Nasdaq Financial Framework™

Nasdaq Direct Drop (DD) Protocol Specification for ASX

Software Version: 6.15.0.x
Document Version: SKR_a76
Publish Date: Oct 09, 2024

Nasdaq

Table of Contents

1 Summary of Changes

Changes between this version and version a73 (6.15.0.x).

No	Changes	Comment
1	New Change reason 101 - MassCancelBy User added	

2 About the Manual

The purpose of this document is to explain the Nasdaq Direct Drop protocol.

2.1 References

For more information, refer to the following document:

Nasdaq OMN Interface Message Reference

3 Overview

The Direct Drop is a protocol that provides a consolidated view of all orders and trades processed by a matching engine (ME). The protocol is based on a schema that is compiled into message parsing code. The transport protocol is shared with OUCH and Nasdaq ITCH; SoupBinTCP.

The Direct Drop mimics the output of the ME.

The Direct Drop contains:

- Order updates (including combination orders and overnight orders)
- Trades and trade cancels (including information from clearing)
- Session changes (including initial session state)
- Reference data for instruments, participants and users.

Note:
A drop session covers a single back-end partition. In a multi-partition set-up, one Direct drop feed is needed per partition.

3.1 Message Flow Basics

Each morning, the Direct Drop starts by providing static Reference Data. The Reference Data load is guaranteed to finish before any real-time messages are published, but intraday updates to reference data may occur at any time.

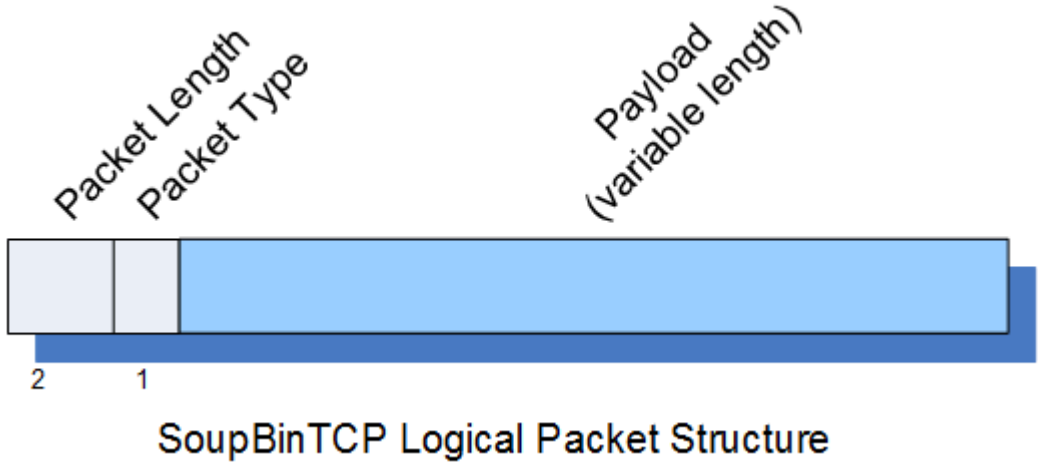
The real-time data includes a mix of orders, trades state change and intra-day reference data messages. Orders and trades are grouped into matching rounds, also called commits. Special messages mark the start and end of a matching round.

3.2 Transport Protocol and Message Recovery

The Direct Drop messages are transported over the SoupBinTCP protocol. SoupBinTCP is a lightweight point-to-point protocol, built on top of TCP/IP sockets that allow delivery of a set of sequenced messages from a server to a client in real time. SoupBinTCP guarantees that the client receives each message generated by the server in sequence, even across underlying TCP/IP socket connection failures.

The sequence numbers are implicit, meaning that the client maintains a counter that is increased every time a message is received. At reconnect after a connection loss, the client submits the last seen sequence number in its Logon message, and the server resends every message starting from that sequence number.

A SoupBinTCP Packet



SoupBinTCP Logical Packet Structure

Note that the payload contains the messages as defined in this spec.

Refer to the SoupBinTCP specification for further details.

Note:
The SoupBinTCP header contains a packet length field. The client is required to handle larger messages than defined in this spec. This ensures that an “old” client will still be able to parse messages that have new information added to the end of the message.

The client must also drop any messages with an unknown message type.

3.3 Field Lengths

The following applies to field lengths in the messages:

Field type	Description
Boolean	A single byte indicating a boolean value (0 - false; 1 - true)
Byte	One byte indicating a byte (8-bit signed integer)
Short	Two bytes indicating a short (16-bit signed integer)
Integer	Four bytes indicating an integer (32-bit signed integer)
Long	Eight bytes indicating a long (64-bit signed integer)
String	The first two bytes (a short) specifies the number of bytes in the string. The contents of strings in Direct Drop are 1 byte ANSI characters.
Char	Indicates a one-byte ANSI character.
Ref (reference)	Indicates a record. These fields are preceded by a boolean indicating if the record contains data or not.

Array type	Description
Single	The first two bytes (a short) specifies the number of elements in the array.

Array type	Description
array	
Double array	The first two bytes (a short) specifies the number of single arrays in the array. The length of the single arrays are specified according to the above.

Note:
References to records (ref) may as both arrays and single fields in Direct Drop messages. Arrays of references are not preceded by a boolean indicating if the record contains data or not. This prefix is only used in record references that are not arrays.

Note:
All fields in Direct Drop are written with little-endian byte order.

3.4 Message Group and Message ID

All messages are prefixed by a message group (short) and a message ID (short) indicating the message type.

4 Messages

4.1 Transactional Data

4.1.1 Version Message

The Version Message is the first message sent by a Direct Drop server shall be a version message. This message contains the required information for a client to see that it has connected to a compatible server.

- Message Group = 10
- Message ID = 23

Version Message

Field	Type	Description
versionInfo		*versionInfo Record* Field / Type / Description **geniumVersion** / String / Nasdaq Financial Framework version number. **geniumBuild** / String / Nasdaq Financial Framework build number.

4.1.2 Start Of Transaction Message

The Start Of Transaction message is sent when a new transaction has started.

- Message Group = 10
- Message ID = 9

Start Of Transaction Message

Field	Type	Description
orderId	Long	Identification of the new transaction.

4.1.3 Commit Message

The Commit message is sent when all messages for the ongoing transaction has been sent.

- Message Group = 10
- Message ID = 10

Commit Message

Field	Type	Description
startTimeStamp	Long	The timestamp indicating transaction. UTC time.
duration	Long	The time (ns) spent processing the message.

4.2 Reference Data

4.2.1 End Of Reference Data Message

The End Of Reference Data message is sent at start-up after all reference data has been sent. It signals when the client should start processing reference data. It contains no fields.

- Message Group = 10
- Message ID = 11

4.2.2 Participant Message

The Participant message is sent for all participants defined in Nasdaq Financial Framework.

- Message Group = 10
- Message ID = 2

Participant Message

Field	Type	Description
timestamp	Long	Time when the message was sent. UTC time.
id	Integer	Numeric identification of the participant. The **id** is unique and unchanged as long as the participant exists in the reference data. If the participant is deleted and then re-inserted the **id** may be different.
participantName	String	Name of the participant. The **participantName** is unique and unchanged as long as the participant exists in the reference data. The **participantName** remains the same if the participant is deleted and re-inserted.
exchangeId	Short	Numeric identification of the exchange the participant is connected to.
exchangeName	String	Name of the exchange.
active	Boolean	Specifies if the participant is active or suspended.
participantType	Byte	The type, such as General Clearing Member or Non Clearing Member.
action	Byte	New/Update/Delete
clearingHouseId	String	Clearing House ID for the participant.
organisationNumber	String	Participant's organization number.
clearingParticipants[ ]		List of valid clearing customers, which consist of the following fields:. **Note:** The default clearing customer is always the last clearing customer in the list. *ClearingParticipant Record* Field / Type / Description **id** / Char[] / The clearing participant id **name** / String / The name of the clearing participant

Field	Type	Description
participantLongName	String	The full name of the participant.

Detailed Field Description

Value	Description
participantType	
1	Direct Clearing Participant
2	General Clearing Participant
3	Non Clearing Participant
4	Clearing House
5	Custody Service
6	CSD
7	NPCH
action	
1	New
2	Update
3	Delete

4.2.3 User Message

The User message is sent for all users defined in Nasdaq Financial Framework.

