result

This table is updated with the results of each order cancel request. The intention is that this table will be a long term 'audit trail' of cancellation activity.

Function :- func_sgx_update_order_cancel() - updates this table and the trans table with the results of each order cancel request.

7.2 SQL Database Parameters:

See 3.2.8 SQL Database Parameters

7.3 npgsql files:

The following files should reside in the same directory as SFEOrders.exe:-

```
      07/07/2007  12:09 AM           282,624 Mono.Security.dll
      28/09/2011  08:55 PM           365,568 Npgsql.dll
```

These files are the "npgsql" .net data provider for PostgreSQL.

7.4 SQL Script files:

The following files create database tables:-

```
28/05/2012  11:59 AM    2,058 sgx_create_orders.sql
03/05/2012  10:13 AM    999   sgx_create_order_cancel_result.sql
```

The following files create database functions:-

```
28/05/2012  12:02 PM             8,636 func_sgx_update_order.sql
03/05/2012  10:14 AM             4,715 func_sgx_update_order_cancel.sql
03/05/2012  03:04 PM               675 func_sgx_wipe_current_orders.sql
```

8. PHP – Web Client:

This is documented elsewhere.

9. Fields Mappings SGXOrders <-> Quest DT:

SGX Orders Field Name	Quest DT Field Name
firm_id	trading_code.ex_customer_s
crader_id	trading_code.user_id_s
crder_id	order_number_u
cl_order_id	exchange_info_s
exec_id	(last trade.trade_number)
exec_trans_type	<No Value>
order_status	order_state_u
order_bos_pos	ob_position_u
account	ex_client_s
exchange_code	<Always “SGX”>
symbol	Series Name – (series.ins_id_s)
Side	bid_or_ask_c
order_qty	mp_quantity_i
Price	premium_i
last_shares	(last trade.quantity)
cum_quantity	<No Value>
transact_time	<No Value>
Text	customer_info_s
order_type	order_type_c
expire_time	time_validity_n.duration
commodity	<No Value>
Month	<No Value>
Year	<No Value>
ob_class	<RJE Derived field>
block	block_n
change_reason	change_reason_c
combo_mark	combo_mark_c
display_qty	display_quantity_i
exch_ord_type	exch_order_type_n
ex_state	ext_t_state_c
give_up_member	give_up_member
limit_premium	Stop orders – limit_premium_i
max_rand_hidden	mp_max_random_hidden_i
oc_request	open_close_request_c
order_no_bin	order_number_u

Party	Party
sequence_no	sequence_number_u
Series	Series
stop_condition	stop_condition_c
stop_series	Stop Orders – stop_series
stop_series_name	Stop Orders -(stop_series.ins_id_s)
time_val_type	time_validity_n.type
total_volume	total_volume_i
trading_code	trading_code
transaction_no	Transaction_number_n
user_code	ex_user_code
trans_ack	(Txstat from last BO5 transaction)
Active	<RJE Derived field>
unique_key	<RJE Derived field>
Created	timestamp_in
Modified	execution_timestamp
created_utc	timestamp_in
modified_utc	execution_timestamp
txn_type	<RJE Derived field>
omx_txn_code	e.g. BO5
omx_click_code	e.g. MO31 (transaction_no)

10. Mass Cancellation:

10.1 New Program Parameters:

*---- SGX Mass Delete - new Configuration settings ------
*---- All of these default to YES - only disable in the event of problems. ----
*DELETE_STOP_ORDERS=NO
*SGX_MASS_DELETIONS=NO
*EXCEPTION_HANDLING=NO

SGX_MASS_DELETIONS – Setting this parameter to NO will turn off the Mass Deletions facility, it is most unlikely that we would ever need to do that.

DELETE_STOP_ORDERS – By default deleting all firm orders will include normal active orders and stop orders. The reason for that is that when activated stop orders become normal orders and could trade. If this is set to NO you can delete stop order via the following additional request types:-

```
CxlFirmStop = "X";      // Stop orders only
CxlUserStop = "Z";      // User Stop orders only
```

EXCEPTION_HANDLING – This is not strictly part of Mass Cancellations; Serious program exceptions will record the relevant information in the log and stop the program running. It is unlikely that these will occur but in some extreme circumstances we may need to temporarily turn this off.

