CROSSTOWER

FIX5 Trading Gateway

Operational manual v1.1, 2020-01-20

Table of Contents

1. General

1.1. Introduction

The FIX5 Trading Gateway enables clients to submit orders and to receive real-time information on executed trades. The interface is a point-to-point service based on the technology and industry standards TCP/IP, FIXT and FIX. The session and application event models and messages are based on versions 1.1 and 5.0 (Service Pack 2) of the FIXT and FIX protocols respectively.

1.2. Purpose

The purpose of this document is to provide a technical description of the FIX Trading Gateway. This document outlines the service, its logic, and message types and fields involved.

1.3. Key

Usage	Meaning
Req	Required?
Y	Yes
N	No
C	Conditional

1.4. Document history

Issue	Date	Description
v1.0	2019-11-08	First edition
v1.1	2020-01-20	Added new messages InitMargin (UIM), SetMargin (USM), CloseMarginPosition (UCMP), MarginReject (UMR)

2. Service Description

2.1. Order Handling

2.1.1. Order Types

Clients may submit the order types outlined below via the New Order – Single message.

Order Type	Description
Market	An order that will execute at the price not worse than 10% from the best available price.
Limit	An order that will execute at or better than the specified price. The remainder, if any, is added to the order book or expired by terms of its TimeInForce.
Stop Market	A market order that remains inactive until the market reaches a specified stop price.
Stop Limit	A limit order that remains inactive until the market reaches a specified stop price.

2.1.2. Time in Force

Type	Description
Day	An order that will expire at the end of the day. Day (session) orders automatically expire at UTC 00:00.
Immediate or Cancel (IOC)	An order that will be executed on receipt and the remainder, if any, immediately expired.
Fill or Kill (FOK)	An order that will be fully executed on receipt or immediately expired.
Good Till Date (GTD)	An order that will expire at the specified date and time.
Good Till Cancel	An order that remains in effect until it is either executed or cancelled.

2.2. Order Management

2.2.1. Cancellation

The remainder of a live order may be cancelled via the Order Cancel Request message. The server will respond with an Execution Report or Order Cancel Reject to confirm or reject the cancellation request respectively.

In a scenario where the Order Cancel Request message is submitted under a different SenderCompID (49) than the SenderCompID (49) of the original order, the Execution Report will be sent to the cancelling SenderCompID. If the user cancelling the order does not have permissions to cancel orders, the cancel request will be rejected.

The client should identify the order being cancelled by either its OrigClOrdID (41) or OrderID (37).

If an Order Cancel Request contains values for both OrigClOrdID (41) and OrderID (37), the server will only process the OrderID (37).

If an order submitted by a different user (PartyID (448)) is being cancelled, the Order Cancel Request should include its OrderID (37).

2.2.2. "Post Only" Mode

Orders for which the client has placed 'participateDoNotInitiate' (6) in the ExecInst (18) field of New Order - Single (D) will be in the order book only if they do not trigger a single transaction. Otherwise, two ExecutionReport messages will be returned to the client:

- ExecType = New (0)
- ExecType = Expired (C)

2.2.3. Amending an order

The following attributes of a live order may be amended via the Order Cancel/Replace Request message:

- Order quantity
- Price
- Stop price

The server will respond with an Execution Report or Order Cancellation Rejection to confirm or reject the amendment request respectively.

In a scenario where the Order Cancel/Replace Request message is submitted by a different SenderCompID (49) than the SenderCompID (49) who submitted the original order, the Execution Report or Order Cancel Reject will be sent to the modifying SenderCompID (49). If the user amending the order does not have permissions to modify orders on behalf of the firm, the amend request will be rejected.

The client should identify the order being amended by either its OrigClOrdID (41) or OrderID (37). If an Order Cancel/Replace Request contains values for both OrigClOrdID (41) and OrderID (37), the server will only process the OrderID (37).

If an order submitted under a different SenderCompID (49) is being amended, the Order Cancel/Replace Request should include its OrderID (37). If the amendment is successful, the order will be treated as one submitted under the SenderCompID (49) that sent the Order Cancel/Replace Request.

The Stop price of a Stop Market / Stop Limit order cannot be amended once the order has been injected into the order book.

If an order receives one or more fills while an amendment request is in flight, the system will not reject the incoming amendment request if its quantity after amending will be greater than the filled quantity. Otherwise the amendment request will be rejected.

An order will lose time priority in the order book if any of its parameters are amended.

2.2.4. Order Execution Report

The ExecutionReport message is used to inform customers of up-to-date information about order status immediately after any occured event. This message contains the fields:

- OrdStatus - this field is used to convey the current state of an order
- ExecType - the type of occurred event is indicated.

The list of order statuses is given in the table below.

OrdStatus:

OrdStatus	Description
New (0)	Order has been registered
PartiallyFilled (1)	Order has been partially filled
Filled (2)	Order has been fully filled
Cancelled (4)	Order has been cancelled
Rejected (8)	Order has been rejected
Expired (C)	Order has been expired or cannot be placed into the order book (post-only, FOK, IOC)
Suspended (9)	Order is awaiting activation

The list of event types is given in the table below.

ExecType:

ExecType	Description
New (0)	Indicates that a new order has been accepted. This message will also be sent when a suspended order is injected and added to the order book.
Canceled (4)	Indicates that an order cancel request has been accepted and successfully processed. This message will also be sent unsolicited if the order was cancelled by the system.
Replaced (5)	Indicates that an order cancel/replace request has been accepted and successfully processed.
Rejected (8)	Indicates that an order has been rejected.
Suspended (9)	Indicates that an order has been parked by the system without adding it to the order book.
Expired (C)	Indicates that an order has expired in terms of its time qualifier; or it is IOC, FOK; or ExecInst (18) contains ParticipateDoNotInitiate (6)
Trade (F)	Indicates that an order has been partially or fully filled. The execution details (e.g. price and quantity) are specified.
OrderStatus (I)	This value is used if the ExecutionReport is sent in response to OrderStatusRequest (H) or OrderMassStatusRequest (AF).

2.2.5. Order and execution identifiers

Client Order IDs

Clients should comply with the FIX protocol and ensure unique ClOrdIDs across all messages (e.g. New Order – Single, Order Cancel Request, etc.) sent under a particular SenderCompID (49). Given that the server supports GTD orders, clients should also ensure that their ClOrdIDs are unique across trading days (e.g. embed the date within the ClOrdID).

Clients must, in terms of the FIX protocol, specify the ClOrdID (11) when submitting an Order Cancel Request or Order Cancel/Replace Request.

Adding SenderCompID to ClOrdID is recommended.

Example:

SenderCompID=login_00119485 ⇒ ClOrdID=001194853fec1f8430da0ec0247ebc6a

Order IDs

The server will use the OrderID (37) field of the Execution Report to keep track of orders within the matching system. Order IDs will be unique across trading days.

Clients have the option of specifying the OrderID (37) when submitting an Order Cancel Request or Order Cancel/Replace Request.

Execution IDs

The server will use the ExecID (17) field to affix a unique identifier for each Execution Report. ExecIDs will be unique across trading days.

Order ID tag length

The system will accept a maximum length of 32 characters. If the ID is longer than 32 characters then it will be rejected. This is valid for the following:

NewOrderSingle – ClOrdID (11)

OrderCancelRequest – OrigClOrdID (41)

OrderCancel/ReplaceRequest – OrigClOrdID (41)

RequestForPositions (AN) - PosReqID (710)

OrderStatusRequest (H) - OrdStatisReqID (790)

OrderMassStatusRequest (AF) - MassStatusReqID (584)

ClOrdID may contain delimiters as desired by the client.

