Electron

Technical Specifications: FIX 5.0 Interface

Order Entry & Market Data

Version: 20260706.2
Date: 06 July 2026

CONFIDENTIAL TECHNICAL INFORMATION

Table of Contents

Change History Log

Version	Date	Description
0.1	17 February 2025	Initial document.
0.2	14 March 2025	Added schedule.
0.3	27 March 2025	Added data dictionary link.
20251014.2	14 October 2025	Removed QuickFix configs, added *Password (554)* to Logon [A], updated *BusinessRejectRefID (379)* description in BusinessMessageReject [j], added *ClOrdID (11)* uniqueness information, removed *TransactTime (60)* from NewOrderSingle [D], noted that *OrderQty (38)* must be integer on NewOrderSingle [D], removed *ExpireTime (126)* from NewOrderSingle [D], marked *CustomerOrderCapacity (582)* required on NewOrderSingle [D], added Time in Force FOK, added Execution Instructions PARTICIPATE_DONT_INITIATE and BEST_LIMIT, updated Cancel on Disconnect description and added Cancel on Logout, added Trade Busts section, update FIX session schedule, add *Commission (12)* and *CommType (13)* to ExecutionReport [8].
20251113	13 November 2025	Added OrderCancelReplaceRequest[G] to order entry. Added *MinQty(110)* to order entry. Removed ExpireTime from market data and from order entry. Removed EXI *OrderID (37)* from market data. Removed Stop Limit (40=4) from order type. Removed *StopPx(99)* from order entry. Removed *NoPartyIDs (453)*, *PartyID (448)*, *PartyIDSource (447)*, and *PartyRole (452)* from order entry.
20251208	8 December 2025	Restructured a number of messages on both gateways and removed superfluous fields. Language has been changed in various documents to more accurately reflect system behavior, such as the documentation in sections regarding Symbology, Instrument States, Cancel on Disconnect (CoD), Cancel on Logout (CoL), and Tick Sizes/Minimum Quantity Increments among others. A new table has been added for BusinessMessageReject [j]. *AccountType (581)* has been removed from NewOrderSingle [D]. An irrelevant reference to *CorporateAction (292)* has been removed from MarketDataSnapshotFullRefresh [W].
20251210.1	10 December 2025	Adjustment to specification of expected reject message for incorrect order quantity increment. More detailed specification of personal order tracking in market data feed.
20251210.2	10 December 2025	Removed *OrdStatus (39)* = 6 (Pending Cancel) and E (Pending Replace). *EventType (865)* = 7 (Last Eligible Trade Date) added. Removed *SecurityType (167)* from NewOrderSingle [D]. Removed *MDEntryType (269)* = g (Trading Reference Price).

20251212.1	12 December 2025	Added further detail and examples to the symbology section.
20260107.1	7 January 2026	Removed reference to *ManualOrderIndicator (1028)* in OrderCancelReplaceRequest [G]. Respecified definitions of *MDEntryDate (272)* and *MDEntryTime (273)* for MarketDataSnapshotFullRefresh [W] and MarketDataIncrementalRefresh [X] messages. Added *ExecType (150)* = 3 (Done for day) to Table 16: ExecutionReport [8] message. Added *MDQuoteType (1070)* to Table 23: MarketDataIncrementalRefresh [X] message. Added *SecurityIDSource (22)*, *OrderQty (39)*, *OrdType (40)*, *Price (44)*, *SecurityID (48)*, *Symbol (55)*, *TimeInForce (59)*, *CustOrderCapacity (582)*, *ManualOrderIndicator (1028)*, *AggressorIndicator (1057)* and removed *SettlCurrAmt (119)* from Table 20: ExecutionReport [8] message for a busted trade.
20260205.1	5 February 2026	Added FIX Sample Message Sequences throughout, removed duplicative example messages, expanded description of *MDEntryID (278)* in Table 23: MarketDataIncrementalRefresh [X], added enum *ExecRestatementReason (378)* = 8 (Cancel on Disconnect) in Table 16: ExecutionReport [8], added enum *CxlRejReason (102)* = 2 (Re-use of ClOrdID in Session) in Table 19: OrderCancelReject [9].
20260309.1	9 March 2026	Added *ExpireTime (126)* and *TimeInForce (59)* = 6 (Good Till Date) for NewOrderSingle [D], ExecutionReport [8], and OrderCancelReplaceRequest [G] messages.
20260406.1	6 April 2026	Added sub-section describing Mass Quote Protection (MQP), added enum *AccountType (581)* = 2 (NON_CUSTOMER), added *ClOrdLinkID (583)* to NewOrderSingle [D].
20260423.1	23 April 2026	Updated Order Cancellation section to include Mass Cancel Functionality. Added OrderMassCancelRequest [q] and OrderMassCancelReport [r] messages with new fields *MassCancelRequestType (530)*, *MassCancelResponse (531)*, and *MassCancelRejectReason (532)* as well as descriptions and examples of functionality.
20260706.2	06 July 2026	Added section outlining support and documentation for Drop Copy FIX sessions, added *OpenCloseSettlFlag (286)* for MarketDataSnapshotFullRefresh [W] and MarketDataIncrementalRefresh [X]. Updated Req'd for *ResetSeqNumFlag (141)* to 'N' in for Logon [A]. Expanded description of chaining in Order Modification section.

Support Contacts

For general inquiries, help with ElectronX's APIs, or any other support-related matters, please reach out to **support@electronx.com**.

Introduction

The purpose of this document is to outline the trading functionality available on ElectronX via a FIX trading protocol.

ElectronX FIX gateways are based on FIX v5 session layer messaging and expect FIX v5.0 SP2 as the default application layer messaging protocol.

FIX Reference Guide

Please refer to the accompanying ElectronX FIX Reference Guide for details on system architecture, network connectivity, sample QuickFIX configurations, data dictionaries, and sample messages.

Symbology

ElectronX uses a string to identify instruments trading on the platform's API using *Symbol (55)*. Currently, there is no support to identify instruments using any other common identifiers such as CUSIP, ISIN or Bloomberg code.

ElectronX Bounded Futures symbols have the following format:
`Product[Bounded Price Range Upper Band]-ISO-[Hub Code]-[Date]-[Hour Ending]`

ElectronX Binary Options symbols have the following format:
`Product-ISO-[Hub Code]-[Date]-[Hour Ending]`

The calendar day is formatted as YYmmdd and the hour-ending is formatted as HH, i.e. 02 or 15). These hour-endings are in the main timezone for the ISO. For ERCOT for example, the hour-endings are for the America/Texas timezone or “Central” time.

With respect to daylight savings (DST), when clocks spring forward, for the skipped 2-3am that denotes the start of DST in March, no contract will be created. Contracts will be listed from hour-ending 2 (02) to hour-ending 4 (04), with no hour-ending 3 (03). When clocks fall back, for the repeated 2am in November, the contract hour code will be “02” for the first hour-ending at 2 and “X2” for the repeated hour-ending at 2.

A list of instruments on the platform can be retrieved using the SecurityListRequest [x] message.

Example 1: Symbol for ERCOT North Hub 0-500 Price Bounded Future for April 2, 2025 at 9am CT
`BFUT500-ERCOT-HB_NORTH-250402-09`

Example 2: Symbol for ERCOT North Hub Binary Option for April 2, 2025 at 9am CT
`BOPT-ERCOT-HB_NORTH-250402-09`

Instrument States

An instrument may be in one of the following states:

Table 1: Instrument States

State	Definition
PENDING	Initial state for a newly created instrument which has not yet begun trading.
OPEN	In this state, the instrument is open for continuous order entry and matching.
CLOSED	In this state, orders can not be entered, modified, or canceled, and no matching occurs. Any existing Day orders will be expired.
SUSPENDED	Orders can be canceled but no matching occurs, and no order entry or modification is allowed.
HALTED	This state is similar to SUSPENDED, with the exception that orders cannot be canceled.
EXPIRED	An instrument moves to this state when its Expiration Date/Time is reached. In this state, any remaining orders are cancelled and no new orders can be entered.
TERMINATED	When an Instrument’s Termination Date is reached, the order book is removed from the matching engine and positions are closed.

FIX Notation

Please note the following presentation notes which apply to message definitions and FIX examples throughout this document:

- FIX tag/value pairs are delimited within a TCP connection using the SOH (ascii character 1) character. Since this is a non-printable character, in this document we use the | character instead, and pad each pair with spaces to make them easier to use.
- Components are blocks of FIX tags which appear frequently in the specification (e.g. header and footers which appear on every FIX message). They are defined centrally for convenience and then referenced throughout the document using <> notation.
- Repeating groups of FIX tags appear in various messages. The depth of a repeating group is indicated using the → marker in FIX message definitions.
- References to individual FIX fields (or “tags”) are presented in italic font, with the tag number following the tag name. For example, *HeartBtInt (108)*.
- References to individual FIX messages are presented in standard font, with the message type following the name in square brackets. For example, Logon [A].

FIX Component Definitions

Standard Header & Trailer

Each FIX message sent to and received from the ElectronX platform must start and end with a message header and trailer components.

Table 2: Standard Header Component

Tag	Field Name	Req'd	Data Type	Comments
8	BeginString	Y	String	FIXT.1.1
9	BodyLength	Y	Length	Standard FIX message body length.
35	MsgType	Y	String	The message type.
34	MsgSeqNum	Y	SeqNum	FIX Message sequence number.
43	PossDupFlag	N	Boolean	Always required for retransmitted messages as the result of a resend request.
49	SenderCompID	Y	String	Sender identity as agreed with the exchange operator.
50	SenderSubID	C	String	Sender sub-identifier representing an individual user (as previously agreed with the exchange operator). **Required on all application messages related to the entry or management of orders.** Example value: "firms/abc/users/def".
52	SendingTime	Y	UTCTimestamp	SendingTime in UTC[ns].
56	TargetCompID	Y	String	Intended target identity as agreed with the exchange operator.
57	TargetSubID	N	String	Target sub-identifier representing an individual user (as previously agreed with the exchange operator).

Table 3: Standard Trailer Component

Tag	Field Name	Req'd	Data Type	Comments
10	Checksum	Y	String	Standard FIX checksum.

FIX Session Level Management

Establishing and Maintaining a FIX Session

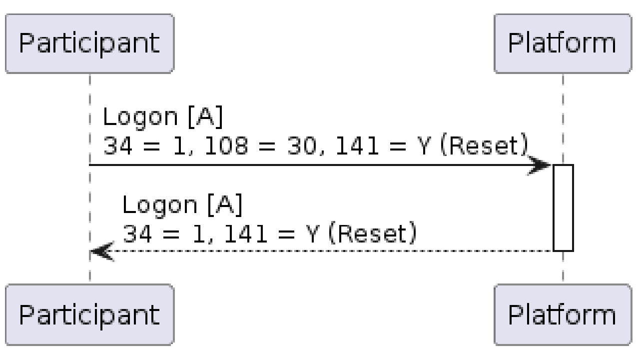
The Logon [A] message initiates a connection to the ElectronX platform. FIX sessions are typically initiated by customers (as opposed to the exchange platform), and can be made any time.

Important: Logon [A] messages must NOT specify a *SenderSubID (50)* in the header. Sending this tag will cause the logon attempt to be rejected.

Table 4: Logon [A] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = A
98	EncryptMethod	Y	Int	Encrypted messages are not supported: 0 = None
108	HeartBtInt	Y	Int	A 30-second interval is recommended.
141	ResetSeqNumFlag	N	Boolean	Indicates both sides of a FIX session should reset sequence numbers back to 1.
554	Password	N	String	If the FIX session is configured with a password, this field is used to supply the password.
1137	DefaultApplVerID	Y	String	Specifies the FIX 5.0 service pack release being applied: 9 = FIX50SP2
< Standard Trailer >		Y		

Figure 1: Successful Logon



Sample Messages Sequence 1: Successful Logon Sequence
Establishes a FIX session between client and exchange.

Client sends Logon [A].

```
< 8=FIXT.1.1 | 9=81 | 35=A | 49=SENDER_ID | 56=EXI | 34=1 | 52=20260122
-20:21:17 | 98=0 | 108=30 | 1137=9 | 141=Y | 10=049 |
```

EXI responds with Logon [A].

```
> 8=FIXT.1.1 | 9=91 | 35=A | 34=1 | 49=EXI | 52=20260122-20:21:17.390650800 |
56=TARGET_ID| 98=0 | 108=30 | 141=Y | 1137=9 | 10=047 |
```

Under normal circumstances, both FIX engines on either side of the connection will regularly exchange Heartbeat messages. The frequency of such a message exchange is determined by the *HeartBtInt (108)* value indicated in the Logon [A] message.

Regular application messages qualify as a heartbeat, i.e. both sides of the connection are behaving nominally if a message is received during the *HeartBtInt (108)*. If no application messages are received during the interval, the Heartbeat [0] message is used as described below.

Table 5: Heartbeat [0] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 0 (zero)
112	TestReqID	N	String	Required when the heartbeat is the result of a TestRequest [1] message.
< Standard Trailer >		Y		

Sample Message Sequence 2: Heartbeat Exchange
Keep-alive message to maintain session connectivity.

Client sends Heartbeat [0].

```
< 8=FIXT.1.1 | 9=56 | 35=0 | 49=SENDER_ID | 56=EXI | 34=2 | 52=20260122
-20:21:19 | 10=156 |
```

EXI responds with its own Heartbeat [0].

