MCTradesSBI (DropCopy)

SBI Japannext Trading System Trade/Order Feeds

This Document:

MCTradesSBI (DropCopy).pdf – details how to install, configure and run MCTradesSBI (DropCopy).

Revision:

01/03/2012 – S.C. – Produced the first version of this manual.
08/08/2012 – Vaasugi – Modified and added the changes required for Drop Copy API
23/08/2012 – Vaasugi – Tabulated the fields in output feeds and database table in Appendix 1
25/10/2012 – Vaasugi – Updated the tables in Appendix 1 with few new fields

Similar MCTrades Products:

Similar MCTrades products exist for other exchanges:- ASX,SFE, HKEX, SGX, TSE contact RJE for more details.

1. Overview

MCTradesSBI (DropCopy) application communicates with the DropCopy Interface of SBI Japannext Trading System via FIX protocol. It extracts Orders/Trades from the consolidated feed of FIX messages supplied by the DropCopy Service. It then provides the Orders/Trades via comma-delimited feeds as well as stores them in the Database.

The following diagram depicts the overall functionality and connectivity of the MCTradesSBI (DropCopy) production system.

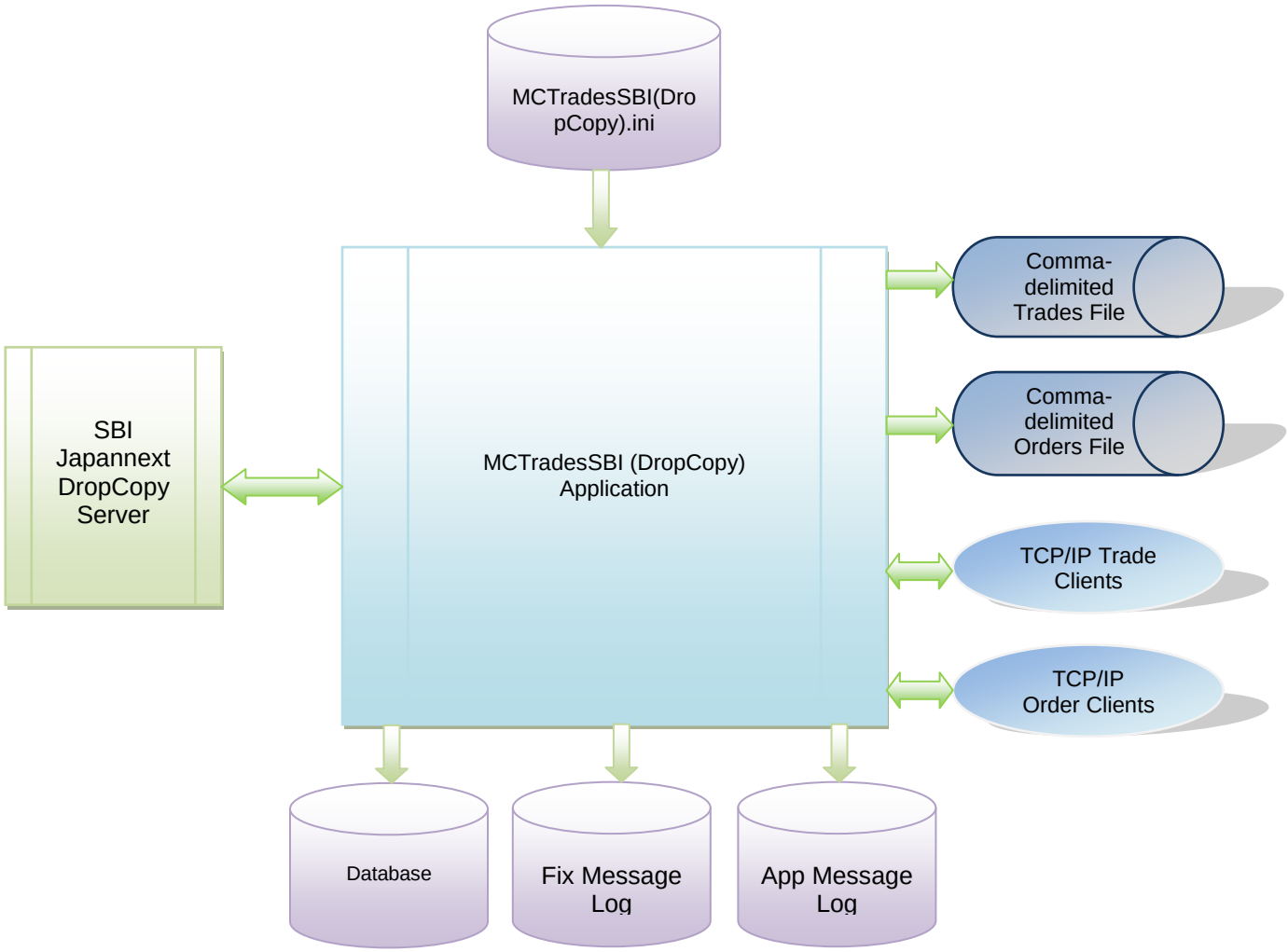


Figure 1. The MCTradesSBI (DropCopy) Production System

1.1 Features:

Trade Feed

Trades are available in the following output forms. Trade feed consists of all the Trade Reports extracted from SBI Japannext Trading System.

