FairX | The Future of Futures

PROPRIETARY & CONFIDENTIAL

FairX

Multicast UDP Market Data API Specification

Version 1.2

Table of Contents

1. Overview

This document describes the FairX multicast UDP market data API. The API allows market participants to receive market information including orders, trades, market state changes and instrument definitions. The API consists of multiple groups of data channels/streams for distinct sets of related products and instruments. Each group is comprised of 3 pairs of channels:

- **Incremental Updates** - A/B UDP multicast groups with real-time updates for orders, trades, market state changes, and instrument definitions.
- **Snapshots** - A/B UDP multicast groups with periodic snapshots of orders and instrument definitions/statuses at a regular interval.
- **Retransmission Service** - UDP unicast with request/response model for updates within specified InstrSeqNum range.

The API is a binary protocol based on the Simple Binary Encoding (SBE) protocol. Each UDP packet begins with a packet header preceding zero or more concatenated SBE messages. This document describes the messages supported by FairX.

1.1. The FairX Hours of Operation

Contact the exchange for the current trading schedule.

1.2. Certification

In order to connect 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.

2. Message Structure

Messages are encoded with Simple Binary Encoding format with Little-endian byte ordering. A UDP packet can contain zero or more messages, up to a maximum length of 1400 bytes.

2.1. Packet Structure

All packets across all channels start with a packet header followed by zero or messages, up to a maximum length of 1400 bytes. The packet header has the following structure:

Packet Header	Type	Length	Offset	Description
SendingTime	int64	8	0	Nanoseconds since epoch
SeqNum	int64	8	8	Sequence number. Different based on context: Incremental - sequence generated from trading system, never reset Snapshot - sequence associated with last incremental update Retransmit reject - sequence sent by client, acts as a correlation id
ChannelId	uint16	2	16	Channel identifier for a ProductCode/Instrument set.
PktFlags	uint8	1	18	Bitset 0x01: incremental update 0x02: snapshot 0x04: retransmit
PktMessageCount	uint8	1	19	Count of messages within packet
SnapshotInstrumentI d	int32	4	20	Instrument id of messages in a snapshot packet (not used for incrementals)

2.2. Message Header

Each message in a packet starts with the following header:

Message Header	Type	Length	Offset	Description
FrameLength	uint16	2	0	Total message size in bytes including this header Includes any repeating groups or var length data Also used to indicate empty space at end of message data for byte alignment purposes
BlockLength	uint16	2	2	Total length of message body in bytes excluding this header and any repeating groups or variable length field
TemplateId	uint16	2	4	Message template identifier, specified for each message type in spec within parenthesis '(##)'
SchemaId	uint16	2	6	Identifier of message schema containing the template. Constant: 1201
Version	uint16	2	8	Version of message schema

Message versioning follows standard SBE versioning practices. New versions will remain backwards compatible with older clients. Existing fields will never be removed or modified, but new fields and messages may be added.

3. Incremental Channel Messages

Incremental update messages are sent on dual A/B multicast channels. Each message has a long (8-byte) monotonically increasing sequence number that is never reset. The stream of messages on the A/B channels are identical; however, the grouping of messages into packets may vary between A and B channels.

The packet header sequence number will be the sequence number of the first message in the packet, or the next expected sequence number if the packet contains no messages (a heartbeat packet). Thus, the expected sequence number of the next packet is always the current packet sequence number plus the current packet message count.

Heartbeat packets will be sent in the absence of new updates every 5 seconds. This is represented by a packet header with PktMessageCount=0

All incremental messages are sent in a transaction. A transaction contains all incremental messages that are published as a result of a single inbound message or event in the trading system. Single-message transactions will have both the start-of-transaction and end-of-transaction flags set.