Example:

0119485 / 3fec1f8430da0ec0247ebc6a

3. Connectivity

3.1. CompIDs

The CompID of each client must be registered before FIX communications can begin. A single client may have multiple connections to the server (i.e. multiple FIX sessions, each with its own CompID).

The CompID of the server is ME. The messages sent to the server should contain the CompID assigned to the client in the field SenderCompID (49) and ME in the field TargetCompID (56). The messages sent from the server to the client will contain ME in the field SenderCompID (49) and the CompID assigned to the client in the field TargetCompID (56).

3.2. Passwords

Each new CompID will be assigned a username and a password on registration.

3.3. Message Rate Throttling

A scheme for throttling message traffic is implemented, where each CompID is only permitted to submit up to a specified number of messages per second.

Every message which exceeds the maximum rate of a CompID will be rejected via a Business Message Reject (with BusinessRejectReason (380) of ThrottleLimitExceeded (8)).

Please note that client Heartbeat messages, reject messages and any other client-initiated administrative messages are not counted towards the throttling limits.

3.4. Mass Cancellation On Disconnect

The client may request to cancel all orders submitted through the FIX session. In the Logon (A) message, the value of the CancelOnDisconnect field (10001) defaults to N.

If it is set to Y, then as soon as the gate detects that the client has disconnected (intentionally or due to a connection loss), it will send requests to cancel working orders. If the connection is lost, the gate can detect a disconnection with a delay of several seconds when Heartbeat messages do not come from the client.

If the CancelOnDisconnect (10001) flag was not set in the Logon (A) message, then cancellation on disconnect can be activated for any new order. To do this, in the New Order - Single (D) message in the ExecInst field (18) add the value "o" (CancelOnConnectionLoss).

4. FIX connections and sessions

4.1. Establishing a FIX connection

FIX connections and sessions between the client and server are maintained as specified in the FIX protocol.

Each client will use the assigned IP address and port to establish a TCP/IP session with the server. The client will initiate a FIX session by sending the Logon (A) message. The client will identify itself using the SenderCompID (49) field. At each connection MsgSeqNum (34) is reset to 1, the client should set ResetSeqNumFlag (141) to "Y". The server will validate the CompID, password and IP address of the client. Once the client is authenticated, the server will respond with a Logon message.

The client must wait for the server's Logon response before sending additional messages. If the client sends messages prior to sending the Logon message or prior to receiving the Logon response, the server will break the TCP/IP connection with the client without sending any message.

If a logon attempt fails (due to an invalid SenderCompID, invalid TargetCompID, invalid IP address, invalid password or not having the appropriate privileges to login to the gateway, or if the user sends a Logon message with duplicated tags), the server will break the TCP/IP connection with the client without sending a Logout or Reject message.

If during a logon of a SenderCompID, the server receives a second connection attempt while a valid FIX session is already underway for that same SenderCompID, the server will break the TCP/IP connection with the second connection without sending a Logout or Reject message.

4.2. Maintaining a FIX session

4.2.1. Message sequence numbers

As outlined in the FIX protocol, the client and server will each maintain a separate and independent set of incoming and outgoing message sequence numbers. Sequence numbers should be initialized to 1 (one) at the start of the FIX session and be incremented throughout the session.

If any message sent by the client contains a sequence number that is less than what is expected and the PossDupFlag (43) is not set to "Y", the server will send a Logout message and terminate the FIX connection. The Logout will contain the next expected sequence number as well as the received sequence number in the Text (58) field.

4.2.2. Heartbeats

The client and server will use the Heartbeat message to exercise the communication line during periods of inactivity and to verify that the interfaces at each end are available. The heartbeat interval will be the HeartBtInt (108) specified in the client's Logon message.

The server will send a Heartbeat anytime it has not transmitted a message for the heartbeat interval. The client is expected to employ the same logic.

As a safety mechanism, the system will not allow the user to login if the HeartBtInt is set to 0.

If the server detects inactivity for a period longer than the heartbeat interval plus a reasonable transmission time, it will send a Test Request message to force a Heartbeat from the client. If a response to the Test Request is not received by a reasonable transmission time, the server will send a Logout and break the TCP/IP connection with the client. The client is expected to employ similar logic if inactivity is detected on the part of the server.

4.2.3. Increasing expected sequence number

The client or server may use the SequenceReset message in Gap Fill mode if it wishes to increase the expected incoming sequence number of the other party.

The client or server may also use the SequenceReset message in SequenceReset mode if it wishes to increase the expected incoming sequence number of the other party. The SequenceReset mode should only be used to recover from an emergency situation. It should not be relied upon as a regular practice.

4.2.4. Terminating a FIX connection

The client will terminate a connection by sending the Logout message. The server will respond with a Logout to confirm the termination. The client will then break the TCP/IP connection with the server.

All open TCP/IP connections will be terminated by the server when it shuts down (a Logout will be sent). Under exceptional circumstances the server may initiate the termination of a connection during the trading day by sending the Logout message.

4.2.5. Re-establishing a FIX session

If a FIX connection is terminated during the trading day it may be re-established via an exchange of Logon messages.

5. Recovery

5.1. Resend requests

Sending and processing ResendRequest is not supported. In response to the incoming ResendRequest the Reject message is sent.

5.2. Possible duplicates

The server handles possible duplicates according to the FIX protocol. The client and server will use the PossDupFlag (43) field to indicate that a message may have been previously transmitted with the same MsgSeqNum (34).

6. Messages

This section provides details on the header and trailer, administrative and application messages utilized by the server. Client-initiated messages not included in this section are rejected by the server via a Reject or Business Message Reject.

6.1. Supported message types

Administrative messages

All administrative messages are initiated by either the client or the server.

Message	MsgType	Usage
Logon	A	Allows the client and server to establish a FIX session.
Logout	5	Allows the client and server to terminate a FIX session.
Heartbeat	0	Allows the client and server to exercise the communication line during periods of inactivity and verify that the interfaces at each end are available.
TestRequest	1	Allows the client or server to request a response from the other party if inactivity is detected.
ResenRequest	2	Allows for the recovery of messages lost during a malfunction of the communications layers.
Reject	3	Used to reject a message that does not comply with FIXT.
SequenceReset	4	Allows the client or server to increase the expected incoming sequence number of the other party.

Application messages: order handling

Client-initiated:

Message	MsgType	Usage
New Order – Single	D	Allows the client to submit a new order.
Order Cancel Request	F	Allows the client to cancel a live order.
Order Cancel/Replace Request	G	Allows the client to cancel/replace a live order.
OrderStatusRequest	H	Allows the client to request order status for one of his orders.
OrderMassStatusRequest	AF	Allows to request the status for all orders of the client.
RequestForPositions	AN	Allows the client to request a Position Report about balance.

Server-initiated:

Message	MsgType	Usage
Execution Report	8	Indicates one of the following: (1) Order accepted (2) Order rejected (3) Order executed (4) Order expired (5) Order cancelled (6) Order cancel/replaced (7) Trade (8) Order status
Order Cancel Reject	9	Indicates that an order cancel request or order cancel/replace request has been rejected.
PositionReport	AP	Comes in response when requesting a balance via RequestForPositions.