- Comma-delimited Trade File (single trade side)
- Comma-Delimited TCP/IP Trade Feed (single trade side)

Order Feed

Orders are available in the following output forms. Order feed consists of all the Execution Reports extracted from SBI Japannext Trading System.

- Comma-delimited order file (single trade side)
- Comma-Delimited TCP/IP order feed (single trade side)

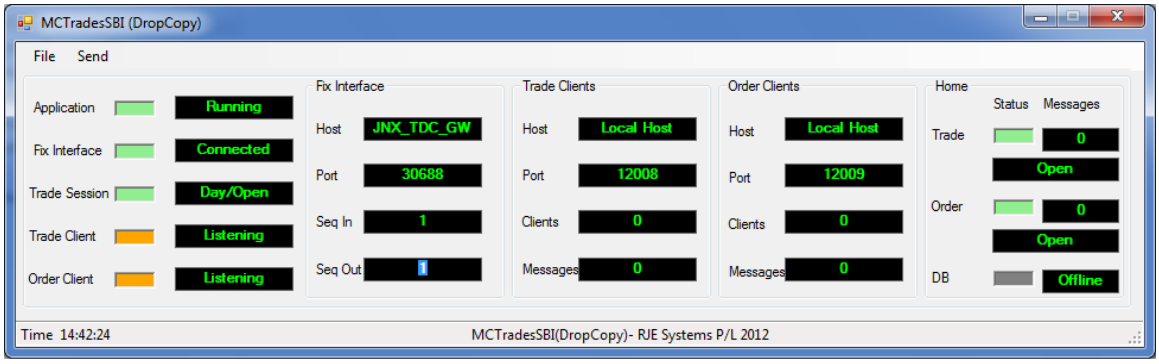
Note: The Comma-Delimited TCP/IP feed is similar to all other MCTrades products.

Data Store

Orders extracted from SBI Japannext Trading System are stored in the database table sbi_orders. A SQL database is required for this feature. More details can be found in 8. Database

The user can configure the application to enable/disable any of the above features. More details can be found in 4.8 Feature Filter Parameters:

1.2 GUI Screen:



The application contains a GUI screen which gives a quick visual indication that everything is working. Typically, good status values are green but status values may transit other states during stopping and starting.

Application Status:-

- Starting (Orange)
- Running (Green) – normal
- Stopping (Red)
- Hibernating (Grey) – normal overnight.
- Waiting (Grey) – normal when user press ‘stop’

FIX Interface Status:-

- Starting (White)
- Recovering (Yellow)
- Connecting (Orange)
- Connected (Green)
- Closing (Grey)

Trade Session Status:-

- Day/Open (Green)
- Day/Closed (Grey)
- Night/Open (Green)
- Night/Closed (Grey)
- Unknown (Grey)

Trade Clients Status:- (if accepting client connections)

- Listening (Orange) - accepting connections
- Connected (Green) – one or more clients connected
- Stopping (Grey)

Order Clients Status:- (if accepting client connections)

- Listening (Orange) - accepting connections
- Connected (Green) – one or more clients connected
- Stopping (Grey)

Trades File Status:-

- Open (Green)
- Closed (Grey)
- Error (Red)

Orders File Status:-

- Open (Green)
- Closed (Grey)
- Error (Red)

Database Status:-

- Open (Green)
- Not connected (Grey)

2. Daily Cycle

SBI Japannext Trading System has multiple Trading Sessions and while MCTradesSBI (DropCopy) runs for multiple days, it shuts down and wakes up at a certain scheduled time each day, each Trading Session.

Refer:- 4.7 Daily Cycle Parameters:

Note: We currently have no way of detecting Market Close in SBI Japannext DropCopy service. A timed shutdown is the only option.

Note: SBI Japannext currently has one Day Trading Session and one Night Trading Session. MCTradesSBI(DropCopy) uses different Session IDs for these, **TDC** is the Trading Session ID for Day Session and **TNC** for Night Session.

3. Installation

Install MCTradesSBI (DropCopy) as follows :-

<Install Directory> :- MCTradesSBI.exe, MCTradesSBI(DropCopy).ini
<Install Directory>:- Mono.Security.dll, Npgsql.dll
<Install Directory>/logs :- Make a subdirectory for logs, trades and orders files.

To run the program, run MCTradesSBI.exe, with the presence of a correctly configured MCTradesSBI(DropCopy).ini file and the two dll files Mono.Security.dll and Npgsql.dll.

