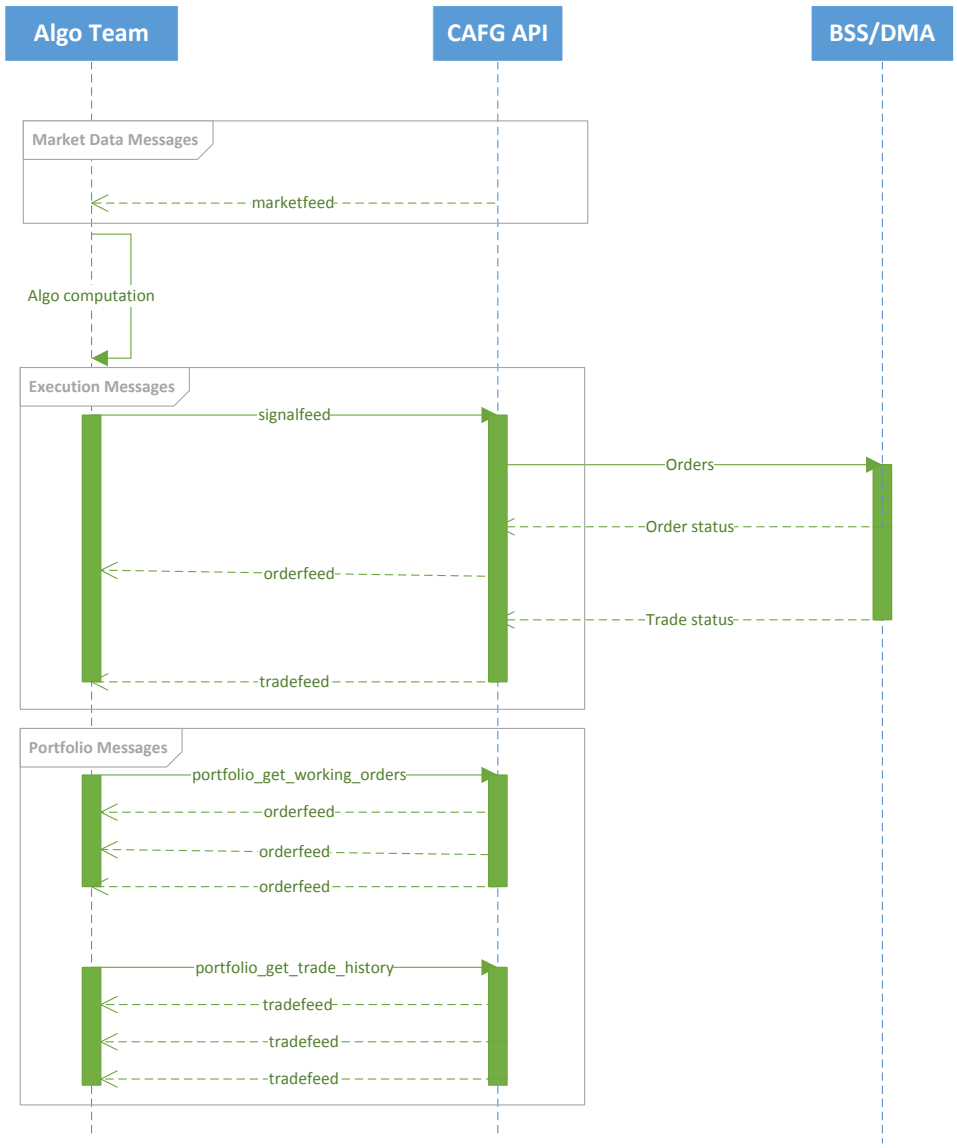
CAFG Protocol Documentation

Overview

CAFG Protocol mainly provides financial market data to Algo Team and takes up a role as a signal receiver from Algo Team message(s) and executes its order(s) to Exchange.
Please note that certain messages are only available in simulation (*connected to BSS testing environment/simulation engine*), production (*connected to BSS real trading environment*) and not in testing (*not connected to BSS e.g. during paper trading testing*).

