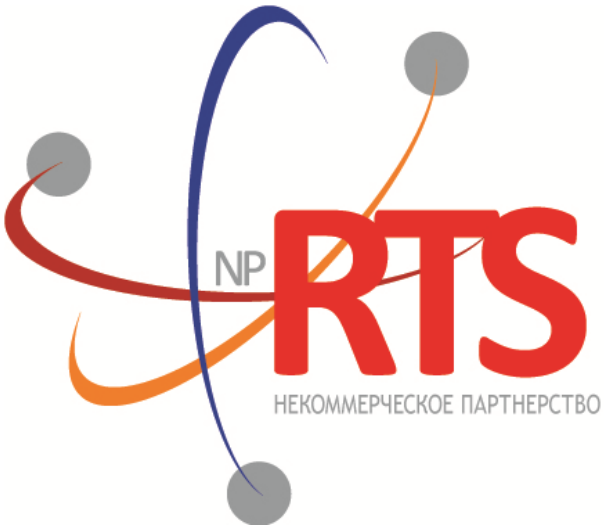

ADVANCED
TRADING & CLEARING
PLATFORM

Trade Gateway FIX (FIX 5.0 SP2)

System version 1.7

Interface version 22

Document version 1.9.0

27 February 2018

Revision history

Version 1.9.0 03 November 2017

1. Terminology changes.
2. Error codes added.

Version 1.8.4 3 April 2017

Values 0 and X of field TimeInForce corrected in messages NewOrderSingle and ExecutionReport.

Version 1.8.0 22 September 2016

1. New value X of field TimeInForce added to messages NewOrderSingle and ExecutionReport.
2. New values 1030, 1031, 1032, 1033 of field ExchangeSpecialInstructions added to messages NewOrderSingle and ExecutionReport.

Version 1.7.0 30 March 2016

1. New field OrdType added to message OrderCancelReject.
2. Functionality of automatic order canceling in case of disconnection is available in this version (please refer to section 3.4.8).

Version 1.6.0 24 December 2015

The order sent for execution at external price is type **OrdType=o** in system reports.

Version 1.5.0 31 August 2015

1. New field OrigClOrdID added to messages OrderCancelRequest, ExecutionReport, and OrderCancelReject.
2. Field ClOrdID changed objectives in messages OrderCancelRequest andOrderCancelReject.

Version 1.4.4 11 February 2015

1. Field BusinessRejectReason in message BusinessMessageReject corrected.
2. Interaction with trade gateway corrected at rejection of negotiated counterorder by counterparty (please refer to section 2.7.2).
3. Field structure in message DontKnowTrade changed.
4. Errors 1115, 1315, 1316, 8103, 8104, 8105, 8106, and 8201 added to error codes table.

Table of Contents

1. Trading system overview 5
1.1. Instruments of trading system 5
1.2. Trading modes 5
1.2.1. Main trades mode 5
1.2.2. Negotiated trades mode 6
1.2.3. Negotiated repo trades mode 6
1.2.4. Closing Auction in the Foreign Securities Market 6
2. Interaction with trading gateway 7
2.1. Reports `ExecutionReport[8]` 7
2.1.1. Distinguishing reports of different levels 7
2.2. Submission of orders 8
2.2.1. Order placement 8
2.3. Orders execution 9
2.4. Remainder cancellation after partial fill 9
2.5. Order remainder cancellation 9
2.6. Order mass cancellation 10
2.7. Negotiated order submission, execution and declining 10
2.7.1. Negotiated counterorder placement 11
2.7.2. Negotiated counterorder declining by counterparty 11
3. Protocol specifications 13
3.1. Datatypes 13
3.2. Format of message components 13
3.3. Liquidity pool identifiers 14
3.4. Session layer 14
3.4.1. Message header and trailer 15
3.4.2. Message sequence number `MsgSeqNum` 15
3.4.3. Session initialization 17
3.4.4. Session termination 17
3.4.5. Heartbeats 18
3.4.6. Message rejection 18
3.4.7. Disconnection 19
3.4.8. Automatic cancellation upon disconnection 19
3.5. Application level 20
3.5.1. Client requests 20
3.5.2. Trading system reports 25
3.5.3. Notification of negotiated counterorder placement 31
A. Error codes 32
B. Revision History 38

List of Tables

3. Format of component MDInc 13
4. Format of component Parties 14
5. Format of message header 15
6. Format of message trailer 15
7. Format of message ResendRequest[2] 16
8. Format of message SequenceReset[4] 16
9. Format of message Logon[A] 17
10. Format of message Logout[5] 18
11. Format of message HeartBeat[0] 18
12. Format of message TestRequest[1] 18
13. Format of message Reject[3] 19
15. Format of message NewOrderSingle[D] 21
16. Format of message OrderCancelRequest[F] 23
18. Format of message OrderMassCancelRequest[q] 24
19. Format of message DontKnowTrade[Q] 25
20. Format of message BusinessMessageReject[j] 25
21. Format of message ExecutionReport[8] 26
22. Format of message OrderCancelReject[9] 30
23. Format of message OrderMassCancelReport[r] 30
24. Format of message MarketDataIncrementalRefresh[X] 31

1. Trading system overview

The trading system is designed to allow users to perform operations on financial markets. The main functions include:

1. Acceptance of orders submitted to over-the-counter and exchange markets.
2. Routing and placing of orders in available liquidity pools.
3. Registration of trades and processing of information on trades at liquidity pools.
4. Transmission of anonymous market data, collected from all liquidity pools, and non-anonymous market data as well as additional and reference data.
5. Control of clearing member's risks on operations with instruments registered in the system.
6. Other functionality for access to trading.

1.1. Instruments of trading system

The Instruments are divided into **exchange** and **over-the-counter (OTC)**. OTC instruments have the following attributes:

- Field `section` in `Instruments` messages has value **OTC**.
- Field `over_the_counter` in `TradeModes` messages has value 1.
- Field `flags` has value 0x400000 (eOverTheCounter).

Table 1. Differences in the interpretation of messages fields

Instrument	Value of `order_id` field	Value of `deal_id` field
Exchange	Order ID	Trade ID
Over-the-counter	Tender ID	Contract ID

All instruments of trading system are available for trades.

1.2. Trading modes

1.2.1. Main trades mode

In the main trades mode anonymous orders are executed at liquidity pools.

The Main trades mode supports five order types. The order type is determined by the set of field values in the message.

1.2.1.1. Order types

1. Market order that will execute at the best available prices until it is fully filled; any remainder will be expired.
2. Day limit order that will execute at the specified or better price; the remainder, if any, is added to the order book and will be active till the end of the trading day.
3. Extended session limit order that will execute at the specified or better price; the remainder, if any, is added to the order book and will be active till the end of the extended trading session.
4. Fill or Kill (FOK) order that will execute immediately and completely, or canceled. This is an order with specified price and volume.
5. Immediate or Cancel (IOC) order that execute immediately, completely or partially, or canceled. This is an order with specified price and volume.

The set of order types available in the trading system may differ from the set of orders supported by a specific liquidity pool.

Iceberg order is not supported in the current system version.

1.2.1.2. Execution of orders

For a group of instruments listed on the trading system, the **Main pool** is determined among several liquidity pools by the highest liquidity level. The Main liquidity pool status may influence the choice of routing strategy: by default the volume that cannot be matched against active orders in the order book will be routed to that pool.

A client order, submitted to the trading system, can be executed at liquidity pools where the indicated instrument is admitted to trading. If there is only one liquidity pool matching this criterion, the entire orders volume is routed to that pool. If there are several liquidity pools like that, the order will be executed in accordance with the best execution principles.

In the course of routing, the incoming order is consecutively matched with counter orders at each price level until the order volume is filled. If all the available price levels were checked and the incoming order has not been filled completely, the remaining volume is routed to the Main liquidity pool. After the volumes to be routed are determined, they are sent to the liquidity pools.

Routing of client order depends on the order type.

A Fill Or Kill order can be filled at one liquidity pool only, where the order initiator can get the best average weighted price; in case of several equal prices the trading system give the priority to the pool providing a lower latency.

An incoming order of other types (limit, market, Immediate Or Cancel) can be routed to several liquidity pools. For each price level consecutively, starting from the best one for the order initiator, the volume to be executed is determined on each available pool. After the volumes to be routed are determined, they are sent to the appropriate price levels to the liquidity pools.

1.2.2. Negotiated trades mode

The Negotiated trades mode supports negotiated orders with fully matching parameters. Negotiated order is an order with an indication of price, volume, initiator and counterparty. The counterparty is notified that order is submitted on its clearing account (for detail on interaction with trading gateway refer to section 2).

1.2.3. Negotiated repo trades mode

Price of order for repo trades is indicated in annual interest rate. In additional price field the client can indicate the price of the first-leg instrument. If client did not indicate a price, the additional price will be settled or will be indicated by the liquidity pool.

Repo trading instrument has three legs (balance instruments):

1. Change in the obligation to deliver securities under the first part of repo trade.
2. Change in the obligation to deliver currency under the first part of repo trade.
3. Change in the obligation to deliver securities under the second part of repo trade.

Currency obligation under the second part of repo trade is changed using the price setting tool for repo trading instrument.

1.2.4. Closing Auction in the Foreign Securities Market

The Closing Auction in the Foreign Securities Market supports only market order with time in force - closing auction. Trades are executed at the official closing price of the instrument of the liquidity pool, on which the security was listed. Orders, leading to cross trade, will be automatically canceled by the liquidity pool.