3.1. Instrument Header

All incremental messages contain the following component as the first field after the message header:

Instrument Header	Type	Length	Offset	Description
Flags	uint8	1	0	Message header bitset 0x01 - start of transaction 0x02 - end of transaction 0x04 - clear book (reserved for future use)
Side	int8	1	1	1 - Buy -1 - Sell 0 - opening fill -128 - null value (used in messages in which side is not applicable)
InstrumentId	int32	4	2	Instrument identifier
InstrSeqNum	uint32	4	6	Per-instrument sequence number. Reset each trading day
TradingSessionDate	int16	2	10	Days since Unix epoch
Padding2	int16	2	12	2 bytes of padding (reserved for future use)
TransactTime	int64	8	14	Event timestamp - nanoseconds since Unix epoch

Side will null (-128) unless otherwise noted for particular messages.

3.2. Outright Instrument Definition

Outright Instrument Definition (10)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
Symbol	char24	24	32	Instrument name or symbol
ProductCode	char8	8	56	Code of underlying product/asset Example: TEC (Nano SuperTech Fut)
Description	char32	32	64	Instrument name
PriceIncrement	int64	8	96	Minimum constant tick for instrument, encoded with 9 decimal places
CfiCode	char8	8	104	ISO standard instrument categorization code
Currency	char8	8	112	Currency used for price
FirstTradingSessionDate	uint16	2	120	Days since Unix epoch
LastTradingSessionDate	uint16	2	122	Days since Unix epoch
ContractSize	int32	4	124	Contract size encoded with 0 decimal places
PriorSettlementPrice	int64	8	128	Price encoded with 9 decimal places
SettlementPrice	int64	8	136	Price encoded with 9 decimal places
LimitDownPrice	int64	8	144	Minimum price that an instrument may currently trade at
LimitUpPrice	int64	8	152	Maximum price that an instrument may currently trade at
ProductId	int32	4	160	Product identifier
ProductGroup	uint8	1	164	0 - Currency 1 - Equity 2 - Energy 3 - Metals 4 - Interest Rate 5 - Agriculture
TradingStatus	uint8	1	165	Trading session status 0 - Pre-open 1 - Open 2 - Halt 3 - Pause 4 - Close 5 - Pre-open (No Cancel) 6 - Expired
InstrumentDefinitionFlags	uint16	2	166	Bitset 0x01 - isPriorSettlementTheoretical

3.3. Spread Instrument Definition

Spread Instrument Definition (11)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
Symbol	char24	24	32	Instrument name or symbol
ProductCode	char8	8	56	Code of underlying product/asset Example: TEC (Nano SuperTech Fut)
Description	char32	32	64	Instrument name
PriceIncrement	int64	8	96	Minimum constant tick for instrument, encoded with 9 decimal places
CfiCode	char8	8	104	ISO standard instrument categorization code
Currency	char8	8	112	Currency used for price
FirstTradingSessionDate	uint16	2	120	Days since Unix epoch
LastTradingSessionDate	uint16	2	122	Days since Unix epoch
ContractSize	int32	4	124	Contract size encoded with 0 decimal places
PriorSettlementPrice	int64	8	128	Price encoded with 9 decimal places
SettlementPrice	int64	8	136	Price encoded with 9 decimal places
LimitDownPrice	int64	8	144	Minimum price that an instrument may currently trade at
LimitUpPrice	int64	8	152	Maximum price that an instrument may currently trade at
ProductId	int32	4	160	Product identifier
ProductGroup	uint8	1	164	0 - Currency 1 - Equity 2 - Energy 3 - Metals 4 - Interest Rate 5 - Agriculture
TradingStatus	uint8	1	165	Trading session status 0 - Pre-open 1 - Open 2 - Halt 3 - Pause 4 - Close 5 - Pre-open (No Cancel) 6 - Expired
Leg1InstrumentId	int32	4	166	Instrument identifier for near leg
Leg2InstrumentId	int32	4	170	Instrument identifier for far leg
SpreadBuyConvention	int8	1	174	1 - Use far leg as bid -1 - Use near leg as bid