Data flow example:



Subscription Message

Firstly, Algo Team server connects to CAFG Protocol market data server via TCP connection. Secondly, Algo Team sends subscription message to CAFG Protocol Server. Thirdly, CAFG Protocol sends the current snapshot(s) of subscribed product market data to Algo Team. After the snapshot(s) are sent, it will stream the live market data which come from the exchange.

Subscription Message Format

[testing/simulation/production]:

Field no.	Field name	Description	Sample
1	Timestamp	timestamp, in the format of YYYYMMDD_HHMMSS_NNNNNN	20140119_091500_123213,
2	Message	Command to the server	subscription,
3	Market	Code of market	HKIF,
4	Product Code	Code of product	HSIF4,
5	Begin Date	Subscription date range (inclusive). To subscribe real-time data, set begin date to TODAY	20140119,
6	End Date	Subscription date range (inclusive). To subscribe real-time data, set end date to TODAY	20140119,
7	[Subscription Attribute]	Optional*(only used for backtesting mode) --- To control the market data replay speed (for backtesting only): replay_speed=10 (market data will be replayed 10x faster than normal speed) --- To send every market data record only after receiving acknowledgement from client: ack_mode=YES	
8	End line	Indicates the end of the message	\n

Example of subscription message:

20140121_092116_405470,subscription,HKIF,HSIF4, 20140119, 20140119

Example of subscription message (control speed):

20140121_092116_405470,subscription,HKIF,HSIF4, 20140119, 20140119,replay_speed=10

Example of subscription message (ack model):

20140121_092116_405470,subscription,HKIF,HSIF4, 20140119, 20140119, ack_mode=YES

Example of subscription message (control speed & ack model):

20140121_092116_405470,subscription,HKIF,HSIF4, 20140119, 20140119,
replay_speed=10;ack_mode=YES

NOTE: later subscription control setting will replace previous one.

Acknowledgement Message

In ack mode, CAFG Protocol will wait for acknowledgement from Algo team after sending every snapshot of subscribed market data.

Acknowledgement Message Format

[testing/simulation]:

Field no.	Field name	Description	Sample
1	Timestamp	timestamp, in the format of YYYYMMDD_HHMMSS_NNNNNN	20140119_091500_123213,
2	Message	Command to the server	acknowledgement,
3	Status	Acknowledgement status: 0: ok 1: error	0,
4	Error Msg	Error message if status = 1	
5	End line	Indicates the end of the message	\n

Example of acknowledgement message without error:

20140121_092116_405470,acknowledgement,0,

Example of acknowledgement message with error:

20140121_092116_405470,acknowledgement,1, network error

Market Data Message

Market Data Message Format

[testing/simulation/production]:

Field no.	Field name	Description	Sample
1	Timestamp	Server timestamp, in the format of YYYYMMDD_HHMMSS_NNNNNN	20140121_092116_405458,
2	Product Code	Code of product	HHIF4,
3	Traded Price	Traded price of the transaction, (if Traded Volume is not 0, otherwise it is the last traded price)	10127,
4	Traded Volume	Number of contract(s) traded in the transaction	0,
5	Start of Bid	(B) indicates Bid	B,
6	First Level Bid Price	Indicates 1st bid price of the product	10125,
7	First Level Bid Quantity	Number of contract(s) traded in the transaction on 1st bid price	5,
8	Second Level Bid Price	Indicates 2nd bid price of the product	10124,

9	Second Level Bid Quantity	Number of contract(s) traded in the transaction on 2nd bid price	5,
10	Third Level Bid Price	Indicates 3rd bid price of the product	10123,
11	Third Level Bid Quantity	Number of contract(s) traded in the transaction on 3rd bid price	4,
12	Fourth Level Bid Price	Indicates 4th bid price of the product	10122,
13	Fourth Level Bid Quantity	Number of contract(s) traded in the transaction on 4th bid price	23,
14	Fifth Level Bid Price	Indicates 5th bid price of the product	10121,
15	Fifth Level Bid Quantity	Number of contract(s) traded in the transaction on 5th bid price	2,
16	Start of Ask	(A) indicates Ask	A,
17	First Level Ask Price	Indicates 1st Ask price of the product	10127,
18	First Level Ask Quantity	Number of contract(s) traded in the transaction on 1st Ask price	10,
19	Second Level Ask Price	Indicates 2nd Ask price of the product	10128,
20	Second Level Ask Quantity	Number of contract(s) traded in the transaction on 2nd Ask price	9,
21	Third Level Ask Price	Indicates 3rd Ask price of the product	10129,
22	Third Level Ask Quantity	Number of contract(s) traded in the transaction on 3rd Ask price	6,
23	Fourth Level Ask Price	Indicates 4th Ask price of the product	10130,
24	Fourth Level Ask Quantity	Number of contract(s) traded in the transaction on 4th Ask price	4,
25	Fifth Level Ask Price	Indicates 5th Ask price of the product	10131,
26	Fifth Level Ask Quantity	Number of contract(s) traded in the transaction on 5th Ask price	14
27	End line	Indicates the end of the message	\n (i.e. hex code = 0x0a)