Application messages: other

Server-initiated:

Message	MsgType	Usage
Business Message Reject	j	Indicates that an application message could not be processed

6.2. Message Format

A standard message includes message header, message body, and message trailer. Fields inside the header, body, and trailer consist of <tag>=<value> pairs, and, unless otherwise noted, these fields can be defined in any order.

Standard Header

The message header fields are given in the table below. The first three fields in the header must be BeginString (8), BodyLength (9) and MsgType (35). Other fields can be defined in any order.

Tag	Field Name	Req	Description
8	BeginString	Y	FIXT.1.1
9	BodyLength	Y	Number of characters after this field up to and including the delimiter immediately preceding the CheckSum.
35	MsgType	Y	Message type.
49	SenderCompID	Y	CompID of the party sending the message.
56	TargetCompID	Y	CompID of the party the message is sent to.
34	MsgSeqNum	Y	Sequence number of the message.
43	PossDupFlag	N	Whether the message was previously transmitted under the same MsgSeqNum (34). Absence of this field is interpreted as Original Transmission (N). ('Y' = Possible Duplicate).
52	SendingTime	Y	Time the message was transmitted.
122	OrigSendingTime	C	Time the message was originally transmitted. If the original time is not available, this should be the same value as SendingTime (52). Required if PossDupFlag (43) is Possible Duplicate ('Y').

Message Body

The message body fields can be defined in any order, except for the NoXXXs tags (where XXX is a

field being counted). These tags start repeating data groups. The fields inside these groups must be defined in the order, specified in this document.

Standard Trailer

The message trailer described in the table below.

Tag	Field Name	Req	Description
10	CheckSum	Y	CheckSum

6.3. Administrative messages

Logon

The Logon message authenticates the client establishing a connection to the FIX Server. In addition to the standard fields, clients will need to populate two additional fields – Username (tag 553) and Password (tag 554).

MsgType (35): A = Logon

Tag	Field Name	Req	Description
Standard Header			
Message Body			
98	EncryptMethod	Y	Method of encryption, 0 = None
108	HeartBtInt	Y	Indicates the heartbeat interval in seconds, no less than 1.
141	ResetSeqNumFlag	Y	Indicates whether the client and server should reset. Should always be set to the value Y = Reset Sequence Numbers, as N is not supported.
383	MaxMessageSize	N	The maximum value that can be presented in the BodyLength field. If a message is received from the client, where BodyLength is greater than MaxMessageSize, the connection will be closed. This field is sent by the server; if sent by the client, it will be ignored. π
553	Username	C	Username assigned to the CompID. Required if the message is sent by the client. Not sent by the server.
554	Password	C	Password assigned to the CompID. Required if the message is sent by the client. Not sent by the server.
1137	DefaultApplVerID	Y	Default version of FIX messages used in this session and this value will be validated by the server, 9 = FIX50SP2
10001	CancelOnDisconnect	N	This flag is set to "Y" to enable cancel on disconnect functionality for all orders placed through this session that allows users to have this orders automatically canceled upon an unintentional loss of session connectivity, 'Y'/'N'. It set to 'N' by default. Parameter could be used by the client and isn't applicable to the server side.
Standard Trailer			

Logout

The Logout message initiates or confirms the termination of a FIX session. It can optionally contain the Text field (tag 58) which is populated with the reason for the Logout.

MsgType (35): 5 = Logout

Tag	Field Name	Req	Description
Standard Header			
Message Body			
1409	SessionStatus	C	Status of the FIX session. Required if the message is generated by the server. 4 = SessionLogoutComplete 5 = InvalidUsernameOrPassword 6 = AccountLocked 9 = ReceivedMsgSeqNumTooLow 10 = ReceivedNextExpectedMsgSeqNumTooHigh
58	Text	N	The field will contain the next expected sequence number as well as the received sequence number if the server terminated the connection after receiving a sequence number that was less than what was expected. In other cases the field will contain the reason for logout.
Standard Trailer			

Heartbeat

The Heartbeat message monitors the status of the communication link and identifies when the last of a string of messages was not received. A Heartbeat is sent every HeartBtInt seconds, as defined in the Logon message, as well as in response to a TestRequest message.

Inclusion of the optional TestReqID field (tag 112) indicates that this Heartbeat is a response to a TestRequest message.

MsgType (35): 0 = Heartbeat

Tag	Field Name	Req	Description
Standard Header			
Message Body			
112	TestReqID	C	Required if the heartbeat is a response to a Test Request. The value in this field should echo the TestReqID (112) received in the Test Request.
Standard Trailer			

TestRequest

The TestRequest message forces a heartbeat from the opposing application. It can be initiated both from the client and from the server.

MsgType (35): 1 = TestRequest

Tag	Field Name	Req	Description
Standard Header			
Message Body			
112	TestReqID	Y	Identifier for the request.
Standard Trailer			

ResendRequest

This function is used if a sequence number gap is detected, if the receiving application lost a message, or as a function of the initialization process.

MsgType (35): 2 = ResendRequest

Tag	Field Name	Req	Description
Standard Header			
Message Body			
7	BeginSeqNo	Y	Sequence number of first message in range.
16	EndSeqNo	Y	Sequence number of last message in range.
Standard Trailer			

Reject

A Reject message is issued when a message is received but cannot be properly processed due to a

session-level rule violation. Whenever possible, it is strongly recommended that the cause of the failure be described in the Text (tag 58) field.

MsgType (35): 3 = Reject

Tag	Field Name	Req	Description
Standard Header			
Message Body			
45	RefSeqNum	Y	MsgSeqNum (34) of the rejected message.
372	RefMsgType	N	MsgType (35) of the rejected message.
371	RefTagID	N	If a message is rejected due to an issue with a particular field, its tag number will be indicated.

Tag	Field Name	Req	Description
373	SessionRejectReason	N	Code specifying the reason for the reject. 0 = InvalidTagNumber 1 = RequiredTagMissing 2 = TagNotDefinedForThisMessageType 3 = UndefinedTag 4 = TagSpecifiedWithoutAValue 5 = ValueIsIncorrect 6 = IncorrectDataFormatForValue 8 = SignatureProblem 10 = SendingTimeAccuracyProblem 11 = InvalidMsgType 13 = TagAppearsMoreThanOnce 14 = TagSpecifiedOutOfRequiredOrder 15 = RepeatingGroupFieldsOutOfOrder 16 = IncorrectNumInGroupCountForRepeatingGroup 17 = Non 18 = Invalid 99 = Other
58	Text	N	Text specifying the reason for the rejection.
Standard Trailer			

SequenceReset

A Sequence Reset message instructs the receiving party to set the next expected message sequence number to a new value, which is contained in the FIX message in the NewSeqNo (tag 36) field.

MsgType (35): 4 = SequenceReset

Tag	Field Name	Req	Description
Standard Header			
Message Body			
36	NewSeqNo	Y	Sequence number of the next message to be transmitted.
123	GapFillFlag	N	Mode in which the message is being used. Absence of this field is interpreted as Sequence Reset ('N' by default). ('Y' = Gap Fill).
Standard Trailer			

6.4. Application messages: order handling

New Order - Single

The New Order message is used by clients wishing to electronically submit orders to the exchange for execution.