- Message Group = 10
- Message ID = 3

User Message

Field	Type	Description
timestamp	Long	Time when the message was sent. UTC time.
id	Integer	Numeric identification of the user. The **id** is unique and unchanged as long as the user exists in the reference data. If the user is deleted and then re-inserted the **id** may be different.
exchangeName	String	Name of the exchange (Country in the Trading Code).
participantName	String	Name of the participant (for example Customer in the Trading Code).
userName	String	Name of the user (User Id in the Trading Code). The **userName** is unique and unchanged as long as the user exists in the reference data. The **userName** remains the same if the user is deleted and re-inserted.
userFullName	String	Full name of the user.
participantId	Integer	Numeric identification of the participant the user is connected to.
preTradeRiskCheckEnabled	Boolean	Specifies if risk checks are enabled for the user.
active	Boolean	Specifies if the user is active or suspended.
action	Byte	New/Update/Delete
locked	Boolean	Specifies if the user is locked, for example after submitting incorrect password.

Field	Type	Description
compID	String	Specifies the FIX Sender CompID.
subID	String	FIX Sender SubID.
locationID	String	FIX Sender location ID.
priceImprovementAllowed	Boolean	Specifies if the user allows price improvement.

Detailed Field Description

Value	Description
action	
1	New
2	Update
3	Delete

4.2.4 Order Book Message

The Order Book message is sent for all order books defined in Nasdaq Financial Framework.

- Message Group = 10
- Message ID = 1

Order Book Message

Field	Type	Description
timestamp	Long	The time when the message was sent. UTC time.
id	Integer	The numeric identification of the order book.
name	String	The name of the security.
exchangeId	Short	The numeric identification of the exchange this order book belongs to.
marketId	Short	The numeric identification of the market this order book belongs to.
instrumentGroupId	Short	The numeric identification of the instrument group this order book belongs to.
modifier	Short	Expiration date modifier. This value is set to zero when the instrument is new. The value is incremented by one each time the instrument is involved in an issue, split, etc. Note that the modifier value can be different for bid and ask options in the same Series.
underlyingId	Integer	The numeric identification of the underlying this order book belongs to. This is identical to the commodity code in the series structure and will always be set, that is, even for cash instruments that do not have an underlying instrument.
strikePrice	Integer	The strike price for this order book. 0 if not an option.
expirationDate	Integer	The expiration date for this order book. 0 if not a derivative. A bit pattern is used. The seven most significant bits are used for year, the next four for month and the five least significant bits for day. All these bits make up an unsigned word. The year-field starts counting from 1990. Thus, 1990=1, 1991=2 ...

Field	Type	Description
		2001=12. **Example:** January 1, 1990: Binary: 0000001 0001 00001 year month day 7 bits 4 bits 5 bits Decimal: 545
firstTradingDate	Long	The first trading date for the instrument. UTC time.
lastTradingDate	Long	The last trading date for the instrument. UTC time.
groupType	Byte	For instance Cash, Forward, Option, or Future.
optionType	Byte	Call or Put. 0 if not an option.
optionStyle	Byte	American, European, and so on. 0 if not an option.
sector	String	The industry sector of the security on string format, that is, not GICS format.
currency	String	The ISO currency the order book is traded in, such as USD.
currencyUnit	Byte	Specifies if the order book is traded in the first unit, such as USD, or in the second unit, such as CENT.
currencyRelation	Integer	Specifies the relation between the second and the first unit. For instance, if the currency is USD and the **currencyUnit** is second, this field will be set to 100, indicating that there are 100 cents on a US dollar.
contractSize	Integer	The contract size. Set to 1 for a cash product.
priceQuotationFactor	Integer	The price quotation factor used to calculate the trade price from the order.
priceUnit	Byte	For instance Price or Yield.
tickSizes[ ]		Array of **tickSizeItem** records, which consist of the following fields: *tickSizeItem Record* Field / Type / Description **lowerLimit** / Long / The lower limit for which the tick size applies **upperLimit** / Long / The upper limit for which the tick size applies **tickSize** / Long / The tick size
decimalsInPrice	Integer	The number of implicit decimals in the price used for trading in this order book.
decimalsInStrikePrice	Integer	The number of implicit decimals in the strike price of the order book.
decimalsInQuantity	Integer	The number of implicit decimals in the quantity used for trading in this order book.
underlyingName	String	The name of the underlying instrument, that is, the symbol. For an instrument that does not have an underlying instrument, such as a cash instrument, the **underlyingName** will be the same as the symbol for the instrument.
IssuerId	Integer	The numeric identification of the issuer, that is, participant.

<table>
<tr><th>Field</th><th>Type</th><th>Description</th></tr>
<tr><td>settlementDate</td><td>Long</td><td>The standard settlement date for the order book. UTC time.</td></tr>
<tr><td>active</td><td>Boolean</td><td>Specifies if the order book is active or suspended.</td></tr>
<tr><td>indexMarket</td><td>Boolean</td><td>Specifies if this order book belongs to an index market, true, or not.</td></tr>
<tr><td>nominalValue</td><td>Long</td><td>The face value or amount that will be repaid when the bond matures.</td></tr>
<tr><td>decimalsInNominalValue</td><td>Integer</td><td>The number of implicit decimals in the nominal value.</td></tr>
<tr><td>fixedIncomeType</td><td>Byte</td><td>Bill, bond etc.</td></tr>
<tr><td>couponInterest</td><td>Long</td><td>The interest rate of the bond with 6 implicit decimals.</td></tr>
<tr><td>couponFrequency</td><td>Integer</td><td>Specifies the frequency that a fixed income security pays interest. Expressed as the number of coupons per year.</td></tr>
<tr><td>nextCouponDate</td><td>Long</td><td>Always set to zero.</td></tr>
<tr><td>dayCountConvention</td><td>Byte</td><td>This is to define a day count fraction to be used when calculating accrued interest and present value of future cash flows.</td></tr>
<tr><td>datedDate</td><td>Long</td><td>Date on which the coupon interest will begin to accrue for a bond. UTC time if JVM is set to UTC.</td></tr>
<tr><td>combinationLegs[ ]</td><td></td><td>Array of combinationLeg records, which consist of the following fields:
combinationLeg Record
<table>
<tr><th>Field</th><th>Type</th><th>Description</th></tr>
<tr><td>singleOrderBookId</td><td>Integer</td><td>The ID of the single order book</td></tr>
<tr><td>buyLeg</td><td>Boolean</td><td>A boolean indicating if this is the buy leg</td></tr>
<tr><td>ratio</td><td>Integer</td><td>The ratio between legs, for example buy 1A sell 3B.</td></tr>
<tr><td>priceQuotationFactor</td><td>Integer</td><td>This field is only used if the legs constitute both a derivative and the underlying instrument, such as a BUY/WRITE.</td></tr>
</table>
Example:
A combination order book to buy 1 id=1 and sell 2 id=2 would be sent as follows:
Item1: 1 true 1 1
Item2: 2 false 2 1

Note:
This field is null for a single order book.</td></tr>
<tr><td>tradingAtSettlement</td><td>Boolean</td><td>If the order book is configured for trading at settlement.</td></tr>
<tr><td>action</td><td>Byte</td><td>New/Update/Delete</td></tr>
<tr><td>participantDefined</td><td>Boolean</td><td>If the orderbook is participant defined.</td></tr>
</table>

Field	Type	Description
contractName	String	Contract Name
tradedInGenium	Boolean	Indicates if the instrument is OTC or traded in Nasdaq Financial Framework.
businessDate	Long	The current business date. Timezone is retrieved from environment variable TZ.
isinCode	String	The ISIN code of the instrument.
upperLevelOrderBookId	Integer	The numeric identification of the upper level OrderBook.
instrumentClassId	String	
derivativeLevel	Integer	The derivative level of the order book.
decimalsInContractSize	Integer	The number of decimals in the contract size.
decimalsInTermCurrencyQuantity	Integer	Specifies the number of decimals in the term currency quantity.
orderTypeAttributes	Short	The Order Attributes configured in CDB in relation to Order Types, Center Point, Undisclosed Quantity, Hidden Volume. If several values apply at the same time, the value of this field will be set to the sum of all applicable values.
orderValidityAttributes	Short	The Order Attributes configured in CDB in relation to Order Validity.
warrant	Boolean	If the message is a warrant.
corporateAction[ ]		Details of Corporation Actions. *corporateAction Record* Field / Type / Description: **code** — String — The corporate action code **type** — Short — The corporate action type: • 1 = Basis of Quotation • 2 = Status Note
underlyingIssuerName	String	The name of the underlying issuer.