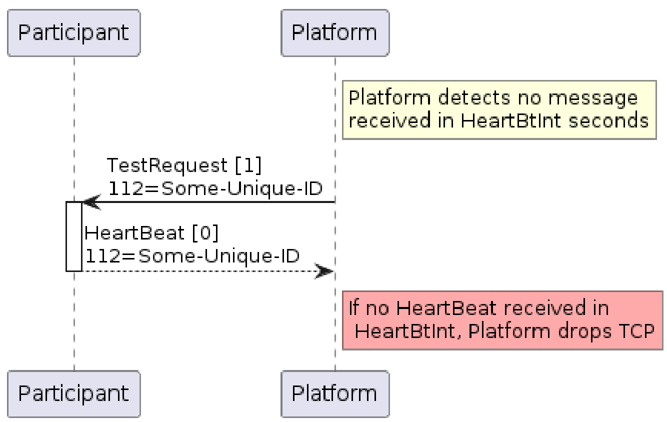
```
> 8=FIXT.1.1 | 9=66 | 35=0 | 34=2 | 49=EXI | 52=20260122-20:21:47.474656756 |
56=TARGET_ID | 10=174 |
```

At any point either counterparty can force the opposing FIX engine to send a Heartbeat [0] message by submitting a TestRequest [1] message containing a *TestReqID (112)* value, which should be echoed in the Heartbeat [0] response.

Table 6: TestRequest [1] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 1
112	TestReqID	Y	String	ID to be returned in the resulting Heartbeat [0] response.
< Standard Trailer >		Y		

Figure 2: Using TestRequest [1] to Request Heartbeat [0]



Sample Message Sequence 3: TestRequest[1]
Sent by either party when expected heartbeat is not received. Requires immediate Heartbeat response with matching TestReqID.

EXI sends a TestRequest [1] with *TestReqID* (*112)* = TEST to the client because it noticed that the client missed sending a heartbeat message.

```
> 8=FIXT.1.1 | 9=75 | 35=1 | 34=7 | 49=EXI | 52=20260122-20:23:55.503021182 |
56=TARGET_ID | 112=TEST | 10=171 |
```

Client responds with a Heartbeat [0] and *TestReqID (112)* = TEST.

```
< 8=FIXT.1.1 | 9=65 | 35=0 | 49=SENDER_ID | 56=EXI | 34=8 | 52=20260122
-20:24:05 | 112=TEST | 10=178 |
```

Cancel on Disconnect

Cancel on Disconnect (CoD) is an optional risk control feature for participants in the event that application connectivity is unexpectedly lost. If a network disconnection occurs where the TCP connection is lost, CoD is instantly triggered and the ElectronX platform will automatically cancel all open (unexecuted) DAY orders for the participant, while GTC and GTD orders will continue to rest. After reconnecting, missed messages may be replayed, including execution reports for any canceled orders. CoD will not be triggered by a graceful logout.

Cancel on Disconnect is enabled by default, and can be set on a session-by-session basis. Contact ElectronX Support to adjust this functionality.

Cancel on Logout

Cancel on Logout (CoL) extends the CoD cancellation functionality to trigger on graceful logout (both client- and exchange-initiated). If the exchange forces a client to logout or the client sends a logout message, open DAY orders will be cancelled if CoL is enabled. Functionally CoL behavior is a superset of CoD, so if CoL is enabled there would be no effect of enabling CoD. If CoL isn't desired itself, then enabling CoD independently is still possible.

Crucially CoD and CoL are optional features that can both be disabled if you wish to keep day orders alive for any logout/disconnect event.

Sequence Number Tracking, Recovery, and Reset

The FIX protocol uses simple, incrementing *MsgSeqNum (34)* values (carried in the StandardHeader of each message) to both detect and request retransmission of missed messages.

Each FIX engine will maintain a simple message count of outbound and inbound messages, values which should increment by one for each message sent or received per FIX session. Should a message be received that does not match the **expected** sequence number given, then it is possible that one or more messages were lost.

In that situation, a ResendRequest [2] message may be sent to request retransmission of a specified range of messages identified by their *MsgSeqNum (34)* value.

Table 7: ResendRequest [2] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 2
7	BeginSeqNo	Y	SeqNum	*MsgSeqNum (34)* of first message in the range to be re-sent (inclusive).
16	EndSeqNo	Y	SeqNum	*MsgSeqNum (34)* of the last message in the range to be resent (inclusive), or "0" to request resend all messages after the indicated *BeginSeqNo (7)*.
< Standard Trailer >		Y		

The expected response to a ResendRequest [2] message is a stream of application messages with *PossDupFlag (43)* set to "Y" (yes) in the header to indicate that this is a potentially duplicative message.

Sample Message Sequence 4: ResendRequest [2]
Requests retransmission of messages due to sequence gap. Used for message recovery.

Client sends ResendRequest [2] for messages from sequence number 5 to current sequence number.
`< 8=FIXT.1.1 | 9=66 | `**`35=2`**` | 49=SENDER_ID | 56=EXI | 34=13 | 52=20260122-20:26:53 | `**`7=5`**` | `**`16=0`**` | 10=083 |`

EXI responds with a SequenceReset [2] with *GapFillFlag (123)* = Y to indicate that the messages in the request range were only administrative messages.
`> 8=FIXT.1.1 | 9=115 | 35=4 | 34=5 | 43=Y | 49=EXI | 52=20260122-20:26:54.026039220 | 56=TARGET_ID | 122=20260122-20:26:54.026039220 | 36=12 | 123=Y | 10=042 |`

It may be appropriate in some situations for the opposite FIX engine to refuse to replay the message. This is typically the case for administrative messages which either have limited value (e.g., Heartbeat messages) or might cause confusion (e.g., Logon messages). In such circumstances, the opposite FIX engine may respond with a SequenceReset [4] message (below) with *GapFillFlag (123)* set to “Y”.

The SequenceReset [4] message can be used in two distinct situations:

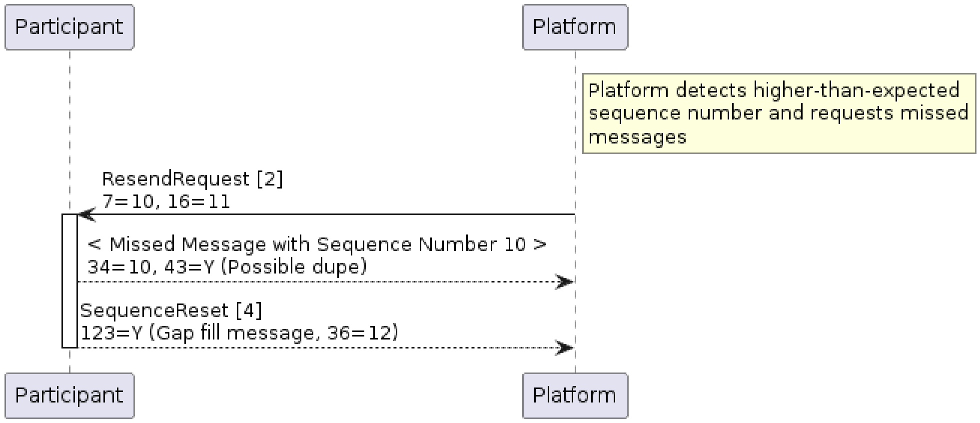
1. To request that the counterparty adjusts their internal sequence number to the indicated *NewSeqNo (36)* value. In this case, *GapFillFlag (123)* is either not present or set to “N” (no).

2. In the message replay situation described above, to replace a FIX message which will not be re-sent. In this case, *GapFillFlag (123)* should be set to “Y” and the *NewSeqNo (36)* field will contain the next sequence number to be sent.

Table 8: SequenceReset [4] Message

Tag	Field Name	Req’d	Data Type	Comments
< Standard Header >		Y		35 = 4
123	GapFillFlag	N	Boolean	Indicates that the Sequence Reset message is replacing administrative or application messages which will not be resent.
36	NewSeqNo	Y	SeqNum	New sequence number.
< Standard Trailer >		Y		

Figure 3: ResendRequest [2] Message Triggers Re-Transmission



Terminating a FIX Session

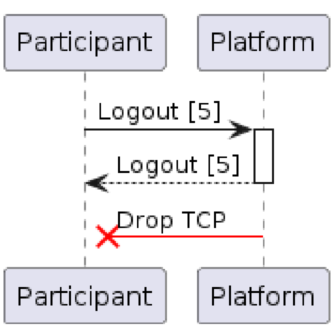
The Logout [5] message initiates or confirms the termination of a FIX session. Customers may log out of their FIX sessions at any time.

Table 9: Logout [5] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 5
58	Text	N	String	Free format text string.
< Standard Trailer >		Y		

The expected response to a successful Logout [5] message is a reciprocal Logout [5] message, **after which** the TCP connection should be dropped.

Figure 4: Graceful Logout Sequence



Participants are recommended to schedule a graceful logout before the start of a scheduled maintenance window. If this is not received, then the ElectronX Platform will initiate the logout process.

Sample Message Sequence 5: Graceful Session Termination
Gracefully terminates the FIX session. Can be initiated by either party.

Client attempts graceful session termination with Logout [5].

```
< 8=FIXT.1.1 | 9=57 | 35=5 | 49=SENDER_ID | 56=EXI | 34=16 | 52=20260122
-20:28:00 | 10=212 |
```

EXI responds with Logout [5].

```
> 8=FIXT.1.1 | 9=67 | 35=5 | 34=14 | 49=EXI | 52=20260122-20:28:00.984837309 |
56=TARGET_ID | 10=228 |
```

General Error Handling

Most transactional messages in FIX are rejected using an application-level message (as described in various sections which follow).

In the event that one of these messages can not be used to reject a message, however, then the ElectronX Platform may return either a Reject [3] message, or a BusinessMessageReject [j] message as described below.

- Reject [3] messages are commonly used to reject messages where the message type is known, but there is something syntactically wrong with its content (e.g. incorrect repeating group).
- BusinessMessageReject [j] is a more generic message used to reject errors such as unsupported message type, malformed messages, or message rate violations.

Table 10: Reject [3] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 3
45	RefSeqNum	Y	SeqNum	The message sequence number being rejected.
58	Text	N	String	Optional string to further describe the error.
371	RefTagID	N	Int	The tag (field) number being rejected.
372	RefMsgType	Y	String	The MsgType being rejected.
373	SessionRejectReason	N	Int	Reason for the rejection: 0 = Invalid tag number 1 = Required tag missing 2 = Tag not defined for this message type 3 = Undefined tag 4 = Tag specified without value 5 = Value is incorrect 6 = Incorrect data format for value

Tag	Field Name	Req'd	Data Type	Comments
				9 = CompID problem 10 = SendingTime accuracy problem 11 = Invalid MsgType 13 = Tag appears more than once 14 = Tag specified out of required order 15 = Repeating group fields out of order 16 = Incorrect NumInGroup count 99 = Other
< Standard Trailer >		Y		

Table 11: BusinessMessageReject [j] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = j
45	RefSeqNum	N	SeqNum	The message sequence number being rejected.
58	Text	N	String	String to further describe the error.
372	RefMsgType	Y	String	The MsgType being rejected.
379	BusinessRejectRefID	N	String	When a FIX gateway rejects a message with a BusinessMessageReject, it provides tag 379 (`BusinessRejectRefID`) on the BusinessMessageReject and populates it with the ClOrdID or MDReqID to allow FIX clients to quickly determine which message was rejected.
380	BusinessRejectReason	Y	Int	The reason for the rejection: 0 = Other 1 = Unknown ID 2 = Unknown Security 3 = Unsupported Message Type 4 = Application Not Available (downtime) 5 = Conditionally required field missing 6 = Not authorized 18 = Invalid price increment (tick size)
< Standard Trailer >		Y		

Sample Message Sequence 6: Business Reject

A correctly formatted message is sent to the exchange, but its content violates business rules and is rejected.

Client mistakenly tries to enter an order via the market data gateway.

```
< 8=FIXT.1.1 | 9=185 | 35=D | 49=SENDER_ID | 56=EXI | 34=3 | 52=20260122
-20:34:57 | 11=Order125 | 21=1 | 1=ACCOUNT | 55=BFUT500-ERCOT-HB_NORTH
```

```
-260122-21 | 54=1 | 60=20260122-20:34:57 | 40=2 | 59=1 | 44=30 | 38=2 | 1028=Y
| 582=2 | 10=248 |
```

EXI sends a BusinessMessageReject [j] listing the reason in *Text (58)*.

```
> 8=FIXT.1.1 | 9=106 | 35=j | 34=3 | 49=EXI | 52=20260122-20:34:57.831289262 |
56=TARGET_ID | 45=3 | 58=Unsupported Message Type | 372=D | 380=3 | 10=013 |
```

Sample Message Sequence 7: Formatting Reject
Indicates a session-level violation. An incorrectly formatted message to the exchange will result in a generic Reject [3] message.

Client sends an order with an invalid order quantity ABC.

```
< 8=FIXT.1.1 | 9=226 | 35=D | 49=SENDER_ID | 56=EXI | 34=47 | 52=20260126
-14:37:38 | 50=SENDER_SUB_ID | 11=Order133 | 21=1 | 1=ACCOUNT | 55=BFUT500-
ERCOT-HB_NORTH-260126-21 | 54=1 | 60=20260126-14:37:38 | 40=2 | 59=1 | 44=5 |
38=ABC | 1028=Y | 582=2 | 18=T | 10=039 |
```

EXI responds with a session level message with the error reason Incorrect data format for value.

```
> 8=FIXT.1.1 | 9=156 | 35=3 | 34=43 | 49=EXI | 52=20260126-14:37:42.122725177 |
56=TARGET_ID | 57=TARGET_SUB_ID | 45=47 | 58=Incorrect data format for value |
371=38 | 372=D | 373=6 | 10=121 |
```

Order Entry and Matching

Overview

ElectronX operates as a continuous central limit order book to match orders within the system according to price-time priority:

- Higher-priced orders to buy have priority over lower-priced bids,
- Lower-priced offers to sell buy have priority over higher-priced sells,
- Within a price level, older orders have priority over newer orders,
- Should an existing order increase its quantity (at the same price), it is assigned a new timestamp and therefore loses time priority, but quantity decreases retain time priority.

