



FAST protocol specification

Version 1.4.3

Moscow 2017

Table of Contents

1. Introduction	7
1.1. Document purpose	7
1.2. Fast Gate — Basic information	7
1.2.1. Data streaming approach	7
1.2.2. Incremental messages	7
1.2.3. FIX format	7
1.2.4. Encoding in the FAST format	8
1.2.5. Data receiving via Multicast	8
1.2.6. Data recovery	8
2. Scenarios of client interactions with Market Data Multicast	9
2.1. Connect client	9
2.2. Incremental Feeds A and B Arbitration	9
3. System functionality	11
3.1. System architecture	11
3.1.1. Main streams (UDP)	11
3.1.2. Recovery streams (UDP)	12
3.1.3. Instrument Definitions streams (UDP)	12
3.1.4. Sessions for requesting missing messages (TCP)	12
3.2. FAST format — details	12
3.2.1. Stop bit encoding	13
3.2.2. Implicit tagging	13
3.2.3. Fields encoding options	13
3.2.4. FAST-template	13
3.2.5. Decoding overview	13
3.2.6. Message fragmentation	14
3.2.7. Data types	14
3.3. Missing data recovery	14
3.3.1. Recovery missing data using Recovery streams (UDP)	14
3.3.2. Recovering missing data using TCP-connection	15
3.4. Message sequence reset	15
4. FIX protocol message specifications	17
4.1. Field groups	17
4.1.1. Standard Message Header	17
4.2. Session layer messages	17
4.2.1. Logon (A)	17
4.2.2. Logout (5)	17
4.2.3. Heartbeat (0)	17
4.2.4. Sequence Reset (4)	18
4.3. Business logic layer messages	18
4.3.1. Security Definition (d)	18
4.3.2. Security Status (f)	21
4.3.3. Trading Session Status (h)	22
4.3.4. Security Definition Update Report (BP)	22
4.3.5. News (B)	23
4.3.6. Market Data Request (V)	23
4.3.7. Market Data - Snapshot / Full Refresh (W)	24
4.3.8. Market Data - Incremental Refresh (X)	25
4.4. Market Fundamentals	27
4.5. Empty book message (MDEntryType = J)	28
5. Stream of anonymous orders and trades	29
5.1. Application architecture	29
5.1.1. Streams	29
5.1.2. Messages fragmentation	29
5.1.3. Instrument IDs	29
5.2. Message templates	29
5.2.1. OrdersLog	29
5.2.2. OrdersBook	30
5.3. Messages Market Data - Incremental Refresh (X)	30
5.3.1. Adding a new order	30
5.3.2. Deleting an order	31
5.3.3. Partial order matching	32
5.3.4. Full order matching	33
5.3.5. Technical trades	34
5.3.6. Trade types	34
5.3.7. Cleaning up active orders pool (specified trading session)	37
5.3.8. Cleaning up active orders pool (all trading sessions)	37
5.4. Messages Market Data - Snapshot / Full Refresh	38
5.4.1. Active orders snapshot	38
5.4.2. When an active orders snapshot is empty	39

6. TCP Recovery (Historical Replay) service limitations	40
---	----

History of changes

Date	Version	Changes
05.02.2015	1.1.0	<ol style="list-style-type: none"> The market segment RTS Board renamed to MOEX Board - section 4.3.1. Security Definition (d) in table Moscow Exchange MarketSegmentID values. In section 4.3.1. Security Definition (d) in table Ukrainian Exchange MarketSegmentID values: <ul style="list-style-type: none"> the market segments deleted: SA, SBOND; the market segment SBONDG renamed to "State bonds"; the market segment SGTS renamed to "Shares and investment securities"; new market segments added: <ul style="list-style-type: none"> SGTF - Foreign shares and depositary receipts SBONDL - Municipal bonds SBONDE - Eurobonds (Ukraine)
21.08.2015	1.2.0	<ol style="list-style-type: none"> The channel INDEX no more transmits exchange rates with Symbol = USD and Symbol = USR. Instead, please use the exchange rate with Symbol = USD/RUB. The messages Market Data - Incremental Refresh (X) now contain the optional field Revision (tag 20018) - a service field of the replication subsystem, which is used for matching FAST messages with messages transmitted via the CGate. The field is transmitted only for quotations, trades and market fundamentals of the Derivatives market. FAST messages are now generated in a size not bigger then that of the MTU parameter, i.e. 1500 bytes, which is typical for Ethernet networks. The messages Market Data - Snapshot / Full Refresh (W) и Market Data - Incremental Refresh (X) now contain the optional field LastFragment (tag 893). For more details see the new section '3.2.6. Message fragmentation'. Fields ContractMultiplier, InstrAttribType = 203 are no more transmitted in the messages Security Definition (d) in the channel OTC-ISSUES. MarketSegmentID field's value is now 'Q'. Message templates do not now contain the compression operators copy, delta and increment. The messages Market Data - Snapshot / Full Refresh (W) and Market Data - Incremental Refresh (X) now contain the optional field OrderSide: side of the initiating order in trade. The channels FUT-BOOK-1, FUT-BOOK-5, FUT-BOOK-20, FUT-BOOK-50, OPT-BOOK-1, OPT-BOOK-5, OPT-BOOK-20, OPT-BOOK-50 in snapshot now transmit the message J (Empty book) when there are now orders for the instrument available. Packet format of the TCP Recovery now matches that of the ASTS FAST. The first 4 bytes of each FAST message now contain its length. Prices in the channel OTC-TRADES are transmitted in the field MDEntryPx in Russian roubles for all trades. The field Currency = RUB.
02.12.2015	1.2.1	<ol style="list-style-type: none"> Added description of a method for obtaining authentic instrument ID from field 'SecurityID'.
11.01.2016	1.2.2	<ol style="list-style-type: none"> Added description for stream of anonymous orders. - see sec. 5.
03.02.2016	1.2.2	<ol style="list-style-type: none"> Section '4.3.1. Security Definition (d)' – added some new info in field 'LegRatioQty' description. Added section '5.3.5. Trade on a leg of multi leg instrument'. Description of field 'MDFlags' has been changed in sections '5.3.3 Partial order matching' and '5.3.4. Full order matching'.
21.03.2016	1.2.3	<ol style="list-style-type: none"> The name of document was changed to 'FAST protocol specification'. The section 'Stream of anonymous orders' renamed to 'Stream of anonymous orders and trades' - sec. 5. The section 'Trade on a leg of multi leg instrument' renamed to 'Technical trades' - sec. 5.3.7.

Date	Version	Changes
		<p>4. Added section 'Trade types' - sec. 5.3.6.</p> <p>5. Description of field 'MDEntryType' has been changed in sections:</p> <ul style="list-style-type: none"> • Partial order matching (sec. 5.3.3), • Full order matching (sec. 5.3.4), • Technical trades (sec. 5.3.5) <p>to:</p> <p>Record type:</p> <ul style="list-style-type: none"> • 0 - Bid • 1 - Ask <p>6. Deleted field 'MDEntrySize' in section 'Deleting an order' (sec. 5.3.2).</p> <p>7. Added field 'Revision' in sections:</p> <ul style="list-style-type: none"> • Adding a new order (sec. 5.3.1), • Deleting an order (sec. 5.3.2), • Partial order matching (sec. 5.3.3), • Full order matching (sec. 5.3.4), • Technical trades (sec. 5.3.5), • Cleaning up active orders pool (specified trading session) (sec. 5.3.7), • Cleaning up active orders pool (all trading sessions) (sec. 5.3.8), <p>8. Added field 'Revision' in message template 'OrdersLog' - sec. 5.2.1.</p>
04.10.2016	1.2.4	1. Added field 'MDEntrySize' in section 'Deleting an order' (sec. 5.3.2),
26.10.2016	1.3.0	<p>1. Changed names of data types in FAST-message template (see sec. 3.2.7):</p> <ul style="list-style-type: none"> • UTC Time Only to UTCTimeOnly • UTC Date Only to UTCTimeOnly • UTC Timestamp to UTCTimestamp <p>2. Changed description of type UTCTimeOnly - see sec. 3.2.7.</p> <p>3. Updated templates for messages OrdersLog и OrdersBook - see sec. 5.2.1 and sec. 5.2.2.</p> <p>4. Added new fields (31) LastPx and (32) LastQty into tables:</p> <ul style="list-style-type: none"> • Adding a new order (sec. 5.3.1), • Deleting an order (sec. 5.3.2), • Partial order matching (sec. 5.3.3), • Full order matching (sec. 5.3.4), • Technical trades (sec. 5.3.5), • Cleaning up active orders pool (specified trading session) (sec. 5.3.7), • Cleaning up active orders pool (all trading sessions) (sec. 5.3.8). <p>5. Changed description of field MDEntrySize in tables:</p> <ul style="list-style-type: none"> • Adding a new order (sec. 5.3.1), • Technical trades (sec. 5.3.5). <p>6. Changed description of field MDEntryPx in tables:</p> <ul style="list-style-type: none"> • Partial order matching (sec. 5.3.3),

Date	Version	Changes
		<ul style="list-style-type: none"> • Full order matching (sec. 5.3.4), • Technical trades (sec. 5.3.5). <p>7. Changed description of message Empty book (sec. 4.5).</p> <p>8. Changed description of message SecurityId of table Security Definition (d) (sec. 4.3.1).</p> <p>9. Updated section Instrument IDs (sec. 5.1.3).</p> <p>10. For sections Market Data - Snapshot / Full Refresh (W) (sec. 4.3.7) and Market Data - Incremental Refresh (X) (sec. 4.3.8):</p> <ul style="list-style-type: none"> • for field MDEntryType, deleted types: <ul style="list-style-type: none"> • 'm' - Market Price • 'p' - Official Current Price; • new values added for field MDEntryTradeType: <ul style="list-style-type: none"> • for <repo type> for Ukrainian Exchange: BR18 - Repo near leg and BR19 - repo far leg; • updated description of field MDEntryDate; • changed description of field MDEntryID. <p>11. For table Standard Message Header updated description of field SenderCompID - see sec. 4.1.1.</p> <p>12. Updated description of message Heartbeat - see sec. 4.2.3.</p> <p>13. For table News (B), added field LastFragment - see sec. 4.3.5.</p> <p>14. For table Active orders snapshot, deleted description of bit: <i>0x1000 End-of-transaction bit</i> of field MDFlags - see sec. 5.4.1.</p> <p>15. Updated section Recovering missing data using TCP-connection - see sec. 3.3.2.</p>
16.01.2017	1.3.1	1. Changed description of field MDEntryTradeType for sections Market Data - Snapshot / Full Refresh (W) (sec. 4.3.7) and Market Data - Incremental Refresh (X) (sec. 4.3.8) .
20.04.2017	1.3.2	<p>1. Added description on FAST message into sections Logon (A) (sec. 4.2.1) and Logout (5) (sec. 4.2.2).</p> <p>2. Added description on new fields into section Market Data Request (V) (sec. 4.3.6).</p>
18.05.2017	1.4.0	<p>1. Added new section TCP Recovery (Historical Replay) service limitations (sec. 6) .</p> <p>2. For table Security Definition (d) added field UnderlyingFutureID - sec. 4.3.1.</p> <p>3. For sections Security Definition (d) (sec. 4.3.1) and Security Status (f) (sec. 4.3.2):</p> <ul style="list-style-type: none"> • updated description of field SecurityTradingStatus: <ul style="list-style-type: none"> • '19' - Not traded on this market
09.06.2017	1.4.1	1. Updated section TCP Recovery (Historical Replay) service limitations (sec. 6).
23.06.2017	1.4.2	1. Added new section Message sequence reset (sec. 3.4) .
22.08.2017	1.4.3	1. Updated sections Fast Gate — Basic information (раздел 1.2), Message sequence reset (sec. 3.4) and FIX protocol message specifications (sec. 4).

