# Order Entry Gateway
# FIX Specification

Please respond to:

newtradingplatform@lme.com

# Table of Contents

## Document History

| Version | Date | Change Description |
|---|---|---|
| 1.0 | 31/12/2019 | Initial draft |
| 1.1 | 09/04/2020 | Updated following internal review |
| 1.2 | 28/01/2022 | Internal updates |
| 1.3 | 24/03/2023 | Internal review |
| 1.4 | 23/06/2023 | RelatedHighPrice (1819) and RelatedLowPrice (1820) added to Order Cancel Reject (9)<br>4.11.7 added example message flow for Order rejected price limits breached. References to Market to Limit in message flow examples corrected to Market. RelatedHighPrice (1819) description includes Stop order handling<br>4.11.7.1 RelatedHighPrice (1819) and RelatedLowPrice (1820) conditional for Order Cancelled (Unsolicited) |
| 1.5 | 13/10/2023 | 1.1.2 password encryption example added<br>3.2.1 strategy creation clarified<br>4.4 timestamp precision<br>4.8.7 and 4.9.1 Text (58) is conditionally required if reject reason is Other<br>4.10.1 301 description<br>4.11.1 LegRatioQty (632) description and message flow examples<br>4.11.4 message description and party block |
| 1.6 | 15/03/2024 | 1.1.4.1 password reuse policy<br>1.2 duplicate connection termination removed<br>3.4 Stop Market and Stop Limit description<br>3.6.1 EncryptedPassword (1402) and EncryptedNewPassword (1404) length 450<br>3.9, 3.10 and 4.11.4 restated triggered Stop orders change order type<br>3.10 and 4.11.4 amendment of attributes and party details<br>3.12 and 3.18 cancellation by tradable instrument<br>4.10.1 LEI usage |

| Version | Date | Change Description |
|---|---|---|
| | | 4.11.6 OrdStatus description<br>4.11.7 OrderOrgination (1724) data type Int, ExecType (150) = 'E' Pending Replace returned in speedbump order handling<br>4.11.7.1 Restated, footnote added for triggered Stops. Triggered, mandatory tags. Rejections, Clearing Firm removed |
| 1.7 | 19/07/2024 | 1.1.2 public key location<br>3.8 expiry conditions<br>4.4 added String data type<br>4.8.7 and 4.9.1 Text (58) optional<br>4.10.1 PartyID format where Alphanumeric changed to String<br>4.11.7 ExpireDate (432) description |
| 1.7.1 | 09/08/2024 | 1.1.1 TargetCompID (56) for Production environment |

## Preface

This document describes the LME implementation of the FIX protocol based on FIX 5.0 SP2 Specification with relevant extension packs.

The document assumes the reader has an understanding of the FIX protocol, see http://www.fixprotocol.org/.

This document should be read in conjunction with related materials on LME.com for LMEselect v10.

## Delivery Phasing

This document covers all the functionality available in LMEselect 10 however functionality will be delivered in phased releases.

Functionality that will be included in a later release is specified in the following table and shown throughout the document in *dark grey italics*. The initial release will contain all functionality that is **not** specified in the table.

| Function | Reference |
|---|---|
| **Futures strategies:**<br>• **3 Month Average**<br>• **6 Month Average**<br>• **12 Month Average**<br>• **Carry Average** | 3.2.1.1 Exchange Defined Strategy Types<br>4.11.1 Security Definition Request (c) |
| **Options strategies:**<br>• **Call spread**<br>• **Put spread** | 3.2.1.1 Exchange Defined Strategy Types<br>4.11.1 Security Definition Request (c) |
| **Custom strategies** | 3.2.1 Strategies<br>3.2.1.2 Custom Strategies<br>4.11.1 Security Definition Request (c)<br>4.11.2 Security Definition (d) (Example Message Flows)<br>4.11.7.1 Execution Report (8) (Example Message Flows) |
| **Option contracts** | 3.2 Security Creation<br>3.2.1.1 Exchange Defined Strategy Types<br>3.2.1.2 Custom Strategies<br>4.11.1 Security Definition Request (c)<br>4.11.2 Security Definition (d) (Example Message Flows) |
| **Order types:**<br>• **Market** | 3.4 Order Types<br>3.6 Order Types and Permitted Order Validity Conditions |

| Function | Reference |
|---|---|
| • Stop Market<br>• Iceberg<br>• Post Only<br>• One Cancels Other | 3.10 Order Amendment (DisplayQty)<br>4.11.3 New Order Single (D)<br>4.11.4 Order Cancel Replace Request (G)<br>4.11.7 Execution Report (8)<br>4.11.7.1 Execution Report Matrix |
| **Order validity condition:**<br>• Fill or Kill (FOK) | 3.5 Order Validity Conditions<br>3.6 Order Types and Permitted Order Validity Conditions<br>4.11.3 New Order Single (D)<br>4.11.4 Order Cancel Replace Request (G)<br>4.11.7 Execution Report (8) |
| **Request for Quote** | 3.14 Request for Quote (RFQ)<br>3.16 Message Throttling<br>4.1 Supported Messages<br>4.2 Inbound Messages<br>4.3 Outbound Messages<br>4.10.11 Quote Request (R)<br>4.11.11 Quote Response (AJ)<br>4.11.12 Quote Request Reject (AG) |
| **Speed Bumps** | 3.15 Speed Bumps<br>4.11.7 Execution Report (8)<br>4.11.7.1 Execution Report Matrix<br>Appendix B: Speed Bump Inflight Order Handling |
| **Self Execution Prevention** | 3.19 Self Execution Prevention (SEP)<br>4.11.3 New Order Single (D)<br>4.11.7 Execution Report (8)<br>4.11.7.1 Execution Report Matrix |

# 1 Session Management

## 1.1 Authentication

### 1.1.1 Comp ID

A FIX session is established by sending a Logon (35=A) request which includes the sender and the target in the Message Header:

- SenderCompID (49) – the party initiating the session.
- TargetCompID (56) – the acceptor of the session as per configuration.

For messages sent to the Exchange, the client should use the session CompID allocated by the Exchange to populate SenderCompID (49). A single client may have multiple connections to the gateway i.e. multiple FIX sessions, each with its own Comp ID.

The TargetCompID (56) in messages sent to the Exchange is environment specific as follows:

Production:

- CGLME

Member Test environments:

- LMEMTA
- LMEMTB

### 1.1.2 Password Encryption

The client should specify their password in EncryptedPassword (1402) in the Logon request.

To encrypt the password, the client is expected to use a 2048-bit RSA (http://en.wikipedia.org/wiki/RSA_(algorithm)) public key circulated by the Exchange on https://www.lme.com/Trading/Systems/LMEselect~~https://www.lme.com/en/Trading/Initiatives/New-trading-platform/Access~~. The binary output of the RSA encryption must be represented in Big Endian PKCS #1 with padding scheme OAEP (https://en.wikipedia.org/wiki/PKCS_1) and then converted to an alphanumeric value by means of standard base-64 encoding (http://en.wikipedia.org/wiki/Base64) when communicating with the gateway.

Password encryption example:

```
public static String encrypt(String value) throws CrytographyException {
try {
Cipher cipher = Cipher.getInstance("RSA/ECB/OAEPWithSHA-1AndMGF1Padding");
cipher.init(Cipher.ENCRYPT_MODE, publicKey);
byte [] bytes = cipher.doFinal(value.getBytes());
return Base64.getEncoder().encodeToString(bytes);
} catch (NoSuchAlgorithmException | NoSuchPaddingException |
InvalidKeyException | IllegalBlockSizeException | BadPaddingException e) {
throw new CrytographyException(e.getMessage());
}
}
…….
```

```
String pubKey = new String(keyBytes, "UTF-8");

pubKey = pubKey.replaceAll("(-+BEGIN PUBLIC KEY-+\\r?\\n|-+END PUBLIC KEY-
+\\r?\\n?)", "");

pubKey = pubKey.replaceAll("(-+BEGIN RSA PUBLIC KEY-+\\r?\\n|-+END RSA
PUBLIC KEY-+\\r?\\n?)", "");

pubKey = pubKey.replaceAll("\\n|\\r","");

KeyFactory keyFactory = KeyFactory.getInstance("RSA");

X509EncodedKeySpec keySpec = new
X509EncodedKeySpec(Base64.getDecoder().decode(pubKey.getBytes()));

publicKey = keyFactory.generatePublic(keySpec);
```

### 1.1.3 Password

The gateway authenticates the participant’s Logon (35=A) request and sends a Logon (35=A) response containing SessionStatus (1409) which indicates whether the logon attempt was successful or not.

Repeated failures in password validation will result in the client account being locked. The participant is expected to contact the Exchange to unlock the client account.

### 1.1.4 Change Password

Each new Comp ID will be assigned a password by the Exchange. The client is expected to change this password upon initial logon.

A password change can be made in a Logon (35=A) request. The client should specify the new encrypted password in EncryptedNewPassword (1404) and the current encrypted password in EncryptedPassword (1402).

The new password must comply with Exchange’s password policy. The status of the new password (i.e. whether it is accepted or rejected) will be specified in the SessionStatus (1409) response from the gateway. The new password, if accepted, will be effective for subsequent logins.

#### 1.1.4.1 Password Policy

The Exchange requires the password to contain:

- Minimum of 8 characters
- At least one number
- Combination of uppercase and lowercase characters.

Password history is retained and therefore the last 24 passwords cannot be reused.

## 1.2 Establishing a FIX Session

The client must wait for a successful Logon response before sending additional messages. If additional messages are received from the client before the exchange of Logon messages, the TCP/IP connection with the client will be disconnected.

If a Logon (35=A) attempt fails for the following reasons, the gateway will send a Logout (35=5) or a Reject (35=3) and then terminate the session:

- Password failure
- Comp ID is locked
- Logon is not permitted during this time

For all other reasons, including the following, the gateway will terminate the session without sending a Logout or Reject:

- Invalid Comp ID

Inbound message sequence number will not be incremented if the connection is abruptly terminated due to the logon failure.

If a session level failure occurs due to a message sent by the client which contains a sequence number that is less than what is expected and the PossDup (43) is not set to Y = Yes, then the gateway will send a Logout (35=5) and terminate the FIX session. In this scenario the inbound sequence number will not be incremented.

## 1.3 Message Sequence Numbers

As outlined in the FIX protocol, the client and gateway 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 day and be incremented throughout the session. Either side of a FIX session will track the:

- NextExpectedMsgSeqNum (789) (starting at 1) in Logon (35=A)
- MsgSeqNum (34) in the Message Header (starting at 1); with respect to the contra-party.

The MsgSeqNum (34) in the Message Header is always incremented by the sender, whereas the NextExpectedMsgSeqNum (789) is only updated as a result of an incoming message.

Monitoring sequence numbers will enable parties to identify and react to missed messages and to gracefully synchronize applications when reconnecting during a FIX session.

Any message sent by either side of a FIX session will increment the sequence number unless explicitly specified for a given message type.

If any message sent by one side of a FIX session contains a sequence number that is LESS than the NextExpectedMsgSeqNum (789) then the other side of this session is expected to send a Logout message and terminate the FIX connection immediately, unless the PossDup flag is set to Y = Yes

A FIX session will not be continued to the next trading day. Both sides are expected to initialize (reset to 1) the sequence numbers at the start of each day.  At the start of each trading day if the client starts with a NextExpectedMsgSeqNum (789) greater than 1 then the gateway will send a Logout message and terminate the session immediately without any further exchange of messages.

## 1.4 Heartbeat and Test Request

The client and the gateway will use the Heartbeat (35=0) message to monitor the communication line during periods of inactivity and to verify that the interfaces at each end are available.

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

If the gateway detects inactivity for a period longer than 3 heartbeat intervals, it will send a Test Request message to force a Heartbeat from the client. If a response to the Test Request (35=1) is not received within a reasonable transmission time (recommended being an elapsed time equivalent to 3 heartbeat intervals), the gateway will send a Logout (35=5) 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 gateway.

## 1.5 Terminating a FIX Session

Session termination can be initiated by either the gateway or the client by sending a Logout (35=5). Upon receiving the Logout request, the contra party will respond with a Logout message signifying a Logout reply. Upon receiving the Logout reply, the receiving party will terminate the connection.

If the contra-party does not reply with either a Resend Request or a Logout reply, the Logout initiator should wait for 60 seconds prior to terminating the connection.

The client is expected to terminate each FIX connection at the end of each trading day before the gateway is shut down. Any open FIX connections will be terminated by the gateway sending a Logout when the service is shut down. Under exceptional circumstances, for example, a slow consumer, the gateway may initiate the termination of a connection during the trading day by sending a Logout.

If, during the exchange of Logout messages, the client or the gateway detects a sequence gap, it should send a Resend Request.

## 1.6 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.

Once the FIX session is re-established, the message sequence numbers will continue from the last message successfully transmitted prior to the termination as described in 2.7 Transmission of Missed Messages.

## 1.7 Sequence Reset

Gap-fill mode can be used by one side when skipping session level messages which can be ignored by the other side.

During a FIX session the gateway or the client may use the Sequence Reset (35=4) message in Gap Fill mode if either side wishes to increase the expected incoming sequence number of the other party.

It will not be possible to reset the client sequence number to 1 using the Logon message. Should a reset be required the participant should contact the Exchange.

The client is required to support a manual request by Exchange to initialize sequence numbers prior to the next login attempt.

## 1.8 Fault Tolerance

After a failure on client side or on gateway side, the client is expected to be able to continue the same session.

If the sequence number is reset to 1 by the gateway, all previous messages will not be available for the client side.

The client and the gateway are expected to negotiate on the NextExpectedMsgSeqNum (789) and Next To Be Received Sequence number by contacting the Exchange prior to initiating the new session and consequently manually setting the sequence number for both ends after having a direct communication with the participant.

## 1.9 Checksum Validation

The gateway performs a checksum validation on all incoming messages into the input services. Incoming messages that fail the checksum validation will be rejected and the connection will be dropped by the gateway without sending a logout.

Conversely, in case of a checksum validation failure, the client is expected to drop the connection and take any appropriate action before reconnecting.

Messages that fail the checksum validation should not be processed.

# 2 Recovery

## 2.1 General Message Recovery

Message gaps may occur which are detected via the tracking of incoming sequence numbers. Recovery will be initiated if a gap is identified when an incoming message sequence number is found to be greater than NextExpectedMsgSeqNum (789) during Logon or the MsgSeqNum (34) at other times.

The Resend Request will indicate the BeginSeqNo (7) and EndSeqNo (16) of the message gap identified and when replying to a Resend Request, the messages are expected to be sent strictly honouring the sequence.

If messages are received outside of the BeginSeqNo and EndSeqNo, then the recovering party is expected to queue those messages until the gap is recovered.

During the message recovery process, the recovering party will increment the Next Expected Sequence number accordingly based on the messages received. If messages applicable to the message gap are received out of sequence then the recovering party will drop these messages.

The party requesting the Resend Request can specify “0” in the EndSeqNo to indicate that they expect the sender to send ALL messages starting from the BeginSeqNo.In this scenario, if the recovering party receives messages with a sequence greater than the BeginSeqNo, out of sequence, the message will be ignored.

Administrative messages such as Sequence Reset, Heartbeat and Test Request which can be considered irrelevant for a retransmission could be skipped using the Sequence Reset message in gap-fill mode. Note that the gateway expects the client to skip Sequence Reset messages when replying to a Resend Request at all times.

When resending messages, the gateway would use either PossDup or PossResend flag to indicate whether the messages were retransmitted earlier. If PossDup flag is set to Y = Yes, it indicates that the same message with the given sequence number with the same business content may have been transmitted earlier. In the case where PossResend flag is set to Y = Yes, it indicates that the same business content may have been transmitted previously but under the different message sequence number. In this case business contents needs to be processed to identify the resend. For example, in Execution Reports the ExecID (17) may be used for this purpose.

## 2.2 Resend Request

The client may use the Resend Request (35=2) message to recover any lost messages. This message may be used in one of three modes:

1. To request a single message. The BeginSeqNo and EndSeqNo should be the same.
2. To request a specific range of messages. The BeginSeqNo should be the first message of the range and the EndSeqNo should be the last of the range.
3. To request all messages after a particular message. The BeginSeqNo should be the sequence number immediately after that of the last processed message and the EndSeqNo should be zero (0).

## 2.3 Logon Message Processing – Next Expected Message Sequence

The session initiator should supply the NextExpectedMsgSeqNum (789) the value next expected from the session acceptor in MsgSeqNum (34). The session acceptor should validate the logon request including that NextExpectedMsgSeqNum (789) does not represent a gap. It then constructs its logon response with NextExpectedMsgSeqNum (789) containing the value next expected from the session initiator in MsgSeqNum (34) having incremented the number above the logon request if that was the sequence expected.

The session initiator must wait until the logon response is received in order to submit application messages. Once the logon response is received, the initiator must validate that NextExpectedMsgSeqNum (789) does not represent a gap.

In case of gap detection from either party (lower than the next to be assigned sequence) recover all messages from the last message delivered prior to the logon through the specified NextExpectedMsgSeqNum (789) sending them in order, then gap fill over the sequence number used in logon and proceed sending newly queued messages with a sequence number one higher than the original logon.

Neither side should generate a Resend Request (35=2) based on MsgSeqNum (34) of the incoming Logon message but should expect any gaps to be filled automatically by following the Next Expected Sequence processing described above. Whilst the gateway is resending messages to the client, the gateway does not allow another Resend Request (35=2) from the client. If a new Resend Request is received during this time, the gateway will terminate the session immediately without sending the Logout (35=5) message.

Note that indicating the NextExpectedMsgSeqNum (789) in the Logon (35=A) is mandatory.

## 2.4 Possible Duplicates

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

## 2.5 Possible Resends

The gateway does not handle possible resends for the client-initiated messages (e.g. New Order, etc.) and the message will be processed without considering the value in the PossResend (97) field. Any message with duplicate ClOrdID (11) will be rejected based on the Client Order ID uniqueness check and messages which conform to the uniqueness check will be processed as normal messages.

The gateway may use the PossResend (97) field to indicate that an application message may have already been sent under a different MsgSeqNum (34). The client should validate the contents (e.g. ExecID (17)) of such a message against those of messages already received during the current trading day to determine whether the new message should be ignored or processed.

## 2.6 Gap Fills

The following messages are expected to be skipped using gap-fills when being retransmitted:

1. Logon

2. Logout
3. Heartbeat
4. Test Request
5. Resend Request
6. Sequence Reset

All other messages are expected to be replayed within a retransmission.

## 2.7 Transmission of Missed Messages

The Execution Report, Order Mass Cancel Report, Business Message Reject, Reject and News messages generated during a period when a client is disconnected from the gateway will be sent to the client when it next reconnects on the same business day. In the unlikely event the disconnection was due to a gateway outage, some messages may not be retransmitted and the messages which will be retransmitted will include a PossResend (97) set to Y = Yes.

# 3 Service Description

## 3.1 Security Identification

Each tradable instruments will be identified using a SecurityID (48) which can be a maximum of 19 digits. It is required to specify SecurityIDSource (22) as ‘8’ Exchange Symbol in conjunction with the SecurityID (48).

## 3.2 Security Creation

A Security Definition Request (35=c) can be submitted to create a new tradable instrument:

| Instrument Request Type | FIX Tag |
|---|---|
| *Option strike* | *SecurityType (167) = OPT*<br>*MaturityDate (541)*<br>*StrikePrice (202)*<br>*PutOrCall (201)* |
| Strategy | SecurityType (167) = MLEG<br>SecuritySubType (762)<br>LegSecurityID (602)<br>LegSecurityIDSource (603)<br>LegRatioQty (623)<br>LegSide (624)<br>*LegPrice (566) [Optional]* |

### 3.2.1 Strategies

A trader can submit Security Definition Request (35=c) for an Exchange defined strategy type or a *custom strategy. A Delta Hedge strategy can be submitted as a custom strategy.*

A strategy can be submitted from either a buy or sell perspective and must include the strategy legs in order of expiry. A Security Definition Request expressed from a sell perspective will be returned with a SecurityResponseType (323) = ‘2’ Accept security proposal with revisions as indicated in the message and the resulting strategy will be created from the buy side perspective.

#### 3.2.1.1 Exchange Defined Strategy Types

The following defined strategy types are supported:

**Futures Strategies**

| SecuritySubType (762) | Strategy Name | Definition (from buy perspective) |
|---|---|---|
| 1 | Carry | Buy near leg, sell far leg |
| *3* | *Average 3M* | *Buying 3 consecutive (monthly) legs* |
| *4* | *Average 6M* | *Buying 6 consecutive (monthly) legs* |
| *5* | *Average 12M* | *Buying 12 consecutive (monthly) legs* |
| *6* | *Carry Average* | *Buy an outright (e.g. 3M), sell a Future Average (e.g. first quarter 2023).* |

*An Average strategy is only permitted in monthly prompts and only the front leg needs to be specified as the remaining legs will be consecutive.*

*A Carry Average is the only permitted nested strategy type.*

***Options Strategies***

| SecuritySubType (762) | Strategy Name | Definition (from buy perspective) |
|---|---|---|
| *7* | *Call Spread* | *Buy a (call) strike, sell a (call) higher strike within the same option expiry* |
| *8* | *Put Spread* | *Buy a (put) strike, sell a (put) lower strike within the same option expiry* |

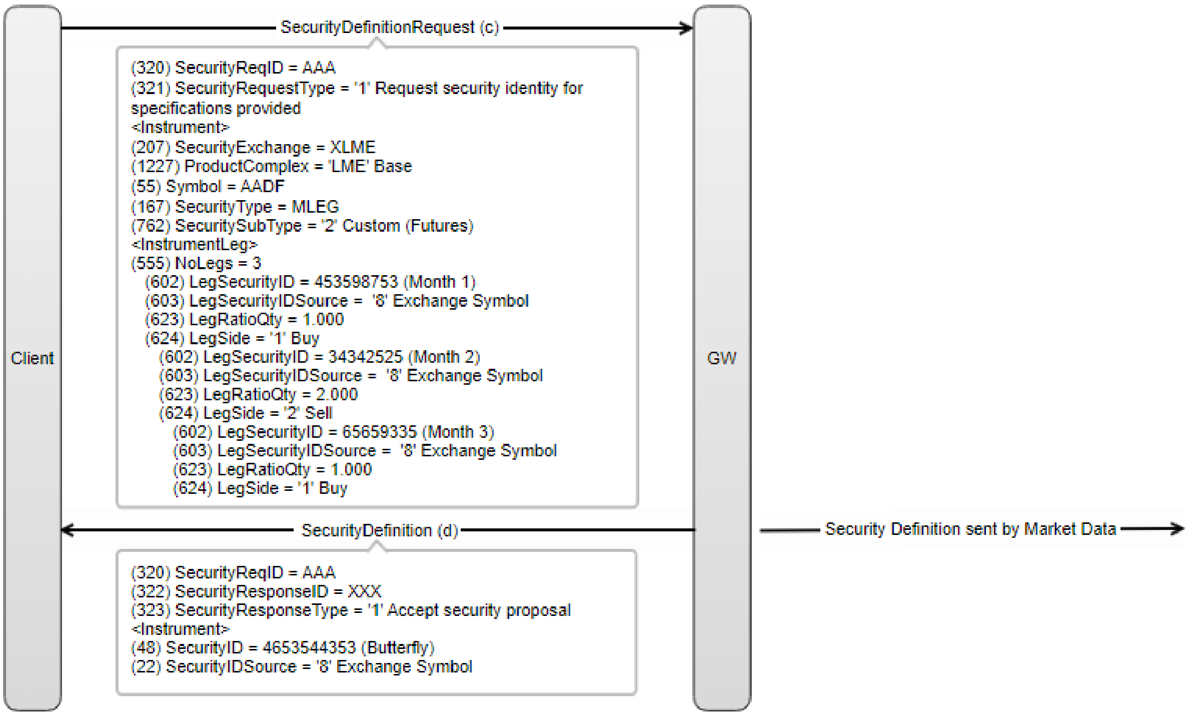
#### *3.2.1.2 Custom Strategies*

*A non-Exchange defined strategy can be submitted as a custom strategy using either:*

- *SecuritySubType (762) = ‘2’ Custom (Futures)*
- *SecuritySubType (762) = ‘9’ Custom (Delta Hedge)*
- *SecuritySubType (762) = ‘10’ Custom (Options).*

*A custom strategy can consist of up to five legs in a Futures contract or premium quoted Option. Each leg in the strategy must be in the same contract except for a delta hedge strategy in premium-based options where the last 1 to 2 legs belong to the underlying futures contract. Note, an Exchange defined strategy cannot be used within a custom strategy.*

*For example, a Futures Butterfly can be defined as buy Month 1, sell Month 2 twice and buy Month 3.*


Client
SecurityDefinitionRequest (c)
GW
SecurityDefinition (d)
Security Definition sent by Market Data

```
(320) SecurityReqID = AAA
(321) SecurityRequestType = '1' Request security identity for
specifications provided
<Instrument>
(207) SecurityExchange = XLME
(1227) ProductComplex = 'LME' Base
(55) Symbol = AADF
(167) SecurityType = MLEG
(762) SecuritySubType = '2' Custom (Futures)
<InstrumentLeg>
(555) NoLegs = 3
  (602) LegSecurityID = 453598753 (Month 1)
  (603) LegSecurityIDSource =  '8' Exchange Symbol
  (623) LegRatioQty = 1.000
  (624) LegSide = '1' Buy
    (602) LegSecurityID = 34342525 (Month 2)
    (603) LegSecurityIDSource =  '8' Exchange Symbol
    (623) LegRatioQty = 2.000
    (624) LegSide = '2' Sell
      (602) LegSecurityID = 65659335 (Month 3)
      (603) LegSecurityIDSource =  '8' Exchange Symbol
      (623) LegRatioQty = 1.000
      (624) LegSide = '1' Buy
```

```
(320) SecurityReqID = AAA
(322) SecurityResponseID = XXX
(323) SecurityResponseType = '1' Accept security proposal
<Instrument>
(48) SecurityID = 4653544353 (Butterfly)
(22) SecurityIDSource = '8' Exchange Symbol
```

## 3.3 Order Submission

It is possible to submit orders for outright futures, options series or strategies using any of the order types specified in 3.4.1 Order Types. An individual order can be submitted using New Order Single (35=D) which includes a unique Client Order identifier in the ClOrdID (11), see Order Identification.

Multiple orders can be submitted using Mass Quotes available in the Binary protocol.

## 3.4 Order Types

The following order types are supported:

| Order Type | FIX Tag and Value |
|---|---|
| **Limit**<br>An order submitted with a price and volume that will trade at the limit price or better for as much of its stated volume as is available in the order book. | OrdType (40) = 2<br>Price (44) |
| ***Market***<br>*An order submitted with a volume specified but no price. The order is executed at the best available price(s) up / down to their assigned limit price. Any order volume which is not fully executed rests in the order book as a limit order at the assigned limit price.* | *OrdType (40) = 1* |
| ***Stop Market*** | *OrdType (40) = 3* |

| Order Type | FIX Tag and Value |
|---|---|
| *An order that is submitted but not visible in the order book until it is triggered by the last traded price and/or best bid/offer. Once triggered the order is entered into the order book as a Stop Market order.*<br>*A previously triggered Stop order will be restated as a Limit order. StopPx (99) and TriggeringInstruction component block tags will not be present.* | *StopPx (99)*<br>*TriggerType (1100) = 4*<br>*TriggerPriceType (1107)* |
| **Stop Limit**<br>An order that is submitted but not visible in the order book until it is triggered by the last traded price and/or best bid/offer. Once triggered the order is entered into the order book as a Stop Limit order at the specified price.<br>A previously triggered Stop order will be restated as a Limit order. StopPx (99) and TriggeringInstruction component block tags will not be present. | OrdType (40) = 4<br>Price (44)<br>StopPx (99)<br>TriggerType (1100) = 4<br>TriggerPriceType (1107) |
| ***Iceberg***<br>*An order submitted with a visible order quantity and a total order quantity. The visible order quantity must be fully executed before it can be replenished with the next visible order quantity.* | *OrdType (40) = 2*<br>*Price (44)*<br>*DisplayQty (1138)*<br>*OrderQty (38)* |
| ***Post Only***<br>*The order must rest in the order book before it can trade. If the order can be executed on entry into the order book it is rejected. If an amendment to the order can result in execution it also is rejected and the original order remains.* | *OrdType (40) = 2*<br>*Price (44)*<br>*ExecInst (18) = 6* |
| ***One Cancels Other (OCO)***<br>*A single order which is a combination of a Limit and a Stop. On submission the Limit price and a Stop trigger price is specified.*<br>*A partial trade at the Limit price will reduce the quantity available in the OCO. If the order is traded out at the Limit price the Stop component will be cancelled. Similarly if the Stop is triggered then the Limit component is cancelled.*<br>*Note: No Execution Report will be generated for the cancelled component.* | *OrdType (40) = 2*<br>*Price (44)*<br>*TriggerType (1100) = 4*<br>*TriggerPrice (1102)*<br>*TriggerPriceType (1107)*<br>*TriggerNewPrice (1110)*<br>*TriggerOrderType (1111)* |

## 3.5 Order Validity Conditions

| Validity Condition | FIX Tag and Value |
|---|---|
| **Day**<br>An order that will expire at the end of the day. | TimeInForce (59) = 0 (default) |
| **Good Till Cancelled (GTC)**<br>An order that is valid until it is either cancelled or matched. | TimeInForce (59) = 1 |
| **Immediate or Cancel (IOC)**<br>An order that is executed at the stated price or better for as much order volume that is available. Any order volume that cannot be traded is cancelled. | TimeInForce (59) = 3 |
| ***Fill or Kill (FOK)***<br>*An order that is only executed if there is sufficient volume available, at the stated price or better, for them to execute fully. Otherwise the entire order is cancelled.* | *TimeInForce (59) = 4* |
| **Good Till Date (GTD)**<br>The order is valid until the end of the trading date specified. | TimeInForce (59) = 6<br>ExpireDate (432) |

Note: A GTC or GTD order cannot be entered into the TOM prompt.

A GTD will be rejected if the expiry date entered is the current trading date. A GTD in a single prompt will be rejected if the date entered exceeds the last trading date.

## 3.6 Order Types and Permitted Order Validity Conditions

| Order Type | Day | GTC | IOC | *FOK* | GTD |
|---|---|---|---|---|---|
| Limit | ✔ | ✔ | ✔ | ✔ | ✔ |
| *Market* | ✔ | ✔ | ✔ | ✔ | ✔ |
| *Stop Market* | ✔ | ✔ | | | ✔ |
| Stop Limit | ✔ | ✔ | | | ✔ |
| *Iceberg* | ✔ | ✔ | | | ✔ |
| *Post Only* | ✔ | ✔ | | | ✔ |
| *OCO* | ✔ | ✔ | | | ✔ |

## 3.7 Order Identification

On order submission a ClOrdID (11) is specified by the originator. The client should comply with the FIX protocol and ensure the Client Order IDs are unique for the duration of the trading day and has not been used already for a currently persisted order belonging to this CompID. When an order is accepted, the system assigns an OrderID (37) that is unique for all orders. When modifying or deleting an order the OrigClOrdID (41) is used to identify the order.

## 3.8 Order Expiry

No Execution Report will be sent for orders with a TimeInForce (59) = '0' Day when they expire at the end of the trading day.

At the end of the day, the order originator will receive an Execution Report with ExecType (150) and OrdStatus (39) = 'C' Expired for TimeInForce (59) = '1' Good Till Cancelled and TimeInForce (59) = '6' Good Till Date in the following cases:

- ExpireDate (432) has passed for a Good Till Date order*
- Last trading date for the tradable instrument has passed
- To prevent restatement into the Tom order book
- Any other Exchange specific configuration for order expiry of persisted orders
- Strategy contains a leg that has expired or meets any of the conditions above
- Legs of a strategy have the same prompt date.

*Note where the expiry date is a non-business date, the order will expire at the start of the next trading date.

## 3.9 Order Restatement

GTC/GTD orders that have not hit their expiry condition are persisted when the respective instrument enters the Close state. The order originator is notified by Execution Report with ExecType (150) and OrdStatus (39) = '3' Done for Day.

On initial logon on the next trading day, Execution Reports are sent for persisted orders that have been returned with ExecType (150) = 'D' Restated, OrdStatus (39) = '0' New or '1' Partially Filled and ExecRestatementReason (378) = '1' GT renewal / restatement.

A previously triggered Stop order will be restated as a Limit order. StopPx (99) and TriggeringInstruction component block tags will not be present.

## 3.10 Order Amendment

An order can be amended by using an Order Cancel Replace Request (35=G) and specifying the OrigClOrdID (41). The client can optionally specify the OrderID (37) in the Order Cancel Replace Request message. If the OrderID (37) is specified, the system will validate whether the OrderID is associated with the correct order as identified using the OrigClOrdID (41). The Order Cancel Replace Request will be rejected if the specified OrderID (37) is invalid based on this validation.

The following order attributes can be modified if they have been specified on the original order:

- Price (44)
- StopPx (99)
- OrderQty (38)
- *DisplayQty (1138)*
- ExpireDate (432)
- *TriggerPrice (1102)*
- *TriggerNewPrice (1110)*
- OrderCapacity (528)
- OrderRestrictions (529)

If an attribute listed above is not present in the Order Cancel Replace Request it will retain its original value.

For the following attribute, if value(s) have been previously specified and are not specified in the Order Cancel Replace Request it indicates that the values have been removed

- OrderOrigination (1724)
- OrderAttributeType (2594)
- Text (58)

The PartyID (448) on the following MiFID regulatory reporting roles can be amended if the PartyRole (452):

- 300 Investment Decision Within Firm
- 302 Investment Decision Country
- 303 Execution Decision Country
- 304 Client Branch Country

For the above roles, if the party was not specified on the original order, it can be added. If previously included it can be amended or can be omitted to indicate that the party has been removed.

The client cannot amend an order that is fully filled or cancelled or expired.

The StopPx (99), *TriggerPrice (1102) or TriggerNewPrice (1110)* cannot be amended if the Stop order has been triggered. Note, a previously triggered Stop Limit or *Stop Market* order will be restated with an OrdType (40) = 2 Limit.

If the client sends an Order Cancel Replace Request for an order for which an Order Cancel Replace Request is being processed the second Order Cancel Replace Request is rejected. If an Order Cancel Request is submitted for an Order Cancel Replace Request that is being processed, the incoming Order Cancel Request will be accepted.

## 3.11 Order Cancellation

An individual order can be cancelled using Order Cancel Request (35=F) by specifying the OrigClOrdID (41).

The client can optionally specify the OrderID (37) in the Order Cancel Request. If the OrderID (37) is specified, the system will validate whether the OrderID is associated with the correct order as identified by the OrigClOrdID. The Order Cancel Request will be rejected if the specified OrderID is invalid based on this validation.

A successful cancellation will return an Execution Report (35=8). If the cancellation request is rejected, an Order Cancel Reject (35=9) is sent containing CxlRejReason (102).

## 3.12 Mass Cancellation

Multiple orders can be cancelled using Order Mass Cancel Request (35=q) by specifying which orders are to be cancelled:

| Cancellation Type | FIX Tag and Value |
|---|---|
| All orders for a FIX Comp ID | MassCancelRequestType (530) = 7 |
| All orders for a tradable instrument | MassCancelRequestType (530) = 1<br>SecurityID (48)<br>SecurityIDSource (22) |
| All orders for a specified contract | MassCancelRequestType (530) = 3<br>SecurityExchange (207)<br>ProductComplex (1227)<br>Symbol (55) |
| All orders for an end client account | MassCancelRequestType (530) = 7<br>PartyRole (452) = ‘81’ Broker Client ID |
| All orders for a specific contract and side of the market | MassCancelRequestType (530) = 3<br>SecurityExchange (207)<br>ProductComplex (1227)<br>Symbol (55)<br>Side (54) |

If the Order Mass Cancel Request is accepted, Execution Reports will be sent for each order cancellation and reference the ClOrdID (11) provided on the Order Mass Cancel Request. An Order Mass Cancel Report (35=r) will follow and specify the TotalAffectedOrders (533) in the MassCancelResponse (531).

