FairX | The Future of Futures

PROPRIETARY & CONFIDENTIAL

FairX

SBE Order API Specification

Version 1.3

Table of Contents

PROPRIETARY & CONFIDENTIAL

1. Revision History

Version	Date	Author	Description
0.0	Mar-5-2020	ET, KW, FY	Initial draft
0.1	Sep-5-2020	KW, FY	Changed timestamps to nanoseconds. Added SetAccount, SetTrader messages. Added execId to all outbound order event messages. Updated smart order replace protocol. Added MassCancelOrder messages. Include defaulted order attributes.
0.2	Oct-19-2020	MG	Replaced Smart/ForceReplaceOrder messages with StreamOrder message. Combined New/ReplaceOrderReject messages into OrderReject message.
0.3	Oct-20-2020	KW	Added receiveTime to OrderEntered and OrderReplaced messages.
0.4	Oct-21-2020	MG	Added isAggressor and removed secondaryExecId from OrderFilled message. Added SpreadOrderFilled message with leg fill prices. Updated Reject Reason codes
0.5	Oct-23-2020	FY	MassOrderCancel Trading Lock messages and fields TradingUnlock request messages and ack/reject messages
0.6	Nov-9-2020	JT	Minor fix to MassOrderCancel side field
0.7	Dec-1-2020	KW	Added LastExecIdRequest, LastExecId, EventResendRequest, EventResendAck and EventResendReject messages. Removed padding fields from the ends of messages.
0.8	Jan-1-2021	KW	Removed RejectReason from MassCancelOrderReject and UnlockTradingReject messages.
0.9	Jan-7-2020	FY	Add MatchId to OrderFilled and SpreadOrderFilled messages, fix field numbering. Add receiveTime to OrderCanceled
1.0	29-Jan-2021	VD	Added SMP details, Updated instrument Info
1.1	Apr-5-2021	KW	Added missing cancel reasons.
1.2	May-21-2021	KW	Added ACTIVE_LIMIT_EXCEEDED cancel reason.
1.3	Jun-6-2021	ET	Added Matching

1. Overview

This document describes Simple Binary Encoding (SBE), deployed by "FairX" as their order entry protocol used by Lead Market Makers to connect to FairX. The API allows connected firms to send, modify and cancel their orders in a simplified and efficient manner. The expected latency for the Binary API for quoting is 30-80 microseconds roundtrip.

1.1.Session Protocol

The FairX SBE API implements a subset of the FIX session protocol. Admin/session messages (Logon, Logout, Heartbeat, TestRequest, ResendRequest, GapFill) behave in much the same way as their FIX counterparts. Sequence numbers are also assigned and validated in the same manner as FIX.

1.2.Hours of Operation

Contact the exchange for the current trading schedule. Orders entered outside trading hours will be rejected. Firms are encouraged to stay connected 15 minutes after the official close to receive execution reports that are generated due to trading session closing logic (e.g. Expired reports, Done for Day). Reset schedule is configured during initial setup.

1.3.Certification

To be able to submit orders to FairX, firms must be certified. FairX provides a separate environment for integration, acceptance testing and certification. Please contact the FairX team to obtain additional information.

1.4.Trade Busts and Corrects

The FairX SBE API does not distribute unsolicited reports about trade busts and corrects. The firms are expected to utilize the FIX Drop Copy connections if they need to receive these messages.

1.5. Expiration Reports

When a day order expires at the close of trading day, an OrderCancelled message will be sent

1.6.CorrelationId

Every application message contains a correlationId, an 8-byte integer. Clients are free to use any value as the correlationId. No validation is performed by the server on this value, although a monotonically increasing value is recommended. Messages from server to client will use the correlationId of the corresponding request message from the client, or the correlationId of the last related request in the case of indirectly correlated messages such as order fill and system cancel notifications.