1. Introduction

1.1. Document purpose

This document overviews the FAST protocol specifications for the derivatives market, indexes, OTC-trades and SKRIN system news.

This document does not cover administrative and technical aspects of network connection. Also, this document does not cover security support aspect.

1.2. Fast Gate — Basic information

The **Fast Gate** system is used for distributing market data in the FAST-format via the UDP protocol in the multicast mode.

This approach combines the FIX protocol structure and message syntax with the FAST protocol dataflow optimization benefits. Also, it provides possibilities for fast and reliable data distribution to multiple clients of the UDP protocol.

The FAST (FIX Adapted for STreaming) protocol is a FIX based protocol developed by FIX Market Data Optimization Working Group in order to optimize financial data exchange performance and reduce latency in distributing large amounts of data. Fast Gate uses the protocol version 1.1: <https://www.fixtrading.org/packages/fast-specification-version-1-1>.

The **Fast Gate** system allows to distribute anonymous data to clients from:

- SPECTRA trading system;
- MOEX and UKEX index replicas;
- SKRIN news database;
- OTC monitor database;
- MOEX Board system.

The **Fast Gate** system distributes data from the following organizations:

- MOEX (Moscow Exchange);
- UKEX (Ukrainian Exchange);
- ETSC (Eurasian Trade System Commodity Exchange).

The distributed market data include:

- trades;
- quotes;
- market fundamentals;
- instruments and instruments status;
- trading session status;
- indexes;
- news;
- OTC-trades;
- stream of anonymous orders and trades.

The service is used for distributing data to large vendors such as Bloomberg and Thomson Reuters as well as to brokers, traders, etc.

1.2.1. Data streaming approach

Using of the data streaming approach allows to transmit data from sender to recipient without breaking it into separate messages. The new approach allows to combine several events into a single message which leads to higher data transfer speed and reduce latency time.

1.2.2. Incremental messages

Using of incremental messages allows to significantly reduce amount of transmitted data. Only the data changed due to the market events are transmitted; also, minimal number of commands are used for refreshing data: 'add new record', 'change record', 'delete record'.

1.2.3. FIX format

The **Fast Gate** system uses the FIX messages format and syntax. Each message consists of header, message body and trailer. Fields are separated with the ASCII symbol — <SOH>.

For more information see sec. 4.

1.2.4. Encoding in the FAST format

The FAST (FIX Adapted for STreaming) protocol is the FIX based protocol developed by FIX Market Data Optimization Working Group in order to optimize financial data exchange performance and reduce latency in distributing large amounts of data.

The following features are used for data compression:

- implicit tagging;
- fields encoding options;
- usage of PMap;
- stop-bit encoding;
- usage of binary encoding method.

In most cases, the FAST format encoding rules are negotiated between counterparties by exchanging XML-templates.

For more information see sec. 3.2.

1.2.5. Data receicing via Multicast

For data distribution, the UDP protocol is used in order to distribute data to more than one client at once.

A single UDP packet may contain several FIX messages in the FAST format. Although, currently the system does not provide a possibility to send more than one FAST-coded message via a single UDP packet. In order to match the restriction, FAST messages are generated in a size not bigger then that of the MTU parameter, i.e. 1500 bytes, which is typical for Ethernet networks.

1.2.6. Data recovery

It is extremely important to clients to be able to recover data instantly in case of any data loss.

Fast Gate provides 2 methods of data recovery:

- recovering big amounts of data by sending snapshots (for example, for the clients connected to the system after the trading session start);
- recovering small amounts of data via TCP-connection (for example, in case of message loss during sending).

2. Scenarios of client interactions with Market Data Multicast

This section covers different scenarios of clients connection to the Market Data Multicast feeds. Also, this section covers loss data recovery procedures details.

2.1. Connect client

When client starts listening to MOEX Market Data Multicast FIX/FAST Platform, it should keep the following procedure:

1. Download the actual multicast IP addresses configuration file from ftp. Configuration file is the XML - file describing the connectivity parameters (feeds, multicast addresses, ports, etc.).
2. Download the FAST template from ftp.
3. Receive the instruments list from **Instrument Replay** feed. Start listening to the **Instruments Incremental** feed.
4. Start listening to the **Incremental** feeds and queue received data.
5. Start listening to the **Snapshot** feeds. Receive and apply actual market data snapshot. In each Market Data - *Snapshot/Full Refresh (W)* tag *369-LastMsgSeqNumProcessed* is equal to tag *34-MsgSeqNum* of the last message Market Data - *Incremental Refresh (X)* of the appropriate stream included in the snapshot. The refresh number of each instrument within the tag *83-RptSeq* of the message Market Data - *Snapshot/Full Refresh (W)* is equal to number of incremental refresh in the tag *83-RptSeq* which corresponds to *MDEntry* of the last message Market Data - *Incremental Refresh (X)*, included into the snapshot. For each instrument, it is necessary to omit all messages with numbers through *369-LastMsgSeqNumProcessed* tag number and apply all that are left. The procedure can be both sequential or parallel. I.e., you can either receive snapshots for all instruments and then process the accumulated data or you can process data after receiving each snapshot.
6. Stop listening to the **Snapshot** feeds.
7. Continue receiving and normal processing incremental data.

2.2. Incremental Feeds A and B Arbitration

Data in all UDP Feeds are disseminated in two identical feeds (A and B) on two different multicast IPs. It is strongly recommended that client receive and process both feeds because of possible UDP packet loss. Processing two identical feeds allows one to statistically decrease the probability of packet loss

It is not specified in what particular feed (A or B) the message appears for the first time. To arbitrate these feeds one should use the message sequence number found in Preamble or in tag 34 - MsgSeqNum. Utilization of the Preamble allows one to determine message sequence number without decoding of FAST message.

Processing messages from feeds A and B should be performed using the following algorithm:

1. Listen feeds A and B.
2. Process messages according to their sequence numbers.
3. Ignore a message if one with the same sequence number was already processed before.
4. If the gap in sequence number appears, this indicates packet loss in both feeds (A and B). Client should initiate one of the Recovery process. But first of all client should wait a reasonable time, perhaps the lost packet will come a bit later due to packet reordering . UDP protocol can't guarantee the delivery of packets in a sequence .

Example:

Feed A
34-MsgSeqNum = 59
34-MsgSeqNum = 60
34-MsgSeqNum = 62
34-MsgSeqNum = 63
34-MsgSeqNum = 65

Feed B
34-MsgSeqNum = 59
34-MsgSeqNum = 60
34-MsgSeqNum = 61
34-MsgSeqNum = 62
34-MsgSeqNum = 65

Messages are received from Feed A and Feed B.

1. Receive message # 59 from Feed A, process it.
2. Receive message #59 from Feed B, discard it, because this message was processed before from Feed A.
3. Receive message # 60 from Feed A, process it.
4. Receive message # 60 from Feed B, discard it, because this message was processed before from Feed A.
5. Receive message # 62 from Feed A, discard it and wait for message #61.
6. Receive message # 61 from Feed B, process it.

7. Receive message # 62 from Feed B, process it.
8. Receive message # 62 from Feed A, discard it, because this message was processed before from Feed B.
9. Receive message # 63 from Feed A, process it.
- 10Receive message # 65 from Feed A, discard it and wait for message #64.
- 11.Receive message # 65 from Feed B, discard it and wait for message #64.
- 12Begin recovery process, because gap is detected. Message #64 is missed.

3. System functionality

3.1. System architecture

UDP channels used to transfer market data from MOEX. UDP channels are also used for recovery process, TCP connection is used to replay sets of lost messages, already published in one of UDP Channels.

Following feeds are used in the system:

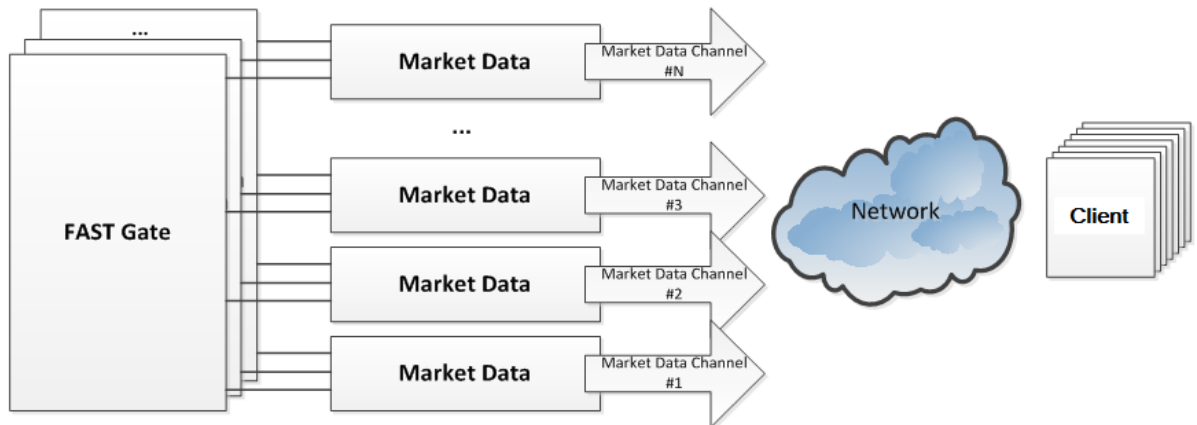
1. Basic:

- Market Data Incremental Refresh feeds;
- Instrument Definition feed;
- Data distribution feed for instrument status change and Trading System connection status.

2. Recovery feeds:

- Market Recovery feed;
- TCP Replay session.