Detailed Field Description

Value	Description
groupType	
0	Undefined. Combination order books do not have a group type and group type 0 shall be used in this case.
1	Option
2	Forward
3	Future
4	FRA
5	Cash
6	Payment
7	Exchange Rate

Value	Description
8	Interest Rate Swap
9	REPO
10	Synthetic Box Leg Or Reference
11	Standard Combination
12	Guarantee
13	OTC General
14	Equity Warrant
15	Security Lending
optionType	
0	Undefined
1	Call
2	Put
optionStyle	
0	Undefined
1	American
2	European
3	Asian
4	Bermudan
5	Knock In
6	Knock Out
7	Binary
8	Ratchet
currencyUnit	
0	Undefined
1	First
2	Second
priceUnit	
0	Undefined
1	Yield
2	Price
3	Point
4	Yield Diff
5	IMM Index
6	Inverted Yield
7	Basis Point
8	Percentage Of Nominal

Value	Description
9	Dirty Price
fixedIncomeType	
0	Undefined
1	Bill
2	Bond
3	Index Linked Bond
4	Floating Bond
5	Lottery Bond
6	Convertible Bond
7	Structured Bond
8	Fixing
9	Credit certificates
dayCountConvention	
0	Undefined
1	ACTACT
2	ACTAFB
3	EU30360
4	US30360
5	ACT365
6	ACT360
action	
1	New
2	Update
3	Delete
OrderTypeAttributes	
0	Undefined
1	LimitAllowed
2	MarketAllowed
4	MarketToLimitAllowed
8	AllOrNoneAllowed
16	BestLimitAllowed
32	CenterPointMarketAllowed
64	CenterPointLimitAllowed
128	CenterPointBlockMarketAllowed
256	CenterPointBlockLimitAllowed
512	CenterPointSweepMarketToLimitAllowed

Value	Description
1024	CenterPointSweepLimitAllowed
OrderValidityAttributes	
0	Undefined
1	FillOrKillAllowed
2	FillAndKillAllowed
4	FillAndStoreAllowed

4.2.5 Session Change Message

The Session Change message is sent when an order book changes session, for instance from a pre-open session to an open session.

- Message Group = 10
- Message ID = 4

Session Change Message

Field	Type	Description
timestamp	Long	The time when the message was sent. UTC time.
id	Integer	The ID of the session.
type	Short	The type number for the session used, for example, when entering a trigger on session order.
name	String	The name of the session, such as OPEN. **Note:** It is up to the customer to define these names, which means that it may be variants of the “OPEN” session.
matchingType	Byte	The type of matching, such as No Matching or Continuous Matching.
orderBookId	Integer	The order book, which has changed session.
level	Integer	Specifies on which level the session change was made, for instance market level or instrument type level. For details on the values of this field, see the configuration in CDB.
endofTrading	Boolean	Specifies if no more trades will occur in the system.

Detailed Field Description

Value	Description
level	
1	Market, that is, the session applies for all order books belonging to this market.
2	Instrument Type, that is, the session applies for all order books belonging to this instrument type (**exchangeId**, **marketId**, and **instrumentGroupId**).
3	Instrument Class, that is, the session applies for all order books belonging to this instrument class (**exchangeId**, **marketId**, **instrumentGroupId**, and **underlyingId**).
4	Order Book, that is, the session only applies for this order book.

Value	Description
5	Underlying (commodity), the session applies for all order books belonging to this underlying.
matchingType	
0	Undefined
1	Continuous matching
2	Fast Market matching
3	No matching
4	Auction matching

4.2.6 Quoting Responsibility Message

The Quoting Responsibility message is sent for all quoting responsibilities defined in the reference data.

- Message Group = 10
- Message ID = 12

Quoting Responsibility Message

Field	Type	Description
timestamp	Long	The time when the message was sent. UTC time.
id	Integer	The ID of the quoting responsibility.
orderBookId	Integer	The identification of the order book the quoting responsibility is for.
ParticipantId	Integer	The identification of the participant (market maker), which is responsible for quoting in the order book.
minimumQuantity	Long	The minimum quantity for the quotes.
maxSpreads[ ]		Array of **maxSpreadItem** records describing the maximum allowed spread for one or more price intervals. For each interval there is one record, which consists of the following fields: *maxSpreadItem Record* **lowerLimit** – Long **upperLimit** – Long **maxSpread** – Long **spreadUnit** – Byte – Set to 1 for percent and 2 for absolute. **decimalsInPrice** – Integer **Example:** Item1: 0.00 4.99 0.20 2 2 (Abs) Item2: 5.00 99.99 0.50 2 2 (Abs)

4.2.7 Market Message

The market message is sent for all defined markets in Nasdaq Financial Framework.

- Message Group = 10
- Message ID = 19

Market Message

Field	Type	Description
timestamp	Long	Time at which the message was sent. UTC time.
marketId	Short	Identifier of the market.
date	Long	Business date of the market (expressed in unix time, nano seconds since epoch).
closedForTrading	Boolean	If the market is closed for trading.
closedForClearing	Boolean	If the market is closed for clearing.
closedForSettlement	Boolean	If the market is closed for settlement.

4.2.8 Business Date Message

The business date message specifies the current business date.

- Message Group = 10
- Message ID = 6

Dates Message

Field	Type	Description
timestamp	Long	The time the message was sent. UTC time.
businessDate	Long	The current business date. UTC time.

4.3 Order and Trade Messages

4.3.1 Order Message

- Message Group = 10
- Message ID = 5

Order Message

Field	Type	Description
timeCreated	Long	The time at which the order was entered into the system, which may be many days ago. UTC time.
timeChanged	Long	The time at which the order was changed, for instance matched. UTC time.
orderBookId	Integer	The order book the order is for.
triggerOrderBookId	Integer	The order book the order will trigger on. Only applicable for trigger on price orders. For instance it is possible to enter a trigger order that triggers on prices from one orderbook, **triggerOrderBookId**, but when it is triggered it will be stored in another order book, **orderBookId**. For instance you may want to trigger on prices in the most liquid future order book (closest expiration), but want to match in another future order

Field	Type	Description
		book.
participantId	Integer	The participant to which the trader belongs.
userId	Integer	The trader owning the order.
onBehalfOfSubmitterId	Integer	Identifies the user submitting the order if not the same as the owner.
orderId	Long	The order identification, which will be global unique for ever.
previousOrderId	Long	The previous order identification. This is only applicable for quotes and will be populated with the **orderId** of the replaced quote when re-quoting. An updated single order will always keep its **orderId.**
clientOrderId	Char[]	The client order identification.
side	Byte	Buy or sell.
price	Long	The price of the order. Set to 0 for a market order.
orderQuantity	Long	The initial quantity of the order if the order has not been amended, that is, when an order is executed only the **leavesQuantity** is updated. However if the order is amended the **orderQuantity** may be changed.
leavesQuantity	Long	The remaining/open quantity of the order.
displayQuantity	Long	The display quantity of a reserve order. Set to Long.MIN_VALUE if not a reserve order.
minimumQuantity	Long	The minimum quantity that the order can trade. Currently only AoN is supported meaning that the Minimum Quantity is equal to the Quantity.
timeValidity	Integer	Defines the validity period for an order transaction, that is, the amount of time an order will remain in the order book if not fully matched. Of the two bytes in the field, the most significant byte (MSB) is used to define the unit of the time validity. If applicable, the least significant byte (LSB) specifies the value of the time validity, expressed in the unit defined in the most significant byte. **Example:** To enter an order, which is to be valid for the rest of the day, use MSB=1 and LSB=0. In binary representation this is MSB=00000001 and LSB=00000000, yielding that the Validity Time field should be set to 00000001 00000000 in binary representation, or 256 in decimal representation. There is an alternative way to enter a time validity when the value exceeds 255. If the top bit of the 2 bytes is set, the 3 next bits are used to define an alternative unit while the remaining 12 bits specify the value expressed in the alternative unit. **Example:** To enter an order, which is to be valid for 365 days. Enter in binary representation 1 001 000101101101 (1 = to indicate that the field has an alternative unit, 001 = alternative unit to specify number of days, 000101101101 = 365 to indicate number of days).
orderType	Short	The order type, such as Limit or Market. If several values apply at the same time, the value of this field will be set