If the Order Mass Cancel Request is rejected, an Order Mass Cancel Report will be sent with the MassCancelResponse (531) = ‘0’ Cancel Request Rejected and will include the MassCancelRejectReason (532).

A mass cancellation request for a tradable instrument will not result in the cancellation of any orders in a merged tradable instrument. Orders will only be cancelled in the SecurityID (48) specified in the Order Mass Cancel Request.

## 3.13 Cancel on Disconnect

The gateway will not automatically cancel a user’s non-persisted orders in the event of a Logout. A user should explicitly cancel such orders prior to Logout using an Order Mass Cancel Request (35=q).

On order submission a user can specify whether non-persisted orders should be cancelled on system disconnection (due to, for example, a network issue or in the event of inactivity such as too many missed heartbeats) using ExecInst (18) = ‘o’ Cancel on Connection Loss.

On detection of a loss of connectivity, the system will use ExecInst (18) = ‘o’ Cancel on Connection Loss to determine whether a user’s non-persisted orders are cancelled.

This feature does not guarantee that all live orders will be successfully cancelled as executions that occur very near to the time of disconnect may not be reported to the client. It also depends on the tradable instrument trading state when the abrupt disconnection is identified by the Exchange system.

## *3.14 Request for Quote (RFQ)*

*A Quote Request (35=R) indicates a trading interest in a specific instrument which is published to market participants by the Market Data service.*

*The Quote Request will include the QuoteRequestType (303) which specifies whether a single quote or streaming quotes are requested. It can optionally specify the side and the quantity for which a price is required.*

*A Quote Response (35=AJ) will be returned by the gateway to acknowledge a successful Quote Request.*

*Trading participants can then respond using standard order functionality.*

## *3.15 Speed Bumps*

*Exchange contracts may be configured with speed bumps. A speed bump will only be applicable to New Order Single and Order Cancel Replace Requests.*

*Passive orders, Order Cancel Requests, Order Mass Cancel Requests and Mass Quotes will be exempt.*

*The status of an order in a speed bump will be reported in ExecTypeReason (2431) in the Execution Report:*

*101 = Order accepted but speed bump applied*

*102 = Order added after speed bump*

*103 = Order cancelled whilst in speed bump delay*

*104 = Original order is in speed bump enforced delay*

*105 = Order updated after speed bump delay*

*106 = Amend is in speed bump delay*

*107 = Order amended after speed bump delay*

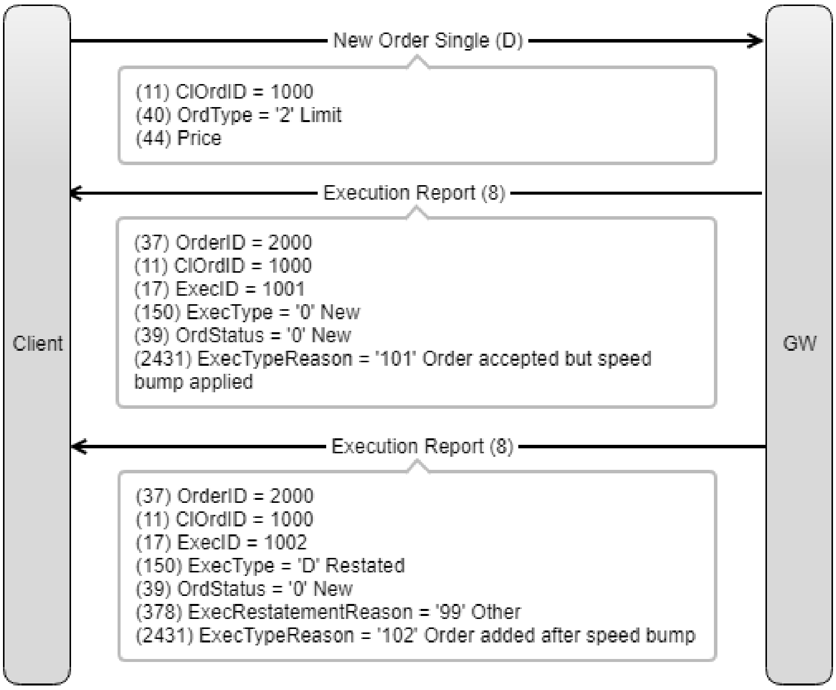
*108 = Order rejected after speed bump delay*

*109 = Unsolicited cancel while in speed bump*

***Order submission is speed bumped***

*If an order is submitted but is subject to a speed bump, the order is held and not added to the order book until the order has been released from the speed bump. The Execution Report sent in acknowledgement includes an ExecTypeReason (2431) = '101' Order accepted but speed bump applied.*

*The Execution Report sent once the order has cleared the speed bump and is added to the order book includes ExecType (150) = 'D' Restated and ExecTypeReason (2431) = '102' Order added after speed bump.*



New Order Single (D)

(11) ClOrdID = 1000
(40) OrdType = '2' Limit
(44) Price

Execution Report (8)

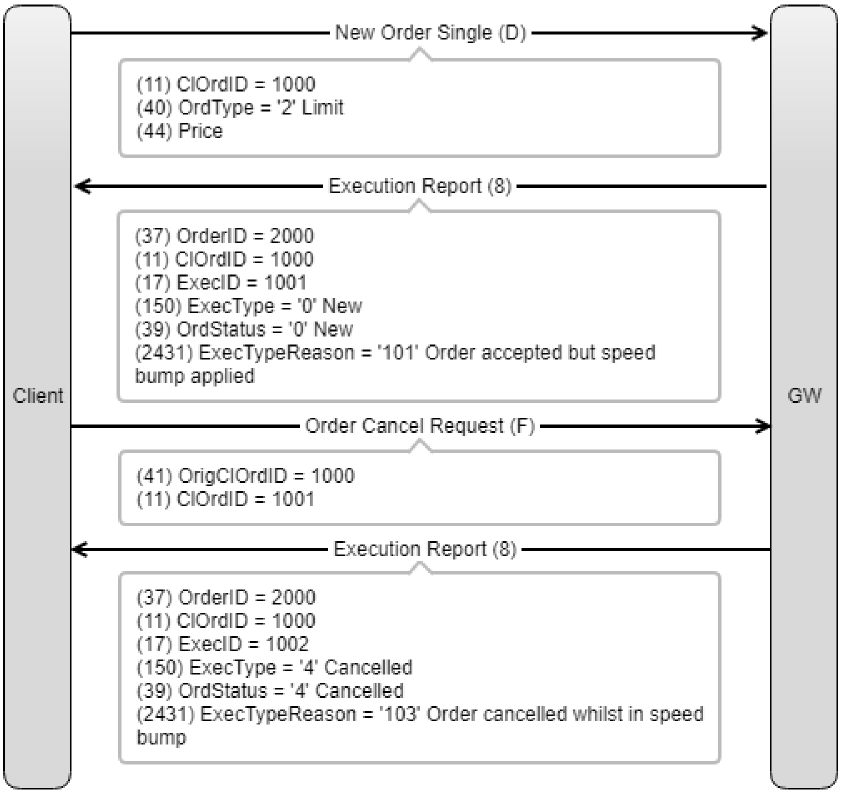
(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1001
(150) ExecType = '0' New
(39) OrdStatus = '0' New
(2431) ExecTypeReason = '101' Order accepted but speed bump applied

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1002
(150) ExecType = 'D' Restated
(39) OrdStatus = '0' New
(378) ExecRestatementReason = '99' Other
(2431) ExecTypeReason = '102' Order added after speed bump

***Order cancellation for a speed bumped order***

*An order cancellation submitted while an order is in the speed bump will be processed without any delay as the Order Cancel Request is not subject to speed bump conditions. The Execution Report sent in response to the cancellation includes ExecTypeReason (2431) = '103' Order cancelled whilst in speed bump delay.*

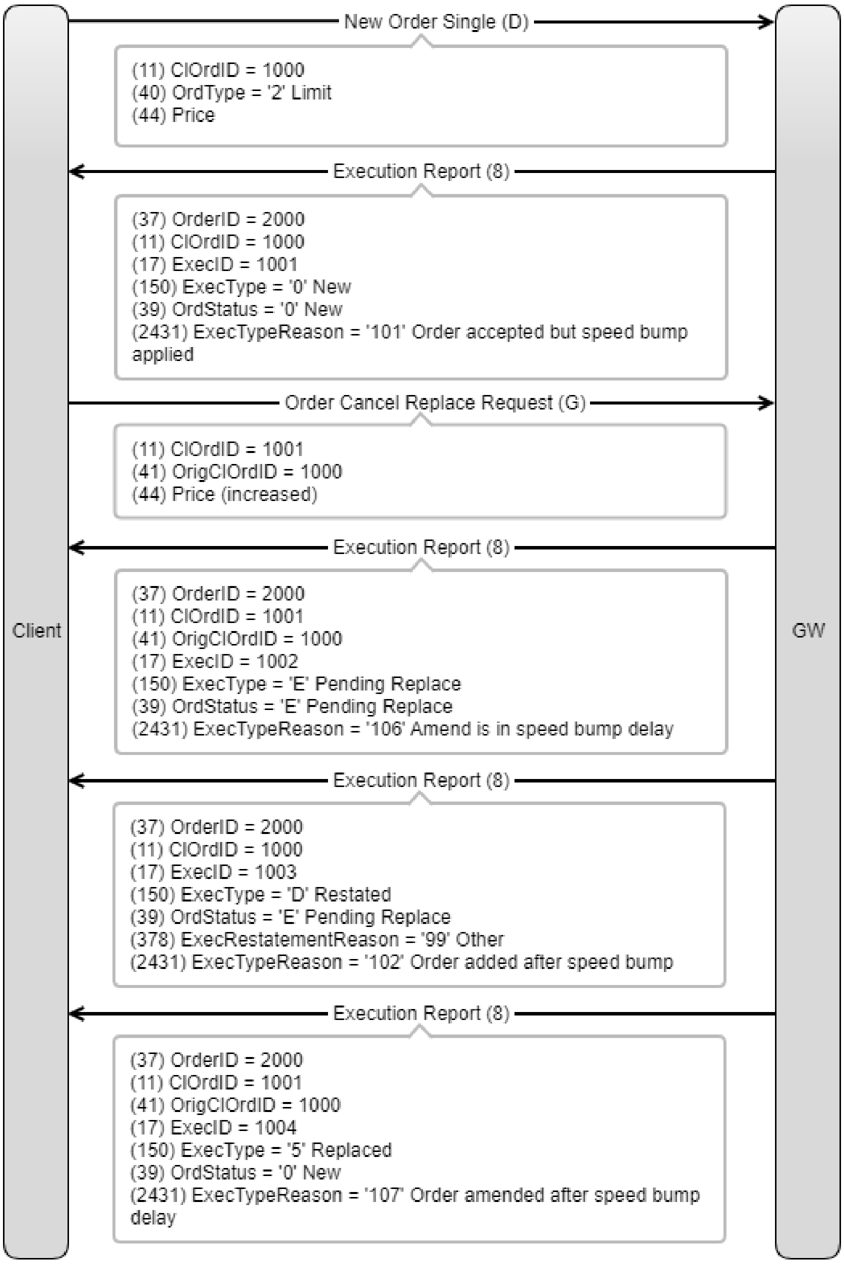


***Executable order amendment for a speed bumped order will be speed bumped***

*An order is submitted which is subject to a speed bump. An Order Cancel Replace Request is submitted while the order submission is in the speed bump queue. The amended order is executable and therefore speed bumped. The Execution Report for the order amendment includes ExecTypeReason (2431) = ‘106’ Amend is in speed bump delay. The Order Cancel Replace Request will not be processed until the original order has cleared the speed bump.*

*The Execution Report sent when the original order submission is released from the speed bump and added to the order book includes ExecType (150) = ‘D’ Restated, OrdStatus (39) = ‘E’ Pending Replace and ExecTypeReason (2431) = ‘102’ Order added after speed bump.*

*Another Execution Report is sent when the order amendment clears the speed bump and replaces the original order. The Execution Report includes ExecType (150) = ‘5’ Replaced and ExecTypeReason (2431) = ‘107’ Order amended after speed bump delay.*

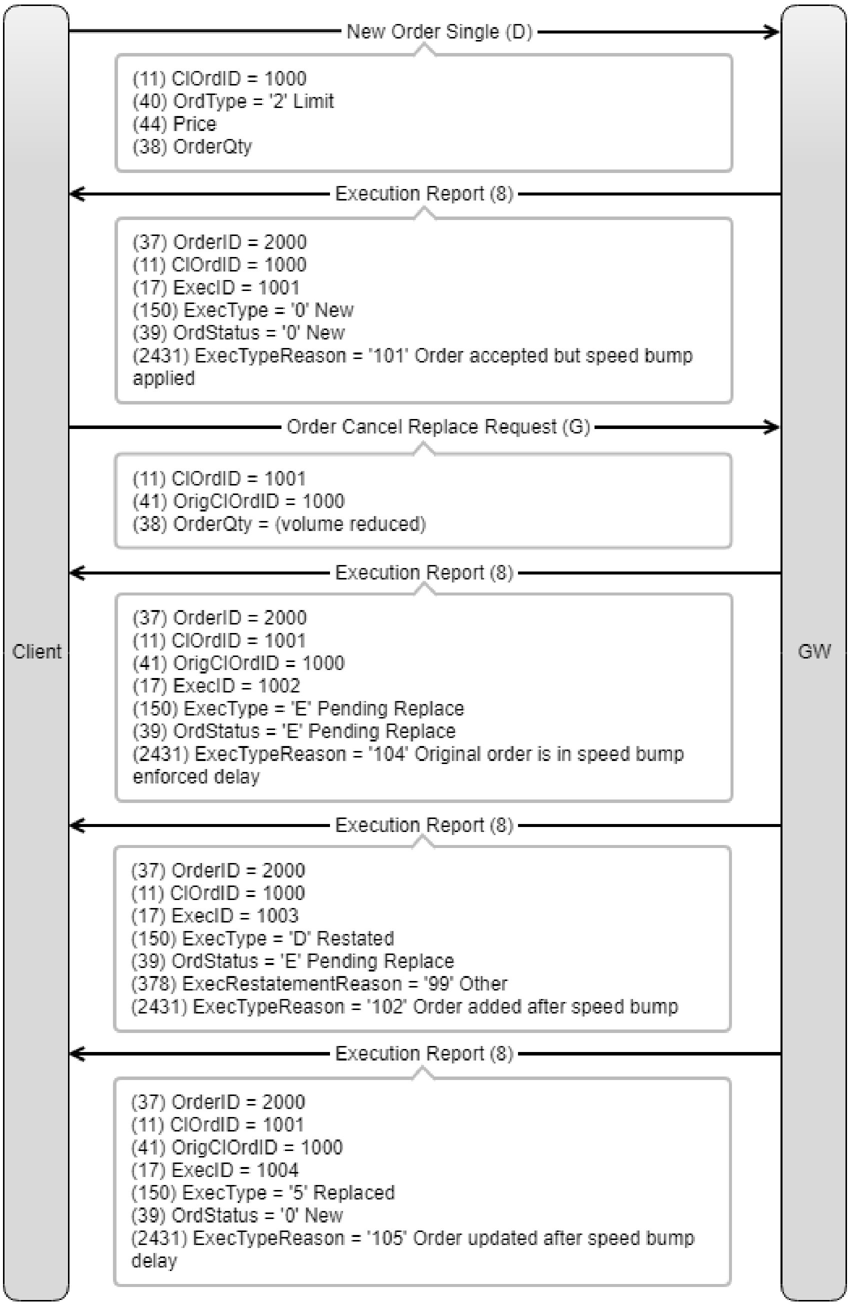


### *Non-executable order amendment for a speed bumped order will not be speed bumped*

*An order is submitted which is subject to a speed bump. An Order Cancel Replace Request is submitted while the order submission is in the speed bump queue. The amended order will rest in the order book and is therefore not subject to speed bump conditions. The Order Cancel Replace Request will not be processed until the original order has cleared the speed bump therefore the Execution Report for the amendment includes ExecType (150) = 'E' Pending Replace and ExecTypeReason (2431) = '104' Original order is in speed bump enforced delay.*

*The Execution Report sent once the order submission has cleared the speed bump and is added to the order book includes ExecType (150) = 'D' Restated and ExecTypeReason (2431) = '102' Order added after speed bump.*

*When the order is replaced the Execution Report includes ExecType (150) = ‘5’ Replaced and ExecTypeReason (2431) = ‘105’ Order updated after speed bump delay.*



New Order Single (D)

(11) ClOrdID = 1000
(40) OrdType = '2' Limit
(44) Price
(38) OrderQty

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1001
(150) ExecType = '0' New
(39) OrdStatus = '0' New
(2431) ExecTypeReason = '101' Order accepted but speed bump applied

Order Cancel Replace Request (G)

(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(38) OrderQty = (volume reduced)

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(17) ExecID = 1002
(150) ExecType = 'E' Pending Replace
(39) OrdStatus = 'E' Pending Replace
(2431) ExecTypeReason = '104' Original order is in speed bump enforced delay

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1003
(150) ExecType = 'D' Restated
(39) OrdStatus = 'E' Pending Replace
(378) ExecRestatementReason = '99' Other
(2431) ExecTypeReason = '102' Order added after speed bump

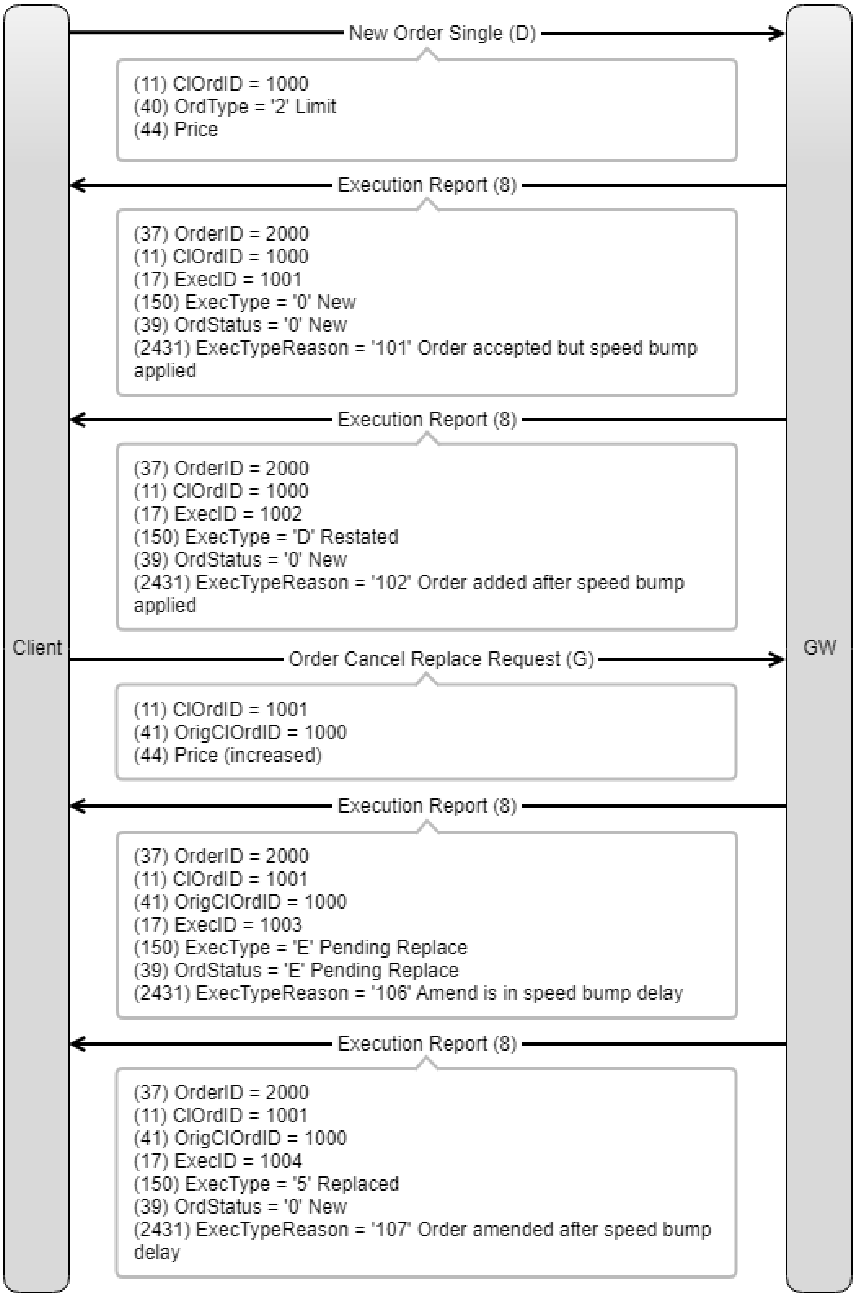
Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(17) ExecID = 1004
(150) ExecType = '5' Replaced
(39) OrdStatus = '0' New
(2431) ExecTypeReason = '105' Order updated after speed bump delay

***Executable order amendment for a resting order will be speed bumped***

*An order amendment is submitted for a resting order that was previously speed bumped. The Order Cancel Replace Request is speed bumped as the amended order will not provide liquidity. The Execution Report for the amendment includes ExecType (150) = ‘E’ Pending Replace with and ExecTypeReason (2431) = ‘106’ Amend is in speed bump delay.*

*When the order is replaced the Execution Report includes ExecType (150) = ‘5’ Replaced and ExecTypeReason (2431) = ‘107’ Order amended after speed bump delay.*



New Order Single (D)

(11) ClOrdID = 1000
(40) OrdType = '2' Limit
(44) Price

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1001
(150) ExecType = '0' New
(39) OrdStatus = '0' New
(2431) ExecTypeReason = '101' Order accepted but speed bump applied

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1002
(150) ExecType = 'D' Restated
(39) OrdStatus = '0' New
(2431) ExecTypeReason = '102' Order added after speed bump applied

Order Cancel Replace Request (G)

(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(44) Price (increased)

Execution Report (8)

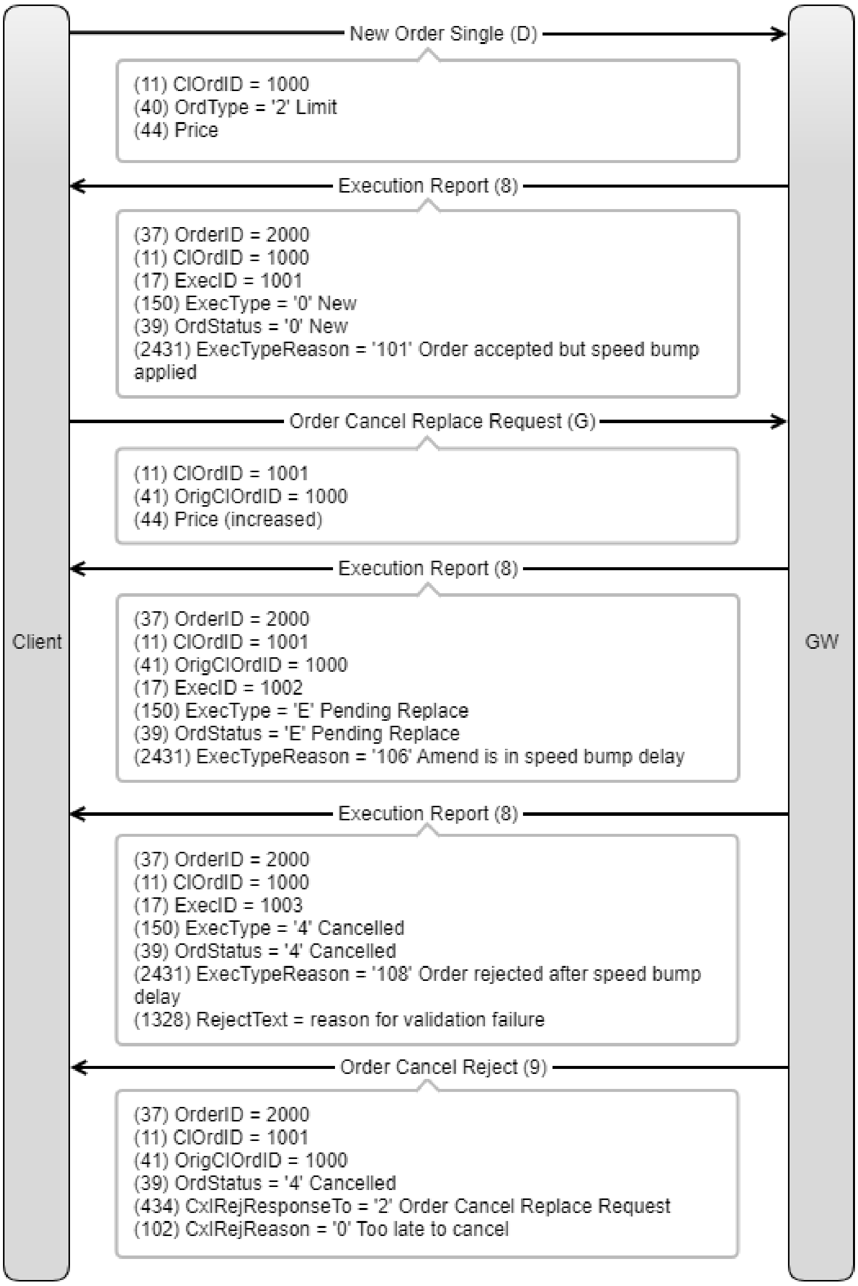
(37) OrderID = 2000
(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(17) ExecID = 1003
(150) ExecType = 'E' Pending Replace
(39) OrdStatus = 'E' Pending Replace
(2431) ExecTypeReason = '106' Amend is in speed bump delay

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(17) ExecID = 1004
(150) ExecType = '5' Replaced
(39) OrdStatus = '0' New
(2431) ExecTypeReason = '107' Order amended after speed bump delay

### *Speed bumped order is cancelled due to validation failure (inflight speed bumped amendment is also cancelled)*

*An order is submitted which is subject to a speed bump. An Order Cancel Replace is accepted which is also subject to speed bump conditions. The original order submission fails business validation on clearing the speed bump and is cancelled. The Execution Report includes ExecType (150) = ‘4’ Cancelled and ExecTypeReason (2431) = ‘108’ Order rejected after speed bump delay with the reason for the business validation failure in RejectText (1328). An Order Cancel Reject is sent for the order amendment.*



New Order Single (D)

(11) ClOrdID = 1000
(40) OrdType = '2' Limit
(44) Price

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1001
(150) ExecType = '0' New
(39) OrdStatus = '0' New
(2431) ExecTypeReason = '101' Order accepted but speed bump applied

Order Cancel Replace Request (G)

(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(44) Price (increased)

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(17) ExecID = 1002
(150) ExecType = 'E' Pending Replace
(39) OrdStatus = 'E' Pending Replace
(2431) ExecTypeReason = '106' Amend is in speed bump delay

Execution Report (8)

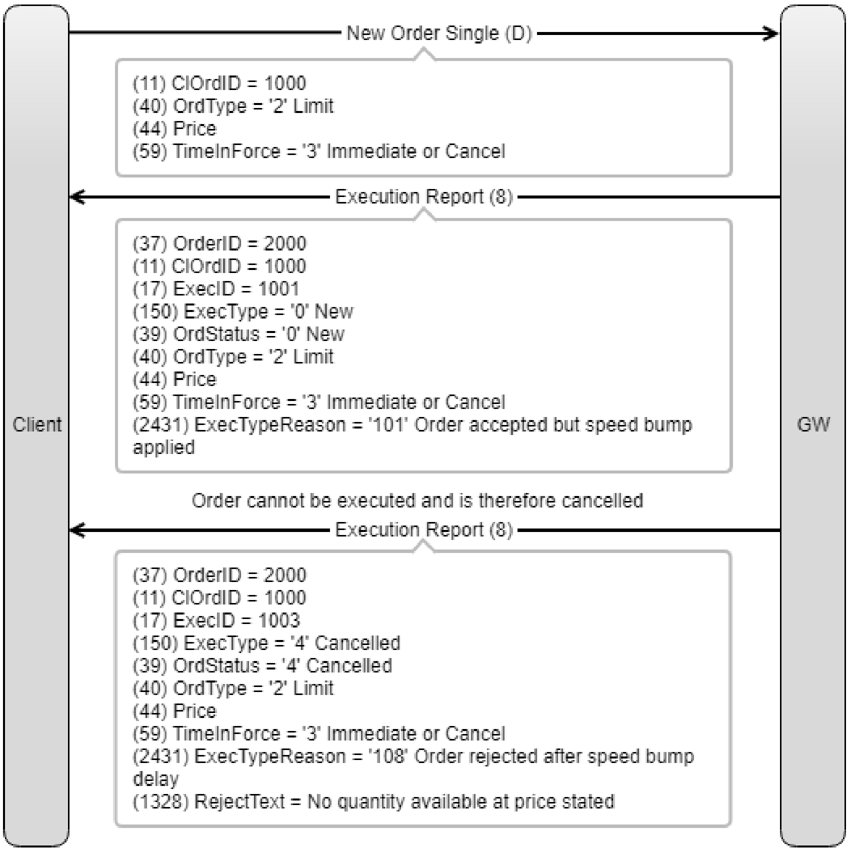
(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1003
(150) ExecType = '4' Cancelled
(39) OrdStatus = '4' Cancelled
(2431) ExecTypeReason = '108' Order rejected after speed bump delay
(1328) RejectText = reason for validation failure

Order Cancel Reject (9)

(37) OrderID = 2000
(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(39) OrdStatus = '4' Cancelled
(434) CxlRejResponseTo = '2' Order Cancel Replace Request
(102) CxlRejReason = '0' Too late to cancel

### *Immediate or Cancel speed bumped order fails validation and is cancelled*

*An Immediate or Cancel order is submitted which is subject to a speed bump. The order fails validation on clearing the speed bump as it cannot be executed and is therefore cancelled. The Execution Report includes ExecType (150) = '4' Cancelled and ExecTypeReason (2431) = '108' Order rejected after speed bump delay with the reason in RejectText (1328) = No quantity available at price stated.*



(11) ClOrdID = 1000
(40) OrdType = '2' Limit
(44) Price
(59) TimeInForce = '3' Immediate or Cancel

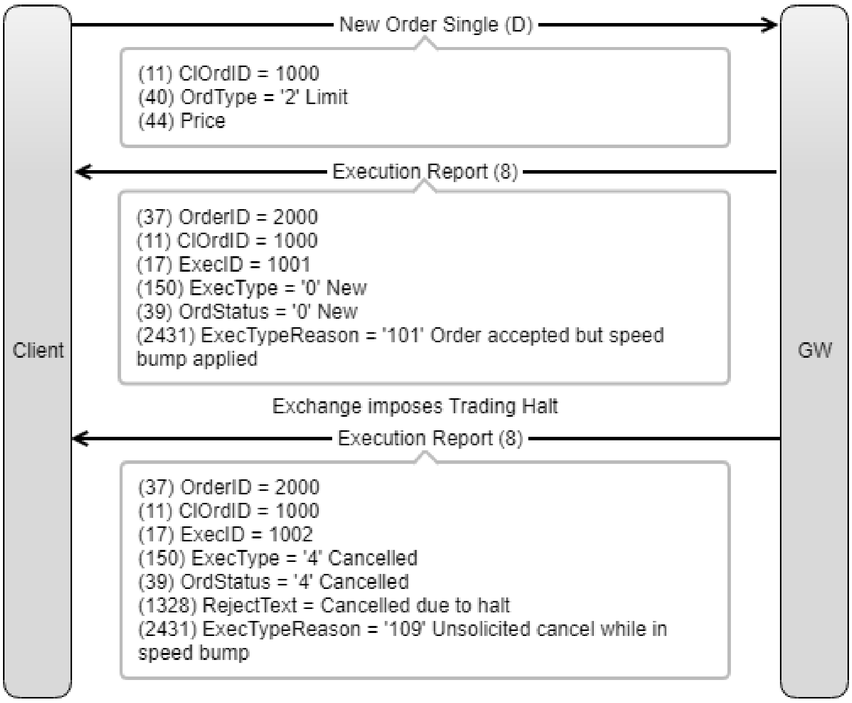
(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1001
(150) ExecType = '0' New
(39) OrdStatus = '0' New
(40) OrdType = '2' Limit
(44) Price
(59) TimeInForce = '3' Immediate or Cancel
(2431) ExecTypeReason = '101' Order accepted but speed bump applied

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1003
(150) ExecType = '4' Cancelled
(39) OrdStatus = '4' Cancelled
(40) OrdType = '2' Limit
(44) Price
(59) TimeInForce = '3' Immediate or Cancel
(2431) ExecTypeReason = '108' Order rejected after speed bump delay
(1328) RejectText = No quantity available at price stated

*Note: In the absence of a speed bump an IOC will be rejected if no quantity is available at the price stated.*

### *Unsolicited order cancellation while in speed bump*

*An order is submitted which is speed bumped. While the order is in the speed bump, the Exchange invokes a Trading Halt and all orders are pulled. The Execution Report sent for the order in the speed bump includes ExecType (150) = '4' Cancelled with ExecTypeReason (2431) = '109' Unsolicited cancel while in speed bump and RejectText (1328) = Cancelled due to halt.*



## 3.16 Message Throttling

The Exchange imposes a message throttle which limits the maximum number of order submissions, amends, *Quote Requests (35=R)* and Security Definition Requests (35=c) that can be submitted per second by a FIX Comp ID. Messages submitted in excess of the throttle limit in any given whole second will result in those messages being rejected by the gateway and will be notified by a Business Message Reject (35=j).

Security Definition Requests are included in the message throttle but also have their own throttle limits.

Note, order cancellation messages are exempt from throttling.

A system protection throttle will disconnect a user if the incoming message volume exceeds a multiple of the threshold limit. Reconnection is permitted after a second.

## 3.17 Security Definition Throttle

The number of Security Definition Requests (35=c) that can be submitted by a FIX Comp ID are set at per day rate and also included in the per second message throttle. A user breaching the daily limit will have further message submissions rejected by the gateway.