InstrumentDefinitionFlags	uint16	2	175	Bitset 0x01 - isPriorSettlementTheoretical

3.4. Trading Status Update

Trading Status Update (17)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
LimitDownPrice	int64	8	32	Minimum price that an instrument may currently trade at
LimitUpPrice	int64	8	40	Maximum price that an instrument may currently trade at
TradingStatus	uint8	1	48	Trading session status 0 - Pre-open 1 - Open 2 - Halt 3 - Pause 4 - Close 5 - Pre-open (No Cancel) 6 - Expired

3.5. Order Put

Sent when a resting order is added or updated. OrderId is unique across the entire channel and across time, but not necessarily across different channels. An orderId will never be reused at a later point for an unrelated order. Side of order is specified in the instrument header (1 or -1).

Order Put (20)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
OrderId	int64	8	32	Unique identifier for order
Price	int64	8	40	Price encoded with 9 decimal places
Quantity	int32	4	48	Quantity encoded with 0 decimal places

3.6. Order Delete

Side of deleted order is specified in instrument header (1 or -1).

Order Delete (21)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
OrderId	int64	8	32	Unique identifier for order

3.7. Implied Order Update

Side is specified in instrument header (1 or -1).

Implied Order Update (22)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
BestPrice	int64	8	32	First level implied price encoded with 9 decimal places Null price encoded as 0x8000000000000000
NextPrice	int64	8	40	Second level implied price encoded with 9 decimal places Null price encoded as 0x8000000000000000
BestQty	int32	4	48	First level implied quantity encoded with 0 decimal places
NextQty	int32	4	52	Second level implied quantity encoded with 0 decimal places

3.8. Trade Summary

Summarizes all fills of an aggressor order. Sent before individual trade messages, as well as order put/delete and market stat messages. Aggressor side specified in instrument header.

Trade Summary (33)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
AggressorOrderId	int64	8	32	Order identifier of aggressing order
AggressorReceiveTime	int64	8	40	Nanoseconds since Unix epoch when we received aggressor new/replace order message on gateway
VwapPrice	int64	8	48	Volume weighted average price encoded with 9 decimal places Null price encoded as 0x8000000000000000
DeepestPrice	int64	8	56	Price of deepest/last resting order that an aggressing order matched
Quantity	int32	4	64	Quantity encoded with 0 decimal places

3.9. Trade

Aggressor side specified in instrument header.
Trade message does not implicitly delete or update matched resting order; a separate OrderPut or OrderDelete will be sent (in the same transaction).

Trade (30)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
MatchId	int64	8	32	Transaction id representing match, shared by all trades within match
BuyOrderId	int64	8	40	Unique identifier for trade buy order In case of implied order, encoded as 0x8000000000000000
SellOrderId	int64	8	48	Unique identifier for trade sell order In case of implied order, encoded as 0x8000000000000000
Price	int64	8	56	Price encoded with 9 decimal places
Quantity	int32	4	64	Quantity encoded with 0 decimal places

3.10. Trade Amend

Trade Amend (31)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
MatchId	int64	8	32	Transaction id representing match, shared by all trades within match
BuyOrderId	int64	8	40	Unique identifier for trade buy order In case of implied order, encoded as 0x8000000000000000
SellOrderId	int64	8	48	Unique identifier for trade sell order In case of implied order, encoded as 0x8000000000000000
OldPrice	int64	8	56	Price encoded with 9 decimal places
NewPrice	int64	8	64	Price encoded with 9 decimal places

3.11. Trade Bust

Trade Bust (32)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
MatchId	int64	8	32	Transaction id representing match, shared by all trades within match
BuyOrderId	int64	8	40	Unique identifier for trade buy order In case of implied order, encoded as 0x8000000000000000
SellOrderId	int64	8	48	Unique identifier for trade sell order In case of implied order, encoded as 0x8000000000000000