Field	Type	Description
		to the sum of all applicable values.
exchangeOrderType	Integer	The exchange order type, such as Short Sell or Session State Order. If several values apply at the same time, the value of this field will be set to the sum of all applicable values. **Example:** If a Sweep order is sent as a Short sell, the Exchange order Type will be 2048 + 2 = 2050
orderCategory	Byte	The order category, such as Order, Quote, Bait. If several values apply at the same time, the value of this field will be set to the sum of all applicable values.
accountId	Char[]	The account to use for this order (called exClient in the external API).
exchangeInfo	Char[]	This field contains exchange-specific information that can be entered with the order. See *Nasdaq OMN Interface Message Reference* for details about the field.
customerInfo	Char[]	This field is a free text field filled in by the client entering the transaction.
changeReason	Short	The reason for sending this message.
triggerCondition	Byte	Specifies what price to compare the trigger price with as well as the trigger direction.
triggerPrice	Long	The trigger price of the order.
triggerSessionType	Short	The trigger session type is used to specify which session type to trigger on. **Note:** This field is only populated for trigger on session orders.
orderStatus	Byte	The status of the order, such as if it is placed on the order book or not.
orderStatusBefore	Byte	The status of the order before the event that caused this message to be sent.
orderBookPosition	Integer	The ranking position when the order first was added to the order book. This field will also be populated if the order has been amended and as a result of this loose priority. The field will be 0 (zero) if the order is updated in a way that it does not loose priority, such as when the order trades.
reloaded	Boolean	Indicates if the order was reloaded at start of the system at a new business date.
giveUpParticipant	Char[]	This field should be populated with the Clearing Participant ID, and may be optionally prefixed with the 2 character country code i.e. “AU”. Hence, it concatenates two parts of give_up_member into a char array of length 7. country_id = first 2 positions, ex_customer = remaining 5 positions, which in essence is Clearing Participant ID.
tradeReportCode	Byte	Specifies the code of the trade report configuration parameters for the reported trade. Note that this field is only applicable for trade reports.
requestedPosition	Byte	Specifies the requested position handling, such as closing or opening position.

Field	Type	Description
messageName	Integer	The message ID with which the order was submitted. For example, if entered with MO31 will be 31.
rankingTime	Long	Ranking time of the order. UTC time.
midTick	Short	Specifies if the wanted price is a half tick better than the given price and if dark execution of order shall be allowed. Dark execution only valid for limit orders.
preferenceOnly	Short	Whether centre point matching should be done only against own orders.
singleFillMinimumQuantity	Short	Whether centre point orders minimum quantity must be satisfied in a single fill.
crossingKey	Integer	The key to be used for booking of crossed trades. If enabled any 2 orders from the same participant and with the same key (non zero) which match will be marked as a crossed trade and not publish to the market as a trade.
regulatoryData	Char[]	The regulatory data required to be submitted with orders and trade reports.
shortSellQuantity	Long	The short sell quantity of the order.
participantOrderAttribute	Long	The participant order attribute of an order.
counterOrderAttributes	Long	The counter order attributes with which an order with crossed participant order attributes may trade.
bidPriceSnapshot	Long	Best bid price snapshot from order details.
offerPriceSnapshot	Long	Best offer price snapshot from order details.
submitterId	Integer	ID of the submitter.
totalMatchedQuantity	Long	Total cumulative matched quantity.
transactionStatus	Integer	The answer to the user in response to the initiating transaction. Contains the same information as Txstat and indicates the action taken as a result of the successful transaction.
nationalBidPriceSnapshot	Long	National bid price snapshot at the time of update.
nationalOfferPriceSnapshot	Long	National offer price snapshot at the time of update.
transferFromUserId	Integer	The ID of the transferring user where the order is a transferred order.
deltaQuantity	Short	Defines if the order changed quantity is an absolute value, a delta change to the original quantity, or an absolute value in relation to the original order quantity (explicit).
blockSize	Long	Minimum number of units (options, futures, forwards and so on) in an order transaction.

Detailed Field Description

Value	Description
orderType	
0	Undefined.
1	Limit order.
2	Market order.

Value	Description
3	Market to limit order.
4	Passive order, that is, cannot match as aggressive.
8	Only at best order (currently not used).
16	Pegged to best price order.
32	Odd lot order (currently not used).
64	Imbalance Order.
exchangeOrderType	
0	Undefined.
1	Force.
2	Short sell.
4	Market bid.
8	Price Stabilization.
16	Override crossing.
32	Undisclosed quantity order.
64	Center point order.
128	Always inactive order.
256	Center Point Crossing order.
512	Trigger on session order (SSO).
1024	Peg order.
2048	Center point sweep order.
4096	Center point block order.
orderCategory	
0	Undefined.
1	Order.
2	Hybrid Quote.
4	Quote.
8	Bait generated for a combination order.
16	Combination Leg order generated when a combination order matches in the leg order books.
32	Trade Report.
64	Crossing Order
changeReason	
0	Undefined.
1	Order canceled by the trader.
3	Order traded.
4	Order inactivated when a user lost connection to the exchange (configurable).
5	Order updated by user.

Value	Description
6	New order.
7	Market order converted – Modified to EP during auction if an auction (market) order is modified during auction.
8	Market order converted to a Limit order (MTL).
9	Order canceled by system.
10	Order canceled on behalf.
11	Bait re-calculated.
12	Trigger order triggered and converted to an active order.
13	Reserved order refreshed.
15	Order canceled by system due to a price limit change.
19	Order expired due to last trading day for the order.
20	Order canceled due to that trading is halted.
21	Order inactivated due to that trading is halted.
23	Order inactivated/purged due to corporate action.
24	Rest-Of-Day order inactivated/purged.
25	Order inactivated due to delisting.
26	Other than Rest-Of-Day order inactivated/purged.
27	Order inactivated/purged due to being outside the purge price limits.
28	Order Ownership transferred.
29	New inactive order.
30	Order reloaded by system on new trading day.
31	Order reload at intraday Market Place restart.
34	Order canceled after opening auction.
35	Order inactivated/purged due to being outside the price limits.
36	Order activated due to changed price limits and the order premium is inside the new limits.
37	Trigger on session order (SSO) triggered due to session change.
38	Trigger on session order inactivated.
39	Undisclosed quantity order converted to a regular order.
40	Volume match order inactivated due to order value.
41	Quote canceled by system due to the delta limit in market maker protection exceeded.
42	Quote canceled by system due to the absolute (quantity) limit in market maker protection exceeded.
43	Order deleted because trader is not allowed to trade with himself
48	Center point sweep MTL order converted.
49	MAQ center point order trades below minimum quantity.
50	Center point sweep order re-loaded without MAQ and mid-tick conditions.
51	Order canceled due having an invalid clearing participant ID.

Value	Description
100	Canceled by system due to session lost.
101	Mass Cancel by User.
triggerCondition	
1	Bid greater than or equal to.
2	Bid less than or equal to.
3	Offer larger than or equal to.
4	Offer less than or equal to.
5	Last Paid greater than or equal to.
6	Last Paid less than or equal to.
orderStatus	
1	The order is stored on the order book.
2	The order is not stored on the order book, for instance an incoming order that has yet not been handled or an order that has been canceled.
3	A trigger order that has not yet been triggered.
4	An inactive order.
timeValidity	
MSB set to 0	Bouncing. The order will not be stored in the order book after the completion of order transaction, if the order is not fully matched. LSB should be set to zero.
MSB set to 1	Rest Of Day. The order will be stored in the order book for the remainder of the business day. LSB should be set to zero
MSB set to 2	Good Till Canceled. The order will be stored in the order book until the instrument expires or the order is canceled. LSB should be set to zero.
MSB set to 5	Days. The order will be stored in the order book for he number of days specified in LSB.
MSB set to 6	Current Max. The order will be stored in the order book for the maximum amount of time allowed for the instrument. LSB should be set to zero.
MSB set to 32	Current Max. The order will be stored in the order book until end of the session type specified in LSB.
MSB set to 64	Purged. LSB should be set to zero.
MSB set to 128	Negative Indicator. LSB should be set to zero.
requestedPosition	
0	Default for the account.