## 3.18 Merged Order Books

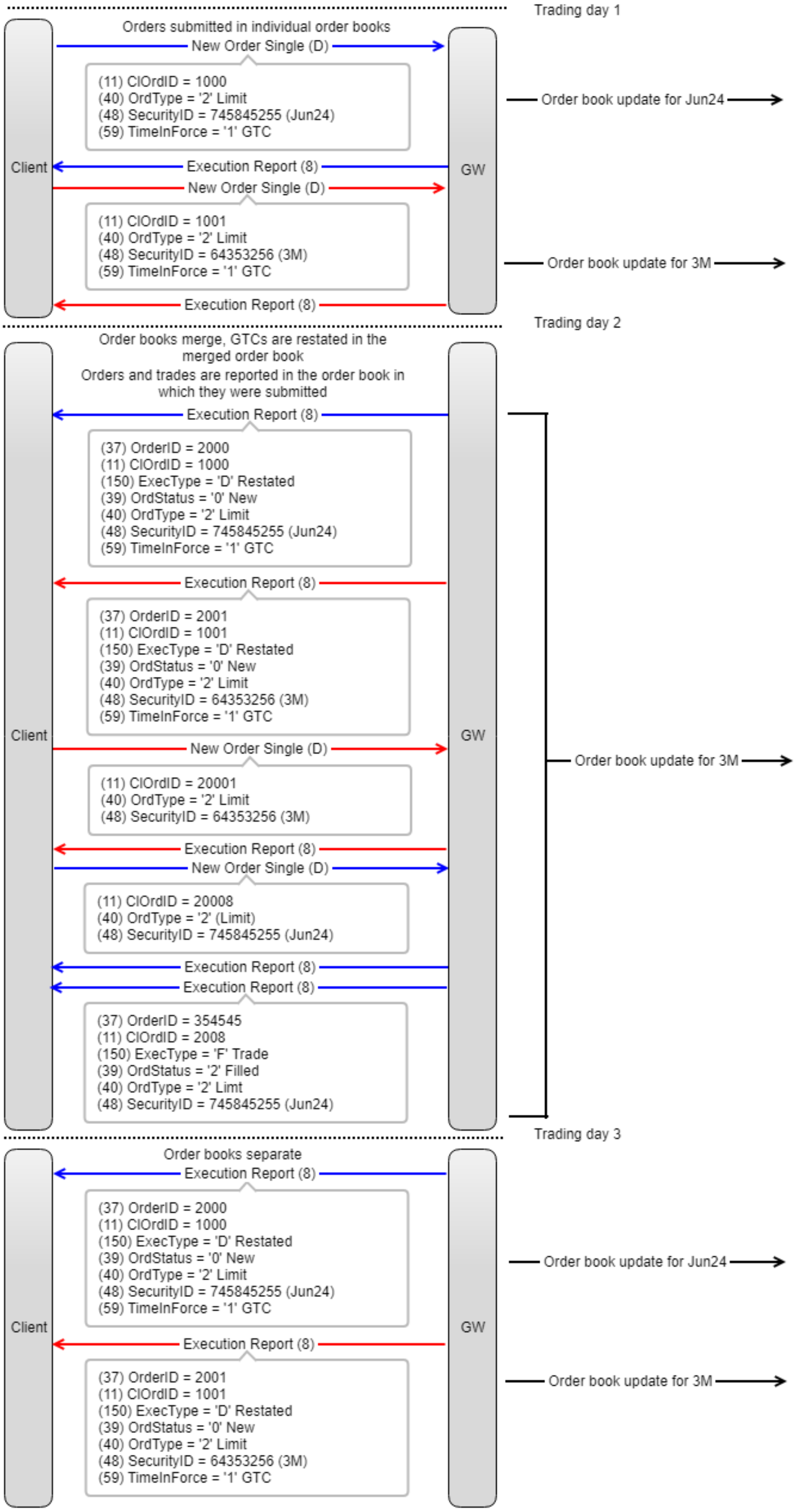
The LME prompt date structure for futures is such that two different prompts can share the same actual date on specific trading dates, for example, on the 3rd Wednesday of a month a 3M rolling prompt date will have the same prompt date as the monthly prompt date. On the trading date on which the prompts share the same actual date prompt, the order books for both prompts will be merged. TOM and Cash prompts will never merge.

Strategy order books that include a rolling leg will also merge. This can occur if a leg or legs share the same actual prompt date.

The merging of order books only affects execution and market data publication. Prompt dates in the merged order book will be available for order entry. The instrument identifier of the rolling prompt will be used by the Market Data service.

GTC and GTD orders will be merged into the order book with precedence and will return to the order book into which they were entered when the order books are no longer merged.

A mass cancellation request for a tradable instrument will not result in the cancellation of any orders in a merged tradable instrument. Orders will only be cancelled in the SecurityID specified in the Order Mass Cancel Request.


Trading day 1
Orders submitted in individual order books
Client
GW
New Order Single (D)
(11) ClOrdID = 1000
(40) OrdType = '2' Limit
(48) SecurityID = 745845255 (Jun24)
(59) TimeInForce = '1' GTC
Order book update for Jun24
Execution Report (8)
New Order Single (D)
(11) ClOrdID = 1001
(40) OrdType = '2' Limit
(48) SecurityID = 64353256 (3M)
(59) TimeInForce = '1' GTC
Order book update for 3M
Execution Report (8)
Trading day 2
Order books merge, GTCs are restated in the merged order book
Orders and trades are reported in the order book in which they were submitted
Client
GW
Execution Report (8)
(37) OrderID = 2000
(11) ClOrdID = 1000
(150) ExecType = 'D' Restated
(39) OrdStatus = '0' New
(40) OrdType = '2' Limit
(48) SecurityID = 745845255 (Jun24)
(59) TimeInForce = '1' GTC
Execution Report (8)
(37) OrderID = 2001
(11) ClOrdID = 1001
(150) ExecType = 'D' Restated
(39) OrdStatus = '0' New
(40) OrdType = '2' Limit
(48) SecurityID = 64353256 (3M)
(59) TimeInForce = '1' GTC
Order book update for 3M
New Order Single (D)
(11) ClOrdID = 20001
(40) OrdType = '2' Limit
(48) SecurityID = 64353256 (3M)
Execution Report (8)
New Order Single (D)
(11) ClOrdID = 20008
(40) OrdType = '2' (Limit)
(48) SecurityID = 745845255 (Jun24)
Execution Report (8)
Execution Report (8)
(37) OrderID = 354545
(11) ClOrdID = 2008
(150) ExecType = 'F' Trade
(39) OrdStatus = '2' Filled
(40) OrdType = '2' Limt
(48) SecurityID = 745845255 (Jun24)
Trading day 3
Order books separate
Client
GW
Execution Report (8)
(37) OrderID = 2000
(11) ClOrdID = 1000
(150) ExecType = 'D' Restated
(39) OrdStatus = '0' New
(40) OrdType = '2' Limit
(48) SecurityID = 745845255 (Jun24)
(59) TimeInForce = '1' GTC
Order book update for Jun24
Execution Report (8)
(37) OrderID = 2001
(11) ClOrdID = 1001
(150) ExecType = 'D' Restated
(39) OrdStatus = '0' New
(40) OrdType = '2' Limit
(48) SecurityID = 64353256 (3M)
(59) TimeInForce = '1' GTC
Order book update for 3M

## *3.19 Self Execution Prevention (SEP)*

*A member can guard against traders in their organisation executing orders with each other.*

*A member can use SEP functionality without configuring a SEP handling action in which case the Exchange configured response type would be triggered to cancel the incoming order. Alternatively a member can configure SEP identifiers and specify the action to be taken if two orders with an identical SEP ID could execute.*

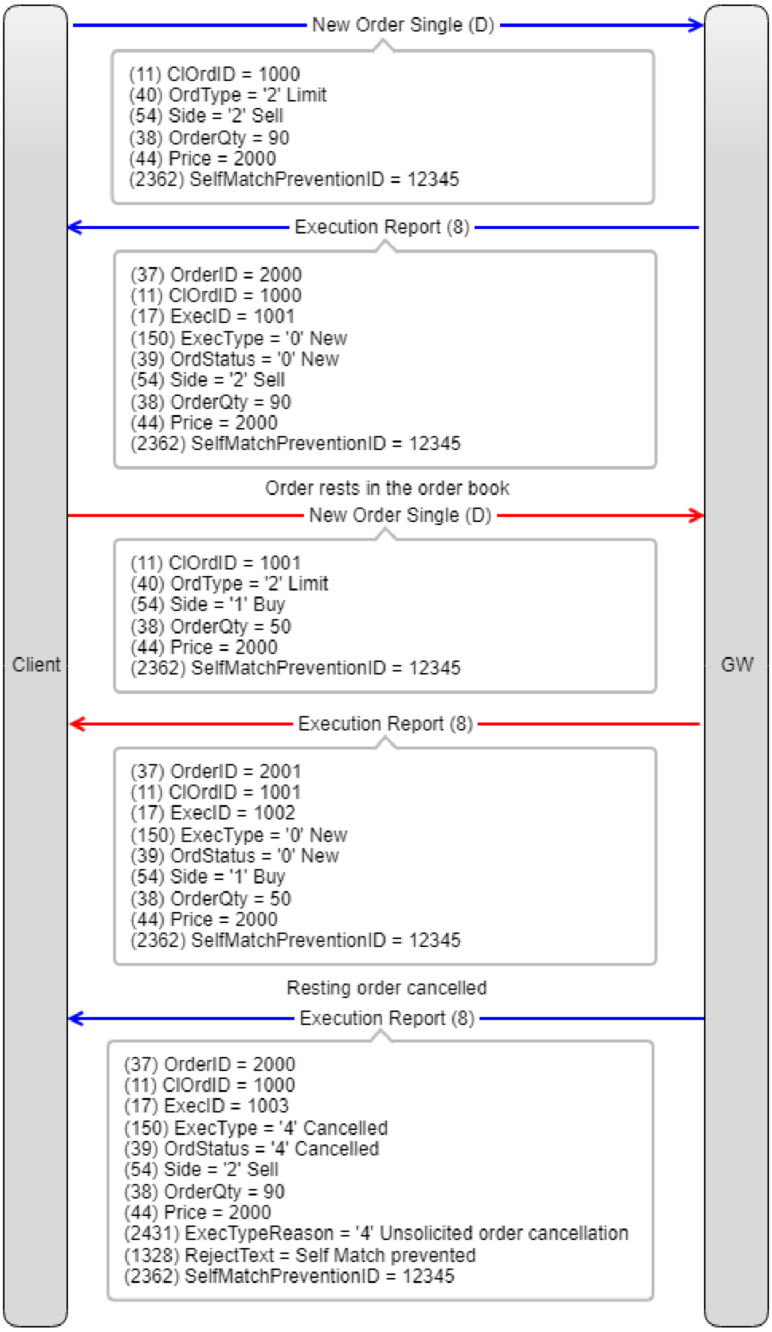
*A SEP ID will be specified as a maximum of 9 digits. A member will use a Party Entitlements Definition Request (35=DA) submitted via the Risk Management Gateway to define the SEP configuration as described in the Risk Management Gateway FIX Specification. This configuration will be effective from the next trading day.*

*A SEP ID can be entered in the SelfMatchPreventionID (2362) on order submission. If orders with an identical SEP ID from the same member firm can cross, the SEP handling action that has been configured is triggered to cancel either the incoming or resting order or both (incoming and resting).*

*The Execution Report sent for the cancelled order will contain RejectText (1328) = Self Match prevented.*

*The availability of SEP functionality will be determined by the Exchange. If an order is submitted with the SelfMatchPreventionID (2362) populated and SEP is not available for the SecurityID (48) specified, the order will be rejected. The Execution Report sent will contain RejectText (1328) = Self Match Prevention not configured for the tradable instrument.*

*Self Execution Prevention triggered – resting order cancelled*



New Order Single (D)

(11) ClOrdID = 1000
(40) OrdType = '2' Limit
(54) Side = '2' Sell
(38) OrderQty = 90
(44) Price = 2000
(2362) SelfMatchPreventionID = 12345

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1001
(150) ExecType = '0' New
(39) OrdStatus = '0' New
(54) Side = '2' Sell
(38) OrderQty = 90
(44) Price = 2000
(2362) SelfMatchPreventionID = 12345

Order rests in the order book

New Order Single (D)

(11) ClOrdID = 1001
(40) OrdType = '2' Limit
(54) Side = '1' Buy
(38) OrderQty = 50
(44) Price = 2000
(2362) SelfMatchPreventionID = 12345

Execution Report (8)

(37) OrderID = 2001
(11) ClOrdID = 1001
(17) ExecID = 1002
(150) ExecType = '0' New
(39) OrdStatus = '0' New
(54) Side = '1' Buy
(38) OrderQty = 50
(44) Price = 2000
(2362) SelfMatchPreventionID = 12345

Resting order cancelled

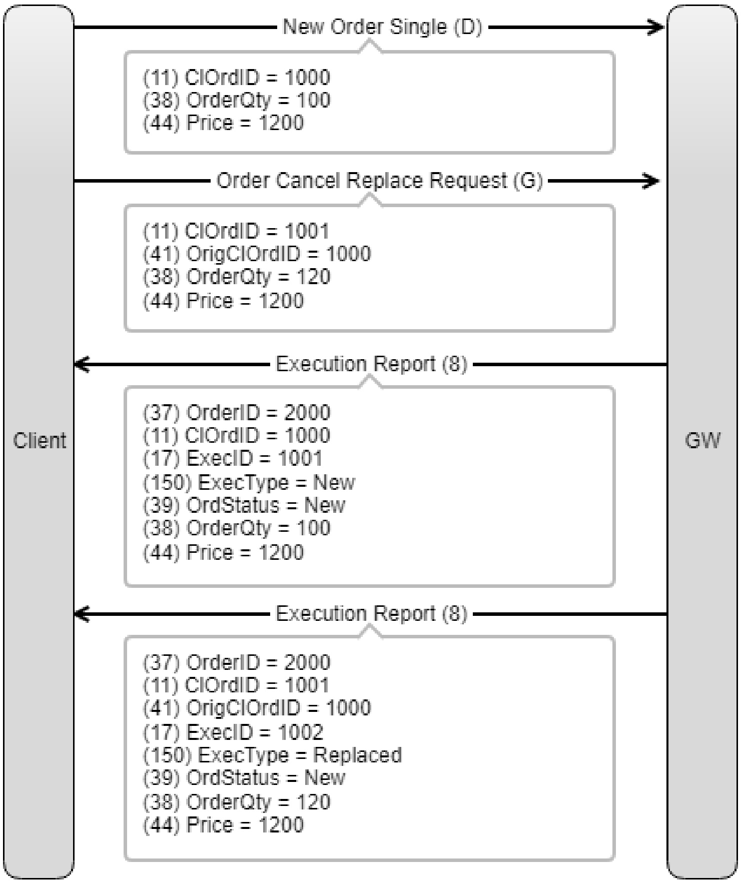
Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1003
(150) ExecType = '4' Cancelled
(39) OrdStatus = '4' Cancelled
(54) Side = '2' Sell
(38) OrderQty = 90
(44) Price = 2000
(2431) ExecTypeReason = '4' Unsolicited order cancellation
(1328) RejectText = Self Match prevented
(2362) SelfMatchPreventionID = 12345

## 3.20 Inflight Order Processing

The gateway will accept a single inflight amend or cancellation request whilst processing a new order. The amend request is queued until the preceding request has been processed. Multiple inflight messages will be rejected.

For example, a New Order Single is submitted followed immediately afterwards by an Order Cancel Replace Request. An Execution Report is returned for the order submission and then the amendment.



New Order Single (D)

(11) ClOrdID = 1000
(38) OrderQty = 100
(44) Price = 1200

Order Cancel Replace Request (G)

(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(38) OrderQty = 120
(44) Price = 1200

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1001
(150) ExecType = New
(39) OrdStatus = New
(38) OrderQty = 100
(44) Price = 1200

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1001
(41) OrigClOrdID = 1000
(17) ExecID = 1002
(150) ExecType = Replaced
(39) OrdStatus = New
(38) OrderQty = 120
(44) Price = 1200

See Appendix A: Inflight Order Handling and Appendix B: Speed Bump Inflight Order Handling for more examples.

## 3.21 Trade Reporting

When an outright order matches the trade half will be assigned an identifier which will be reported in the TrdMatchID (880) on the Execution Report (35=8). A strategy trade will be reported in a single Execution Report including the leg details. The legs of strategy trade will be assigned a LegAllocID (1366) which will be shared with either the LegAllocID of another strategy trade or the TrdMatchID of an outright trade.

## 3.22 Order Attribute Type Usage

The OrderAttributeGrp is used to specify additional attributes on the order. The following values are supported for OrderAttributeType (2594):

0 = Aggregated order

1 = Pending allocation

2 = Liquidity provision order

3 = Risk reduction order.

OrderAttributeValue (2595) will always be set to Y.

An order will be rejected if a client identifier or a value of 0 to indicate no client is submitted for the client short code i.e. PartyIDSource (447) = P and PartyRole (452) = ‘3’ Client ID for an order specified as either 0 = Aggregated order or 1 = Pending allocation.

An order will also be rejected that specifies both 0 = Aggregated order and 1 = Pending allocation.

# 4 Message Definitions

## 4.1 Supported Messages

| Message |
|---|
| **Logon (A)**<br>Establishes a FIX session |
| **Heartbeat (0)**<br>Used to check connectivity |
| **Test Request (1)**<br>Used to verify a connection is still active |
| **Resend Request (2)**<br>Request the retransmission of messages |
| **Reject (3)**<br>Issued when a message is received but cannot be properly processed due to a session-level rule violation |
| **Sequence Reset (4)**<br>Indicates there is a gap in the message sequence numbers |
| **Logout (5)**<br>Terminates a FIX session |
| **Business Message Reject (j)**<br>Reject an application level message which fulfils session level rules |
| **News (B)**<br>Disseminates text information |
| **Security Definition Request (c)**<br>Request the creation of a tradable instrument |
| **Security Definition (d)**<br>Response to a Security Definition Request |

| Message |
|---|
| **New Order Single (D)**<br>Submit a new order for execution |
| **Order Cancel Replace Request (G)**<br>Amend an existing order |
| **Order Cancel Request (F)**<br>Request the cancellation of all the remaining quantity of an existing order |
| **Order Cancel Reject (9)**<br>Reports that an Order Cancel Request or Order Cancel Replace Request has been rejected |
| **Execution Report (8)**<br>Sent in response to order and fill related client messages |
| **Order Mass Cancel Request (q)**<br>Cancel multiple orders |
| **Order Mass Cancel Report (r)**<br>Acknowledgement of an Order Mass Cancel Request |
| ***Quote Request (R)***<br>*Requests prices from market participants* |
| ***Quote Response (AJ)***<br>*Acknowledgement of a Quote Request* |
| ***Quote Request Reject (AG)***<br>*Reports that a Quote Request has been rejected* |

## 4.2 Inbound Messages

- Logon (A)
- Heartbeat (0)
- Test Request (1)
- Resend Request (2)
- Sequence Reset (4)
- Logout (5)
- Security Definition Request (c)
- New Order Single (D)

- Order Cancel Replace Request (G)
- Order Cancel Request (F)
- Order Mass Cancel Request (q)
- *Quote Request (R)*

## 4.3 Outbound Messages

- Logon (A)
- Heartbeat (0)
- Test Request (1)
- Resend Request (2)
- Sequence Reset (4)
- Logout (5)
- Reject (3)
- Business Message Reject (j)
- News (B)
- Security Definition (d)
- Order Cancel Reject (9)
- Execution Report (8)
- Order Mass Cancel Report (r)
- *Quote Response (AJ)*
- *Quote Request Reject (AG)*

## 4.4 Data Types

Data types used are based on the published standard FIX specifications. The field length in characters is shown in brackets. The length of numeric fields is the number of digits in that value and not the size of the value in bytes. If a data type has a specific value this will be provided in the description.

| Data Type | Format |
|---|---|
| UTCTimestamp (27) | Incoming<br>YYYYMMDD-HH:mm:ss.SSSSSS<br>YYYYMMDD-HH:mm:ss.SSSSSSSSS<br>Timestamps will be represented as UTC and accepted to microsecond or nanosecond precision |
| | Outgoing<br>YYYYMMDD-HH:mm:ss.SSSSSSSSS<br>Note: Timestamps will be represented as UTC up to microsecond precision with the nanosecond element being represented by trailing zeros. |

| Data Type | Format |
|---|---|
| Price (20) | Can be up to 12 significant digits before the decimal point (with provision for a negative value) and at the most 6 decimal places<br>For example,<br>1234567891234.567891<br>-123456789123.456789 |
| String (*n*) | Permitted ASCII characters are A-Z, a-z, 0-9<br>ClOrdID (11), OrigClOrdID (41) and PartyID (448) for PartyRole (452) = ‘81’ Broker Client ID also permit hyphen (‘-‘) and underscore (‘_’). |

## 4.5 Required Fields

The following conventions are used for fields in the message definitions:

| | |
|---|---|
| Y | Required by FIX |
| Y* | Required by LME |
| C | Conditionally required by FIX |
| C* | Conditionally required by LME |
| N | Not required / optional. |

## 4.6 Message Header

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 8 | BeginString | Y | String (8) | Always set to FIXT.1.1 |
| 9 | BodyLength | Y | Length (4) | Message length, in bytes, forward to the CheckSum field.<br>Maximum value 9999 |
| 35 | MsgType | Y | String (3) | Defines message type. |
| 1128 | ApplVerID | N | String (1) | Version of FIX used in the message:<br>9 = FIX50SP2<br>Returned by the gateway |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 49 | SenderCompID | Y | String (10) | Identifies the sender of the message, see Comp ID |
| 56 | TargetCompID | Y | String (10) | Identifies the receiver of the message, see Comp ID |
| 34 | MsgSeqNum | Y | SeqNum (9) | Outbound message sequence number. Always incremented by the sender. |
| 43 | PossDupFlag | N | Boolean | Indicates whether the message was previously transmitted with the same MsgSeqNum (34).<br>Absence of this field is interpreted as original transmission (N). |
| 97 | PossResend | N | Boolean | Indicates whether the message was previously transmitted under a different MsgSeqNum (34).<br>Absence of this field is interpreted as original transmission (N). |
| 52 | SendingTime | Y | UTCTimestamp | Time the message was transmitted. |
| 122 | OrigSendingTime | C | UTCTimestamp | Conditionally required for messages sent as a result of a Resend Request (2).<br>If the original time is not available, this should be the same value as SendingTime (52). |

## 4.7 Message Trailer

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 10 | CheckSum | Y | String (7) | Standard check sum described by FIX protocol.<br>Always last field in the message; i.e. serves, with the trailing <SOH>, as the end-of-message delimiter.<br>Always defined as three characters. |

## 4.8 Administrative Messages

### 4.8.1 Logon (A)

The first messages exchanged in a FIX session are the Logon request and the Logon response. The main purposes of the Logon request and response are:

- To authenticate the client.
- To agree on the sequence numbers.

On initial logon the status of persisted orders is communicated to the FIX session by the publication of Execution Reports for all open orders.

The list of available tradable instruments for the current trading day will be published by the Market Data service independently of the FIX Logon request.

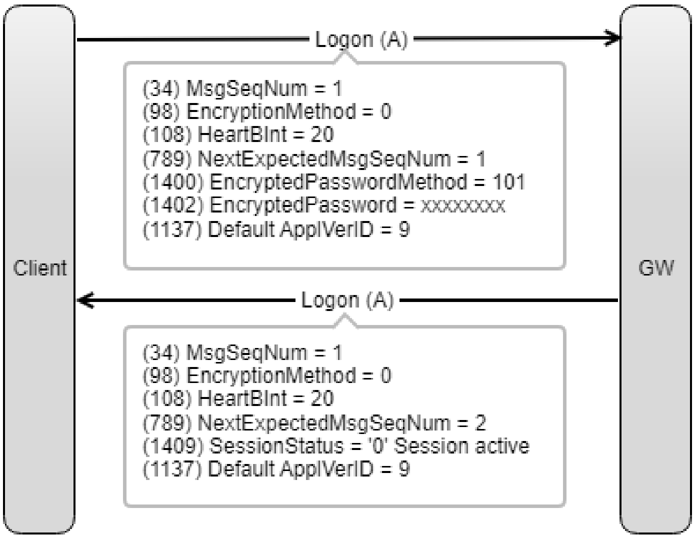
| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 98 | EncryptMethod | Y | Int | Method for encryption.<br>Valid value is: 0 = None |
| 108 | HeartBtInt | Y | Int | Heartbeat interval in seconds. |
| 789 | NextExpectedMsgSeqNum | Y* | SeqNum (9) | Next expected MsgSeqNum (34) value to be received. Always updated as a result of an incoming message. |
| 1400 | EncryptedPasswordMethod | N | Int | Enumeration defining the encryption method used to encrypt password fields:<br>101 = RSA |
| 1402 | EncryptedPassword | Y | Data (450) | Encrypted password – encrypted via the method specified in EncryptedPasswordMethod (1400) |
| 1404 | EncryptedNewPassword | N | Data (450) | Encrypted new password – encrypted via the method specified in EncryptedPasswordMethod (1400) |
| 1137 | DefaultApplVerID | Y | String (1) | The default version of FIX being used in this session:<br>9 = FIX50SP2 |

A Logon message is returned in response to an incoming Logon message to initiate a FIX session. The SessionStatus (1409) indicates whether the logon attempt was successful or not.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 98 | EncryptMethod | Y | Int | Method for encryption.<br>Valid value is: 0 = None |
| 108 | HeartBtInt | Y | Int | Heartbeat interval in seconds. |
| 789 | NextExpectedMsgSeqNum | Y* | SeqNum (9) | Next expected MsgSeqNum (34) value to be received. Always updated as a result of an incoming message. |
| 1409 | SessionStatus | N | Int | Status of the FIX session.<br>Valid values:<br>0 = Session active<br>1 = Session password changed |
| 1137 | DefaultApplVerID | Y | String (1) | The default version of FIX being used in this session:<br>9 = FIX50SP2 |

**Example Message Flow**

*Initial Logon*



(34) MsgSeqNum = 1
(98) EncryptionMethod = 0
(108) HeartBInt = 20
(789) NextExpectedMsgSeqNum = 1
(1400) EncryptedPasswordMethod = 101
(1402) EncryptedPassword = xxxxxxxx
(1137) Default ApplVerID = 9

(34) MsgSeqNum = 1
(98) EncryptionMethod = 0
(108) HeartBInt = 20
(789) NextExpectedMsgSeqNum = 2
(1409) SessionStatus = '0' Session active
(1137) Default ApplVerID = 9

### 4.8.2 Heartbeat (0)

Heartbeat (35=0) is sent at the interval specified in HeartBtInt (108) in Logon (35=A). It is also sent in response to a Test Request (35=1).

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 112 | TestReqID | C | String (20) | Conditionally required if the heartbeat is a response to a Test Request (1). The value in this field should echo the TestReqID (112) received in the Test Request. |

### 4.8.3 Test Request (1)

Test Request (35=1) can be sent by either the client or gateway to verify a connection is active. The recipient responds with a Heartbeat (35=0).

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 112 | TestReqID | Y | String (20) | Identifier included in Test Request message to be returned in resulting Heartbeat (0). |

### 4.8.4 Resend Request (2)

Resend Request (35=2) is used to initiate the retransmission of messages if a sequence number gap is detected.

To request a single message. The BeginSeqNo and EndSeqNo should be the same.

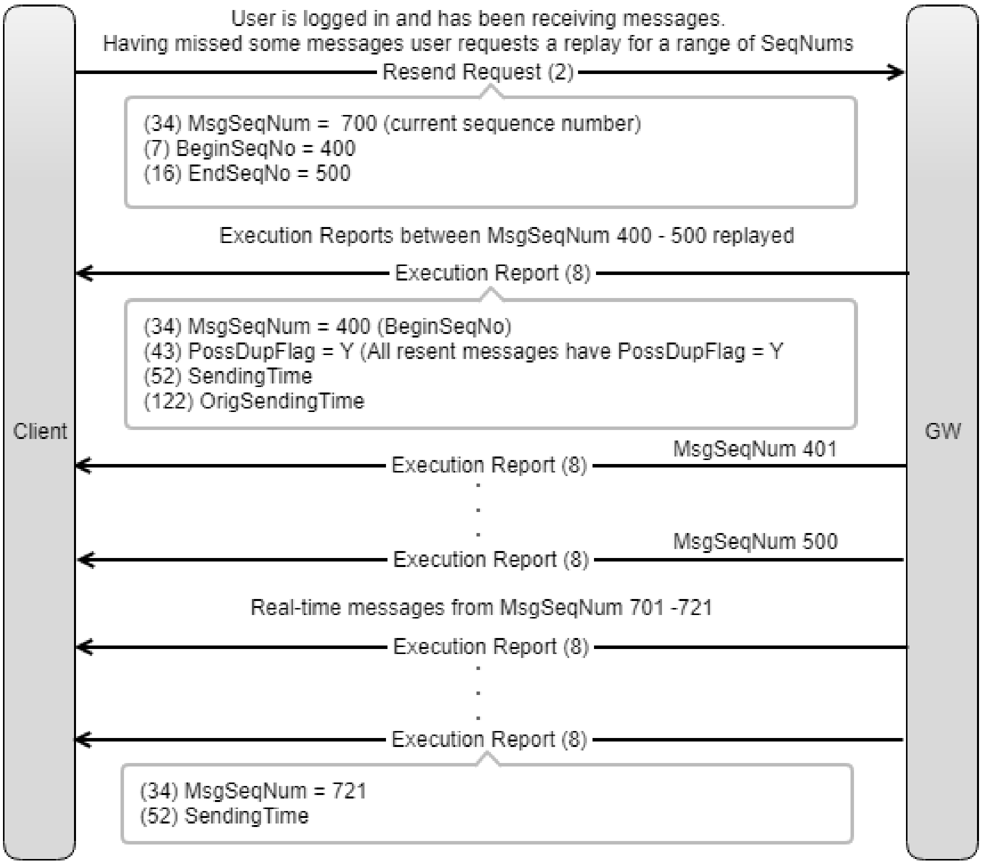
To request a specific range of messages. The BeginSeqNo should be the first message of the range and the EndSeqNo should be the last of the range.

To request all messages after a particular message. The BeginSeqNo should be the sequence number immediately after that of the last processed message and the EndSeqNo should be zero (0).

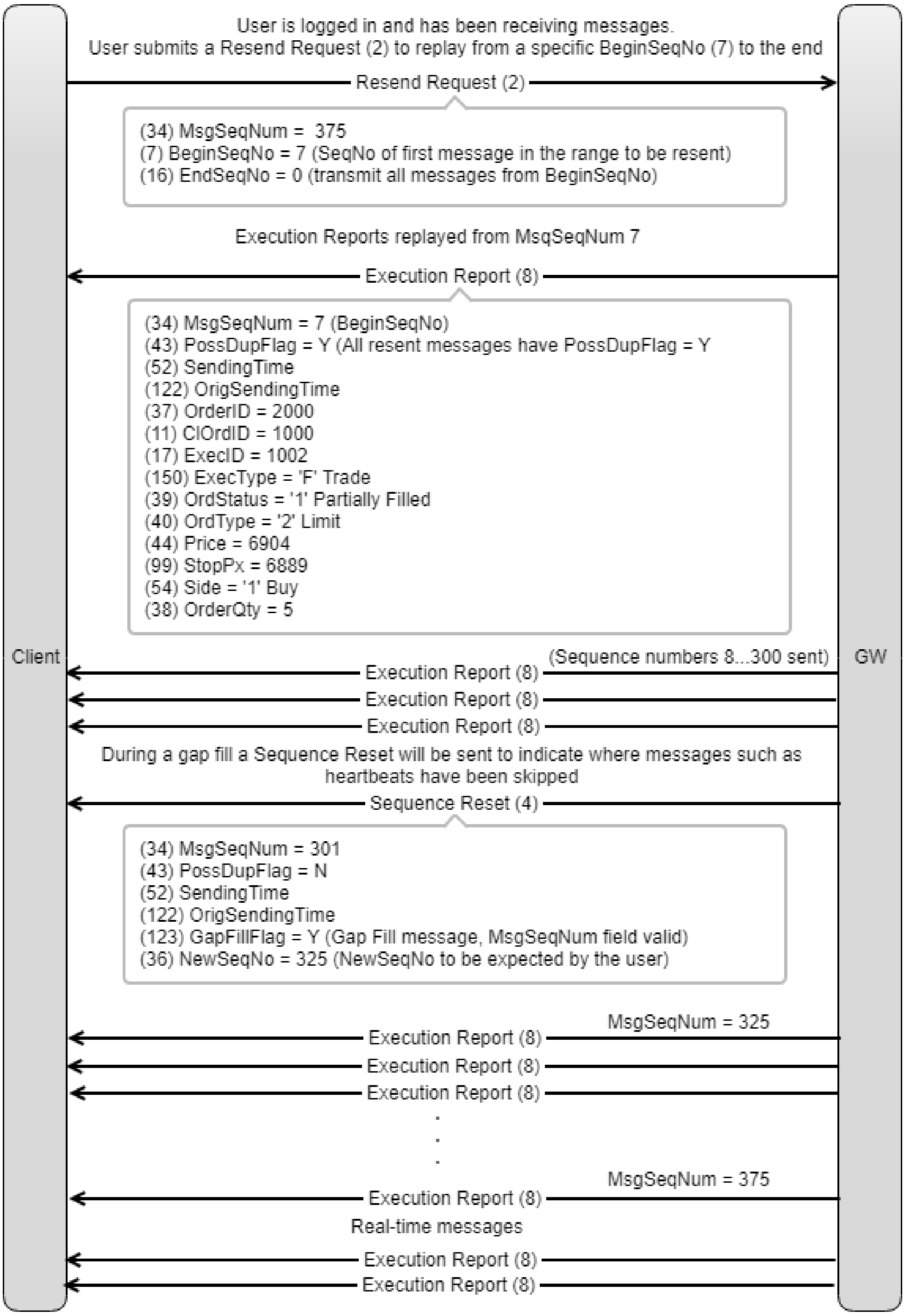
| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 7 | BeginSeqNo | Y | SeqNum (9) | Message sequence number of the first message in the range to be resent. |
| 16 | EndSeqNo | Y | SeqNum (9) | Sequence number of the last message expected to be resent.<br>This may be set to 0 to request the sender to transmit ALL messages starting from BeginSeqNo (7). |

### Example Message Flow

*Resend Request for a range of messages*


Client
GW
User is logged in and has been receiving messages.
Having missed some messages user requests a replay for a range of SeqNums
Resend Request (2)
(34) MsgSeqNum = 700 (current sequence number)
(7) BeginSeqNo = 400
(16) EndSeqNo = 500
Execution Reports between MsgSeqNum 400 - 500 replayed
Execution Report (8)
(34) MsgSeqNum = 400 (BeginSeqNo)
(43) PossDupFlag = Y (All resent messages have PossDupFlag = Y
(52) SendingTime
(122) OrigSendingTime
Execution Report (8)
MsgSeqNum 401
Execution Report (8)
MsgSeqNum 500
Real-time messages from MsgSeqNum 701 -721
Execution Report (8)
Execution Report (8)
(34) MsgSeqNum = 721
(52) SendingTime

*Resend Request for all messages after a particular message*



User is logged in and has been receiving messages.
User submits a Resend Request (2) to replay from a specific BeginSeqNo (7) to the end

Resend Request (2)

(34) MsgSeqNum = 375
(7) BeginSeqNo = 7 (SeqNo of first message in the range to be resent)
(16) EndSeqNo = 0 (transmit all messages from BeginSeqNo)

Execution Reports replayed from MsqSeqNum 7

Execution Report (8)

(34) MsgSeqNum = 7 (BeginSeqNo)
(43) PossDupFlag = Y (All resent messages have PossDupFlag = Y
(52) SendingTime
(122) OrigSendingTime
(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1002
(150) ExecType = 'F' Trade
(39) OrdStatus = '1' Partially Filled
(40) OrdType = '2' Limit
(44) Price = 6904
(99) StopPx = 6889
(54) Side = '1' Buy
(38) OrderQty = 5

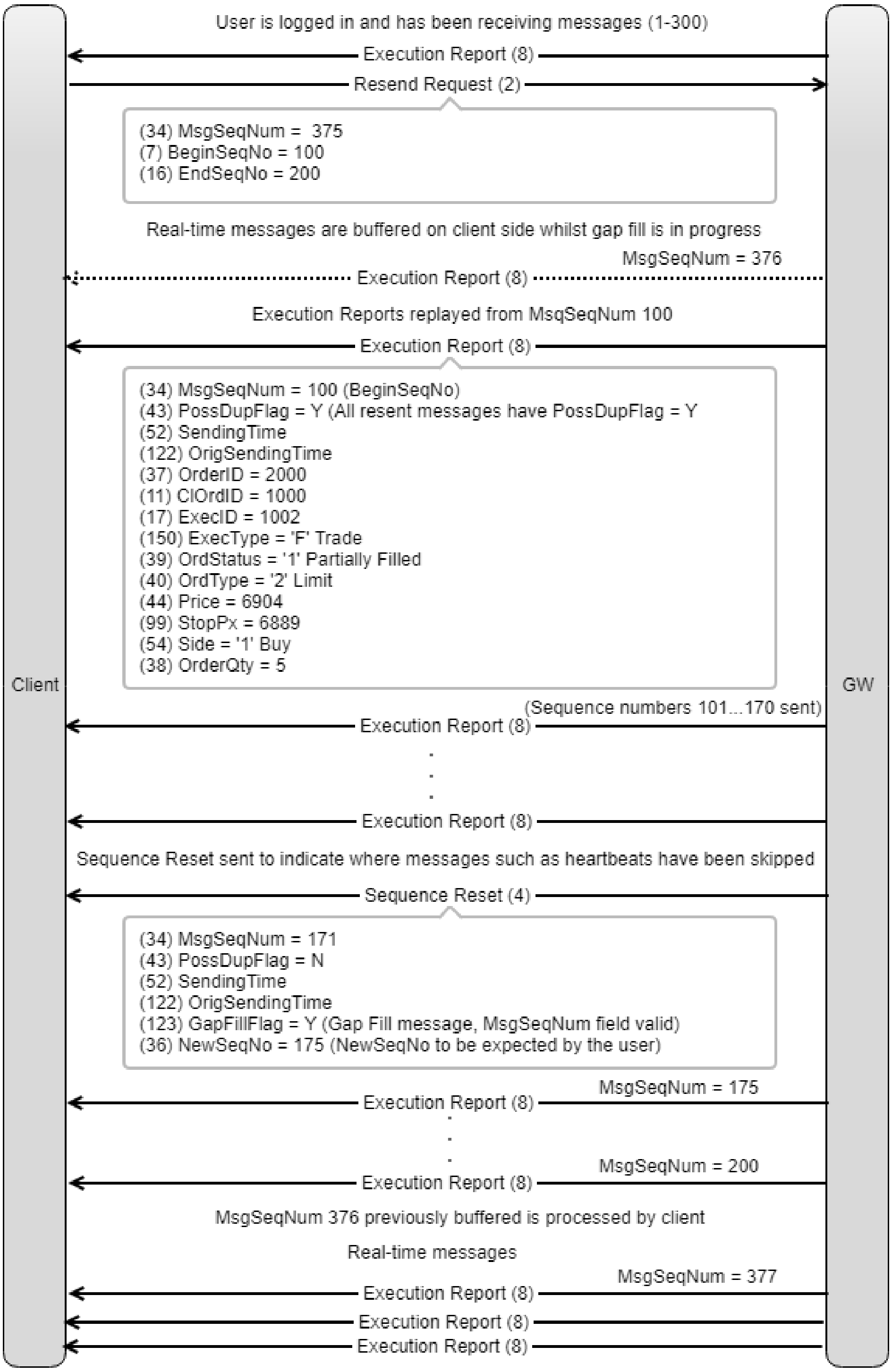
During a gap fill a Sequence Reset will be sent to indicate where messages such as heartbeats have been skipped

Sequence Reset (4)

(34) MsgSeqNum = 301
(43) PossDupFlag = N
(52) SendingTime
(122) OrigSendingTime
(123) GapFillFlag = Y (Gap Fill message, MsgSeqNum field valid)
(36) NewSeqNo = 325 (NewSeqNo to be expected by the user)

*Resend Request - incoming message buffered by Client*

A Resend Request is submitted but before gap fill messages have been transmitted an incoming message is received. The client will hold the message until all the gap fill messages have been received and then process the buffered message. All messages should be processed in sequence number order.