You must set the following parameters correctly:-

- Parameters – Fix Connection Configuration 4.1 FIX Connection Parameters:
- Parameters – Fix Logon Configuration 4.2 FIX Logon Parameters:

If you wish to run the programs without a GUI refer 4.9 Other parameters:

Note: When upgrade to a new version intra-day you should copy the FIX log file if installing in a new directory.

4 Configuration

All configuration parameters are stored in MCTradesSBI(DropCopy).ini

4.1 FIX Connection Parameters:

FIX connection parameters should be configured for each Trading Session. Parameter names come with the Trading Session ID.

FIX_TDC_SERVER_HOST = Name of SBI Japannext DropCopy Server for Day Trading Session
e.g **FIX_TDC_SERVER_HOST**=JNX_TDC_GW

FIX_TNC_SERVER_HOST = Name of SBI Japannext DropCopy Server for Night Trading Session
e.g **FIX_TNC_SERVER_HOST**=JNX_TNC_GW

FIX_TDC_SERVER_PORT = Port to connect SBI Japannext DropCopy Day Trading Session
e.g **FIX_TDC_SERVER_PORT**=30688

FIX_TNC_SERVER_PORT = Port to connect SBI Japannext DropCopy Night Trading Session
e.g **FIX_TNC_SERVER_PORT**=30688

SBI Japannext will supply values for these parameter settings.

Note: IP address of the Drop Copy Server Hosts can be entered in the ini file. Alternatively an entry for each Drop Copy Server Host can be made in the windows host file (C:\Windows\System32\drivers\etc\hosts) with a name for example JNX_TDC_GW and that name in turn can be used in the ini file.

4.2 FIX Logon Parameters:

FIX logon parameters should be configured for each Trading Session. Parameter names come with the Trading Session ID.

FIX_TDC_SENDER_ID = Part of Fix header for Day Trading, a valid value must be specified.
e.g **FIX_TDC_SENDER_ID**=TDC12479A

FIX_TNC_SENDER_ID = Part of Fix header for Night Trading, a valid value must be specified.
e.g **FIX_TNC_SENDER_ID**=TNC12479A

FIX_TDC_TARGET_ID = Part of Fix header for Day Trading, a valid value must be specified.
e.g **FIX_TDC_TARGET_ID**=TDJNX

FIX_TNC_TARGET_ID = Part of Fix header for Night Trading, a valid value must be specified.
e.g **FIX_TNC_TARGET_ID**=TNJNX

FIX_TDC_USER_NAME = user for Fix Logon for Day Trading.
e.g **FIX_TDC_USER_NAME**=tdc12479

FIX_TNC_USER_NAME = user for Fix Logon for Night Trading.
e.g **FIX_TNC_USER_NAME**=tnc12479

FIX_TDC_PASSWORD = Password for Fix Logon for Day Trading.
e.g **FIX_TDC_PASSWORD**=Password

FIX_TNC_PASSWORD = Password for Fix Logon for Night Trading.
e.g **FIX_TNC_PASSWORD**=Password

SBI Japannext will supply values for these parameter settings.

4.3 Optional FIX Parameters:

FIX_HEARTBEAT=<Heartbeat interval> (Seconds) – default = 30
Example **FIX_HEARTBEAT**=30

Note: You should consult SBI Japannext before setting this parameter, the default of 30 seconds is recommended.

4.4 Trade Feed Parameters:

These are TCP/IP ports that applications can connect to receive a feed of trade data.

The format of the data is described in 5. Comma Delimited Application Development

TRADES_PORT = TCP/IP port for trades.
e.g **TRADES_PORT**=12008

4.5 Order Feed Parameters:

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

The format of the data is described in 5. Comma Delimited Application Development

ORDERS_PORT = TCP/IP port for all Orders.
e.g. **ORDERS_PORT**=12009

4.6 Logging Parameters:

Application and FIX logs are text files that can be used for trouble shooting. The logs' names contain the Trading Session ID which the logs belong to, making it easy for the user to differentiate logs of one session from others.

APP_LOG_FILE = file base for application log, a new log is taken each run; the application log includes Trading Session ID, current date and time.
e.g **APP_LOG_FILE**= MCTradesSBI
The name of the file e.g MCTradesSBI.TDC.App.Messages.20120808_144159.log

FIX_LOG_FILE = file base for FIX Message Log; a new log is taken each trading session; the filename always includes the Trading Session ID and the current date.
e.g **FIX_LOG_FILE**= MCTradesSBI
The name of the file e.g MCTradesSBI.TDC.Fix.Messages.20120808

APP_LOG_DIRECTORY=Directory where the application log is stored.
e.g **APP_LOG_DIRECTORY**=logs

FIX_LOG_DIRECTORY=Directory where FIX Message Log is stored.
e.g **FIX_LOG_DIRECTORY**=logs

APP_DATA_DIRECTORY=Directory where the Trades and Orders files are stored.
e.g **APP_DATA_DIRECTORY**=data

Note: APP_DATA_DIRECTORY defaults to APP_LOG_DIRECTORY if not specified.