Value	Description
1	Open.
2	Close.
3	Mandatory close.
4	Set to default for account (valid only when updating an order).
midTick	
0	Undefined
1	Yes
2	No
3	Dark execution
4	Dark execution with midtick
5	Any price block execution
6	Any price block execution with midtick
TransactionStatus	
1	No part of the order placed in the Order book and no part closed.
2	The whole order closed.
3	The order partially closed and nothing placed in the Order Book.
4	The whole order placed in the Order Book.
6	The order partially placed in the Order Book and partially closed.
17	Circuit breaker started, no part of the order placed in the Order Book and no part closed.
19	Circuit breaker started, the order partially closed and nothing placed in the Order Book.
side	
0	Undefined
1	Buy
2	Sell
preferenceOnly	
1	Yes
2	No
singleFillMinimumQuantity	
1	Yes
2	No
deltaQuantity	
1	Absolute Quantity
2	Delta Quantity
3	Explicit Quantity

4.3.2 Rejected Order Message

The rejected order message summarizes a rejected order insert or update. It includes the reason for rejection and some of fields entered, such as price.

> **Note:**
> Some fields may not be populated in some messages. For example, orderId is only populated for order updates and will be 0 for a rejected order insert.

- Message Group = 10
- Message ID = 22

Rejected Order Message Additional Fields

Field	Type	Description
userId	Integer	User ID of the owner of the order.
orderId	Long	Order ID (only populated for order update).
orderBookId	Integer	Submitted order-book ID.
side	Byte	Submitted side.
price	Long	Submitted price.
quantity	Long	Submitted quantity.
errorCode	Integer	Error code indicating why the order entry/update was rejected.
timestamp	Long	The timestamp of the rejected order update. Message does not have a timestamp.

Detailed Field Description

Value	Description
side	
0	Undefined
1	Buy
2	Sell

4.3.3 Trade Message

- Message Group = 10
- Message ID = 7

Trade Message

Field	Type	Description
tradeTime	Long	The timestamp when the trade was made. UTC time.
orderBookId	Integer	The order book for the trade.
userId	Integer	The trader that has traded.
participantId	Integer	The participant the trader belongs to.
orderId	Long	The order identification.

Field	Type	Description
quoteMessageId	Long	If the trade is from a quote, this identifies the quote message in which this quote was entered. For orders this field will be 0.
matchId		The trade identification. This is a record which consists of the fields below. **Note:** This field is preceded by a boolean written by the serializer. If this boolean is set to **true**, the field **matchId** contains data. *matchId Record* **matchGroupId** — Long — A monotonically increasing number per partition, assigned to an execution event. **notUsed** — Integer — This field is not used. **combinationMatchId** — Integer — Sequential number of an execution sequence number.
orderPrice	Long	The order price.
tradePrice	Long	The trade price.
averagePrice	Long	The average trade price, such as VWAP price.
quantity	Long	The traded quantity.
side	Byte	Buyer or seller.
dealSource	Short	Specifies under which circumstances the deal was made, such as in continuous matching or in the opening auction.
tradeType	Byte	The status of the trade, such as Standard, Reversing or Overtaking.
passiveAggressive	Byte	Specifies if this was the aggressive side or passive side in the deal. It could also be set to neither of this.
accountId	Char[]	The account that was used for the deal.
exchangeInfo	Char[]	This field contains exchange-specific information that can be entered with the order. See *Nasdaq OMN Interface Message Reference* for details about the field.
customerInfo	Char[]	This field is a free text field filled in by the client entering the transaction.
settlementDate	Long	The date for settlement of this deal. UTC time if JVM is set to UTC. **Note:** Only set for trade reports.
yieldOrPrice	Long	If the order book is traded in yield, specified by the price Unit field in the Order Book message, this field is populated with the corresponding clean price. If the order book is trade in clean price or dirty price this field is populated with the corresponding yield. Only applicable for fixed income instruments.
accruedInterest	Long	The accrued interest for this deal.

Field	Type	Description
		Only applicable for fixed income instruments.
giveUpParticipant	Char[]	This field should be populated with the Clearing Participant ID, and may be optionally prefixed with the 2 character country code i.e. “AU”. Hence, it concatenates two parts of give_up_member into a char array of length 7. country_id = first 2 positions, ex_customer = remaining 5 positions, which in essence is Clearing Participant ID.
originalTrade	Boolean	Specifies if this is the original trade received directly from the matching engine, true, or a rectified trade received from the clearing system, false.
tradeReportCode	Byte	Specifies the code of the trade report configuration parameters for the reported trade. Note that this field is only applicable for trade reports.
reportTime	Long	The timestamp when the trade was reported (trade reports only). UTC time.
extendedPrice	Long	The trade price with additional decimals.
shortSellQuantity	Long	The short sell quantity of the trade.
tradeSlipNumber	Long	Trade slip number, an unique identifier per deal. Format is 1DINNNNNNN where 1 = always set to "1", D = last digit of date calculated as number of days since 31/12 previous year up to current business date MOD 10, I = running instance of server, NNNNNNN = running number of deal with a value range of 0000001 to 9999999 per Trade Handler instance for Matching Engine deals (resets to 0000001 every new business day).
nationalBidPriceSnapshot	Long	National bid price snapshot at the time of update.
nationalOfferPriceSnapshot	Long	National offer price snapshot at the time of update.
tradeCondition	Integer	The condition in which a trade was executed.
counterOrderCapacity	Byte	The order agent's capacity for cross trades.

Detailed Field Description

Value	Description
dealSource	
0	Undefined.
1	Two orders matched in continuous matching.
3	Reported Trades between different participants, that is, the buyer and seller belongs to different participants.
4	Reported Trades by the exchange between different participants.
5	Reported Trades for the same participant, that is, the buyer and seller belong to the same participant.
6	Reported Trades by the exchange for the same participant.
7	Two orders for a standard combination order book matched in continuous matching.
20	Two orders match in an (opening) auction.
36	Two orders for a Tailor Made combination order book (TMC) matched in continuous matching.
42	Priority crossing.
43	Combination order against outright.

Value	Description
44	Book Trade Continuous Matching.
45	Book Trade Auction.
46	Preference Matched.
47	CentrePoint
48	BookTradeCentrePoint
49	Preference Only Matched.
50	Any Price Block Matched.
51	Preference Any Price Block Matched
52	Preference Only Any Price Block Matched.
90	Two orders matched in Post Close session.
91	Book Trade during Post Close session.
tradeType	
0	Undefined.
1	Standard.
2	Transitory.
3	Overtaking.
4	Reversing.
5	Transfer.
6	Exercise.
7	Assign.
8	Closing.
9	Issuing.
10	New Contract.
passiveAggressive	
0	Passive
1	Aggressive
2	Neither
side	
0	Undefined
1	Buy
2	Sell
tradeCondition	
0	No condition
2	Internal Trade/Crossing
8	Buy Write
0	NotApplicable

Value	Description
1	Excluded
2	Reincluded

4.3.4 Circuit Breaker Informational Message

The circuit breaker informational message will be sent whenever a circuit breaker is tripped. It contains information about which order book the circuit breaker triggered in and why the circuit breaker was triggered.

- Message Group = 10
- Message ID = 24

Circuit Breaker Info Message

Field	Type	Description
orderBookId	Integer	The order book id where the circuit breaker has tripped.
incomingOrder		Circuit Breaker Incoming Order Info Record: *incomingOrder Structure* Field / Type / Description: **orderBookId** / Integer / The order book id where the circuit breaker has tripped. Always the same value as the higher level **orderBookId**. **userId** / Long / Owner of the order **orderId** / Long / The order ID **price** / Long / Order's limit price **quantity** / Long / Remaining order quantity **side** / Byte / Set to 1 for buy and 2 for sell.
hitOrder		Circuit Breaker Hit Order Info Record: *hitOrder Structure* Field / Type / Description: **userId** / Long / Owner of the order **orderId** / Long / The order ID **price** / Long / Order's limit price **quantity** / Long / Remaining order quantity **side** / Byte / Set to 1 for buy and 2 for sell.
triggerDetail		The circuit breaker trigger details: *Circuit Breaker Trigger Details* Field / Type / Description: **matchPrice** / Long / The attempted match price **matchQuantity** / Long / The attempted match quantity
triggerCondition		The triggering condition:

Field	Type	Description
		Triggering Condition Structure Field / Type / Description **upperLimit** / Integer / Circuit Breaker upper limit **lowerLimit** / Integer / Circuit Breaker lower limit
sessionSequenceName	String	The session sequence name of the triggered circuit breaker.