3.12. Spread Trade Amend

Spread Trade Amend (34)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
MatchId	int64	8	32	Transaction id representing match, shared by all trades within match
BuyOrderId	int64	8	40	Unique identifier for trade buy order In case of implied order, encoded as 0x8000000000000000
SellOrderId	int64	8	48	Unique identifier for trade sell order In case of implied order, encoded as 0x8000000000000000
OldPrice	int64	8	56	Price encoded with 9 decimal places
NewPrice	int64	8	64	Price encoded with 9 decimal places
OldLeg1Price	int64	8	72	Spread leg price encoded with 9 decimal places
NewLeg1Price	int64	8	80	Spread leg price encoded with 9 decimal places
OldLeg2Price	int64	8	88	Spread leg price encoded with 9 decimal places
NewLeg2Price	int64	8	96	Spread leg price encoded with 9 decimal places

3.13. Market Stat

Market Stat (40)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
Price	int64	8	32	Price encoded with 9 decimal places
StatType	char	1	40	4 - Day Opening Price 5 - Closing Price 6 - Settlement Price 7 - Trading Session High Price 8 - Trading Session Low Price F - Reference Price I - Initial Opening Price

3.14. Trade Session Volume

Trade Session Volume (41)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
VwapPrice	int64	8	32	Volume weighted average price encoded with 9 decimal places Null price encoded as 0x8000000000000000
TradeVolume	int32	4	40	Total day traded volume for instrument as of the last trade in message

3.15. Open Interest

Open Interest (42)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
<Instrument Header>	<InstHeader>	22	10	Common instrument header
Quantity	int32	4	32	Quantity encoded with 0 decimal places

4. Snapshot Channel Messages

Per-instrument snapshots are sent continuously on the dual A/B snapshot channels. The frequency of repeated snapshots is intentionally not specified, but the maximum delay between snapshots (for different instruments) will be 5 seconds.

A single (instrument) snapshot will consist of 2 or more messages, starting with instrument-type-specific start message which includes the instrument definition, followed by an order snapshot message for each active order of the instrument, followed by an EndOfSnapshot message. The snapshot messages may be split across multiple packets if the full snapshot will not fit in a single packet. All packets of the snapshot will share the same packet sequence number, which corresponds to the most recent incremental feed sequence number included in the snapshot.

All snapshot messages have SnapshotSeqNum as their first field. This sequence number always starts at 0 for the first message of a snapshot and will not wrap around. A single instrument will not have more than 65534 active orders.

4.1. Start of Outright Instrument Snapshot

Start Of Outright Instrument Snapshot (110)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
SnapshotSeqNum	uint16	2	10	Sequence number of message within snapshot
LastInstrSeqNum	uint32	4	12	Snapshot incorporates all incremental messages with instrSeqNum up and including this value
Symbol	char24	24	16	Instrument name or symbol
ProductCode	char8	8	40	Code of underlying product/asset Example: TEC (Nano SuperTech Fut)
Description	char32	32	48	Instrument name
PriceIncrement	int64	8	80	Minimum constant tick for instrument, encoded with 9 decimal places
CfiCode	char8	8	88	ISO standard instrument categorization code
Currency	char8	8	96	Currency used for price
ProductId	int32	4	104	Product identifier
ContractSize	int32	4	108	Contract size encoded with 0 decimal places
OrderCount	int32	4	112	Number of orders in snapshot
FirstTradingSessionDate	uint16	2	116	Days since Unix epoch
LastTradingSessionDate	uint16	2	118	Days since Unix epoch
TradingSessionDate	int16	2	120	Days since Unix epoch
ProductGroup	uint8	1	122	0 - Currency 1 - Equity 2 - Energy 3 - Metals 4 - Interest Rate 5 - Agriculture
TradingStatus	uint8	1	123	Trading session status 0 - Pre-open 1 - Open 2 - Halt 3 - Pause 4 - Close 5 - Pre-open (No Cancel) 6 - Expired