Data are distributed via group of feeds, each of that contain data for financial instrument group. The instruments are grouped by the Trading System according to particular parameters. The dedicated Market Data Multicast instance is responsible for distribution in each Feed. A single Market Data Multicast instance is responsible for a single Feed data distribution.



Pic. 1. Market Data distribution feeds

Each feed is a) a bunch of several UDP-feeds with continuous data distribution; b) TCP-port which is used for requesting messages missed in the UDP-feed.

All streams are transmitted using the UDP multicast protocol and every stream is transmitted using a dedicated multicast address. The A and B streams transmit the same data in order to decrease the probability for missing UDP-packets.

Apart from transmitting data in UDP streams, Market Data multicast can accept incoming TCP connections for letting clients request missing data. Clients can request missing messages using one of the next UDP streams (data are available for a period of time specified in the configuration file (not earlier than from beginning of the day), number of messages to be sent at one is limited, number of requests per day is limited, too. All limits are specified in the system configuration file.

3.1.1. Main streams (UDP)

The main streams (incr) in the multicast mode with the UDP protocol is used to distribute the following market data:

- FUT-BOOK-1, FUT-BOOK-5, FUT-BOOK-20, FUT-BOOK-50, OPT-BOOK-1, OPT-BOOK-5, OPT-BOOK-20, OPT-BOOK-50 streams — aggregated Order-book data refresh;
- FUT-TRADES, OPT-TRADES streams — trade table and Derivatives market statistics data refresh;
- INDEX stream — index tables data, currency exchange rates and their fixings;
- OTC-TRADES stream — OTC trades data;
- NEWS stream — Derivatives market related news;
- NEWS-SKRIN stream — SKRIN system related news;

- SPOT-BOOK-1, SPOT-BOOK-20 - updates for aggregated quotations table;
- SPOT-TRADES - updates for trades table and market statistics;
- ORDERS-LOG - anonymous orders.

The data are distributed as FIX-messages Market Data - Incremental Refresh (X) coded in the FAST format. Each message may contain refresh data for several financial instruments.

When trading session status changes, the Trading System sends the Trading Session Status (h) message into the corresponding UDP incremental refresh streams.

3.1.2. Recovery streams (UDP)

The Recovery (snap) streams in the multicast mode with the UDP protocol are used to periodically distribute the current snapshot of the corresponding data as FIX-messages Market Data - Snapshot/Full Refresh (W) coded in the FAST format. Each message contains data for a single instrument only.

It is not necessary for clients to be constantly connected to these streams. After receiving the missing data, it is recommended to disconnect from these streams.

Every snapshot has the Trading Session Status (h) message in the end.

3.1.3. Instrument Definitions streams (UDP)

The Instrument Replay (inst replay) streams are used to periodically distribute descriptions of financial instruments as FIX messages coded in the FAST format. Each message contains description for a single financial instrument.

In case of instrument status change, collateral volume change or price limits change, the Instrument Incremental (inst incr) stream transmits the Security Status (f) FIX-messages.

Transmitted data:

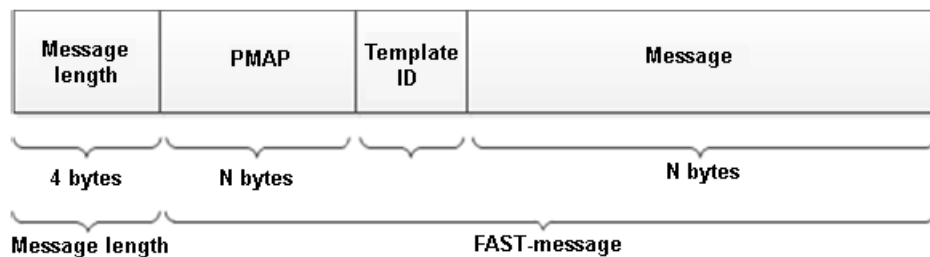
- FUT-INFO stream — futures;
- OPT-INFO stream — options;
- OTC-ISSUES stream — OTC-instruments;
- SPOT-INFO stream – shares, bonds, commodities.

3.1.4. Sessions for requesting missing messages (TCP)

This service allows to request the resend of missing messages within a specified range of numbers.

The request contains a range of message (numbers) to resend. The request is sent as the Market Data Request (V) FIX-message using the client-initiated TCP-connection. The respond messages are sent to the client as FIX-messages coded in the FAST format using the same TCP-connection. Upon completion of sending, Market Data Multicast closes this TCP-connection. Please note, that maximum number of messages to resend is limited.

The first 4 bytes of each message transmitted in a TCP stream contain its length.



Pic. 2. Message structure in TCP stream

When all FAST messages have been sent out, the gateway sends the message Logout to the FAST client, expecting the message Logout from the client in respond. Finishing FIX session also causes TCP session to close.

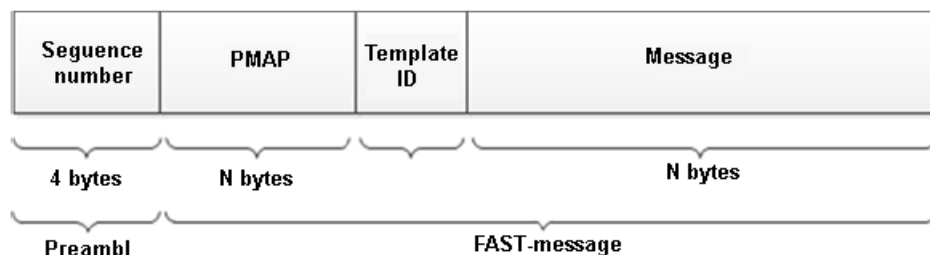
Please also note, that this service should be used only when all other methods are unavailable. This service does not provide high performance and is not available for streams containing aggregated Book-order data.

3.2. FAST format — details

All messages sent by MOEX Market Data Multicast are in the FIX-format coded in the FAST (FIX Adapted for STreaming) protocol. The FAST protocol was developed by FIX Market Data Optimization Working Group in order to optimize financial data flow via distributing bigger amounts of data with less latency.

A specific feature of data distribution via the MOEX Market Data Multicast streams is, that there is a 4-bytes preamble added before every FAST-message. The preamble contains the 34-th tag (SeqNum) value. The 34-th tag is located right after the preamble.

It allows to receive the message sequence number (both when processing messages from the A and B streams and in case of missing messages) without decoding the FAST-message itself; this leads to time saving during processing of streams.



Pic. 3. Message structure

3.2.1. Stop bit encoding

Encoding stop bit is a constitutive procedure of FAST. The coding allows to exclude redundancy on the data field link layer using the stop bit instead of the standard byte separator. In FAST, stop bit is used instead of the standard FIX-separator (<SOH> byte); therefore, 7 bits of every byte are used for data transmission while the 8th bit indicates the field end.

3.2.2. Implicit tagging

According to the FIX protocol standards, every message is as: **Tag = Value <SOH>**, where:

Tag — number of the field, which is now transmitted;

Value — actual value in this field;

<SOH> – ASCII symbol, used as a separator.

Example:

35=x|268=3 (message header) 279=0|269=2|270=9462.50|271=5|48=800123|22=8 (trade) 279=0|269=0|270=9462.00|271=175|1023=1|48=800123|22=8|346=15 (new bid 1) 279=0|269=0|270=9461.50|271=133|1023=2|48=800123|22=8|346=12 (new bid 2)

FAST allows to avoid this redundancy by using a template which describes the whole message structure. This method is called 'implicit tagging', as FIX tags become implicit parts of the transmitted data. FAST-template exchanges the 'Tag=Value' syntax with 'implicit tagging' according to the following rules:

- tags numbers are not transmitted in message but specified in the template;
- sequence of fields in the message is alike to one of the tags in the template;
- the template specifies a structured bunch of fields with their operators.

3.2.3. Fields encoding options

FAST operates as a state machine, which must 'know' all values to store in memory each moment of time. FAST compares the current field value with the previous one and decides how to act:

- use the constant specified in the template as a new value;
- use the default value (in case of absence of a new field value).

3.2.4. FAST-template

A FAST template corresponds to the FIX message type and uniquely identifies order of fields in each message.

The template also includes syntax indicating the type of field and transfer decoding to apply. Each FAST message contains template ID which is used for decoding. The message templates are available at: <ftp://ftp.moex.com/pub/FAST/Spectra/>.

3.2.5. Decoding overview

Below is the order of decoding procedure:

1. Transport. A client receives an encoded FAST message.
2. Packet decoding:

- identification of a template;
 - withdrawal of binary encoded bits;
 - determining correspondences between the received bits and template fields.
3. Fields decoding: using operators to determine value according to the template.
 4. Generation of FIX-message.
 5. Processing the FIX-message.

3.2.6. Message fragmentation

In order to prevent UDP packets from exceeding MTU size of 1500 bytes (typical for Ethernet networks), messages are fragmented into several parts.

If the message Market Data - Snapshot / Full Refresh (W) does not contain the tag 893-LastFragment, it means that snapshot was transmitted as a single message. All fragmented messages except the last one contain the tag 893-LastFragment = 0. The last fragmented messages contains the tag 893-LastFragment = 1. Therefore, receiving a message with the tag 893-LastFragment = 1 indicates that snapshot has been completely transmitted.

If the message Market Data - Incremental Refresh (X) does not contain the tag 893-LastFragment, it means that messages have not been fragmented, and the database is consistent after processing the message. All fragmented messages except the last one contain the tag 893-LastFragment = 0. The last fragmented messages contains the tag 893-LastFragment = 1. Therefore, receiving a message with the tag 893-LastFragment = 1 indicates that the database is consistent.

3.2.7. Data types

A field within a FAST template will have one of the standard Data Types indicating the required decoding action: ASCII string, Unicode string, Signed Integer, Unsigned Integer and Decimal. Decimal exponent and mantissa will be encoded as a single, composite field.

FAST does not natively support timestamps. FAST gate will convert the timestamp to an integer value depending on the field type. The decoding application should convert the integer to the FIX UTC format after decoding. Time is always displayed in UTC.

Samples of timestamps encoding:

FIX Type	FIX Pattern	Sample FIX value	Sample FAST value	FAST field type
UTCTimeOnly	HH:MM:SS.ssssssss (nanoseconds)	18:44:24.123456789	184424123456789	uInt64
UTCDateOnly	YYYYMMDD	20080812	20080812	uInt32
UTCTimestamp	YYYYM- MDD-HH:MM:SS.sss	20080812-18:23:54.213	20080812182354123	uInt64

3.3. Missing data recovery

MOEX Market Data Multicast FIX/FAST Platform disseminates Market Data in all feeds over two UDP subfeeds: Feed A and Feed B. In Feeds A and B the identical messages are sent. It lowers the probability of packets loss and provides the first level of protection against missed messages.