1.7.Byte Alignment and Message Padding

Messages are layed out so 8-byte fields start on 8-byte boundaries, 4-byte fields start on 4-byte boundaries and 2-byte fields start on 2-byte boundaries. The frame length of all outbound messages to the client will be rounded up to the nearest multiple of 8. Clients are encouraged to do the same with inbound messages, although this is not required.

1.8.Failover and Delivery Guarantees

FairX will operate a pair of active gateways. Clients are free to connect to one or the other or both. All order events for a user will be sent to both gateways and be available for resend regardless of whether or not the client has an active session. Sessions on each gateway are distinct and thus have independent sequence numbers. However, an event execId is guaranteed to uniquely identify an event and can be used to de-duplicate events received from dual sessions.

Order requests may be sent via either gateway, but should not be sent via both simultaneously. FairX may reject duplicate requests, but they may also result in duplicate fills.

If the event of a disconnect with one or more pending/unacked requests, the client should connect to the other gateway (if not already connected) and send a LastExecIdRequest to determine the execId of the last order event sent by the trading system to the client. If the client is missing events (the last seen execId is less than lastExecId returned in LastExecId message), the client may either send an EventResendRequest or a session-level ResendRequest message to retrieve missed events. Having recovered all missed events, the client can safely assume that any requests for which there was no corresponding event was either not processed by the trading system or was rejected.

1.9.Cancel On Disconnect

All orders submitted via this API are implicitly cancel-on-disconnect. If a client is connected to both active gateways and disconnects from one of them, only the orders submitted (or last replaced/modified) by the disconnected session will be canceled. For example, if a client submits an order on gateway A and then updates that order on gateway B, the order will not be canceled if they disconnect from gateway A but will be canceled if they disconnect from gateway B.

1.10. Matching

All orders submitted via the Binary API will be submitted as “Post-Only Quotes” which can interact only with orders submitted through the FIX Orders API.

PROPRIETARY & CONFIDENTIAL

2. Session

2.1. Message Header

Each message will begin with the following SBE message header containing message type information as well as message length and API version number.

Field	Name	Type	Length	Description
1	protocolId	uint8	1	Constant (= 0xF1)
2	flags	uint8	1	Bitset of flags 0x01 = resend
3	frameLength	uint16	2	Total length of frame including this header and body.
4	sequenceNumber	uint32	4	Sequence number of the message.
5	lastProcessedSeqNum	uint32	4	Sequence number of the last message received/processed by the sender of this message.
6	reserved		4	Padding
7	sendTimeEpochNanos	int64	8	Sending time in nanoseconds since epoch.
8	blockLength	uint16	2	Length of message root block in bytes, before variable data commences.
9	templateId	uint16	2	Message Type Id
10	schemaId	uint16	2	Message schema Id (1100 for admin/session messages, 1101 for orders)
11	version	uint16	2	Message version number

2.2.Logon

Each client will use the assigned IP address and port to establish a TCP/IP session with the server. The client will initiate a session at the start of each trading day by sending the Logon message. If the client does not initiate the session by sending the Logon message within two heartbeats interval of establishing the session, the connection will be dropped by the server. The client will identify itself using the Username field. The server will validate the Username and password of the client.

Field	Name	Type	Length	Description
	Logon	100	81	Sent to initiate connections
1	username	String16	16	Logon username
2	password	String32	32	Logon password
3	resetSeqNum	uint8	1	(1 = true, 0 = false)

2.3.LogonConf

Once the client is successfully authenticated, the server will respond with a LogonConf message.

Field	Name	Type	Length	Description
	LogonConf	200	36	Logon confirmation
1	heartbeatIntervalSeconds	int32	4	Heartbeat interval in seconds.