MsgType (35): D = New Order - Single

Tag	Field Name	Req	Description
Standard Header			
Message Body			
11	ClOrdID	Y	Client specified identifier of the order.
1	Account	Y	An account number.
453	NoPartyIDs	Y	Number of party identifiers. Will be '1'.
>448	PartyID	Y	Name of a user.
>447	PartyIDSource	Y	'D' = Proprietary
>452	PartyRole	Y	Role of the specified PartyID. '3' = Client ID
55	Symbol	Y	Symbol
40	OrdType	Y	Type of the order. 1 = Market 2 = Limit 3 = Stop Market 4 = Stop Limit

Tag	Field Name	Req	Description
59	TimeInForce	Y	Time qualifier of the order. 0 = Day 1 = GTC 3 = IOC 4 = FOK 6 = GTD
126	ExpireTime	C	Date and time the order expires. For GTD only.
54	Side	Y	Side of the order. 1 = Buy 2 = Sell
38	OrderQty	Y	Total order quantity in lots.
44	Price	C	Limit price. Required if OrderType (40) is Limit (2) or Stop Limit (4).
99	StopPx	C	Stop price. Required if OrderType (40) is Stop Market (3) or Stop Limit (4).
18	ExecInst	N	6 = ParticipateDoNotInitiate o = CancelOnConnectionLoss
544	CashMargin	N	1 = Cash (default) 2 = MarginOpen
60	TransactTime	N	Ignored.
168	EffectiveTime	N	Time required for processing of an order, the order will be declined when processing run out of time. There is no default value.
Standard Trailer			

Order Cancel Request

The Order Cancel Request message requests the cancellation of all of the remaining quantity of an existing order.

Each Cancel Request is assigned a unique ID in the ClOrdID (tag 11) field, and is subsequently treated as a separate entity. If rejected, the ClOrdID (tag 11) of the cancel request will be sent in the Order Cancel Reject (9) message ClOrdID (tag 11) field, whereas the ClOrdID (tag 11) of the actual order in the OrigClOrdID (tag 41) field. The ClOrdID (tag 11) assigned to the cancel request must be unique amongst the ClOrdID (tag 11) fields assigned to regular orders and replacement orders.

MsgType (35): F = Order Cancel Request

Tag	Field Name	Req	Description
Standard Header			
Message Body			
11	ClOrdID	Y	Client specified identifier of the cancel request.
41	OrigClOrdID	C	ClOrdID (11) of the order being cancelled. Required if OrderID (37) is not specified.
37	OrderID	C	Server specified identifier of the order being cancelled. Required if OrigClOrdID (41) is not specified. In the case when both parameters were specified, OrderID has priority over OrigClOrdID (41).
55	Symbol	N	Symbol of the order.
54	Side	N	Side of the order. 1 = Buy 2 = Sell
1	Account	Y	An account number.
453	NoPartyIDs	Y	Number of party identifiers. Will be '1'.
>448	PartyID	Y	Name of a user.
>447	PartyIDSource	Y	'D' = Proprietary
>452	PartyRole	Y	Role of the specified PartyID. '3' = Client ID
60	TransactTime	N	Ignored.
Standard Trailer			

Order Cancel/Replace Request

The Order Cancel/Replace Request message can be used to change the details of an existing order.

When processing a Cancel/Replace the existing order is cancelled and a new order is placed on the order book. This is executed as a single atomic transaction. However, the newly-placed order loses its order book time priority. Hence, if the intention is to increase the quantity of an existing order, it is recommended that clients send a New Order message for the increased quantity rather than a Cancel/Replace.

Clients should not use this message to cancel the remaining quantity of an outstanding order. Instead, they should send an Order Cancel Request message.

The Cancel/Replace Request will only be accepted if the order can successfully be pulled back from

the exchange without executing. Requests which cannot be processed will be rejected using the Cancel Reject message.

Each Cancel/Replace Request is assigned a unique ID in the ClOrdID (tag 11) field, and is subsequently treated as a separate entity. If rejected, the ClOrdID (tag 11) of the cancel request will be sent in the Order Cancel Reject (9) message ClOrdID (tag 11) field, whereas the ClOrdID (tag 11) of the actual order in the OrigClOrdID (tag 41) field. The ClOrdID (tag 11) assigned to the Order Cancel/Replace Request must be unique amongst the ClOrdID (tag 11) fields assigned to regular orders and replacement orders.

MsgType (35): G = Order Cancel/Replace Request

Tag	Field Name	Req	Description
Standard Header			
Message Body			
11	ClOrdID	Y	Client specified identifier of the cancel/replace request.
41	OrigClOrdID	Y	ClOrdID (11) of the order being amended.
37	OrderID	N	Unsupported.
1	Account	Y	Account associated with the order that the client is requesting to cancel/replace.
453	NoPartyIDs	Y	Number of party identifiers. Will be '1'.
>448	PartyID	Y	Username.
>447	PartyIDSource	Y	'D' = Proprietary
>452	PartyRole	Y	Role of the specified PartyID. '3' = Client ID
55	Symbol	Y	Symbol of the order.
40	OrdType	N	Ignored.
54	Side	Y	Side of the order. 1 = Buy 2 = Sell
38	OrderQty	Y	Total order quantity in lots.
44	Price	C	Limit price. Required if OrderType (40) is Limit (2) or Stop Limit (4).
99	StopPx	C	Stop price. Required if OrderType (40) is Stop Market (3) or Stop Limit (4).
126	ExpireTime	C	Date and time the order expires. For GTD only. Required only if TimeInForce (59) is GTD (6).
60	TransactTime	N	Ignored.
168	EffectiveTime	N	Time required for processing of an order, the order will be declined when processing run out of time. There is no default value.
Standard Trailer			

Order Cancel Reject

MsgType (35): 9 = Order Cancel Reject

Tag	Field Name	Req	Description
Standard Header			
Message Body			
11	ClOrdID	Y	ClOrdID (11) that was submitted with the order cancel or cancel/replace request being rejected.
41	OrigClOrdID	Y	Value of the parameter is equal to the value of OrigClOrdID (41) of the cancel request preceding.
37	OrderID	N	Unsupported.
39	OrdStatus	Y	Status of the order. The current implementation of this message always returns '8' = Rejected.
434	CxlRejResponseTo	Y	Type of request being rejected. 1 = Order Cancel Request 2 = Order Cancel/Replace Request
102	ClxRejReason	Y	Code specifying the reason for the rejection. 0 = TooLateToCancel 1 = UnknownOrder 6 = DuplicateClOrdID 99 = Other
58	Text	N	Text specifying the reason for the rejection.
Standard Trailer			

Execution Report

The Execution Report message is used to:

- confirm the receipt of an order
- confirm changes to an existing order (i.e. accept cancel and replace requests)
- relay order status information
- relay fill information on working orders
- reject orders

Each execution report contains two fields which are used to communicate both the current state of

the order as understood by the broker - OrdStatus (tag 39) - and the purpose of the message - ExecType (tag 150).