Sometimes, messages may be missed on both feeds, requiring a recovery process to take place. Message loss can be detected using the FIX message sequence numbers (tag MsgSeqNum (34)), which are also found in the Preamble. The message sequence number is an incrementing number; therefore, if a gap is detected between messages in the tag MsgSeqNum (34) value, or the Preamble sequence number, this indicates a message has been missed. In addition, tag RptSeq (83) can be used to detect a gap between the messages at the instrument level. In this case client system should assume that market data maintained in it is no longer correct and should be synchronized to the latest state using one of the recovery mechanisms.

MOEX Market Data Multicast FIX/FAST Platform offers several options for recovering missed messages and synchronizing client system to the latest state. Market Recovery process together with Instruments Replay Feed is the recommended mechanism for recovery. TCP Replay provides less performance mechanism recommended only for emergency recovering of small amount of lost messages when other mechanisms cannot be used for some reason. Instrument level sequencing and natural refresh can be utilized to supplement the recovery process.

3.3.1. Recovery missing data using Recovery streams (UDP)

This recovery method is preferable to use for large - scale data recovery and for late joiners. Recovery feeds contains Market Data - Snapshot/Full Refresh (W) messages. The sequence number (LastMsgSeqNumProcessed(369)) in the Market Data - Snapshot/Full Refresh (W) message corresponds to the sequence number (MsgSeqNum(34)) of the last Market Data - Incremental Refresh (X) message in the corresponding feed. Instrument level sequence number (RptSeq(83)) in Market Data - Snapshot/Full Refresh (W) message correspond to the sequence number (RptSeq(83)) in the MDEntry from last Market Data - Incremental Refresh (X) message. Thus, tag MsgSeqNum(34) shows the gap at the messages level, tag RptSeq(83) shows gap at the instrument level.

After value of RptSeq(83) tag from Market Data - Incremental Refresh (X) becomes more than value of RptSeq(83) tag from Market Data - Incremental Refresh (X) , market data becomes actual.

After value of MsgSeqNum(34) from Market Data - Incremental Refresh (X) message becomes more than value of tag LastMsgSeqNumProcessed(369) from Market Data - Snapshot/Full Refresh (W) message, market data becomes actual.

Messages sequence numbers begins from #1 in Market Data - Snapshot/Full Refresh (W) messages in each cycle.

If a message does not contain the tag 893-LastFragment, it means that snapshot was transmitted as a single message. Otherwise, the last fragmented messages contains the tag 893-LastFragment = 1. Therefore, receiving a message with the tag 893-LastFragment = 1 indicates that snapshot has been completely transmitted.

Clients should keep queuing real - time data until all missed data is recovered. The recovered data should then be applied prior to data queued.

Consequence of recovery is equal to that described in sec. 2.1 (steps 4 - 7).

Since clients have retrieved recovery data, it is recommended to stop listening Market Recovery feeds.

3.3.2. Recovering missing data using TCP-connection

If there any market data missing in incremental streams Indexes, Trades and ORDERS-LOG (anonymous orders and trades), it can be recovered over the TCP historical replay component using the sequence number range. TCP Replay is a low performance recovery option and should only be used if other options are unavailable or for small - scale data recovery. Number of messages which can be requested by client during TCP connection is limited to 1000.

To request missing data, you should do the following:

1. Establish TCP connection with MOEX Market Data Multicast.
2. Send FIX message Logon(A) with sequence number 1 to server. After successful authorization server sends the FAST - encoded Logon(A) message.
3. Send Market Data Request (V) message with:
 - a. Range of sequence numbers - ApplBegSeqNum(1182) and ApplEndSeqNum (1183) tags.

If request is correct, server sends FAST messages according to requested sequence numbers.

If request is incorrect, server sends FAST Logout (5) message with reject reason.

After server responses, the connection is closed.

Server will process only first user request, second and others will be ignored. If the server does not receive Market Data Request within an established timeout interval after logon, the connection is closed.

Recovery channel has 1 second incoming request timeout.

3.4. Message sequence reset

Every 24 hours, the Fast Gate is being cleaned up from the last day trading session messages, and its message sequences are being reset. When the message sequences have been reset, a message 'Sequence Reset' with a new value in the field 'NewSeqNo' will be transmitted in the (incr) streams. Upon receiving the message 'Sequence Reset', the client is to set the message number value to that transmitted in the message 'NewSeqNo', and reset 'RptSeq' numbers.

Below is the break time schedule for Fast Gate. In the end of each break time, message sequence numbers will be reset:

- MOEX Derivatives Market - 0:00 AM (Moscow Time) till 09:00 AM (Moscow Time);
- OTC Monitor, MOEX Board, SKRIN News - 0:00 AM (Moscow Time) till 00:02 AM (Moscow Time);
- UKEX (Ukrainian Exchange) - 0:00 AM (Moscow Time) till 09:00 AM (Moscow Time);
- ETSC (Eurasian Trade System Commodity Exchange) - 0:00 AM (Moscow Time) till 06:00 AM (Moscow Time);
- Derivatives market test environment - 0:00 AM (Moscow Time) till 05:55 AM (Moscow Time);
- Test environment, i.e. OTC Monitor, MOEX Board, SKRIN News - 0:00 AM (Moscow Time) till 00:02 AM (Moscow Time);

For all the main (incr) streams, excluding the streams 'FUT-TRADES' and 'OPT-TRADES', the message sequence number will be set to 1, and 'RptSeq' number will be set to 1. For the main (incr) streams 'FUT-TRADES' and 'OPT-TRADES', the message number sequences will be set to a value $N \geq 1$, and 'RptSeq' number will be set to a value which is ≥ 1 . Also, the message pair 'Sequence Reset' will be transmitted:

- Sequence Reset: MsgSeqNum=N NewSeqNo[36]=1
- Sequence Reset: MsgSeqNum=N NewSeqNo[36]=N

After those messages have been transmitted, the FAST-messages containing trading data of the last evening trading session, with numbers from 1 till N-1 inclusive, will become available through the TCP Recovery service. The initial 'RptSeq' number value can be obtained with one of the following methods:

- request and process messages with numbers from 1 till N-1 available through the TCP Recovery service;
- connect to Recovery (UDP) stream, in accordance with information provided in section 3.3.1 Recovery missing data using Recovery streams (UDP).

4. FIX protocol message specifications

The protocol message specifications description below is based on the standard FIX protocol specification v. 5.0 SP2 (<https://www.fixtrading.org/standards/fix-5-0-sp-2>). It is recommended for users to read some general information about the protocol before commencing with this specification.

4.1. Field groups

Many messages contain the same fields. For example, the 'Standard Message Header' group fields contain some administrative information and are mandatory for every message.

4.1.1. Standard Message Header

The standard header, mandatory for every message.

Tag	Field	Mandatory	Details	Available values
34	MsgSeqNum	Y	Message sequence number	
35	MsgType	Y	Message type	
49	SenderCompID	Y	Message sender ID	<ul style="list-style-type: none"> 'MOEX' - Moscow Exchange 'UKEX' - Ukrainian Exchange (UX) 'ETSC' - Kazakhstan Exchange (ETS)
52	SendingTime	Y	Message sending time	
1128	ApplVerID	Y	FIX protocol version ID	"9" (FIX50SP2)

4.2. Session layer messages

4.2.1. Logon (A)

A FIX message which is used to initiate a session establishment to the service TCP Recovery.

Tag	Field	Mandatory	Details
8	BeginString	Y	Allowable values: 'FIX.4.4' and 'FIXT.1.1'.
9	BodyLength	Y	Message length.
35	MsgType	Y	'A'
553	Username	N	Any string
554	Password	N	Any string
10	Checksum	Y	Checksum.

A FAST message which is used to confirm a session establishment to the service TCP Recovery.

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	Message type 'A'.

4.2.2. Logout (5)

A FIX message which is used to initiate a session closure with the service TCP Recovery.

Tag	Field	Mandatory	Details
8	BeginString	Y	Allowable values: 'FIX.4.4' and 'FIXT.1.1'.
9	BodyLength	Y	Message length.
35	MsgType	Y	'5'
10	Checksum	Y	Checksum.

A FAST message which is used to confirm a session closure with the service TCP Recovery.

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	Message type '5'.

4.2.3. Heartbeat (0)

The message **HeartBeat** is sent by FastGate when there were no messages sent in the stream within a 30 seconds time interval.

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	Message type '0'.

4.2.4. Sequence Reset (4)

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	Message type '4'.
36	NewSeqNo	Y	New sequence number.

4.3. Business logic layer messages

This section describes messages of all streams, excluding messages of the anonymous orders and trades stream (ORDERS-LOG) (see sec. 5 below).

The following FIX messages are supported:

- **Security Definition** – Information on instrument.
- **Security Status** – Status and price limit change, change of collateral volume for instrument.
- **Trading Session Status** – Trading session status.
- **Security Definition Update Report** – Volatility and theoretical prices for options.
- **News** – SPECTRA system administrator's messages. SKRIN system news.
- **Market Data Request** - Missed data request.
- **Market Data - Snapshot / Full Refresh** – Data snapshot (for example, the Order-book full status).
- **Market Data - Incremental Refresh** – Data refresh.

4.3.1. Security Definition (d)

Information on instrument.

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	Message type 'd'
911	TotNumReports	Y	Total messages number in the current list.
1301	MarketId*	Y	Exchange MIC: <ul style="list-style-type: none"> • 'MOEX' - Moscow Exchange • 'UKEX' - Ukrainian Exchange • 'ETSC' - ETS Eurasian Trading System Commodity Exchange
48	SecurityId	C	Instrument unique ID.
22	SecurityIdSource	C	"8" - Exchange Symbol
55	Symbol	N	Symbol code of the instrument
107	SecurityDesc	N	Instrument name
167	SecurityType	N	Multileg type - 'MLEG' — calendar spread
461	CFICode	N	Financial instrument class according to ISO-10962. Valid values are shown in the table below.
231	ContractMultiplier	N	Units of underlying asset in instrument.
969	MinPriceIncrement	N	Minimum price step.
1146	MinPriceIncrementAmount	N	Price step cost.
15	Currency	N	Currency: <ul style="list-style-type: none"> • RUB - roubles • USD - US dollars • XXX - percent points
1148	LowLimitPx*	N	Lower price limit.
1149	HighLimitPx*	N	Upper price limit.
1300	MarketSegmentId*	N	Market segments. Valid values are shown in the table below.