2.4.Heartbeat

The client and server will use the Heartbeat message to exercise the communication line during periods of inactivity and to verify that the interfaces at each end are available. The heartbeat interval will be three seconds. The server will send a Heartbeat anytime it has not transmitted a message for the heartbeat interval. The client is expected to employ the same logic. If the server detects inactivity for five heartbeat intervals, the server will send a Logout and break the TCP/IP connection with the client. The client is expected to employ similar logic if inactivity is detected on the part of the server

Field	Name	Type	Length	Description
	Heartbeat	10	40	Connection Heartbeat, may also be sent in response to a TestRequest
1	correlationId	int64	8	Optional id if sent in response to a TestRequest

2.5.TestRequest

The TestRequest is utilized to force a heartbeat from the opposing application. The message is useful for checking sequence numbers or verifying communication line status. The opposite application will respond to the TestRequest with a Heartbeat.

Field	Name	Type	Length	Description
	TestRequest	11	40	TestRequest to request heartbeat, receiver should respond with Heartbeat message with the provided test request id included
1	correlationId	int64	8	Correlation id to be echoed by receiver

2.6.ResendRequest

The client may send a ResendRequest to initiate retransmission of previously sent messages. Like the FIX protocol, GapFill messages will be sent in place of admin and missing/unavailable messages.

Field	Name	Type	Length	Description
	ResendRequest	102	40	Resend request
1	fromSequenceNumber	uint32	4	Sequence number of first message to resend.
2	toSequenceNumber	uint32	4	Sequence number of last message to resend (or 0 to resend all messages from formSequenceNumber).

2.7.GapFill

While retransmitting messages in response to a ResendRequest, the server will send GapFill messages in place of admin and missing/unavailable messages.

Field	Name	Type	Length	Description
	ResendRequest	202	40	Resend request
1	newSequenceNumber	uint32	4	Sequence number of the next message to be sent.
2	padding	uint32	4	

2.8.Logout

The client is expected to terminate each connection at the end of each trading day before the server shuts down. The client will terminate a connection by sending the Logout message. The server will respond with a Logout message if the client's request is successful. The client will then break the TCP/IP connection with the server. All open TCP/IP connections will be terminated by the server when it shuts down (a Logout will be sent). Under exceptional circumstances the server may initiate the termination of a connection during the trading day by sending the Logout message. Either party that wishes to terminate the connection may wait for the heartbeat interval duration before breaking the TCP/IP connection, in order to ensure that the other party received the Logout message

Field	Name	Type	Length	Description
	Logout	101	96	Logout message to gateway
1	reason	String64	64	Logout reason

2.9.LoggedOut

The server will send a LoggedOut message before terminating the connection, either in response to a Logout message from the or for other reasons.

Field	Name	Type	Length	Description
	Logout	201	96	Logout message from gateway to client
1	details	String64	64	Logout details

3. Instrument Messages

3.1. InstrumentInfoRequest

Field	Name	Type	Length	Description
	InstrumentInfoRequest	103	40	Request all instrument available
1	correlationId	int64	8	

3.2. InstrumentInfo

Sent in response to InstrumentInfoRequest message, and later when the status of an instrument changes.

Field	Name	Type	Length	Description
	InstrumentInfo	203	80	Information about an instrument
1	correlationId	int64	8	requestId for correlation
2	instrumentId	int32	4	Numeric instrumentId
3	securityType	uint8	1	SecurityType 0 = FUTURES 1 = OPTIONS
4	status	uint8	1	0 = PRE_OPEN 1 = READY_TO_TRADE 2 = TRADING_HALTED 3 = PAUSE 4 = CLOSE 5 = PRE_OPEN_NO_CANCEL 6 = EXPIRED
5	isLastMessage	int8	1	0 = not last instrument info 1 = last instrument info for requestId
5	reserved	int8	1	
6	symbol	String32	32	Instrument symbol

4. Order Messages

4.1.SetAccount

Send SetAccount message to set the account to be used for subsequent NewOrder messages.