MsgType (35): 8 = Execution Report

Tag	Field Name	Req	Description
Standard Header			
Message Body			
11	CIOrdID	Y	Client specified identifier of the order.
41	OrigClOrdID	C	Will be filled with the actual original client order id of the order regardless of whether or not OrigClOrdID was specified (valid or invalid value) in the order cancel or cancel/replace request.
37	OrderID	Y	Server specified identifier of the order.
17	ExecID	Y	Unique identifier of execution message.
790	OrdStatusReqID	C	Required if this is is sent in reply to an OrderStatusRequest message.
584	MassStatusReqID	C	Required if this message is sent in reply to an OrderMassStatusRequest message.
911	TotNumReports	C	Required if this message is sent in reply to an OrderMassStatusRequest message.
912	LastRptRequested	C	Required if this message is sent in reply to an OrderMassStatusRequest message.
150	ExecType	Y	Reason the execution report was generated. 0 = New 4 = Canceled 5 = Replaced 8 = Rejected 9 = Suspended C = Expired F = Trade I = OrderStatus

Tag	Field Name	Req	Description
39	OrdStatus	Y	Current status of the order. 0 = New 1 = Partially Filled 2 = Filled 4 = Canceled 8 = Rejected C = Expired 9 = Suspended
103	OrdRejReason	C	Code specifying the reason for the reject. Required if ExecType (150) is '8' = Rejected or 'C' = Expired.
58	Text	N	Text specifying the reason for the rejection, cancellation or expiration.
151	LeavesQty	Y	Quantity available for further execution. Will be '0' if OrdStatus '39' is Filled '2', Cancelled '4', Rejected '8' or Expired 'C'.
14	CumQty	Y	Total cumulative quantity filled.
55	Symbol	C	Could be absent in the case where ExecType (150) is '8' = Rejected, i.e. if there is no order found to the OrderStatusRequest (H) request.
1	Account	Y	An account number.
453	NoPartyIDs	Y	Number of party identifiers. Will be '1'.
>448	PartyID	Y	Name of user.
>447	PartyIDSource	Y	'D' = Proprietary
>452	PartyRole	Y	Role of the specified PartyID. '3' = Client ID

Tag	Field Name	Req	Description
40	OrdType	C	Value submitted with the order. 1 = Market 2 = Limit 3 = Stop Market 4 = Stop Limit It could be absent if ExecType (150) is '8' = Rejected.
59	TimeInForce	N	Time qualifier of the order. 0 = Day 1 = GTC 3 = IOC 4 = FOK 6 = GTD It could be absent if ExecType (150) is '8' = Rejected.
126	ExpireTime	C	Value submitted with the order. It is required TimeInForce (59) is '6' = GTD
54	Side	C	Value submitted with the order. It could be absent if ExecType (150) is '8' = Rejected.
38	OrderQty	Y	Value submitted with the order in lots.
44	Price	C	Value submitted with the order. Required only if OrdType (40) is Limit (2) or Stop Limit (4).
99	StopPx	C	Value submitted with the order. Required only if OrdType (40) is '3' = Stop Market or '4' = Stop Limit.
18	ExecInst	N	6 = ParticipateDoNotInitiate o = CancelOnConnectionLoss
60	TransactTime	N	Time the transaction represented by the Execution Report occurred. It could be absent if ExecType (150) is '8' = Rejected.

Tag	Field Name	Req	Description
851	LastLiquidityInd	C	Whether the order added or removed liquidity. 1 = Added Liquidity 2 = Removed Liquidity It' required for ExecType (150) is 'F' = Trade.
880	TrdMatchID	C	The unique ID of the trade. It' required for ExecType (150) is 'F' = Trade.
6	AvgPx	C	Calculated average price of all fills on this order.
31	LastPx	C	Price of this fill. It' required for ExecType (150) is 'F' = Trade.
32	LastQty	C	Quantity executed in this fill. It's required for ExecType (150) is 'F' = Trade.
136	NoMiscFees	C	Required for ExecType (150) is 'F' = Trade.
>137	MiscFeeAmt	Y	Fee value.
>138	MiscFeeCurr	Y	Fee currency.
>139	MistFeeType	Y	Fee type. It will be '4' = ExchangeFees.
544	CashMargin	N	1 = Cash (default) 2 = MarginOpen
2618	PositionID	C	Unique position identifier. Required if `CashMargin` = 2
Standard Trailer			

OrderStatusRequest

The OrderStatusRequest message is used by clients to request order status for one of their orders. Server will respond with an Execution Report message with ExecType (150) = I - Order Status.

In case of any errors for example if the Order is not found, server will respond with an ExecutionReport, where ExecType (150) = I - OrderStatus, OrdStatus (39) = 8 - Rejected, OrdRejReason (103) and Text (58) will contain the error code and its description.

MsgType (35): H = OrderStatusRequest

Tag	Field Name	Req	Description
Standard Header			
Message Body			
790	OrdStatisReqID	Y	Client specified identifier of the status request.
11	ClOrdID	C	ClOrdID (11) of the order that status is being requested. Required if OrderID (37) is not specified.
37	OrderID	C	Server specified identifier of the order that status is being requested. Required if ClOrdID (11) is not specified. In the case when both parameters were specified, OrderID has priority over ClOrdID (11).
453	NoPartyIDs	C	Number of party identifiers. Will be '1'.
>448	PartyID	C	Name of a user. Required if ClOrdID (11) was specified.
>447	PartyIDSource	C	'D' = Proprietary
>452	PartyRole	C	Role of the specified PartyID. '3' = Client ID.
54	Side	N	Side of the order. 1 = Buy 2 = Sell
55	Symbol	N	Symbol of the order.
Standard Trailer			

OrderMassStatusRequest

The OrderMassStatusRequest message requests the status for all orders of the client.

A mass status request is assigned a ClOrdID (tag 11) and is treated as a separate entity.

Execution Reports with ExecType (150) = I - Order Status are returned for all orders matching the criteria provided on the request (i.e. for the Client). Such an Execution Report will have the MassStatusReqID (tag 584) field filled in, and the very last one will have the LastRptRequested (912) flag populated.

MsgType (35): AF = OrderMassStatusRequest

Tag	Field Name	Req	Description
Standard Header			
Message Body			
584	MassStatusReqID	Y	Value assigned by issuer of Mass Status Request to uniquely identify the request.
585	MassStatusReqType	Y	'8' = StatusForOrdersForAPartyID. All active orders information by user will returned.
453	NoPartyIDs	Y	Number of party identifiers. Will be '1'.
>448	PartyID	Y	Name of a user.
>447	PartyIDSource	Y	'D' = Proprietary.
>452	PartyRole	Y	Role of the specified PartyID. '3' = Client ID.
Standard Trailer			

RequestForPositions

The value of the PosReqType (tag 724) field will be either 101 = Balance (custom, non-standard, does not allow subscription, only snapshot) or 102 = MarginPositions (custom, non-standard, allows subscription). PositionReport (AP) is sent as a reply to the Balance request type. MarginPositionReport (UMPR) is sent as a reply to the MarginPositions request type.

MsgType (35): AN = RequestForPositions

Tag	Field Name	Req	Description
Standard Header			
Message Body			
710	PosReqID	Y	Unique identifier for the position maintenance request as assigned by the submitter.
724	PosReqType	Y	101 = Balance 102 = MarginPositions
263	SubscriptionRequestType	N	0 = Snapshot (default) 1 = SnapshotAndUpdates 2 = DisablePreviousSnapshot" If PosReqType (724) is '101' = Balance, the parameter will be ignored.
1	Account	Y	An account number.
453	NoPartyIDs	Y	Number of party identifiers. Will be '1'.
>448	PartyID	Y	Name of a user.
>447	PartyIDSource	Y	'D' = Proprietary.
>452	PartyRole	Y	Role of the specified PartyID. '3' = Client ID.
15	Currency	C	Required if PosReqType (724) is '101' = Balance. Ignored if PosReqType (724) is '102' = MarginPositions.
55	Symbol	C	Required if PosReqType (724) is '102' = MarginPositions. Ignored if PosReqType (724) is '101' = Balance.
715	ClearingBusinessDate	N	The current date. Ignored
60	TransactTime	N	Ignored.
Standard Trailer			

PositionReport

Comes in response to requesting a balance via RequestForPositions. The currency field will contain the currency for which the balance was requested. The Position repeating group will contain 2 elements (NoPosAmt = 2):

- PosAmtType = CASH
- PosAmtType = CRES

In case of a balance request error, a PositionReport is sent with NoPosAmt = 0.