Tag	Field	Mandatory	Details
336	TradingSessionId	N*	Trading session type: <ul style="list-style-type: none"> '1' — main session '3' — early session '5' — evening session
5842	ExchangeTradingSessionId*	N	Trading session ID.
5678	Volatility*	N	Option volatility.
20006	TheorPrice*	N	Option theoretical price.
20007	TheorPriceLimit*	N	Option theoretical price (limits adjusted).
20002	InitialMarginOnBuy*	N	<ul style="list-style-type: none"> futures — buyer collateral options — underlying collateral for buying futures-style option
20000	InitialMarginOnSell*	N	<ul style="list-style-type: none"> futures — seller collateral options — underlying collateral for one uncovered position
20001	InitialMarginSyntetic*	N	Underlying collateral for one uncovered position (RUB). Options only.
326	SecurityTradingStatus*	N	Instrument trading status: <ul style="list-style-type: none"> '21' — session initiated '17' — session started '2' — session paused '18' — session ended '19' — not traded on this market '20' — unknown or invalid
711	NoUnderlyings	N	=1
=> 311	UnderlyingSymbol	N	Underlying asset code
=> 309	UnderlyingSecurityID	N	Futures instrument ID
=> 2620	UnderlyingFutureID	N	ID of the base futures instrument, applicable to options only.
1141	NoMDFeedTypes	N*	Number of duplicated blocks
=>1022	MDFeedType	N	Feed type.
=>264	MarketDepth	N	Order-book depth.
=>1021	MDBookType	N	Order-book type: <ul style="list-style-type: none"> '1' — Top of Book '2' — Price Depth
555	NoLegs	N	=2
=> 600	LegSymbol	N	Symbol code of the multi-leg instrument
=> 602	LegSecurityID	N	Multi-leg instrument code
=> 623	LegRatioQty	N	Quantity ratio. Value of field 'LegRatioQty' indicates both amount and direction of a multi leg instrument, i.e. if the field 'LegRatioQty' contains a value greater than 0, then the multi leg instrument has the same direction as the multi leg order, while a value less than 0 indicates a direction of this multi leg instrument opposite to that of the multi leg order. The absolute value of the field 'LegRatioQty' multiplied by multi leg instrument amount in the order allows to obtain the instrument amount value for field 'LegSymbol'.
455	SecurityAltID*	N	Instrument symbol code.
456	SecurityAltIDSource*	N	Class for SecurityAltID (455): <ul style="list-style-type: none"> '8' — Exchange Symbol '4' — ISIN number
864	NoEvents	N	<ul style="list-style-type: none"> '2' — for futures

Tag	Field	Mandatory	Details
			• '4' — for options
=>865	EventType	N	EventType=7. Last trading day.
=>866	EventDate		
=>1145	EventTime		
=>865	EventType	N	EventType=5. Instrument trading start date.
=>866	EventDate		
=>1145	EventTime		
=>865	EventType	N	EventType=100. Instrument exercise start date.
=>866	EventDate		
=>1145	EventTime		
=>865	EventType	N	EventType=101. Instrument exercise end date.
=>866	EventDate		
=>1145	EventTime		
541	MaturityDate	N	Instrument settlement date. Futures only.
1079	MaturityTime	N	
870	NoInstrAttrib	N	=3
=> 871	InstrAttribType	N	=203
=> 872	InstrAttribValue	N	Instrument type by SWIFT.
=> 871	InstrAttribType	N	=204
=> 872	InstrAttribValue	N	State registration number.
=> 871	InstrAttribType	N	=200
=> 872	InstrAttribValue	N	Total number of securities by issuer, in units.
202	StrikePrice	N	Strike price.
20005	QuotationList	N	Quotation List.
879	UnderlyingQty	N	Security nominal value.
318	UnderlyingCurrency	N	Code of currency of the security nominal value.

* - differs from the standard FIX protocol.

Table 1. Moscow Exchange MarketSegmentID values

MarketId	MarketSegmentId	CFIcode	SecurityType	Description
MOEX	F	FXXXXX		Futures
MOEX	F	FXXXXX	MLEG	Calendar spreads
MOEX	O	OCAFPX OPAFPX OCEFPX OPEFPX		Options: <ul style="list-style-type: none"> • "OCAFPX" — American-style option Call • "OPAFPX" — American-style option Put • "OCEFPX" — European-style option Call • "OPEFPX" — European-style option Put
MOEX	I			Indexes
MOEX	Q	ESXXXX DBXXXX		OTC Trades
MOEX	SBOARD	ESXXXX		MOEX Board, Equities
MOEX	SBOARDC	ESXXXX		MOEX Board, Equities
MOEX	SBOARDDB	ESXXXX		MOEX Board, Bonds
MOEX	SKRIN			SKRIN News

Table 2. Ukrainian Exchange MarketSegmentID values

MarketId	MarketSegmentId	CFI Code	SecurityType	Description
UKEX	SBONDG	DBXXXX		State bonds
UKEX	SGTS	ESXXXX		Shares and investment securities
UKEX	SGTB	ESXXXX		Corporate bonds
UKEX	F	FXXXXX		Futures
UKEX	O	OCAFPX OPAFPX OCEFPX OPEFPX		Options: <ul style="list-style-type: none"> • "OCAFPX" — American-style option Call • "OPAFPX" — American-style option Put • "OCEFPX" — European-style option Call • "OPEFPX" — European-style option Put
UKEX	SGTF	ESXXXX		Foreign shares and depositary receipts
UKEX	SBONDL	DBXXXX		Municipal bonds
UKEX	SBONDE	DBXXXX		Eurobonds (Ukraine)

Table 3. ETS MarketSegmentID values

MarketId	MarketSegmentId	CFI Code	SecurityType	Description
ETSC	SAG_CLS	ESXXXX		Quote driven market, agricultural products
ETSC	SAG_CLSN	ESXXXX		Quote driven market, agricultural products (without value added tax)
ETSC	SOL_CLS	ESXXXX		Section of trade in oil products
ETSC	SOL_CLSN	ESXXXX		Section of trade in oil products (without value added tax)
ETSC	SMT_CLS	ESXXXX		Section of trade in metals
ETSC	SAG_AA	ESXXXX		Agricultural products
ETSC	SMT_AA	ESXXXX		Manufactured goods

4.3.2. Security Status (f)

The message is transmitted at change of instrument status, price limits or collateral volume.

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	Message type 'f'
48	SecurityId	C	Instrument numerical code
22	SecurityIdSource	C	'8' — Exchange Symbol
55	Symbol	N	Symbol code of the instrument
326	SecurityTradingStatus*	N	Instrument trading status: <ul style="list-style-type: none"> • '21' — session initiated • '17' — session started • '2' — session paused • '18' — session ended • '19' — not traded on this market • '20' — unknown or invalid
1148	LowLimitPx*	N	Lower price limit.
1149	HighLimitPx*	N	Upper price limit.
20002	InitialMarginOnBuy*	N	<ul style="list-style-type: none"> • futures — buyer collateral • options — underlying collateral for buying futures-style option
20000	InitialMarginOnSell*	N	<ul style="list-style-type: none"> • futures — seller collateral • options — underlying collateral for one uncovered position

Tag	Field	Mandatory	Details
20001	InitialMarginSyntetic*	N	Underlying collateral for one uncovered position (RUB). Options only.

4.3.3. Trading Session Status (h)

The message is transmitted at the start and in the end of trading sessions and intraday clearing session.

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	Message type 'h'
336	TradingSessionId	Y	Trading session type: <ul style="list-style-type: none"> '1' — main session '3' — early session '5' — evening session
5842	ExchangeTradingSessionID*	N	Trading session ID
340	TradSesStatus	Y	Состояние торговой сессии: <ul style="list-style-type: none"> '4' — session initiated '2' — session started '1' — session paused '3' — session ended
1368	TradSesEvent	N	Trading session events: <ul style="list-style-type: none"> '0' — Trading resumed after intraday clearing session '1' — Start and end of trading session '3' — Trading session status change
1301	MarketId	N*	Exchange MIC: <ul style="list-style-type: none"> 'MOEX' — Moscow Exchange 'UKEX' — Ukranian Exchange 'ETSC' — ETS Eurasian Trading System Commodity Exchange
1300	MarketSegmentId	N*	Market segments: <ul style="list-style-type: none"> 'F' — Futures 'O' — Options 'I' — Indexes 'S*' — Securities, Bonds, Commodities
342	TradSesOpenTime	N	Trading session open time and date
344	TradSesCloseTime	N	Trading session close time and date
5840	TradSesIntermClearingStartTime*	N	Intraday clearing session start time
5841	TradSesIntermClearingEndTime*	N	Intraday clearing session end time

* - differs from the standard FIX protocol.

4.3.4. Security Definition Update Report (BP)

Options volatility and theoretical prices.

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	MESSAGE TYPE 'BP'
48	SecurityId	C	Instrument numeric code
22	SecurityIdSource	C	'8' — Exchange Symbol
5678	Volatility*	N	Option volatility.
20006	TheorPrice*	N	Option theoretical price.

Tag	Field	Mandatory	Details
20007	TheorPriceLimit*	N	Option theoretical price (limits adjusted)

* - differs from the standard FIX protocol.

4.3.5. News (B)

SPECTRA system administrator's messages. SKRIN system news.

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	Message type 'B'
893	LastFragment	N	This field indicates whether the message is the last in the series generated for a news message with NewsID. Allowable values: <ul style="list-style-type: none"> • 0 – last message • 1 – not the last message The field is non-mandatory. When absent, the message packet is considered as non-fragmented.
1472	NewsID	N	News ID
42	OrigTime	N	News date and time
1474	LanguageCode	N	Language
61	Urgency	N	Urgency
148	Headline	Y	Header text
33	NoLinesOfText	Y	=1
=>58	Text	Y	Message text body. The string is transmitted by FAST gate in UTF-8 character set
1301	MarketId	N	Exchange MIC: <ul style="list-style-type: none"> • 'MOEX' — Moscow Exchange • 'UKEX' — Ukrainian Exchange • 'ETSC' — ETS Eurasian Trading System Commodity Exchange
1300	MarketSegmentId	N	Market segments: <ul style="list-style-type: none"> • 'F' — futures • 'O' — options • 'SKRIN'

4.3.6. Market Data Request (V)

A FIX message which is used to request missing data in the session to the service TCP Recovery.

Tag	Field	Mandatory	Details
8	BeginString	Y	Allowable values: <ul style="list-style-type: none"> • FIX.4.4 • FIXT.1.1
9	BodyLength	Y	Message length
35	MsgType	Y	"V"
262	MDReqId	Y	Request ID
1182	ApplBegSeqNum	N	Sequence number of the first requested message.
1183	ApplEndSeqNum	N	Sequence number of the last requested message. If a single message is requested, then ApplBegSeqNum(1182)=ApplEndSeqNum(1183). If all messages are requested (no more than total messages sent) after a particular message number, then ApplEndSeqNum(1183)=0(infinity).
10	Checksum	Y	Checksum

4.3.7. Market Data - Snapshot / Full Refresh (W)

Data snapshot.