Trading in the Closing Auction:

1. During the trading day, clients submit market orders in the trading system.
2. Submission of orders is stopped according to the approved schedule of trading and orders become unavailable to cancel.
3. Closing auction is held — counterorders, sorted by ascending of the time of submission, are matched together at instrument's closing price at Main liquidity pool.
4. Remainders of orders and unfilled orders are canceled.

2. Interaction with trading gateway

2.1. Reports `ExecutionReport[8]`

The trading system sends a report `ExecutionReport[8]` to the client at any change in status or volume of client's order:

1. acceptance of order by the trading system,
2. rejection of order by the trading system,
3. acceptance of order by liquidity pool,
4. rejection of order by liquidity pool,
5. trade,
6. partial or full execution of order's volume,
7. remainder cancellation after order execution,
8. partial or complete cancellation of order.

Each report `ExecutionReport` has two fields specifying the type of event that caused the report generation. They indicate the status of order and the type of report: `OrdStatus[39]` and `ExecType[150]`, respectively.

Table 2. Types of reports and statuses of order

Event	Status of order `OrdStatus[39]`	Report type `ExecType[150]`	Volume ratio
order successfully accepted by the trading system Order routing to the liquidity pool is successful	`0`	`0`	`CumQty=0` `LeavesQty=OrderQty`
Order rejected by the trading system Order rejected by liquidity pool	`8`	`8`	`CumQty=0` `LeavesQty=0`
Trade: order volume partially executed	`1`	`F`	`0<CumQty<OrderQty` `0<LeavesQty<OrderQty`
Trade: order volume fully executed	`2`	`F`	`CumQty=OrderQty` `LeavesQty=0`
Order cancellation	`4`	`4`	`CumQty<OrderQty` (may be equal 0) `LeavesQty=0`

Each `ExecutionReport` contains the client's identifier of order `ClOrdID[11]`. After the order is accepted by the trading system, all related reports will contain identifier `OrderID[37]`. The liquidity pool assigns its identifier to the accepted order and transmits it to the client in field `SecondaryOrderID[198]`.

2.1.1. Distinguishing reports of different levels

The `ExDestination[100]` value specified in the report unambiguously attribute the report to either order in the trading system (1001) or to the result of routing into a liquidity pool (1000).

The field `SecondaryOrderID[198]` is filled only in reports, sent after the order has been successfully routed to a liquidity pool. If the order has been rejected by a liquidity pool, this field is not set. In case of full rejection of an order on a liquidity pool's side, a client receives three `ExecutionReport` messages: about instruction placement, order remainder cancellation and about rejected routing at liquidity pool.

The client should employ one of the following mode:

1. Client order reports processing mode. In this mode only the order number on the trading system side is available. The order number on the liquidity pool's side is not available.

2. Order routing reports processing mode (recommended). This mode provides the whole information on orders execution in luquidity pools (the `ExecutionReport` has the `OrderID[37]` and `SecondaryOrderID[198]` numbers). However, discrimination between an order and a result of order routing by the `SecondaryOrderID[198]` field may be ambiguous.
3. Both levels' reports processing mode. When using this mode, the client should ignore a report duplicating an event of previous report. Due to asynchronous report generation, the client may first receive the value `LeavesQty[151]=0` in order cancel report and then a non-zero value of `LeavesQty[151]` in order add report, followed by `LeavesQty[151]=0` in remainder cancellation report.

In either mode, the client should process the Reject message.

2.2. Submission of orders

To submit an order, the client should send the `NewOrderSingle[D]` message (NOS) to the trading platform gateway. The client specifies the `ClOrdID[11]` identifier, unique for each login during the trading session.

After accepting the order, the trading platform will return `ExecutionReport[8]` (ER) to the client with `OrderID[37]`, and `OrdStatus[39]=0` and `ExecType[150]=0`. If the trading system rejects the order (due to invalid values or closed liquidity pool), no order identifier will be assigned and the client will receive `ExecutionReport[8]` with values `OrdStatus[39]=8` and `ExecType[150]=8`, while `OrdRejReason[103]` may explain reasons for rejection.

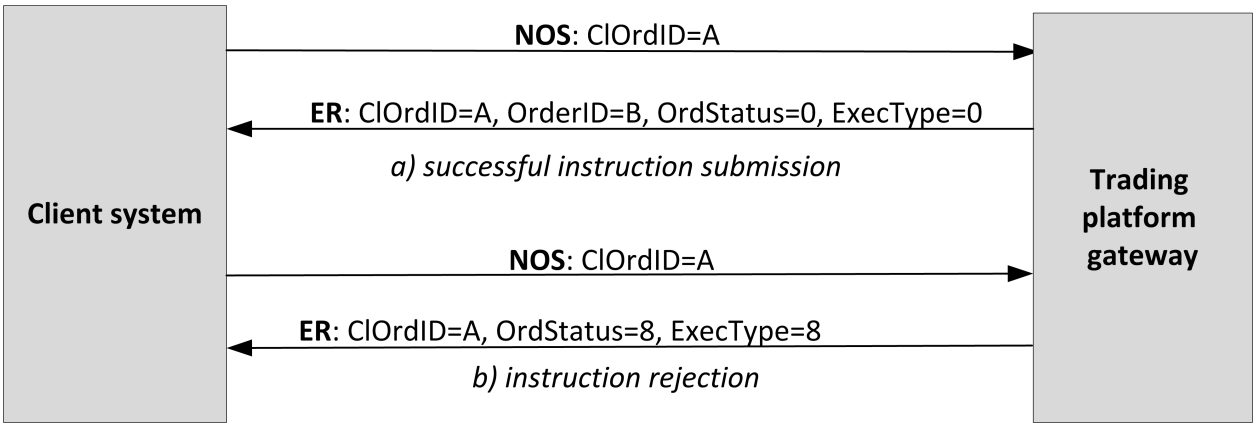

Figure 1. Adding orders

2.2.1. Order placement

To ensure best execution, the order volume is split according to the order books of the liquidity pools and splitted results are routed to liquidity pools. When a liquidity pool confirms order acceptance or rejection, the trading system sends corresponding report `ExecutionReport[8]` to the client containing order identifier `SecondaryOrderID` and values `OrdStatus[39]=0` and `ExecType[150]=0`.

If a liquidity pool rejects an order, the trading platform will return `ExecutionReport[8]` to the client (`OrdStatus[39]=8` and `ExecType[150]=8`) along with partial cancel report of the volume of the rejected order. In any cancel report, the value of the `OrderQty[38]` field will indicate rejected volume, not initial.

A Fill Or Kill order can be routed to one liquidity pool only.If the liquidity pool can fully fill the order, the client will receive all reports in the usual way. If the order cannot be executed, the liquidity pool will reject it and the client will be notified of, first, order placement, then order rejection, and, thirdly, order cancellation.

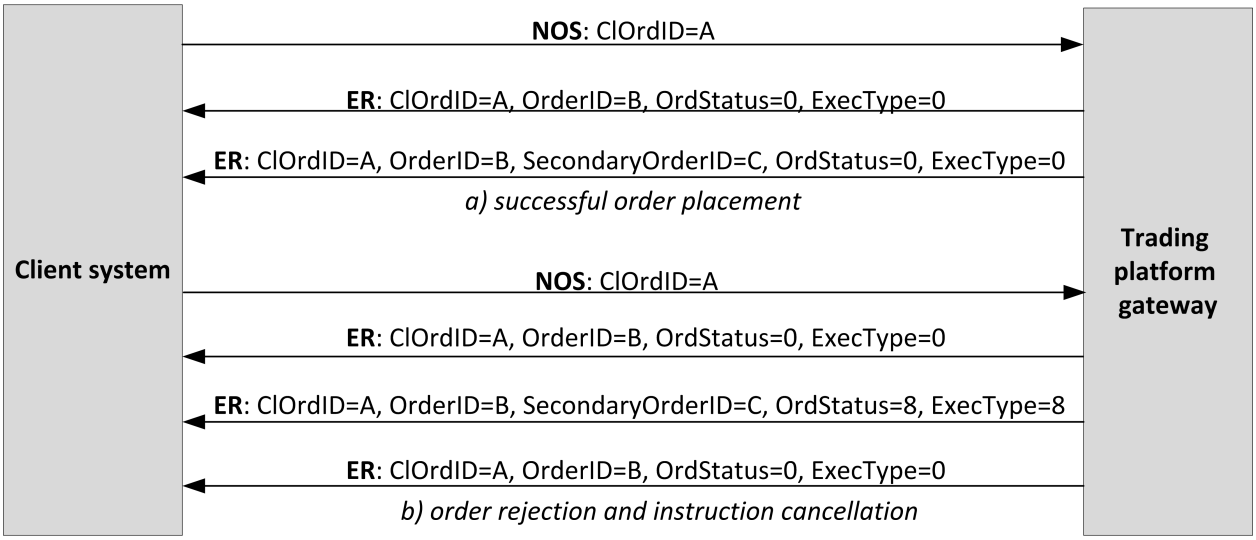

Figure 2. Order placement or rejection

2.3. Orders execution

After a liquidity pool accepts an order, the client will be sent reports (`ExecType[150]=F`) about deals with routed volumes and then on order change. All such reports include the trade ID `TrdMatchID[880]`.

The graph below shows the submission of an order and the following receipt of reports as seen by one side of the trade. The order is submitted and fully executed.