MsgType (35): AP = PositionReport

Tag	Field Name	Req	Description
Standard Header			
Message Body			
721	PosMaintRptID	Y	Unique report ID
710	PosReqID	Y	Unique identifier for the position maintenance request as assigned by the submitter. Value is taken from the request.
724	PosReqType	Y	0 = Positions 101 = Balance '101' = Balance — custom, non-standard, does not allow subscription, only snapshot. Value is taken from the request.
325	UnsolicitedIndicator	Y	Always is 'false'.
728	PosReqResult	Y	0 = ValidRequest 1 = InvalidOrUnsupportedRequest 2 = NoPositionsFoundThatMatchCriteria 3 = NotAuthorizedToRequestPositions 4 = RequestForPositionNotSupported 99 = Other
58	Text	N	Text specifying the reason for the rejection.
715	ClearingBusinessDate	Y	The current date.
1	Account	Y	Client reference information.
453	NoPartyIDs	Y	Number of party identifiers. Will be '1'.
>448	PartyID	Y	Name of a user.
>447	PartyIDSource	Y	'D' = Proprietary
>452	PartyRole	Y	Role of the specified PartyID. '3' = Client ID
15	Currency	C	Currency for which a balance was requested. Required if PosReqType (724) is '101' = Balance.

Tag	Field Name	Req	Description
55	Symbol	C	Instrument associated with position, which is the case when PosReqType (724) is '0' = Positions. It will be ignored if PosReqType (724) is '101' = Balance.
702	NoPositions	C	Number of position entries (0). Required if PosReqType (724) is '0' = Positions.
703	PosType	Y	Type of quantity being returned.
704	LongQty	C	Long quantity.
705	ShortQty	C	Short quantity.
753	NoPosAmt	Y	Number of position amount entries (0 or 2). It will be '0' if an error occurred. '2' signifies a successful balance return.
>707	PosAmtType	Y	Type of position amount. One of two values: - CASH (Cash Amount) is the total amount of this currency on the user’s account. - CRES (Cash Residual Amount) is the amount not blocked by the position, that is available for opening positions.
>708	PosAmt	Y	Position amount
Standard Trailer			

MarginPositionReport

MsgType (35): UMPR = MarginPositionReport

Tag	Field Name	Req	Description
Standard Header			
Message Body			
2618	PositionID	Y	Unique position identifier
710	PosReqID	Y	PosReqID from RequestForPositions (AN)
11	ClOrdID	C	Unique client-provided identifier for the order led to position update

Tag	Field Name	Req	Description
10002	MPExecType	Y	1 = Created 2 = Open Trade 3 = Close Trade 4 = Flip Trade 5 = Updated 6 = Collateral Changed 7 = Closed 8 = Status
453	NoPartyIDs	Y	2 elements: User name, Central Counterparty Account
>448	PartyID	Y	User name (if PartyRole (452) = 3) or central counterparty account (if PartyRole (452) = 4)
>447	PartyIDSource	Y	D = Proprietary
>452	PartyRole	Y	3 = ClientID 4 = ClearingFirm
55	Symbol	Y	Trading symbol
10003	MPStatus	Y	1 = Normal (Default status of a margin position) 2 = Blocked (User can’t make any changes to position) 3 = CloseOnly (Reject all orders except the close-only ones)
10004	MarginMode	Y	1 = Isolated 2 = Cross
10005	Leverage	Y	Position leverage
1703	NoCollateralAmounts	Y	3 elements: Margin, Required Margin, Orders Margin
>1704	CurrentCollateralAmount	Y	Current collateral amount
>1705	CollateralCurrency	Y	Collateral currency

Tag	Field Name	Req	Description
>2632	CollateralAmountType	Y	101 = Margin 102 = RequiredMargin 103 = OrdersMargin
14	CumQty	Y	Position quantity (it will be a negative value for short positions)
6	AvgPx	C	Weighted average price of opening trades. None if CumQty (14) = 0
10006	EntryValue	C	Position value. None if CumQty (14) = 0
10007	CloseOnlyPx	C	None if CumQty (14) = 0
10008	LiquidationPx	C	None if CumQty (14) = 0
10009	BankruptcyPx	C	None if CumQty (14) = 0
10010	RealisedPnL	C	Realised profit/loss
10012	MPUntouchableFlag	C	Position cannot be force-closed. Not required if MPUntouchableFlag (10012) = false
10013	MPMarginCallFlag	Y	Additional margin for the position is required
42	OrigTime	Y	Datetime of position creation with nanosec precision
779	LastUpdateTime	Y	Datetime of position update with nanosec precision
Standard Trailer			

SetMargin

Sets margin for position or creates position, if it does not exist.

MsgType (35): USM = SetMargin

Tag	Field Name	Req	Description
Standard Header			
Message Body			
710	PosReqID	Y	Unique identifier for the position maintenance request as assigned by the submitter
1	Account	Y	An account number
453	NoPartyIDs	Y	Number of party identifiers. Must be '1'
>448	PartyID	Y	Name of a user
>447	PartyIDSource	Y	'D' = Proprietary
>452	PartyRole	Y	Role of the specified PartyID. '3' = Client ID
55	Symbol	Y	Trading symbol
1703	NoCollateralAmounts	Y	1 element: Margin
>1704	CurrentCollateralAmount	Y	New collateral amount
>1705	CollateralCurrency	Y	Collateral currency
>2632	CollateralAmountType	Y	101 = Margin
Standard Trailer			

CloseMarginPosition

MsfType (35): UCMP = CloseMarginPosition

Tag	Field Name	Req	Description
Standard Header			
Message Body			
710	PosReqID	Y	Unique identifier for the position maintenance request as assigned by the submitter
1	Account	Y	An account number
453	NoPartyIDs	Y	Number of party identifiers. Must be '1'
>448	PartyID	Y	Name of a user
>447	PartyIDSource	Y	'D' = Proprietary
>452	PartyRole	Y	Role of the specified PartyID. '3' = Client ID
55	Symbol	Y	Trading symbol
40	OrdType	Y	1 = Market 2 = Limit
44	Price	C	Only if OrdType (40) = Limit (2)
Standard Trailer			

MarginReject

MsgType (35): UMR = MarginReject

Tag	Field Name	Req	Description
Standard Header			
Message Body			
710	PosReqID	Y	Unique identifier for the position maintenance request as assigned by the submitter
55	Symbol	N	Trading symbol
10011	MPRejectReason	Y	Code specifying the reason for the reject: 0 = AccessDenied 1 = BadAccount 2 = BadAmount 3 = BadMarginValue 4 = BadOrdType 5 = BadPosReqId 6 = BadPositionId 7 = BadPrice 8 = BadSymbol 9 = DuplicateRequestId 10 = ExchangeClosed 11 = NoAvailableMargin 12 = PositionAlreadyExists 13 = PositionNotChanged 14 = PositionNotFound 15 = UnknownUser 99 = Other
58	Text	N	Text specifying the reason for the rejection
Standard Trailer			

6.5. Application messages: other

Business Message Reject

A Business Message Reject message is sent when a client message cannot be processed.