ClOrdID (Client Order ID)

ClOrdID is a unique, participant-created identifier for this Order. Uniqueness must be guaranteed across a session (i.e. between logon and logout), which may span multiple days.

EXI caches ClOrdIDs of active orders. As soon as a GTC order becomes inactive, its ClOrdID can be re-used.

Tick Sizes / Minimum Quantity Increments

Each instrument on the ElectronX Platform has a minimum price increment ("tick size") and minimum quantity increment ("lot size"). These instrument-level settings are indicated in the SecurityList [y] message in the *MinPriceIncrement (969)* and *MinTradeVol (562)* fields respectively.

Orders with a price which doesn't comply with the tick size will be rejected using ExecutionReport [8] with *ExecType (150)* = 8 (Rejected) with *Text (58)* describing the issue. Orders with a quantity which does comply with the lot size will be rejected using Reject [3] with *Text (58)* describing the issue.

Order Types, Time In Force & Execution Instructions

The intended behavior of orders entering the ElectronX Platform is controlled using a combination of *OrdType (40)*, *TimeInForce (59)* and *ExecInst (18)* fields. These are explained in turn below.

All orders are LIMIT orders. See below for the equivalent to a market order.

Table 12: *OrdType (40)* Definitions

Order Type	Value	Definition
LIMIT	2	A priced order that can only trade at a price equal to or better than the price specified.

Table 13: *TimeInForce (59)* Conditions

Time In Force	Value	Definition
DAY	0	[Default] The order remains open for the current trading day only.
GOOD TILL CANCEL (GTC)	1	The order remains available for execution until fully executed or canceled.
IMMEDIATE OR CANCEL (IOC)	3	The order is immediately executed as far as possible upon entry, with all unfilled quantity expired. Can be used in conjunction with *MinQty (110)* to specify a minimum quantity which must be executed immediately.
FILL OR KILL (FOK)	4	If the entire order quantity can not be satisfied immediately, then the order is canceled in full.
GOOD TILL DATE (GTD)	6	The order is active until a specific date and time (expressed in UTC) as indicated in *ExpireTime (126)*.

In addition to order-level Time In force conditions, participants can also specify a minimum order quantity using *MinQty (110)*, which requires that an order receives at least a minimum execution quantity upon entry (possibly in multiple fills). When used in combination with Time In Force, this field offers fine-grained control over the conditions under which an order can participate. For example:

- An order marked as IOC with *MinQty (110)* of 200 must receive an immediate, minimum trade quantity of 200. If this is not possible, then the entire order immediately expires.
- In the case of orders marked with a FOK *TimeInForce (59)*, *MinQty (110)* can be included but has no material effect, as the FOK requires a full fill upon entry.
- A Good Till Cancel order with a *MinQty (110)* specified must trade at least the specified quantity upon first entry into the order book, otherwise the order expires.

Note that *MinQty (110)* only limits order behavior as it enters the order book; it does not prevent smaller trades occurring against the order once it is resting on the order book.

Table 14: *ExecInst (18)* Definitions

Exec Instruction	Value	Definition
PARTICIPATE DON'T INITIATE	6	Require that the order is only accepted if it would NOT immediately match. This guarantees that the order will always be the passive side in any trades. An order that would result in a match would be rejected with reason "Order may participate but not initiate in the market". Should not be used in combination conditions which require immediate executions - *TimeInForce (59)* = 3 or 4 (IOC or FOK).
BEST_LIMIT	R	A flag that if set indicates that the price of a limit order shall be set to the price at the top of the book on the same side as this order.
IMMEDIATELY_EXECUTABLE_LIMIT	T	A flag that if set indicates that the price of a limit order shall be set to the price at the top of the book on the opposing side as this order, thus able to immediately match.

Order Entry

Participants may place orders into the ElectronX platform order book to buy or sell securities using a NewOrderSingle [D] message.

Table 15: NewOrderSingle [D] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = D
1	Account	Y	String	Account reference as previously advised to the exchange operator.
11	ClOrdID	Y	String	Unique, participant-created identifier for this Order. Uniqueness must be guaranteed across a session (i.e. between logon and logout), which may span multiple days.

Tag	Field Name	Req'd	Data Type	Comments
18	ExecInst	N	MultipleCharValue	Execution instruction. See Table 14: *ExecInst (18)* definitions.
38	OrderQty	Y	Qty	Order quantity. Must be a positive integer.
40	OrdType	Y	Char	Order type. See Table 12: *OrdType (40)* definitions.
44	Price	N	Price	Price per share/unit.
54	Side	Y	Char	1 = Buy 2 = Sell
55	Symbol	Y	String	Instrument symbol.
59	TimeInForce	Y	Char	See Table 13: *TimeInForce (59)* definitions.
110	MinQty	N	Qty	Minimum execution quantity upon entry for new order or order immediately expires.
126	ExpireTime	C	UTCTimestamp	Order expiry date and time for orders where TimeInForce = Good Till Date.
582	CustOrderCapacity	Y	Int	The values below correspond 1-to-1 with standard CTI codes: 1=OWN_ACCOUNT 2=PROPRIETARY_ACCOUNT 3=FINANCIAL_ADVISOR 4=ALL_OTHER
583	ClOrdLinkID	N	String	Permits order originators to tie together groups of orders.
1028	ManualOrderIndicator	Y	Boolean	Indicates if the order was initially received manually (as opposed to algorithmically).
< Standard Trailer >		Y		

Example 3: Entry of New Limit Buy Order for 30 lots at $40.00

```
8=FIXT.1.1 | 9=246 | 35=D | 34=2 | 49=SENDER_ID | 50=SENDER_SUB_ID |
52=20251124-19:56:06.000 | 56=TARGET_ID | 1=ACCOUNT | 11=ORDER_20251124_195606
| 38=30 | 40=2 | 44=40 | 54=1 | 55=BFUT100-ERCOT-HB_TEST-251124-23 |
60=20251124-19:56:06 | 132=21 | 167=CMDTY | 460=5 | 582=2 | 1028=N | 10=166
```

Order Acknowledgement & Execution

The ExecutionReport [8] message is used to acknowledge various order lifecycle events including the acceptance, rejection, and expiry of orders, as well as providing details of matches (fills) against orders.

All orders entering the Platform are first either accepted or rejected using an ExecutionReport [8]. The order may then receive further ExecutionReport [8] messages as necessary given their marketability and *TimeInForce (59)* constraints. For example:

1. A badly-formatted order which fails validation upon entry will receive an ExecutionReport [8] with *OrdStatus (39)* = 8 (Rejected).

2. A good for day order which is not immediately executable will receive an ExecutionReport [8] with *OrdStatus (39)* = 0 (New). Should it receive executions during the course of the day, then these will be notified in subsequent ExecutionReport [8] messages with *OrdStatus (39)* = 1 (Partially Filled) and/or 2 (Fully Filled).

3. An order with TimeInForce (59) = 6 (good till date) will receive an ExecutionReport [8] with OrdStatus (39) = 0 (New) upon entry. If it remains unexecuted, it will expire at the indicated ExpireTime (126) with an Execution Report [8] message with OrdStatus (39) = C (Expired).

4. A Fill Or Kill order with *TimeInForce (59)* = 4 which can not be fully executed will first receive an ExecutionReport [8] with *OrdStatus (39)* = 0 (New) and then an immediate Execution Report [8] message with *OrdStatus (39)* = C (Expired).

5. An order with *TimeInForce (59)* = 3 (immediate or cancel) which can be partially filled upon entry will receive at least three ExecutionReport [8] messages; the first will indicate *OrdStatus (39)* = 0 (New), the next messages will detail the fill(s), and the final ExecutionReport [8] message will expire the remainder with *OrdStatus (39)* = C (Expired).

Table 16: ExecutionReport [8] Message

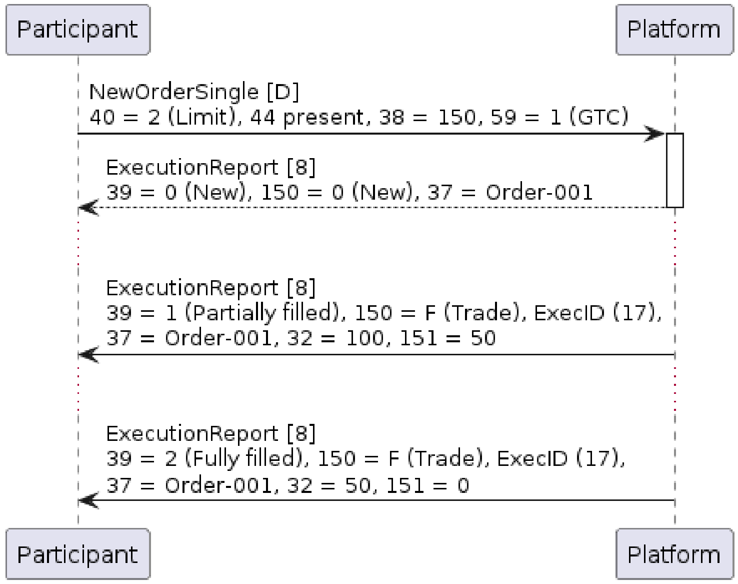
Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 8
1	Account	Y	String	Account reference if indicated on the original order.
6	AvgPx	Y	Price	Volume-weighted average price of all trades against this order. May be zero for unexecuted orders.
11	ClOrdID	Y	String	The participant-assigned ClOrdID value as-sent on the last order action message from the Participant (new order, amendment or cancel).
12	Commission	N	Amt	Sent when *ExecType (150)* is 'F' (Trade). Total amount of execution and clearing fees charged for a trade.

Tag	Field Name	Req'd	Data Type	Comments
13	CommType	N	Amt	Sent when *ExecType (150)* is 'F' (Trade): 3=Absolute
14	CumQty	Y	Qty	Cumulative executed quantity of this order so far. May be zero for unexecuted orders.
17	ExecID	Y	String	Unique identifier for the Execution Report as assigned by ElectronX. Typically a 13-character alphanumeric string.
18	ExecInst	N	MultipleChar	Must match the original order (cannot be amended). Instructions for order handling. Note that Price Validity Checks can only be ignored (c) for market-to-limit orders to sell. See Table 14: *ExecInst (18)* definitions.
22	SecurityIDSource	Y	String	Identifies source of *SecurityID (48)* value: 8 = Exchange symbol
31	LastPx	Y	Price	Price of this last fill. Will be zero for messages not relating to a trade.
32	LastQty	Y	Qty	Quantity traded on this last fill. Will be zero for messages not relating to a trade.
37	OrderID	Y	String	Unique identifier for Order as assigned by ElectronX. Typically a 13-character alphanumeric string.
38	OrderQty	Y	Qty	Total order quantity (amended as necessary). Must be a positive integer.
39	OrdStatus	Y	Char	The latest status of the order after any changes have been applied: 0 = New 1 = Partially filled 2 = Fully filled 3 = Done for day 4 = Canceled 8 = Rejected A = Pending new C = Expired
40	OrdType	Y	Char	Order type. See Table 12: *OrdType (40)* definitions.
41	OrigClOrdID	N	String	Sent in the case of order amendment or cancellation. References the prior

Tag	Field Name	Req'd	Data Type	Comments
				ClOrdID (11) value that the action amended/canceled.
44	Price	Y	Price	Order limit price (amended as necessary).
48	SecurityID	Y	String	Security identifier; will always match *Symbol (55).*
54	Side	Y	Char	1 = Buy 2 = Sell
55	Symbol	Y	String	Instrument symbol.
58	Text	N	String	Text string further describing report.
59	TimeInForce	Y	Char	Echoed from New Order Single.
60	TransactTime	Y	UTCTimestamp	Timestamp when the business transaction represented by the message occurred.
103	OrdRejReason	N	Int	Rejection reason (where *OrdStatus (39)* = Rejected): 0 = Broker/Exchange Option 1 = Unknown symbol 2 = Exchange closed (maintenance) 3 = Order exceeds limit (price validation) 5 = Unknown order 6 = Duplicate order *(ClOrdID)* 11 = Unsupported order characteristic 12 = Surveillance option 13 = Incorrect quantity (lot size) 15 = Unknown account (tag 1) 16 = Price exceeds current price band 18 = Invalid price increment (tick size) 99 = Other
119	SettlCurrAmt	N	Amt	Present on trades only. Total amount of this last fill. Equal to *LastPx (31)* x *LastQty (32)*.
126	ExpireTime	C	UTCTimestamp	Order expiry date and time for orders where TimeInForce = Good Till Date.
150	ExecType	Y	Char	The reason that the ElectronX Platform sent this Execution Report: 0 = New 3 = Done for day 4 = Canceled 5 = Replaced

Tag	Field Name	Req'd	Data Type	Comments
				8 = Rejected C = Expired F = Trade
151	LeavesQty	Y	Qty	Remaining, unexecuted quantity left on the order. May be zero for fully filled orders.
378	ExecRestatementReason	N	Int	Indicates reason for unsolicited cancel: 8 = Exchange option for CoD 99 = Self-match prevention
381	GrossTradeAmt	N	Amt	Present on orders which have been filled. Total amount traded across all fills for this order. Equal to *AvgPx (6)* x *CumQty (14)*.
581	AccountType	Y	Int	Type of Account <1> associated with an order: 1 = CUSTOMER 2 = NON_CUSTOMER
582	CustOrderCapacity	N	Int	The values below correspond 1-to-1 with standard CTI codes: 1=OWN_ACCOUNT 2=PROPRIETARY_ACCOUNT 3=FINANCIAL_ADVISOR 4=ALL_OTHER
828	TrdType	N	int	Present on trades only. 0 = Regular trade
880	TrdMatchID	C	String	Always populated for trades. Note that buyer and seller will receive the same value. Will match the *TradeID (1003)* value on market data updates. Typically a 13-character alphanumeric string.
1028	ManualOrderIndicator	Y	Boolean	Indicates if the order was initially received manually (as opposed to algorithmically).
1057	AggressorIndicator	C	Boolean	Always populated for trades. Identifies whether this order was the aggressor in the trade.
< Standard Trailer >		Y		