10.2 Mass Cancel Types:

These are the Mass Cancel Types:-

- `CxlUser = "U";`
- `CxlFirm = "F";`
- `CxlUserInactive = "V";`
- `CxlFirmInactive = "W";`
- `CxlFirmStop = "X";`
- `CxlUserStop = "Z";`

All of these follow the message formats detail in the examples below.

Inactive orders are not include on User or Firm delete requests and must delete via a separate request.

In all cases orders are tracked and Country, Market, Underlying level for each broker and delete requests are only issued when orders for that broker are present in a particular Country, Market, Underlying.

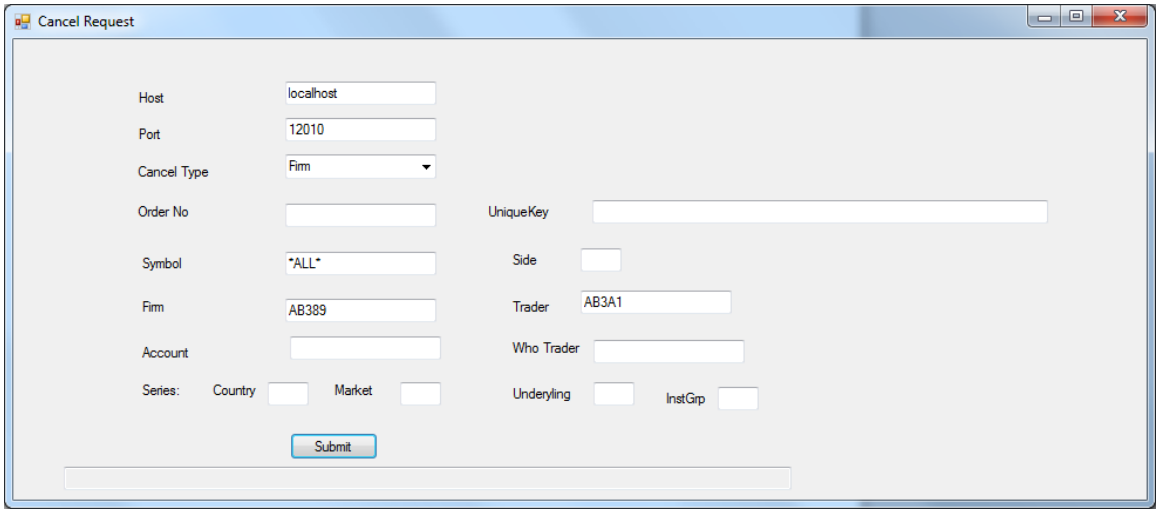
Similarly delete requests for Stop orders or inactive orders are only performed when they are present for the broker in a particular Country, Market, Underlying.

10.3 Mass Cancel All Orders for Firm:

The request format is identical to that currently used for OSEOrders, by default Stop orders are also deleted a separate requests must be made to delete stop orders where these are present for a country, market, underlying. (**CxlType=F|)**

We must issue a separate delete request for each country, market, underlying combination; we only do that when orders are present for that particular combination.

At the end of the process a %END% response is sent to the client providing the overall number of orders cancelled.



2014-06-13 12:08:44 |CmdClient:**CANCEL_REQUEST|USER=MARK|REQUEST_NO=6a4d5427-a249-4402-b667-67ed975e40cf|CxlType=F|OrderID=|Symbol=*ALL*|Side=|Country=SG|Firm=AB389|Trader=AB3A1|Account=|~**