MsgType (35): j = Business Message Reject

Tag	Field Name	Req	Description
Standard Header			
Message Body			
379	BusinessRejectRefID	N	Client specified identifier (e.g. ClOrdID, OrdStatusReqID, etc.) of the rejected message if it is available.
45	RefSeqNum	N	MsgSeqNum (34) of the rejected message.
371	RefTagID	N	If a message is rejected to due to an issue with a particular field its tag number will be indicated.
372	RefMsgType	Y	MsgType (35) of the rejected message.
380	BusinessRejectReason	Y	Code specifying the reason for the reject. 0 = Other 1 = UnknownID 2 = UnknownSecurity 3 = UnsupportedMessageType 4 = ApplicationNotAvailable 5 = ConditionallyRequiredFieldMissing 6 = NotAuthorized 7 = DeliverToFirmNotAvailableAtThisTime 8 = ThrottleLimitExceeded 9 = ThrottleLimitExceededSessionDisconnected 10 = ThrottledMessagesRejectedOnRequest 18 = InvalidPriceIncrement
58	Text	N	Text specifying the reason for the rejection (not including TagID. See RefTagID for this information).
Standard Trailer			

FIX5 Market Data Gateway

Operational manual v1.1, 19-04-2021

Table of Contents

1. General

1.1. Introduction

The FIX5 Market Data Gateway enables clients to anonymously receive order book market data in real time. The interface is a point-to-point service based on the technology and industry standards TCP/IP, FIXT and FIX. The session and application event models and messages are based on versions 1.1 and 5.0 (Service Pack 2) of the FIXT and FIX protocols respectively.

1.2. Purpose

The purpose of this document is to provide a technical description of the FIX Market Data Gateway. This document outlines the service, its logic, and message types and fields involved.

1.3. Key

Usage	Meaning
Req	Required?
Y	Yes
N	No
C	Conditional

1.4. Document history

Issue	Date	Description
1.0	30-10-2019	First edition
1.1	19-04-2021	Added a separate service sending snapshots
1.2	04-06-2021	Added an empty snapshot on a subscription to updates

2. Service Overview

Market data consists of snapshots, increments, and trades. Trades are sent in the same messages as increments.

A snapshot is the current state of the order book for some specific trading pair and depth. Snapshots are sent periodically for all trading pairs and for different depths.

An increment is an incremental update from the previous state of the order book, based on the last snapshot. On the sending side, increments are buffered into a batch. Increments can be sent from this batch

1. Periodically, at an internally configured **interval** of time. The cut-off for this interval occurs for all trading pairs of a given depth at the same time.
2. When the batch buffer has filled to an internally configured size **limit**. The limit cut-off occurs independently of trading pairs and depths.

This means that, if **interval** = 40ms and **limit** = 100, an increment would be sent to the client either after 40ms, or after the 100th update to the market (e.g. 100 new orders were placed), whichever comes first.

Snapshots and increments are sent independently of each other by separate services and both may contain market data for the same time frame. Changes in Market data are indicated by a sequence number increased by 1 after any snapshot or increment.

3. Connectivity

3.1. CompIDs

The CompID of each client must be registered before FIX communications can begin. A single client may have multiple connections to the server (i.e. multiple FIX sessions, each with its own CompID).

The CompID of the server is ME. The messages sent to the server should contain the CompID assigned to the client in the field SenderCompID (49) and ME in the field TargetCompID (56). The messages sent from the server to the client will contain ME in the field SenderCompID (49) and the CompID assigned to the client in the field TargetCompID (56).

3.2. Passwords

Each new CompID will be assigned a username and a password on registration.

3.3. Message Rate Throttling

A scheme for throttling message traffic is implemented, where each CompID is only permitted to submit up to a specified number of messages per second.

Every message which exceeds the maximum rate of a CompID will be rejected via a Business Message Reject (with BusinessRejectReason (380) of ThrottleLimitExceeded (8)).

Please note that client Heartbeat messages, reject messages and any other client-initiated administrative messages are not counted towards the throttling limits.

4. FIX connections and sessions

4.1. Establishing a FIX connection

FIX connections and sessions between the client and server are maintained as specified in the FIX protocol.

Each client will use the assigned IP addresses and ports to establish a TCP/IP sessions with the server. Subscription to market data is available after establishing connection via ports according to the table below.

Market Data Feed	Port
Market Data — Incremental Refresh feed	4503
Market Data — Full Refresh feed	4513

NOTE These ports must be used by default if there were no specific ports assigned.

System supports sending increments independently from snapshots, meaning that snapshots would not affect the intensity of increments' publication. In order to access this functionality, the client must subscribe to Incremental Refresh feed with a custom subscription type ('50' = Updates) via port 4503 and then use port 4513 for requesting snapshots only.

The client will initiate a FIX session by sending the Logon (A) message. The client will identify themself using the SenderCompID (49) field. At each connection MsgSeqNum (34) is reset to 1, the client should set ResetSeqNumFlag (141) to "Y". The server will validate the CompID, password and IP address of the client. Once the client is authenticated, the server will respond with a Logon message.

The client must wait for the server’s Logon response before sending additional messages. If the client sends messages prior to sending the Logon message or prior to receiving the Logon response, the server will break the TCP/IP connection with the client without sending any message.

If a logon attempt fails because of an invalid SenderCompID, invalid TargetCompID, invalid IP address, invalid password or not having the appropriate privileges to login to the gateway, or if the user sends a Logon message with duplicated tags, the server will break the TCP/IP connection with the client without sending a Logout or Reject message.

If during a logon of a SenderCompID, the server receives a second connection attempt while a valid FIX session is already underway for that same SenderCompID, the server will break the TCP/IP connection with the second connection without sending a Logout or Reject message.

4.2. Maintaining a FIX session

4.2.1. Message sequence numbers

As outlined in the FIX protocol, the client and server will each maintain a separate and independent set of incoming and outgoing message sequence numbers. Sequence numbers should be initialized to 1 (one) at the start of the FIX session and be incremented throughout the session.

If any message sent by the client contains a sequence number that is less than what is expected and the PossDupFlag (43) is not set to “Y”, the server will send a Logout message and terminate the FIX connection. The Logout will contain the next expected sequence number as well as the received sequence number in the Text (58) field.

4.2.2. Heartbeats

The client and server will use the Heartbeat message to exercise the communication line during periods of inactivity and to verify that the interfaces at each end are available. The heartbeat interval will be the HeartBtInt (108) specified in the client’s Logon message.

The server will send a Heartbeat anytime it has not transmitted a message for the heartbeat interval. The client is expected to employ the same logic.

As a safety mechanism, the system will not allow the user to login if the HeartBtInt is set to 0.

If the server detects inactivity for a period longer than the heartbeat interval plus a reasonable transmission time, it will send a Test Request message to force a Heartbeat from the client. If a response to the Test Request is not received by a reasonable transmission time, the server will send a Logout and break the TCP/IP connection with the client. The client is expected to employ similar logic if inactivity is detected on the part of the server.

5. Handling Market Data

Clients can use the Market Data Request message to both subscribe to and unsubscribe from real-time receipt of Market Data from the exchange.

When making a Subscription Request, clients must supply the symbols they are requesting market data for and whether they want the entire order book or just the top of the book. Clients can request a one-time Snapshot, or a Snapshot plus ongoing streamed updates.

When unsubscribing, the client must pass the ID of the Subscription Request they wish to cancel.

A successful Market Data Subscription Request returns one or more Market Data (type W or type X) messages, each containing one or more Market Data Entries. Each Market Data Entry can be a Bid, an Offer or a Trade associated with a currency pair.