Tag	Field	Mandatory	Details
<Standard	Message Header>	Y	Message type 'W'
893	LastFragment	N	Indicates the last message in the message group for the instrument. Allowable values: <ul style="list-style-type: none"> • 0 – not the last message • 1 – the last message The field is not mandatory. If a message does not contain this field, it means that the packet with message has not been fragmented.
911	TotNumReports	Y	Messages in the current list, total.
83	RptSeq	Y	The 'RptSeq' number of the last incremental update included in the current market data snapshot for instrument.
369	LastMsgSeqNumProcessed	N	The 'MsgSeqNum' of the last message sent into incremental feed at the time of the current snapshot generation.
48	SecurityId	N	Instrument numeric code.
22	SecurityIdSource	C	'8' — Exchange Symbol
55	Symbol	N	Symbol code of the instrument, currency exchange rate name, index name.
1151	SecurityGroup	N	=OTC
268	NoMDEntries	Y	Number of 'MDEntry' records in the current message.
=>20003	MDEntryTradeType	N	MDEntryTradeType is sent for Spot market trades. The tag value format is <trade type><repo type><trade status>. Possible trade types: <ul style="list-style-type: none"> • 'Q' - quote-based transaction • 'B' - two-sided transaction • 'A' - auction-based transaction • 'I' - IPO transaction Possible repo types: <ul style="list-style-type: none"> • ' ' (space) - regular trade • 'R' - repurchase agreement Possible trade statuses: <ul style="list-style-type: none"> • '18' - Repo near leg (Ukrainian Exchange repos) • '19' - Repo far leg (Ukrainian Exchange repos)
=>269	MDEntryType	Y	Record type: <ul style="list-style-type: none"> • '0' — Bid • '1' — Ask • '2' — Trade • '3' — Index Value • '4' — Opening Price • '5' — Closing Price • '6' — Settlement Price • '7' — Trading Session High Price • '8' — Trading Session Low Price

Tag	Field	Mandatory	Details
			<ul style="list-style-type: none"> '9' — Trading Session VWAP Price 'B' — Cumulative Trade Volume 'C' — Open Interest 'v' — Total Bid Volume 'w' — Total Offer Volume 's' — Market Capitalization 'J' — Empty book
=>5842	ExchangeTradingSessionId*	N	Trading session ID
=>278	MDEntryID	N	Trade ID
=>264	MarketDepth	N	Market depth
=>270	MDEntryPx	N	Price, rate and index values.
=>271	MDEntrySize	N	Volume, quantity.
=>1023	MDPriceLevel	N	Price level.
=>272	MDEntryDate	N	Record last change date. Value NULL indicates the current date.
=>273	MDEntryTime	N	Record last change time.
=>346	NumberOfOrders	N	<ul style="list-style-type: none"> Sell orders quantity (for records with MDEntryType=v (Total bid volume)*). Buy orders quantity (for records with MDEntryType=w (Total offer volume)*).
=>828	TrdType	C	Trade type: <ul style="list-style-type: none"> '0' — Market trade. '22' — Negotiated trade. '45' — Option exercise. '1000' — Futures exercise (standard method). Non-deliverable futures exercise. '1002' — T+0 trade (position rollover). '1003' — Option expiration. The field is mandatory for records with MDEntryType=2 (Trade).
=>20017	MDFlags*	N	<ul style="list-style-type: none"> '0x1' — Flag of fixing for the main clearing session. '0x100' — Flag of fixing.
=>15	Currency	N	Currency code
=>10504	OrderSide	N	Side of the initiating order in trade. Transmitted only for the Derivatives market. Allowable values: <ul style="list-style-type: none"> '1' – buy order (Buy); '2' – sell order (Sell).

* - differs from the standard FIX protocol.

4.3.8. Market Data - Incremental Refresh (X)

Data refresh.

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	Message type 'X'
893	LastFragment	N	Indicates the last message in the message group for the instrument. Allowable values: <ul style="list-style-type: none"> '0' – not the last message

Tag	Field	Mandatory	Details
			<ul style="list-style-type: none"> '1' – the last message <p>The field is not mandatory. If a message does not contain this field, it means that the packet with message has not been fragmented.</p>
268	NoMDEntries	Y	Number of 'MDEntry' records in the current message.
=>83	RptSeq	Y	Incremental refresh sequence number
=>279	MDUpdateAction	Y	<p>Incremental refresh type:</p> <ul style="list-style-type: none"> '0' — New '1' — Change '2' — Delete
=>20003	MDEntryTradeType	N	<p>MDEntryTradeType is sent for Spot market trades.</p> <p>The tag value format is <trade type><repo type><trade status>.</p> <p>Possible trade types:</p> <ul style="list-style-type: none"> 'Q' - quote-based transaction 'B' - two-sided transaction 'A' - auction-based transaction 'I' - IPO transaction <p>Possible repo types:</p> <ul style="list-style-type: none"> ' ' (space) - regular trade 'R' - repurchase agreement <p>Possible trade statuses:</p> <ul style="list-style-type: none"> '18' - Repo near leg (Ukrainian Exchange repos) '19' - Repo far leg (Ukrainian Exchange repos)
=>31	LastPx	N	Used for MDUpdateAction=2 only. Last exchange trade price.
=>269	MDEntryType	Y	<p>Record type:</p> <ul style="list-style-type: none"> '0' — Bid '1' — Ask '2' — Trade '3' — Index Value '4' — Opening Price '5' — Closing Price '6' — Settlement Price '7' — Trading Session High Price '8' — Trading Session Low Price '9' — Trading Session VWAP Price 'B' — Cumulative Trade Volume 'C' — Open Interest 'v' — Total Bid Volume 'w' — Total Offer Volume 's' — Market Capitalization 'J' — Empty book
=>48	SecurityId	N	Instrument numeric code

Tag	Field	Mandatory	Details
=>22	SecurityIdSource	C	'8' — Exchange Symbol
=>5842	ExchangeTradingSessionId*	N	Trading session ID
=>278	MDEntryID	N	Trade ID
=>264	MarketDepth	N	Market depth
=>270	MDEntryPx	N	Price, rate and index values.
=>271	MDEntrySize	N	Volume, quantity.
=>1023	MDPriceLevel	N	Price level.
=>272	MDEntryDate	N	Record last change date. Value NULL indicates the current date.
=>273	MDEntryTime	N	Record last change time.
=>346	NumberOfOrders	N	<ul style="list-style-type: none"> Sell orders quantity (for records with MDEntryType=v (Total bid volume)*). Buy orders quantity (for records with MDEntryType=w (Total offer volume)*).
=>828	TrdType	C	Trade type: <ul style="list-style-type: none"> '0' — Market trade. '22' — Negotiated trade. '45' — Option exercise. '1000' — Futures exercise (standard method). Non-deliverable futures exercise. '1002' — T+0 trade (position rollover). '1003' — Option expiration. The field is mandatory for records with MDEntryType=2 (Trade).
=>55	Symbol	N	Symbol code of the instrument, currency exchange rate name, index name.
=>20017	MDFlags*	N	<ul style="list-style-type: none"> '0x1' — Flag of fixing for the main clearing session. '0x100' — Flag of fixing.
=>15	Currency	N	Currency code.
=>1151	SecurityGroup	N	=OTC
=>20018	Revision	N	Service field of the replication subsystem. The field is transmitted only for quotations, trades and market fundamentals of the Derivatives market.
=>10504	OrderSide	N	Side of the initiating order in trade. Transmitted only for the Derivatives market. Allowable values: <ul style="list-style-type: none"> '1' – buy order (Buy); '2' – sell order (Sell).

* - differs from the standard FIX protocol.

4.4. Market Fundamentals

There are several aggregate values reflecting instrument market during a session, which may be changed by market events:

Market Data Entry type	MDEntryType	MDEntryPx	MDEntrySize	Market
Opening Price	4	•	–	SIFO
Closing Price	5	•	–	SIFO
Settlement Price	6	•	–	SIFO
Trading Session High Price	7	•	–	SIFO
Trading Session Low Price	8	•	–	SIFO
Trading Session VWAP Price	9	•	–	SIFO
Cumulative Trade Volume	B	•	•	SIFO

Market Data Entry type	MDEntryType	MDEntryPx	MDEntrySize	Market
Open Interest	C	–	•	FO
Market capitalization	S	•	–	I
Total bid volume	V	–	•	FO
Total offer volume	W	–	•	FO

Markets:

- **F** – Futures,
- **O** – Options,
- **I** – Indexes,
- **S** – Securities, Bonds, Commodities.

Exceptions from FIX standards:

- Trade Volume messages contain:
 - **MDEntrySize** – daily number of shares or contracts traded (standard FIX meaning);
 - **MDEntryPx** – accumulated daily turnover expressed in the currency of the instrument;
- Open Interest messages contain:
 - **MDEntrySize** – number of shares or contracts;
- Market capitalization messages contain:
 - **MDEntryPx** – index equities capitalization;
- Total Bid Volume/Offer Volume contain:
 - **MDEntrySize** – total number of shares or contracts.

4.5. Empty book message (MDEntryType = J)

The message Empty Book obliges a client to delete data on a certain instrument.

The field **SecurityID** contains the instrument ID.

The field **ExchangeTradingSessionID** remains empty.

The message is sent on session's close, or on technical breaks via the main streams (incr): FUT-BOOK-1, FUT-BOOK-5, FUT-BOOK-20, FUT-BOOK-50, OPT-BOOK-1, OPT-BOOK-5, OPT-BOOK-20, OPT-BOOK-50, SPOT-BOOK-1, SPOT-BOOK-20.

The stream Recovery (snap): FUT-BOOK-1, FUT-BOOK-5, FUT-BOOK-20, FUT-BOOK-50, OPT-BOOK-1, OPT-BOOK-5, OPT-BOOK-20, OPT-BOOK-50, SPOT-BOOK-1, SPOT-BOOK-20 transmits the message **J (Empty book)** in case there is no orders added on the certain instrument.

5. Stream of anonymous orders and trades

The service provides the following operations:

- distributing the entire order book using protocol UDP Multicast;
- distributing active orders snapshot book using protocol UDP Multicast;
- rendering service TCP Recovery

The service distributes messages using protocol FAST (see sec. 5.2); all messages are coded according to the templates **OrdersLog**, **OrdersBook** (see sec. 5.3).

5.1. Application architecture

5.1.1. Streams

The service distributes data as follows:

- the main streams distribute messages Market Data - Incremental Refresh (X) as incremental refreshes for entire order book;
- the data refresh streams distribute messages Market Data - Snapshot / Full Refresh (W) as actual snapshots containing active orders on instruments;
- the history of entire order book refreshes for the current trading session is available via the missed messages request session (TCP Recovery (Historical Replay)).