### 4.8.5 Sequence Reset (4)

Sequence Reset (35=4) allows the client or the gateway to increase the expected incoming sequence number of the other party.

In a Gap Fill it is sent as notification of the next sequence number to be transmitted.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 123 | GapFillFlag | N | Boolean | Indicates that the Sequence Reset message is replacing administrative or application messages which will not be resent.<br>Valid values:<br>Y = Gap Fill message, MsgSeqNum (34) field valid.<br>N = Sequence Reset, ignore MsgSeqNum (tag 34).<br>If omitted default value is N. |
| 36 | NewSeqNo | Y | SeqNum (9) | Sequence number of the next message to be transmitted. |

### 4.8.6 Logout (5)

Logout (35=5) initiates or confirms the termination of a FIX session. FIX clients should terminate their sessions gracefully by logging out.

If a FIX user is disabled by LME Market Operations while logged in then a Logout message will be sent to the user and the session will be disconnected.

If a FIX user has their password reset by LME Market Operations and attempts to login with their previous password, the user will receive a Logout with SessionStatus (1409) = ‘100’ Password change is required.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 1409 | SessionStatus | N | Int | Session status at time of logout.<br>Valid values:<br>3 = New session password does not comply with policy<br>4 = Session logout complete<br>5 = Invalid username or password<br>6 = Account locked<br>7 = Logons are not allowed at this time<br>8 = Password expired<br>100 = Password change is required<br>101 = Other |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 58 | Text | C | String (50) | Reason for logout.<br>Conditionally required if SessionStatus (1409) = ‘101’ Other |

### 4.8.7 Reject (3)

Reject (35=3) will be sent when a message is received but cannot be properly processed by the gateway due to a session level rule violation. For example, a message missing a mandatory tag.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 45 | RefSeqNum | Y | SeqNum (9) | Sequence number of the message which caused the rejection. |
| 371 | RefTagID | N | Int | If a message is rejected due to an issue with a particular field its tag number will be indicated. |
| 372 | RefMsgType | N | String (2) | Message type of the rejected message. |
| 373 | SessionRejectReason | N | Int | Code specifying the reason for the session level rejection:<br>Valid values:<br>0 = Invalid Tag Number<br>1 = Required Tag Missing<br>2 = Tag not defined for this message<br>4 = Tag specified without a value<br>5 = Value is incorrect (out of range) for this tag<br>6 = Incorrect data format for value<br>9 = CompID problem<br>10 = Sending Time Accuracy problem<br>11 = Invalid Msg Type<br>13 = Tag appears more than once<br>15 = Repeating group fields out of order<br>16 = Incorrect NumInGroup count for repeating group<br>99 = Other. |
| 58 | Text | ~~C*~~N | String (50) | ~~Conditionally required if SessionRejectReason (373) = ’99’ Other.~~<br>Text specifying the reason for the rejection. |

## 4.9 Other Messages

### 4.9.1 Business Message Reject (j)

Once an application level message passes validation at FIX session level it will then be validated at business level. If business level validation detects an error condition then a rejection should be issued. Many business level messages have specific tags for rejection handling where a specific tag is not available the Business Message Reject message (35=j) will be returned.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 45 | RefSeqNum | N | SeqNum (9) | Sequence number of the message which caused the rejection. |
| 372 | RefMsgType | Y | String (2) | Message type of the rejected message. |
| 379 | BusinessRejectRefID | N | String (20) | Client specified unique identifier on the message that was rejected.<br>For example, for a New Order Single this would be the client specified identifier in the ClOrdID (11). |
| 380 | BusinessRejectReason | Y | Int | Code specifying the reason for the rejection of the message.<br>Valid values:<br>0 = Other<br>2 = Unknown Security<br>3 = Unsupported Message Type<br>5 = Conditionally required field missing<br>8 = Throttle limit exceeded<br>9 = Throttle limit exceeded, session will be disconnected. |
| 58 | Text | ~~C*~~N | String (50) | ~~Conditionally required if BusinessRejectReason (380) = ‘0’ Other.~~<br>Text specifying the reason for the rejection. |

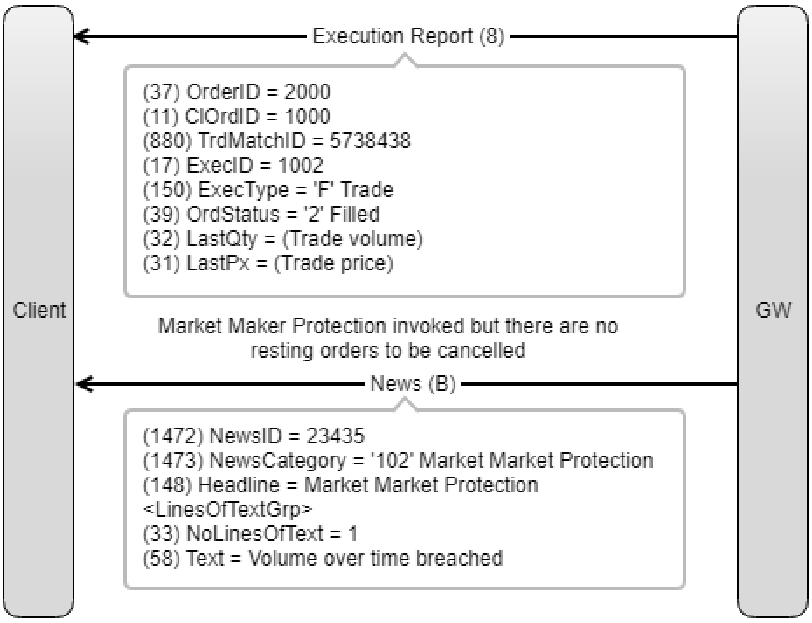
### 4.9.2 News (B)

A News message (35=B) is a general free format message from the exchange. *A News message is also sent in response to a market maker protection breach, see Market Maker Protection in the Order Entry Gateway Binary Specification.*

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 1472 | NewsID | Y* | String (20) | Unique identifier for News message. |
| 1473 | NewsCategory | Y* | Int | Category of News message.<br>Valid values:<br>101 = Market message<br>*102 = Market Maker Protection* |
| 42 | OrigTime | Y* | UTCTimestamp | Time of message origination |
| 148 | Headline | Y | String (75) | Specifies the headline text either Market message or *Market Maker Protection* |
| Component Block <LinesOfTextGrp> | | | | |
| 33 | NoLinesOfText | Y | NumInGrp (1) | Specifies the number of repeating lines of text specified.<br>This value is always set to 1. |
| >58 | Text | Y | String (250) | Free text field for Market message or *one of the following for Market Maker Protection:*<br>• *Cumulative percent over time breached*<br>• *Volume over time breached*<br>• *Number of tradable instruments traded over time breached* |
| End Component Block | | | | |

### Example Message Flow

*Market Maker Protection breached*


Client
GW
Execution Report (8)
(37) OrderID = 2000
(11) ClOrdID = 1000
(880) TrdMatchID = 5738438
(17) ExecID = 1002
(150) ExecType = 'F' Trade
(39) OrdStatus = '2' Filled
(32) LastQty = (Trade volume)
(31) LastPx = (Trade price)
Market Maker Protection invoked but there are no resting orders to be cancelled
News (B)
(1472) NewsID = 23435
(1473) NewsCategory = '102' Market Market Protection
(148) Headline = Market Market Protection
<LinesOfTextGrp>
(33) NoLinesOfText = 1
(58) Text = Volume over time breached

## 4.10 Parties Component Block

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 453 | NoPartyIDs | Y* | NumInGrp (2) | Number of parties specified. |
| >448 | PartyID | Y* | String<br>See PartyRole Usage | Party identifier/code.<br>Required if NoPartyIDs (453) > 0. |
| >447 | PartyIDSource | Y* | Char | Source of the PartyID (448) value. Required if NoPartyIDs (453) > 0.<br>Valid values:<br>P = Client Short Code<br>D = Proprietary/Custom<br>E = ISO Country Code (i.e. two letter ISO country code)<br>N = Legal Entity ID - LEI |
| >452 | PartyRole | Y* | Int | Role of the specified PartyID (448). Required if NoPartyIDs (453) > 0.<br>Valid values:<br>1 = Executing Firm<br>3 = Client ID |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
|  |  |  |  | 4 = Clearing Firm<br>7 = Entering Firm<br>11 = Order Origination Trader<br>24 = Customer Account<br>26 = Correspondent broker<br>36 = Entering Trader<br>66 = Market Maker<br>81 = Broker Client ID<br>122 = Decision Maker<br>300 = Investment Decision Within Firm<br>301 = Execution Decision Within Firm<br>302 = Investment Decision Country<br>303 = Execution Decision Country<br>304 = Client Branch Country |

### 4.10.1 PartyRole Usage

PartyRole (452) values used by LME are described below:

| PartyRole (452) | | PartyID (448) format | PartyIDSource (447) | Description | Usage |
|---|---|---|---|---|---|
| 1 | Executing Firm | Char (3) | D = Proprietary/Custom | Identifier of the executing firm.<br>Cannot be entered in requests but will be returned in Execution Reports. | N/A for order entry<br>Used for Transaction Reporting and Order Record Keeping |
| 3 | Client ID | Int (8 bytes) | P = Client Short Code | Client short code identifier.<br>Required only for client orders i.e. AccountType (581) = 1, 8 or 101 where OrderAttributeType (2594) = 0 or 1 has not been specified, see Order Attribute Type Usage.<br>Note: PartyID (448) can be set to 0 = No Client for if there is no client where AccountType (581) = 3.<br>PartyID (448) is not valid if populated with either 1, 2 or 3. | Conditionally required for Client orders<br>Used for Transaction Reporting and Order Record Keeping<br>Up to two instances of PartyRole (452) = '3' Client ID can be specified but PartyIDSource (447) values must be unique. |
| | | ~~Alphanumeric~~String (<=40) | D = Proprietary/Custom | Proprietary or Custom Client ID as assigned by the member.<br>Required only for client orders i.e. AccountType (581) = 1, 8 or 101. | |
| | | ~~Alphanumeric~~String (<=40) | N = Legal Entity ID | LEI. | Optional |

| PartyRole (452) | | PartyID (448) format | PartyIDSource (447) | Description | Usage |
|---|---|---|---|---|---|
| | | | | | Up to two instances of PartyRole (452) = '3' Client ID can be specified but PartyIDSource (447) values must be unique. |
| 4 | Clearing Firm | Char (3) | D = Proprietary/Custom | Identifier of the clearing firm. A 3 character broker code (Member mnemonic).<br>Cannot be entered in requests but is returned in Execution Reports for all fills. | N/A for order entry<br>Used for Transaction Reporting and Order Record Keeping |
| 7 | Entering Firm | Char (3) | D = Proprietary/Custom | Identifier of the entering firm. A 3 character broker code (Member mnemonic).<br>Required for MiFID as agent relationships are not captured in the LME participant structure. | Optional<br>Used for Transaction Reporting and Order Record Keeping |
| 11 | Order Origination Trader | String (<=40) | D = Proprietary/Custom | Order Origination Trader (associated with Order Origination Firm i.e. trader who initiates/submits the order).<br>Required as could be more than one individual under a FIX Comp ID.<br>Required in New Order Single and will be returned in Execution Reports. | Mandatory for House and Client orders |

| PartyRole (452) | | PartyID (448) format | PartyIDSource (447) | Description | Usage |
|---|---|---|---|---|---|
| 24 | Customer Account | ~~Alphanumeric~~String (<=30) | D = Proprietary/Custom | Identification of the Client Account Code where the AccountType (581) = 1, 8 or 101. | Conditional - Mandatory for Client orders |
| 26 | Correspondent broker - Non-executing broker | Char (3) | D = Proprietary/Custom | A 3 character broker code (Member mnemonic). | Optional<br>Used for Order Record Keeping |
| 36 | Entering Trader | String (<=10) | D = Proprietary/Custom | Identifier of the trader entering the order. Cannot be entered in requests but will be returned in Execution Reports. | N/A for order entry |
| 66 | Market Maker | Char (1) | D = Proprietary/Custom | This should be set to Y if the trader qualifies for a Market Maker initiative | Optional |
| 81 | Broker Client ID | String (<=16) | D = Proprietary/Custom | Identifier of the entity in a risk group.<br>Required in New Order Single and will be returned in Execution Reports. | Mandatory used for Risk Management |
| 122 | Decision Maker | Int (8 bytes) | P = Client Short Code | Decision maker short code, required on client orders to identify the investment decision maker. Also used under the power of representation clause where the investment decision maker may be a third party in accordance with Article 8 of Commission Delegated Regulation (EU) …/… | Conditional - Mandatory for Client orders<br>Used for Transaction Reporting |

| PartyRole (452) | | PartyID (448) format | PartyIDSource (447) | Description | Usage |
|---|---|---|---|---|---|
| | | | | 22 on transaction reporting under Article 26 of Regulation EU No 600/2014.<br>Required only for client orders i.e. AccountType (581) = 1, 8 or 101. | |
| 300 | Investment Decision Within Firm | Int (8 bytes) | P = Client Short Code | Short code to identify the individual who is responsible for the investment decision. | Optional<br>Used for Transaction Reporting and Order Record Keeping |
| 301 | Execution Decision Within Firm | Int (8 bytes) | P = Client Short Code | Short code to identify the execution decision maker with the firm.<br>Required in New Order Single and Order Cancel Replace Requests and will be returned in Execution Reports. | Mandatory for House and Client orders<br>Used for Transaction Reporting and Order Record Keeping |
| 302 | Investment Decision Country | Char (2) | E = ISO Country Code | ISO Country Code of the branch responsible for the person making the investment decision. | Optional<br>Used for Transaction Reporting |
| 303 | Execution Decision Country | Char (2) | E = ISO Country Code | ISO Country Code of the branch responsible for the person making the execution decision. | Optional<br>Used for Transaction Reporting |

| PartyRole (452) | | PartyID (448) format | PartyIDSource (447) | Description | Usage |
|---|---|---|---|---|---|
| 304 | Client Branch Country | Char (2) | E = ISO Country Code | ISO Country Code to identify the branch that received the client order or made an investment decision for a client.<br>Required for client orders i.e. AccountType (581) = 1, 8 or 101 | Conditional - Mandatory for Client orders<br>Used for Transaction Reporting |

## 4.11 Application Messages

### 4.11.1 Security Definition Request (c)

Security Definition Request (35=c) is used to request the creation of either an *option strike* or a strategy. A Security Definition (35=d) will be sent in response to the request.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 320 | SecurityReqID | Y | String (18) | Unique ID of a Security Definition Request. |
| 321 | SecurityRequestType | Y | String (1) | Type of Security Definition Request.<br>1 = Request security identity for the specifications provided |
| Component Block <Instrument> | | | | |
| 207 | SecurityExchange | Y* | Exchange (4) | Market which is used to identify the security:<br>XLME |
| 1227 | ProductComplex | Y* | String (4) | Identifies an entire suite of products for a given market.<br>Valid values:<br>LME = Base |
| 55 | Symbol | Y* | String (20) | Symbol for the LME contract code e.g. CADF (Copper Future) or OCDF (Copper Monthly Average Future). |
| 167 | SecurityType | Y* | String (4) | Indicates the type of security whether outright or strategy e.g. MLEG for strategy.<br>Valid values:<br>*OPT = Option*<br>MLEG = Multileg instrument |
| 762 | SecuritySubType | Y* | Int | Indicates the security sub type.<br>Valid values:<br>*0 = Outright*<br>1 = Carry<br>*2 = Custom (Futures)*<br>*3 = 3 Month Average*<br>*4 = 6 Month Average* |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | *5 = 12 Month Average*<br>*6 = Carry Average*<br>*7 = Call Spread*<br>*8 = Put Spread*<br>*9 = Custom (Delta Hedge)*<br>*10 = Custom (Options)* |
| *541* | *MaturityDate* | *C** | *LocalMktDate* | *Expiration date for options (YYYYMMDD)*<br>*Conditionally required if SecurityType (167) = 'OPT' Option.*<br>*Not required if SecurityType (167) = MLEG.* |
| *202* | *StrikePrice* | *C** | *Price (20)* | *Strike Price for an Option.*<br>*Conditionally required if SecurityType (167) = 'OPT' Option.* |
| *201* | *PutOrCall* | *C** | *Int* | *Used to express option right*<br>*Valid values:*<br>*0 = Put*<br>*1 = Call*<br>*Conditionally required if SecurityType (167) = 'OPT' Option.* |
| Component Block <InstrmtLegGrp> | | | | |
| 555 | NoLegs | C* | NumInGrp (1) | Conditionally required if SecurityType (167) = 'MLEG'<br>Number of InstrumentLeg repeating group instances. Cannot be *more than 5* or less than 2.<br>*Note this will only be 1 for a 3 Month Average, 6 Month Average and 12 Month Average.* |
| *Component Block <InstrumentLeg>* | | | | |
| >602 | LegSecurityID | C* | Int | SecurityID of the leg derived from the SecurityID (48) of the outright.<br>Conditionally required if SecurityType (167) = MLEG. |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | *For an Average strategy only the LegSecurityID of the first leg of the strategy is provided as the other months are consecutive.* |
| >603 | LegSecurityIDSource | C* | String (1) | Identifies the source of the LegSecurityID value.<br>Valid value:<br>8 = Exchange defined.<br>Conditionally required when LegSecurityID (602) is specified. |
| >623 | LegRatioQty | C* | Float | The ratio of quantity for this individual leg relative to the entire multileg security. Conditionally required if SecurityType (167) = MLEG.<br>*For example, for a custom strategy such as a Butterfly the leg ratio would be 1:2:1 (1.000:2.000:1.000), for the first leg LegRatioQty = 1.000 (buy near contract month), second leg LegRatioQty = 2.000 (sell two contracts in far month) and third leg LegRatioQty = 1.000 (buy one contract in yet farther month).*<br>*For a Carry Average the front leg must include a ratio for the number of average legs. For example, 3M-3Q (Jul/Aug/Sep) Carry Average, 3M leg LegRatioQty = 3.000, legs 2/3/4 would have LegRatioQty = 1.000.*<br>*For a Delta Hedge Custom, this is the delta used to determine the covering quantity.* |
| >624 | LegSide | C* | Char | The side of this individual leg.<br>Valid values:<br>1 = Buy<br>2 = Sell.<br>Conditionally required if SecurityType (167) = MLEG. |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| *>566* | *LegPrice* | *C** | *Price* | *Used to specify an anchor price for a leg as part of the definition or creation of the strategy - not used for execution price.*<br>*Conditionally required for the futures legs of SecuritySubType (762) = '9' Delta Hedge Custom to specify the underlying futures price.* |
| End Component Blocks | | | | |

### 4.11.2 Security Definition (d)

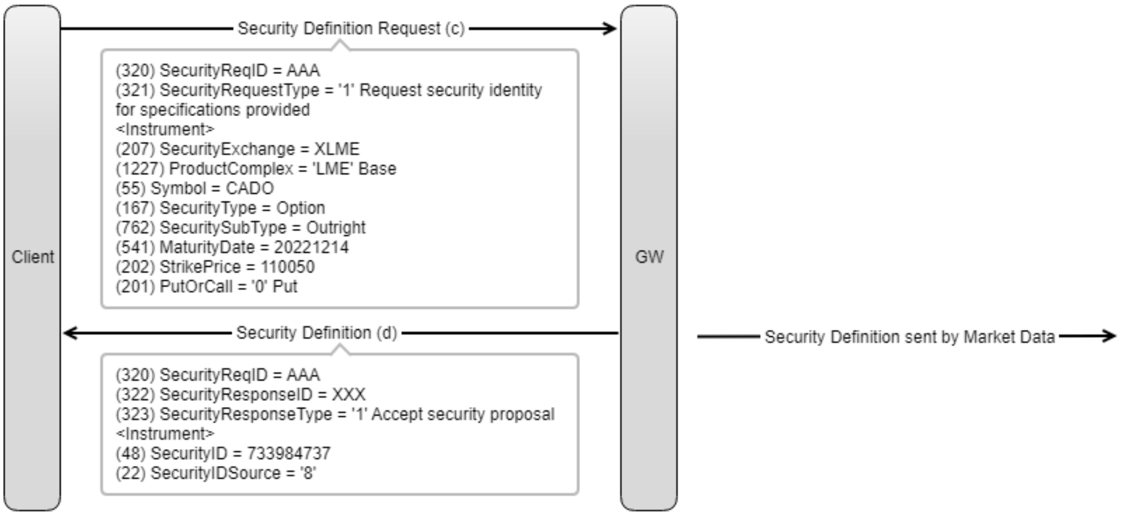
Security Definition (35=d) will be returned to the originator of the Security Definition Request (35=c) to accept, accept with revisions or reject the creation of a tradable instrument. Market participants will be notified of a newly created instrument by the Market Data service.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 320 | SecurityReqID | Y* | String (18) | Client generated ID supplied on the Security Definition Request. |
| 322 | SecurityResponseID | Y* | String (20) | Unique ID of a Security Definition (d) message. |
| 323 | SecurityResponseType | Y* | Int | Type of Security Definition message response.<br>Valid values:<br>1 = Accept security proposal<br>2 = Accept security proposal with revisions as indicated in the message<br>5 = Reject security proposal |
| 1607 | SecurityRejectReason | C | Int | Identifies the reason a security definition request is being rejected.<br>Conditionally required to specify a rejection reason when SecurityResponseType (323) = ‘5’ Reject security proposal.<br>Valid values:<br>99 = Other<br>101 = Throttle limit exceeded<br>102 = Invalid strike price<br>103 = LegSecurityID (602) does not exist |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | 104 = Invalid prompt date<br>105 = Invalid SecuritySubType (762) |
| Component Block <Instrument> | | | | |
| 48 | SecurityID | C* | Int | Tradable instrument identifier<br>Conditionally required if SecurityResponseType (323) = '1' Accept security proposal or '2' Accept security proposal with revisions as indicated in the message |
| 22 | SecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when SecurityID (48) is specified. |
| End Component Block | | | | |
| 58 | Text | C* | String (75) | Identifies the reason for rejection.<br>Conditionally required if SecurityRejectReason (1607) = '99' Other |

### Example Message Flows

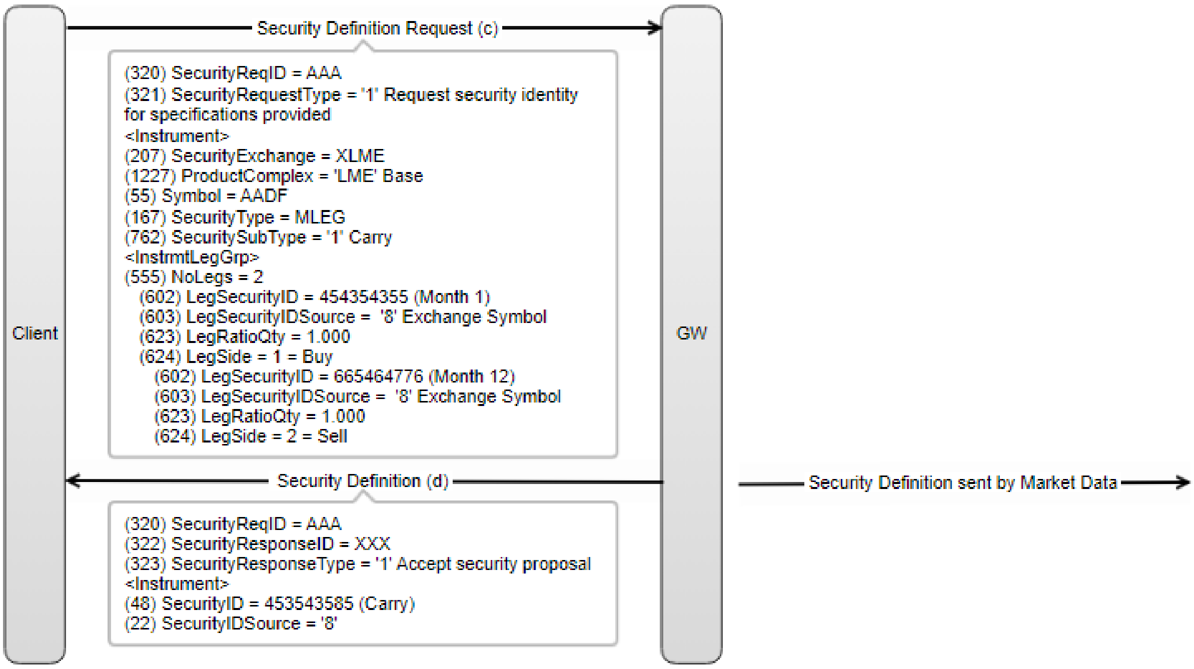
*Option Strike Request*


Client
Security Definition Request (c)
GW
Security Definition (d)
Security Definition sent by Market Data

(320) SecurityReqID = AAA
(321) SecurityRequestType = '1' Request security identity for specifications provided
<Instrument>
(207) SecurityExchange = XLME
(1227) ProductComplex = 'LME' Base
(55) Symbol = CADO
(167) SecurityType = Option
(762) SecuritySubType = Outright
(541) MaturityDate = 20221214
(202) StrikePrice = 110050
(201) PutOrCall = '0' Put

(320) SecurityReqID = AAA
(322) SecurityResponseID = XXX
(323) SecurityResponseType = '1' Accept security proposal
<Instrument>
(48) SecurityID = 733984737
(22) SecurityIDSource = '8'

*Futures Strategy Request*



Security Definition Request (c)

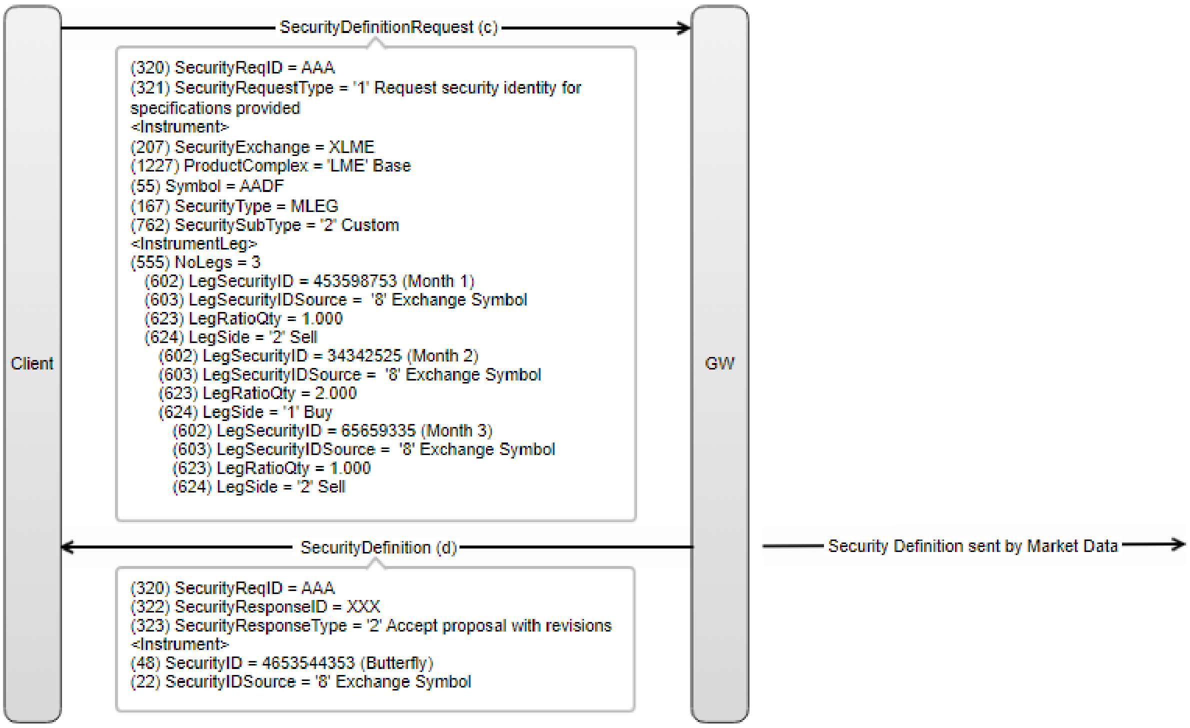
```
(320) SecurityReqID = AAA
(321) SecurityRequestType = '1' Request security identity
for specifications provided
<Instrument>
(207) SecurityExchange = XLME
(1227) ProductComplex = 'LME' Base
(55) Symbol = AADF
(167) SecurityType = MLEG
(762) SecuritySubType = '1' Carry
<InstrmtLegGrp>
(555) NoLegs = 2
  (602) LegSecurityID = 454354355 (Month 1)
  (603) LegSecurityIDSource = '8' Exchange Symbol
  (623) LegRatioQty = 1.000
  (624) LegSide = 1 = Buy
    (602) LegSecurityID = 665464776 (Month 12)
    (603) LegSecurityIDSource = '8' Exchange Symbol
    (623) LegRatioQty = 1.000
    (624) LegSide = 2 = Sell
```

Security Definition (d)

```
(320) SecurityReqID = AAA
(322) SecurityResponseID = XXX
(323) SecurityResponseType = '1' Accept security proposal
<Instrument>
(48) SecurityID = 453543585 (Carry)
(22) SecurityIDSource = '8'
```

Security Definition sent by Market Data

*Inverse Custom Strategy Request*



SecurityDefinitionRequest (c)

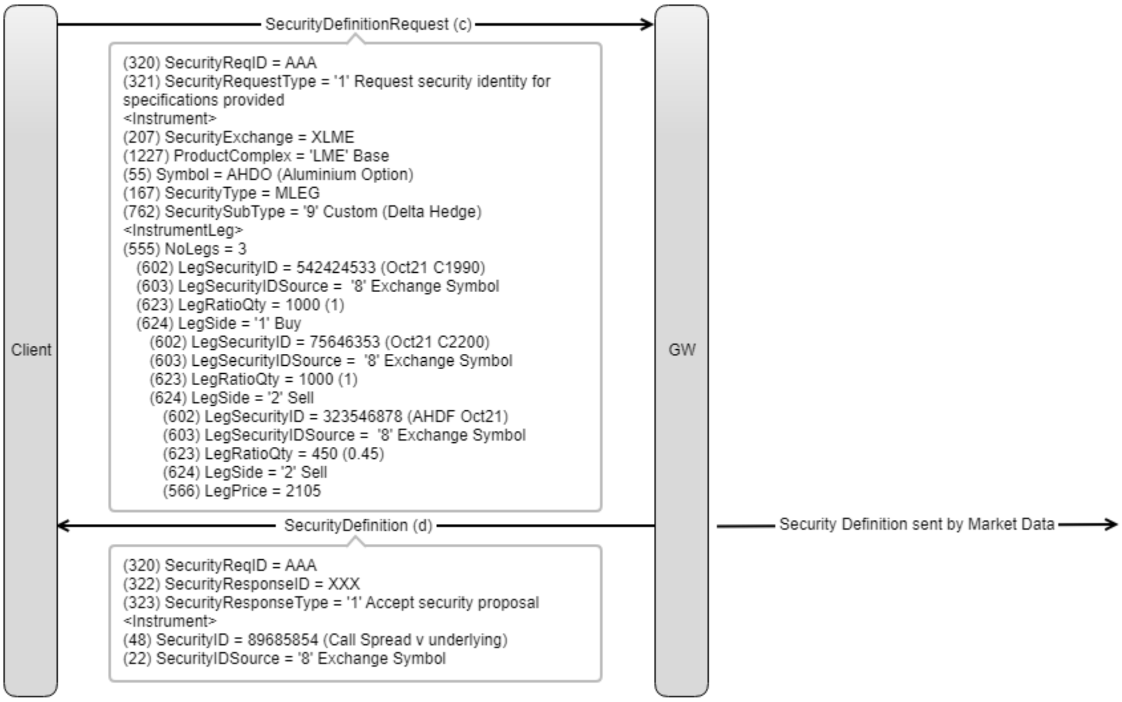
```
(320) SecurityReqID = AAA
(321) SecurityRequestType = '1' Request security identity for
specifications provided
<Instrument>
(207) SecurityExchange = XLME
(1227) ProductComplex = 'LME' Base
(55) Symbol = AADF
(167) SecurityType = MLEG
(762) SecuritySubType = '2' Custom
<InstrumentLeg>
(555) NoLegs = 3
  (602) LegSecurityID = 453598753 (Month 1)
  (603) LegSecurityIDSource = '8' Exchange Symbol
  (623) LegRatioQty = 1.000
  (624) LegSide = '2' Sell
    (602) LegSecurityID = 34342525 (Month 2)
    (603) LegSecurityIDSource = '8' Exchange Symbol
    (623) LegRatioQty = 2.000
    (624) LegSide = '1' Buy
      (602) LegSecurityID = 65659335 (Month 3)
      (603) LegSecurityIDSource = '8' Exchange Symbol
      (623) LegRatioQty = 1.000
      (624) LegSide = '2' Sell
```