4.3.5 Quote Request Message

The Quote Request message is sent when a quote request is received by the system.

- Message Group = 10
- Message ID = 18

Quote Request Message

Field	Type	Description
timestamp	Long	Timestamp when the quote request was received. UTC time.
orderBookId	Integer	Order book for the trade.
userId	Integer	Trader that has traded.
quoteRequestId	Long	Quote request Id.
side	Byte	Buyer or Seller.
quantity	Long	
onBehalfOfSubmitterId	Integer	User who entered the request (only for on-behalf requests)

4.4 Price Information Messages

4.4.1 Price Information Message

The Price Information Message is sent to report the edited price information per orderbook.

- Message Group = 10
- Message ID = 31

Price Information Message

Field	Type	Description
timestamp	Long	The time the message was sent. UTC time.
orderBookId	Integer	The order book the price is for.
openingPrice	Long	The first traded price for the day.
highPrice	Long	The highest traded price during the day.
lowPrice	Long	The lowest traded price during the day.
closingPrice	Long	The last traded price for the previous day.
lastPrice	Long	The last traded price during the day.

Field	Type	Description
extendedOpeningPrice	Long	The first traded price during the day in extended price format.
extendedHighPrice	Long	The highest trade price during the day in extended price format.
extendedLowPrice	Long	The lowest trade price during the day in extended price format.
extendedLastPrice	Long	The last trade price during the day in extended price format.
volume	Long	Trade volume.
turnOver	Long	The number of traded contracts during the day. If there are 100 contracts in a deal (100 bids and 100 asks) the turnover will increase with 100.
turnOverValue	Long	The total traded amount today.
numberDeals	Integer	The number of deals executed today.
numberTrades	Integer	Number of trades executed during the day.
updatedTimestamp	Char[]	Time in ASCII. Format: HHMMSS. UTC time.
reason	Byte	Reason why the Price Information Message was sent.
dealSource	Short	Specifies under which circumstances the deal was made, such as in continuous matching or in the opening auction. For more information about the values of this field, see *Trade Message*.
trendIndicator	Char	Trend indicator for the latest price compared to the previous one.
extendedLastPrice	Long	The last trade price during the day in extended price format.

Detailed Field Description

Value	Description
trendIndicator	
+	Up
-	Down
=	Same
<null>	None
reason	
0	Void and not used
1	Sent due to refresh of data
2	Sent due to execution of deal
3	Sent due to correction of data
4	Sent due to deletion of deal
5	Sent due to exclusion of deal in trade statistics
6	Sent due to reinclusion of deal in trade statstics
7	Sent due to reset of trade statstics

4.4.2 Equilibrium Price Message

The Equilibrium Price message is sent when the equilibrium price for an order book has been calculated.

- Message Group = 10
- Message ID = 8

Equilibrium Price Message

Field	Type	Description
timestamp	Long	The time at which the message was sent. UTC time.
orderBookId	Integer	The order book the equilibrium price is for.
equilibriumPrice	Long	The equilibrium price for the order book.
bidQuantity	Long	The aggregated bid quantity at the equilibrium price or better.
offerQuantity	Long	The aggregated offer quantity at the equilibrium price or better.
bidImbalanceQuantity	Long	If the imbalance is on the bid side this field is populated with **bidQuantity** – **offerQuantity**, otherwise set to 0.
offerImbalanceQuantity	Long	If the imbalance is on the offer side this field is populated with **offerQuantity** – **bidQuantity**, otherwise set to 0.
sessionId	Integer	The ID of the current session for the order book.
bestBidPrice	Long	The best bid price.
bestBidQuantity	Long	The aggregated quantity at the best bid price.
bestOfferPrice	Long	The best offer price.
bestOfferQuantity	Long	The aggregated quantity at the best offer price.

4.4.3 Index Price Message

The Index Price message is sent when the index has been re-calculated.

- Message Group = 10
- Message ID = 16

Index Price Message

Field	Type	Description
timestamp	Long	The time at which the message was sent. UTC time.
orderBookId	Integer	The order book the index price is for.
price	Long	The index price.
changePrevious	Integer	Percentage change since previous corresponding information dissemination.
changeYesterday	Integer	Percentage change compared to the previous day's value.
pointsOfMovement	Integer	The change between two index values expressed as number of points. The value includes implicit decimals with the number as of the index itself.
calculationTime	Char[]	The date and time of the Index Price calculation in UTC.
distributionTime	Char[]	The date and time of the Index Price distribution in UTC.

4.4.4 Price Limits Message

The Price Limits message is sent when the price limits for an order book has changed.

- Message Group = 10
- Message ID = 13

Price Limits Message

Field	Type	Description
timestamp	Long	The time at which the message was sent. UTC time.
orderBookId	Integer	The order book the limits are for.
upperLimit	Long	The upper limit.
lowerLimit	Long	The lower limit.
priceLimits	Boolean	Price limits or circuit breaker limits, price limits = true.
dynamic	Boolean	Dynamic or static limits, dynamic = true.
referencePrice	Long	The reference price used when calculating the limits.

4.4.5 Reference Price Message

The Reference Price message is sent when the reference price for an order book has changed.

- Message Group = 10
- Message ID = 14

Reference Price Message

Field	Type	Description
timestamp	Long	The time the message was sent. UTC time.
orderBookId	Integer	The order book the reference price is for.
referencePrice	Long	The reference price used when calculating the equilibrium price as well as when calculating the leg prices for a combination/combination match, and there is no BBO for the leg order book.
referencePriceSource	Short	Not in use. Always set to zero.
priceLimitsReferencePrice	Long	The reference price used when calculating the static price limits.
circuitBreakerLimitsReferencePrice	Long	The reference price used when calculating the static circuit breaker limits.
priceLimitsReferencePriceSource	Short	The source for the price limits reference price, such as externally injected or last business day's closing price. See *Detailed Field Description* table for **PriceSourceIndicator** below.
circuitBreakerLimitsReferencePriceSource	Short	The source for the circuit breaker limits reference price, such as externally injected or last business day's closing price. See *Detailed Field Description* table for **PriceSourceIndicator** below.

Detailed Field Description

Value	Description
PriceSourceIndicator	
0	Externally injected.

Value	Description
1	Opening price.
2	Uncross price.
3	Previous Last Paid (Last business day's closing price).
4	Settlement price.
5	Ever last paid.
6	Opening price, no uncross.
7	Price limit reference price.
8	Settlement price diff.

4.4.6 Settlement Price Message

The settlement price message provides a real-time update of settlement prices received by the matching engine.

- Message Group = 10
- Message ID = 20

Daily Settlement Price Message

Field	Type	Description
timestamp	Long	The date and time of receipt of the settlement price by the matching engine. UTC time.
orderBookId	Integer	The order book the settlement price applies to.
settlementPrice	Long	The settlement price value.
settlementPriceType	Integer	The type of settlement price.

Detailed Field Description

Value	Description
settlementPriceType	
2	Normal

4.4.7 Event Message

The Event message is used to notify that certain information is at hand, or that specific events have occurred. The nature of the message lies within the broadcast's information type and is interpreted according to the list given in the documentation of the field infoType.

- Message Group = 10
- Message ID = 32

Events Message

Field	Type	Description
timestamp	Long	The time the message was sent. UTC time.
level	Integer	Specifies on which level the signal was issued, for instance market level or instrument type level.

Field	Type	Description
marketId	Integer	Identifier of the market for market level updates.
orderBookId	Integer	Identifier of the order book for order book level updates.
infoType	Integer	The code denoting the particular information type relevant to the event.
businessDate	Long	The current business date at time of the event. UTC time.
clearingDate	Short	The current clearing date at time of the event. UTC time.

4.4.8 Market Maker Protection Message

The Market Maker Protection message provides the calculated absolute and delta quantity protection for market makers per underlying instrument. The message is issued in real time when Market Maker protection is triggered.

- Message Group = 10
- Message ID = 33

Market Maker Protection Message

Field	Type	Description
timestamp	Long	The time of issue of the new calculated quantity protection values. UTC time.
participantId	Integer	Numeric identification of the market making participant to which the protection quantities apply.
underlyingId	Integer	The identifier of the underlying instrument to which the protection quantities apply.
calculatedQuantityProtection	Long	The calculated quantity value for market maker protection.
calculatedDeltaQuantityProtection	Long	The calculated delta quantity value for market maker protection.