If you don't specify these settings, defaults will apply.

Note: In this application the FIX Message Log is important see 6.1 FIX Message Log: for more details.

4.7 Daily Cycle Parameters:

Refer:- 2. Daily Cycle

Wake up and Shut down times should be configured for each Trading Session. Parameter names come with the Trading Session ID.

TDC_WAKE_TIME = time when program wakes up each morning for day trading session (hour:min), default 08:20.
e.g **TDC_WAKE_TIME**=08:20

TDC_SHUT_TIME = time when the program shutdown after day trading session and hibernation occurs (hour:min) default 16:30.
e.g **TDC_SHUT_TIME**=16:30

TNC_WAKE_TIME = time when program wakes up every night for night trading session (hour:min), default 19:00.
e.g **TNC_WAKE_TIME**=19:00

TNC_SHUT_TIME = time when the program shutdown after night trading session and hibernation occurs (hour:min) default 23:59.
e.g **TNC_SHUT_TIME**=23:59

4.8 Feature Filter Parameters:

Refer:- 1.1 Features:

TRADE_FEED=TRUE – set TRUE to enable Trade Feed

ORDER_FEED=TRUE – set TRUE to enable Order Feed

DB_CONNECTION=TRUE – set TRUE to enable database connection and storing of orders in the database

4.9 Other Parameters:

NO_GUI=YES – set to enable the application running with no GUI

NO_TEST_ORDERS=YES – this should always be set to YES

See also

- 8.2 Database Parameters:

4.10 Configuration File Example :

```
*********************************
* APPLICATION VERSION        *
*********************************
APP_VERSION=MCTradesSBI(DropCopy)- RJE Systems P/L 2012
*********************************
* RUN WITHOUT GUI            *
*********************************
*NO_GUI=YES
*********************************
* RUN WITHOUT TEST ORDERS     *
*********************************
NO_TEST_ORDERS=YES
*********************************
* FIX Session properties      *
*********************************
FIX_TDC_SERVER_HOST=JNX_TDC_GW
FIX_TNC_SERVER_HOST=JNX_TNC_GW
FIX_TDC_SERVER_PORT=30688
FIX_TNC_SERVER_PORT=30688
FIX_TDC_SENDER_ID=TDC12479A
FIX_TNC_SENDER_ID=TNC12479A
FIX_TDC_USER_NAME=tdc12479
FIX_TNC_USER_NAME=tnc12479
FIX_TDC_PASSWORD=f88XW556
FIX_TNC_PASSWORD=2Hk7l2U9
FIX_TDC_TARGET_ID=TDJNX
FIX_TNC_TARGET_ID=TNJNX
FIX_CHANGE_PASS=NO
FIX_HEARTBEAT=30
*********************************
* TCP Clients properties      *
*********************************
TRADES_PORT=12008
ORDERS_PORT=12009
ORDER_REFRESH=DELETE
************************************
* Application Log File properties *
************************************
APP_LOG_FILE=MCTradesSBI
APP_LOG_DIRECTORY=logs
LOGGING_LEVEL=9
*********************************
* Fix Log File properties    *
*********************************
FIX_LOG_FILE=MCTradesSBI
FIX_LOG_DIRECTORY=logs
************************************
* BROKER_LIST To filter own trades*
************************************
BROKER_LIST=ABN01
************************************
* FEATURE SELECTOR              *
************************************
TRADE_FEED=TRUE
```

```
ORDER_FEED=TRUE
DB_CONNECTION=TRUE
***********************************
* WAKE UP/SHUT DOWN            *
***********************************
TDC_WAKETIME =08:20
TDC_SHUTTIME =16:30
TNC_WAKETIME =19:00
TNC_SHUTTIME =23:59
***********************************
* SQL Database Parameters      *
***********************************
SQL_DATABASE_NAME=webdb
SQL_DATABASE_SERVER=127.0.0.1
SQL_DATABASE_PORT=5432
SQL_USER_ID=postgres
SQL_PASSWORD=rjeadmin
************ END **************
```

5 Comma-Delimited Application Development

One option for developers is to make a TCP/IP separate connections to MCTradesSBI trade/order feed ports and receive trades/orders data in comma-delimited format separately. Data is simply sent when it is available; there is no need to request data. In this case trades are single sided, all data received from SBI Japannext is included.

The port for clients connections is configured in :- 4.4 Trade Feed Parameters:

5.1 Comma-Delimited Header:

Most applications would process the header as it gives a list of field names corresponding to field positions.