4.2. Start of Spread Instrument Snapshot

Start Of Spread Instrument Snapshot (111)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
SnapshotSeqNum	uint16	2	10	Sequence number of message within snapshot
LastInstrSeqNum	uint32	4	12	Snapshot incorporates all incremental messages with instrSeqNum up and including this value
Symbol	char24	24	16	Instrument name or symbol
ProductCode	char8	8	40	Code of underlying product/asset Example: TEC (Nano SuperTech Fut)
Description	char32	32	48	Instrument name
PriceIncrement	int64	8	80	Minimum constant tick for instrument, encoded with 9 decimal places
CfiCode	char8	8	88	ISO standard instrument categorization code
Currency	char8	8	96	Currency used for price
ProductId	int32	4	104	Product identifier
ContractSize	int32	4	108	Contract size encoded with 0 decimal places
OrderCount	int32	4	112	Number of orders in snapshot
FirstTradingSessionDate	uint16	2	116	Days since Unix epoch
LastTradingSessionDate	uint16	2	118	Days since Unix epoch
TradingSessionDate	int16	2	120	Days since Unix epoch
ProductGroup	uint8	1	122	0 - Currency 1 - Equity 2 - Energy 3 - Metals 4 - Interest Rate 5 - Agriculture
TradingStatus	uint8	1	123	Trading session status 0 - Pre-open 1 - Open 2 - Halt 3 - Pause 4 - Close 5 - Pre-open (No Cancel) 6 - Expired
Leg1InstrumentId	int32	4	124	Instrument identifier for near leg
Leg2InstrumentId	int32	4	128	Instrument identifier for far leg
SpreadBuyConvention	int8	1	132	1 - Use far leg as bid -1 - Use near leg as bid

4.3. Order Snapshot

Order Snapshot (120)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
SnapshotSeqNum	uint16	2	10	Sequence number of message within snapshot
SignedQuantity	int32	4	12	Signed quantity with 0 decimal places. Positive value = buy; negative value = sell
TransactTime	int64	8	16	Event timestamp - nanoseconds since Unix epoch
OrderId	int64	8	24	Unique identifier for order
Price	int64	8	32	Price encoded with 9 decimal places

4.4. End of Snapshot

End Of Snapshot (122)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
SnapshotSeqNum	uint16	2	10	Sequence number of message within snapshot
TradeVolume	int32	4	12	Total day traded volume for instrument as of the last trade in message
IndicativeOpenPrice	int64	8	16	Price encoded with 9 decimal places
DayOpenPrice	int64	8	24	Price encoded with 9 decimal places
ClosePrice	int64	8	32	Price encoded with 9 decimal places
LowPrice	int64	8	40	Price encoded with 9 decimal places
HighPrice	int64	8	48	Price encoded with 9 decimal places
VwapPrice	int64	8	56	Volume weighted average price encoded with 9 decimal places Null price encoded as 0x8000000000000000
SettlementPrice	int64	8	64	Price encoded with 9 decimal places
LastTradePrice	int64	8	72	Price encoded with 9 decimal places
LastTradeTime	int64	8	80	Nanoseconds since Unix epoch
BestBidImpliedPrice	int64	8	88	First level implied price encoded with 9 decimal places Null price encoded as 0x8000000000000000
BestAskImpliedPrice	int64	8	96	First level implied price encoded with 9 decimal places Null price encoded as 0x8000000000000000
NextBidImpliedPrice	int64	8	104	Second level implied price encoded with 9 decimal places Null price encoded as 0x8000000000000000
NextAskImpliedPrice	int64	8	112	Second level implied price encoded with 9 decimal places Null price encoded as 0x8000000000000000
LimitDownPrice	int64	8	120	Minimum price that an instrument may currently trade at
LimitUpPrice	int64	8	128	Maximum price that an instrument may currently trade at
LastTradeQty	int32	4	136	Quantity encoded with 0 decimal places
OpenInterest	int32	4	140	The total open interest for the market at the close of the prior trading session
BestBidImpliedQty	int32	4	144	First level implied quantity encoded with 0 decimal places
BestAskImpliedQty	int32	4	148	First level implied quantity encoded with 0 decimal places
NextBidImpliedQty	int32	4	152	Second level implied quantity encoded with 0 decimal places
NextAskImpliedQty	int32	4	156	Second level implied quantity encoded with 0 decimal places
PriorSettlementPrice	int64	8	160	Price encoded with 9 decimal places
InstrumentDefinitionFlags	uint16	2	168	Bitset 0x01 - isPriorSettlementTheoretical