Field	Name	Type	Length	Description
	SetAccount	105	56	Set the current account
1	correlationId	int64	8	Client-assigned ID
2	account	String16	16	0-padded ASCII string

4.2.SetTrader

Send SetTrader message to set the trader to be used for subsequent NewOrder messages.

Field	Name	Type	Length	Description
	SetTrader	106	56	Set the current trader
1	correlationId	int64	8	Client-assigned ID
2	trader	String16	16	0-padded ASCII string

4.3.SetAck

Sent in response to SetAcount and SetTrader messages.

Field	Name	Type	Length	Description
	SetAck	205	40	Acknowledgment of Set request
1	correlationId	int64	8	Client-assigned ID

4.4.NewOrder

Used to enter an order in the system.

Field	Name	Type	Length	Description
	NewOrder	110	65	Place a new order
1	clientOrderId	int64	8	Numeric client order id
2	correlationId	int64	8	Client-assigned ID
3	limitPrice	int64	8	Price with 9 decimal places
4	quantity	int32	4	Quantity with 0 decimal places
5	instrumentId	int32	4	Numeric instrumentId
6	side	int8	1	1 = BUY -1 = SELL

PROPRIETARY & CONFIDENTIAL

4.5.OrderEntered

OrderEntered messages are sent in response to a NewOrder message if successful.

Field	Name	Type	Length	Description
	OrderEntered	210	80	NewOrder acknowledgement
1	transactTime	int64	8	Nanoseconds since epoch
2	execId	int64	8	Exchange-assigned event id (akin to execId in FIX)
3	clientOrderId	int64	8	Numeric client order id
4	correlationId	int64	8	Return the client-assigned ID
5	orderId	int64	8	Numeric exchange-assigned order id
6	receiveTime	int64	8	Nanoseconds since Unix epoch when we received the NewOrder message on gateway.

4.6.ReplaceOrder

The ReplaceOrder message allows you to alter the price and quantity of an order in a single message. This is more efficient than canceling an existing order and immediately succeeding it with a new orderRequest to modify an order. The order will be cancelled if newQuantity is less than or equal the current total filled quantity.

Field	Name	Type	Length	Description
	ReplaceOrder	120	64	Replace an order
1	clientOrderId	int64	8	Numeric client order id
2	correlationId	int64	8	Client-assigned ID
3	newLimitPrice	int64	8	Price with 9 decimal places

4	newQuantity	int32	4	Quantity with 0 decimal places
5	instrumentId	int32	4	Numeric instrumentId

4.7.StreamOrder

A StreamOrder will modify an existing order with the same ClientOrderId if it is active and has no fills; OrderReplaced will be sent in response. If the existing order is partially filled, the remaining balance will be canceled and a new order will be entered; OrderCanceled will be sent for the existing order, followed by OrderEntered for the new order. If the existing order has been fully filled or otherwise not found, a new order will be placed and OrderEntered will be sent in response. The quantity of the new order will be newQuantity less any fills that occurred on the existing order after lastProcessedExecId.

Example:

```
-> StreamOrder (clOrdId=1234, corrId=100, qty=10)
<- OrderEntered (corrId=100, orderId=1, execId=1, qty=10)

-> StreamOrder (clOrdId=1234, corrId=101, lastProcessedExecId=1, qty=9)
<- OrderReplaced (corrId=101, orderId=1, execId=2, qty=9)

-> StreamOrder (clOrdId=1234, corrId=102, lastProcessedExecId=2, qty=10)
- Before StreamOrder with corrId=102 processed, order fills for 5
<- OrderFilled (clOrdId=1234, corrId=101, orderId=1, execId=3, fillQty=5)
<- OrderCanceled (clOrdId=1234, corrId=101, orderId=1, execId=4)
<- OrderEntered (corrId=102, orderId=5, execId=5, qty=5)

-> StreamOrder (clOrdId=1234, corrId=103, lastProcessedExecId=5, qty=10)
<- OrderReplaced (corrId=103, orderId=5, execId=6, qty=10)

- Order aggressed for 4.
<- OrderFilled (clOrdId=1234, corrId=103, orderId=5, execId=7, fillQty=4)

-> StreamOrder (clOrdId=1234, corrId=104, lastProcessedExecId=7, qty=10)
<- OrderReplaced (corrId=104, orderId=5, execId=8, qty=10)

- Order aggressed for 10.
<- OrderFilled (clOrdId=1234, corrId=104, orderId=5, execId=9, fillQty=10)

-> StreamOrder (clOrdId=1234, corrId=105, lastProcessedExecId=9, qty=8)
<- OrderEntered (corrId=105, orderId=10, execId=10, qty=8)
```