Trades

Country|S,Exchange|S,Market|S,FirmID|S,TraderID|S,ClientID|S,MsgSeqNo|N,OrderID|S,ClOrderID|S,ExecID|S,ExecType|N,Account|S,Symbol|S,Side|S,Price|N,OrderQty|N,OrderType|N,OrderCapacity|S,OrdStatus|N,ExecTransType|N,Description|S,AvgPrice|N,LastPrice|N,LastShares|N,TradeValue|N,CumQty|N,LeavesQty|N,TimeInForce|N,MinQty|N,MaxFloor|N,LastLiquidityInd|N,TransactTime(UTC)|T,ExecutedDate(Local)|D,ExecutedTime(Local)|T,TradeDate(JST)|D,TimeStamp(UTC)|TS,~

Orders

Country|S,Exchange|S,Market|S,FirmID|S,TraderID|S,ClientID|S,MsgSeqNo|N,OrderID|S,ClOrderID|S,OrgClOrderID|S,ExecID|S,ExecType|N,Account|S,Symbol|S,Side|S,Price|N,OrderQty|N,OrderType|N,OrderCapacity|S,OrderStatus|N,ExecTransType|N,Description|S,AvgPrice|N,LastPrice|N,LastShares|N,CumQty|N,LeavesQty|N,TimeInForce|N,MinQty|N,MaxFloor|N,LastLiquidityInd|N,TransactTime(UTC)|T,ExecutedDate(Local)|D,ExecutedTime(Local)|T,TradeDate(JST)|D,TimeStamp(UTC)|TS,~

5.2 Comma-Delimited Data:

Fields that are not relevant are simply empty:-

Trades

JAPAN,SBI,SBI,TDJNX,TDC12479A,GR12479XXXA,39,20121025-1888f,1509015,20121025-4cb8a,2,ABN01,7203,1,3200,1000,2,A,2,0,,3102.5,3102.5,1000,3102500.00,1000,0,0,,,2,20121025-04:09:17,20121025,15:09:17,,20121025-04:09:17,~

Orders

JAPAN,SBI,SBI,TDJNX,TDC12479A,GR12479XXXA,195,20121025-188d7,1509020,1509020,20121025-4e085,4,ABN01,7203,2,3200,1000,2,A,4,0,,3200,,,600,0,0,,,,20121025-04:56:56,20121025,15:56:56,,20121025-04:56:56,~

Note: Additional examples are available from RJE.

5.3 Trades File:

A Trade file is produced for each trading session with a comma-delimited header and a comma-delimited trade record for each trade. The contents of this file are identical to the data that would be sent for a trades feed.

On a restart mid-session, the internal copy of the trades is recreated from the FIX Message Log and the old trades file gets replaced by a new trades file.

The trades file's name contains the Trading Session ID which the file belongs to, making it easy for the user to differentiate it from the others.

e.g MCTradesSBI.TDC.20120808.trades

5.4 Orders File:

An Order file is produced for each trading session with a comma-delimited header and a comma-delimited execution record for each execution. The contents of this file are identical to the data that would be sent of an orders feed.

On a restart mid-session, the internal copy of the orders is recreated from the FIX Message Log and the old orders file gets replaced by a new orders file.

The orders file's name contains the Trading Session ID which the file belongs to, making it easy for the user to differentiate it from the others.

e.g MCTradesSBI.TDC.20120808.orders

6 Message Sequence Numbers

Message Sequence Numbers start from 1 each trading session. By default when reconnecting/restarting mid-session, the sequence numbers at both ends continue on from their previous values and any missing messages are recovered. Hence, on a restart the application reprocesses the FIX Message log to re-establish outbound/inbound sequence numbers.

6.1 FIX Message Log:

Typically the FIX session is continued across runs and there is a single FIX Message log for each trading session. Messages sent/received are recovered from the FIX Messages log at startup. When resuming the FIX session the application only fetches the new messages.

You can specify a filename/directory for this file in 4.6 Logging Parameters:

6.2 Missing FIX Message Log:

A missing FIX message log could be caused by the following things:-

- Running from a different directory or with different .ini settings.
- Deleting or renaming the file.

This can cause problems with the sequence number of the login message we send to the SBI Japannext Trading System. If the sequence number is less than expected the exchange will ignore this message and you will eventually get the following error:-

```
|*********************************************************************************************
*** Fatal Error - Exceeded FIX Logon retries - Check Config FIX_SERVER_PORT / FIX_SERVER_HOST.  ***
*** If settings above are corrent then could be a problem with the Fix Message log.                 ***
*** See MCTradesSBI.PDF - 6.2 Missing Fix Message Log.                                              ***
*** You can run MCTradesSBI -s 'nnn'.   Where 'nnn' = last message sequence no from our end.        ***
|*********************************************************************************************
```

This error could mean the FIX Message log has been deleted or you could simply be connecting to the wrong host/port.