Figure 5: Limit Order for 150 Lots is Ack'd, Partially Filled for 100, and then Fully Filled



Example 4: Acknowledgement of New Limit Buy Order for 30 Lots at $40.00

```
8=FIXT.1.1 | 9=377 | 35=8 | 34=2 | 49=SENDER_ID | 52=20251124
-19:56:06.697175230 | 56=TARGET_ID | 57=TARGET_SUB_ID | 1=ACCOUNT | 6=0.00 |
11=ORDER_20251124_195606 | 14=0 | 17=6JWM7RJ0012P | 22=8 | 31=0.00 | 32=0 |
37=6DVA2NTH82BQ | 38=30 | 39=0 | 40=2 | 44=40.00 | 48=BFUT100-ERCOT-HB_TEST
-251124-23 | 54=1 | 55=BFUT100-ERCOT-HB_TEST-251124-23 | 59=0 | 60=20251124
-19:56:06.691945600 | 99=0.00 | 150=0 | 151=30 | 581=1 | 582=2 | 1028=N |
10=109 |
```

Example 5: Execution of 20 Lots at $40.00

```
8=FIXT.1.1 | 9=338 | 35=8 | 34=33 | 49=TARGET_ID | 52=20240517
-19:06:47.985808615 | 56=SENDER_ID | 57=SENDER_SUB_ID | 1=ACCOUNT | 6=40.00 |
11=1182560826 | 14=20 | 17=1HPT7DPFMC5M5 | 22=8 | 31=40.00 | 32=30 |
37=1HQ4A5T0EDM07 | 38=20 | 39=2 | 40=2 | 44=40.00 | 48=BFUT5000-ERCOT-HB_NORTH
-251107-13| 54=2 | 55=BFUT5000-ERCOT-HB_NORTH-251107-13| 59=0 | 60=20240517
-19:06:47.977567695 | 119=800.00 | 150=F | 151=10 | 381=800.00 | 582=1 |
880=1HPT7DPFMC5M4 | 1057=Y | 10=116 |
```

Example 6: Execution of Remaining 10 Lots at $40.00

```
8=FIXT.1.1 | 9=338 | 35=8 | 34=33 | 49=TARGET_ID | 52=20240517
-19:06:51.48521379 | 56=SENDER_ID | 57=SENDER_SUB_ID | 1=ACCOUNT | 6=40.00 |
11=1182560826 | 14=10 | 17=1HPT7DPFMC5M5 | 22=8 | 31=40.00 | 32=30 |
37=1HQ4A5T0EDM07 | 38=30 | 39=2 | 40=2 | 44=40.00 | 48=BFUT5000-ERCOT-HB_NORTH
-251107-13| 54=2 | 55=BFUT5000-ERCOT-HB_NORTH-251107-13| 59=0 | 60=20240517
-19:06:51.54392187 | 119=400.00 | 150=F | 151=0 | 381=1200.00 | 582=1 |
880=1HPT7DPFMC5M4 | 1057=Y | 10=116 |
```

Order Modification

The EXI Platform supports the amendments of orders via the OrderCancelReplaceRequest [G] message.

In the use of this message, the order to be amended is identified by using the last-entered *ClOrdID (11)* value relating to the relevant order as the *OrigClOrdID (41)* field in the request. Following FIX convention, the modification response message will contain a new *ClOrdID (11)* value which can then be used by the Participant for further modification of the relevant order. Note that the EXI Platform does not require the entry of the (platform-generated) *OrderID (37)*.

Table 17: OrderCancelReplaceRequest [G] Message

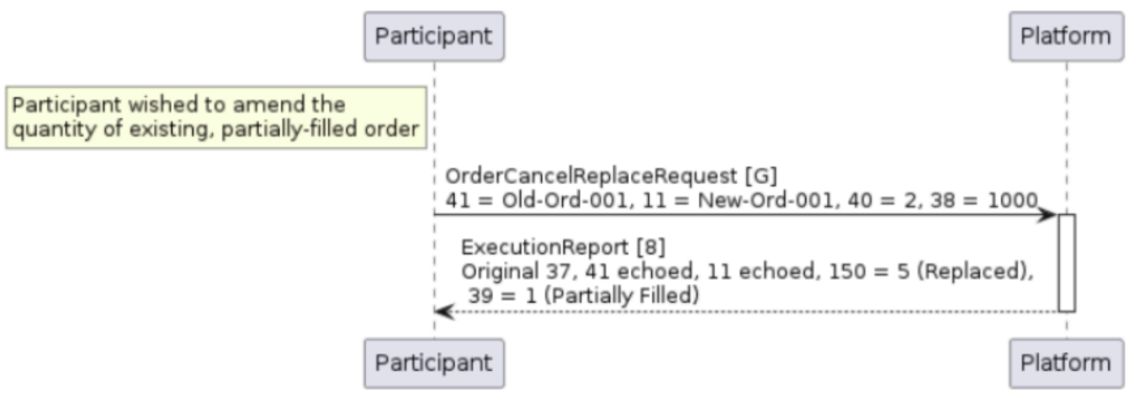
Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = G
11	ClOrdID	Y	String	Fresh, participant-generated, unique reference for this message. Must be different from the original *ClOrdID (11)* for the order.
41	OrigClOrdID	Y	String	The last (participant-generated) *ClOrdID (11)* reference for the order. Either from the initial order add (for a first amendment) or the most recent order amendment.
1	Account	Y	String	Account reference as declared to the exchange operator.
18	ExecInst	N	MultipleChar	Must match the original order (cannot be amended). Instructions for order handling. Note that Price Validity Checks can only be ignored (c) for market-to-limit orders to sell. See Table 14: *ExecInst (18)* definitions
55	Symbol	Y	String	Must match the original order (no amends). Instrument symbol.
38	OrderQty	C	Qty	Updated total order quantity. Only required if amending the quantity. Must be a positive integer. Note that if the order has already been partially filled, then this still refers to the overall order quantity (i.e. not the remaining quantity required). Cannot be amended below *CumQty* (14).
44	Price	N	Price	Updated order price.
59	TimeInForce	N	Char	See Table 13: *TimeInForce (59)* definitions.
126	ExpireTime	C	UTCTimestamp	Order expiry date and time for orders where TimeInForce = Good Till Date.
< Standard Trailer >		Y		

If the order has been successfully modified, the EXI Platform will respond with an ExecutionReport [8] message which echoes many of the key (updated) order attributes, and with *ExecType (150)* = 5 (replaced).

After N successful replaces, the (N+1)-th replace should supply *OrigClOrdID (41)* = the *ClOrdID (11)* value sent on the N-th replace request, not the original NewOrderSingle ClOrdID nor any earlier replacements ClOrdID. EXI tracks a single "current" ClOrdID per live order, which advances with each accepted replacement, but does cache each orders' previous ClOrdID, to help resolve updates using IDs that are one update/generation stale. Sending any prior ClOrdID beyond the last two preceding IDs will yield *CxlRejReason (102)* = 1 (Unknown order).

Note that a modification which **increases** the remaining order quantity will **lose** its time priority at a price level. Any modifications to **reduce** order quantity (or other non-price modifications which do not change quantity) will not cause the order to lose their order book priority.

Figure 6: Successful Amend of Existing Order's *OrderQty (38)*

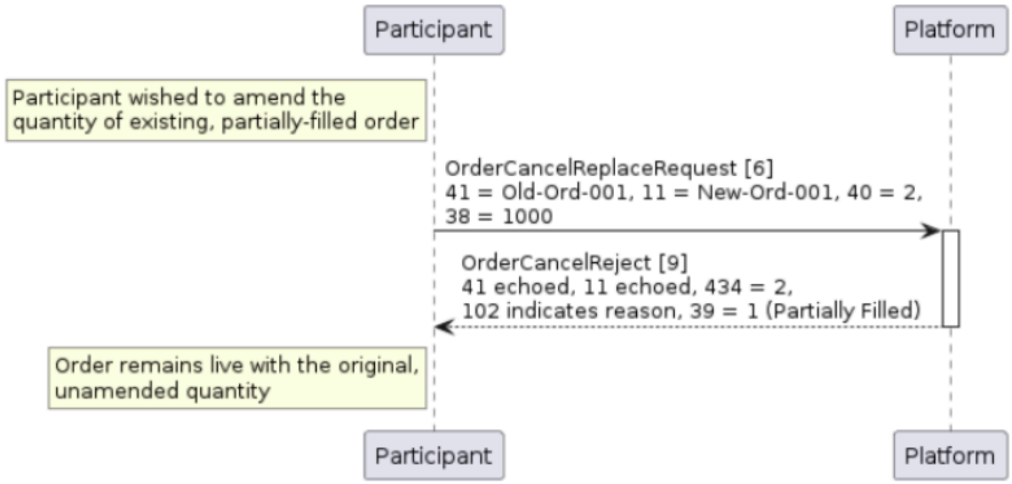


Example 7: CancelReplaceRequest [G] for Existing Order

```
8=FIXT.1.1 | 9=127 | 35=G | 49=SENDER_ID | 56=TARGET_ID | 34=66 | 52=20240517
-19:17:08 | 50=SENDER_SUB_ID | 41=1182560827 | 11=1182560836 | 55=BOPT-ERCOT
-HB_NORTH-251107-13 | 54=2 | 40=2 | 38=1000 | 44=100 | 10=061 |
```

If the order cannot be modified for any reason (for example if the requested price/order size are not acceptable, or the order has already expired), then the EXI Platform will respond with an OrderCancelReject [9] message indicating the reason. When the amendment is rejected, the existing order remains with its prior attributes.

Figure 7: Unsuccessful *OrderQty (38)* Amendment



Sample Message Sequence 8: Successful Order Price Modification
Modification of an existing order's price.

Client sends a new order message [D].

```
< 8=FIXT.1.1 | 9=218 | 35=D | 49=SENDER_ID | 56=EXI | 34=4 | 52=20260126
-14:19:39 | 50=SENDER_SUB_ID | 11=Order133 | 21=1 | 1=ACCOUNT | 55=BFUT500
-ERCOT-HB_NORTH-260126-21 | 54=1 | 60=20260126-14:19:39 | 40=2 | 59=1 | 44=5 |
38=2 | 1028=Y | 582=2 | 10=100 |
```

EXI acknowledges the order.

```
> 8=FIXT.1.1 | 9=365 | 35=8 | 34=4 | 49=EXI | 52=20260126-14:19:43.351919665 |
56=SENDER_ID | 57=SENDER_SUB_ID | 1=ACCOUNT | 6=0.00 | 11=Order133 | 14=0 |
17=7V9Z6K1G02KQ | 22=8 | 31=0.00 | 32=0 | 37=7QXG5N028051 | 38=2 | 39=0 | 40=2
| 44=5.00 | 48=BFUT500-ERCOT-HB_NORTH-260126-21 | 54=1 | 55=BFUT500-ERCOT
-HB_NORTH-260126-21 | 59=1 | 60=20260126-14:19:43.342238079 | 99=0.00 | 150=0 |
151=2 | 581=1 | 582=2 | 1028=Y | 10=125 |
```

Client sends a [CancelReplaceRequest [G]](#) to amend price.

```
< 8=FIXT.1.1 | 9=176 | 35=G | 49=SENDER_ID | 56=EXI | 34=8 | 52=20260126
-14:21:15 | 41=Order133 | 11=REPLACE-Order133 | 21=1 | 55=BFUT500
-ERCOT-HB_NORTH-260126-21 | 54=1 | 60=20260126-14:21:15 | 40=2 | 44=10 | 10=151
|
```

EXI accepts the amendment.

```
> 8=FIXT.1.1 | 9=391 | 35=8 | 34=8 | 49=EXI | 52=20260126-14:21:19.109694976 |
56=TARGET_ID | 57=TARGET_SUB_ID | 1=ACCOUNT | 6=0.00 | 11=REPLACE-Order133 |
14=0 | 17=7V9Z7Q5002KQ | 22=8 | 31=0.00 | 32=0 | 37=7QXG5N028051 | 38=2 | 39=0
| 40=2 | 41=Order133 | 44=10.00 | 48=BFUT500-ERCOT-HB_NORTH-260126-21 | 54=1 |
55=BFUT500-ERCOT-HB_NORTH-260126-21 | 59=0 | 60=20260126-14:21:19.100007245 |
99=0.00 | 150=5 | 151=2 | 581=1 | 582=2 | 1028=Y | 10=224 |
```

Sample Message Sequence 9: Order Replace Rejection
Rejection of an order amendment.

Client sends a CancelReplaceRequest [G].

```
< 8=FIXT.1.1 | 9=171 | 35=G | 49=SENDER_ID | 56=EXI | 34=42 | 52=20260122
-20:54:24 | 41=Order121 | 11=REPLACE-Order129 | 21=1 | 55=BFUT500-ERCOT
-HB_NORTH-260122-21 | 54=1 | 60=20260122-20:54:24|40=2 | 10=181 |
```

EXI rejects the amendment, listing “Unknown order” as the reason.

```
> 8=FIXT.1.1 | 9=151 | 35=9 | 34=37 | 49=EXI | 52=20260122-20:54:24.404859302 |
56=TARGET_ID | 11=REPLACE-Order129 | 37=NONE | 39=8 | 41=Order121 | 58=Unknown
order | 102=1 | 434=2 | 10=034 |
```

Order Cancellation

Participants may request to cancel any open order using the OrderCancelRequest [F] message.