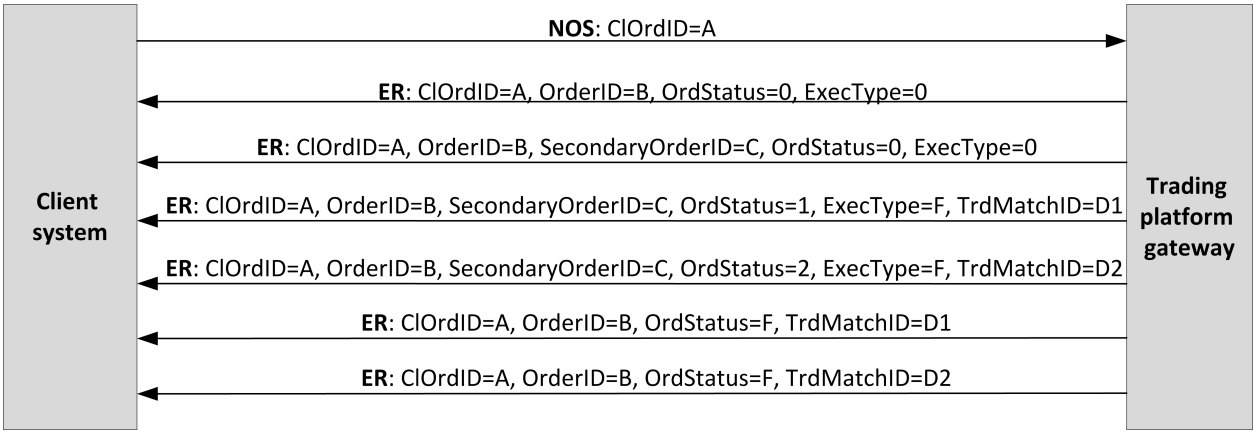

Figure 3. Submission of order and receipt of reports

2.4. Remainder cancellation after partial fill

In some instances, the liquidity pool will cancel an order remainder, e.g. the unfilled portion of a market or IOC order, or to prevent a cross trade. So after reports on order acceptance routing and trade reports, the client should also expect `ExecutionReport[8]` (`OrdStatus[39]=4` and `ExecType[150]=4`)—reports on partial or full cancellation of the order.

Moreover, to ensure best execution, the trading platform may cancel an order at a trading venue and place it to another. In this case, the client will receive a cancellation report and a new placement report.

2.5. Order remainder cancellation

After an order has been successfully routed, a single routed volume cannot be canceled. Only the whole order can be canceled.

The client can cancel the unfilled remainder of an order. The client shall send `OrderCancelRequest[F]` (OCRq) to the trading platform gateway and specify the identifier and certain parameters of the order.

After the order is successfully canceled, the client will receive `ExecutionReport`(`OrdStatus[39]=4` and `ExecType[150]=4`)—reports on routed volumes cancellation and then report on order cancellation.

If an order remainder cannot be canceled or the sender has no permissions, the request will be rejected with `OrderCancelReject[9]` (OCRj).

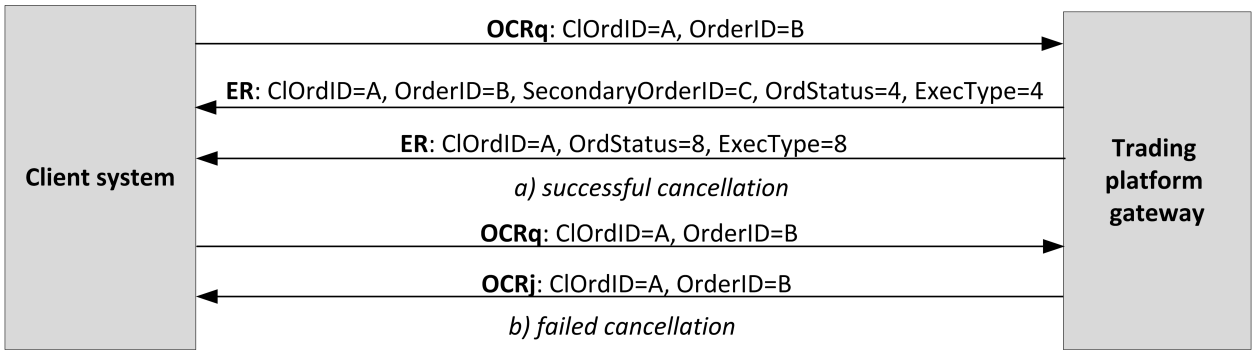

Figure 4. Order cancellation

2.6. Order mass cancellation

The client may request to cancel several orders, based on some criteria, for instance the orders referring to a certain instrument submitted from the particular login. The client shall send `OrderMassCancelRequest[q]` (MCRq) to the trading platform gateway and specify the cancellation mode and, if necessary, certain parameters of instructions.

The trading platform receives the request and selects orders to cancel by the specified criteria, and then generates cancellation request and routes them to liquidity pools. If the orders are canceled successfully, the client will receive reports on orders and instructions cancellation and the report on execution of order `OrderMassCancelReport[r]` (MCRt) specifying the number of canceled instructions. If no instruction to cancel is found, the gateway will only return `OrderMassCancelReport[r]`.

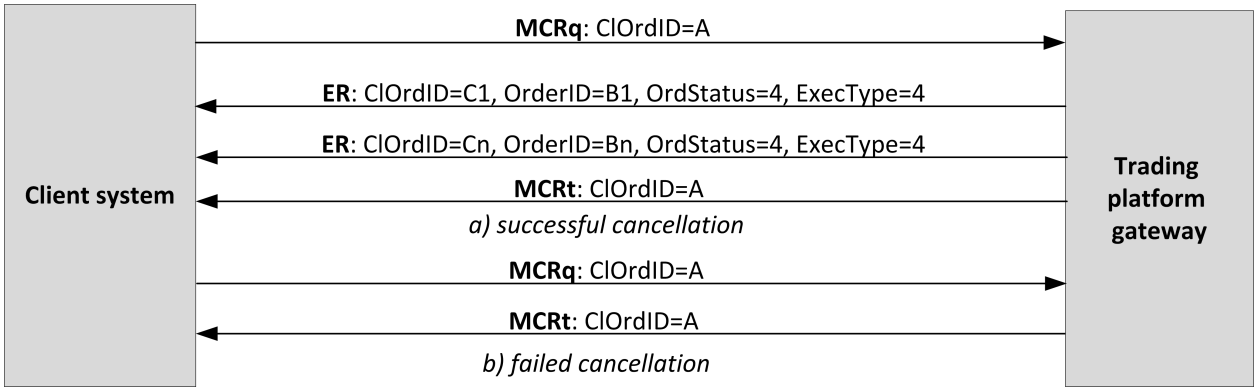

Figure 5. Orders mass cancellation

2.7. Negotiated order submission, execution and declining

To submit a negotiated order, the client should send `NewOrderSingle[D]` (NOS) to the trading platform gateway with unique `ClOrdID[11]` assigned.

After accepting the negotiated order, the trading platform will return `ExecutionReport[8]` (ER) to the client-sender with `OrderID[37]` and values `OrdStatus[39]=0` and `ExecType[150]=0`, and the client-receiver is sent `MarketDataIncrementalRefresh[X]` (MD) with identifier of update type `MDUpdateAction[279]=0`. If the trading system rejects the order (due to invalid values or closed market), no order identifier will be assigned and the client-sender will re-

ceive `ExecutionReport[8]` with values `OrdStatus[39]=8` and `ExecType[150]=8`, while field `OrdRejReason[103]` may explain reasons for rejection.

After the trading system and the liquidity pool accept the negotiated order, the client-sender may cancel it before the counterparty submits the counterorder. To cancel the negotiated order, the client should send `OrderCancelRequest[F]` (OCRq) to the gateway specifying the identifier and certain parameters of the order. If the negotiated order is successfully canceled, the sender will receive `ExecutionReport[8]` (`OrdStatus[39]=4` and `ExecType[150]=4`) and the counterparty will get `MarketDataIncrementalRefresh[X]` with `MDUpdateAction[279]=2`.

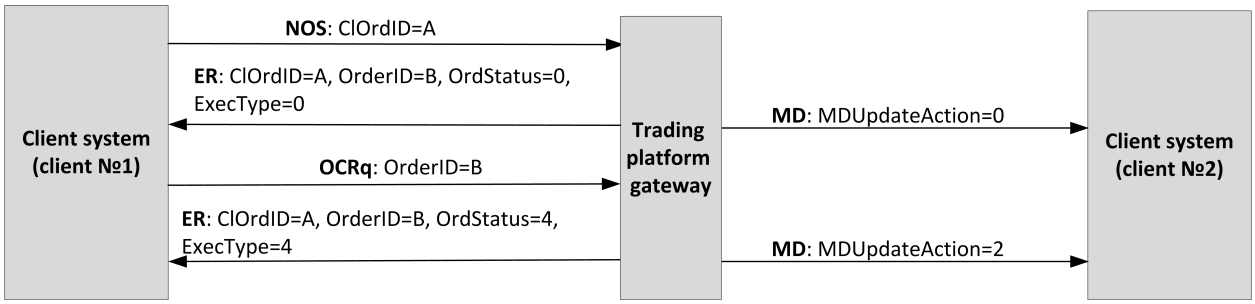

Figure 6. Negotiated order placement and cancellation

2.7.1. Negotiated counterorder placement

To take the offer, the counterparty shall send the counterorder with the same quantity of the instrument at the same price and the opposite side.