4.4.9 Open Balance Message

The Open Balance message is sent at start of the system to inform about the open balance from yesterday.

- Message Group = 10
- Message ID = 15

Open Balance Message

Field	Type	Description
timestamp	Long	The time the message was sent. UTC time.
orderBookId	Integer	The order book.
openBalance	Long	The open balance from yesterday.

4.4.10 External Prices Message

The external prices message specifies the external bid and offer price.

- Message Group = 10
- Message ID = 26

External Prices Message

Field	Type	Description
timestamp	Long	The time the message was sent. UTC time.
orderBookId	Integer	The order book the external price is for.
userId	Integer	The user session responsible for submitting the external price to the central system.
bidPrice	Long	The bid price.
offerPrice	Long	The offer price.
creationTime	Long	The date and time the external price is received by the central system. This differs to the timestamp field which represents the time the external price is received by the matching engine. Timezone sent in with MO2148.

4.5 Miscellaneous Messages

4.5.1 News Message

A Market Control Message is sent when the Market Control staff wants to send a message. It is normally sent to a whole market, that is, with Level set to Market.

Market Control Message

(destination_level_c = 1) but it can sometimes be sent on Underlying or Series level. This message will be sent when the message type is set to Market Message/Market Control (message_information_type_c = 2). It can be sent with three different priorities: normal, high and low. A Market Control Message can be sent with destination to all markets. This is indicated by series in destination_item is set to null (no specific market is indicated), and destination_level_c = 1 (Market level).

Company Announcement

with Level set to Underlying (destination_level_c = 2) or Series level (destination_level_c = 3). This message will be sent when the message type is set to Company Announcement (message_information_type_c = 1). It can be sent with three different priorities: normal, high and low.

- Message Group = 10
- Message ID = 17

News Message

Field	Type	Description
timestamp	Long	The time the message was sent. UTC time.
sequenceNumber	Integer	A sequence number starting at 1 every day.
exchangeId	Short	The numeric identification of the exchange. This field is only populated if the **destinationLevel** is Market or Instrument.
marketId	Short	The numeric identification of the market. This field is only populated if

Field	Type	Description
		the **destinationLevel** is Market or Instrument.
instrumentGroupId	Short	The numeric identification of the instrument group. This field is only populated if the **destinationLevel** is Instrument.
modifier	Short	Expiration date modifier. This value is set to zero when the instrument is new. The value is incremented by one each time the instrument is involved in an issue, split, etc. Note that the modifier value can be different for bid and ask options in the same Series.
underlyngId	Integer	The numeric identification of the underlying. This field is only populated if the **destinationLevel** is Underlying or Instrument.
strikePrice	Integer	The strike price for this order book. This field is only populated if the **destinationLevel** is Instrument.
expirationDate	Integer	The expiration date. This field is only populated if the **destinationLevel** is Instrument.
messagePriority	Byte	The priority of the message.
destinationLevel	Byte	The level the news applies for, the whole market, a specific underlying or a specific instrument.
typeOfInformation	Byte	The type of message sent in announcement.
header	Char[]	80 characters header.
source	Char[]	80 characters source.
text[ ]	Char[]	10 lines of text where each text is 80 characters.
urlLength	Integer	The actual length of the URL.
url	Char[]	255 characters URL pointing to a document where, for example, a full announcement can be found.

Detailed Field Description

Value	Description
messagePriority	
1	Low.
2	Medium.
3	High.
4	Critical.
typeOfInformation	
0	Not applicable.
1	Company announcement.
2	Market message.
3	Static line.
4	Notice received.
destinationLevel	
0	Not applicable.

Value	Description
1	Market.
2	Underlying.
3	Series.

4.5.2 Clearing Trade Message

The Clearing Trade Message provides trade details derived from post match handling.

- Message Group = 10
- Message ID = 30

Clearing Trade Message

Field	Type	Description
orderBookId	Integer	The order book the trade is for.
combinationOrderBookId	Integer	Outright trade, identifies the combination in which the trade was made.
userId	Integer	The trader that has traded.
participantId	Integer	The participant the trader belongs to.
orderId	Long	The order identification.
matchId[ ]		The trade identification. This is a record which consists of the following fields: *matchId Record* Field / Type / Description **matchGroupId** / Long / A monotonically increasing number per partition, assigned to an execution event. **notUsed** / Integer / This field is not used. **combinationMatchId** / Integer / Sequential number of an execution sequence number. **Note:** This field is preceded by a boolean written by the serializer. If this boolean is set to **true**, the field **matchId** contains data.
price	Long	The trade price.
quantity	Long	The traded quantity.
side	Byte	Buyer or seller.
dealSource	Short	Specifies under which circumstances the deal was made, such as in continuous matching or in the opening auction.
tradeType	Byte	The status of the trade, such as Standard, Reversing or Overtaking.
accountId	Char[]	The account that was used for the deal.
customerInfo	Char[]	This field is a free text field filled in by the client entering the transaction.

Field	Type	Description
settlementDate	Long	The date for settlement of this deal. UTC time if JVM is set to UTC.
giveUpParticipant	Char[]	This field should be populated with the Clearing Participant ID, and may be optionally prefixed with the 2 character country code i.e. “AU”. Hence, it concatenates two parts of the of give_up_member into a char array of length 7. country_id = first 2 positions, ex_customer = remaining 5 positions, which in essence is Clearing Participant ID.
tradeReportCode	Byte	The code of the trade report configuration parameters for the reported trade. Note that this field is only applicable for trade reports.
tradeSlipNumber	Long	A unique identifier per deal on the format 1DINNNNNNN: • The Trade Slip Number always starts with "1". • D is set to the number of days since 1/1 up to current business date + 50 MOD 10. • I is set to the running instance of server. • NNNNNNN is set to a sequential number representing the deal. The value ranges from 0000001 to 9999999 and resets to 0000001 every new business day.
bigAttention	Long	Provides information about the trade.
exchangeOrderType	Short	The exchange order type, such as Short Sell or Session State Order.
executionTimestamp	Long	Timestamp of the execution. UTC time.
openCloseReq	Short	Describes how the requested position account should be updated
tradeCondition	Integer	The condition in which a trade was executed.
regulatoryData	Char[]	Data required to be submitted with orders and trade reports.
shortSellQuantity	Long	The short sell quantity of the trade.
tradeReportAttribute	Byte	Trade report attribute.
orderType	Short	The order type, such as Limit or Market.
counterOrderCapacity	Byte	If a trade is a cross, this field publishes the opposing orders Agent Capacity.
tradeNumber	Integer	An increasing sequence number assigned to each trade. Trade number is unique within Instance and Instrument type.
asOf	Long	The date an object is valid for. UTC time.
timeStamp	Long	When the message was sent. UTC time.
extendedPrice	Long	Trade Report related extended price.
modifiedTime	Long	Defines what time the item was last changed. UTC time.
timeOfAgreement	Long	Time when the trade was agreed. UTC time in production.
submitterId	Integer	ID of the submitter.
exchangeInfo	Char[]	This field contains exchange-specific information that can be entered with the order. The content is the same as in the field with the same name in *Trade Message*. See *Nasdaq OMN Interface Message Reference* for details about the field.

Detailed Field Description Cont.

Value	Description
tradeCondition	
0	No condition.
1	Late Trade.
2	Internal Trade/Crossing.
4	Bulletin Board Trade.
8	Buy Write.
16	Off Market.
openCloseReq	
0	Default for the account.
1	Open.
2	Close/net.
3	Mandatory close.
4	Set to default to the account (valid only for alter order).
tradeReportAttribute	
0	No trade report.
1	Initial trade report.
2	Regular/delayed trade report.

5 Appendices - Scenarios

An incoming order will always result in that two Order messages are sent. The first Order message exactly mirrors the incoming order before it has been processed by the matching engine. The second Order message mirrors the order after being processed by the matching engine. The second Order message hence gives information about if and how match the order matched, if the order was added to the order book, and so on.

Within a transaction the Order message for the incoming order will be sent before any other Order messages for orders impacted within the transaction.