The order to be canceled is identified using its last-entered *ClOrdID (11)* value for the *OrigClOrdID (41)* field in the cancel message.

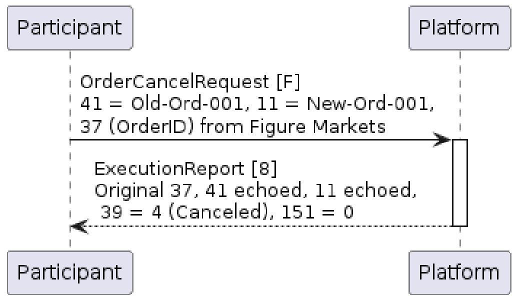
If the order has been canceled, the ElectronX Platform will respond with an ExecutionReport [8] message which echoes many of the key (updated) order attributes, and with *ExecType (150)* = 4 (canceled).

Table 18: OrderCancelRequest [F] Message

Tag	Field Name	Req’d	Data Type	Comments
< Standard Header >		Y		35 = F
11	ClOrdID	Y	String	Fresh, participant-generated, unique reference for this OrderCancelRequest. Must be different from the original *ClOrdID (11)* for the order.
41	OrigClOrdID	Y	String	The last (participant-generated) *ClOrdID (11)* reference for the order. This might be from the original NewOrderSingle or prior amends.
55	Symbol	Y	String	Instrument symbol as indicated on the order.
< Standard Trailer >		Y		

If the request to cancel the order passed validation, then the ElectronX Platform will acknowledge the order cancellation with an ExecutionReport [8] message indicating *OrdStatus (39)* = 4 (Canceled), and *LeavesQty (151)* = 0.

Figure 8: Successful Cancellation of Existing Order



Example 8: OrderCancelRequest [F] Message

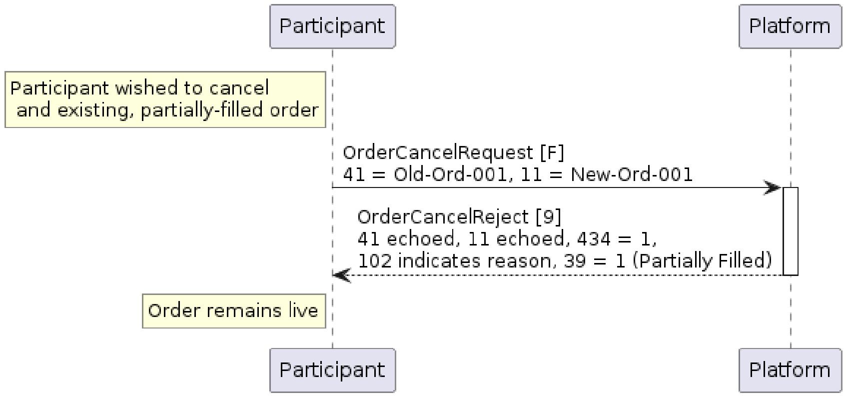
```
8=FIXT.1.1 | 9=226 | 35=F | 34=3 | 49=SENDER_ID | 50=SENDER_SUB_ID |
52=20251124-19:56:12.000 | 56=TARGET_ID | 11=CANCEL_20251124_195612 |
41=ORDER_20251124_195606 | 55=BFUT100-ERCOT-HB_TEST-251124-23 | 60=20251124
-19:56:12 | 10=189
```

Rejected attempts to cancel orders for any reason (for example order is already filled/cancelled/rejected) are rejected using the OrderCancelReject [9] message, which identifies the reason for the rejection.

Table 19: OrderCancelReject [9] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = 9
11	ClOrdID	Y	String	Echoed from the OrderCancelRequest.
37	OrderID	Y	String	Will be 'NONE' for unknown orders.
39	OrdStatus	Y	Char	Status of the cancellation request: 8 = Rejected
41	OrigClOrdID	Y	String	Echoed from the OrderCancelRequest.
58	Text	Y	String	Free text string providing additional rejection reason.
102	CxlRejReason	Y	Int	Reason for the rejection: 0 = Too late to cancel 1 = Unknown order 2 = Re-use of ClOrdID in Session 6 = Duplicate ClOrdID 18 = Invalid price increment (tick size) 99 = Other
434	CxlRejResponseTo	Y	Char	1 = Order cancel request 2 = Order amend request
< Standard Trailer >		Y		

Figure 9: Unsuccessful Attempt to Cancel Existing Order



Example 9: Rejection of Cancel Attempt

```
8=FIXT.1.1 | 9=145 | 35=9 | 34=91 | 49=TARGET_ID | 52=20240521
-09:26:30.378549737 | 56=SENDER_ID | 57=SENDER_SUB_ID | 11=1886428723 | 37=NONE
| 39=8 | 41=1886428676 | 58=Unknown order | 102=1 | 434=1 | 10=198 |
```

Sample Message Sequence 10: Successful Order Cancellation

Request to cancel an existing order.

Client submits NewOrderSingle [D].

```
< 8=FIXT.1.1 | 9=220 | 35=D | 49=SENDER_ID | 56=EXI | 34=23 | 52=20260122
-20:48:27 | 50=SENDER_SUB_ID | 11=Order128 | 21=1 | 1=ACCOUNT | 55=BFUT500
-ERCOT-HB_NORTH-260122-21 | 54=1 | 60=20260122-20:48:27 | 40=2 | 59=1 | 44=10 |
38=2 | 1028=Y | 582=2 | 10=170 |
```

EXI acknowledges the new order and rests it in the book.

```
> 8=FIXT.1.1 | 9=367 | 35=8 | 34=21 | 49=EXI | 52=20260122-20:48:27.923776511 |
56=TARGET_ID | 57=TARGET_SUB_ID | 1=ACCOUNT | 6=0.00 | 11=Order128 | 14=0 |
17=7RWFA5T0018V | 22=8 | 31=0.00 | 32=0 | 37=7QXG5N020051 | 38=2 | 39=0 | 40=2
| 44=10.00 | 48=BFUT500-ERCOT-HB_NORTH-260122-21 | 54=1 | 55=BFUT500-ERCOT
-HB_NORTH-260122-21 | 59=1 | 60=20260122-20:48:27.913942017 | 99=0.00 | 150=0 |
151=2 | 581=1 | 582=2 | 1028=Y | 10=186 |
```

Client sends OrderCancelRequest [F].

```
< 8=FIXT.1.1 | 9=139 | 35=F | 49=SENDER_ID | 56=EXI | 34=27 | 52=20260122
-20:49:10 | 41=Order128 | 11=CANCEL-Order128 | 55=BFUT500-ERCOT-HB_NORTH
-260122-21 | 54=1 | 10=191 |
```

EXI cancels the order and returns ExecutionReport [8].

```
> 8=FIXT.1.1 | 9=391 | 35=8 | 34=25 | 49=EXI | 52=20260122-20:49:10.191535190 |
56=TARGET_ID | 57=TARGET_SUB_ID | 1=ACCOUNT | 6=0.00 | 11=CANCEL-Order128 |
14=0 | 17=7RWFFM10018V | 22=8 | 31=0.00 | 32=0 | 37=7QXG5N020051 | 38=2 | 39=4
| 40=2 | 41=Order128 | 44=10.00 | 48=BFUT500-ERCOT-HB_NORTH-260122-21 | 54=1 |
```

```
55=BFUT500-ERCOT-HB_NORTH-260122-21 | 59=1 | 60=20260122-20:49:10.183957206 |
99=0.00 | 150=4 | 151=0 | 581=1 | 582=2 | 1028=Y | 10=170 |
```

Sample Message Sequence 11: Order Cancel Rejection
Exchange rejection of cancel request message.

Client sends an OrderCancelRequest [F] for the order in the previous sample message sequence.

```
< 8=FIXT.1.1 | 9=145 | 35=F | 49=SENDER_ID | 56=EXI | 34=33 | 52=20260122
-20:51:18 | 41=Order128 | 11=CANCEL-Order128 | 55=BFUT500-ERCOT-HB_NORTH
-260122-21 | 54=1 | 44=15 | 10=198 |
```

EXI rejects the cancellation request because the referenced order was already cancelled.

```
> 8=FIXT.1.1 | 9=150 | 35=9 | 34=30 | 49=EXI | 52=20260122-20:51:19.053399575 |
56=TARGET_ID | 11=CANCEL-Order128 | 37=NONE | 39=8 | 41=Order128 | 58=Unknown
order | 102=1 | 434=1 | 10=213 |
```

Mass Cancellation

Participants may request to cancel multiple open orders using the OrderMassCancelRequest [q] message. Multiple orders can be cancelled for a FIX session as outlined by the following table:

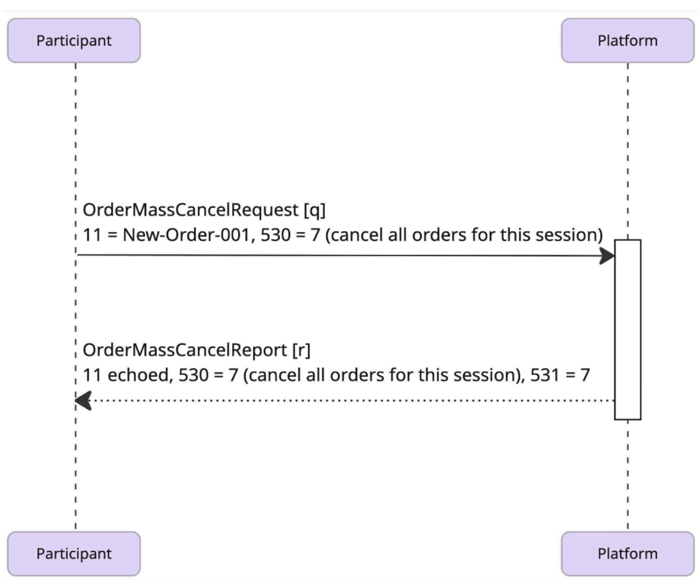
Mass Cancel Type	***MassCancelRequestType (530)***	**Required Filters**
By Instrument	Cancel Orders for an Instrument (1)	*Symbol (55)* or *SecurityID (48)* + *SecurityIDSource (22)*
By account	Cancel All Orders (7)	*Account (1)*
By account + ClOrdLinkID	Cancel All Orders (7)	*Account (1)* and *ClOrdLinkID (583)*
Cancel all orders for the FIX session	Cancel All Orders (7)	Do not provide any filters

Table 20: OrderMassCancelRequest [q] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = q
1	Account	N	String	Account reference if mass cancellation is by account or account + ClOrdLinkID.
11	ClOrdID	Y	String	Fresh, participant-generated, unique reference for this OrderMassCancelRequest. Must be different from the original *ClOrdID (11)* for the order.

Tag	Field Name	Req'd	Data Type	Comments
22	SecurityIDSource	C	String	Required if *SecurityID (48)* is provided. The only acceptable value is 8 (Exchange Symbol).
48	SecurityID	C	String	Instrument symbol if mass cancel is by symbol. Either *SecurityID (48)* or *Symbol (55)* must be provided if *MassCancelRequestType (530)* is 7 (symbol). Required if *SecurityIDSource (22)* is provided.
55	Symbol	C	String	Instrument symbol if mass cancel is by symbol. Either *SecurityID (48)* or *Symbol (55)* must be provided if *MassCancelRequestType (530)* is 7 (symbol).
530	MassCancelRequestType	Y	Char	1 = Cancel orders for a security 7 = Cancel all orders
583	ClOrdLinkID	N	String	Permits sender to tie together groups of orders in which trades resulting from orders are associated for a specific purpose.
< Standard Trailer >		Y		

Figure 10: Unsuccessful Attempt to Cancel Existing Order



Example 10: Example of mass order cancel message for open orders

```
8=FIXT.1.1 | 9=97 | 35=q | 49=SENDER | 56=EXI | 34=101 |52=20260423-
14:50:11.121575160 | 11=New-Order-001 | 530=7 | 60=20260423-14:50:11.121575160
| 10=099 |
```

Table 21: OrderMassCancelReport [r] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = r
11	ClOrdID	Y	String	Echoed from the OrderMassCancelRequest.
22	SecuritySourceID	Y	String	8 = Exchange Symbol
48	SecurityID	Y	String	Will match *Symbol (55)*.
55	Symbol	Y	String	Instrument symbol.
58	Text	N	String	Used to explain the reason for a mass cancel reject.
60	TransactTime	Y	UTCTimestamp	Time at which OrderMassCancelRequest [q] was initiated by the matching engine.
530	MassCancelRequestType		Char	1 = Cancel orders for a security 7 = Cancel all orders
531	MassCancelResponse	Y	Char	0 = Cancel Request Rejected - See *MassCancelRejectReason (532)* 1 = Cancel orders for a security 7 = Cancel all orders
532	MassCancelRejectReason	Y	Int	This is used in conjunction with a description of the reason for the rejection in *Text (58)*: 99 = Other
< Standard Trailer >		Y		

The OrderMassCancelReport [r] message is sent by the EXI Platform in response to an OrderMassCancelRequest [q]. Exactly one report is generated for each mass cancel request, regardless of the number of orders affected.

When a mass cancel request is accepted, the report will indicate success via the *MassCancelResponse (531)* field, whose value reflects the scope of the cancellation performed and mirrors the *MassCancelRequestType (530)* submitted on the request. Individual order cancellations resulting from an accepted request are communicated separately using ExecutionReport [8] messages with *ExecType (150)* = 4 (Canceled).

When a mass cancel request is rejected, the EXI Platform returns a report indicating rejection. In this case, *MassCancelResponse (531)* is set to 0 (Cancel Request Rejected), *MassCancelRejectReason (532)* is populated, and *Text (58)* provides a human-readable explanation of the rejection. When a mass cancel request is rejected, no execution report messages are generated for order cancellations.