A StreamOrder submitted with lastProcessedExecId=0 will be entered without the potential reduction of quantity based on recent fills of the existing order. Thus, the existing order will be amended if it is active and has no fills, with OrderReplaced sent in response. Otherwise the existing order will be canceled if active but partially filled (and OrderCanceled sent in response), and a new order will be entered.

Field	Name	Type	Length	Description
	StreamOrder	121	73	Enter a stream order
1	clientOrderId	int64	8	Numeric client order id
2	correlationId	int64	8	Client-assigned ID
3	lastProcessedExecId	int64	8	Last ExecId received by the client If set to 0, forces replace without quantity adjustment
4	limitPrice	int64	8	Price with 9 decimal places
5	quantity	int32	4	Quantity with 0 decimal places
6	instrumentId	int32	4	Numeric instrumentId
7	side	int8	1	1 = BUY -1 = SELL

4.8. OrderReject

OrderReject messages are sent in response to NewOrder, ReplaceOrder and StreamOrder if the request is rejected.

Field	Name	Type	Length	Description
	OrderReject	221	112	Reject message for NewOrder, ReplaceOrder and StreamOrder
1	transactTime	int64	8	Nanoseconds since epoch
2	clientOrderId	int64	8	Client order id
3	correlationId	int64	8	Return the client-assigned ID
4	orderId	int64	8	Exchange assigned order id Set to 0 if order is unknown
5	rejectReason	uint8	1	1 = ERROR 2 = INVALID_INSTRUMENT 3 = CL_ORD_ID_IN_USE 4 = VALIDATION_FAILURE 5 = UNKNOWN_ORDER
6	details	char	47	Null (0) padded string

4.9.OrderReplaced

This message acknowledges the receipt and acceptance of a valid ReplaceOrder, as well as StreamOrder if the existing order had no fills.

Field	Name	Type	Length	Description
	OrderReplaced	220	92	Reply to ReplaceOrder after success
1	transactTime	int64	8	Nanoseconds since epoch
2	execId	int64	8	Exchange-assigned event id
3	clientOrderId	int64	8	Client order id
4	correlationId	int64	8	Return the client-assigned ID
5	orderId	int64	8	Exchange-assigned order id
6	receiveTime	int64	8	Nanoseconds since Unix epoch when we received the replace/update message on gateway.
7	totalFilled	int32	4	Filled amount
8	availableQty	int32	4	Remaining quantity available for matching
9	instrumentId	int32	4	

4.10. CancelOrder

The CancelOrder message is used to request that an order be canceled.

Field	Name	Type	Length	Description
	CancelOrder	130	52	Cancel an Order
1	clientOrderId	int64	8	Numeric client order id
2	correlationId	int64	8	Client-assigned id
3	instrumentId	int32	4	instrumentId

4.11. MassCancelOrder

The MassCancelOrder message is used to request multiple orders be canceled matching various optional criteria. InstrumentId and side are optional values, but both must be specified if limit price is specified; will cancel buys with limit prices at or above the specified limit, or sells with limit prices at or below the specified limit. 'currentSessionOnly' flag is used to specify that only orders submitted via the current session should be considered versus all sessions of the client's firm. The 'requestTradingLock' flag can be used to thereafter reject subsequent orders until a TradingUnlock request is sent.
OrderCanceled messages will be sent for each order canceled, followed by a MassCancelOrderAck message.