SecurityDefinition (d)

```
(320) SecurityReqID = AAA
(322) SecurityResponseID = XXX
(323) SecurityResponseType = '2' Accept proposal with revisions
<Instrument>
(48) SecurityID = 4653544353 (Butterfly)
(22) SecurityIDSource = '8' Exchange Symbol
```

Security Definition sent by Market Data

*Delta Hedge Strategy Request – Call Spread versus underlying*


SecurityDefinitionRequest (c)
Client
GW
SecurityDefinition (d)
Security Definition sent by Market Data

```
(320) SecurityReqID = AAA
(321) SecurityRequestType = '1' Request security identity for
specifications provided
<Instrument>
(207) SecurityExchange = XLME
(1227) ProductComplex = 'LME' Base
(55) Symbol = AHDO (Aluminium Option)
(167) SecurityType = MLEG
(762) SecuritySubType = '9' Custom (Delta Hedge)
<InstrumentLeg>
(555) NoLegs = 3
  (602) LegSecurityID = 542424533 (Oct21 C1990)
  (603) LegSecurityIDSource =  '8' Exchange Symbol
  (623) LegRatioQty = 1000 (1)
  (624) LegSide = '1' Buy
    (602) LegSecurityID = 75646353 (Oct21 C2200)
    (603) LegSecurityIDSource =  '8' Exchange Symbol
    (623) LegRatioQty = 1000 (1)
    (624) LegSide = '2' Sell
      (602) LegSecurityID = 323546878 (AHDF Oct21)
      (603) LegSecurityIDSource =  '8' Exchange Symbol
      (623) LegRatioQty = 450 (0.45)
      (624) LegSide = '2' Sell
      (566) LegPrice = 2105
```

```
(320) SecurityReqID = AAA
(322) SecurityResponseID = XXX
(323) SecurityResponseType = '1' Accept security proposal
<Instrument>
(48) SecurityID = 89685854 (Call Spread v underlying)
(22) SecurityIDSource = '8' Exchange Symbol
```

### 4.11.3 New Order Single (D)

New Order Single (35=D) is used to submit a new order for execution. An Execution Report (35=8), Reject (35=3) or Business Message Reject (35=j) is sent in response.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 11 | ClOrdID | Y | String (18) | Unique identifier set by the order originator. |
| Component Block <Parties> | | Y* | See Parties Component Block<br>The following PartyRole (452) values are mandatory:<br>'11' Order Origination Trader<br>‘81’ Broker Client ID<br>'301' Execution Decision Within Firm | |
| 581 | AccountType | Y* | Int | Specifies the type of account associated with the order.<br>Valid values:<br>1 = Client ISA<br>3 = House<br>8 = Joint back office account (JBO)<br>= Gross OSA<br>101 = Client OSA |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | For contracts assigned to the T4 booking model only 3 = House is valid whereas for the T2 booking model all account types are valid. |
| 18 | ExecInst | N | MultipleCharValue | Instructions for order handling. If more than one instruction is applicable to an order, this field can contain multiple instructions separated by space.<br>Valid values:<br>*6 = Participate but don't initiate (required for Post only order submission)*<br>o = Cancel on connection loss |
| *Component Block <DisplayInstruction>* | | | | |
| *1138* | *DisplayQty* | *C** | *Qty* | *Visible quantity.*<br>*Conditionally required for Iceberg orders.*<br>*If present, must be < OrderQty (38)* |
| *End Component Block* | | | | |
| Component Block <Instrument> | | | | |
| 48 | SecurityID | Y* | Int | Tradable instrument identifier |
| 22 | SecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when SecurityID (48) is specified. |
| End Component Block | | | | |
| 54 | Side | Y | Char | Side of the order<br>Valid values:<br>1 = Buy<br>2 = Sell |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 60 | TransactTime | Y | UTCTimestamp | Timestamp when the message was generated. |
| Component Block <OrderQtyData> | | | | |
| 38 | OrderQty | Y | Qty | Total quantity of the order. |
| End Component Block | | | | |
| 40 | OrdType | Y | Char | Order type applicable to the order.<br>Valid values:<br>*1 = Market*<br>2 = Limit<br>*3 = Stop Market*<br>4 = Stop Limit |
| 44 | Price | C | Price (20) | Price of the order.<br>Conditionally required if OrdType (40) = ‘2’ Limit or ‘4’ Stop Limit |
| 99 | StopPx | C | Price (20) | The Stop trigger price.<br>Conditionally required if OrdType (40):<br>*3 = Stop Market*<br>4 = Stop Limit<br>TriggerPriceType (1107) is required if a Stop Price is specified. |
| Component Block <TriggeringInstruction> | | | | |
| 1100 | TriggerType | C | Char | Trigger prompt for stop order elements.<br>Conditionally required if any other Triggering tags are specified.<br>Valid value:<br>4 = Price Movement |
| *1102* | *TriggerPrice* | *C** | *Price (20)* | *Stop order price of the OCO.*<br>*Conditionally required for an OCO.* |
| 1107 | TriggerPriceType | C* | Char | Type of price event that triggers the stop order: |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | Valid values:<br>2 = Last Trade<br>4 = Best Bid or Last Trade<br>5 = Best Offer or Last Trade<br>Conditionally required if StopPx (99) *or TriggerPrice (1102)* is specified |
| *1110* | *TriggerNewPrice* | *C** | *Price (20)* | *Limit order price of the stop once triggered.*<br>*Conditionally required if Trigger Order Type (1111) = '2' Limit* |
| *1111* | *TriggerOrderType* | *C** | *Char* | *Order type of the stop once triggered.*<br>*Valid values:*<br>*1 = Market*<br>*2 = Limit*<br>*Conditionally required for an OCO.* |
| End Component Block | | | | |
| 59 | TimeInForce | N | Char | Specifies how long the order remains in effect.<br>Valid values:<br>0 = Day<br>1 = Good Till Cancel<br>3 = Immediate or Cancel<br>*4 = Fill or Kill*<br>6 = Good Till Date<br>Absence of this field indicates Day. |
| 432 | ExpireDate | C | LocalMktDate | The expiry date of an order.<br>Conditionally required if TimeInForce (59) = Good Till Date.<br>Format is YYYYMMDD. |
| 528 | OrderCapacity | Y* | Char | Indicates the trading capacity.<br>Valid values:<br>A (agency) = AOTC<br>P (principal) = DEAL |

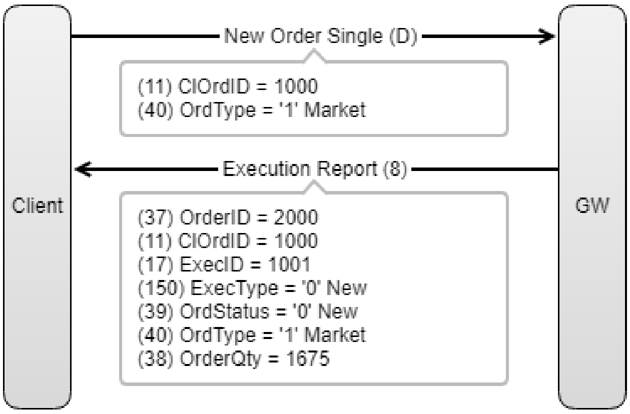
| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | R (riskless principal) = MTCH |
| 529 | OrderRestrictions | Y* | Char | Restrictions associated with an order.<br>Valid values:<br>D = Non-algorithmic (human)<br>E = Algorithmic (algo) |
| 58 | Text | N | String (50) | Free text. |
| 1724 | OrderOrigination | N | Int | Identifies the origin of the order.<br>Valid value:<br>5 = Order received from a direct access or sponsored access customer (the trader has direct electronic access – DEA).<br>Absence of this field indicates DEA = false |
| *2362* | *SelfMatchPreventionID* | *N* | *Int (9)* | *Identifies an order that should not be matched to an opposite order if both buy and sell orders for the trade contain the same SelfMatchPreventionID (2362) and are submitted by the same member.* |
| Component Block <OrderAttributeGrp> | | | | |
| 2593 | NoOrderAttributes | N | NumInGrp (1) | Number of order attribute entries. |
| >2594 | OrderAttributeType | C* | Int | The type of order attribute, see Order Attribute Type Usage.<br>Conditionally required if NoOrderAttributes (2593) > 0.<br>Valid values:<br>0 = Aggregated order. In the context of ESMA RTS 24 Article 2(3), when OrderAttributeValue (2595) = Y, it signifies that the order consists of several orders |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | aggregated together. This maps to ESMA RTS value "AGGR".<br>1 = Pending allocation. In the context of ESMA RTS 24 Article 2(2), when OrderAttributeValue (2595) = Y, it signifies that the order submitter "is authorized under the legislation of a Member State to allocate an order to its client following submission of the order to the trading venue and has not yet allocated the order to its client at the time of the submission of the order". This maps to ESMA RTS value "PNAL".<br>2 = Liquidity Provision Order. In the context of ESMA RTS 24 Article 3, when OrderAttributeValue (2595) = Y, it signifies that the order was submitted "as part of a market making strategy pursuant to Articles 17 and 18 of Directive 2014/65/EU or is submitted as part of another activity in accordance with Article 3" (of RTS 24).<br>3 = Risk Reduction Order. In the context of ESMA RTS 22 Article 4(2)(i), when OrderAttributeValue (2595) = Y, it signifies that the commodity derivative order is a transaction "to reduce risk in an objectively measurable way in accordance with Article 57 of Directive 2014/65/EU". |
| >2595 | OrderAttributeValue | C* | String (1) | The value associated with the order attribute type specified in OrderAttributeType (2594).<br>Conditionally required if NoOrderAttributes (2593) > 0.<br>Valid value:<br>Y = Yes |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| End Component Block | | | | |

**Example Message Flow**

*Market order*



### 4.11.4 Order Cancel Replace Request (G)

Order Cancel Replace Request (35=G) is used to change the parameters of an existing order. If successful an Execution Report (35=8) is returned to confirm replacement of the order otherwise an Order Cancel Reject (35=9) is returned if the request is rejected and the order remains unchanged.

The following tags will not be available on an Order Cancel Replace Request and will therefore retain the value supplied on order entry:

- AccountType (581)
- *SelfMatchPreventionID (2362)*
- *TriggerPriceType (1107)*
- *TriggerOrderType (1111).*

Similarly the Party details that cannot be amended will also not be present and therefore remain unchanged. The submission of party details that cannot be amended will result in the Order Cancel Replace Request being rejected.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 37 | OrderID | N | String (19) | A unique order identifier assigned by the trading system. This identifier is not changed by cancel/replace messages; it will remain the same for all chain of orders. |
| Component Block <Parties> | | Y* | See Parties Component Block | |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | The PartyID (448) of the following mandatory PartyRole (452) can be modified:<br>301 = Execution Decision Within Firm<br>The PartyID (448) of the following PartyRole (452) values can be modified<br>300 = Investment Decision Within Firm<br>302 = Investment Decision Country<br>303 = Execution Decision Country<br>304 = Client Branch Country<br>For the above roles, if the party was not specified on the original order, it can be added. If previously included it can be removed or amended. | |
| 41 | OrigClOrdID | Y | String (18) | Original order identified as the order to be amended. It is the ID of the latest non-rejected order (not the initial order of the day). |
| 11 | ClOrdID | Y | String (18) | Unique identifier set by the order originator. |
| 18 | ExecInst | C* | MultipleCharValue | Instructions for order handling. If more than one instruction is applicable to an order, this field can contain multiple instructions separated by space.<br>Valid value:<br>*6 = Participate but don't initiate (required for Post only order submission)*<br>o = Cancel on connection loss<br>Conditionally required if specified on the original order and must match the previous submission. |
| *Component Block <DisplayInstruction>* | | | | |
| *1138* | *DisplayQty* | *C** | *Qty* | *Visible quantity for an Iceberg order.*<br>*Conditionally required if specified on the original order.* |
| *End Component Block* | | | | |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| Component Block <Instrument> | | | | |
| 48 | SecurityID | Y* | Int | Tradable instrument identifier.<br>Must be the same as the original order. |
| 22 | SecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when SecurityID (48) is specified. |
| End Component Block | | | | |
| 54 | Side | Y | Char | Must be the same value as original order.<br>Valid values:<br>1 = Buy<br>2 = Sell |
| 60 | TransactTime | Y | UTCTimestamp | Timestamp when the message was generated. |
| Component Block <OrderQtyData> | | | | |
| 38 | OrderQty | Y | Qty | New order quantity. Note: this is not the LeavesQty (151) but the new total quantity of the order. |
| End Component Block | | | | |
| 40 | OrdType | Y | Char | The order type must be the same as the original order however a previously triggered Stop Limit or *Stop Market* order will be restated as a Limit order.<br>Valid values:<br>*1 = Market*<br>2 = Limit<br>*3 = Stop Market*<br>4 = Stop Limit |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 44 | Price | C | Price (20) | Price of the order.<br>Conditionally required for all Limit order types. |
| 99 | StopPx | C | Price (20) | The Stop trigger price.<br>Conditionally required if OrdType (40):<br>*3 = Stop Market*<br>4 = Stop Limit. |
| Component Block <TriggeringInstruction> | | | | |
| 1100 | TriggerType | C* | Char | Trigger prompt for stop order elements.<br>*Conditionally required if OCO Triggering tags are specified.*<br>Not required if StopPx (99) is specified.<br>Valid value:<br>4 = Price Movement |
| *1102* | *TriggerPrice* | *C** | *Price (20)* | *Stop order price of the OCO. Conditionally required if specified on the original order.* |
| *1110* | *TriggerNewPrice* | *C** | *Price (20)* | *Limit order price of the stop once triggered. Conditionally required if specified on the original order.* |
| End Component Block | | | | |
| 59 | TimeInForce | C* | Char | Specifies how long the order remains in effect. Conditionally required if specified on the original order and must match with the previous submission.<br>Valid values:<br>0 = Day<br>1 = Good Till Cancel<br>3 = Immediate or Cancel<br>*4 = Fill or Kill*<br>6 = Good Till Date |

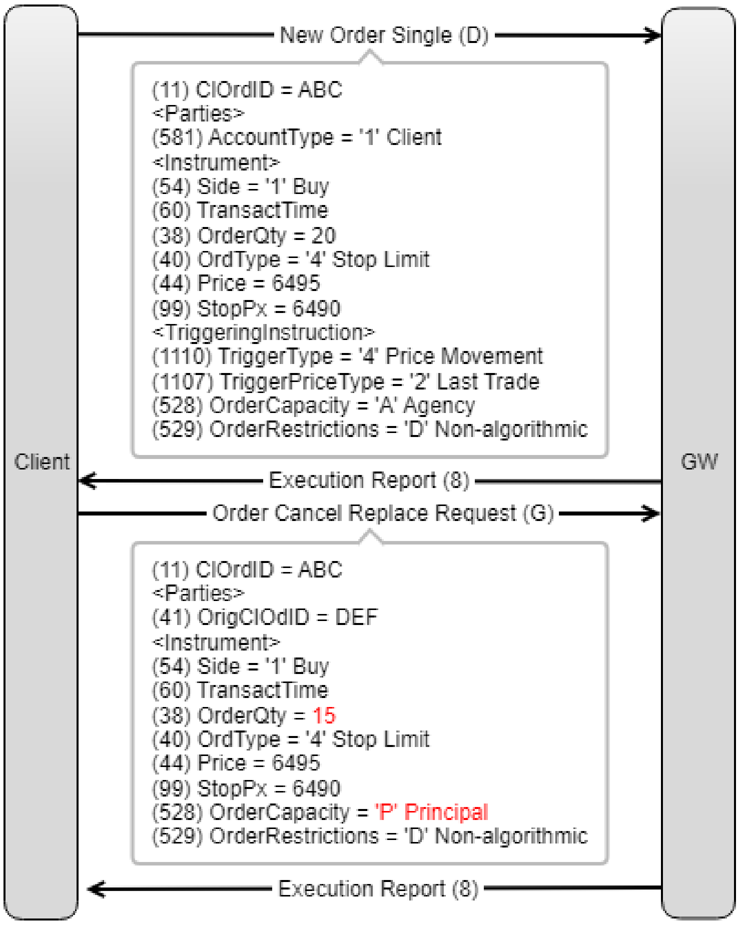
| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | Absence of this field indicates Day. |
| 432 | ExpireDate | C | LocalMktDate | The expiry date of an order.<br>Conditionally required if TimeInForce (59) = Good Till Date is specified.<br>Format is YYYYMMDD. |
| 528 | OrderCapacity | Y* | Char | Indicates the trading capacity.<br>Valid values:<br>A (agency) = AOTC<br>P (principal) = DEAL<br>R (riskless principal) = MTCH |
| 529 | OrderRestrictions | Y* | Char | Restrictions associated with an order.<br>Valid values:<br>D = Non-algorithmic (human)<br>E = Algorithmic (algo) |
| 58 | Text | N | String (50) | Free text. Can be amended if supplied.<br>If the field is not specified on the original order and is added this indicates an amendment.<br>Absence of the field indicates that previously entered free text has been removed. |
| 1724 | OrderOrigination | N | Int | Identifies the origin of the order.<br>Valid value:<br>5 = Order received from a direct access or sponsored access (the trader has direct electronic access – DEA)<br>Absence of this field indicates DEA = false |
| Component Block <OrderAttributeGrp><br>An OrderAttributeType (2594) e.g. 3 = Risk Reduction Order specified on the original order and present on the amendment indicates that the attribute is unchanged.<br>Absence of the attribute indicates that it has been removed. | | | | |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| If an attribute is not specified on the original order and is added this indicates an amendment. | | | | |
| 2593 | NoOrderAttributes | N | NumInGrp (1) | Number of order attribute entries. |
| >2594 | OrderAttributeType | C* | Char | The type of order attribute, see Order Attribute Type Usage.<br>Conditionally required if NoOrderAttributes (2593) > 0.<br>Valid values:<br>0 = Aggregated order. In the context of ESMA RTS 24 Article 2(3), when OrderAttributeValue (2595) = Y, it signifies that the order consists of several orders aggregated together. This maps to ESMA RTS value "AGGR".<br>1 = Pending allocation. In the context of ESMA RTS 24 Article 2(2), when OrderAttributeValue (2595) = Y, it signifies that the order submitter "is authorized under the legislation of a Member State to allocate an order to its client following submission of the order to the trading venue and has not yet allocated the order to its client at the time of the submission of the order". This maps to ESMA RTS value "PNAL".<br>2 = Liquidity Provision Order. In the context of ESMA RTS 24 Article 3, when OrderAttributeValue (2595) = Y, it signifies that the order was submitted "as part of a market making strategy pursuant to Articles 17 and 18 of Directive 2014/65/EU or is submitted as part of another activity in accordance with Article 3" (of RTS 24).<br>3 = Risk Reduction Order. In the context of ESMA RTS 22 Article 4(2)(i), when OrderAttributeValue (2595) = Y, it signifies that the |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | commodity derivative order is a transaction "to reduce risk in an objectively measurable way in accordance with Article 57 of Directive 2014/65/EU". |
| >2595 | OrderAttributeValue | C* | String (1) | The value associated with the order attribute type specified in OrderAttributeType (2594). Conditionally required if NoOrderAttributes (2593) > 0.<br>Valid value:<br>Y = Yes |
| End Component Block | | | | |

**Example Message Flow**

*Amend Order*



(11) ClOrdID = ABC
<Parties>
(581) AccountType = '1' Client
<Instrument>
(54) Side = '1' Buy
(60) TransactTime
(38) OrderQty = 20
(40) OrdType = '4' Stop Limit
(44) Price = 6495
(99) StopPx = 6490
<TriggeringInstruction>
(1110) TriggerType = '4' Price Movement
(1107) TriggerPriceType = '2' Last Trade
(528) OrderCapacity = 'A' Agency
(529) OrderRestrictions = 'D' Non-algorithmic

(11) ClOrdID = ABC
<Parties>
(41) OrigClOdID = DEF
<Instrument>
(54) Side = '1' Buy
(60) TransactTime
(38) OrderQty = 15
(40) OrdType = '4' Stop Limit
(44) Price = 6495
(99) StopPx = 6490
(528) OrderCapacity = 'P' Principal
(529) OrderRestrictions = 'D' Non-algorithmic

### 4.11.5 Order Cancel Request (F)

Order Cancel Request (35=F) is used to cancel the remaining quantity of an existing order. An Execution Report (35=8) is returned to confirm cancellation or an Order Cancel Reject (35=9) if the cancel is rejected.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 41 | OrigClOrdID | Y | String (18) | Order identifier for the order to cancel. It is the ID of the latest non-rejected order (not the initial order of the day). |
| 37 | OrderID | N | String (19) | A unique order identifier set by the trading system. This identifier is not changed by cancel/replace messages; it will remain the same for all chain of orders. |
| 11 | ClOrdID | Y | String (18) | Unique identifier set by the order originator. |
| Component Block <Instrument> | | | | |
| 48 | SecurityID | Y* | Int | Tradable instrument identifier |
| 22 | SecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when SecurityID (48) is specified. |
| End Component Block | | | | |
| 54 | Side | Y | Char | Side of the order.<br>Valid values:<br>1 = Buy<br>2 = Sell |
| 60 | TransactTime | Y | UTCTimestamp | Timestamp when the message was generated. |

### 4.11.6 Order Cancel Reject (9)

Order Cancel Reject (35=9) is returned in response to Order Cancel/Replace Request (35=G) or Order Cancel Request (35=F) that cannot be honoured. The unchanged order will remain in the order book if it has not already been cancelled or has expired.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 37 | OrderID | Y | String (19) | A unique order identifier set by the trading system. This identifier is not changed by cancel/replace messages; it will remain the same for all chain of orders.<br>Set to NONE when OrdStatus (39) = Rejected and CxlRejReason (102) = '1' Unknown order. |
| 11 | ClOrdID | Y | String (18) | Unique identifier set by the order originator. |
| 41 | OrigClOrdID | Y | String (18) | Original order identified for the order to modify. It is the ID of the latest non-rejected order (not the initial order of the day). |
| 39 | OrdStatus | Y | Char | Order status as at the time of rejection.<br>Valid values:<br>0 = New<br>1 = Partially Filled<br>2 = Filled<br>3 = Done for day<br>4 = Cancelled<br>6 = Pending Cancel<br>8 = Rejected<br>A = Pending New<br>C = Expired<br>E = Pending Replace |
| 60 | TransactTime | Y | UTCTimestamp | Timestamp when the message was generated. |
| 434 | CxlRejResponseTo | Y | Char | Identifies the type of request that an Order Cancel Reject (9) is in response to.<br>Valid values:<br>1 = Order Cancel Request (F)<br>2 = Order Cancel/Replace Request (G) |
| 102 | CxlRejReason | Y* | Int | Code that identifies the reason for the rejection.<br>Valid values:<br>0 = Too late to cancel<br>1 = Unknown order |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | 3 = Order already in Pending Cancel or Pending Replace Status<br>6 = Duplicate ClOrderID (11) received<br>18 = Invalid price increment<br>99 = Other |
| 1328 | RejectText | C* | String (75) | Conditionally required if CxlRejReason (102) = '99' Other.<br>Text specifying the reason for rejection. |
| 1819 | RelatedHighPrice | C* | Price | Upper price limit value |
| 1820 | RelatedLowPrice | C* | Price | Lower price limit value |

### 4.11.7 Execution Report (8)

Execution Report (35=8) is used to:

- confirm the receipt of an order
- confirm changes to an existing order (i.e. accept cancel and replace requests)
- confirm or convey an order cancellation or expiration
- convey order or trade cancellation by Market Operations
- convey fill information
- convey triggering of a stop order
- reject orders
- *convey speed bump processing*
- convey information about restated long orders carried from one trading day to the next.

ExecType (150) identifies the purpose of the execution report message and OrdStatus (39) conveys the current state of the order.

The attributes that can be returned in an Execution Report for each execution type are listed in the Execution Report Matrix.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 37 | OrderID | Y | String (19) | A unique order identifier set by the trading system. This identifier is not changed by cancel/replace messages; it will remain the same for all chain of orders. |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 11 | ClOrdID | Y* | String (18) | Client specified identifier in the message that caused this Execution Report. |
| 41 | OrigClOrdID | C | String (18) | ClOrdID (11) of the previous order (NOT the initial order of the day) as assigned by the institution. Identifies the previous order in cancel and cancel/replace requests.<br>Conditionally required according to Execution Report Matrix. |
| Component Block <Parties> | | Y* | See Parties Component Block | |
| 880 | TrdMatchID | C* | String (19) | Identifier assigned by the trading system which joins buy and sell half trades.<br>Conditionally required if ExecType (150) = ‘F’ Trade. |
| 17 | ExecID | Y | String (19) | Unique identifier assigned by the trading system to the execution message. |
| 19 | ExecRefID | C* | String (19) | Reference identifier used with Trade Cancel execution type.<br>Conditionally required if ExecType (150) = ‘H’ Trade Cancel. |
| 150 | ExecType | Y | Char | Describes the specific Execution Report.<br>Valid values:<br>0 = New<br>3 = Done<br>4 = Cancelled<br>5 = Replaced<br>8 = Rejected<br>C = Expired<br>D = Restated<br>*E = Pending Replace*<br>F = Trade |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | H = Trade Cancel<br>L = Triggered or Activated by the System |
| 39 | OrdStatus | Y | Char | Identifies current status of order.<br>Valid values:<br>0 = New<br>1 = Partially Filled<br>2 = Filled<br>3 = Done for day<br>4 = Cancelled<br>6 = Pending Cancel<br>8 = Rejected<br>A = Pending New<br>C = Expired<br>E = Pending Replace |
| 103 | OrdRejReason | C* | Int | Code to identify reason for order rejection.<br>Conditionally required if ExecType (150) = ‘8’ Rejected<br>Valid values:<br>6 = Duplicate Order<br>15 = Unknown Account(s)<br>18 = Invalid price increment<br>99 = Other |
| 378 | ExecRestatementReason | C* | Int | Conditionally required if ExecType (150) = ‘D’ Restated.<br>The reason for restatement.<br>Valid values:<br>1 = GT renewal / restatement<br>*99 = Other. See ExecTypeReason (2431) for speed bump handling.* |
| 581 | AccountType | Y* | Int | Specifies the type of account associated with the order.<br>Valid values:<br>1 = Client ISA<br>3 = House |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | 8 = Joint back office account (JBO) = Gross OSA<br>101 = Client OSA<br>For contracts assigned to the T4 booking model only 3 = House is valid whereas for the T2 booking model all account types are valid. |
| 1115 | OrderCategory | C* | Char | Conditionally required for a trade from an implied order when ExecType (150) = ‘F’ Trade.<br>Defines the type of interest behind a trade (fill or partial fill).<br>Valid value:<br>7 = Implied Order |
| Component Block <Instrument> | | | | |
| 48 | SecurityID | Y* | Int | Tradable instrument identifier |
| 22 | SecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when SecurityID (48) is specified. |
| End Component Block | | | | |
| 54 | Side | Y | Char | Side of the order<br>Valid values:<br>1 = Buy<br>2 = Sell |
| Component Block <OrderQtyData> | | | | |
| 38 | OrderQty | Y* | Qty | Total order quantity of the order. |
| End Component Block | | | | |
| 40 | OrdType | Y* | Char | Order type applicable to the order.<br>Valid values: |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | *1 = Market*<br>2 = Limit<br>*3 = Stop Market*<br>4 = Stop Limit |
| 44 | Price | C | Price (20) | The order price.<br>Conditionally required if OrdType (40) = ‘2’ Limit or ‘4’ Stop Limit. |
| 99 | StopPx | C* | Price (20) | The Stop trigger price. Conditionally required if OrdType (40) = *‘3’ Stop Market* or ‘4’ Stop Limit.<br>TriggerPriceType (1107) is required if StopPx is specified. |
| Component Block <TriggeringInstruction> | | | | |
| 1100 | TriggerType | C* | Char | Trigger prompt for the stop order elements.<br>Conditionally required if any other Triggering tags are specified.<br>Valid value:<br>4 = Price Movement |
| *1102* | *TriggerPrice* | *C** | *Price (20)* | *Stop order price of the OCO.*<br>*Conditionally required for an OCO.* |
| 1107 | TriggerPriceType | C* | Char | Type of price event that triggers the stop order:<br>Valid values:<br>2 = Last Trade<br>4 = Best Bid or Last Trade<br>5 = Best Offer or Last Trade<br>Conditionally required if StopPx (99) *or TriggerPrice (1102)* is specified |
| *1110* | *TriggerNewPrice* | *C** | *Price (20)* | *Limit order price of the stop once triggered.* |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | *Conditionally required if TriggerOrderType (1111) = '2' Limit* |
| *1111* | *TriggerOrderType* | *C** | *Char* | *Order type of the order once triggered.*<br>*Valid values:*<br>*1 = Market*<br>*2 = Limit*<br>*Conditionally required for an OCO.* |
| End Component Block | | | | |
| 59 | TimeInForce | Y* | Char | Specifies how long the order remains in effect.<br>Valid values:<br>0 = Day<br>1 = Good Till cancel (GTC)<br>3 = Immediate or cancel (IOC)<br>*4 = Fill or Kill*<br>6 = Good Till Date (GTD) |
| 432 | ExpireDate | C | LocalMktDate | The expiry date of an order.<br>Conditionally required if TimeInForce (59) = ‘0’ Day or ‘6’ Good Till Date~~ is not specified~~.<br>Format is YYYYMMDD. |
| 18 | ExecInst | C* | MultipleCharValue | Instructions for order handling. If more than one instruction is applicable to an order, this field can contain multiple instructions separated by space.<br>Valid values:<br>*6 = Participate but don’t initiate for Post Only orders*<br>o = Cancel on connection loss<br>Conditionally required according to Execution Report Matrix. |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 1057 | AggressorIndicator | C* | Boolean | Indicates if a matching order is an aggressor or not in the trade.<br>Y = Aggressor<br>N = Passive<br>Conditionally required if ExecType (150) = ‘F’ Trade. |
| 528 | OrderCapacity | Y* | Char | Designates the capacity of the firm placing the order.<br>Valid values:<br>A (agency) = AOTC<br>P (principal) = DEAL<br>R (riskless principal) = MTCH |
| 529 | OrderRestrictions | Y* | MultipleCharValue | Indicates if the order is entered either by an algo trader or a human.<br>Valid values:<br>D = Non-algorithmic (human)<br>E = Algorithmic (algo) |
| 32 | LastQty | C | Qty | Conditionally required if ExecType (150) = ‘F’ Trade.<br>The total volume of this trade. |
| 31 | LastPx | C | Price (20) | Conditionally required if ExecType (150) = ‘F’ Trade.<br>The price of this trade. |
| 151 | LeavesQty | Y | Qty | The quantity open for further execution.<br>If OrdStatus (39) = '4' Cancelled, 'C' Expired or '8' Rejected then LeavesQty (151) could be 0 otherwise LeavesQty (151) will be OrderQty (38) - CumQty (14) |
| 14 | CumQty | Y | Qty | The quantity of the order that has been executed so far. |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 60 | TransactTime | Y* | UTCTimestamp | Timestamp when the message was generated. |
| *Component Block <DisplayInstruction>* | | | | |
| *1138* | *DisplayQty* | *C** | *Qty* | *Visible quantity for Iceberg orders.*<br>*Conditionally required for an Iceberg order.* |
| *End Component Block* | | | | |
| 58 | Text | C* | String (50) | Contains the value supplied in this field on the order.<br>Conditionally required according to Execution Report Matrix. |
| Component Block <InstrmtLegExecGrp> | | | | |
| 555 | NoLegs | C* | NumInGrp (2) | Conditionally required if ExecType (150) = ‘F’ Trade on a multileg tradable instrument.<br>Number of InstrumentLeg repeating group instances. |
| Component Block <InstrumentLeg> - Required if NoLegs (555) > 0. | | | | |
| >602 | LegSecurityID | C* | Int | Conditionally required if ExecType (150) = ‘F’ Trade on a multileg tradable instrument.<br>Multileg tradable instrument's individual SecurityID. |
| >603 | LegSecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when LegSecurityID (602) is specified. |
| >624 | LegSide | C* | Char | Conditionally required if ExecType (150) = ‘F’ Trade on a multileg tradable instrument. |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | The side of this individual leg (multileg security).<br>Valid values:<br>1 = Buy<br>2 = Sell |
| End Component Block | | | | |
| >1366 | LegAllocID | C* | String (19) | Strategy leg trade identifier assigned by the trading system which is shared by half trades.<br>Conditionally required if ExecType (150) = ‘F’ Trade on a multileg tradable instrument. |
| >637 | LegLastPx | C* | Price (20) | Conditionally required if ExecType (150) = ‘F’ Trade on a multileg tradable instrument.<br>Execution price assigned to the leg of the multileg tradable instrument. |
| >1418 | LegLastQty | C* | Qty | Conditionally required if ExecType (150) = ‘F’ Trade on a multileg tradable instrument.<br>Fill quantity for the instrument leg. |
| End Component Block | | | | |
| 1328 | RejectText | C* | String (75) | Identifies the reason for rejection.<br>Conditionally required if ExecTypeReason (2431) = '4' Unsolicited order cancellation or OrdRejReason (103) = ‘99’ Other. |
| 1724 | OrderOrigination | C* | Int | Origin of the order<br>Valid value:<br>5 = Order received from a direct access or sponsored access |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | customer (the trader has direct electronic access – DEA)<br>Absence of this field indicates DEA = false.<br>Conditionally required according to Execution Report Matrix. |
| 2431 | ExecTypeReason | C* | Int | The initiating event for the Execution Report.<br>Conditionally required to report unsolicited cancellation and *order status in speed bump processing*.<br>Valid values:<br>4 = Unsolicited order cancellation<br>*101 = Order accepted but speed bump applied*<br>*102 = Order added after speed bump*<br>*103 = Order cancelled whilst in speed bump delay*<br>*104 = Original order is in speed bump enforced delay*<br>*105 = Order updated after speed bump delay*<br>*106 = Amend is in speed bump delay*<br>*107 = Order amended after speed bump delay*<br>*108 = Order rejected after speed bump delay*<br>*109 = Unsolicited cancel while in speed bump* |
| *2362* | *SelfMatchPreventionID* | *C** | *Int (9)* | *Identifies an order that should not be matched to an opposite order if both buy and sell orders for the trade contain the same SelfMatchPreventionID (2362) and are submitted by the same member.*<br>*Conditionally required according to Execution Report Matrix.* |