The following FIX messages are supported:

- **Market Data Request** - request for missed data;
- **Market Data - Snapshot/ Full Refresh** – active orders snapshot;
- **Market Data - Incremental Refresh** – incremental refresh of entire order book.

5.1.2. Messages fragmentation

The message fragmentation is used when sending data via:

- main streams (Incremental);
- recovery streams (Snapshot).

You can find more information about message fragmentation above (see sec. 3.2.6). Also, the first message in an active orders snapshot contains the tag **7944-RouteFirst** with value **1**.

5.1.3. Instrument IDs

The unique instrument IDs from the trading system SPECTRA are transmitted in fields **SecurityID** of messages **Market Data - Incremental Refresh** and **Market Data - Snapshot/ Full Refresh**.

5.2. Message templates

There are two certain message templates used for distributing the entire order book data:

- **OrdersLog** - see sec. 5.2.1
- **OrdersBook** - see sec. 5.2.2

5.2.1. OrdersLog

This template is used for data refresh purpose. Also, it is used by the service TCP Recovery.

```
<template name="OrdersLog" id="14">
  <string name="ApplVerID" id="1128">
    <constant value="9"/>
  </string>
  <string name="MessageType" id="35">
    <constant value="X"/>
  </string>
  <string name="SenderCompID" id="49">
    <constant value="MOEX"/>
  </string>
  <uInt32 name="MsgSeqNum" id="34"/>
  <uInt64 name="SendingTime" id="52"/>
  <uInt32 name="LastFragment" id="893"/>
  <sequence name="MDEntries">
```

```

<length name="NoMDEntries" id="268"/>
<uInt32 name="MDUpdateAction" id="279"/>
<string name="MDEntryType" id="269"/>
<int64 name="MDEntryID" id="278" presence="optional"/>
<uInt64 name="SecurityID" id="48" presence="optional"/>
<uInt32 name="SecurityIDSource" id="22">
  <constant value="8"/>
</uInt32>
<uInt32 name="RptSeq" id="83" presence="optional"/>
<uInt32 name="MDEntryDate" id="272" presence="optional"/>
<uInt64 name="MDEntryTime" id="273"/>
<decimal name="MDEntryPx" id="270" presence="optional"/>
<int64 name="MDEntrySize" id="271" presence="optional"/>
<decimal name="LastPx" id="31" presence="optional"/>
<int64 name="LastQty" id="32" presence="optional"/>
<int64 name="TradeID" id="1003" presence="optional"/>
<uInt32 name="ExchangeTradingSessionID" id="5842" presence="optional"/>
<int64 name="MDFlags" id="20017" presence="optional"/>
<uInt64 name="Revision" id="20018" presence="optional"/>
</sequence>
</template>

```

5.2.2. OrdersBook

This template is used for distributing snapshots.

```

<template name="OrdersBook" id="15">
  <string name="ApplVerID" id="1128">
    <constant value="9"/>
  </string>
  <string name="MessageType" id="35">
    <constant value="W"/>
  </string>
  <string name="SenderCompID" id="49">
    <constant value="MOEX"/>
  </string>
  <uInt32 name="MsgSeqNum" id="34"/>
  <uInt64 name="SendingTime" id="52"/>
  <uInt32 name="LastMsgSeqNumProcessed" id="369"/>
  <uInt32 name="RptSeq" id="83" presence="optional"/>
  <uInt32 name="LastFragment" id="893"/>
  <uInt32 name="RouteFirst" id="7944"/>
  <uInt32 name="ExchangeTradingSessionID" id="5842"/>
  <uInt64 name="SecurityID" id="48" presence="optional"/>
  <uInt32 name="SecurityIDSource" id="22">
    <constant value="8"/>
  </uInt32>
  <sequence name="MDEntries">
    <length name="NoMDEntries" id="268"/>
    <string name="MDEntryType" id="269"/>
    <int64 name="MDEntryID" id="278" presence="optional"/>
    <uInt32 name="MDEntryDate" id="272" presence="optional"/>
    <uInt64 name="MDEntryTime" id="273"/>
    <decimal name="MDEntryPx" id="270" presence="optional"/>
    <int64 name="MDEntrySize" id="271" presence="optional"/>
    <int64 name="TradeID" id="1003" presence="optional"/>
    <int64 name="MDFlags" id="20017" presence="optional"/>
  </sequence>
</template>

```

5.3. Messages Market Data - Incremental Refresh (X)

5.3.1. Adding a new order

Adds an order into the pool of active orders.

Tag	Field	Mandatory	Details
279	MDUpdateAction	Y	Incremental refresh type: "0" (New).
269	MDEntryType	Y	Record type: <ul style="list-style-type: none"> 0 - Bid;

Tag	Field	Mandatory	Details
			• 1 - Ask.
278	MDEntryID	Y	Order ID
48	SecurityID	Y	Instrument numeric code.
83	RptSeq	Y	Incremental refresh sequence number.
272	MDEntryDate	N	Record last change date. Value NULL in field MDEntryDate indicates the current date.
273	MDEntryTime	Y	Record last change time. Format: HHMMSSssssssssss .
270	MDEntryPx	Y	Order price.
271	MDEntrySize	Y	Volume, quantity.
31	LastPx	N	Is missing
32	LastQty	N	Is missing
1003	TradeID	N	Is missing
5842	ExchangeTradingSessionID	Y	Trading session ID
20017	MDFlags	Y	<p>The field is a bit mask:</p> <ul style="list-style-type: none"> • 0x01 Day order • 0x02 IOC order • 0x04 OTC order • 0x10 Client collateral was not checked while adding order • 0x1000 End-of-transaction bit • 0x100000 The record results from replacing the order • 0x200000 The record results from cancelling the order • 0x400000 The record results from mass cancelling • 0x20000000 Flag of cancelling the left balance of the order because of a cross-trade • 0x00080000 FOK order • 0x100000000 The record results from cancelling an order via 'Cancel on Disconnect' service.
20018	Revision	Y	Service field (replication system)

5.3.2. Deleting an order

Deletes an order from the pool of active orders according to the order's ID.

Tag	Field	Mandatory	Details
279	MDUpdateAction	Y	Incremental refresh type: "2" (Delete).
269	MDEntryType	Y	<p>Record type:</p> <ul style="list-style-type: none"> • 0 - Bid; • 1 - Ask.
278	MDEntryID	Y	Order ID
48	SecurityID	Y	Instrument numeric code.
83	RptSeq	Y	Incremental refresh sequence number.
272	MDEntryDate	N	Record last change date. Value NULL in field MDEntryDate indicates the current date.
273	MDEntryTime	Y	Record last change time. Format: HHMMSSssssssssss .
270	MDEntryPx	Y	Order price.
271	MDEntrySize	Y	Volume to delete.
31	LastPx	N	Is missing
32	LastQty	N	Is missing
1003	TradeID	N	Is missing

Tag	Field	Mandatory	Details
5842	ExchangeTradingSessionID	Y	Trading session ID
20017	MDFlags	Y	<p>The field is a bit mask:</p> <ul style="list-style-type: none"> • 0x01 Day order • 0x02 IOC order • 0x04 OTC order • 0x10 Client collateral was not checked while adding order • 0x1000 End-of-transaction bit • 0x100000 The record results from replacing the order • 0x200000 The record results from cancelling the order • 0x400000 The record results from mass cancelling • 0x20000000 Flag of cancelling the left balance of the order because of a cross-trade • 0x00080000 FOK order • 0x100000000 The record results from cancelling an order via 'Cancel on Disconnect' service.
20018	Revision	Y	Service field (replication system)

5.3.3. Partial order matching

When an order is partly matched into a trade, its attributes in the pool of active order change.

Tag	Field	Mandatory	Details
279	MDUpdateAction	Y	Incremental refresh type: "1" (Change).
269	MDEntryType	Y	<p>Record type:</p> <ul style="list-style-type: none"> • 0 - Bid • 1 - Ask
278	MDEntryID	Y	Order ID
48	SecurityID	Y	Instrument numeric code.
83	RptSeq	Y	Incremental refresh sequence number.
272	MDEntryDate	N	Record last change date. Value NULL in field MDEntryDate indicates the current date.
273	MDEntryTime	Y	Record last change time. Format: HHMMSSssssssss .
270	MDEntryPx	Y	Order price.
271	MDEntrySize	Y	Instrument units left in order
31	LastPx	Y	Matched trade price.
32	LastQty	Y	Trade volume.
1003	TradeID	Y	Trade ID
5842	ExchangeTradingSessionID	Y	Trading session ID
20017	MDFlags	Y	<p>The field is a bit mask:</p> <ul style="list-style-type: none"> • 0x4 – OTC-trade; • 0x8 – position transfer trade; • 0x20 – option exercise trade; • 0x80 – flag of instrument expiration (exercise for futures, expiration for options); • 0x1000 - the last record in transaction; • 0x20000 – repo trade; • 0x40000 – series of trade;

Tag	Field	Mandatory	Details
			<ul style="list-style-type: none"> • 0x800000 – option expiration trade; • 0x2000000 – clearing session trade; • 0x4000000 – negotiated trade; • 0x8000000 – multi-leg trade; • 0x10000000 – trade on non-delivery; • 0x40000000 - futures exercise trade.
20018	Revision	Y	Service field (replication system)

5.3.4. Full order matching

After an order is fully matched into a trade, this order will be deleted from the pool of active orders.