Example of market data message:

```
20140121_092116_405470,HSIF4,23056.000,0,B,23055.000,5,23054.000,2,23053.000,9,23052.000,5
,23050.000,8,A,23057.000,4,23058.000,6,23059.000,1,23060.000,3,23061.000,6
```

Order & Trade Messages

After the computation of Algo Team's algorithmic program, the program shall send signal message(s) ("signalfeed") to CAFG Protocol for order(s) execution.
The command and field values are case sensitive.

"signalfeed" message format (From Algo Team -> CAFG Protocol) [testing/simulation/production]:

Field no.	Field name	Description	Sample
1	Timestamp	timestamp, in the format of YYYYMMDD_HHMMSS_NNNNNN	20140119_091500_123213,
2	Message	Command to the server	signalfeed,
3	Market	Code of market	HKIF,
4	Product Code	Code of product	HSIF4,
5	Order ID	User supplied global unique order ID	0000001,
6	Price	order price for the signal	23401,
7	Quantity	order quality for the signal	5,
8	Open or Close	open,close; empty if Order Action = delete	close,
9	Buy or Sell	1- buy or 2 – sell	1,
10	Order Action	insert or delete	insert,
11	Order Type	limit_order; empty if Order Action = delete	limit_order,
12	Order Validity	today,fill_and_kill,fill_or_kill; empty if Order Action = delete	today
13	[Order Attribute]	Optional * (only used for auction order) Either empty or "auction=true"	auction=true
14	End line	Indicates the end of the message	\n

Example of signalfeed messages:

Algo Team -> CAFG Protocol send in a buy order

```
20140122_113136_123213,signalfeed,HKIF,HSIF4,0000001,23401,5,open,1,insert,limit_order,today
```

Once order has been received, CAFG Protocol sends message(s) ("orderfeed") to Algo Team in order to notify the order status.

"Orderfeed" message format (From CAFG Protocol->Algo Team) **[simulation/production]**:

Field no.	**Field name**	**Description**	**Sample**
1	Timestamp	Server timestamp, in the format of YYYYMMDD_HHMMSS_NNNNNN	20140110_101010_900000,
2	Feed name	Feed name of the message	orderfeed,
3	Market(string)	Market(string) from signalfeed	HKIF,
4	Product Code(string)	Product Code(string) from signalfeed	HSIF4,
5	Order ID(string)	Order ID(string) from signalfeed	0000001,
6	Price (double)	order price sent to the exchange	23401,
7	Quantity (double)	order quality sent to the exchange	10,
8	Open or Close (string)	open,close	open,
9	Buy or Sell (int)	1- buy or 2 - sell from	1,
10	Quantity Filled (double)	order quantity filled	6,
11	Deleted (int)	1 - if order is deleted; 0 – otherwise	1,
12	Order Status(int)	0 - OK, 1 – Error	0,
13	Error Description(string)	empty if order status is OK; error message if order status is Error	,
14	Source (int)	0 - "Market" for real-time updated from exchange 1 - "MessageReply"	0,

15	Order Type	current order type (indicate what order type this order is currently in) e.g. limit_order	limit_order
16	End line	Indicates the end of the message	\n

Remarks:

1) For "signalfeed" Order Action = Delete, the deleted field in the "orderfeed" is the confirmation for the order actually deleted. You should call "portfolio_get_working_orders" to verify if the order is still there.