Note: Resetting the message sequence number will solve this error. The version of FIX protocol being used (version 4.2 with some features of 4.4 back ported) supports resetting sequence numbers. For more info see 7.2 User set Message Sequence Numbers:

Note: A message log error can only be a problem if you have successfully connected earlier.

6.3 Specifying a Restart Sequence No:

If you know what the outbound FIX sequence number from your end should be, you can specify it as follows.

MCTradesSBI –S nnn

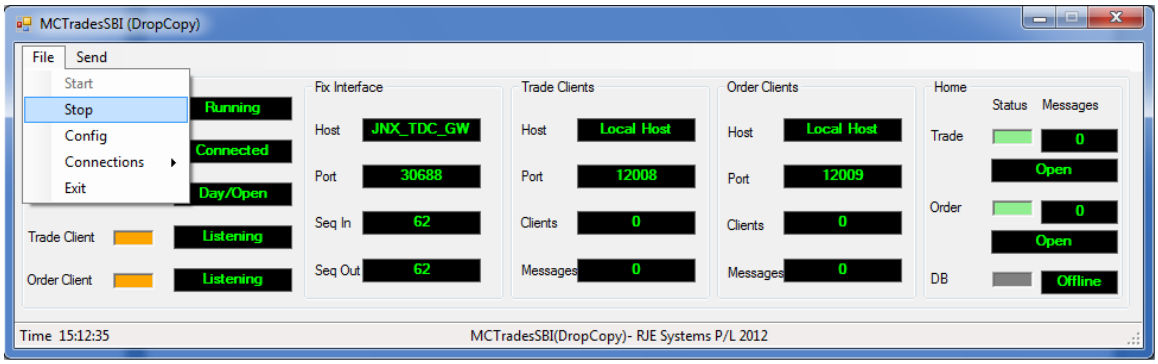
Note: Where nnn is the sequence number.

You should be able to get the number from the previous FIX Message Log. If you don't know this number you can obtain it from the SBI Japannext administrator or he can reset the FIX session (as the last option). In this mode the application will re-request all trades for the trading session from SBI Japannext.

7 Features for Testing

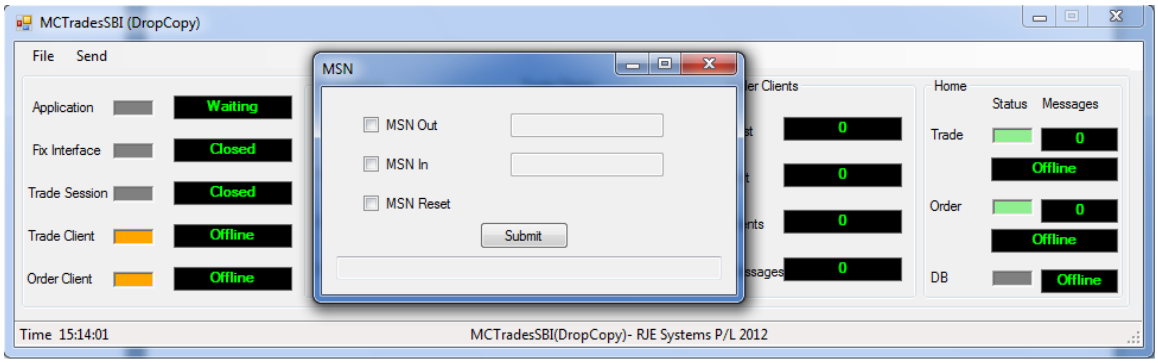
The following features are included solely for the purpose of facilitating the testing of this application.

7.1 Stop and Start:



The 'Stop' and the 'Start' submenus provided under the File menu facilitate stopping and restarting the application without exiting from the application.

7.2 User set Message Sequence Numbers:



When restarting the application via Stop/Start menu, the user is given the option of changing the last outbound (MSN Out) and inbound (MSN In) message sequence numbers in MCTradesSBI(DropCopy) application. He also can reset the message sequence number if he wants. This feature could be used during the conformance tests to see how both parties react (application and exchange) for such changes in the message sequence number as well as to resolve the error when the FIX log file is missing.

See 6.2 Missing FIX Message Log:

8 Database

Database design, tables and functions have been developed and tested with a ProsgreSQL database running under Windows and Linux.

The MCTradesSBI(DropCopy) application uses the "npgsql" .net data provider for PostgreSQL. It calls ProgreSQL Functions (Stored Procedures) for database access and update.

8.1 Database Tables:

Table - system

The purpose of this table is to allocate a unique Guid (uuid) to each system. In this context MCTradesSBI(DropCopy) is one system. All data of MCTradesSBI(DropCopy) has the system_id of MCTradesSBI.