Field	Name	Type	Length	Description
	MassCancelOrder	131	55	Cancel multiple Orders
1	correlationId	int64	8	Client-assigned id
2	limitPrice	int64	8	Optional price Null value = 0xffffffff_ffffffffL
3	instrumentId	int32	4	Optional instrumentId Null value = 0xffffffff
4	side	int8	1	-128 = both sides 1 = BUY -1 = SELL
5	currentSessionOnly	int8	1	1 = orders of current session only 0 = orders of all sessions of Firm
6	requestTradingLock	int8	1	1 = lock trading for all sessions under cancel scope 0 = just cancel orders without engaging trading lock

4.12. OrderCanceled

An OrderCanceled Message informs you that an order has been canceled. This could be acknowledging a Cancel Order Message, or it could be the result of the order timing out or being canceled automatically.

Field	Name	Type	Length	Description
	OrderCanceled	230	81	Sent when an order is canceled
1	transactTime	int64	8	Nanoseconds since epoch
2	execId	int64	8	Exchange-assigned event id
3	clientOrderId	int64	8	Client order id
4	correlationId	int64	8	Return the client-assigned ID
5	orderId	int64	8	Exchange-assigned order id
6	receiveTime	int64	8	Nanoseconds since Unix epoch when we received the cancel message on gateway. Null value = 0xffffffff_ffffffffL for unsolicited cancels
6	totalFilled	int32	4	Filled amount
7	instrumentId	int32	4	Numeric instrumentId
8	cancelReason	uint8	1	0 = EXPIRED 1 = CANCELED_BY_USER 2 = SELF_MATCH_PREVENTION 3 = CLIENT_DISCONNECT 4 = PRICE_LIMIT 5 = ADMIN_CANCEL 6 = MASS_CANCEL 7 = STREAM_REPLACE 8 = ACTIVE_LIMIT_EXCEEDED

4.13. CancelOrderReject

A CancelOrderReject message may be sent in response to a CancelOrder if the cancel cannot be accepted at this time.

Field	Name	Type	Length	Description
	CancelOrderReject	233	88	Reject message for order replace
1	transactTime	int64	8	Nanoseconds since epoch
2	clientOrderId	int64	8	Client order id
3	correlationId	int64	8	Return the client-assigned ID
4	orderId	int64	8	Exchange assigned order id
5	rejectReason	uint8	1	1 = ERROR 2 = UNKNOWN_ORDER 3 = ORDER_FILLED
6	details	char	23	Null (0) padded string

4.14. MassCancelOrder

The MassCancelOrder message is used to request multiple orders be canceled matching various optional criteria. InstrumentId and side are optional values, but both must be specified if limit price is specified; will cancel buys with limit prices at or above the specified limit, or sells with limit prices at or below the specified limit. 'currentSessionOnly' flag is used to specify that only orders submitted via the current session should be considered versus all sessions of the client's firm. The 'requestTradingLock' flag can be used to thereafter reject subsequent orders until a TradingUnlock request is sent.
OrderCanceled messages will be sent for each order canceled, followed by a MassCancelOrderAck message.

Field	Name	Type	Length	Description
	MassCancelOrder	131	55	Cancel multiple Orders
1	correlationId	int64	8	Client-assigned id
2	limitPrice	int64	8	Optional price Null value = 0x80000000_00000000L
3	instrumentId	int32	4	Optional instrumentId Null value = 0x80000000
4	side	int8	1	-128 = both sides 1 = BUY -1 = SELL
5	currentSessionOnly	int8	1	1 = orders of current session only 0 = orders of all sessions of Firm
6	requestTradingLock	int8	1	1 = lock trading for all sessions under cancel scope 0 = just cancel orders without engaging trading lock

4.15. MassCancelOrderAck

The MassCancelOrderAck message is sent in response to a MassCancelOrder message following OrderCanceled messages for each canceled order. If a tradingLock was requested, the scope of sessions affected will be reported as well.