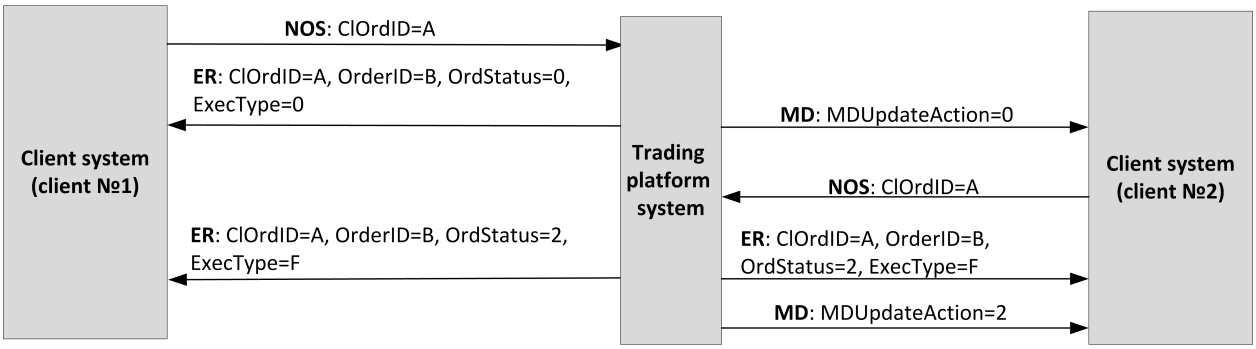

Figure 7. Successful submission of negotiated counterorder placement

In case of mismatch in price, amount, and instrument of the order, the counterorder will be placed as a new one and expect matching.

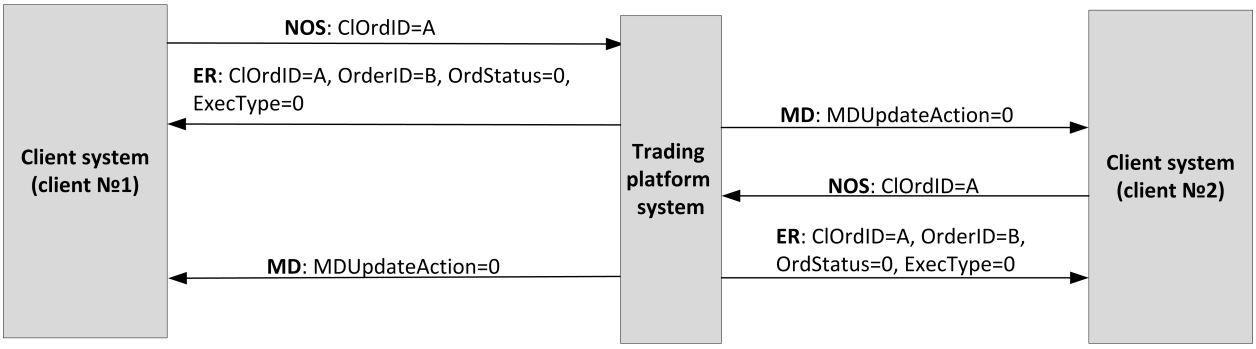

Figure 8. Failed submission of negotiated counterorder placement

2.7.2. Negotiated counterorder declining by counterparty

The counterparty can decline the negotiated order. The client should send `DontKnowTrade[Q]` (DKT) to the trading platform gateway and specify the identifier and certain parameters of the order.

After successful rejection, the client will receive the rejection response `DontKnowTrade[Q]` (it will differ from the request by `OrdStatus[39]=4`) and `MarketDataIncrementalRefresh[X]` (`MDUpdateAction[279]=2`), while the order initiator will be sent the cancellation report `ExecutionReport` (`OrdStatus[39]=2` and `ExecType[150]=F`).

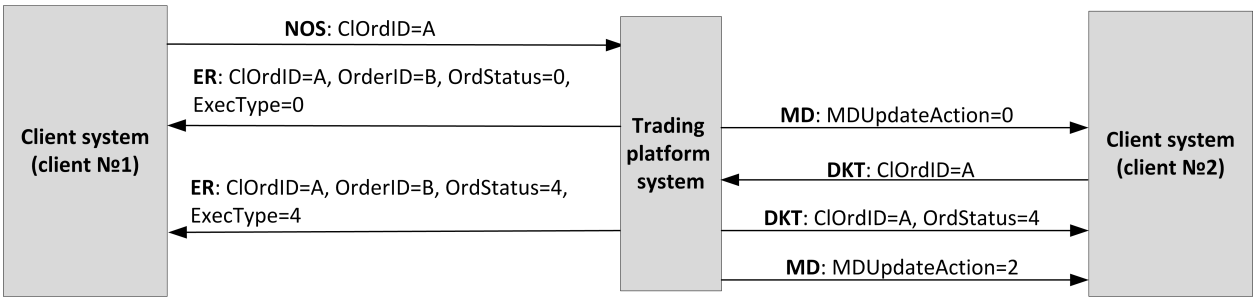

Figure 9. Negotiated counterorder rejection

3. Protocol specifications

3.1. Datatypes

This section contains tables describing the message formats

The message type defined in field `MsgType[35]` of the header is specified in brackets after the message name.

Fields:

- R [required];
- N [nonrequired];
- C [conditionally required].

Datatypes

Bool, logical field containing one of two values: `Y` (yes) and `N` (no).

Char, single-character datatype. Valid values are ASCII characters: letters, numbers, and punctuation marks. Null and Start of Heading characters are invalid.

Int - integer.

Length - positive integer to indicate length in bytes.

MultipleChar - string of single-character values separated by spaces. For example: `18=o z`.

NumInGroup - integer to indicate number of entries in a group.

Price - float to indicate price with point separator.

Qty - integer to indicate number of securities lots.

SeqNum - integer to indicate message sequence number.

String - string datatype. String can be in any encoding. Null and Start of Heading characters are invalid.

Timestamp - string datatype to indicate time and date of the World Time (UTC) within the accuracy of milliseconds in format `YYYYMMDD-HH:MM:SS.sss`.

3.2. Format of message components

Table 3. Format of component `MDInc`

Tag	Field	✓	Type	Values	Description
268	NoMDEntries	R	NumInGroup		Number of entries in repeated group
48	SecurityId	N	String		Numeric ID of trading instrument
22	SecurityIdSource	N	String		Liquidity pool ID of order placement. For values please refer to section 3.3
279	MdUpdateAction	R	Char	0 (new order); 2 (execution, cancellation or rejection of order)	Type of update
278	MdEntryId	R	String		Order ID assigned by trading system

Tag	Field	✓	Type	Values	Description
269	MdEntryType	R	Char	0 (buy); 1 (sell)	Side
270	MdEntryPx	N	Price		Price
271	MdEntrySize	N	Qty		Volume
272	MdEntryDate	R	UTCDateOnly		Date of update
273	MdEntryTime	R	UTCTimeOnly		Time of update

Table 4. Format of component `Parties`

Tag	Field	✓	Type	Values	Description
453	NoPartyIDs	R	NumInGroup		Number of entries in repeated group
448	PartyId	R	String		Subject ID corresponding to specified PartyRole
447	PartyIdSource	R	Char	D	Identifies class or source of the PartyID
452	PartyRole	R	Int	1 (trading member); 3 (client code); 13 (initiator of negotiated order); 17 (counterparty for negotiated order)	Role of the subject specified in PartyID

3.3. Liquidity pool identifiers

Liquidity pools' identifiers may be in fields `ExDestination[100]`, `LastMkt[30]` and `ExchangeSpecialInstructions[1139]`.

`0` (DEFAULT) — liquidity pool is defined by the trading system.

`1001` (TRADSYS) — all available liquidity pools.

`1000` — liquidity pool of Saint-Petersburg Exchange.

`1010` — liquidity pool of Moscow Exchange.

`1015` — execution at United States liquidity pools.

`1016` — market data from United States liquidity pools.

`1030` — liquidity pool of NYSE.

`1031` — liquidity pool of ARCA.

`1032` — liquidity pool of NASDAQ.

`1033` — liquidity pool of BATS.

3.4. Session layer

The session layer is mostly compliant with FIX Session Protocol 1.1.

A FIX session is established over TCP-connection between a client gateway and the trading platform gateway. Session participants are identified by fields `SenderCompID[49]` and `TargetCompID[56]`.

The ID of the trading platform gateway is ECN_EQR and that of a client is the user name.

3.4.1. Message header and trailer

Each message begins with the header and ends with the trailer.

The first three fields have fixed positions in the header, namely: `BeginString[8]=FIXT.1.1` always comes first, followed by field `BodyLength[9]` and then `MsgType[35]`. The value of `BodyLength[9]` is the message length in bytes, which is calculated starting from the tag following `BodyLength[9]` and ending with the separator before `CheckSum[10]`.

Table 5. Format of message header

Tag	Field	✓	Type	Values	Description
8	BeginString	R	String	FIXT.1.1	First field
9	BodyLength	R	Length		Length of message body
35	MessageType	R	String		Message type
49	SenderCompId	R	String		Sender ID
56	TargetCompId	R	String		Receiver ID
34	MsgSeqNum	R	SeqNum		Sequence number of message
43	PosDupFlag	N	Boolean		Resending message indicator
52	SendingTime	R	UTCTimestamp		Time of message transmission
122	OrigSendingTime	N	UTCTimestamp		Time of message resending when responding to ResendRequest[2]
369	LastMsgSeqNumProcessed	N	SeqNum		Sequence number of the last processed message. Specified by the trading system gateway only

The message trailer consists of `CheckSum[10]` including a three-byte simple check sum.

Table 6. Format of message trailer

Tag	Field	✓	Type	Features
10	CheckSum	R	String	Message check sum (3 bytes)

3.4.2. Message sequence number `MsgSeqNum`

All messages exchanged by the parties within a FIX session have a sequence number. The number is specified in the field `MsgSeqNum[34]` in the header of each message. The number of each subsequent message of a FIX session should be incremented, except for the cases of forced increase of the message number by request `SequenceReset[4]`.