Function :- get_system_info() create/retrieve system table information for a particular system.

```
-- Table: "system"

-- DROP TABLE "system";

CREATE TABLE "system"
(
  id bigserial NOT NULL,
  system_id uuid,
  system_name character varying(50),
  exchange character varying(10)
)
WITH (
  OIDS=TRUE
);
ALTER TABLE "system" OWNER TO postgres;
GRANT ALL ON TABLE "system" TO postgres;
GRANT ALL ON TABLE "system" TO public;
```

Table – system_state

This table shows the current state of a system indicating if the system is ready to store the data.

Currently defined system states are:-

```
enum SessionState : int
{
    Connecting = 10,
    Connected  = 20,
    Ready      = 30,
    Closed     = 90
}
```

The table is also updated periodically to provide `memory_trans` and `database_trans` counters. These provide feedback of whether orders table updating is keeping with the rate execution reports are being sent by the SBI Japannext Trading System.

Function :- update_system_state() - updates the system_state table.

```
-- Table: system_state

-- DROP TABLE system_state;

CREATE TABLE system_state
(
  id bigserial NOT NULL,
  system_id uuid,
  system_state integer,
  host_name character varying(10),
  port_no integer,
  memory_trans integer,
  database_trans integer,
  last_update timestamp without time zone
)
WITH (
  OIDS=TRUE
);
ALTER TABLE system_state OWNER TO postgres;
GRANT ALL ON TABLE system_state TO postgres;
GRANT ALL ON TABLE system_state TO public;
```

Table – sbi_orders

This is the main table of interest which stores orders' data. As execution reports occur the current state of the database is updated to reflect the current state of the order. When the field order_active='Y' the order is an active order which is a candidate for cancellation. As orders trade out or are cancelled order_active is set to 'N'.

The orders information is kept in the DB indefinitely as it may be useful.

Function :- sbi_update_order() – Updates the orders table for each execution report transaction.

```
-- Table: sbi_orders

-- DROP TABLE sbi_orders;

CREATE TABLE sbi_orders
(
  id bigserial NOT NULL,
  system_id uuid NOT NULL,
  country character varying(10),
  exchange character varying(10),
  market character varying(10),
  firm_id character varying(30),
  trader_id character varying(30),
  client_id character varying(30),
  message_no integer,
  order_active character(1),
  order_id character varying(50) NOT NULL,
  clord_id character varying(50),
  org_clord_id character varying(50),
  exec_id character varying(50),
  exec_type character varying(4),
  account character varying(30),
  symbol character varying(50),
  side character(1),
  price numeric(18,4),
  order_qty numeric,
  order_type character varying(4),
  order_capacity character(1),
  order_status character(1),
  exec_transact_type character varying(4),
  description character varying(50),
  avg_price numeric(18,4),
  last_price numeric(18,4),
  last_fill numeric,
```

```
  no_of_fills smallint,
  cum_qty numeric,
  leaves_qty numeric,
  time_in_force character(1),
  min_qty numeric,
  max_floor numeric,
  liquidity_ind character(1),
  transact_time timestamp without time zone,
  trade_date character varying(10),
  time_stamp timestamp without time zone,
  CONSTRAINT sbi_order_pkey PRIMARY KEY (system_id, order_id)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE sbi_orders OWNER TO postgres;
GRANT ALL ON TABLE sbi_orders TO postgres;
```

8.2 Database Parameters:

SQL_DATABASE_NAME=Name of the database to access.
e.g SQL_DATABASE_NAME=webdb

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

SQL_DATABASE_PORT=Port for the PostgreSQL database.
e.g SQL_DATABASE_PORT=5432

SQL_USER_ID=PostgreSQL database user.
e.g SQL_USER_ID=postgres

SQL_PASSWORD= PostgreSQL database user password*
e.g SQL_PASSWORD=rjexxxxxx

8.3 npgsql files:

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

```
Mono.Security.dll
Npgsql.dll
```

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

8.4 SQL Script Files:

The following files create database tables:-

CREATE TABLE sbi_orders.sql
CREATE TABLE system.sql
CREATE TABLE system_state.sql

The following files create database functions:-

FUNCTION sbi_update_order.sql
FUNCTION get_system_info.sql
FUNCTION update_system_state.sql

Appendix 1

Order Feed:

Field Number	Field Name in Output	Presence	Fix Field Name	Fix Tag
1	Country	{Always}	(Internal)	-
2	Exchange	{Always}	(Internal)	-
3	Market	{Always}	(Internal)	-
4	FirmID	{Always}	SenderCompID	49
5	TraderID	{Always}	TargetCompID	56
6	ClientID	{Not Always}	ClientID	109
7	MsgSeqNo	{Always}	MessageSeqNo	34
8	OrderID	{Always}	OrderID	37
9	ClOrdID	{Not Always}	ClOrdID	11
10	OrgClOrdID	{Not Always}	OrgClOrdID	41
11	ExecID	{Always}	ExecID	17
12	ExecType	{Always}	ExecType	150
13	Account	{Not Always}	Account	1
14	Symbol	{Always}	Symbol	55
15	Side	{Always}	Side	54
16	Price	{Always}	Price	44
17	OrderQty	{Always}	OrderQty	38
18	OrderType	{Always}	OrdType	40
19	OrderCapacity	{Not Always}	Capacity (Rule80A)	47
20	OrderStatus	{Always}	OrdStatus	39
21	ExecTransType	{Always}	ExecTransType	20
22	Description	{Not Always}	Text	58
23	AvgPrice	{Always}	AvgPx	6
24	LastPrice	{Trade/Always}	LastPx	31
25	LastShares	{Trade/Always}	LastShares	32
26	CumQty	{Always}	CumQty	14
27	LeavesQty	{Always}	LeavesQty	151
28	TimeInForce	{Not Always}	TimeInForce	59