Field	Name	Type	Length	Description
	MassCancelOrderAck	231	62	Acknowledgement of MassOrderCancel request.
1	transactTime	int64	8	Nanoseconds since epoch
2	execId	int64	8	Exchange-assigned event id
3	correlationId	int64	8	Client-assigned id
4	canceledCount	int32	4	Total number of orders canceled
5	onlyCurrentSession	int8	1	1 = only current session affected 0 = all sessions under Firm affected
6	tradingLockApplied	int8	1	1 = trading lock applied 0 = not applied

4.16. MassCancelOrderReject

A MassCancelOrderReject message may be sent in response to a MassCancelOrder if the mass cancel cannot be accepted at this time.

Field	Name	Type	Length	Description
	MassCancelOrderReject	232	80	Reject message for order replace
1	transactTime	int64	8	Nanoseconds since epoch
2	correlationId	int64	8	Return the client-assigned ID
3	errorMessage	char	32	Null (0) padded string

4.17. UnlockTrading

Sent to unlock trading after a MassOrderCancel has been sent when the flag 'requestTradingLock'. This message will disengage all client requested trading locks on all session under the firm unless 'currentSessionOnly' is set, in which case will only disengage trading locks on the current session. Trading locks will reject all incoming orders until disengaged by either client through this message or exchange administrator.

Field	Name	Type	Length	Description
	UnlockTrading	132	41	Unlock Client Requested Trading Lock
1	correlationId	int64	8	Client-assigned id
2	currentSessionOnly	int8	1	1 = unlock trading for current session only 0 = unlock trading for all sessions of Firm

4.18. UnlockTradingAck

An UnlockTradingAck Message reports the result of an unlock trading request, including the number of users affected.

Field	Name	Type	Length	Description
	UnlockTradingAck	234	60	Acknowledgement of UnlockTrading
1	transactTime	int64	8	Nanoseconds since epoch
2	execId	int64	8	Exchange-assigned event id
3	correlationId	int64	8	Client-assigned id
4	numUsersAffected	int32	4	Total number of users unlocked

4.19. UnlockTradingReject

Reports a reject of a requested trading unlock

Field	Name	Type	Length	Description
	UnlockTradingReject	235	80	Reject of UnlockTrading
1	transactTime	int64	8	Nanoseconds since epoch
2	correlationId	int64	8	Client-assigned id
3	errorMessage	char	32	Null (0) padded string

4.20. OrderFilled

An OrderFilled Message informs you that all or part of an outright order has been executed

Field	Name	Type	Length	Description
	OrderFilled	240	113	Sent when an order is partially or fully filled
1	transactTime	int64	8	Nanoseconds since epoch
2	execId	int64	8	Exchange-assigned event id
3	matchId	int64	8	Transaction id representing match, shared by all fills within match
4	clientOrderId	int64	8	Client order id
5	correlationId	int64	8	Return the client-assigned ID
6	orderId	int64	8	Exchange assigned order id
7	filledVwap	int64	8	Filled vwap with 9 decimal places
8	totalFilled	int32	4	Filled amount
9	availableQty	int32	4	Remaining quantity available for matching
10	fillPrice	int64	8	Price filled with 9 decimals
11	fillQty	int32	4	Fill quantity
12	instrumentId	int32	4	Numeric instrumentId
13	isAggressor	uint8	1	0 - False 1 - True

4.21. SpreadOrderFilled

A SpreadOrderFilled Message informs you that all or part of a spread order has been executed