As reference information for a client, the number of the last message processed by the trading system is indicated in the field `LastMsgSeqNumProcessed[369]`.

When receiving a message with the number higher than expected, a client should send `ResendRequest[2]`.

When the server receives a messages with the numbers lower than expected, a client will be sent `Logout[5]` with the value `SessionStatus[1409]=1` followed by TCP disconnection.

3.4.2.1. Request for resending

To request the messages previously sent by the server, a client can use `ResendRequest[2]`, in particular for the purpose of restoring missing messages. When receiving a message with the number higher than expected, a client should also send `ResendRequest[2]`.

a client may request resending of all messages, sent during the current and previous trading days. If a client has intentionally reset message numbering (`ResetSeqNumFlag[141]=Y` in the message `Logon[A]`), a request for resending messages, sent prior to the reset, is not possible.

The fields `BeginSeqNo[7]` and `EndSeqNo[16]` set the range of requested messages. If a client uses `BeginSeqNo[7]=0` and `EndSeqNo[16]=0`, the gateway will resend all messages starting from the lowest number available. If a client specifies 0 only for `EndSeqNo[16]`, the server will resend all messages of current trading session starting from `BeginSeqNo[7]`. All possible cases are as follows:

1. `BeginSeqNo=n, EndSeqNo=m` (request for messages from *n* to *m*),
2. `BeginSeqNo=0, EndSeqNo=n` (request for messages from the lowest number available to *n*),
3. `BeginSeqNo=n, EndSeqNo=0` (request for messages from *n* to the highest number available),
4. `BeginSeqNo=0, EndSeqNo=0` (request for all available messages).

Number range for requested messages is not limitless (for more details please refer to *Network Connectivity*). When a client requires more messages, a client should send several consecutive requests. If the gateway has not finished sending messages on previous request, new requests for messages will be rejected.

Table 7. Format of message `ResendRequest[2]`

Tag	Field	✓	Type	Values	Description
7	BeginSeqNo	R	SeqNum		Number of first requested message
16	EndSeqNo	R	SeqNum		Number of last requested message

In response to `ResendRequest[2]`, the gateway will return the requested messages or will modify `MsgSeqNum[34]` by the message `SequenceReset[4]`. The value `PossDupFlag[43]=Y` is a flag of resent message.

After receiving `ResendRequest[2]`, the server will resend messages of the application layer only and will never resend session messages. Therefore, in response to message resend request a client should expect, among others, `SequenceReset[4]` with `GapFillFlag[123]=Y` and the number of the next expected message in `NewSeqNo[36]`.

If a client wants to increase the message number expected from the server, a client should sent `SequenceReset[4]` with `GapFillFlag[123]=N` and the new expected number in the field `NewSeqNo[36]`.

During resending, the server may also transmit new trading messages, so a client should also expect messages with a number exceeding the requested range. To ensure quick message processing, a client is not recommended to ignore such messages with larger numbers.

Table 8. Format of message `SequenceReset[4]`

Tag	Field	✓	Type	Values	Description
36	NewSeqNo	R	SeqNum		New sequence number
123	GapFillFlag	N	Boolean	N (mode Reset ignoring field MsgSeqNum; specified by the client); Y (mode GapFill using field MsgSeqNum; specified by the server)	Indicator of gap fill

3.4.2.2. Resetting message sequence numbers

The value `ResetSeqNumFlag[141]=Y` in the `Logon[A]` message allows to reset sequence numbers. This functionality may be useful to avoid procedures for requesting and restoring missing or allegedly missing messages. It is not recommended to use this feature during the trading session when trading messages have already been sent, because after the reset these messages will not be available for resend request.

In response to a client `Logon[A]` with `ResetSeqNumFlag[141]=Y` the trading system will send `Logon[A]` with `ResetSeqNumFlag[141]=Y`, `MsgSeqNum[34]=1`, and `NextExpectedMsgSeqNum[789]=2`. Thus, each party will have the next message number equal to `2`.

3.4.3. Session initialization

The `Logon[A]` initiates a FIX session or confirms its start. After establishing a TCP connection, the session initiator (client) sends this message and expects `Logon[A]` in response.

A receipt of a correct `Logon[A]` shall always result in sending response message `Logon[A]`, even if `MsgSeqNum[34]` is higher than expected. Any error in `Logon[A]` shall cause a disconnection, and the number of the next expected message will not be incremented.

Table 9. Format of message `Logon[A]`

Tag	Field	✓	Type	Values	Description
98	EncryptMethod	R	Int	0 (Encryption not supported)	Method of encryption
108	HeartBtInt	R	Int		Timeout. Value in seconds. Recommended value: from 20 to 30
95	RawDataLength	C	Length	1	The field must be present if there is RawData[96]
96	RawData	N	data	0 (do not activate automatic); 1 (activate automatic)	Automatic cancellation of all orders submitted by this login at disconnection
141	ResetSeqNumFlag	N	Boolean		Reset of sequence numbers
789	NextExpectedMsgSeqNum	N	SeqNum		Number of next messages to be sent by the client. To be filled by the server
554	Password	N	String		Login password
1137	DefaultApplVerId	R	String	9 (FIX50SP2)	Protocol version

3.4.4. Session termination

`Logout[5]` initiates or confirms the session termination and shall be sent after a long-term absence of messages (please refer to section 3.4.5) or after receiving a message with number lower than expected.

The reason for rejection is specified in the tag `SessionStatus[1409]`. The field `Text[58]` may contain report on the session termination reasons.

Table 10. Format of message `Logout[5]`

Tag	Field	✓	Type	Values	Description
1409	SessionStatus	N	Int	5 (invalid login or password); 5000 (violation of message exchange protocol); 5002 (client not active); 5003 (server stopped); 5200 (login is already in active session)	Numeric code of the reason. To be filled by the server only
58	Text	N	String		Report on session termination reason

3.4.5. Heartbeats

Client and server exchange messages `Heartbeat[0]` to monitor the connection status. A heartbeat is to be sent by a party if it passed no messages (of the session or application layer) within the heartbeat interval. A client specifies the timeout value `HeartBtInt[108]` in the message `Logon[A]`; the recommended value is from 20 to 30 seconds.

After the absence of messages during an interval exceeding `HeartBtInt[108]`, a party should send `TestRequest[1]` with the `TestReqID[112]` identifier. In answer the counterparty should send `Heartbeat[0]` containing the same identifier. If no response within the heartbeat interval, the server disconnects after sending message `Logout[5]` to a client. A client is expected to act the same.

If a client prefers not to send or receive heartbeats during this FIX session, 0 should be specified in `HeartBtInt[108]`.

Table 11. Format of message `HeartBeat[0]`

Tag	Field	✓	Type	Values	Description
112	TestReqId	C	String		Request ID of TestRequest, to which this message is a response

Table 12. Format of message `TestRequest[1]`

Tag	Field	✓	Type	Values	Description
112	TestReqId	R	String		Request ID Maximum length is 32 characters. Valid characters are letters and numbers

3.4.6. Message rejection

The message `Reject[3]` is sent in response to any invalid message (incorrectly generated or transmitted) from the other party. The reasons for rejection may be the absence of required fields, invalid message type or length, and invalid data type, etc. All session level messages with invalid value of any field are also rejected by the message `Reject`.

The server specifies the rejected message number in the field `RefSeqNum[45]`. The value `RefSeqNum[45]=0` means that the field `MsgSeqNum[34]` is missing in the rejected message. If the server detects an invalid value, the tag will be indicated in `RefTagID[371]`. The field `SessionRejectReason[373]` may contain the rejection reason code and `Text[58]` may have a textual description of error.

Table 13. Format of message `Reject[3]`

Tag	Field	✓	Type	Values	Description
45	RefSeqNum	R	SeqNum		Number of rejected message
371	RefTagId	N	Int		Tag which caused the error
372	RefMsgType	N	String		Type of rejected message
373	SessionRejectReason	N	Int	0 (invalid tag number); 1 (required tag missing); 2 (invalid tag in the message); 4 (tag with no value); 5 (invalid value); 6 (invalid datatype); 11 (incorrect message type); 13 (tag repeated in message); 14 (tag CheckSum[10] misplaced); 15 (tag from the group misplaced); 16 (invalid number of group entries)	Reason for rejection
58	Text	N	String		Error report

3.4.7. Disconnection

The TCP connection will be dropped if the server receives a message with an error in one of the first three fields (`BeginString[8]`, `BodyLength[9]`, and `MsgType[35]`) or the `Logon[A]` message of invalid format or containing invalid values.

3.4.8. Automatic cancellation upon disconnection

All active orders submitted by the login can be canceled at FIX session termination. The option should be enabled during the session initialization by the values `RawDataLength[95]=1` and `RawData[96]=1` in the `Logon[A]` message. By default, the automatic cancellation is disabled.

Orders submitted by the user login (i.e. via a client gateway) will be canceled at FIX session termination if

1. TCP connection is dropped by a client gateway,
2. there is no answer received to `TestRequest[0]` during the heartbeat interval,
3. the `Logout[5]` is received.

If the automatic cancellation was enabled, all client orders, including those submitted in previous sessions, will be canceled. The cancellation will be reported to the logins which have access to non-anonymous market data. The `ExecutionReport[8]` will have the indication `Text[58]=Cancel on disconnect`.

Otherwise, a client may enable the automatic cancellation for a single order by specifying the value `ExecInst[18]=o` in the `NewOrderSingle[D]`. This order will be canceled upon disconnection even if the option was not enabled at the session initialization.