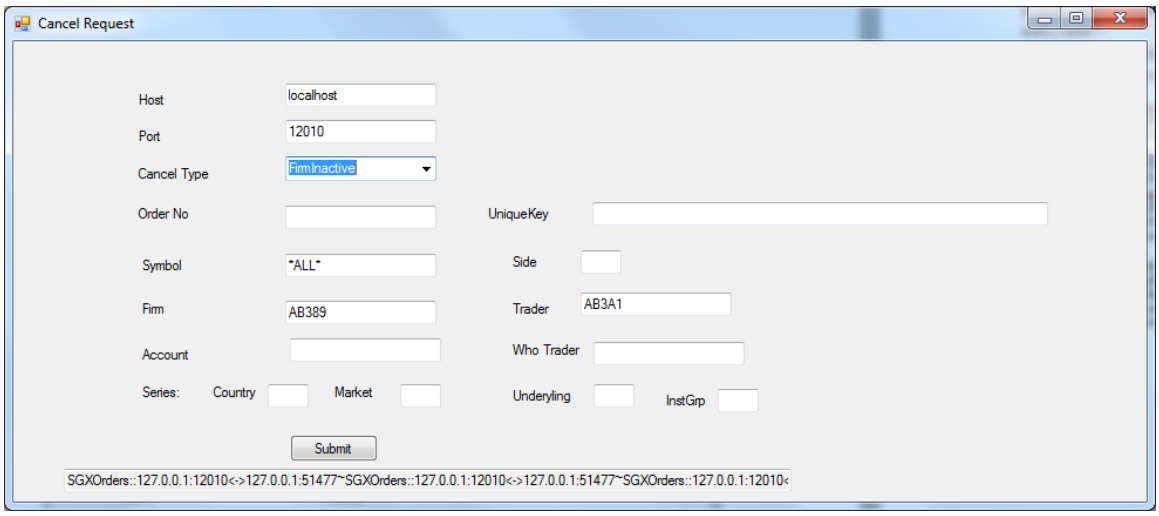
2014-06-13 12:08:44 |SendClientRequest - RequestID: 110
|MCT_FUNC=SEND|MSG=MSG_C_DELETE_ORDER|CODE=1640|DCY=Y|SER=83-155--1-204---1--1|OBC=1|WHO=SG-AB389--|TCD=SG-AB389-AB3A1|BOS=*|
2014-06-13 12:08:44 |SendClientRequest - RequestID: 111
|MCT_FUNC=SEND|MSG=MSG_C_DELETE_ORDER|CODE=1640|DCY=Y|SER=83-53--1-2000---1--1|OBC=1|WHO=SG-AB389--|TCD=SG-AB389-AB3A1|BOS=*|
2014-06-13 12:08:44 |SendClientRequest - RequestID: 112
|MCT_FUNC=SEND|MSG=MSG_C_DELETE_ORDER|CODE=1640|DCY=Y|SER=83-155--1-204---1--1|OBC=7|WHO=SG-AB389--|TCD=SG-AB389-AB3A1|BOS=*|

2014-06-13 12:08:45 |Series:13384 Underyings:84 Inst Groups:154 Instruments:807 Inst Classes:980 Orders:293 Trades:46 Order Trans:137
2014-06-13 12:08:45 |6a4d5427-a249-4402-b667-67ed975e40cf|request no| 110|Cancelled 7|Orders. Retained 0
2014-06-13 12:08:46 |6a4d5427-a249-4402-b667-67ed975e40cf|request no| 111|Cancelled 3|Orders. Retained 0
2014-06-13 12:08:46 |6a4d5427-a249-4402-b667-67ed975e40cf|request no| 112|Cancelled 8|Orders. Retained 0
2014-06-13 12:08:46 |CmdClient:SEND:REQUEST_NO=6a4d5427-a249-4402-b667-67ed975e40cf|Cancelled=18|Retained=0|Errors0%END%~

10.4 Mass Inactive Orders:

Inactive Orders cannot trade unless activated, separate requests must be issued to delete them.

They are deleted via a separate Mass Cancel request. **CxlType=W|**



2014-06-13 12:13:03
|CmdClient:CANCEL_REQUEST|USER=MARK|REQUEST_NO=9a92852b-1f76-4cea-bf1d-8bbc3d619566|CxlType=W|OrderID=|Symbol=*ALL*|Side=|Country=SG|Firm=AB389|Trader=AB3A1|Account=|~

2014-06-13 12:13:03 |SendClientRequest - RequestID: 114 |MCT_FUNC=SEND|MSG=MSG_C_DELETE_ORDER|CODE=1640|DCY=Y|SER=83-155--1-204---1--1|OBC=2|WHO=SG-AB389--|TCD=SG-AB389-AB3A1|BOS=*|
2014-06-13 12:13:03 |9a92852b-1f76-4cea-bf1d-8bbc3d619566|request no| 114|Cancelled 26|Orders. Retained 0
2014-06-13 12:13:03 |CmdClient:SEND:REQUEST_NO=9a92852b-1f76-4cea-bf1d-8bbc3d619566|Cancelled=26|Retained=0|Errors0%END%~