Certain mass cancel requests require additional filters in order to be valid:

- When canceling all orders using a *ClOrdLinkID (583)*, *Account (1)* must also be supplied because ClOrdLinkID values are not globally unique across accounts.
    - If not provided, this results in an OrderMassCancelReport [r] indicating rejection, as described above.
- Requests to cancel orders for a security require either *Symbol (55)* or *SecurityID (48)* + *SecurityIDSource (22)* to be provided.
    - If neither is provided, this results in a BusinessMessageReject [j] message.

Example 11: Rejection of a mass order cancel request where a *ClOrdLinkID (583)* was provided without *Account (1)*.

```
8=FIXT.1.1 | 9=162 | 35=r | 49=TARGET | 56=EXI | 34=102 | 52=20260423-
12:39:14.321454163 | 11=EXAMPLE-ORDER-001 | 530=7 | 531=0 | 532=0 | 58=at least
one account must be provided when ClOrdLinkID is provided | 10=033 |
```

Mass Quote Protection (MQP)

Mass Quote Protection (MQP) is a risk control that automatically cancels an account's remaining eligible resting orders when that account trades too much quantity within a short, rolling time window. MQP is intended to limit the impact of runaway quoting or unexpected market conditions by quickly removing outstanding liquidity once an execution threshold is reached. MQP is configured per account with two parameters:

- Interval (Y): duration of the rolling time window (e.g., 3s)
- Traded Quantity (X): quantity that may be executed within the interval before MQP triggers (e.g., 10)

MQP triggers when the sum of executed quantity for a bucket within a rolling interval Y reaches or exceeds the traded quantity threshold X. The interval is rolling, not fixed to wall-clock boundaries. At any moment, the system considers executions in the trailing Y seconds when computing the bucket's traded quantity. As a result:

- executions separated by more than Y seconds do not accumulate toward the same trigger
- executions clustered within Y seconds do accumulate

MQP tracking is done per bucket, where a bucket is defined as:

- (account, *ClOrdLinkId*)

Orders are associated to a bucket using the order's *ClOrdLinkId (583)* value. Orders submitted without a *ClOrdLinkId* (blank / missing) are still subject to MQP. They are tracked in their own bucket:

- (account, "")

When MQP triggers for a given bucket, the matching engine cancels all remaining eligible resting orders in that same bucket:

- same account
- same *ClOrdLinkId* (or blank bucket)
- regardless of instrument
- regardless of side (buy/sell)

Orders in other buckets are not affected, including:

- same account but different *ClOrdLinkId*
- same *ClOrdLinkId* but different account

MQP is applied as part of the matching process:

- The execution that causes the bucket's traded quantity to reach or exceed the threshold is processed normally.
- Once MQP triggers, the matching engine cancels remaining eligible resting orders in that bucket.
- MQP does not prevent an aggressing order from matching against all eligible resting orders it can reach during that matching event. This means the traded quantity may exceed the configured threshold during a single aggressing order's sweep.

After MQP triggers and cancels orders in a bucket, the bucket's rolling interval tracking is reset after 150ms. The account may reuse the same *ClOrdLinkId (583)* and continue trading without automatic cancellation unless and until the threshold is reached again in a new interval.

The MQP Interval (Y) and Traded Quantity (X) values must be configured by EXI's Market Operations team. **Please reach out to support@electronx.com for MQP configuration.**

Trade Busts

When the exchange busts a trade, the EXI Platform will send an ExecutionReport [8] message for each trade side that echoes the order attributes but with *ExecType (150)* = H (trade cancel).

Note that the order attributes reflect the values at the time the trade occurred, not those from the latest state of the order.

Certain attributes of interest on the bust Execution Report are highlighted in Table 20 below.

Table 22: ExecutionReport [8] Message for Busted Trade

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		**Y**		**35 = 8**
6	**AvgPx**	**Y**	**Price**	**The volume-weighted average price of all trades against this order as recorded at the time of the trade.**
11	**ClOrdID**	**Y**	**String**	**The participant-assigned ClOrdID value as of the trade.**
14	**CumQty**	**Y**	**Qty**	**Cumulative quantity on the order as of the trade.**
17	**ExecID**	**Y**	**String**	**Unique identifier for the bust Execution Report as assigned by EXI. Typically a 13-character alphanumeric string.**
19	**ExecRefID**	**Y**	**String**	**The identifier for the Execution Report of the trade that was busted.**

22	SecurityIDSource	Y	Int	8 = Exchange symbol
31	LastPx	Y	Price	Price of the trade.
32	LastQty	Y	Qty	Quantity of the trade.
37	OrderID	Y	String	Unique identifier for the order as assigned by EXI. Typically a 13-character alphanumeric string.
39	OrdStatus	Y	Char	The status of the order as of the trade: 1 = Partially filled 2 = Fully filled
40	OrdType	Y	Char	Order type. See Table 12: *OrdType (40)* definitions.
44	Price	Y	Price	Order limit price.
48	SecurityID	Y	String	Will always equal *Symbol (55).*
54	Side	Y	Char	1 = Buy 2 = Sell
55	Symbol	Y	String	Instrument symbol.
58	Text	Y	String	The reason for the trade bust as given by the exchange.
59	TimeInForce	Y	Char	See Table 13: *TimeInForce (59)* definitions.
60	TransactTime	Y	UTCTimestamp	Timestamp when the bust occurred.
150	ExecType	Y	Char	H = Trade Cancel
151	LeavesQty	Y	Qty	Remaining, unexecuted quantity left on the order as of the trade.
381	GrossTradeAmt	Y	Amt	Total amount traded across all fills for this order as of the trade. Equal to *AvgPx* (6) x *CumQty (14)*.
582	CustOrderCapacity	N	Int	The values below correspond 1-to-1 with standard CTI codes: 1=OWN_ACCOUNT

				2=PROPRIETARY_ACCOUNT **3=FINANCIAL_ADVISOR** **4=ALL_OTHER**
880	**TrdMatchID**	**Y**	**String**	**The trade identifier as assigned by EXI. Will match the *TradeID (1003)* value on market data updates. Typically a 13-character alphanumeric string.**
1028	**ManualOrderIndicator**	**Y**	**Boolean**	**Indicates if the order was initially received manually (as opposed to algorithmically).**
1057	**AggressorIndicator**	**C**	**Boolean**	**Always populated for trades. Identifies whether this order was the aggressor in the trade.**
< Standard Trailer >		**Y**		

Sample Message Sequence 12: Trade Bust Workflow
Post-trade cancellation initiated by exchange operations to correct an erroneous execution.

Client submits new order.
```
< 8=FIXT.1.1 | 9=225 | 35=D | 49=SENDER_ID | 56=EXI | 34=45 |
52=20260122-20:55:40 | 50=SENDER_SUB_ID | 11=Order131 | 21=1 | 1=ACCOUNT |
55=BFUT500-ERCOT-HB_NORTH-260122-21 | 54=1 | 60=20260122-20:55:40 | 40=2 | 59=1
| 44=10 | 38=2 | 1028=Y | 582=2 | 18=T | 10=154 |
```

EXI acknowledges the order.
```
> 8=FIXT.1.1 | 9=372 | 35=8 | 34=40 | 49=EXI | 52=20260122-20:55:40.478553756 |
56=TARGET_ID | 57=TARGET_SUB_ID | 1=ACCOUNT | 6=0.00 | 11=Order131 | 14=0 |
17=7RWG3MVG018V | 18=T | 22=8 | 31=0.00 | 32=0 | 37=7QXG5N024051 | 38=2 | 39=0
| 40=2 | 44=35.00 | 48=BFUT500-ERCOT-HB_NORTH-260122-21 | 54=1 | 55=BFUT500
-ERCOT-HB_NORTH-260122-21 | 59=1 | 60=20260122-20:55:40.467403291 | 99=0.00 |
150=0 | 151=2 | 581=1 | 582=2 | 1028=Y | 10=214 |
```

EXI reports that the order has been filled.
```
> 8=FIXT.1.1 | 9=437 | 35=8 | 34=41 | 49=EXI | 52=20260122-20:55:40.478755506 |
56=TARGET_ID | 57=TARGET_SUB_ID | 1=ACCOUNT | 6=35.00 | 11=Order131 | 12=0.10 |
13=3 | 14=2 | 17=7RWG3PEG018V | 18=T | 22=8 | 31=35.00 | 32=2 | 37=7QXG5N024051
| 38=2 | 39=2 | 40=2 | 44=35.00 | 48=BFUT500-ERCOT-HB_NORTH
-260122-21 | 54=1 | 55=BFUT500-ERCOT-HB_NORTH-260122-21 | 59=1 | 60=20260122
-20:55:40.467403291 | 99=0.00 | 119=70.00 | 150=F | 151=0 | 381=70.00 | 581=1 |
582=2 | 828=0 | 880=7RWG3NMG018V | 1028=Y | 1057=Y | 10=161 |
```
The executed trade meets the cancellation criteria, so EXI's Operations Team busts the trade and sends an Execution Report to the trader with *ExecType (150)* = H (Trade Cancel).

```
> 8=FIXT.1.1 | 9=439 | 35=8 | 34=50 | 49=EXI | 52=20260122-21:00:04.910765563 |
56=TARGET_ID | 57=TARGET_SUB_ID | 1=ACCOUNT | 6=35.00 | 11=Order131 | 14=2 |
17=7QZ3GKJ0J18P | 18=T | 19=7RWG3PEG018V | 22=8 | 31=35.00 | 32=2 |
37=7QXG5N024051 | 38=2 | 39=2 | 40=2 | 44=35.00 | 48=BFUT500-ERCOT-HB_NORTH
-260122-21 | 54=1 | 55=BFUT500-ERCOT-HB_NORTH-260122-21 | 58=TEST | 59=1 |
60=20260122-21:00:04.336832006 | 99=0.00 | 150=H | 151=0 | 381=70.00 | 581=1 |
582=2 | 828=0 | 880=7RWG3NMG018V | 1028=Y | 1057=Y | 10=093 |
```

Drop Copy Sessions

Overview

The EXI Drop Copy FIX feed provides FIX execution reports to exchange participants and third-party service providers. The service can be configured to provide data for a specific participant, clearing member, or all participants. Drop copies provided via this service will include all execution reports regardless if the orders or quotes were sent via a FIX session or the EXI Trading UI.

Supported FIX Messages

All FIX traffic on the EXI Drop Copy feed includes the same standard header and trailer as used in all other FIX sessions on the platform. The EXI Drop Copy feed supports the delivery of Execution Reports for order updates, fills, busts, and cancels as shown in tables Table 16: ExecutionReport [8] Message and Table 22: ExecutionReport [8] Message for Busted Trade.

Configurations

Drop Copy sessions can be configured to only transmit fill messages. To do so, ask our support team to set DropCopyFillsOnly=Y when establishing a connection.

Market & Reference Data

The market data sessions for the ElectronX Platform are available via a separate FIX gateway; they are not accessible via the order-entry session.

When displaying the order book, the ElectronX Platform provides a Market-by-Order view; i.e. each order is displayed individually with a corresponding timestamp (used to determine time priority within a price level). Each order on the market data feed carries an MDEntryID reference which will be the same as the order ID from the order entry gateway, allowing participants to identify their own orders in market data. The market data feed supports different levels of requested depth from full book depth (25 levels), top of book, or any depth in between.

Subscribing to Market Data

Participants can subscribe to market data for a given symbol(s) using a MarketDataRequest [V] message.

Table 23: MarketDataRequest [V] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = V
146	NoRelatedSym	Y	NumInGroup	Number of symbols requested
→ 55	Symbol	Y	String	Instrument symbol.
262	MDReqID	Y	String	Unique ID for this request.
263	SubscriptionRequestType	Y	Char	Type of subscription requested: 0 = Snapshot 1 = Snapshot plus Updates 2 = Delete previous request (unsubscribe)
264	MarketDepth	Y	Int	Depth requested (Maximum of 25 levels): 0 = Full book depth 1 = Top of book (best prices only) 2+ = Number of levels requested
267	NoMDEntryTypes	N	NumInGroup	
→ 269	MDEntryType	N	char	A repeating group of MD Entry Types requested 0 = Bid 1 = Offer 2 = Trade 4 = Opening Price 5 = Closing Price 6 = Settlement Price 7 = Trading Session High Price 8 = Trading Session Low Price B = Trade Volume C = Open Interest
< Standard Trailer >		Y		

If the MarketDataRequest [V] message is valid, ElectronX Platform will respond with a single MarketDataSnapshotFullRefresh [W] message for each requested Instrument, providing details of all orders in the order book (all levels as a repeating group within a single message). Note that the (repeating) *MDEntryType (269)* field can be specified if required to tailor the elements returned.

Example 12: Request Snapshot of All Market Data via MarketDataRequest [V] Message

```
8=FIXT.1.1 | 9=137 | 35=V | 34=2 | 49=SENDER_ID | 52=20251125-18:40:11.000 |
56=TARGET_ID | 146=1 | 55=BFUT100-ERCOT-HB_TEST-251125-23 |
262=MD_REQ_20251125_134006 | 263=1 | 264=5 | 10=204
```

Note that since the EXI returns market data split by order, requesting only the best price using *MarketDepth (264)* = 1 (Top of book) may still return multiple bid and offer entries if there is more than one order at this price. For [MarketDataSnapshotFullRefresh [W]] messages with *MDEntryType (269)* = 2, the message returns information only for the latest trade. In the case where *MDEntryType (269)* = B (Trade Volume), the *MDEntrySize (271)* is the total aggregate traded volume for the relevant contract since listing.