3.5. Application level

3.5.1. Client requests

3.5.1.1. Submission of orders

To submit a new order to the trading system, the client should send the message `NewOrderSingle[D]` specifying the following:

- clearing account in the field `Account[1]`,
- trading instrument in the field `SecurityID[48]` (please refer to the *Instrument reference data*),
- routing algorithm in the fields `ExDestination[100]` and `ExchangeSpecialorders[1139]` (for more information please refer to section 3.5.1.1.1),
- side of order in the field `Side[54]`,
- type of order in the field `OrdType[40]`,
- period, during which the order remains in effect, in the field `TimeInForce[59]`,
- quantity of instrument lots in the field `OrderQty[38]`.

For all order types, except for the market one (`OrdType[40]=1`), price must be set in `Price[44]`.

Table 14. Concordance between the order type and field values in messages

Type of orders	Required fields
Market	`OrdType[40]=1` `TimeInForce[59]=3`
Market order at closing auction	`OrdType[40]=1` `TimeInForce[59]=7`

Limit orders at closing auction	`OrdType[40]=2`
	`TimeInForce[59]=7`
	`Price[44]`

Day active limit	`OrdType[40]=2`
	`TimeInForce[59]=0`
	`Price[44]`

Limit order in extended trading session	`OrdType[40]=2`
	`TimeInForce[59]=X`
	`Price[44]`

Fill Or Kill (FOK)	`OrdType[40]=2`
	`TimeInForce[59]=4`
	`Price[44]`

Immediate Or Cancel (IOC)	`OrdType[40]=2`
	`TimeInForce[59]=3`
	`Price[44]`

Iceberg	`OrdType[40]=2`
	`TimeInForce[59]=0`
	`0<DisplayQty[1138]<OrderQty[38]`
	`DisplayWhen[1083]=2`
	`DisplayMethod[1084]=1`
	`Price[44]`

Type of orders	Required fields
Negotiated	`OrdType[40]=n` `TimeInForce[59]=0 Price[44]`

The Closing Auction in the Foreign Securities Market only allows market (`OrdType[40]=1`) and the Closing Auction in the Russian Securities Market allows market (`OrdType[40]=1`) and limit (`OrdType[40]=2`) orders.

The trading member and the client code, on whose behalf the order is issued, should be specified in the field `PartyID[448]` in the `Parties` group; the first group entry contains the trading member with `PartyRole[452]=1` and the second entry defines the client code with `PartyRole[452]=3`. In a negotiated order, the `Parties` group should include two more entries - the order initiator and recipient.

The client should set the client order identifier in the field `ClOrdID[11]`.The trading system requires this identifier to be unique during the trading session for each client gateway. It is not recommended to reuse `ClOrdID[11]` of rejected orders.

Special identifier `RefOrderID[1080]` must be assigned for a negotiated order. The counterorder must contain the same ID, otherwise the orders will not match.

After processing a client order, the trading system will either reject it with the message `BusinessMessageReject[j]` or confirm with the message `ExecutionReport[8]` with status `ExecType[150]=0` and `OrdStatus[39]=0`.

The client can provide an order with a comment in the field `Text[58]` (23 bytes in UTF-8).

At the end of the trading session or extended trading session all active orders (`TimeInForce[59]=0` or `TimeInForce[59]=X`) will be canceled and the client will receive `ExecutionReport[8]` with the indicator `ExecRestatementReason[378]=EXPIRED`.

Table 15. Format of message `NewOrderSingle[D]`

Tag	Field	✓	Type	Values	Description
11	ClOrdId	R	String		Client order ID. Maximum length is 20 characters. Valid characters are Latin letters and numbers
60	TransactTime	R	UTCTimestamp		Time of order submission by user
100	ExDestination	R	Exchange		Liquidity pool ID where the order is sent to. For values please refer to section 3.5.1.1.1
48	SecurityId	R	String		Numeric ID of trading instrument
9303	RoutingInstruction	N	String		Routing algorithm
54	Side	R	Char	1 (buy); 2 (sell)	Side of order
40	OrdType	R	Char	1 (market); 2 (limit); n (negotiated)	Type of order

Tag	Field	✓	Type	Values	Description
59	TimeInForce	R	Char	0 (during the trading session); 2 (opening auction); 3 (immediate or cancel, IOC); 4 (fill or kill, FOK); 7 (closing auction); X (during the extended trading session)	Period the order remains in effect
44	Price	C	Price		Price. For repo trading: annual interest yield, the value to be indicated in percentage
38	OrderQty	R	Qty		Volume of order in lots
1138	DisplayQty	N	Qty		Disclosed quantity of order. Required for icebergs: • `0<DisplayQty<OrderQty` (iceberg); • `DisplayQty`not defined (disclosed orders)
1084	DisplayMethod	N	Char	1 (iceberg)	Required for icebergs
1	Account	R	String		Clearing account of the client submitting order
	Component Parties	R			
58	Text	N	String		Comment. Maximum length is 23 characters
1139	ExchangeSpecialInstructions	N	String		The main liquidity pool. For values please refer to section 3.3
1080	RefOrderId	N	String		Identifier for matching negotiated orders
10104	Price1	N	Price		Additional price. For a repo the trade price can be specified

3.5.1.1.1. Order routing options

The client sets the order routing in the two fields:

1. `ExDestination` is the identifier of a liquidity pool, where the client order is sent to; for values please refer to section 3.3;
2. `ExchangeSpecialorders` is the Main liquidity pool, where the order remainder, if any, will be sent to; for values please refer to section 3.3.

3.5.1.2. Cancellation of order remainder

After the order has been routed to the liquidity pools, the client can cancel the order quantity that is still not filled. The client should send `OrderCancelRequest[F]` to the trading system. The order being canceled is identified in either of the two fields: `ClOrdID[11]` or `OrderID[37]` (allowed only for the login submitted the order). While canceling an order submitted by another login, the user should specify `OrderID[37]`.

After processing the request, the trading system either rejects it with the message `BusinessMessageReject[j]` or confirms the cancellation with `ExecutionReport[8]`.

Table 16. Format of message `OrderCancelRequest[F]`

Tag	Field	✓	Type	Values	Description
41	OrigClOrdId	C	String		Client ID of order to cancel. Maximum length is 20 characters. Valid characters are Latin letters and numbers
11	ClOrdId	R	String		Client ID of the command to cancel order. Maximum length is 20 characters. Valid characters are Latin letters and numbers
37	OrderId	C	String		Order ID assigned by the trading system
60	TransactTime	R	UTCTimestamp		Date and time of request generation
100	ExDestination	R	Exchange		Liquidity pool ID specified in the order. For values please refer to section 3.5.1.1.1
48	SecurityId	R	String		Numeric ID of the trading instrument
54	Side	R	Char	1 (buy); 2 (sell)	Side of order
1	Account	R	String		Clearing account
	Component Parties	R			

3.5.1.3. Order mass cancellation

Mass cancellation of orders is available in several modes, that should be set in `OrderMassCancelRequest[q]` by the value of `MassCancelRequestType[530]`.

Table 17. Order mass cancellation modes

Mode	Required fields
Cancellation of the orders submitted by the requesting login	`MassCancelRequestType[530]=7`
Cancellation of all orders of the instrument submitted by the requesting login	`MassCancelRequestType[530]=1` `SecurityID[48]`

Mode	Required fields

Cancellation of all orders of the specified instrument and the clearing account	`MassCancelRequestType[530]=1`
	`SecurityID[48]`
	`Account[1]`

Cancellation of all orders of the specified instrument and the client code	`MassCancelRequestType[530]=1`
	`SecurityID[48]`
	group `Parties`

When setting the mode for cancellation of orders submitted by requesting login (`MassCancelRequestType[530]=7`), the client should not fill the fields `SecurityID[48]` and `ExDestination[100]`.

After processing the request, the trading system confirms cancellation of each cancelled order with a separate report `ExecutionReport[8]` with statuses `ExecType[150]=4` and `OrdStatus[39]=4`, and then sends `OrderMassCancelReport[r]`.

Table 18. Format of message `OrderMassCancelRequest[q]`

Tag	Field	✓	Type	Values	Description
11	ClOrdId	R	String		Client ID of the command to cancel order. Maximum length is 20 characters. Valid characters are Latin letters and numbers
530	MassCancelRequestType	R	Char	1 (for the instrument); 7 (all instructions)	Type of cancellation
60	TransactTime	R	UTCTimestamp		Date and time of request generation
100	ExDestination	N	Exchange		Liquidity pool ID specified in the order. For values please refer to section 3.5.1.1.1
48	SecurityId	C	String		Numeric ID of trading instrument. Required when `MassCancelRequestType[530]=1`
1	Account	N	String		Clearing account
	Component Parties	N			

3.5.1.4. Negotiated counterorder declining

The client can decline a negotiated order. The client should send `DontKnowTrade[Q]` to the trading platform gateway with the order identifier `OrderID[11]`, the counterparties in the `Parties` group, and, if needed, the match identifier `RefOrderID[1080]`.

After processing the request, the trading platform either rejects it with the message `BusinessMessageReject[j]` or confirms the cancellation with the `DontKnowTrade[Q]` report, which differs from the request by the indicator `OrdStatus[39]=4`, and the notification `MarketDataIncrementalRefresh[X]`.