5.1 Order entered and added to the order book

Message	id	price	orderQuantity	leavesQuantity	orderStatus	changeReason
StartOfTransaction	1					
Order	1	10	100	100	Incoming (2)	New (6)
Order	1	10	100	100	Active (1)	New (6)
Commit						

5.2 Order entered but not added to the order book

Message	id	price	orderQuantity	leavesQuantity	orderStatus	changeReason
StartOfTransaction	1					
Order	1	10	100	100	Incoming (2)	New (6)
Order	1	10	100	100	Incoming (2)	New (6)
Commit						

5.3 Order entered and partly trades before added to the order book

Order 2 is entered and is partly matched.

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction	2					
Order	2	10	100	100	Incoming (2)	New (6)
Order	2	10	100	50	Active (1)	Trade (3)
Order	1	10	50	0	Canceled (2)	Trade (3)
Trade	2	10	50			Auto (1)
Trade	1	10	50			Auto (1)
Commit						

5.4 Order entered and fully traded

Order 2 is entered and is fully matched.

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction	2					
Order	2	10	100	100	Incoming (2)	New (6)
Order	2	10	100	0	Incoming (2)	Trade (3)
Order	1	10	200	100	Active (2)	Trade (3)
Trade	2	10	100			Auto (1)
Trade	1	10	100			Auto (1)
Commit						

5.5 Trigger order entered without triggering

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction	1					
Order	1	10	100	100	Incoming (2)	New (6)
Order	1	10	100	100	Untriggered (3)	New (6)
Commit						

5.6 Trigger order entered and triggered immediately

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction	1					
Order	1	10	100	100	Incoming (2)	New (6)
Order	1	10	100	100	Active (1)	Triggered (12)
Commit						

5.7 Order updated and added to the order book

The quantity is changed from 100 to 200.

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction	1					

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
Order	1	10	200	200	Active (1)	Updated (5)
Order	1	10	200	200	Active (1)	Updated (5)
Commit						

5.8 Order updated, traded and added to the order book

The price of order 2 is changed from 10 to 11 and hence the order can trade.

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction	2					
Order	2	11	100	100	Active (1)	Updated (5)
Order	2	11	100	50	Active (1)	Trade (3)
Order	1	11	50	0	Canceled (2)	Trade (3)
Trade	2	10	50			Auto (1)
Trade	1	10	50			Auto (1)
Commit						

5.9 Order updated and fully traded

The price of order 2 is changed from 10 to 11 and hence the order can trade.

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction	2					
Order	2	11	100	100	Active (1)	Updated (5)
Order	2	11	100	0	Canceled (2)	Trade (3)
Order	1	11	200	100	Active (1)	Trade (3)
Trade	2	10	100			Auto (1)
Trade	1	10	100			Auto (1)
Commit						

5.10 Order canceled by user

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction	1					
Order	1	10	100	100	Canceled (2)	Canceled (1)
Commit						

5.11 Reserve order refreshed

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction	1					
Order	1	10	100	90	Active (2)	Refreshed (13)
Commit						

5.12 Combination order entered generating baits

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource	orderBookId
StartOfTransaction	1						
Order	1	0	100	100	Active (2)	New (6)	101
Order	1	10	100	100	Active (2)	New (6)	1
Order	1	10	100	100	Active (2)	New (6)	2
Commit							

5.13 Order entered and Circuit Breaker triggered

Message	id	price	orderQuantity	leavesQuantity	orderStatus	changeReason
StartOfTransaction						
Equilibrium Price						
Order	1	10	100	100	Incoming (2)	New (6)
Order	1	10	100	100	Active (1)	New (6)
Circuit Breaker Message						
Commit						

5.14 Uncrossing auction of multiple instruments

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction						
Session Change						
Equilibrium Price						
Order	1	10	100	100	Incoming (2)	Trade (3)
Order	2	10	100	100	Incoming (2)	Trade (3)

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
Trade	1	10	100			Open allocation (20)
Trade	2	10	100			Open allocation (20)
Price limit						
Session Change						
Equilibrium Price						
Order	3	11	100	100	Incoming (2)	Trade (3)
Order	4	11	100	100	Incoming (2)	Trade (3)
Trade	3	11	100			Open allocation (20)
Trade	4	11	100			Open allocation (20)
Price limit						
Commit						

5.15 Session change that results in bait generation

The following scenario changes session state in series FUT_NK225_1506 from HALT to OPEN.

The session state change results in a bait being generated in series FUT_NK225_1509.

Message	id	price	orderQuantity	leavesQuantity	orderStatus	changeReason	orderBookId
StartOfTransaction							
Session Change							Leg 2
Equilibrium Price							Leg 2
Session Change							Combo
Equilibrium Price							Combo
Order	1	10	100	100	Incoming (2)	New (6)	Leg 1
Order	1	10	100	100	Active (1)	Bait (11)	Leg 1
Commit							

5.16 Session change that results in bait removal

The following scenario changes session state in series FUT_NK225_1506 from OPEN to HALT.

The session state change results in a bait being removed in series FUT_NK225_1509.

Message	id	price	orderQuantity	leavesQuantity	orderStatus	changeReason	orderBookId
StartOfTransaction							
Session Change							Leg 2
Equilibrium Price							Leg 2
Order	1	10	100	100	2	Canceled (9)	Leg 1
Session Change							Combo

Message	id	price	orderQuantity	leavesQuantity	orderStatus	changeReason	orderBookId
Equilibrium Price							Combo
Equilibrium Price							Leg 2
Commit							

5.17 Matching combination vs leg

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource	orderBookId
StartOfTransaction							
Order	1	10	100	100	2	New (6)	Combo
Order	1	10	100	0	2	Trade (3)	Combo
Order	1	0	100	100	2	New (6)	Leg 1
Order	1	0	100	0	2	Trade (3)	Leg 1
Order	1	0	100	100	2	New (6)	Leg 2
Order	1	0	100	0	2	Trade (3)	Leg 2
Order	2	15010	100	0	2	Trade (3)	Leg 1
Order	3	15000	100	0	2	Trade (3)	Leg 2
Trade	1	15010	100			Auto (1)	Leg 1
Trade	2	15010	100			Auto (1)	Leg 1
Trade	3	15000	100			Auto (1)	Leg 2
Trade	1	15000	100			Auto (1)	Leg 2
Trade	1	10	100			ComboVsOutright (43)	Combo
Price Limit							Leg 1
Price Limit							Leg 2
Commit							

5.18 Session state order triggered

Message	id	price	orderQuantity	leavesQuantity	orderStatus	changeReason
StartOfTransaction						
Session Change						
Equilibrium Price						
Order	1	15000	100	100	Active (1)	SSO trigger (37)
Equilibrium Price						
Commit						

5.19 FaK-Order entered and partially traded

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction						
Order	1		150	150	Incoming (2)	New (6)
Order	1		150	50	Incoming (2)	Trade (3)
Order	2		100	0	Incoming (2)	Trade (3)
Trade	2		100			Auto (1)
Trade	1		100			Auto (1)
Price limit						
Commit						

5.20 Combo vs combo matching

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource	orderBookId
StartOfTransaction							
Order	1	0	100	100	Incoming (2)	New (6)	Combo
Order	1	0	100	0	Incoming (2)	Trade (3)	Combo
Order	2	0	100	0	Incoming (2)	Trade (3)	Combo
Trade	2	0	100			Combo (7)	Combo
Trade	1	0	100			Combo (7)	Combo
Trade	2	1500	100			Combo (7)	Leg 1
Trade	1	1500	100			Combo (7)	Leg 1
Trade	1	1500	100			Combo (7)	Leg 2
Trade	2	1500	100			Combo (7)	Leg 2
Commit							

5.21 FoK-Order entered and not traded

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction						
Order	1	1500	100	100	Incoming (2)	New (6)
Order	1	1500	100	100	Incoming (2)	New (6)
Commit						

5.22 Incoming order rejected

Message	id	price	orderQuantity/ tradedQuantity	leavesQuantity	orderStatus	changeReason/ dealSource
StartOfTransaction						
RejectedOrder	1	1500	100			
Commit						

5.23 Circuit Breaker triggered and reference price updated

Message	id	price	upperLimit/lowerLimit	referencePrice	Dynamic	priceLimits
StartOfTransaction						
SessionChange	1					
EqulibriumPrice	1	16170				
PriceLimit	1		14310/16650	15480	false	True
PriceLimit	1		15360/15600	15480	true	False
CircuitBreakerinfo						
Commit						