Table 24: MarketDataSnapshotFullRefresh [W] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = W
22	SecurityIDSource	Y	Int	8 = Exchange symbol
48	SecurityID	Y	String	Will always equal *Symbol (55)*.
55	Symbol	Y	String	Instrument symbol.
167	SecurityType	Y	String	Indicates type of security.
262	MDReqID	Y	String	The ID of the request as indicated on the request.
268	NoMDEntries	Y	NumInGroup	The number of market data levels returned. Might be zero if the symbol is valid but there are currently no bids/offers in this symbol.
→ 269	MDEntryType	Y	Char	Type of entry: 0 = Bid 1 = Offer 2 = Trade 4 = Opening Price 5 = Closing Price 6 = Settlement Price 7 = Trading Session High Price 8 = Trading Session Low Price B = Trading Session Volume C = Open Interest
→ 270	MDEntryPx	Y	Price	Price level.
→ 271	MDEntrySize	C	Qty	Quantity of the individual order or trade, or the aggregate quantity where *MDEntryType (269)* = B (Trading Session Volume). Not sent for session open / high / low.
→ 272	MDEntryDate	Y	UTCDateOnly	Time priority (date) of the book event.
→ 273	MDEntryTime	Y	UTCTimeOnly	Time priority (time) of the book event (in UTC).
→ 278	MDEntryID	Y	String	Sent for *MDEntryType (269)* = 0 (Bid) or 1 (Offer). Unique reference for the entry.

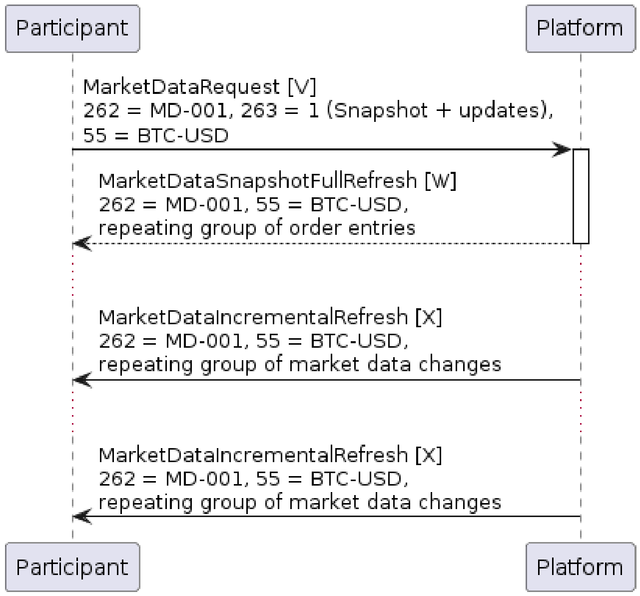
Tag	Field Name	Req'd	Data Type	Comments
				Typically 13-character alphanumeric string. Will match sender's *OrderID (37)*.
→ 828	TrdType	N	Int	0 = Regular Trade
→ 1070	MDQuoteType	C	Int	Identifies market data quote type. Only sent for *MDEntryType (269)* = 4 (Opening Price) as 1 (Tradable).
→ 336	TradingSessionID	C	String	[See Table 1: Instrument States.]
→ 286	OpenCloseSettlFlag	N	MultipleCharValue	Flag that identifies a market data entry. Valid values: 0 = Daily Open / Close / Settlement 1 = Session Open / Close / Settlement 2 = Delivery Settlement entry 3 = Expected entry 4 = Entry from previous business day 5 = Theoretical Price value
1151	SecurityGroup	C	String	The name of the group of related securities to which this instrument belongs.
< Standard Trailer >		Y		

Example 13: Initial Market Data Snapshot (repeating groups color-coded)

```
8=FIXT.1.1 | 9=1082 | 35=W | 34=2 | 49=SENDER_ID | 52=20251125
-18:40:11.898784152 | 56=TARGET_ID | 22=8 | 48=BFUT100-ERCOT-HB_TEST-251125-23
| 55=BFUT100-ERCOT-HB_TEST-251125-23 | 167=EVENT | 262=MD_REQ_20251125_134006 |
268=8 | 269=0 | 270=22.75 | 271=5 | 272=20251125 | 273=18:37:59.099215855 |
278=6KCP14C002BN | 269=0 | 270=22.50 | 271=5 | 272=20251125 |
273=18:37:59.099215855 | 278=6KC2EJD002BN | 269=0 | 270=22.25 | 271=5 |
272=20251125 | 273=18:37:59.099215855 | 278=6KCP15Y002BN | 269=0 | 270=22.00 |
271=5 | 272=20251125 | 273=18:37:59.099215855 | 278=6K9E4TGG02BN | 269=0 |
270=21.75 | 271=5 | 272=20251125 | 273=18:37:59.099215855 | 278=6KCP16Q002BN |
269=1 | 270=27.75 | 271=5 | 272=20251125 | 273=18:37:59.099215855 |
278=6KD2Z2H002BN | 269=1 | 270=28.00 | 271=5 | 272=20251125 |
273=18:37:59.099215855 | 278=6K9HGFV002BN | 269=1 | 270=28.25 | 271=5 |
272=20251125 | 273=18:37:59.099215855 | 278=6KCP189G02BN | 336=OPEN |
1151=BFUT100-ERCOT-HB_TEST | 10=075 |
```

Note the response will contain ONLY a snapshot of the current order book; it does not contain information about historic trades that have occurred on the platform.

Figure 11: Successful Market Data Subscription with Snapshot and Incremental Updates



If the participants subscribed to ongoing updates for the instrument(s), the platform will then start sending unsolicited MarketDataIncrementalRefresh [X] messages which contain a mixture of trade and/or order book update messages.

These incremental market data messages may contain a repeating group of updates in a single message. For example, an incoming sell order which enters the order book, executes against a resting buy order with the remainder written to the order book will trigger a MarketDataIncrementalRefresh [X] message containing the following updates:

1. The deletion of the previous best bid (as a result of the immediate fill),
2. The addition of a new best bid (next-best bid price),
3. The deletion of the best offer (new order has a better price),
4. The addition of a new best offer (representing the balance of the incoming sell order),
5. A record relating to the trade, and
6. A record relating to updated overall market volume

Table 25: MarketDataIncrementalRefresh [X] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = X
262	MDReqID	Y	String	The ID of the request as indicated on the request.
268	NoMDEntries	Y	NumInGroup	

Tag	Field Name	Req'd	Data Type	Comments
→ 279	MDUpdateAction	Y	Char	The type of action conveyed by this block: 0 = New 1 = Change 2 = Delete
→ 269	MDEntryType	C	Char	Type of entry. Set where *MDUpdateAction (279)* = (0) New or 1 (Change): 0 = Bid 1 = Offer 2 = Trade 4 = Opening Price 5 = Closing Price 6 = Settlement Price 7 = Trading Session High Price 8 = Trading Session Low Price B = Trade Volume C = Open Interest
→ 278	MDEntryID	N	String	Unique reference for MD entry. Typically a 13-character alphanumeric string. Will match sender's *OrderID (37)*. For MDEntryType = 2 (Trade), its value will equal to TradeID + "P" char.
→ 1070	MDQuoteType	C	Int	Identifies market data quote type. Only sent for *MDEntryType (269)* = 4 (Opening Price) as 1 (Tradable).
→ 55	Symbol	Y	String	Instrument symbol.
→ 48	SecurityID	Y	String	Will always equal *Symbol (55).*
→ 22	SecurityIDSource	Y	Int	8 = Exchange symbol
→167	SecurityType	Y	String	Indicates type of security.
→1151	SecurityGroup	C	String	The name of the group of related securities to which this instrument belongs.
→ 270	MDEntryPx	N	Price	- Order level price where *MDEntryType (269)* = 0 (Bid) or 1 (Offer) - Traded price where *MDEntryType (269)* = 2 (Trade) - Total value traded where *MDEntryType (269)* = B (Trade Volume)
→ 271	MDEntrySize	N	Qty	- Remaining order size where *MDEntryType (269)* = 0 (Bid) or 1 (Offer). Will be zero where *MDUpdateAction (279)* = 2 (Delete)

Tag	Field Name	Req'd	Data Type	Comments
				- Trade size where *MDEntryType (269)* = 2 (Trade) - Total quantity traded where *MDEntryType (269)* = B (Trade Volume)
→ 272	MDEntryDate	N	UTCDateOnly	The date on which the book event occurred.
→ 273	MDEntryTime	N	UTCTimeOnly	The time at which the book event occurred (in UTC).
→ 828	TrdType	N	Int	0 = Regular Trade
→ 1003	TradeID	N	String	Only sent for trade updates. Will match the *TrdMatchID (880)* in the ExecutionReport fill, allowing Participants to identify their own trades within market data.
→ 2446	AggressorSide	N	Char	Only sent for trades. Indicates which side was the aggressor in a trade. 1 = Buy 2 = Sell
→ 336	TradingSessionID	C	String	See Table 1: Instrument States.
→ 286	OpenCloseSettlFlag	N	MultipleCharValue	Flag that identifies a market data entry. Valid values: 0 = Daily Open / Close / Settlement 1 = Session Open / Close / Settlement 2 = Delivery Settlement entry 3 = Expected entry 4 = Entry from previous business day 5 = Theoretical Price value
< Standard Trailer >		Y		

Example 14: Market Data Incremental Refresh Containing Multiple Updates (repeating groups color-coded)

```
8=FIXT.1.1 | 9=676 | 35=X | 34=5 | 49=SENDER_ID | 52=20251125
-00:03:41.720412588 | 56=TARGET_ID | 262=MD_REQ_20251124_190220 | 268=3 | 279=0
| 269=1 | 278=6JSX4KH002BN | 55=BFUT100-ERCOT-HB_TEST-251124-23 |
48=BFUT100-ERCOT-HB_TEST-251124-23 | 22=8 | 167=EVENT | 1151=BFUT100-ERCOT
-HB_TEST | 270=79.75 | 271=11 | 272=20251125 | 273=00:03:41.674009712 | 279=0 |
269=1 | 278=6JT7CD3G02BN | 55=BFUT100-ERCOT-HB_TEST-251124-23 | 48=BFUT100
-ERCOT-HB_TEST-251124-23 | 22=8 | 167=EVENT | 1151=BFUT100-ERCOT-HB_TEST |
270=79.75 | 271=6 | 272=20251125 | 273=00:03:41.674009712 | 279=2 | 269=1 |
278=6JT7CCA002BN | 55=BFUT100-ERCOT-HB_TEST-251124-23 | 48=BFUT100-ERCOT
```

```
-HB_TEST-251124-23 | 22=8 | 167=EVENT | 1151=BFUT100-ERCOT-HB_TEST | 270=79.50
| 271=6 | 272=20251125 | 273=00:03:41.674009712 | 10=097 |
```

Sample Message Sequence 13: Market Data Subscription Workflow
Subscribe to market data updates for specified instruments.

Client sends MD request [V] for BFUT500 ERCOT NORTH HUB HE22 with incremental updates.

```
< 8=FIXT.1.1 | 9=149 | 35=V | 49=SENDER_ID | 56=EXI | 34=59 | 52=20260122
-21:01:58 | 146=1 | 55=BFUT500-ERCOT-HB_NORTH-260122-22 | 262=MDRequestTest1 |
263=1 | 264=1 | 265=1 | 267=2 | 269=0 | 269=1 | 10=081 |
```

EXI responds with a full book snapshot [W] (repeating groups color-coded).

```
> 8=FIXT.1.1 | 9=354 | 35=W | 34=58 | 49=EXI | 52=20260122-21:01:58.464086850 |
56=TARGET_ID | 22=8 | 48=BFUT500-ERCOT-HB_NORTH-260122-22 | 55=BFUT500-ERCOT
-HB_NORTH-260122-22 | 167=EVENT | 262=MDRequestTest1 | 268=2 | 269=0 |
270=25.50 | 271=10 | 272=20260122 | 273=20:53:10.546649026 | 278=7RRTXFH002CR |
269=1 | 270=31.00 | 271=10 | 272=20260122 | 273=20:53:10.546649026 |
278=7RRSFD3002CR | 1151=BFUT500-ERCOT-HB_NORTH | 10=011 |
```

Incremental updates [X] describing new order book updates: adds, deletes, trades, etc. (repeated groups color-coded).

```
> 8=FIXT.1.1 | 9=481 | 35=X | 34=82 | 49=EXI | 52=20260122-21:13:37.083620384 |
56=TARGET_ID | 262=MDRequestTest1 | 268=2 | 279=0 | 269=0 | 278=7RRSFCA002CR |
55=BFUT500-ERCOT-HB_NORTH-260122-22 | 48=BFUT500-ERCOT-HB_NORTH-260122-22 |
22=8 | 167=EVENT | 1151=BFUT500-ERCOT-HB_NORTH | 270=25.25 | 271=10 |
272=20260122 | 273=21:13:37.078952355 | 279=2 | 269=0 | 278=7RRTXFH002CR |
55=BFUT500-ERCOT-HB_NORTH-260122-22 | 48=BFUT500-ERCOT-HB_NORTH-260122-22 |
22=8 | 167=EVENT | 1151=BFUT500-ERCOT-HB_NORTH | 270=25.50 | 271=10 |
272=20260122 | 273=21:13:37.078952355 | 10=228 |

> 8=FIXT.1.1 | 9=481 | 35=X | 34=83 | 49=EXI | 52=20260122-21:13:37.407351820 |
56=TARGET_ID | 262=MDRequestTest1 | 268=2 | 279=0 | 269=1 | 278=7RRXS7YG02CR |
55=BFUT500-ERCOT-HB_NORTH-260122-22 | 48=BFUT500-ERCOT-HB_NORTH-260122-22 |
22=8 | 167=EVENT | 1151=BFUT500-ERCOT-HB_NORTH | 270=30.75 | 271=10 |
272=20260122 | 273=21:13:37.401747365 | 279=2 | 269=1 | 278=7RRSFD3002CR |
55=BFUT500-ERCOT-HB_NORTH-260122-22 | 48=BFUT500-ERCOT-HB_NORTH-260122-22 |
22=8 | 167=EVENT | 1151=BFUT500-ERCOT-HB_NORTH | 270=31.00 | 271=10 |
272=20260122 | 273=21:13:37.401747365 | 10=217 |
```

Unsubscribing from Market Data

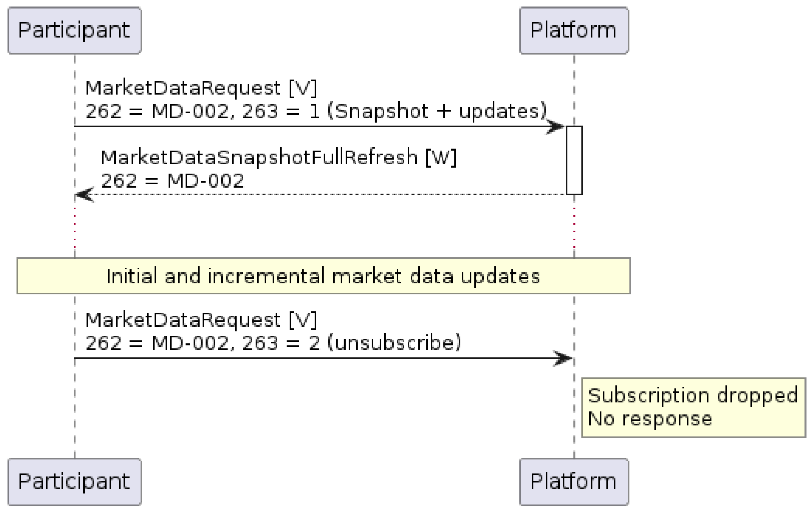
Participants can unsubscribe from a piece of market data by entering a second [MarketDataRequest [V]] message, referencing the original *MDReqID (262)* reference and setting *SubscriptionRequestType (263)* = 2 (unsubscribe).