Table 19. Format of message `DontKnowTrade[Q]`

Tag	Field	✓	Type	Values	Description
37	OrderId	R	String		Order ID assigned by the trading system
48	SecurityId	R	String		Numeric ID of the trading instrument
54	Side	R	Char	1 (buy); 2 (sell)	Side
40	OrdType	R	Char	1 (market); 2 (limit); n (negotiated); o (expanded liquidity pool)	Type of order
	Component Parties	R			
1080	RefOrderId	N	String		Identifier for matching negotiated orders
39	OrdStatus	R	Char	4 (canceled); 8 (rejected)	Status of order

3.5.2. Trading system reports

3.5.2.1. `BusinessMessageReject[j]`

A client order with an invalid combination of conditionally required fields, including the indication of order type, will be rejected with `BusinessMessageReject[j]`.

Table 20. Format of message `BusinessMessageReject[j]`

Tag	Field	✓	Type	Values	Description
45	RefSeqNum	R	SeqNum		Number of rejected message
372	RefMsgType	R	String		Type of rejected message
380	BusinessRejectReason	R	Int	5 (conditionally required field missing); 100 (undefined tag); 6000 (both account and parties filled)	Error code
371	RefTagId	N	Int		Tag causing the error
58	Text	N	String		Error text

3.5.2.2. `ExecutionReport[8]`

`ExecutionReport[8]` will be sent to the client in case of rejection, cancellation, modification, and expiration of a client order, as well as when routing an order to the liquidity pools (for the report types please refer to section 2.1).

The cancellation report (`OrdStatus[39]=4` and `ExecType[150]=4`) usually contains the cancellation reason `ExecRestatementReason[378]`.

The trade report (`ExecType[150]=F`) includes the identifier of the trade `TrdMatchID[880]`, assigned by a liquidity pool, and specifies the liquidity pool of the trade in `LastMkt[30]`.

When orders are rejected, the report will contain rejection reasons in the field `OrdRejReason[103]`.

Each report contains the client's identifier of order `ClOrdID[11]`. The event causing the report can be defined by the fields `OrdStatus[39]` and `ExecType[150]`. The report containing the order identifier `SecondaryOrderID[198]` refers to the results of routing the order in liquidity pools, not the initial client order.

Table 21. Format of message `ExecutionReport[8]`

Tag	Field	✓	Type	Values	Description
	[gate_header]	R			Standard header
1	Account	R	String		Clearing account
100	ExDestination	R	Exchange		Liquidity pool ID specified in the order. For values please refer to section 3.5.1.1.1
10104	Price1	N	Price		Price of the first part of repo (to be filled only for repo orders)
103	OrdRejReason	C	Int	1	Reasons for order rejection. Indicated when `ExecType(150)=8`. For values please refer to Table 25
1080	RefOrderId	N	String		Identifier for matching negotiated orders
1083	DisplayWhen	N	Char	2	Required for iceberg
1084	DisplayMethod	N	Char	1 (iceberg)	Required for iceberg
11	ClOrdId	R	String		Client ID of the command
1138	DisplayQty	N	Qty		Disclosed (visible) part of the order amount. Used for icebergs: • `0<DisplayQty<OrderQty` (iceberg); • `DisplayQty` not defined (visible order)
1139	ExchangeSpecialInstructions	C	String		Основной пул ликвидности. For values please refer to section 3.3. Filled when `ExecType[150]=0 or F`, if it was indicated by the user at submission
14	CumQty	N	Qty		Executed quantity of order

Tag	Field	✓	Type	Values	Description
150	ExecType	R	Char	0 (adding); 4 (cancellation); 8 (rejection of invalid order); F (trade)	Type of report
151	LeavesQty	R	Qty		Non-executed quantity of order
18	ExecInst	N	MultipleCharValue		Command for order handling
198	SecondaryOrderId	N	String		Order ID at liquidity pool. If filled, the report refers to the amount, routed to a liquidity pool. Otherwise, the report refers to the client order
30	LastMkt	N	Exchange		Exchange of last trade. For values please refer to section 3.3
31	LastPx	R	Price		Price of last trade. Filled when ExecType[150]=F
32	LastQty	R	Qty		Quantity of last trade. Filled when ExecType[150]=F
37	OrderId	N	String		Order ID assigned by the trading system

Tag	Field	✓	Type	Values	Description
378	ExecRestatementReason	C	Int	100 (canceled on client's `OrderCancelRequest [F]`); 101 (canceled on client's `OrderMassCancelRequest [q]`); 102 (canceled on broker's `OrderCancelRequest [F]`); 104 (canceled on broker's `OrderMassCancelRequest [q]`); 105 (canceled on disconnection); 106 (canceled on expiration); 108 (canceled by trading system operator); 109 (IoC remainder cancel); 110 (canceled to prevent a cross trade); 111 (canceled to prevent a crossed book); 112 (canceled on counterparty's `DontKnowTrade [Q]`); 114 (negotiated trade); 115 (canceled on rejection by liquidity pool); 116 (canceled on expiration of order at liquidity pool)	The reason for cancellation of order. Indicated when `ExecType(150)=4`
38	OrderQty	R	Qty		Quantity of order in lots
388	DiscretionInst	N	Char	0	Required for a discretionary order
39	OrdStatus	R	Char	0 (active); 1 (partially executed); 2 (executed); 4 (canceled); 8 (rejected)	Status of order

Tag	Field	✓	Type	Values	Description
40	OrdType	C	Char	1 (market); 2 (limit); n (negotiated); o (expanded liquidity pool)	Type of order. Not present when `ExecType[150]=4`
41	OrigClOrdId	N	String		Client ID of order to cancel
44	Price	C	Price		Lot price
453	Component Parties	R			
48	SecurityId	R	String		Numeric ID of the trading instrument
529	OrderRestrictions	N	MultipleCharValue	5 (acting as market maker)	Restrictions associated with order
54	Side	R	Char	1 (buy); 2 (sell)	Side
58	Text	N	String		Comment by client
59	TimeInForce	C	Char	0 (during the trading session); 2 (opening auction); 3 (immediate or cancel, IOC); 4 (fill or kill, FOK); 7 (closing auction); X (during the extended trading session)	Period the order remains in effect. Not present when `ExecType[150]=4`
60	TransactTime	R	UTCTimestamp		Date and time of report generation
841	DiscretionMoveType	N	Int	0	Required for discretionary order
843	DiscretionLimitType	N	Int	2	Required for discretionary order
880	TrdMatchId	R	String		Trade ID assigned by liquidity pool. Filled when ExecType[150]=F
9303	RoutingInstruction	N	String		Routing algorithm ID

3.5.2.3. Report on rejection of order cancellation request

If the requested order cannot be canceled or the cancellation request `OrderCancelRequest[F]` contains invalid parameters, the trading system will reject the request and send `OrderCancelReject[9]` to the client.

Table 22. Format of message `OrderCancelReject[9]`

Tag	Field	✓	Type	Values	Description
37	OrderId	R	String		Order ID assigned by the trading system
41	OrigClOrdId	N	String		Client ID of order to cancel
11	ClOrdId	R	String		Client ID of the command to cancel order
60	TransactTime	R	UTCTimestamp		Date and time of report generation
102	CxlRejReason	R	Int	1	The reason for rejection of cancellation request. For values please refer to Table 25
40	OrdType	R	Char	1 (market); 2 (limit); n (negotiated); o (expanded liquidity pool)	Type of order. Not present when `ExecType[150]=4`
39	OrdStatus	R	Char	8 (rejected)	Request status
100	ExDestination	R	Exchange		Liquidity pool ID specified in the order. For values please refer to section 3.5.1.1.1
48	SecurityId	R	String		Numeric ID of the trading instrument
54	Side	R	Char	1 (buy); 2 (sell)	Side
1	Account	R	String		Trading and clearing account
	Component Parties	R			
30	LastMkt	C	Exchange		Exchange of last trade. For values please refer to section 3.3

3.5.2.4. Report on mass cancellation of orders

In response to `OrderMassCancelRequest[q]` the server returns the report on massive cancellation `OrderMassCancelReport[r]`. If some orders were canceled on request, the report will be preceded by individual reports on cancellation of each order `ExecutionReport[8]` with `ExecType[150]=4` and `OrdStatus[39]=4`.

Table 23. Format of message `OrderMassCancelReport[r]`

Tag	Field	✓	Type	Values	Description
11	ClOrdId	R	String		Client ID of the command to cancel order
1369	MassActionReportId	R	String		Operation number

Tag	Field	✓	Type	Values	Description
530	MassCancelRequestType	R	Char	1 (for the instrument); 7 (all orders)	Type of cancellation
531	MassCancelResponse	R	Char	0 (request rejected); 1 (orders of the specified instrument canceled); 7 (all orders canceled)	Status of command processing
533	TotalAffectedOrders	N	Int		Number of canceled orders
60	TransactTime	R	UTCTimestamp		Date and time of report generation
100	ExDestination	N	Exchange		Liquidity pool ID specified in the order. For values please refer to section 3.5.1.1.1
48	SecurityId	N	String		Numeric ID of the trading instrument
1	Account	N	String		Trading and clearing account
	Component Parties	N			

3.5.3. Notification of negotiated counterorder placement

At submission, execution, or cancellation of a negotiated order directed to the client, the gateway will send the notification `MarketDataIncrementalRefresh[X]` containing one entry of the group `MDEntry` specifying the order parameters.

The `MDUpdateAction` value indicates the event: `1` at submission of a new negotiated order and `2` at execution or cancellation of negotiated order.