5. Retransmit Channel Messages

A retransmit request can be sent to recover lost messages. The request message shares the same packet header as all channels/messages. The client may set the packet sequence number to any value. If the retransmit request is rejected, the response packet sequence number will be set to the request packet sequence number, and can be used as a correlationId. If the retransmit request is successful, a single UDP packet will be sent in response, containing messages beginning from BeginSeqNum. The number of messages in the packet will be the lesser of the number of requested messages or whatever number will fit within the maximum packet size.

Note that only one retransmit request is supported per packet. Additional messages in the same packet will be ignored.

5.1. Retransmit Request

Retransmit Request (200)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
BeginSeqNum	int64	8	10	Sequence number of first requested message
ReqMessageCount	uint8	1	18	Number of requested messages

5.2. Retransmit Reject

Retransmit Reject (202)	Type	Length	Offset	Description
<Message Header>	<MsgHeader>	10	0	Common message header
RetryDelayNanos	int64	8	10	Minimum time to wait in nanoseconds before sending another retransmit request
Details	char40	40	18	Retransmit reject reason in text
Reason	uint8	1	58	1 - Sequence too low 2 - Sequence too high 3 - Rate limit exceeded 4 - Other error

6. Appendix

6.1. Revision History

Version	Date	Author	Description
0.1	Sep-22-2020	KW/JT	Initial draft
0.2	Oct-04-2020	JT	Updated spec to match SBE xml
0.3	Oct-07-2020	KW	Added initial settlement and limit up/down prices to instrument definition and status messages. Changed retransmit reject details to fixed width fields (now all messages have fixed width).
0.4	Oct-13-2020	KW	Added Trade Summary message.
0.5	Oct-23-2020	KW	Added Trade Session Volume message. Adjusted packet and instrument headers to ensure 8-byte alignment.
0.6	Nov-02-2020	KW	Added snapshotInstrumentId to packet header and removed from snapshot messages.
0.7	Dec-28-2020	FY	Updated Symbol Length to 24
0.8	Feb-3-2021	MG	Added Open Interest message
0.9	Feb-12-2021	KW	Replaced OpenPrice with IndicativeOpenPrice and DayOpenPrice.
1.0	Mar-2-2021	MG	Added TradingStatus to Outright Instrument Definition Added SpreadBuyConvention to Spread Instrument Definition Removed OldQuantity and NewQuantity from Trade Amend Removed Quantity and Price from Trade Bust Added Spread Trade Amend message
1.1	Apr-25-2021	KW	Added priorSettlementPrice to EndOfSnapshot message.
1.2	May-14-2021	KW/JT	Fixed missing fields in Start of Outright Instrument Snapshot and Start of Spread Instrument Snapshot Fixed product code length reduced from 16 to 8 Added -128 (0x80) to Side in InstrumentHeader Added InstrumentDefinitionFlags fields to Outright Instrument Definition, Spread Instrument Definition, and End of Snapshot Added a note around maximum order count Updated Flags & bit set fields to Hex Values Section 2.2: Added additional details around versioning Section 3.5/3.9: Added additional explanation of orderPut & Trades messages Section: 3/3.8: Added details around incremental transaction semantics