2) Order should not be deleted twice.

Example of orderfeeds messages:

 # Algo Team <- CAFG Protocol orderfeeds update the status for the order submitted

```
20140122_113136_311333,orderfeed,HKIF,HSIF4,0000001,23401,5,open,1,0,0,0,,0,limit_order
20140122_113136_315328,orderfeed,HKIF,HSIF4,0000001,23401,5,open,1,0,0,0,,0,limit_order
20140122_113136_378242,orderfeed,HKIF,HSIF4,0000001,23401,5,open,1,0,0,0,,0,limit_order
```

Further, messages ("tradefeed") will be sent by CAFG Protocol to Algo Team in order to notify the order has been executed.

Tradefeed messages format (From CAFG Protocol-> Algo Team) **[simulation/production]**:

Field no.	**Field name**	**Description**	**Sample**
1	Timestamp	Server timestamp, in the format of YYYYMMDD_HHMMSS_NNNNNN	20140110_101010_900000
2	Feed name	Feed name of the message	tradefeed
2	Market(string)	Market(string) from signalfeed	HKIF,
3	Product Code(string)	Product Code(string) from signalfeed	HSIF4,
4	Order ID(string)	Order ID(string) from signalfeed	0000001,
5	Price (double)	traded price	23420,
6	Quantity (double)	Quantity traded	6,
7	Buy or Sell (int)	1- buy or 2 - sell from orderfeed	1,

8	Trade ID (string)	unique trade ID for the trade executed	2,
9	Source (int)	0 - "Market" for real-time updated from exchange 1 - "MessageReply" for reply of portfolio related messages	0
10	End line	Indicates the end of the message	\n

Remarks:

1) All fields from Algo Team to CAFG Protocol must be alpha numeric and underscore only, no consecutive underscore should be used.

2) Note that Algo Team to CAFG Protocol messages should be sent during trading hours only

Example of Tradefeed message:

```
20140122_113156_291416,tradefeed,HKIF,HSIF4,0000002,23401,1,2,24186,0
```

Portfolio Messages

After Algo Team Server connected to CAFG Protocol order & trade server, Algo Team can send "portfolio_get_working_orders" message to retrieve all the working orders for today in the market.

portfolio_get_working_orders messages format (From Algo Team->CAFG Protocol) [simulation/production]:

Field no.	Field Name	Description	Sample
1	Timestamp,	Server timestamp	20140119_091500_123213,
2	Message,	Command to the server	portfolio_get_working_orders,
3	Period,	Work order period, only support "Today" for now	today
4	End line	Indicates the end of the message	\n

After CAFG Protocol received the "portfolio_get_working_orders" message , it replies the working orders in the format of ("orderfeed") with source set as "1" to Algo Team Server. *(Please refer to the format of "orderfeed" which has been described before)*

On the other hand, Algo Team Server sends ("portfolio_get_trade_history") message to CAFG Protocol for retrieving all the trade history for today.

Example of portfolio_get_working_orders messages:

Algo Team -> CAFG Protocol get working orders for today

```
20140122_113451_123213,portfolio_get_working_orders,today
```

Algo Team <- CAFG Protocol order status update to show that 0000001 is the working order

```
20140122_113451_396372,orderfeed,HKIF,HSIF4,0000001,23401,5,close,1,1,0,0,,1,limit_order
```

portfolio_get_trade_history messages format (From Algo Team->CAFG Protocol) [simulation/production]:

Field no.	Field Name	Description	Sample
1	Timestamp,	Server timestamp	20140119_091500_123213,
2	Message,	Command to the server	portfolio_get_trade_history,
3	Period,	The period of trade history, only support "today" for now (*For Interactive Brokers, "all" is allowed for get past-week trade logs, as IB only provides one-week trade logs*)	today
4	End line	Indicates the end of the message	\n

Then, CAFG Protocol replies all the trade in the format of ("tradefeed") with source set as "1" to Algo Team server. *(Please refer to the format of "orderfeed" which has been described before)*

Example of portfolio_get_trade_history messages:

Algo Team -> CAFG Protocol retrieve all trade for today

```
20140122_113542_123213,portfolio_get_trade_history,today
```

Algo Team <- CAFG Protocol trade reply for the portfolio_get_trade_history

```
20140122_113542_628836,tradefeed,HKIF,HSIF4,0000002,23401,1,2,24186,1
20140122_113542_628869,tradefeed,HKIF,HSIF4,0000001,23401,1,1,24187,1
```

Ping Message

The ping message can be used to check if the connection is alive.