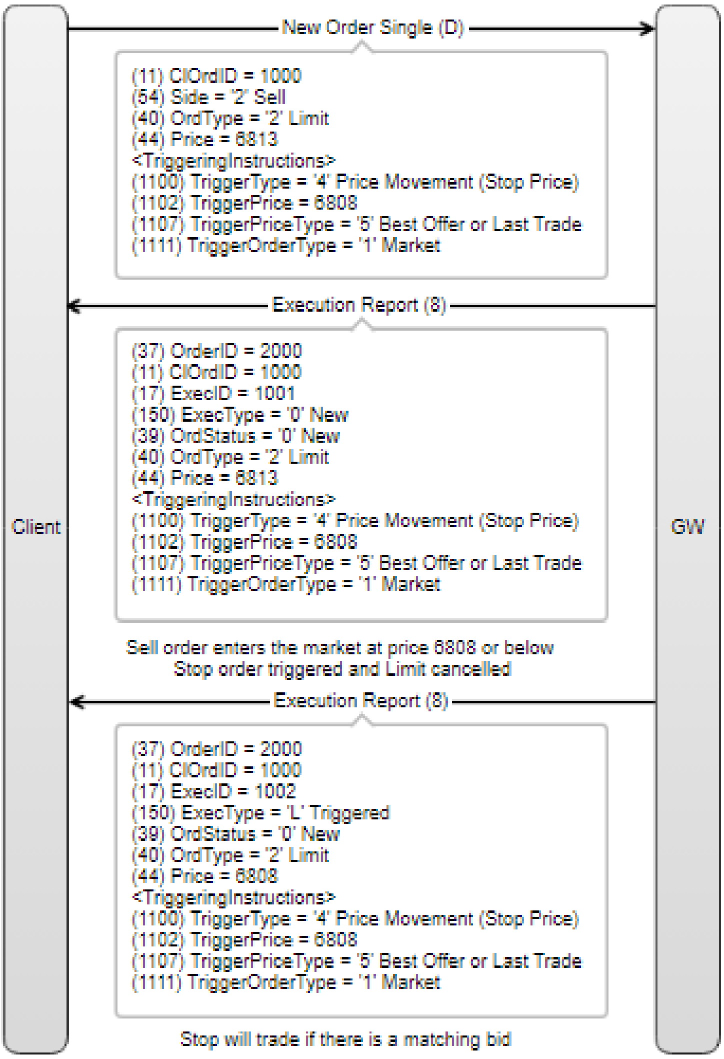
| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| Component Block <OrderAttributeGrp> - Conditionally required if specified. | | | | |
| 2593 | NoOrderAttributes | C* | NumInGrp (1) | Number of order attribute entries. |
| >2594 | OrderAttributeType | C* | Int | The type of order attribute, see Order Attribute Type Usage.<br>Conditionally required if NoOrderAttributes (2593) > 0.<br>Valid values:<br>0 = Aggregated order. In the context of ESMA RTS 24 Article 2(3), when OrderAttributeValue (2595) = Y, it signifies that the order consists of several orders aggregated together. This maps to ESMA RTS value "AGGR".<br>1 = Pending allocation. In the context of ESMA RTS 24 Article 2(2), when OrderAttributeValue (2595) = Y, it signifies that the order submitter "is authorized under the legislation of a Member State to allocate an order to its client following submission of the order to the trading venue and has not yet allocated the order to its client at the time of the submission of the order". This maps to ESMA RTS value "PNAL".<br>2 = Liquidity Provision Order. In the context of ESMA RTS 24 Article 3, when OrderAttributeValue (2595) = Y, it signifies that the order was submitted "as part of a market making strategy pursuant to Articles 17 and 18 of Directive 2014/65/EU or is submitted as part of another activity in accordance with Article 3" (of RTS 24). |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | 3 = Risk Reduction Order. In the context of ESMA RTS 22 Article 4(2)(i), when OrderAttributeValue (2595) = Y, it signifies that the commodity derivative order is a transaction "to reduce risk in an objectively measurable way in accordance with Article 57 of Directive 2014/65/EU". |
| >2595 | OrderAttributeValue | C* | String (1) | The value associated with the order attribute type specified in OrderAttributeType (2594).<br>Conditionally required if NoOrderAttributes (2593) > 0.<br>Valid value:<br>Y = Yes |
| End Component Block | | | | |
| 1819 | RelatedHighPrice | C* | Price | Upper price limit value<br>For Stop orders, this will be the stop tolerance band. |
| 1820 | RelatedLowPrice | C* | Price | Lower price limit value |

## Example Message Flows

*OCO submitted, Stop triggered and Limit cancelled*

*An OCO order is submitted as a Limit offer with a Market Stop trigger price of 6808, an incoming offer triggers the Stop order and cancels Limit element of the OCO. An Execution Report is not sent for cancellation. The triggered Market Stop is converted to a Limit order at a trigger new price of 6808.*



New Order Single (D)

(11) ClOrdID = 1000
(54) Side = '2' Sell
(40) OrdType = '2' Limit
(44) Price = 6813
<TriggeringInstructions>
(1100) TriggerType = '4' Price Movement (Stop Price)
(1102) TriggerPrice = 6808
(1107) TriggerPriceType = '5' Best Offer or Last Trade
(1111) TriggerOrderType = '1' Market

Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1001
(150) ExecType = '0' New
(39) OrdStatus = '0' New
(40) OrdType = '2' Limit
(44) Price = 6813
<TriggeringInstructions>
(1100) TriggerType = '4' Price Movement (Stop Price)
(1102) TriggerPrice = 6808
(1107) TriggerPriceType = '5' Best Offer or Last Trade
(1111) TriggerOrderType = '1' Market

Sell order enters the market at price 6808 or below
Stop order triggered and Limit cancelled

Execution Report (8)

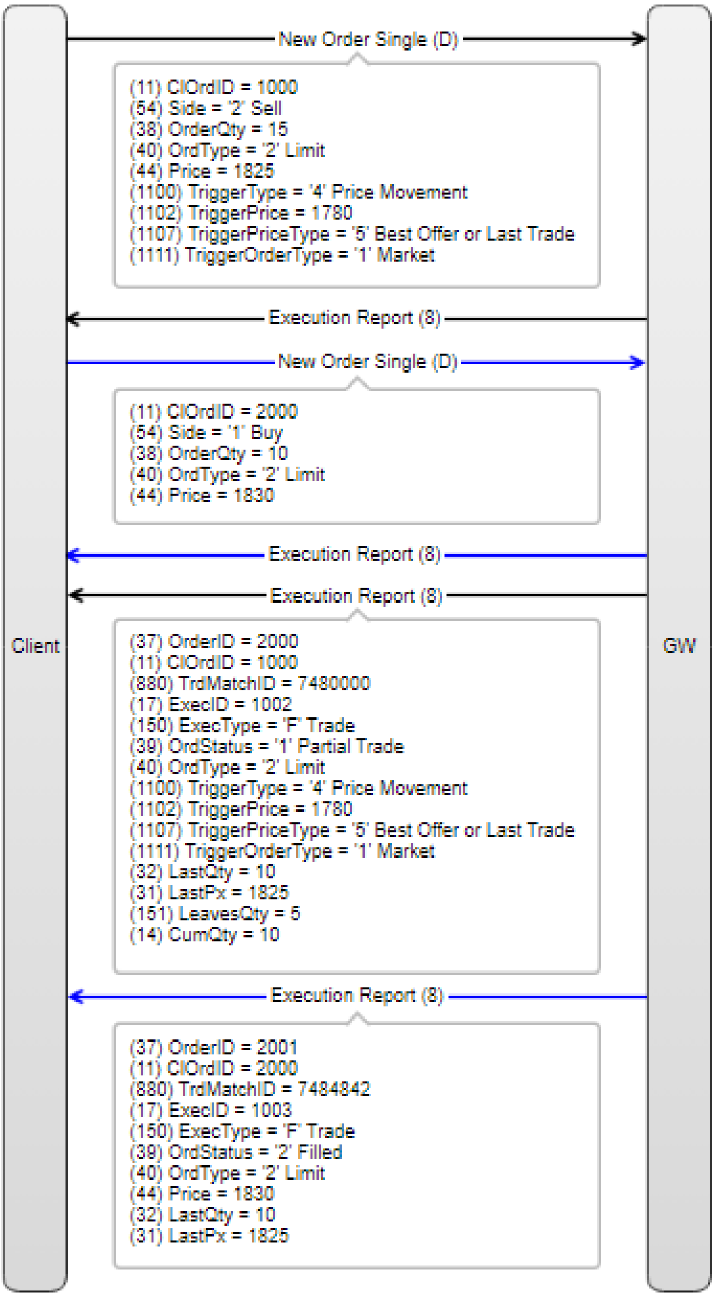
(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1002
(150) ExecType = 'L' Triggered
(39) OrdStatus = '0' New
(40) OrdType = '2' Limit
(44) Price = 6808
<TriggeringInstructions>
(1100) TriggerType = '4' Price Movement (Stop Price)
(1102) TriggerPrice = 6808
(1107) TriggerPriceType = '5' Best Offer or Last Trade
(1111) TriggerOrderType = '1' Market

Stop will trade if there is a matching bid

*OCO Partial Trade*

*OCO is submitted as Limit offer for 15 lots at 1825 with a Market Stop trigger price of 1780.*

*A Limit bid is submitted at 1830 for 10 lots. The OCO order is not triggered but trades 10 lots with the incoming order. The OCO remains in the order book with a residual volume of 5 lots*



New Order Single (D)

(11) ClOrdID = 1000
(54) Side = '2' Sell
(38) OrderQty = 15
(40) OrdType = '2' Limit
(44) Price = 1825
(1100) TriggerType = '4' Price Movement
(1102) TriggerPrice = 1780
(1107) TriggerPriceType = '5' Best Offer or Last Trade
(1111) TriggerOrderType = '1' Market

Execution Report (8)

New Order Single (D)

(11) ClOrdID = 2000
(54) Side = '1' Buy
(38) OrderQty = 10
(40) OrdType = '2' Limit
(44) Price = 1830

Execution Report (8)

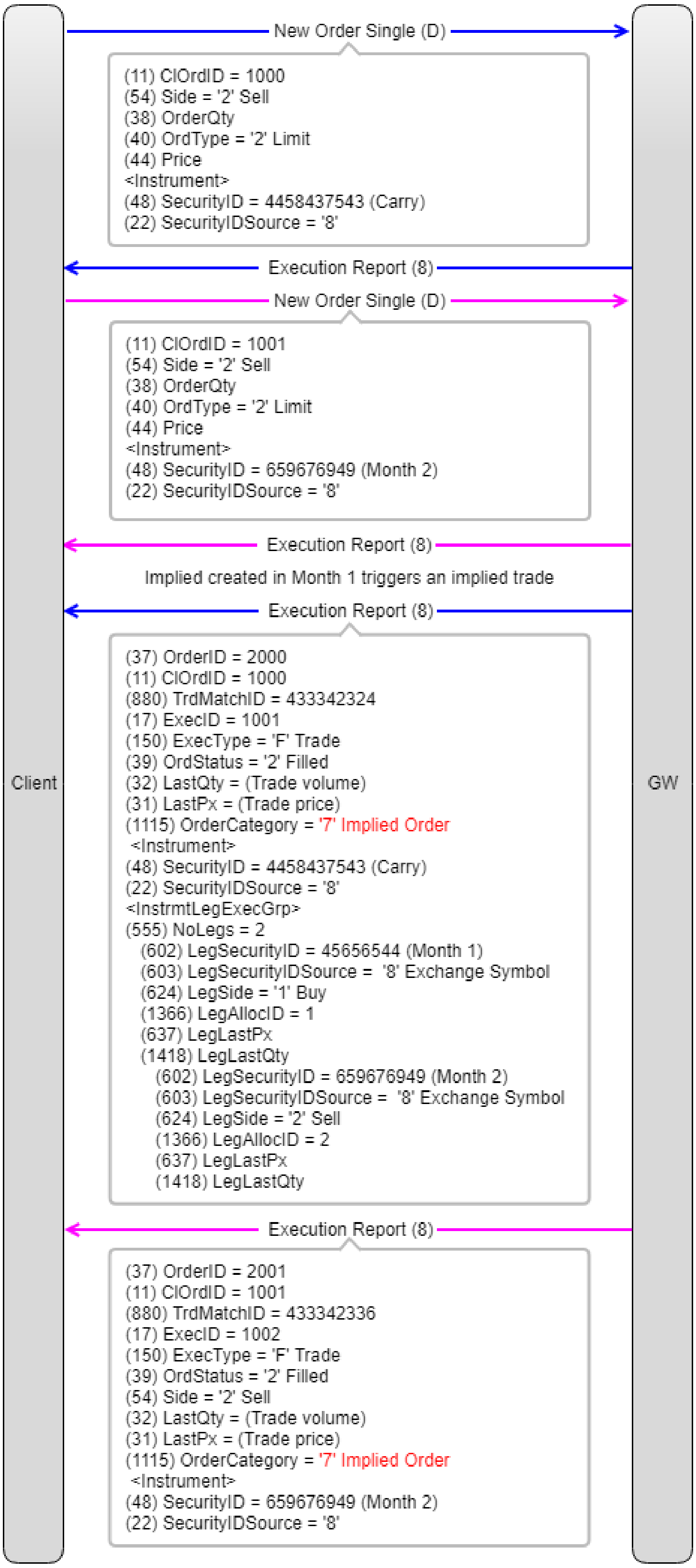
Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(880) TrdMatchID = 7480000
(17) ExecID = 1002
(150) ExecType = 'F' Trade
(39) OrdStatus = '1' Partial Trade
(40) OrdType = '2' Limit
(1100) TriggerType = '4' Price Movement
(1102) TriggerPrice = 1780
(1107) TriggerPriceType = '5' Best Offer or Last Trade
(1111) TriggerOrderType = '1' Market
(32) LastQty = 10
(31) LastPx = 1825
(151) LeavesQty = 5
(14) CumQty = 10

Execution Report (8)

(37) OrderID = 2001
(11) ClOrdID = 2000
(880) TrdMatchID = 7484842
(17) ExecID = 1003
(150) ExecType = 'F' Trade
(39) OrdStatus = '2' Filled
(40) OrdType = '2' Limit
(44) Price = 1830
(32) LastQty = 10
(31) LastPx = 1825

*Implied trade*



New Order Single (D)

(11) ClOrdID = 1000
(54) Side = '2' Sell
(38) OrderQty
(40) OrdType = '2' Limit
(44) Price
<Instrument>
(48) SecurityID = 4458437543 (Carry)
(22) SecurityIDSource = '8'

Execution Report (8)

New Order Single (D)

(11) ClOrdID = 1001
(54) Side = '2' Sell
(38) OrderQty
(40) OrdType = '2' Limit
(44) Price
<Instrument>
(48) SecurityID = 659676949 (Month 2)
(22) SecurityIDSource = '8'

Execution Report (8)

Implied created in Month 1 triggers an implied trade

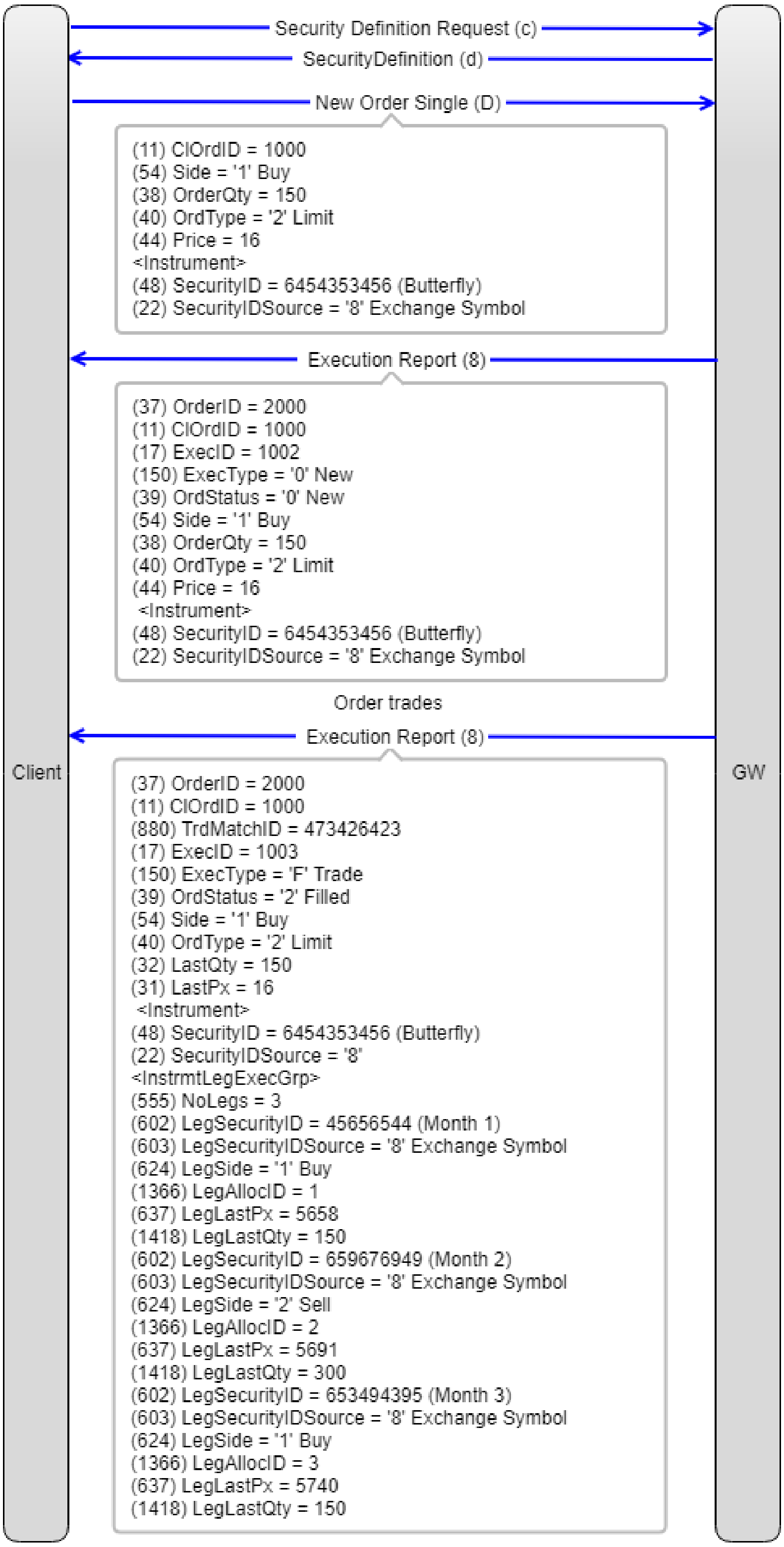
Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(880) TrdMatchID = 433342324
(17) ExecID = 1001
(150) ExecType = 'F' Trade
(39) OrdStatus = '2' Filled
(32) LastQty = (Trade volume)
(31) LastPx = (Trade price)
(1115) OrderCategory = '7' Implied Order
<Instrument>
(48) SecurityID = 4458437543 (Carry)
(22) SecurityIDSource = '8'
<InstrmtLegExecGrp>
(555) NoLegs = 2
  (602) LegSecurityID = 45656544 (Month 1)
  (603) LegSecurityIDSource = '8' Exchange Symbol
  (624) LegSide = '1' Buy
  (1366) LegAllocID = 1
  (637) LegLastPx
  (1418) LegLastQty
    (602) LegSecurityID = 659676949 (Month 2)
    (603) LegSecurityIDSource = '8' Exchange Symbol
    (624) LegSide = '2' Sell
    (1366) LegAllocID = 2
    (637) LegLastPx
    (1418) LegLastQty

Execution Report (8)

(37) OrderID = 2001
(11) ClOrdID = 1001
(880) TrdMatchID = 433342336
(17) ExecID = 1002
(150) ExecType = 'F' Trade
(39) OrdStatus = '2' Filled
(54) Side = '2' Sell
(32) LastQty = (Trade volume)
(31) LastPx = (Trade price)
(1115) OrderCategory = '7' Implied Order
<Instrument>
(48) SecurityID = 659676949 (Month 2)
(22) SecurityIDSource = '8'

*Custom strategy Butterfly trades*



Security Definition Request (c)

SecurityDefinition (d)

New Order Single (D)

(11) ClOrdID = 1000
(54) Side = '1' Buy
(38) OrderQty = 150
(40) OrdType = '2' Limit
(44) Price = 16
<Instrument>
(48) SecurityID = 6454353456 (Butterfly)
(22) SecurityIDSource = '8' Exchange Symbol

Execution Report (8)

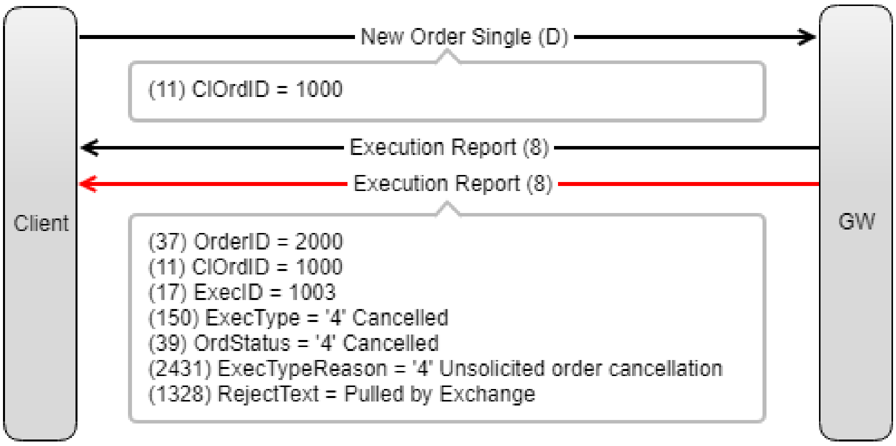
(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1002
(150) ExecType = '0' New
(39) OrdStatus = '0' New
(54) Side = '1' Buy
(38) OrderQty = 150
(40) OrdType = '2' Limit
(44) Price = 16
<Instrument>
(48) SecurityID = 6454353456 (Butterfly)
(22) SecurityIDSource = '8' Exchange Symbol

Order trades

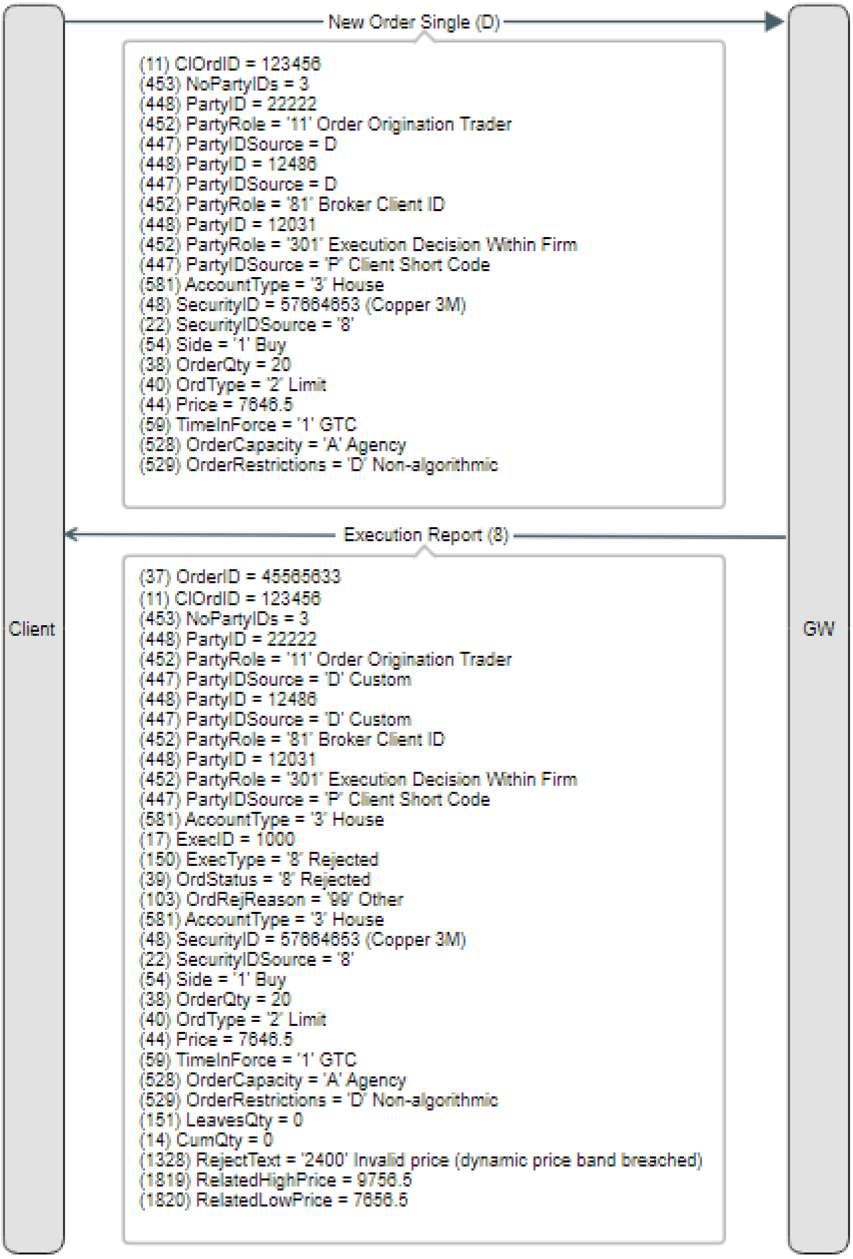
Execution Report (8)

(37) OrderID = 2000
(11) ClOrdID = 1000
(880) TrdMatchID = 473426423
(17) ExecID = 1003
(150) ExecType = 'F' Trade
(39) OrdStatus = '2' Filled
(54) Side = '1' Buy
(40) OrdType = '2' Limit
(32) LastQty = 150
(31) LastPx = 16
<Instrument>
(48) SecurityID = 6454353456 (Butterfly)
(22) SecurityIDSource = '8'
<InstrmtLegExecGrp>
(555) NoLegs = 3
(602) LegSecurityID = 45656544 (Month 1)
(603) LegSecurityIDSource = '8' Exchange Symbol
(624) LegSide = '1' Buy
(1366) LegAllocID = 1
(637) LegLastPx = 5658
(1418) LegLastQty = 150
(602) LegSecurityID = 659676949 (Month 2)
(603) LegSecurityIDSource = '8' Exchange Symbol
(624) LegSide = '2' Sell
(1366) LegAllocID = 2
(637) LegLastPx = 5691
(1418) LegLastQty = 300
(602) LegSecurityID = 653494395 (Month 3)
(603) LegSecurityIDSource = '8' Exchange Symbol
(624) LegSide = '1' Buy
(1366) LegAllocID = 3
(637) LegLastPx = 5740
(1418) LegLastQty = 150

*Order cancellation by Exchange*


Client
GW
New Order Single (D)
(11) ClOrdID = 1000
Execution Report (8)
Execution Report (8)
(37) OrderID = 2000
(11) ClOrdID = 1000
(17) ExecID = 1003
(150) ExecType = '4' Cancelled
(39) OrdStatus = '4' Cancelled
(2431) ExecTypeReason = '4' Unsolicited order cancellation
(1328) RejectText = Pulled by Exchange

*Order rejected price limits breached*



```
(11) ClOrdID = 123456
(453) NoPartyIDs = 3
(448) PartyID = 22222
(452) PartyRole = '11' Order Origination Trader
(447) PartyIDSource = D
(448) PartyID = 12486
(447) PartyIDSource = D
(452) PartyRole = '81' Broker Client ID
(448) PartyID = 12031
(452) PartyRole = '301' Execution Decision Within Firm
(447) PartyIDSource = 'P' Client Short Code
(581) AccountType = '3' House
(48) SecurityID = 57664653 (Copper 3M)
(22) SecurityIDSource = '8'
(54) Side = '1' Buy
(38) OrderQty = 20
(40) OrdType = '2' Limit
(44) Price = 7646.5
(59) TimeInForce = '1' GTC
(528) OrderCapacity = 'A' Agency
(529) OrderRestrictions = 'D' Non-algorithmic
```

```
(37) OrderID = 45565633
(11) ClOrdID = 123456
(453) NoPartyIDs = 3
(448) PartyID = 22222
(452) PartyRole = '11' Order Origination Trader
(447) PartyIDSource = 'D' Custom
(448) PartyID = 12486
(447) PartyIDSource = 'D' Custom
(452) PartyRole = '81' Broker Client ID
(448) PartyID = 12031
(452) PartyRole = '301' Execution Decision Within Firm
(447) PartyIDSource = 'P' Client Short Code
(581) AccountType = '3' House
(17) ExecID = 1000
(150) ExecType = '8' Rejected
(39) OrdStatus = '8' Rejected
(103) OrdRejReason = '99' Other
(581) AccountType = '3' House
(48) SecurityID = 57664653 (Copper 3M)
(22) SecurityIDSource = '8'
(54) Side = '1' Buy
(38) OrderQty = 20
(40) OrdType = '2' Limit
(44) Price = 7646.5
(59) TimeInForce = '1' GTC
(528) OrderCapacity = 'A' Agency
(529) OrderRestrictions = 'D' Non-algorithmic
(151) LeavesQty = 0
(14) CumQty = 0
(1328) RejectText = '2400' Invalid price (dynamic price band breached)
(1819) RelatedHighPrice = 9756.5
(1820) RelatedLowPrice = 7656.5
```

### 4.11.7.1 Execution Report Matrix

An Execution Report can be returned in response to a request e.g. New Order Single (35=D) or unsolicited in response to a particular action.

The tags that can be included are contingent on the purpose of the message and any mandatory or conditionally supplied tags specified by the originator in the initiating request or returned response to a particular action.

**Legend:**

M = Mandatory

C = Conditional

P = Returned in first outbound message if present in original message.

The following table indicates the tags that will be returned for specific execution types:

| Tag | Order Accepted | Order Restated (GTC/GTD) | Order Triggered | *Order Restated (Speed Bump)* | Order Replaced | *Order Pending Replace* | *Order Replaced (after Speed Bump)* | Order Cancelled (Solicited) | Order Cancelled (Unsolicited) | Order Expired | Done for Day | Outright Filled | Strategy Filled | Trade Busted | Order Rejected |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OrderID (37) | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| ClOrdID (11) | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| OrigClOrdID (41) | | | | | M | *M* | *M* | M | | | | | | | |
| Parties component block<br>PartyRole (452) | | | | | | | | | | | | | | | |
| 1 = Executing Firm | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |

| Tag | Order Accepted | Order Restated (GTC/GTD) | Order Triggered | *Order Restated (Speed Bump)* | Order Replaced | *Order Pending Replace* | *Order Replaced (after Speed Bump)* | Order Cancelled (Solicited) | Order Cancelled (Unsolicited) | Order Expired | Done for Day | Outright Filled | Strategy Filled | Trade Busted | Order Rejected |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 = Client ID | P | P | | | | | | | | | | P | P | P | P |
| 4 = Clearing Firm | | | | | | | | | | | | M | M | M | |
| 7 = Entering Firm | P | P | | | | | | | | | | P | P | P | P |
| 11 = Order Origination Trader | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| 24 = Customer Account | P | P | | | | | | | | | | P | P | P | P |
| 26 = Correspondent broker | P | P | | | | | | | | | | P | P | P | P |
| 36 = Entering Trader | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| 66 = Market Maker | P | P | | | | | | | | | | P | P | P | P |
| 81 = Broker Client ID | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| 122 = Decision Maker | P | P | | | | | | | | | | P | P | P | P |
| 300 = Investment Decision Within Firm | P | P | | | P | | *P* | | | | | P | P | P | P |
| 301 = Execution Decision Within Firm | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |

| Tag | Order Accepted | Order Restated (GTC/GTD) | Order Triggered | *Order Restated (Speed Bump)* | Order Replaced | *Order Pending Replace* | *Order Replaced (after Speed Bump)* | Order Cancelled (Solicited) | Order Cancelled (Unsolicited) | Order Expired | Done for Day | Outright Filled | Strategy Filled | Trade Busted | Order Rejected |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 302 = Investment Decision Country | P | P | | | P | | *P* | | | | | P | P | P | P |
| 303 = Execution Decision Country | P | P | | | P | | *P* | | | | | P | P | P | P |
| 304 = Client Branch Country | P | P | | | P | | *P* | | | | | P | P | P | P |
| TrdMatchID (880) | | | | | | | | | | | | M | M | M | |
| ExecID (17) | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| ExecRefID (19) | | | | | | | | | | | | | | M | |
| ExecType (150) | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| OrdStatus (39) | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| OrdRejReason (103) | | | | | | | | | | | | | | | M |
| ExecRestatementReason (378) | | M | | *M* | | | | | | | | | | | |
| AccountType (581) | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| OrderCategory (1115) | | | | | | | | | | | | C | C | C | |

| Tag | Order Accepted | Order Restated (GTC/GTD) | Order Triggered | *Order Restated (Speed Bump)* | Order Replaced | *Order Pending Replace* | *Order Replaced (after Speed Bump)* | Order Cancelled (Solicited) | Order Cancelled (Unsolicited) | Order Expired | Done for Day | Outright Filled | Strategy Filled | Trade Busted | Order Rejected |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SecurityID (48) | M | M | M | *M* | M | *M* | M | M | M | M | M | M | M | M | M |
| SecurityIDSource (22) | C | C | C | *C* | C | *C* | *C* | C | C | C | C | C | C | C | C |
| Side (54) | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| OrderQty (38) | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| OrdType (40) | M | M | M | *M* | M | *M* | *M* | M | M | M | M | M | M | M | M |
| Price (44) | C | C | M | *C* | C | *C* | *C* | C | C | C | C | M | M | M | C |
| StopPx (99) | C | C[1] | M |  | C | *C* | *C* | C | C | C | C | C | C | C | C |
| TriggerType (1100) | C | C[1] | M | *C* | C | *C* | *C* | C | C | C | C | C | C | C | C |
| *TriggerPrice (1102)* | *C* | *C*[1] | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* |
| TriggerPriceType (1107) | C | C[1] | M | *C* | C | *C* | *C* | C | C | C | C | C | C | C | C |
| *TriggerNewPrice (1110)* | *C* | *C*[1] | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* |
| *TriggerOrderType (1111)* | *C* | *C*[1] | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* |

[1] StopPx (99) and TriggeringInstruction component block tags will not be present when a previously triggered Stop order is restated as a Limit order.

| Tag | Order Accepted | Order Restated (GTC/GTD) | Order Triggered | *Order Restated (Speed Bump)* | Order Replaced | *Order Pending Replace* | *Order Replaced (after Speed Bump)* | Order Cancelled (Solicited) | Order Cancelled (Unsolicited) | Order Expired | Done for Day | Outright Filled | Strategy Filled | Trade Busted | Order Rejected |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TimeInForce (59) | M | M | M | *M* | M | M | *M* | M | M | M | M | M | M | M | M |
| ExpireDate (432) | C | C | C | *C* | C | C | *C* | C | C | C | C | C | C | C | C |
| ExecInst (18) | C | C | C | *C* | C | C | *C* | C | C | C | C | C | C | C | C |
| AggressorIndicator (1057) | | | | | | | | | | | | M | M | M | |
| OrderCapacity (528) | M | M | M | *M* | M | M | *M* | M | M | M | M | M | M | M | M |
| OrderRestrictions (529) | M | M | M | *M* | M | M | *M* | M | M | M | M | M | M | M | M |
| LastQty (32) | | | | | | | | | | | | M | M | M | |
| LastPx (31) | | | | | | | | | | | | M | M | M | |
| LeavesQty (151) | M | M | M | *M* | M | M | *M* | M | M | M | M | M | M | M | M |
| CumQty (14) | M | M | M | *M* | M | M | *M* | M | M | M | M | M | M | M | M |
| TransactTime (60) | M | M | M | *M* | M | M | *M* | M | M | M | M | M | M | M | M |
| *DisplayQty (1138)* | *C* | *C* | | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | *C* | | *C* | *C* |
| Text (58) | P | P | | | P | | *P* | | | | | P | P | P | P |

| Tag | Order Accepted | Order Restated (GTC/GTD) | Order Triggered | *Order Restated (Speed Bump)* | Order Replaced | *Order Pending Replace* | *Order Replaced (after Speed Bump)* | Order Cancelled (Solicited) | Order Cancelled (Unsolicited) | Order Expired | Done for Day | Outright Filled | Strategy Filled | Trade Busted | Order Rejected |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| NoLegs (555) | | | | | | | | | | | | | M | C | |
| LegSecurityID (602) | | | | | | | | | | | | | M | C | |
| LegSecurityIDSource (603) | | | | | | | | | | | | | M | C | |
| LegSide (624) | | | | | | | | | | | | | M | C | |
| LegAllocID (1366) | | | | | | | | | | | | | M | C | |
| LegLastPx (637) | | | | | | | | | | | | | M | C | |
| LegLastQty (1418) | | | | | | | | | | | | | M | C | |
| RejectText (1328) | | | | | | | | | C | | | | | | C |
| OrderOrigination (1724) | P | P | | | P | | ***P*** | | | | | P | P | P | P |
| ExecTypeReason (2431) | C | | | *M* | | M | *M* | C | M | | | | | | |
| *SelfMatchPreventionID (2362)* | *P* | *P* | | | | | | | *C* | | | | | | *P* |
| NoOrderAttributes (2593) | P | P | | | P | | *P* | | | | | P | P | P | P |

| Tag | Order Accepted | Order Restated (GTC/GTD) | Order Triggered | *Order Restated (Speed Bump)* | Order Replaced | *Order Pending Replace* | *Order Replaced (after Speed Bump)* | Order Cancelled (Solicited) | Order Cancelled (Unsolicited) | Order Expired | Done for Day | Outright Filled | Strategy Filled | Trade Busted | Order Rejected |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OrderAttributeType (2594) | P | P | | | P | | *P* | | | | | P | P | P | P |
| OrderAttributeValue (2595) | P | P | | | P | | *P* | | | | | P | P | P | P |
| RelatedHighPrice (1819) | | | | | | | | | C | | | | | | C |
| RelatedLowPrice (1820) | | | | | | | | | C | | | | | | C |

### 4.11.8 Order Mass Cancel Request (q)

Order Mass Cancel Request (35=q) is used to cancel the remaining quantity of a group of orders matching criteria specified within the message. Persisted orders will be included in the cancellation request. An Execution Report will be sent for each order cancelled followed by the Order Mass Cancel Report (35=r).