Table 24. Format of message `MarketDataIncrementalRefresh[X]`

Tag	Field	✓	Type	Values	Description
	Component MDInc	R			
	Component Parties	R			

Appendix A. Error codes

Table 25. Error codes list

Code	Description
0	Ok
5	Missed tag.
100	Filled excess tag.
999	Internal error.
1000	Incorrect login.
1001	Incorrect instrument.
1002	Incorrect client ID.
1003	Invalid member_id.
1004	Invalid account.
1005	Incorrect client group.
1006	Incorrect exchange.
1007	Instrument not traded.
1008	Invalid routing options.
1100	Invalid order direction.
1101	Incorrect price.
1102	Incorrect price_extra.
1103	Incorrect amount.
1104	Incorrect amount_extra.
1105	Invalid order type.
1106	Invalid time_in_force.
1107	Invalid passive_only.
1108	Invalid auto_cancel.
1109	Invalid flags.
1110	Invalid mode.
1111	Incorrect clorder_id.
1112	Incorrect orig_clorder_id.
1113	Invalid prime_exchange.
1114	Invalid date_expire.
1115	Invalid comment.
1200	Invalid segment.

Code	Description
1201	Incorrect extra1.
1202	Incorrect OTC code for negotiated trade initiator.
1203	Incorrect OTC code for counter party.
1204	Invalid order_type for this instrument.
1205	Order_type not supported by exchange.
1206	Invalid order_type for Client ID.
1207	Incorrect price for this order_type.
1208	Incorrect amount_extra for this order_type.
1209	Invalid time_in_force for this order_type.
1210	Invalid flags for this order_type.
1211	Invalid instrument for replacement mode.
1212	Invalid member_id for replacement mode.
1213	Invalid client_id for replacement mode.
1214	Invalid account for replacement mode.
1215	Invalid parameters of declined counter order.
1216	Invalid replacement parameters.
1217	Invalid time_in_force for this instrument.
1218	Invalid replacement mode for this login.
1219	Invalid flags for this instrument.
1300	Both orig_clorder_id and order_id filled.
1301	Duplicate clorder_id.
1302	Price exceeds limits.
1303	Order type not supported for this client ID.
1304	Order type not supported by exchange.
1305	Invalid prime_exchange for this instrument.
1306	Liquidity pool unavailable for client ID.
1307	Invalid order_type for this instrument.
1308	User has no permissions to cancel orders of account specified.
1309	User has no permissions to replace orders of account specified.
1310	User has no permissions to decline this order.
1311	Order currently being replaced.
1312	Order sent before system crash, but received after recovery.

Code	Description
1313	Limitation not available for this instrument.
1314	User has no permissions to use this mode.
1315	This exchange is prohibited for clearing member.
1316	This exchange is prohibited for trade member.
1317	Order submission via the login is blocked.
1318	Order submission via the login is blocked for the client code.
1319	Order submission via the login is blocked for the TCA.
1400	Instrument not available for market maker.
1401	No permissions to trade this instrument.
1402	No permissions to indicate 'No matching another market maker's orders'.
1403	Client has no permissions to trade with using this account.
1404	Liquidity pool not available for this smart order router.
1500	Trade engine IDs (te_id) do not match.
1501	Incorrect te_id.
1502	Request received during the limited margin update.
1700	User has no permission for limited margin service.
1701	Client has no permissions for limited margin service.
1702	Client group has no permissions for limited margin service.
1703	Account has no permissions for limited margin service.
1704	Main account has no permissions for limited margin service.
1710	Invalid parameters for limited margin of client.
1711	Invalid parameters for limited margin of client group.
1712	Invalid parameters for limited margin of account.
1713	Invalid parameters for limited margin of main account.
1714	Request for limited margin update for client received when the previous request still processing.
1715	Request for limited margin update for client group received when the previous request still processing.
1716	Request for limited margin update for TCA received when the previous request still processing.
1717	Request for limited margin update for principal TCA received when the previous request still processing.
1720	Incorrect limit for limited margin.
1721	Incorrect instrument limit for limited margin.
1722	Incorrect order limit for limited margin.
1723	Incorrect extra limit for limited margin.

Code	Description
1750	Insufficient limit for limited margin of client.
1751	Insufficient instrument limit for limited margin of client.
1752	Insufficient order limit for limited margin of client.
1753	Insufficient extra limit for limited margin of client.
1754	Insufficient limit for limited margin of client group.
1755	Insufficient instrument limit for limited margin of client group.
1756	Insufficient order limit for limited margin of client group.
1757	Insufficient extra limit for limited margin of client group.
1758	Insufficient limit for limited margin of account.
1759	Insufficient instrument limit for limited margin of account.
1760	Insufficient order limit for limited margin of account.
1761	Insufficient extra limit for limited margin of account.
1762	Insufficient limit for limited margin of main account.
1763	Insufficient instrument limit for limited margin of main account.
1764	Insufficient order limit for limited margin of main account.
1765	Insufficient extra limit for limited margin of main account.
1766	The client has active orders of limited margin.
1767	The client group has active orders of limited margin.
1768	The TCA has active orders of limited margin.
1769	The principal TCA has active orders of limited margin.
1770	Limited margin suspended for client.
1771	Limited margin suspended for client group.
1772	Limited margin suspended for account.
1773	Limited margin suspended for main clearing account.
1780	Invalid liquidity pool for limited margin service.
1980	Invalid stages in info field.
2100	Account does not belong to member_id.
2200	No permissions to submit trading instructions.
2300	No permissions to place an unsecured order.
2400	No permissions to cancel order.
2600	No permissions to set limit for clearing account.
2601	No permissions to set limits for client ID.

Code	Description
2602	No permissions to set limits for client group.
2603	Invalid type.
2604	Invalid value.
2605	Ambiguous type.
2700	Client ID has insufficient funds.
2701	Client ID has insufficient assets.
2702	Client group has insufficient funds.
2703	Client group has insufficient assets.
2704	Account has insufficient funds.
2705	Account has insufficient assets.
2706	Main clearing account has insufficient funds.
2707	Main clearing account has insufficient assets.
2708	Clearing member has insufficient funds.
2709	Insufficient blocked assets.
3000	Market or IOC order expired after no trades.
3001	Order canceled after no trades, to avoid a cross trade.
3002	Order canceled after no trades, to avoid a crossed book.
3003	Client order not found.
3004	Instrument trading suspended.
3100	TCA of maker and that of taker have no conversion bank indicator.
3911	Incorrect te_id.
4000	ECN not available or no liquidity pool available.
4001	The specified liquidity pool not available.
4002	Order forcedly routed to a liquidity pool after declined by risk management at the trading system.
4003	Client ID not registered at all the available liquidity pools.
4004	Client ID not registered at the trading system.
4005	Client ID not registered at liquidity pool.
4006	Order cannot be routed to any liquidity pool.
4100	Order pending cancel.
4200	Invalid client for TCA registered at liquidity pool.
4201	Invalid TCA for liquidity pool.
5000	Invalid application message type.

Code	Description
5001	Invalid routing_dest.
5002	Invalid message type for this login.
5003	Login has no permissions to submit such instruction.
5200	User already logged in.
5201	Discovery service settings timeout.
5202	Incorrect heatbeat_ms.
5203	Incorrect user ID / password.
5204	Incorrect message sequence number.
5205	Invalid session message type.
5206	User not logged in.
5207	Another resend request processing in progress.
5208	Incorrect range limit.
5209	Invalid reset_seq.
5210	Requested messages range excess.
5211	Invalid session message size.
5212	Disconnected by the operator.
5300	Invalid topic.
5301	Subscription already activated.
5302	Subscription not activated.
5303	Requested data not available.
5304	Another request processing in progress.
5400	Reset_seq indicated, but seqnums cannot be reset.
5601	Both account and parties filled.
7000	Order canceled before sending to ASTS.
7001	Order canceled as no answer received.

Also you can get errors come in range —11000-11999. These are the error codes returned by the trading system of the Moscow stock exchange (ASTS). To get the ASTS error id , you need to subtract 11000 from the internal error id. The description of these errors, a client can get from the ASTS documentation.

Appendix B. Revision History

Version 1.4.3 15 December 2014

Requirement to specify primary stock exchange in the order corrected.

Version 1.4.2 28 November 2014

Errors 9103, 9205, 9300, 9400, 9401, 9402, 9500, 9600, and 9601 added to error codes table.

Version 1.4.1 21 November 2014

1. Sections “Mode of negotiated repo transactions” and “Closing auction” added to section “Trading modes.”
2. New instruction types added.
3. New error codes added.
4. Necessity of fields OrdType and ExchangeSpecialInstructions for message ExecutionReport corrected.
5. Field BusinessRejectReason in message BusinessMessageReject corrected.
6. Field ExecRestatementReason in message ExecutionReport corrected.

Version 1.3.0 29 October 2014

1. New field Price1 added and description of field Price changed in messages NewOrderSingle and ExecutionReport.
2. Field DiscretionPrice added to ExecutionReport.

Version 1.2.3 16 October 2014

Necessity of field OrderQty for message ExecutionReport corrected.

Version 1.2.2 10 October 2014

1. Field ExchangeSpecialInstructions added to messages NewOrderSingle and ExecutionReport.
2. Section 3.5.1.1.1 on instruction routing added.
3. Field OrdType for negotiated order corrected.
4. New values of field BusinessRejectReason in message BusinessMessageReject corrected.
5. Field ExecRestatementReason in message ExecutionReport corrected.

Version 1.2.1 2 October 2014

New values of field TimeInForce added.

Version 1.1.0 9 June 2014

Functionality of canceling active orders on Moscow Stock Exchange by request MassCancel not available in this version.

Version 1.0 6 June 2014

Functionality of automatic order canceling in case of disconnection is not available in this version.

Version 0.3 June 2, 2014

Fields RefOrderID[1080] and ExecInst[18] added to message format NewOrderSingle[D] and ExecutionReport[8].

Version 0.2 May 8, 2014

Negotiated trading support added.