Ping messages format (From Algo Team->CAFG Protocol)

[testing/simulation/production]:

Field no.	Field Name	Description	Sample
1	Timestamp,	Server timestamp	20140119_091500_123213,
2	Message,	Command to the server	ping
3	End line	Indicates the end of the message	\n

Example of ping messages:

Algo Team -> CAFG Protocol ping message

```
20140122_113542_123213,ping
```

Algo Team <- CAFG Protocol ping reply

```
ping
```

Reset Message

The reset message can be used to reset the system to the original state.
Effects: Market data subscription is canceled. Working orders and trade history will be deleted. Position and PnL will be set to 0.
This message should not be used in testing and production environment.

Reset messages format (From Algo Team->CAFG Protocol)

[simulation]:

Field no.	Field Name	Description	Sample
1	Timestamp,	Server timestamp	20140119_091500_123213,
2	Message,	Command to the server	reset
3	End line	Indicates the end of the message	\n

Example of reset messages:

Algo Team -> CAFG Protocol reset message

```
20140122_113542_123213,reset
```

Algo Team <- CAFG Protocol reset reply

```
reset
```

Login Message

The login message can be used to login into the system.

Loginfeed messages format (From Algo Team->CAFG Protocol)

Field no.	Field Name	Description	Sample
1	Timestamp,	Server timestamp	20140119_091500_123213,
2	Message,	Command to the server	loginfeed,
3	Login Name,	User name	atu,
4	Login Password	User password	atu
5	End line	Indicates the end of the message	\n

Loginrspfeed messages format (From Algo Team->CAFG Protocol)

Field no.	Field Name	Description	Sample
1	Timestamp,	Server timestamp	20140119_091500_123213,
2	Message,	Command to the server	loginrspfeed,
3	Login Status,	Status = 0: successful Status > 0: failed	0,
4	Error Message,	Empty if status = 0	
5	End line	Indicates the end of the message	\n

Example of login messages:

Algo Team -> CAFG Protocol reset message

```
20140122_113542_123213,loginfeed,atu,atu
```

Algo Team <- CAFG Protocol login reply

Successful:

```
20140122_113543_123213,loginrspfeed,0,
```

Failed:

```
20140122_113543_123213,loginrspfeed,1, User or Password is not correct!
```

Remarks:

1) Only one connection to each market is allowed (multiple connection will kick the old connection)

2) Algo Team should update their order book if necessary

3) Algo Team should update their trade records if necessary

4) Algo Team should update their trade records and order records if necessary

5) Algo Team should connect to the market data server as well as the order server at the same time

Test Cases for signalfeed, orderfeed and tradefeed

Test Case	Details *Notes: "AT" refers to Algo Team "TH" to Test Host*	Actual Commands and Results
1	**Overview:** Placement of limit order(s) with validity set to today. Full execution.	
1.1	**Simulation:** TH: place the limit order with 6 contracts at 22000 ask price	
1.2	AT: insert the limit order with 6 contracts at 22000 bid price and its validity is set to today	20140604_080326_976976,signalfeed,HKIF,HSIK4,EATEST_00001,22000,6,open,1,insert,limit_order,today
1.3	*Expected Result: orderfeed update(s), orderfeed received with the 6 order filled and deleted flag marked and tradefeed received for the 6 contracts traded*	20140604_080327_014663,orderfeed,HKIF,HSIK4,EATEST_00001,22000,6,open,1,0,0,0,,0,limit_order 20140604_080327_019408,orderfeed,HKIF,HSIK4,EATEST_00001,22000,6,open,1,0,0,0,,0,limit_order 20140604_080327_040140,orderfeed,HKIF,HSIK4,EATEST_00001,22000,6,open,1,6,1,0,,0,limit_order 20140604_080327_040190,tradefeed,HKIF,HSIK4,EATEST_00001,22000,6,1,214,0
2	**Overview:** Placement of limit order(s) with validity set to fill and kill. Partial execution.	
2.1	**Simulation:** TH: place the limit order with 3 contracts at 22000 ask price	