Order Mass Cancel Report will be returned if the request is accepted or rejected.

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 11 | ClOrdID | Y | String (18) | Unique ID of Order Mass Cancel Request as assigned by the institution.<br>This identifier will be returned in ClOrdID (11) in the Execution Report of each order cancelled. |
| 530 | MassCancelRequestType | Y | Char | Specifies the type of cancellation requested<br>Valid values:<br>1 = Cancel orders for a Security ID (tradable instrument)<br>3 = Cancel orders for a Product (contract e.g. CADF - Copper Future)<br>7 = Cancel all orders |
| Component Block <Parties> | | C* | | See Parties Component Block<br>Conditionally required for MassCancelRequestType (530) = '7' (Cancel all orders) to specify the PartyID (448) when cancelling orders for a specific end client, PartyRole (452) = '81' Broker Client ID.<br>If not specified, orders will be cancelled for the FIX Comp ID of the message originator. |
| Component Block <Instrument> | | | | |
| 207 | SecurityExchange | C* | Exchange (4) | Market which is used to identify the security:<br>XLME |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| | | | | Conditionally required if Symbol (55) is specified only valid for MassCancelRequestType (530) = ‘3’ Cancel orders for a Product. |
| 1227 | ProductComplex | C* | String (4) | Identifies an entire suite of products for a given market.<br>Valid values:<br>LME = Base<br>Conditionally required if Symbol (55) is specified only valid for MassCancelRequestType (530) = ‘3’ Cancel orders for a Product |
| 55 | Symbol | C* | String (20) | Symbol for the LME contract code e.g. CADF (Copper Future) or OCDF (Copper Monthly Average Future)<br>Conditionally required if MassCancelRequestType (530) = '3' Cancel orders for a Product |
| 48 | SecurityID | C* | Int | Tradable instrument identifier.<br>Conditionally required if MassCancelRequestType (530) = '1' Cancel orders for a Security ID. |
| 22 | SecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when SecurityID (48) is specified. |
| End Component Block | | | | |
| 54 | Side | N | Char | Optional qualifier to indicate the side of the market for which orders are to be cancelled. Can be used if MassCancelRequestType (530) = '3' Cancel orders for a Product.<br>Absence of this field indicates that orders are to be cancelled regardless of side. |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 60 | TransactTime | Y | UTCTimestamp | Timestamp when the message was generated. |

### 4.11.9 Order Mass Cancel Report (r)

Order Mass Cancel Report (35=r) is returned in response to an Order Mass Cancel Request (35=q).

Each affected order that is cancelled is acknowledged with a separate Execution Report (35=8).

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 11 | ClOrdID | N | String (18) | ClOrdID provided on the Order Mass Cancel Request. |
| 1369 | MassActionReportID | Y | String (20) | Unique Identifier for the Order Mass Cancel Report assigned by the system |
| 530 | MassCancelRequestType | Y | Char | Specifies the type of cancellation required:<br>Valid values:<br>1 = Cancel orders for a SecurityID<br>3 = Cancel orders for a Product (Symbol)<br>7 = Cancel all orders |
| 531 | MassCancelResponse | Y | Char | Indicates the action taken on the cancel request:<br>Valid values:<br>0 = Cancel request rejected<br>1 = Cancel orders for a SecurityID<br>3 = Cancel orders for a Product (Symbol)<br>7 = Cancel all orders |
| 532 | MassCancelRejectReason | C* | Int | Indicates why the Order Mass Cancel Request was rejected. Conditionally required if MassCancelResponse (531) = ‘0’ Cancel request rejected.<br>Valid values:<br>1 = Invalid or Unknown Security<br>3 = Invalid or Unknown Product<br>99 = Other |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 533 | TotalAffectedOrders | Y* | Int | Indicates the total number of orders affected by the Order Mass Cancel Request. |
| Component Block <Parties> | | C* | | See Parties Component Block<br>Conditionally required for MassCancelRequestType (530) = 7 (Cancel all orders) to specify the PartyID (448) when cancelling orders for a specific end client, PartyRole (452) = '81' Broker Client ID. |
| Component Block <Instrument> | | | | |
| 207 | SecurityExchange | C* | Exchange (4) | Market which is used to identify the security:<br>XLME<br>Conditionally required if Symbol (55) is specified. |
| 1227 | ProductComplex | C* | String (4) | Identifies an entire suite of products for a given market.<br>Valid values:<br>LME = Base<br>Conditionally required if Symbol (55) is specified |
| 55 | Symbol | C* | String (20) | Symbol for the LME contract code e.g. CADF (Copper Future) or OCDF (Copper Monthly Average Future)<br>Conditionally required if MassCancelRequestType (530) = '3' Cancel orders for a Product |
| 48 | SecurityID | C* | Int | Tradable instrument identifier.<br>Conditionally required if MassCancelRequestType (530) = '1' Cancel orders for a Security ID. |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 22 | SecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when SecurityID (48) is specified. |
| End Component Block | | | | |
| 54 | Side | N | Char | Optional qualifier to indicate the side of the market for which orders are to be cancelled. Can be used if MassCancelRequestType (530) = '3' Cancel orders for a Product.<br>Absence of this field indicates that orders are to be cancelled regardless of side. |
| 60 | TransactTime | Y* | UTCTimestamp | Timestamp when the message was generated. |
| 58 | Text | C* | String (75) | Identifies the reason for rejection.<br>Conditionally required if MassCancelRejectReason (532) = '99' Other. |

### *4.11.10 Quote Request (R)*

*Quote Request (35=R) is used to requests prices from market participants.*

*The Quote Request is disseminated via the Market Data service to market participants. If successful a Quote Response (35=AJ) is sent in acknowledgement otherwise a Quote Request Reject (AG) is returned if the request is rejected.*

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| *131* | *QuoteReqID* | *Y* | *String (18)* | *Client specified identifier for quote request.* |
| *Component Block <QuotReqGrp>* | | | | |
| *146* | *NoRelatedSym* | *Y* | *NumInGrp (1)* | *Number of related symbols (instruments) in this request.*<br>*The value can only be 1.* |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| Component Block <Instrument> | | | | |
| >48 | SecurityID | Y* | Int | Tradable instrument identifier |
| >22 | SecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when SecurityID (48) is specified. |
| End Component Block | | | | |
| >303 | QuoteRequestType | Y* | Int | Indicates the type of Quote Request being generated.<br>Valid values:<br>1 = Manual - used to indicate a single quote request<br>2 = Automatic - used to indicate a streaming quote request |
| >54 | Side | N | Char | Side of order. If not defined indicates a two-sided quote is required.<br>Valid values:<br>1 = Buy<br>2 = Sell |
| >60 | TransactTime | Y | UTCTimestamp | Timestamp when the message was generated. |
| Component Block <OrderQtyData> | | | | |
| >38 | OrderQty | N | Qty | Order quantity.<br>If not entered, a volume of 0 will be published |
| End Component Blocks | | | | |

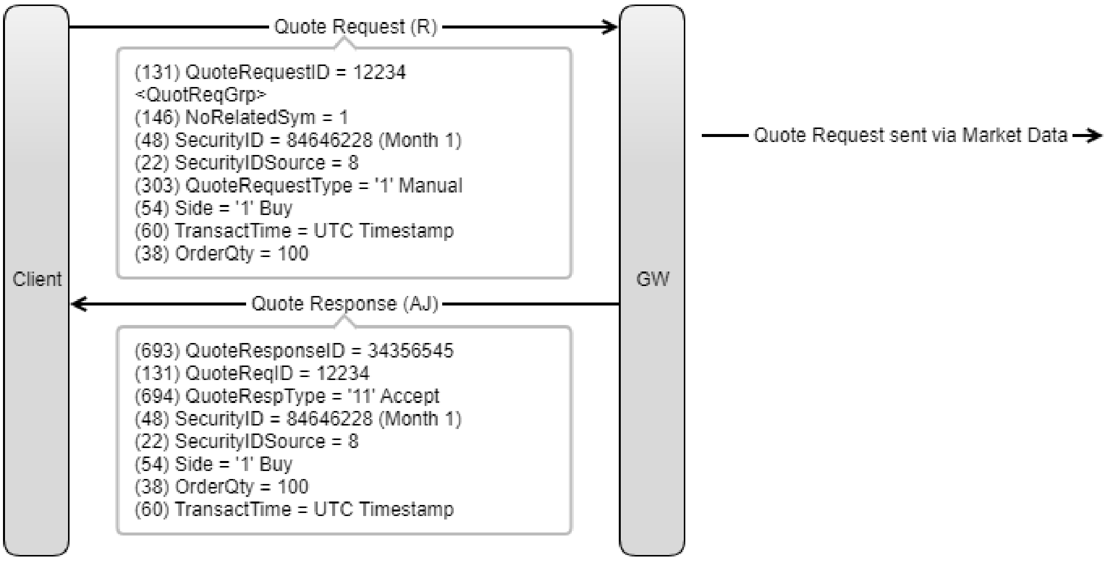
### 4.11.11 Quote Response (AJ)

Quote Response (35=AJ) is returned by the gateway to acknowledge a successful Quote Request (35=R).

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| 693 | QuoteResponseID | Y | String (18) | Message reference for the Quote Response. |
| 131 | QuoteReqID | Y | String (18) | Identifier supplied on Quote Request (R) message. |
| 694 | QuoteRespType | Y | Int | Identifies the type of Quote Response.<br>Valid value:<br>11 = Accept |
| Component Block <Instrument> | | | | |
| 48 | SecurityID | Y* | Int | Tradable instrument identifier |
| 22 | SecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when SecurityID (48) is specified. |
| End Component Block | | | | |
| 54 | Side | C* | Char | Side of order.<br>Valid values:<br>1 = Buy<br>2 = Sell<br>Conditionally required if specified on the original message. |
| 60 | TransactTime | Y | UTCTimestamp | Timestamp when the message was generated. |
| Component Block <OrderQtyData> | | | | |
| 38 | OrderQty | C* | Qty | Order quantity.<br>Conditionally required if specified on the original message. |
| End Component Block | | | | |

### Example Message Flow

*Successful RFQ Submission*



(131) QuoteRequestID = 12234
<QuotReqGrp>
(146) NoRelatedSym = 1
(48) SecurityID = 84646228 (Month 1)
(22) SecurityIDSource = 8
(303) QuoteRequestType = '1' Manual
(54) Side = '1' Buy
(60) TransactTime = UTC Timestamp
(38) OrderQty = 100

(693) QuoteResponseID = 34356545
(131) QuoteReqID = 12234
(694) QuoteRespType = '11' Accept
(48) SecurityID = 84646228 (Month 1)
(22) SecurityIDSource = 8
(54) Side = '1' Buy
(38) OrderQty = 100
(60) TransactTime = UTC Timestamp

### *4.11.12 Quote Request Reject (AG)*

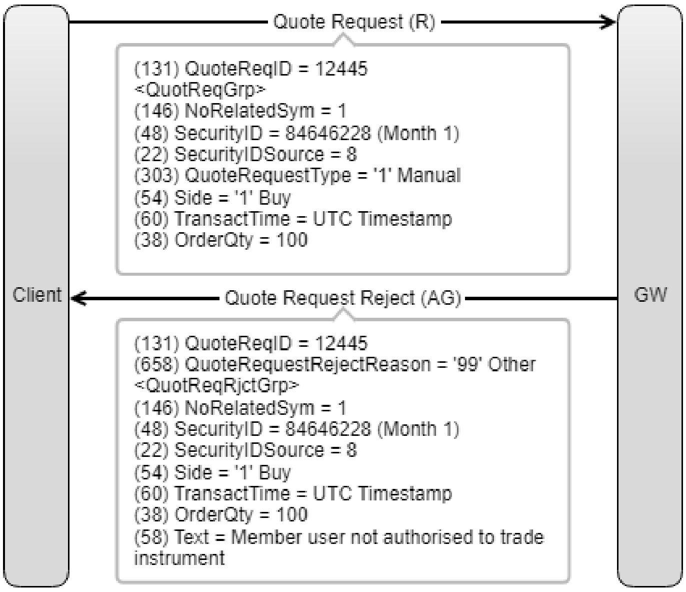
*Quote Request Reject (35=AG) notifies the originator that their Quote Request (35=R) has been rejected.*

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| *131* | *QuoteReqID* | *Y* | *String (18)* | *Client specified identifier for the quote request.* |
| *658* | *QuoteRequestRejectReason* | *Y* | *Int* | *Reason the Quote Request (R) was rejected.*<br>*Valid value:*<br>*99 = Other.*<br>*Text (58) will contain more specific information.* |
| *Component Block <QuotReqRjctGrp>* | | | | |
| *146* | *NoRelatedSym* | *Y* | *NumInGrp (1)* | *Number of related symbols (instruments) in the Quote Request.*<br>*The value can only be 1.* |
| *Component Block <Instrument>* | | | | |

| Tag | Field Name | Req | Data Type | Description |
|---|---|---|---|---|
| >48 | SecurityID | Y* | Int | Tradable instrument identifier |
| >22 | SecurityIDSource | C* | String (1) | Identifies the source of the SecurityID (48):<br>8 = Exchange Symbol<br>Conditionally required when SecurityID (48) is specified. |
| End Component Block | | | | |
| >54 | Side | C* | Char | Side of order.<br>Valid values:<br>1 = Buy<br>2 = Sell<br>Conditionally required if specified on the original message. |
| >60 | TransactTime | Y | UTCTimestamp | Timestamp when the message was generated. |
| Component Block <OrderQtyData> | | | | |
| >38 | OrderQty | C* | Qty | Order quantity.<br>Conditionally required if specified on the original message. |
| End Component Blocks | | | | |
| 58 | Text | C* | String (75) | Identifies the reason for rejection.<br>Conditionally required if QuoteRequestRejectReason (658) = '99' Other. |

### Example Message Flow

*Quote Request rejected*



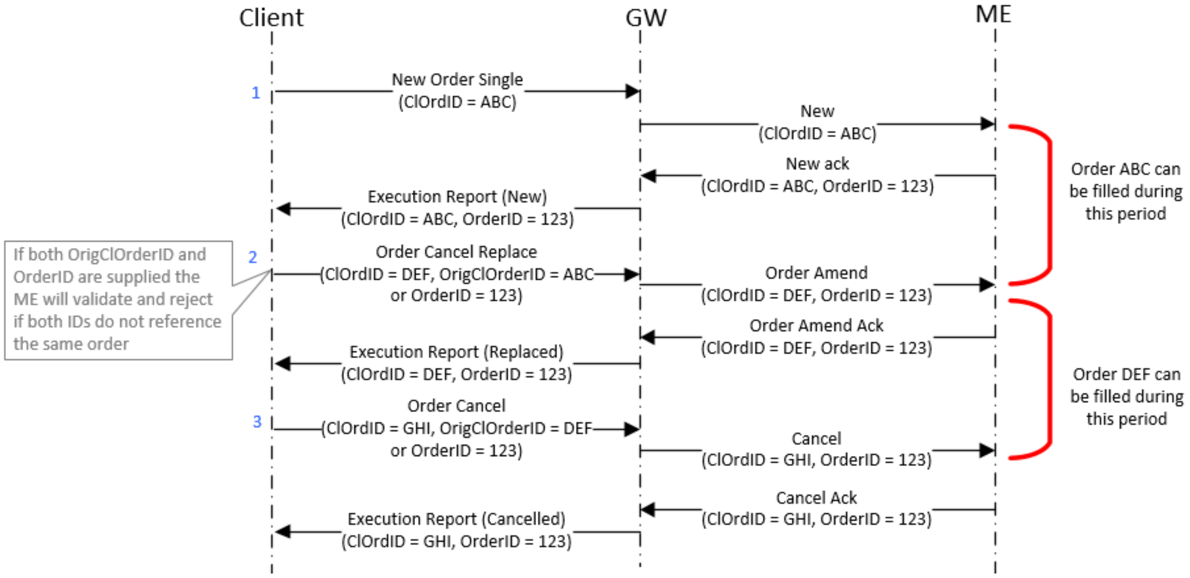
(131) QuoteReqID = 12445
<QuotReqGrp>
(146) NoRelatedSym = 1
(48) SecurityID = 84646228 (Month 1)
(22) SecurityIDSource = 8
(303) QuoteRequestType = '1' Manual
(54) Side = '1' Buy
(60) TransactTime = UTC Timestamp
(38) OrderQty = 100

(131) QuoteReqID = 12445
(658) QuoteRequestRejectReason = '99' Other
<QuotReqRjctGrp>
(146) NoRelatedSym = 1
(48) SecurityID = 84646228 (Month 1)
(22) SecurityIDSource = 8
(54) Side = '1' Buy
(60) TransactTime = UTC Timestamp
(38) OrderQty = 100
(58) Text = Member user not authorised to trade instrument

# Appendix A: Inflight Order Handling

## A.1 Standard Gateway Behaviour

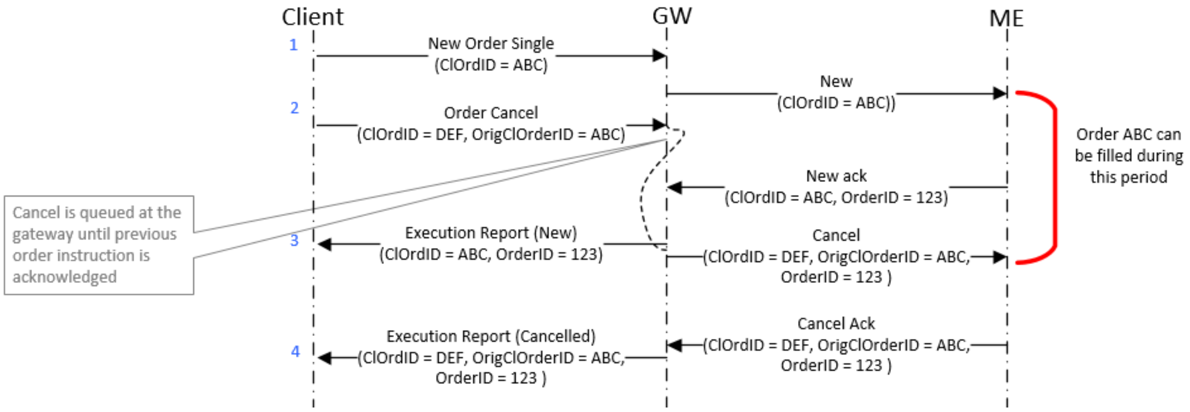
No inflight order handling, order instructions follow platform acknowledgement.


Client
GW
ME
1
New Order Single
(ClOrdID = ABC)
New
(ClOrdID = ABC)
New ack
(ClOrdID = ABC, OrderID = 123)
Execution Report (New)
(ClOrdID = ABC, OrderID = 123)
Order ABC can be filled during this period
If both OrigClOrderID and OrderID are supplied the ME will validate and reject if both IDs do not reference the same order
2
Order Cancel Replace
(ClOrdID = DEF, OrigClOrderID = ABC
or OrderID = 123)
Order Amend
(ClOrdID = DEF, OrderID = 123)
Order Amend Ack
(ClOrdID = DEF, OrderID = 123)
Execution Report (Replaced)
(ClOrdID = DEF, OrderID = 123)
Order DEF can be filled during this period
3
Order Cancel
(ClOrdID = GHI, OrigClOrderID = DEF
or OrderID = 123)
Cancel
(ClOrdID = GHI, OrderID = 123)
Cancel Ack
(ClOrdID = GHI, OrderID = 123)
Execution Report (Cancelled)
(ClOrdID = GHI, OrderID = 123)

1. A new order is submitted by the client and confirmed by the Matching Engine (ME).
2. This is followed by an amendment to the parent order which is also confirmed.
3. Followed by a cancellation of the amended order which is also confirmed.

## A.2 Inflight Cancellation

Cancellation for the parent order is queued in the gateway until an acknowledgement has been received from the ME by the gateway.


Client
GW
ME
1
New Order Single
(ClOrdID = ABC)
New
(ClOrdID = ABC))
2
Order Cancel
(ClOrdID = DEF, OrigClOrderID = ABC)
Order ABC can be filled during this period
New ack
(ClOrdID = ABC, OrderID = 123)
Cancel is queued at the gateway until previous order instruction is acknowledged
3
Execution Report (New)
(ClOrdID = ABC, OrderID = 123)
Cancel
(ClOrdID = DEF, OrigClOrderID = ABC,
OrderID = 123 )
Cancel Ack
(ClOrdID = DEF, OrigClOrderID = ABC,
OrderID = 123 )
4
Execution Report (Cancelled)
(ClOrdID = DEF, OrigClOrderID = ABC,
OrderID = 123 )

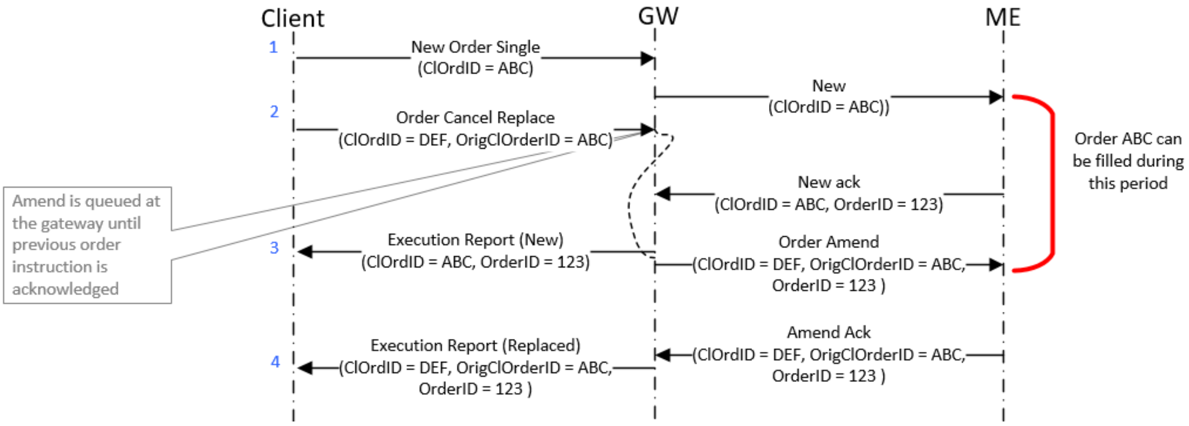
1. A new order is submitted by the client.
2. Whilst the trading platform (gateway and ME) is processing the new order request, the client sends a cancellation request. The cancellation request is queued in the gateway until the gateway has received an acknowledgement for the new order from the ME.

3. On receipt of the acknowledgement for the parent order from the ME, the gateway releases the cancellation request to the ME to be processed.

4. Once the cancellation has been processed by the ME. The ME sends an acknowledgement to the gateway which is sent onward to the client.

## A.3 Inflight Amendment

The process for an inflight amendment follows an identical process to that for an inflight cancellation.


Client
GW
ME
1
New Order Single
(ClOrdID = ABC)
New
(ClOrdID = ABC))
2
Order Cancel Replace
(ClOrdID = DEF, OrigClOrderID = ABC)
Order ABC can be filled during this period
Amend is queued at the gateway until previous order instruction is acknowledged
New ack
(ClOrdID = ABC, OrderID = 123)
3
Execution Report (New)
(ClOrdID = ABC, OrderID = 123)
Order Amend
(ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123 )
Amend Ack
(ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123 )
4
Execution Report (Replaced)
(ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123 )

1. A new order is submitted.

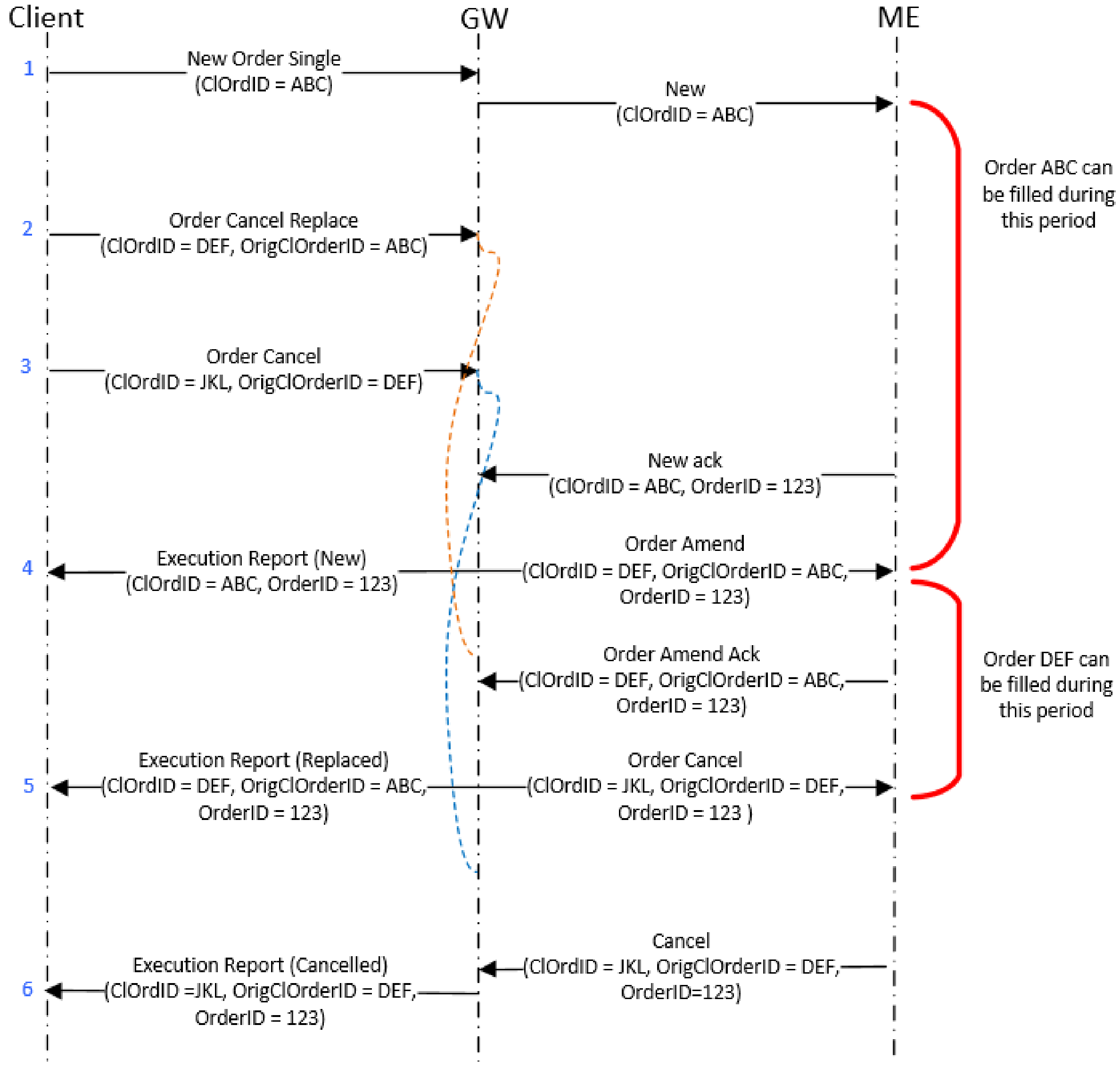
2. Whilst the new order request is being processed, the client sends an amendment request. The amendment is queued in the gateway.

3. On receipt of the acknowledgement for the parent order from the ME, the gateway releases the amendment request to the ME.

4. Once the amendment has been processed by the ME and the order has been replaced. The ME sends an acknowledgement to the gateway which is sent onward to the client.

## A.4 Inflight Amendment and Cancellation

Inflight amendment and cancellation request are queued in the gateway until an acknowledgement has been received for the previous instruction by the gateway from the ME.


Client
GW
ME
1
New Order Single (ClOrdID = ABC)
New (ClOrdID = ABC)
Order ABC can be filled during this period
2
Order Cancel Replace (ClOrdID = DEF, OrigClOrderID = ABC)
3
Order Cancel (ClOrdID = JKL, OrigClOrderID = DEF)
New ack (ClOrdID = ABC, OrderID = 123)
4
Execution Report (New) (ClOrdID = ABC, OrderID = 123)
Order Amend (ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
Order Amend Ack (ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
Order DEF can be filled during this period
5
Execution Report (Replaced) (ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
Order Cancel (ClOrdID = JKL, OrigClOrderID = DEF, OrderID = 123 )
Cancel (ClOrdID = JKL, OrigClOrderID = DEF, OrderID=123)
6
Execution Report (Cancelled) (ClOrdID =JKL, OrigClOrderID = DEF, OrderID = 123)

1. A new order is submitted.

2. Followed by an amendment to the original order (before the parent order has been acknowledged) and is queued at the gateway until parent order is acknowledged.

3. Followed by a cancellation for the amendment (before the parent and amendment have been acknowledged). This is queued in the gateway until the parent order and then the amendment has been processed.

4. The ME sends an acknowledgement for the parent order to the gateway. The gateway forwards an acknowledgement back to the client and releases the amendment request to the ME.

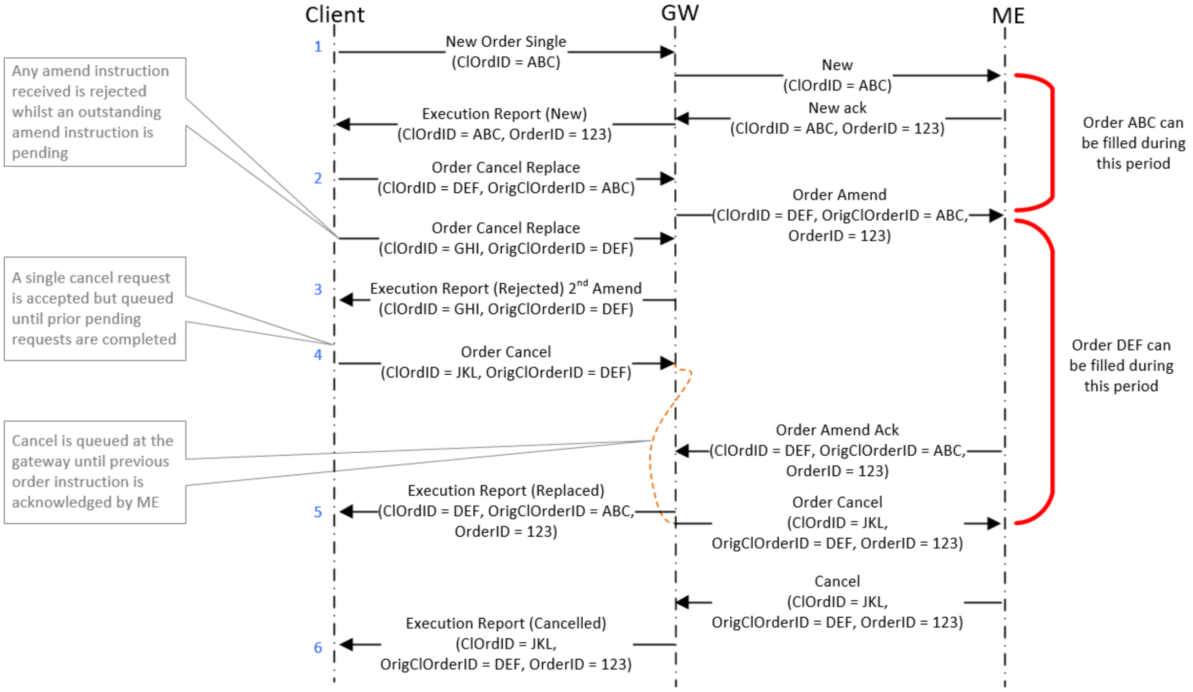
5. Once the amendment has been processed by the ME and the order has been replaced. The ME sends an acknowledgement for the amended order to the gateway. The gateway forwards the acknowledgement to the client and releases the cancellation to the ME.

6. The cancellation is processed after all the previous instructions have been processed/acknowledged.

## A.5 Multiple Inflight Amendments and a Cancellation

The Gateway permits a single amendment request, a second amendment request to replace the inflight amendment for the parent order is rejected.

An amendment request submitted for the parent order followed by a second amendment request which is rejected. A cancellation request for the amended parent order is queued in the gateway until the initial amendment request is acknowledged by the ME.