Field	Name	Type	Length	Description
	SpreadOrderFilled	241	129	Sent when an order is partially or fully filled
1	transactTime	int64	8	Nanoseconds since epoch
2	execId	int64	8	Exchange-assigned event id
3	matchId	int64	8	Transaction id representing match, shared by all fills within match
4	clientOrderId	int64	8	Client order id
5	correlationId	int64	8	Return the client-assigned ID
6	orderId	int64	8	Exchange assigned order id
7	filledVwap	int64	8	Filled vwap with 9 decimal places
8	totalFilled	int32	4	Filled amount
9	availableQty	int32	4	Remaining quantity available for matching
10	fillPrice	int64	8	Price filled with 9 decimals
11	leg1fillPrice	int64	8	Price filled with 9 decimals on underlying leg 1
12	leg2fillPrice	int64	8	Price filled with 9 decimals on underlying leg 2
13	fillQty	int32	4	Fill quantity
14	instrumentId	int32	4	Numeric instrumentId
15	isAggressor	uint8	1	0 - False 1 - True

4.22. LastExecIdRequest

Send this message to request the execId of the last (most recent) event sent by the trading system to this user/session. Can be used to determine if the client missed any events while disconnected. Also serves as a means of validating that the trading system is available and accepting requests.

Field	Name	Type	Length	Description
	LastExecIdRequest	150	40	Request execId of last event sent to user
1	correlationId	int64	8	Client-assigned id

4.23. LastExecId

Sent in response to LastExecIdRequest.

Field	Name	Type	Length	Description
	LastExecId	250	56	Response to LastExecIdRequest
1	timestamp	int64	8	Nanoseconds since epoch
2	lastExecId	int64	8	ExecId of last event sent to this user.
3	correlationId	int64	8	Client-assigned id

4.24. EventResendRequest

Send this message to request order events in the specified range be resent. Since this is an application-level request, resent messages will have new sequence numbers and the resend flag in the message will not be set. Rejects (and any other message that does not contain an execId) will not be resent.

Field	Name	Type	Length	Description
	EventResendRequest	152	56	Response to LastExecIdRequest
1	correlationId	int64	8	Client-assigned id
2	beginExecId	int64	8	Lower bound (inclusive) of execIds.
3	endExecId	int64	8	Upper bound (inclusive) of execIds if positive. Resend all events up the last known event if non-positive.

4.25. EventResentComplete

Sent in response to a successful EventResendRequest following all resent events.

Field	Name	Type	Length	Description
	EventResendComplete	252	44	Sent on fulfillment of an EventResendRequest
1	correlationId	int64	8	Client-assigned id
2	resentEventCount	int32	4	Total number of events resent.

4.26. EventResendReject

Sent in response to an EventResendRequest if the request cannot be fulfilled.

Field	Name	Type	Length	Description
	EventResendReject	253	64	Reject an EventResendRequest
1	correlationId	int64	8	Client-assigned id
2	rejectReason	uint8	1	1 = BEGIN_EXEC_ID_TOO_SMALL 2 = END_EXEC_ID_TOO_LARGE 3 = RESEND_ALREADY_IN_PROGRESS 4 = TOO_MANY_RESEND_REQUESTS 5 = SERVER_ERROR
3	details	char	23	Null (0) padded string

4.27. Default session values:

During session creation the following values will be defaulted for the Liquidity Provider

Field	Name	Type	Length	Description
	CustOrderCapacity			CTI Code
	Manual or Automated			"N" indicates the message was generated by automated trading logic.
	OrderCapacity			The type of business conducted. 0 = Customer/Agency 1 = Principal

	CustomerOrderHandlingInst			W = Desk
				Y = Electronic
				C = Vendor provided
				G = Sponsored Access
				H = Premium Algorithmic trading
				D = Other

	Self Match Prevention ID			Orders with the same Self Match Prevention ID for the same executing firm will not match
	Self Match Prevention Strategy			This value defines the strategy of dealing with matching orders if self-match prevention is triggered. The exchange can either cancel the aggressor order, the resting order or both