2.2	AT: insert the limit order with 6 contracts at 22000 bid price and its validity set to fill and kill	20140604_080431_566566,signalfeed,HKIF,HSIK4,EATEST_00002,22000,6,open,1,insert,limit_order,fill_and_kill
2.3	*Expected Result: orderfeed update(s), orderfeed received with 3 order filled and deleted flag markedand tradefeed received for 3 contracts traded*	20140604_080431_597619,orderfeed,HKIF,HSIK4,EATEST_00002,22000,6,open,1,0,0,0,,0,limit_order 20140604_080431_602205,orderfeed,HKIF,HSIK4,EATEST_00002,22000,6,open,1,0,0,0,,0,limit_order 20140604_080431_622521,orderfeed,HKIF,HSIK4,EATEST_00002,22000,6,open,1,3,0,0,,0,limit_order 20140604_080431_622532,tradefeed,HKIF,HSIK4,EATEST_00002,22000,3,1,216,0 20140604_080431_641530,orderfeed,HKIF,HSIK4,EATEST_00002,22000,6,open,1,3,1,0,,0,limit_order
3	**Overview:** Placement of limit order(s) with validity set to fill or kill. No execution.	
3.1	**Simulation:** TH: place the limit order with 3 contracts at 22000 ask price	
3.2	AT: insert the limit order with 6 contracts at 22000 bid price and its validity is set to fill or kill	20140604_080511_876876,signalfeed,HKIF,HSIK4,EATEST_00003,22000,6,open,1,insert,limit_order,fill_or_kill
3.3	TH: delete the limit order	
3.4	*Expected Result: no tradefeed received, orderfeed update(s), orderfeed received with deleted flag marked and no order filled*	20140604_080511_907881,orderfeed,HKIF,HSIK4,EATEST_00003,22000,6,open,1,0,0,0,,0,limit_order 20140604_080511_912670,orderfeed,HKIF,HSIK4,EATEST_00003,22000,6,open,1,0,0,0,,0,limit_order 20140604_080511_916130,orderfeed,HKIF,

		HSIK4,EATEST_00003,22000,6,open,1,0,1,0,,0,limit_order
4	**Overview:** Placement of limit order(s) with validity set to today. Partial execution. Then, deletion of the limit order(s).	
4.1	**Simulation:** TH: place the limit order with 3 contracts at 22000 ask price and its validity is set to today	
4.2	AT: insert the limit order with 6 contracts at 22000 bid price and its validity is set to today	20140604_080737_235235,signalfeed,HKIF,HSIK4,EATEST_00004,22000,6,open,1,insert,limit_order,today
4.3	*Expected Result: orderfeed recevied tradefeed with 3 contracts received,* orderfeed update(s), *orderfeed with 3 contracts filled*	20140604_080737_266897,orderfeed,HKIF,HSIK4,EATEST_00004,22000,6,open,1,0,0,0,,0,limit_order 20140604_080737_271328,orderfeed,HKIF,HSIK4,EATEST_00004,22000,6,open,1,0,0,0,,0,limit_order 20140604_080737_280818,tradefeed,HKIF,HSIK4,EATEST_00004,22000,3,1,218,0 20140604_080737_280827,orderfeed,HKIF,HSIK4,EATEST_00004,22000,6,open,1,3,0,0,,0,limit_order
4.4	AT: delete the limit order	20140604_080742_241241,signalfeed,HKIF,HSIK4,EATEST_00004,22000,6,,1,delete,,
4.5	*Expected Result: orderfeed with deleted flag marked*	20140604_080742_269467,orderfeed,HKIF,HSIK4,EATEST_00004,22000,6,open,1,3,0,0,,0,limit_order 20140604_080742_275233,orderfeed,HKIF,HSIK4,EATEST_00004,22000,6,open,1,3,1,0,,0,limit_order