29	MinQty	{Not Always}	MinQty	110
30	MaxFloor	{Not Always}	MaxFloor	111
31	LastLiquidityInd	{Trade/Always}	LastLiquidityInd	851
32	TransactTime (UTC)	{Always}	TransactTime	60
33	ExecutedDate (Local)	{Always}	Date(TransactTime)	-
34	ExecutedTime (Local)	{Always}	Time(TransactTime)	-
35	TradeDate (JST)	{Trade/Not Always}	TradeDate	75
36	TimeStamp (UTC)	{Always}	TransactTime	-

Trade Feed:

Field Number	Field Name in Output	Presence	Fix Field Name	Fix Tag
1	Country	{Always}	(Internal)	-
2	Exchange	{Always}	(Internal)	-
3	Market	{Always}	(Internal)	-
4	FirmID	{Always}	SenderCompID	49
5	TraderID	{Always}	TargetCompID	56
6	ClientID	{Not Always}	ClientID	109
7	MsgSeqNo	{Always}	MessageSeqNo	34
8	OrderID	{Always}	OrderID	37
9	ClOrdID	{Not Always}	ClOrdID	11
10	ExecID	{Always}	ExecID	17
11	ExecType	{Always}	ExecType	150
12	Account	{Not Always}	Account	1
13	Symbol	{Always}	Symbol	55
14	Side	{Always}	Side	54
15	Price	{Always}	Price	44
16	OrderQty	{Always}	OrderQty	38
17	OrderType	{Always}	OrdType	40
18	OrderCapacity	{Not Always}	Capacity (Rule80A)	47
19	OrderStatus	{Always}	OrdStatus	39
20	ExecTransType	{Always}	ExecTransType	20

21	Description	{Not Always}	Text	58
22	AvgPrice	{Always}	AvgPx	6
23	LastPrice	{Always}	LastPx	31
24	LastShares	{Always}	LastShares	32
25	TradeValue	{Always}	(Derived)	-
26	CumQty	{Always}	CumQty	14
27	LeavesQty	{Always}	LeavesQty	151
28	TimeInForce	{Not Always}	TimeInForce	59
29	MinQty	{Not Always}	MinQty	110
30	MaxFloor	{Not Always}	MaxFloor	111
31	LastLiquidityInd	{Always}	LastLiquidityInd	851
32	TransactTime (UTC)	{Always}	TransactTime	60
33	ExecutedDate (Local)	{Always}	Date(TransactTime)	-
34	ExecutedTime (Local)	{Always}	Time(TransactTime)	-
35	TradeDate (JST)	{Not Always }	TradeDate	75
36	TimeStamp (UTC)	{Always}	TransactTime	-

Order Table:

Field Number	Colum Name	Fix Field Name	Fix Tag
1	country	(Internal)	-
2	exchange	(Internal)	-
3	market	(Internal)	-
4	firm_id	SenderCompID	49
5	trader_id	TargetCompID	56
6	client_id	ClientID	109
7	message_no	MessageSeqNo	34
8	order_active	(Derived)	-
9	order_id	OrderID	37
10	clord_id	ClOrdID	11
11	org_clord_id	OrgClOrdID	41
12	exec_id	ExecID	17

13	exec_type	ExecType	150
14	account	Account	1
15	symbol	Symbol	55
16	side	Side	54
17	price	Price	44
18	order_qty	OrderQty	38
19	order_type	OrdType	40
20	order_capacity	Capacity (Rule80A)	47
21	order_status	OrdStatus	39
22	exec_trans_type	ExecTransType	20
23	description	Text	58
24	avg_price	AvgPx	6
25	last_price	LastPx	31
26	last_fill	LastShares	32
27	no_of_fills	(Derived)	-
28	cum_qty	CumQty	14
29	leaves_qty	LeavesQty	151
30	time_in_force	TimeInForce	59
31	min_qty	MinQty	110
32	max_floor	MaxFloor	111
33	liquidity_ind	LastLiquidityInd	851
34	transact_time	TransactTime	60
35	trade_date	TradeDate	75
36	time_stamp	(Database Update Time)	-