Requesting just the top of book will result in receipt of just the Best Bid and Best Offer. For a full book, the Bid and Offer side may each have several Market Data Entries.

A Snapshot causes the current state of the market to be sent. A Snapshot + Updates causes the current state of the market to be sent, and any updates as they occur, until the client requests that the Snapshot + Updates be disabled.

Each Market Data — Snapshot / Full Refresh (W) message contains a full order book for only one currency pair.

A Market Data Request Reject (message type Y) may be sent in response to a request that cannot be honoured.

6. Message Formats

Client-initiated:

Message	MsgType	Usage
Market Data Request	V	A general request for market data (order book snapshot, order book updates subscription/unsubscription).

Server-initiated:

Message	MsgType	Usage
Market Data Request Reject	Y	A message sent if the Market Data Request was incorrect. Indicates that the request cannot be processed.
Market Data - Snapshot/Full Refresh	W	A response to a Market Data Request message (full order book or order book top level).
Market Data - Incremental Refresh	X	A message contains order book updates and trades.

6.1. Message Header and Trailer

6.1.1. Message Header

Tag	Field Name	Req	Description
8	BeginString	Y	FIXT.1.1
9	BodyLength	Y	Number of characters after this field up to and including the delimiter immediately preceding the CheckSum.
35	MsgType	Yes	Message type.
49	SenderCompID	Y	CompID of the party sending the message.
56	TargetCompID	Y	CompID of the party the message is sent to.
34	MsgSeqNum	Y	Sequence number of the message.
43	PossDupFlag	N	Whether the message was previously transmitted under the same MsgSeqNum (34). Absence of this field is interpreted as Original Transmission (N). ('Y' = Possible Duplicate).
52	SendingTime	Y	Time the message was transmitted.
122	OrigSendingTime	C	Time the message was originally transmitted. If the original time is not available, this should be the same value as SendingTime (52). Required if PossDupFlag (43) is Possible Duplicate ('Y').

6.1.2. Message Trailer

Tag	Field Name	Req	Description
10	CheckSum	Y	CheckSum

6.2. Market Data messages

Market Data Request

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	V = Market Data Request
Message Body			
262	MDReqID	Y	Id of the request.
263	SubscriptionRequestType	Y	0 = Snapshot 1 = SnapshotAndUpdates 2 = DisablePreviousSnapshot 50 = Updates
264	MaketDepth	Y	0 = full book 1 = top of book
265	MDUpdateType	N	0 = FullRefresh (not supported) 1 = IncrementalRefresh (by default)
266	AggregatedBook	N	Y = Book entries to be aggregated (by default) N (not supported)
267	NoMDEntryTypes	Y	Must be 4, and the group must contain 0, 1, 2, J entry types.
>269	MDEntryType	Y	0 = Bid 1 = Offer 2 = Trade J = EmptyBook
146	NoRelatedSym	Y	Number of symbols.
>55	Symbol	Y	Symbol for which Market Data is requested.
Standard Trailer			

Market Data Request Reject

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	V = Market Data Request
Message Body			
262	MDReqID	Y	Id of the request.
281	MDReqRejReason	Y	0 = UnknownSymbol 1 = DuplicateMDReqID 2 = InsufficientBandwidth 3 = InsufficientPermissions 4 = UnsupportedSubscriptionRequestType 5 = UnsupportedMarketDepth 6 = UnsupportedMDUpdateType 7 = UnsupportedAggregatedBook 8 = UnsupportedMDEntryType 9 = UnsupportedTradingSessionID A = UnsupportedScope B = UnsupportedOpenCloseSettleFlag C = UnsupportedMDImplicitDelete D = InsufficientCredit
58	Text	N	Text specifying the reason for the rejection. If MDReqRejReason(281) is '0' = UnknownSymbol, it also contains the code of a symbol which caused the rejection.
Standard Trailer			

Market Data — Snapshot/Full Refresh

The Market Data Snapshot Full Refresh message (type W) is used as the response to a Market Data Request (type V) message.

A Snapshot is the current state of the order book at the time of the request; it contains only bids and

offers for the depth specified in the request.

A Snapshot is always sent in response to a Market Data Request (V) message, where the message type is '0' = Snapshot.

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	W = Market Data — Snapshot / Full Refresh
Message Body			
262	MDReqID	Y	Conditionally required if this message is in response to a MarketDataRequest (V).
55	Symbol	Y	Symbol for which Market Data is requested.
779	LastUpdateTime	Y	Timestamp of last update to data item. Date and time with nanosecond precision.
268	NoMDEntries	Y	Number of entries following.
>269	MDEntryType	Y	0 = Bid 1 = Offer J = EmptyBook
>270	MDEntryPx	Y	Price of the Market Data Entry.
>271	MDEntrySize	Y	Quantity represented by the Market Data Entry.
10020	MDSeqNum	N	Sequence number of the current Market data state.
Standard Trailer			

NOTE Tags in the range from 10000 to 10100 are custom. Any of them may be ignored if it doesn’t occur in this specification.

Market Data — Incremental Refresh

The Market Data Incremental Refresh (X) message is used as the response to a Market Data Request (type V) message with SubscriptionRequestType (263) set to '1' = SnapshotAndUpdates or '50' = Updates. It contains trades and incremental updates for the order book, such as new, changed, or deleted order book levels. A single message may contain several updates, but it may contain data for only one symbol.

Subscription to incremental updates apart from snapshots may result in a situation when the client cannot determine if they missed any updates since the subscription and before requesting a snapshot because of, possibly, low trading activity. To avoid that, the server will immediately send an empty snapshot with zero sequence number as the response to a successful subscription.

Incremental updates may arrive before Snapshots. To handle this, those are accumulated in a buffer. All incoming Increments are available to the client immediately, so those won't lose their relevance until a new Snapshot. Concurrent Increments and the Snapshot share common sequence, so the buffer is cleared from the incremental data which occurs in the Snapshot. Updates with exceeding sequence numbers are appended to the Snapshot just arrived.

Tag	Field Name	Req	Description
Standard Header			
35	MsgType	Y	X = Market Data — Incremental Refresh
Message Body			
262	MDReqID	Y	Conditionally required if this message is in response to a Market Data Request.
779	LastUpdateTime	Y	Date and time with nanosecond precision.
268	NoMDEntries	Y	Number of entries following.
>279	MDUpdateAction	Y	0 = New 1 = Change 2 = Delete
>269	MDEntryType	Y	0 = Bid 1 = Offer 2 = Trade
>55	Symbol	Y	Symbol for which Market Data is requested.
>270	MDEntryPx	Y	Price of the Market Data Entry.
>271	MDEntrySize	C	Quantity represented by the Market Data Entry. Trade size or final level size. It’s absent if MDUpdateAction (279) is '2' = Delete.
>1003	TradeID	C	Unique ID of the trade entity. It takes place if MDEntryType(269) is '2' = Trade only.
>272	MDEntryDate	C	Date of Market Data Entry. It takes place if MDEntryType(269) is '2' = Trade only.
>273	MDEntryTime	C	Time of Market Data Entry. Date and time with nanosecond precision. It takes place if MDEntryType(269) is '2' = Trade only.
>2446	AggressorSide	C	1 = Buy 2 = Sell It takes place if MDEntryType(269) is '2' = Trade only.
10020	MDSeqNum	N	Sequence number of the current Market data state.
Standard Trailer			

NOTE	Tags in the range from 10000 to 10100 are custom. Any of them may be ignored if it doesn’t occur in this specification.