Note that there is no explicit response to requests to unsubscribe from the ElectronX Platform; a successful unsubscription will simply prevent further MarketDataIncrementalRefresh [X] messages.

Example 15: Market Data Unsubscribe Request

```
8=FIXT.1.1| 9=137| 35=V | 34=6 | 49=SENDER_ID | 52=20251125-18:32:37.000 |
56=TARGET_ID | 146=1 | 55=BFUT100-ERCOT-HB_TEST-251125-23 |
262=MD_REQ_20251125_133057 | 263=2 | 264=5 | 10=223 |
```

Figure 12: Successful Market Data Subscription and Unsubscription



Sample Message Sequencer 14: Market Data Unsubscription

Unsubscribing to market data uses the same message as subscribing but instead includes tag *263* = 2.

```
< 8=FIXT.1.1 | 9=113 | 35=V | 49=SENDER_ID | 56=EXI | 34=92 | 52=20260122
-21:17:45 | 263=2 | 262=MDRequestTest1 | 55=BFUT500-ERCOT-HB_NORTH-260122-22 |
10=250 |
```

Reference Data

Participants may download a list of the instruments available (or reference data about them) to trade on the ElectronX Platform by submitting a SecurityListRequest [x] message.

Table 26: SecurityListRequest [x] Message

Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = x
320	SecurityReqID	Y	String	Unique ID associated with this request.
336	TradingSessionID	N	String	Used to filter for instruments of the specified market state. See Table 1: Instrument States.
559	SecurityListRequestType	Y	Int	The type of request being made: 0 = Individual symbol 4 = All Securities
55	Symbol	C	String	Required if *SecurityListRequestType (559)* = 0 (individual security).

Tag	Field Name	Req'd	Data Type	Comments
< Standard Trailer >		Y		

Example 16: Security List Request for All Instruments

```
8=FIXT.1.1 | 9=106 | 35=x | 34=2 | 49=SENDER_ID | 52=20251125-18:33:45.000 |
56=TARGET_ID | 320=SEC_REQ_20251125_133342 | 559=4 | 10=166
```

Upon receipt, the ElectronX Platform will respond with a SecurityList [y] message containing the requested information. Note that the content provided by the SecurityList [y] message does not differ depending on whether an individual symbol was listed versus all securities.

Table 27: SecurityList [y] Message

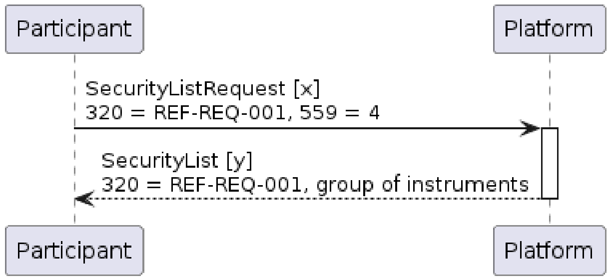
Tag	Field Name	Req'd	Data Type	Comments
< Standard Header >		Y		35 = y
146	NoRelatedSym	Y	NumInGroup	Number of symbols requested
→ 55	Symbol	Y	String	Instrument symbol.
→ 48	SecurityID	Y	String	Will always equal *Symbol (55).*
→ 22	SecurityIDSource	Y	Int	8 = Exchange symbol.
→ 460	Product	N	Int	Indicates the type of product the security is associated with: **14=ENERGY**
→ 167	SecurityType	Y	String	Indicates type of security.
→ 231	ContractMultiplier	N	Float	The ratio or multiplier to convert from "nominal" units (e.g. contracts) to total units (e.g. shares). Applicable for Fixed Income, Derivatives, etc.
→ 864	NoEvents	Y	NumInGroup	Number of repeating EventType entries.
→ → 865	EventType	Y	Int	Code to represent the type of event: 5 = Activation 7 = Last Eligible Trade Date
→ → 866	EventDate	Y	LocalMktDate	Date that the instrument first started (or will start) trading on the exchange in YYYYMMDD format.
→ → 868	EventText	Y	String	Event string.
→ → 1145	EventTime	N	UTCTimestamp	Specific time of event.
→ 969	MinPriceIncrement	Y	Float	Minimum price increment (tick size).

Tag	Field Name	Req'd	Data Type	Comments
→ 1151	SecurityGroup	Y	String	The name of the group of securities to which this instrument belongs.
→ 562	MinTradeVol	N	Int	The minimum quantity allowed on an order.
→ 1306	PriceLimitType	N	Int	0 = Price
→ 1148	LowLimitPrice	N		
→ 1149	HighLimitPrice	N		
→ 15	Currency	Y	Currency	Currency for the instrument.
320	SecurityReqID	Y	String	Unique ID associated with this request.
322	SecurityResponseID	Y	String	Unique ID for this response. Typically a 13 character alphanumeric string.
560	SecurityRequestResult	Y	Int	The status of the request: 0 = Valid request 1 = Invalid or unsupported request 2 = No instruments found that match selection criteria 3 = Not authorized to retrieve instrument data 4 = Instrument data temporarily unavailable
< Standard Trailer >		Y		

Example 17: SecurityList [y] Containing Two Instruments (repeating groups color-coded)

```
8=FIXT.1.1 | 9=716 | 35=y | 34=2 | 49=SENDER_ID | 52=20251125
-18:40:02.016328755 | 56=TARGET_ID | 146=2 | 55=BFUT100-ERCOT-HB_NORTH
-251123-01 | 48=BFUT100-ERCOT-HB_NORTH-251123-01 | 22=8 | 460=14 | 167=EVENT |
231=1 | 864=3 | 865=5 | 866=20251118 | 868=StartDate | 865=7 | 866=20251122 |
868=LastTradeDate | 1145=20251122-02:00:00.000000000 | 865=7 | 866=20251123 |
868=ExpirationDate | 1145=20251123-07:00:00.000000000 | 969=0.25 |
1151=BFUT100-ERCOT-HB_NORTH | 555=0 | 562=1 | 1306=0 | 1148=0.00 | 1149=100.00
| 15=USD | 55=BFUT100-ERCOT-HB_TEST-251123-01 | 48=BFUT100-ERCOT-HB_TEST
-251123-01 | 22=8 | 460=14 | 167=EVENT | 231=1 | 864=3 | 865=5 | 866=20251118 |
868=StartDate | 865=7 | 866=20251123 | 868=LastTradeDate | 1145=20251123
-06:15:00.000000000 | 865=7 | 866=20251123 | 868=ExpirationDate |
1145=20251123-07:00:00.000000000 | 969=0.25 | 1151=BFUT100-ERCOT-HB_TEST |
555=0 | 562=1 | 1306=0 | 1148=0.00 | 1149=100.00 | 15=USD | 320=2007026312 |
322=1HPT7F2AA6404 | 560=0 | 10=007 |
```

Figure 13: Request from Security Reference Data



Sample Message Sequence 15: Security Request
Request details about a financial instrument or list of available instruments.

Client sends a [x] Security List request.
```
< 8=FIXT.1.1 | 9=114 | 35=x | 49=SENDER_ID | 56=EXI | 34=4 | 52=20260122
-19:09:59 | 320=SecurityRequest2 | 559=0 | 55=BFUT500-ERCOT-HB_NORTH-260122-22
| 10=028 |
```

Sample Message Sequence 16: Security List Response (repeating groups color-coded).
Response containing instrument specifications and trading parameters.
```
> 8=FIXT.1.1 | 9=476 | 35=y | 34=4 | 49=EXI | 52=20260122-19:09:59.238563901 |
56=TARGET_ID | 146=1 | 55=BFUT500-ERCOT-HB_NORTH-260113-22 | 48=BFUT500-ERCOT
-HB_NORTH-260113-22 | 22=8 | 460=14 | 167=EVENT | 231=1 | 864=3 | 865=5 |
866=20260109 | 868=StartDate | 865=7 | 866=20260114 | 868=LastTradeDate |
1145=20260114-02:00:00.000000000 | 865=7 | 866=20260114 | 868=ExpirationDate |
1145=20260114-04:00:00.000000000 | 969=0.25 | 1151=BFUT500-ERCOT-HB_NORTH |
555=0 | 562=1 | 1306=0 | 1148=0.00 | 1149=500.00 | 15=USD |
320=SecurityRequest2 | 322=7JSR1NPG6192 | 560=0 | 10=209 |
```

FIX Session Schedule

FIX Sessions run for a week at a time without sequence number reset. At 06:15 Sunday UTC there is a sequence number reset.

FIX Session START	Sunday 06:15:00 UTC
FIX Session END	Sunday 06:14:59 UTC

Rate Limiting

If too many messages are sent over the rate-limited time period, further messages will be rejected with a BusinessMessageReject (including cancel requests).

Example 16: BusinessMessageReject from EXI Due to Rate Limiting

```
8=FIXT.1.1 | 9=82 | 35=j | 34=116343 | 49=EXI | 52=20250331-19:49:00.158130086
| 56=TARGET_ID | 57=TARGET_SUB_ID | 45=116372 | 58=Message rate limit throttled
for session FIXT.1.1:SENER_ID->TARGET_ID | 372=D | 380=0 | 10=017 |
```

Appendix A: FIX Data Types

Type Name	Data Type	Description
Int	Signed integer. Zeros permitted.	Sequence of digits without commas or decimals and optional sign character (ASCII characters - and 0 - 9). The sign character utilizes one byte (i.e. positive int is 99999 while negative int is -99999). Note that int values may contain leading zeros (e.g. 00023 = 23).
Length	Strictly positive integer.	int field representing the length in bytes.
SeqNum	Strictly positive integer.	int field representing a message sequence number.
NumInGroup	Strictly positive integer.	int field representing the number of entries in a repeating group.
Float	Signed float, with optional decimal point.	The absence of the decimal point within the string will be interpreted as the float representation of an integer value. All float fields must accommodate up to fifteen significant digits. Float values may contain leading zeros (e.g. 00023.23 = 23.23) and may contain or omit trailing zeros after the decimal point (e.g. 23.0 = 23.0000 = 23 = 23.).
Qty	Positive float.	float field capable of storing either a whole number (no decimal places) of shares (securities denominated in whole units) or a decimal value containing decimal places for non-share quantity asset classes (securities denominated in fractional units).
Price	Signed float price.	float field representing a price with a varying number of decimal places. For certain asset classes prices may be negative values. For example, prices for options strategies can be negative under certain market conditions.
Amt	Signed float	float field typically representing a Price times a Qty.
Char	Case-sensitive, single alphanumeric character	Can include any alphanumeric character or punctuation except the delimiter (SOH). All char fields are case sensitive (i.e. m != M).
Boolean	'Y' = True/Yes 'N' = False/No	char field containing one of two possible values.
String	Case-sensitive, alphanumeric string.	Can include any character or punctuation except the delimiter. All String fields are case sensitive (i.e. morstatt != Morstatt).

Type Name	Data Type	Description
MultipleCharValue	Space-delimited sequence of single character values	string field containing one or more space-delimited values (e.g. | 18=G c |).
Currency	Three character string.	string field representing a currency type using ISO 4217 Currency code (3 character) values.

UTCTimestamp	Formatted UTC timestamp (date and time)	string field representing time/date combination represented in UTC in either:
		- YYYYMMDD-HH:MM:SS (whole seconds) or
		- YYYYMMDD-HH:MM:SS.sss* format
		Where:
		- SS can be 60 seconds in the rare case of UTC leap second, and
		- sss* represents fractions of seconds, either
		    - 3 digits to represent milliseconds,
		    - 6 digits to convey microseconds,
		    - 9 digits to convey nanoseconds, or
		    - 12 digits to convey picoseconds.

UTCTimeOnly	Formatted UTC timestamp (time only)	string field representing a time represented in UTC in either: - HH:MM:SS (whole seconds) or - HH:MM:SS.sss* format
UTCDateOnly	Formatted UTC timestamp (date only)	string field representing Date represented in UTC in YYYYMMDD format.
LocalMktDate	Date in the timezone local to the sender	string field representing Date represented in sender’s timezone in YYYYMMDD format.

Electron