Client
GW
ME
1
New Order Single (ClOrdID = ABC)
New (ClOrdID = ABC)
New ack (ClOrdID = ABC, OrderID = 123)
Execution Report (New) (ClOrdID = ABC, OrderID = 123)
2
Order Cancel Replace (ClOrdID = DEF, OrigClOrderID = ABC)
Order Amend (ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
Order Cancel Replace (ClOrdID = GHI, OrigClOrderID = DEF)
3
Execution Report (Rejected) 2nd Amend (ClOrdID = GHI, OrigClOrderID = DEF)
4
Order Cancel (ClOrdID = JKL, OrigClOrderID = DEF)
Order Amend Ack (ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
5
Execution Report (Replaced) (ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
Order Cancel (ClOrdID = JKL, OrigClOrderID = DEF, OrderID = 123)
Cancel (ClOrdID = JKL, OrigClOrderID = DEF, OrderID = 123)
6
Execution Report (Cancelled) (ClOrdID = JKL, OrigClOrderID = DEF, OrderID = 123)
Any amend instruction received is rejected whilst an outstanding amend instruction is pending
A single cancel request is accepted but queued until prior pending requests are completed
Cancel is queued at the gateway until previous order instruction is acknowledged by ME
Order ABC can be filled during this period
Order DEF can be filled during this period

1. A new order submitted and acknowledged, followed by an amendment to the parent order.

2. Before the amendment has been processed by the ME, a second amendment instruction is submitted.

3. The second amendment is rejected as the initial amendment is in progress.

4. A cancellation request is submitted for the amendment in progress is queued at the gateway until the amendment has been processed.

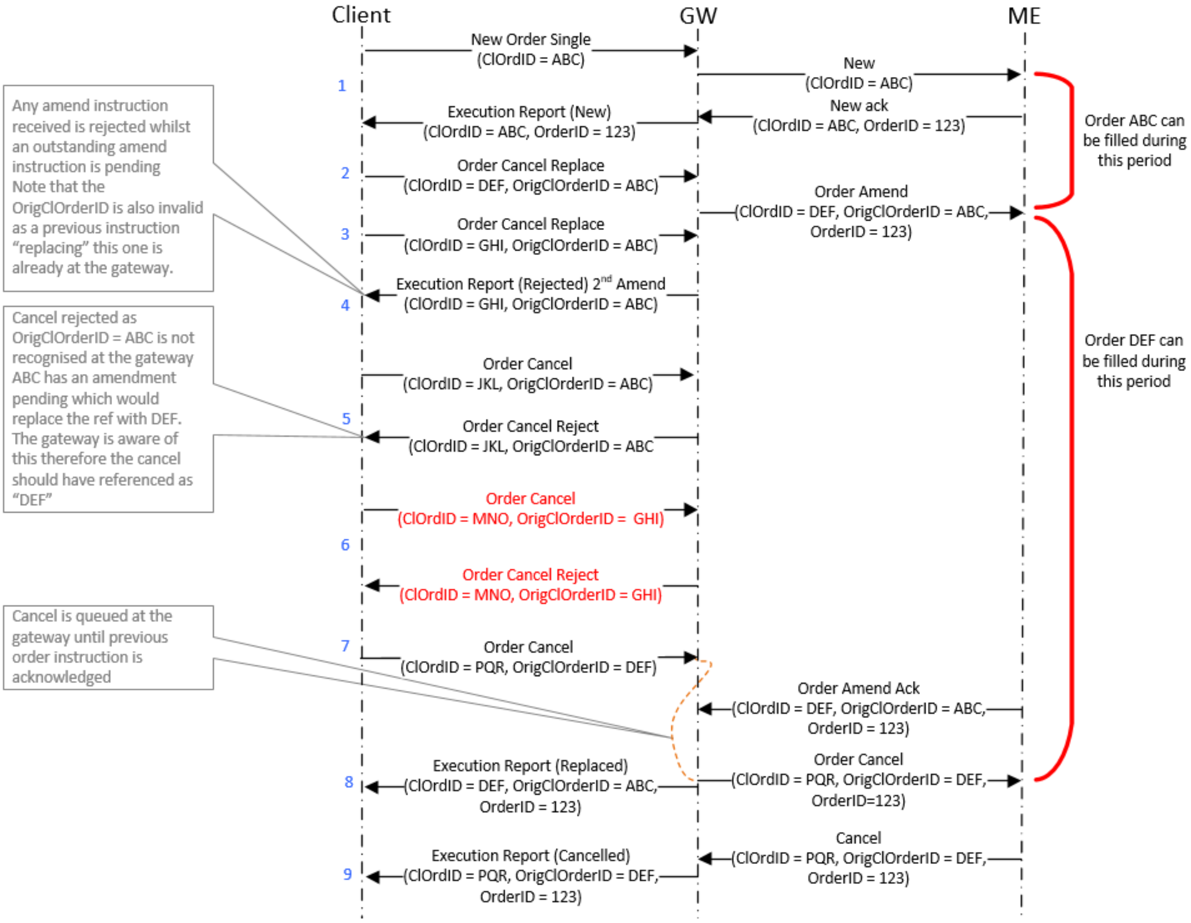
5. The ME sends an acknowledgement for the amended order to the gateway. The gateway forwards the acknowledgement to the client and releases the cancellation to the ME.

6. The cancellation is processed after all the previous instructions have been processed/acknowledged.

## A.6 Multiple Inflight Amendments and Cancellations

### Example 1: Two inflight amendments followed by a cancellation referencing the original parent, followed by a cancellation referencing a rejected amendment request and one more referencing the amended order

An amendment request submitted for the parent order followed by a second amendment request which is rejected (only one inflight amendment request is permitted). Multiple cancellation requests are submitted which are rejected due to incorrectly referencing previous order instructions. A cancellation for the original parent order is rejected followed by a cancellation request for the second amendment request which was rejected. A cancellation request for the amended parent order is queued until the amendment is actually processed.


Client
GW
ME
New Order Single (ClOrdID = ABC)
1
New (ClOrdID = ABC)
Any amend instruction received is rejected whilst an outstanding amend instruction is pending Note that the OrigClOrderID is also invalid as a previous instruction “replacing” this one is already at the gateway.
Execution Report (New) (ClOrdID = ABC, OrderID = 123)
New ack (ClOrdID = ABC, OrderID = 123)
Order ABC can be filled during this period
2
Order Cancel Replace (ClOrdID = DEF, OrigClOrderID = ABC)
Order Amend (ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
3
Order Cancel Replace (ClOrdID = GHI, OrigClOrderID = ABC)
4
Execution Report (Rejected) 2nd Amend (ClOrdID = GHI, OrigClOrderID = ABC)
Cancel rejected as OrigClOrderID = ABC is not recognised at the gateway ABC has an amendment pending which would replace the ref with DEF. The gateway is aware of this therefore the cancel should have referenced as “DEF”
Order DEF can be filled during this period
Order Cancel (ClOrdID = JKL, OrigClOrderID = ABC)
5
Order Cancel Reject (ClOrdID = JKL, OrigClOrderID = ABC
Order Cancel (ClOrdID = MNO, OrigClOrderID = GHI)
6
Order Cancel Reject (ClOrdID = MNO, OrigClOrderID = GHI)
Cancel is queued at the gateway until previous order instruction is acknowledged
7
Order Cancel (ClOrdID = PQR, OrigClOrderID = DEF)
Order Amend Ack (ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
8
Execution Report (Replaced) (ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
Order Cancel (ClOrdID = PQR, OrigClOrderID = DEF, OrderID=123)
9
Execution Report (Cancelled) (ClOrdID = PQR, OrigClOrderID = DEF, OrderID = 123)
Cancel (ClOrdID = PQR, OrigClOrderID = DEF, OrderID = 123)

1. A new order is submitted and fully acknowledged.
2. The first amendment to the parent order is submitted.
3. Before the first amendment has been processed, a second amendment instruction is submitted.
4. The second amendment is rejected by the gateway as the existing amendment is in progress.
5. A cancellation instruction submitted for the parent order is rejected as it refers to cancelling the parent ClOrdID rather than ClOrdID of the amendment.

6. A second cancellation is submitted referencing the second amendment instruction that was rejected. This cancellation request is also rejected as it refers to the ClOrdID of the amendment that has already been rejected.

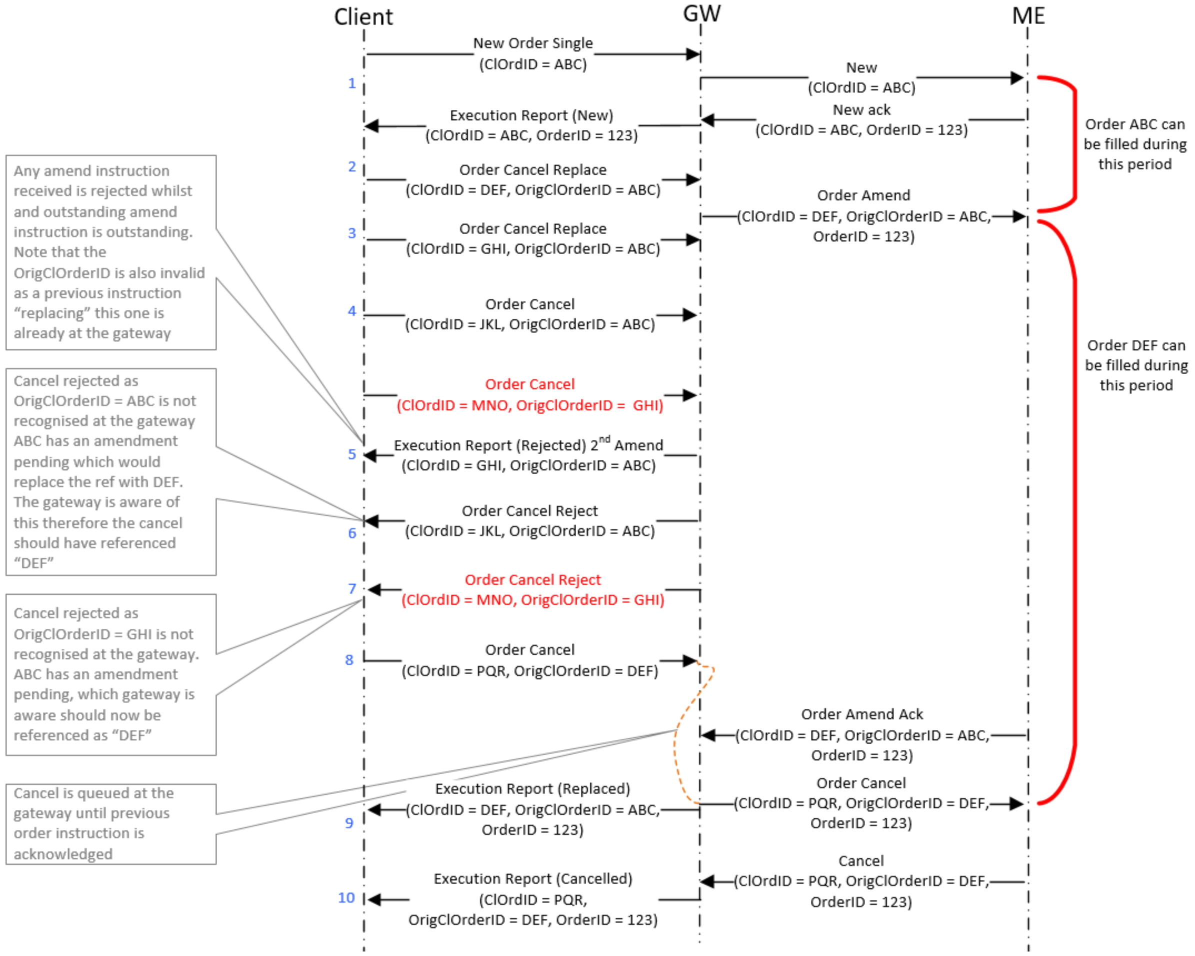
7. A cancellation is submitted for the amendment in progress is queued until the amendment has been processed.

8. The ME sends an acknowledgement for the amended order to the gateway. The gateway forwards the acknowledgement to the client and releases the cancellation to the ME

9. The cancellation is then processed after all the previous instructions are processed/acknowledged.

### Example 2: Two inflight amendments followed by a cancellation referencing the original parent and a cancellation referencing the amended order

An amendment request received for a parent order followed by a second amendment request which is rejected (only one inflight amendment request is permitted). A cancellation for the original parent order is rejected due to incorrectly referencing previous order instructions. A second cancellation request is submitted for the amended order which is queued and then processed.


Client
GW
ME
New Order Single
(ClOrdID = ABC)
1
New
(ClOrdID = ABC)
Execution Report (New)
(ClOrdID = ABC, OrderID = 123)
New ack
(ClOrdID = ABC, OrderID = 123)
Order ABC can be filled during this period
2
Order Cancel Replace
(ClOrdID = DEF, OrigClOrderID = ABC)
Order Amend
(ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
3
Order Cancel Replace
(ClOrdID = GHI, OrigClOrderID = ABC)
4
Order Cancel
(ClOrdID = JKL, OrigClOrderID = ABC)
Order DEF can be filled during this period
Order Cancel
(ClOrdID = MNO, OrigClOrderID = GHI)
5
Execution Report (Rejected) 2nd Amend
(ClOrdID = GHI, OrigClOrderID = ABC)
Order Cancel Reject
6
(ClOrdID = JKL, OrigClOrderID = ABC)
7
Order Cancel Reject
(ClOrdID = MNO, OrigClOrderID = GHI)
8
Order Cancel
(ClOrdID = PQR, OrigClOrderID = DEF)
Order Amend Ack
(ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
Execution Report (Replaced)
(ClOrdID = DEF, OrigClOrderID = ABC, OrderID = 123)
9
Order Cancel
(ClOrdID = PQR, OrigClOrderID = DEF, OrderID = 123)
Cancel
(ClOrdID = PQR, OrigClOrderID = DEF, OrderID = 123)
Execution Report (Cancelled)
10
(ClOrdID = PQR, OrigClOrderID = DEF, OrderID = 123)

Any amend instruction received is rejected whilst and outstanding amend instruction is outstanding. Note that the OrigClOrderID is also invalid as a previous instruction “replacing” this one is already at the gateway

Cancel rejected as OrigClOrderID = ABC is not recognised at the gateway ABC has an amendment pending which would replace the ref with DEF. The gateway is aware of this therefore the cancel should have referenced “DEF”

Cancel rejected as OrigClOrderID = GHI is not recognised at the gateway. ABC has an amendment pending, which gateway is aware should now be referenced as “DEF”

Cancel is queued at the gateway until previous order instruction is acknowledged

1. A new order is submitted and fully acknowledged.

2. The first amendment to the parent order is submitted.

3. Before the first amendment has been processed, a second amendment instruction is submitted.

4. Before any of the above have been processed, a cancellation is submitted referencing the parent order and the second amendment.

5. The second amendment instruction is rejected by the gateway as an existing amendment is in progress.

6. The first cancellation submitted for the parent order is rejected as it refers to cancelling the parent ClOrdID rather than ClOrdID of the amendment still pending.

7. The second cancellation instruction is rejected as a second inflight cancellation is not permitted and also the request refers to cancelling an unknown ClOrdID.

8. A third cancellation instruction is submitted which references the amendment in progress. This request is queued until the amendment has been processed.

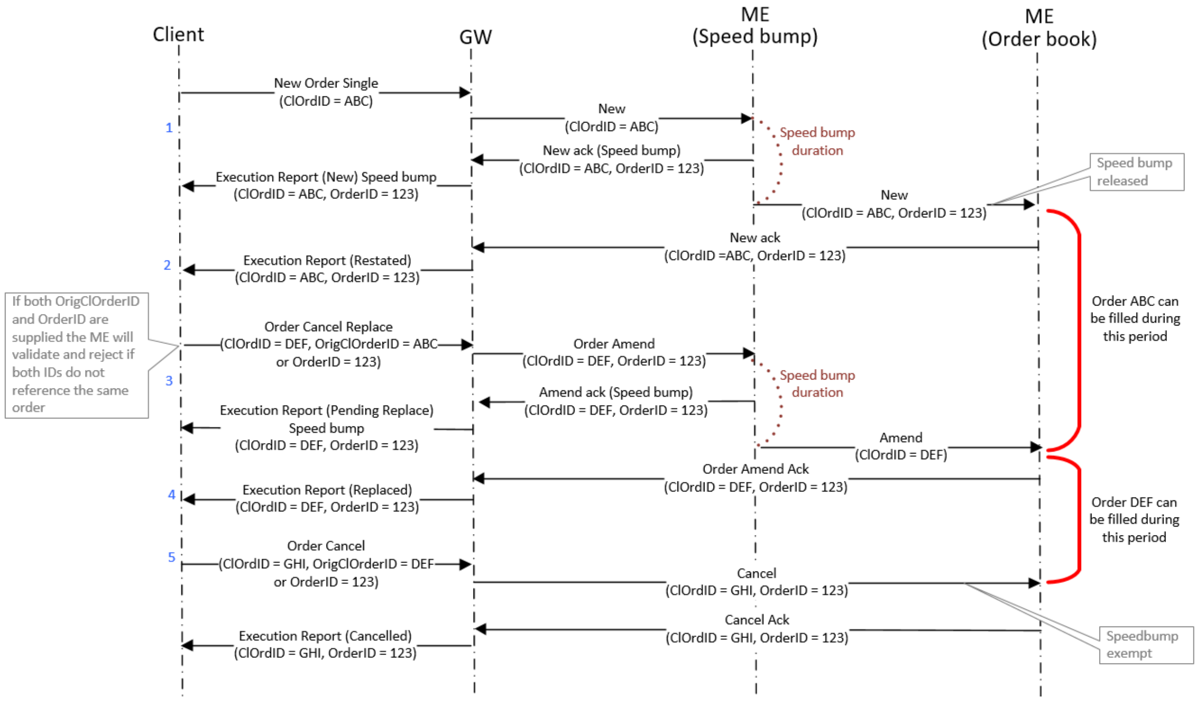
9. The ME sends an acknowledgement for the amended order to the gateway. The gateway forwards the acknowledgement to the client and releases the cancellation to the ME

10. The cancellation is then processed after all previous instructions are processed/acknowledged.

# Appendix B: Speed Bump Inflight Order Handling

## B.1 Speed Bump Message Flow

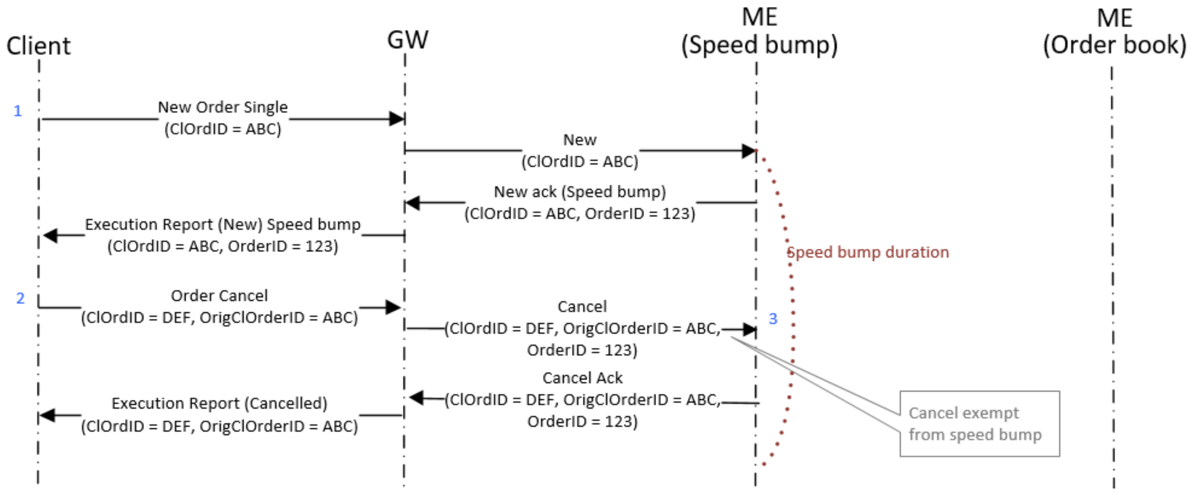
*The message flow follows that described in 3.13 Speed Bumps - Executable order amendment for a resting order will be speed bumped.*


Client
GW
ME
(Speed bump)
ME
(Order book)
New Order Single
(ClOrdID = ABC)
New
(ClOrdID = ABC)
1
Speed bump
duration
New ack (Speed bump)
(ClOrdID = ABC, OrderID = 123)
Speed bump
released
Execution Report (New) Speed bump
(ClOrdID = ABC, OrderID = 123)
New
(ClOrdID = ABC, OrderID = 123)
New ack
(ClOrdID =ABC, OrderID = 123)
2
Execution Report (Restated)
(ClOrdID = ABC, OrderID = 123)
Order ABC can
be filled during
this period
If both OrigClOrderID
and OrderID are
supplied the ME will
validate and reject if
both IDs do not
reference the same
order
Order Cancel Replace
(ClOrdID = DEF, OrigClOrderID = ABC
or OrderID = 123)
Order Amend
(ClOrdID = DEF, OrderID = 123)
3
Speed bump
duration
Amend ack (Speed bump)
(ClOrdID = DEF, OrderID = 123)
Execution Report (Pending Replace)
Speed bump
(ClOrdID = DEF, OrderID = 123)
Amend
(ClOrdID = DEF)
Order Amend Ack
(ClOrdID = DEF, OrderID = 123)
4
Execution Report (Replaced)
(ClOrdID = DEF, OrderID = 123)
Order DEF can
be filled during
this period
Order Cancel
5
(ClOrdID = GHI, OrigClOrderID = DEF
or OrderID = 123)
Cancel
(ClOrdID = GHI, OrderID = 123)
Cancel Ack
(ClOrdID = GHI, OrderID = 123)
Execution Report (Cancelled)
(ClOrdID = GHI, OrderID = 123)
Speedbump
exempt

1. *A new order is submitted and acknowledged as being speed bumped.*
2. *The order is added to the order book and acknowledged once it has cleared the speedbump.*
3. *An amendment is submitted for the parent order and acknowledged as being speed bumped.*
4. *The amended order replaces the parent and is acknowledged once it has cleared the speed bump.*
5. *A cancellation is submitted which is exempt from the speed bump cancels the amended order in the order book.*

## B.2 Inflight Cancellation in Speed Bump

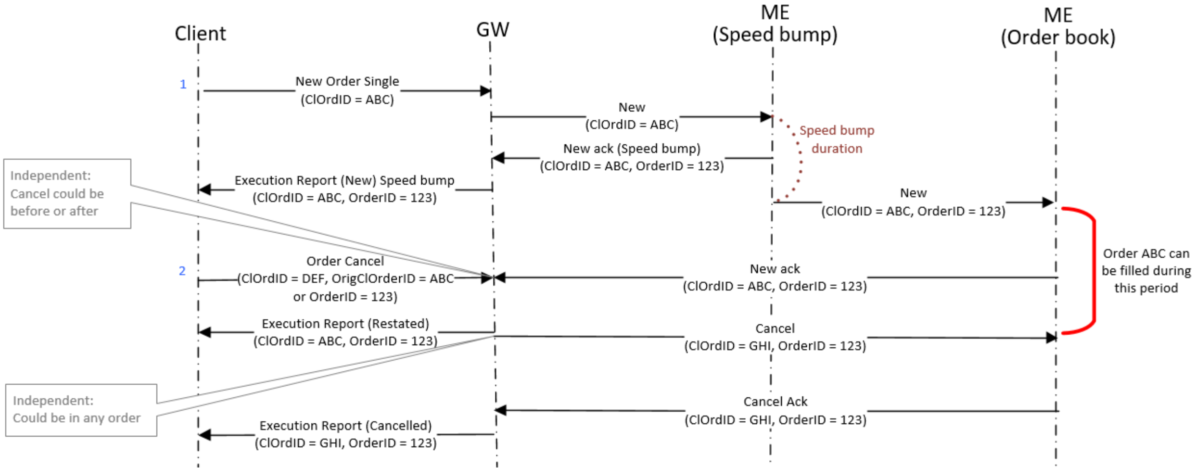
*The message flow follows that described in* [3.13 Speed Bumps - Order cancellation for a speed bumped order]().


Client
GW
ME
(Speed bump)
ME
(Order book)
1
New Order Single
(ClOrdID = ABC)
New
(ClOrdID = ABC)
New ack (Speed bump)
(ClOrdID = ABC, OrderID = 123)
Execution Report (New) Speed bump
(ClOrdID = ABC, OrderID = 123)
Speed bump duration
2
Order Cancel
(ClOrdID = DEF, OrigClOrderID = ABC)
Cancel
(ClOrdID = DEF, OrigClOrderID = ABC,
OrderID = 123)
3
Cancel Ack
(ClOrdID = DEF, OrigClOrderID = ABC,
OrderID = 123)
Execution Report (Cancelled)
(ClOrdID = DEF, OrigClOrderID = ABC)
Cancel exempt
from speed bump

1. *A new order submitted and acknowledged as being speed bumped.*
2. *A cancellation is submitted for the order which is in the speed bump.*
3. *The cancellation is exempt from the speed bump cancels the order in the speed bump.*

## B.3 Inflight Cancellation past Speed Bump

*The message flow follows that described in* [3.13 Speed Bumps - Order submission is speed bumped]() *but also includes a cancellation.*

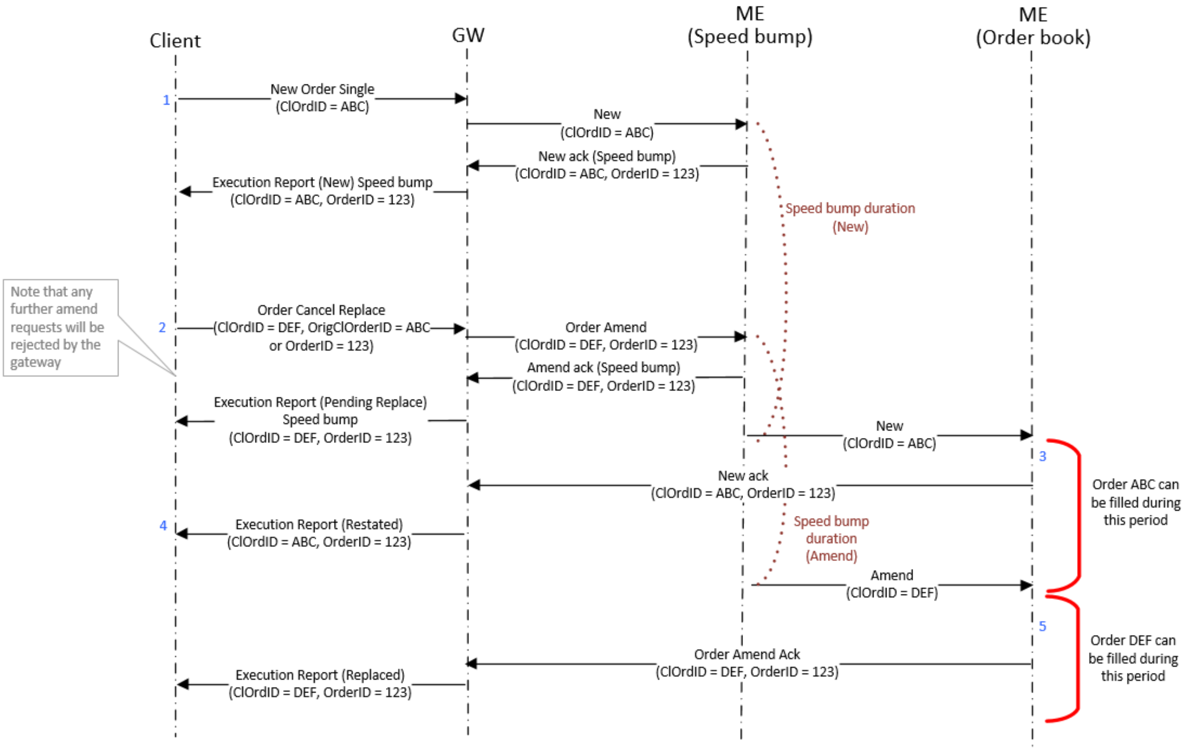

Client
GW
ME
(Speed bump)
ME
(Order book)
1
New Order Single
(ClOrdID = ABC)
New
(ClOrdID = ABC)
Speed bump
duration
New ack (Speed bump)
(ClOrdID = ABC, OrderID = 123)
Independent:
Cancel could be
before or after
Execution Report (New) Speed bump
(ClOrdID = ABC, OrderID = 123)
New
(ClOrdID = ABC, OrderID = 123)
Order ABC can
be filled during
this period
2
Order Cancel
(ClOrdID = DEF, OrigClOrderID = ABC
or OrderID = 123)
New ack
(ClOrdID = ABC, OrderID = 123)
Execution Report (Restated)
(ClOrdID = ABC, OrderID = 123)
Cancel
(ClOrdID = GHI, OrderID = 123)
Independent:
Could be in any order
Cancel Ack
(ClOrdID = GHI, OrderID = 123)
Execution Report (Cancelled)
(ClOrdID = GHI, OrderID = 123)

1. *A new order is submitted and acknowledged as being speed bumped. The order is added to the order book and acknowledged once it has cleared the speedbump.*
2. *A cancellation is submitted which is exempt from the speed bump cancels the order in the order book.*

*Note: The cancellation can be sent regardless of whether acknowledgements have been received for the previous instruction.*

## B.4 Inflight Amendment Speed Bumped

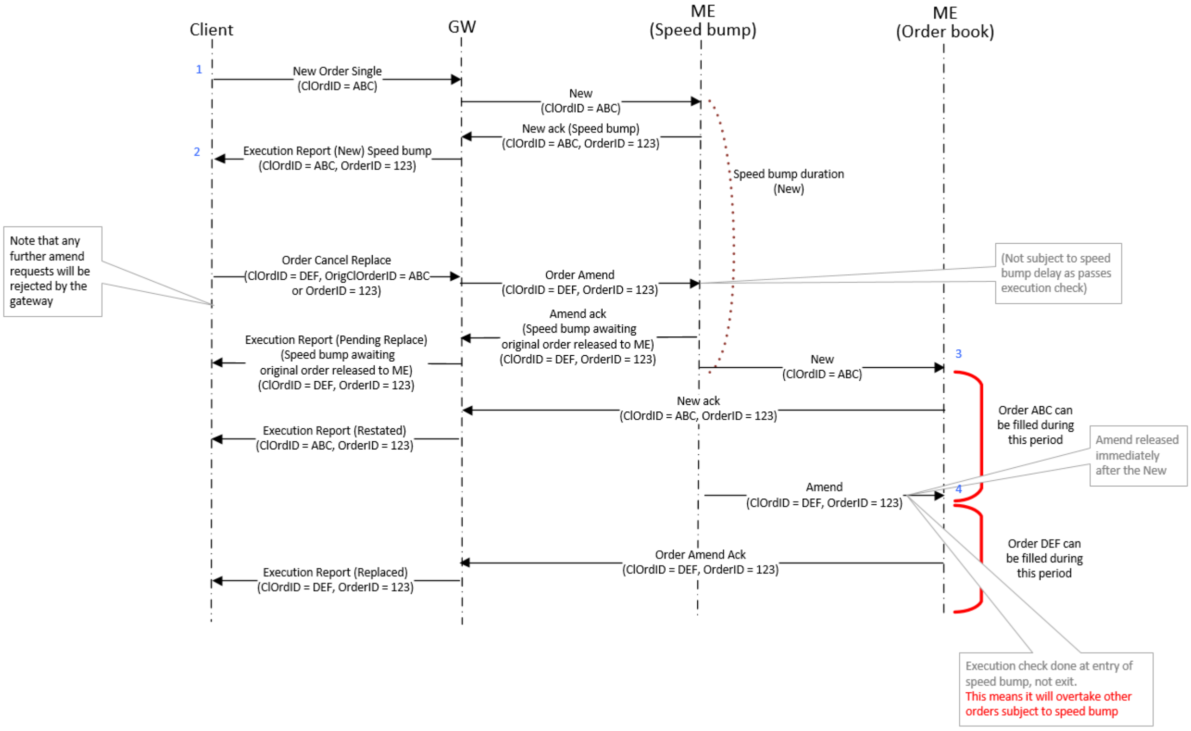
*The message flow follows that described in* 3.13 Speed Bumps - Executable order amendment for a speed bumped order will be speed bumped.


Client
GW
ME (Speed bump)
ME (Order book)
1
New Order Single (ClOrdID = ABC)
New (ClOrdID = ABC)
New ack (Speed bump) (ClOrdID = ABC, OrderID = 123)
Execution Report (New) Speed bump (ClOrdID = ABC, OrderID = 123)
Speed bump duration (New)
Note that any further amend requests will be rejected by the gateway
2
Order Cancel Replace (ClOrdID = DEF, OrigClOrderID = ABC or OrderID = 123)
Order Amend (ClOrdID = DEF, OrderID = 123)
Amend ack (Speed bump) (ClOrdID = DEF, OrderID = 123)
Execution Report (Pending Replace) Speed bump (ClOrdID = DEF, OrderID = 123)
New (ClOrdID = ABC)
3
Order ABC can be filled during this period
New ack (ClOrdID = ABC, OrderID = 123)
4
Execution Report (Restated) (ClOrdID = ABC, OrderID = 123)
Speed bump duration (Amend)
Amend (ClOrdID = DEF)
5
Order DEF can be filled during this period
Order Amend Ack (ClOrdID = DEF, OrderID = 123)
Execution Report (Replaced) (ClOrdID = DEF, OrderID = 123)

1. *A new order is submitted and acknowledged as being speed bumped.*
2. *An amendment is submitted for the order in the speedbump. The amendment is acknowledged as being speed bumped according to the execution check.*
3. *Amendment will not be processed until the parent order has cleared the speed bump and been added to the order book.*
4. *The parent order is added to the order book and acknowledged once it has cleared the speedbump.*
5. *The amendment is released from the speed bump and replaces the parent order. The amended order is acknowledged once it has been added to the order book.*

## B.5 Inflight Amendment not Speed Bumped

*The message flow follows that described in* 3.13 Speed Bumps - Non-executable order amendment for a speed bumped order will not be speed bumped*.*


Client
GW
ME
(Speed bump)
ME
(Order book)
1
New Order Single
(ClOrdID = ABC)
New
(ClOrdID = ABC)
New ack (Speed bump)
(ClOrdID = ABC, OrderID = 123)
2
Execution Report (New) Speed bump
(ClOrdID = ABC, OrderID = 123)
Speed bump duration
(New)
Note that any further amend requests will be rejected by the gateway
Order Cancel Replace
(ClOrdID = DEF, OrigClOrderID = ABC
or OrderID = 123)
Order Amend
(ClOrdID = DEF, OrderID = 123)
(Not subject to speed bump delay as passes execution check)
Amend ack
(Speed bump awaiting
original order released to ME)
(ClOrdID = DEF, OrderID = 123)
Execution Report (Pending Replace)
(Speed bump awaiting
original order released to ME)
(ClOrdID = DEF, OrderID = 123)
3
New
(ClOrdID = ABC)
New ack
(ClOrdID = ABC, OrderID = 123)
Order ABC can be filled during this period
Execution Report (Restated)
(ClOrdID = ABC, OrderID = 123)
Amend released immediately after the New
4
Amend
(ClOrdID = DEF, OrderID = 123)
Order DEF can be filled during this period
Order Amend Ack
(ClOrdID = DEF, OrderID = 123)
Execution Report (Replaced)
(ClOrdID = DEF, OrderID = 123)
Execution check done at entry of speed bump, not exit.
This means it will overtake other orders subject to speed bump

1. *A new order is submitted and acknowledged as being speed bumped.*
2. *An amendment is submitted for the order in the speed bump. The amendment is not speed bumped but cannot be processed until the parent order has cleared the speed bump.*
3. *The parent order is added to the order book and acknowledged once it has cleared the speedbump*
4. *The amendment is then released from the queue and replaces the parent order. The amended order is acknowledged once it has been added to the order book.*