Tag	Field	Mandatory	Details
279	MDUpdateAction	Y	Incremental refresh type. If deleted, then "2" (Delete).
269	MDEntryType	Y	Record type: <ul style="list-style-type: none"> • 0 - Bid • 1 - Ask
278	MDEntryID	Y	Order ID
48	SecurityID	Y	Instrument numeric code.
83	RptSeq	Y	Incremental refresh sequence number.
272	MDEntryDate	N	Record last change date. Value NULL in field MDEntryDate indicates the current date.
273	MDEntryTime	Y	Record last change time. Format: HHMMSSssssssss .
270	MDEntryPx	Y	Order price.
271	MDEntrySize	N	Is missing
31	LastPx	Y	Matched trade price.
32	LastQty	Y	Trade volume.
1003	TradeID	Y	Trade ID
5842	ExchangeTradingSessionID	Y	Trading session ID
20017	MDFlags	Y	The field is a bit mask: <ul style="list-style-type: none"> • 0x4 – OTC-trade; • 0x8 – position transfer trade; • 0x20 – option exercise trade; • 0x80 – flag of instrument expiration (exercise for futures, expiration for options); • 0x1000 - the last record in transaction; • 0x20000 – repo trade; • 0x40000 – series of trade; • 0x800000 – option expiration trade; • 0x2000000 – clearing session trade; • 0x4000000 – negotiated trade; • 0x8000000 – multi-leg trade; • 0x10000000 – trade on non-delivery; • 0x40000000 - futures exercise trade.
20018	Revision	Y	Service field (replication system)

5.3.5. Technical trades

Tag	Field	Mandatory	Details
279	MDUpdateAction	Y	Incremental refresh type: "0" (New).
269	MDEntryType	Y	Record type: <ul style="list-style-type: none"> • 0 - Bid • 1 - Ask
278	MDEntryID	Y	Order ID
48	SecurityID	Y	Instrument numeric code.
83	RptSeq	Y	Incremental refresh sequence number.
272	MDEntryDate	N	Record last change date. Value NULL in field MDEntryDate indicates the current date.
273	MDEntryTime	Y	Record last change time. Format: HHMMSSssssssss.
270	MDEntryPx	N	Is missing.
271	MDEntrySize	N	Is missing
31	LastPx	Y	Matched trade price.
32	LastQty	Y	Trade volume.
1003	TradeID	Y	Trade ID
5842	ExchangeTradingSessionID	Y	Trading session ID
20017	MDFlags	Y	The field is a bit mask: <ul style="list-style-type: none"> • 0x4 – OTC-trade; • 0x8 – position transfer trade; • 0x20 – option exercise trade; • 0x80 – flag of instrument expiration (exercise for futures, expiration for options); • 0x1000 - the last record in transaction; • 0x20000 – repo trade; • 0x40000 – series of trade; • 0x800000 – option expiration trade; • 0x2000000 – clearing session trade; • 0x4000000 – negotiated trade; • 0x8000000 – multi-leg trade; • 0x10000000 – trade on non-delivery; • 0x40000000 - futures exercise trade.
20018	Revision	Y	Service field (replication system)

5.3.6. Trade types

Trade types, created upon execution and expiration of futures and options are listed in the table below:

Operation type	Position closing trade	Position opening trade	Date and time of trades availability in reports and in the gateway
Classical execution of futures	<ul style="list-style-type: none"> • id in reports is 0, id in gateways is nonzero. • Trade price is rounded to minimal step price. • Technical trade is not a trade legally. • Signs in gateways and reports (bitmask): 0x4 (OTC- 	No.	On the execution day, in the morning.

Operation type	Position closing trade	Position opening trade	Date and time of trades availability in reports and in the gateway
	trade), 0x80 (expiration of the instrument), 0x40000000 (futures exercise trade).		
Execution of non-deliverable futures	<ul style="list-style-type: none"> id in reports is 0, id in gateways is nonzero. Trade price is rounded to the minimal price step. Technical trade is not a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x80 (expiration of the instrument), 0x40000000 (futures exercise trade). 	No.	On the execution day, in the evening.
Exercising of option	<ul style="list-style-type: none"> id in gateways is nonzero. id in reports is 0 (trade at the evening clearing session), nonzero id (trade at the intermediate clearing session). Trade price is 0. Technical trade is not a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x20 (option exercise trade). 	<ul style="list-style-type: none"> id in gateways is nonzero. id in reports is 0. Trade price is rounded to 5 places. This trade is a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x20 (option exercise trade). 	<ul style="list-style-type: none"> At the intermediate clearing session At the evening clearing session <p>Depending on time of applying the option (the next clearing session after applyinh).</p>
Expiration of option	<ul style="list-style-type: none"> id in gateways is nonzero. id in reports is 0. Trade price is 0. Technical trade is not a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x80 (expiration of the instrument), 0x800000 (option expiration trade). 	No.	On the futures execution day, in the evening.

Trades are shown as following:

Operation type	Operations info
Stock futures trade based on a negotiated order	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to the minimal price step. This trade is a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x4000000 (negotiated trade).
Stock futures trade based on a system order	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to the minimal price step. This trade is a trade legally. Signs in gateways and reports (bitmask):bits value is 0.
Stock futures option trade based on a negotiated order	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to the minimal price step. This trade is a trade legally.

Operation type	Operations info
	<ul style="list-style-type: none"> Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x4000000 (negotiated trade).
Stock futures option trade based on a system order	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to the minimal price step. This trade is a trade legally. Signs in gateways and reports (bitmask):bits value is 0.
Position transfer trade	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to 5 places. This trade is not a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x8 (position transfer trade), 0x4000000 (negotiated trade).
Technical trade based on repo negotiated order (1st leg)	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to 5 places. This trade is a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x20000 (repo trade), 0x4000000 (negotiated trade), 0x8000000 (multi-leg trade).
Technical trade based on repo negotiated order (2nd leg)	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to 5 places. This trade is a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x20000 (repo trade), 0x4000000 (negotiated trade), 0x8000000 (multi-leg trade).
Technical trade based on repo system order (1st leg)	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to 5 places. This trade is not a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x20000 (repo trade), 0x8000000 (multi-leg trade).
Technical trade based on repo system order (2nd leg)	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to 5 places. This trade is not a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x20000 (repo trade), 0x8000000 (multi-leg trade).
Technical trade based on the negotiated pseudo-repo (1st leg)	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to 5 places. This trade is a trade legally. Signs in gateways and reports (bitmask): 0x4 OTC-trade), 0x40000 (bulk of trades), 0x4000000 (negotiated trade), 0x8000000 (multi-leg trade).
Technical trade based on the negotiated pseudo-repo (2nd leg)	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to 5 places. This trade is a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x40000 (bulk of trades), 0x4000000 (negotiated trade), 0x8000000 (multi-leg trade).
Technical trade based on the system pseudo-repo (1st leg)	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero.

Operation type	Operations info
	<ul style="list-style-type: none"> Trade price is rounded to 5 places. This trade is a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x40000 (bulk of trades), 0x8000000 (multi-leg trade).
Technical trade based on the system pseudo-repo (2nd leg)	<ul style="list-style-type: none"> id in gateways and reports is unique and nonzero. Trade price is rounded to 5 places. This trade is a trade legally. Signs in gateways and reports (bitmask): 0x4 (OTC-trade), 0x40000 (bulk of trades), 0x8000000 (multi-leg trade).

5.3.7. Cleaning up active orders pool (specified trading session)

After receiving the message, all orders of the specified trading session are to be deleted on the client side.

Tag	Field	Mandatory	Details
279	MDUpdateAction	Y	Incremental refresh type: "0" (New).
269	MDEntryType	Y	Record type: "J" (Empty Book).
278	MDEntryID	N	Is missing
48	SecurityID	N	Is missing
83	RptSeq	N	Is missing
272	MDEntryDate	N	Record last change date. Value NULL in field MDEntryDate indicates the current date.
273	MDEntryTime	Y	Record last change time. Format: HHMMSSssssssss.
270	MDEntryPx	N	Is missing
271	MDEntrySize	N	Is missing
31	LastPx	N	Is missing
32	LastQty	N	Is missing
1003	TradeID	N	Is missing
5842	ExchangeTradingSessionID	Y	Trading session ID
20017	MDFlags	N	Is missing
20018	Revision	Y	Service field (replication system)

5.3.8. Cleaning up active orders pool (all trading sessions)

After receiving the message, all orders are to be deleted on the client side. After that, you should perform steps 4-7 in the section **Connect client** - see sec. 2.1.

Tag	Field	Mandatory	Details
279	MDUpdateAction	Y	Incremental refresh type: "0" (New).
269	MDEntryType	Y	Record type: "J" (Empty Book).
278	MDEntryID	N	Is missing
48	SecurityID	N	Is missing
83	RptSeq	N	Is missing
272	MDEntryDate	N	Record last change date. Value NULL in field MDEntryDate indicates the current date.
273	MDEntryTime	Y	Record last change time. Format: HHMMSSssssssss.
270	MDEntryPx	N	Is missing
271	MDEntrySize	N	Is missing
31	LastPx	N	Is missing
32	LastQty	N	Is missing
1003	TradeID	N	Is missing
5842	ExchangeTradingSessionID	N	Is missing

Tag	Field	Mandatory	Details
20017	MDFlags	N	Is missing
20018	Revision	N	Is missing

5.4. Messages Market Data - Snapshot / Full Refresh

5.4.1. Active orders snapshot

Snapshots are distributed as one or several messages (for each instrument).

Tag	Field	Mandatory	Details
<Standard Message Header>		Y	Message type 'W'
369	LastMsgSeqNumProcessed	Y	The 'MsgSeqNum' of the last message sent into incremental feed at the time of the current snapshot generation.
83	RptSeq	Y	The 'RptSeq' number of the last incremental update included in the current market data snapshot for instrument.
893	LastFragment	Y	Indicates the last message in the message group for the instrument. Allowable values: <ul style="list-style-type: none"> • 0 – not the last message • 1 – the last message
7944	RouteFirst	Y	Indicates whether the message is the first of the ones generated for the given instrument or not. Allowable values: <ul style="list-style-type: none"> • 0 – not the first message • 1 – the first message
5842	ExchangeTradingSessionId	Y	Trading session ID
48	SecurityId	Y	Instrument numeric code
22	SecurityIdSource	C	8 - Exchange Symbol
268	NoMDEntries	Y	Number of 'MDEntry' records in the current message.
=>269	MDEntryType	Y	Record type: <ul style="list-style-type: none"> • 0 - Bid • 1 - Ask
=>278	MDEntryID	Y	Order ID.
=>272	MDEntryDate	N	Record last change date.
=>273	MDEntryTime	Y	Record last change time. Format: HHMMSSssssssss .
=>270	MDEntryPx	Y	Order price.
=>271	MDEntrySize	Y	Volume, quantity.
=>1003	TradeID	C	None if no trade was matched, else contains last trade ID.
=>20017	MDFlags	Y	The field is a bit mask: <ul style="list-style-type: none"> • 0x01 Day order • 0x02 IOC order • 0x04 OTC order • 0x10 Client collateral was not checked while adding order • 0x100000 The record results from replacing the order • 0x200000 The record results from cancelling the order • 0x400000 The record results from mass cancelling • 0x20000000 Flag of cancelling the left balance of the order because of a cross-trade

Tag	Field	Mandatory	Details
			<ul style="list-style-type: none"> • 0x00080000 FOK order • 0x100000000 The record results from cancelling an order via 'Cancel on Disconnect' service.

5.4.2. When an active orders snapshot is empty

If the active orders snapshot is empty, and a trading session has not yet started, the snapshot will not be distributed. If the active orders snapshot gets empty during a trading session, it will be distributed in the form of the message below.

Tag	Field	Mandatory	Details
<Standard	Message Header>	Y	Message type 'W'
369	LastMsgSeqNumProcessed	Y	The 'MsgSeqNum' of the last message sent into incremental feed at the time of the current snapshot generation.
83	RptSeq	Y	The 'RptSeq' number of the last incremental update included in the current market data snapshot for instrument.
893	LastFragment	Y	Indicates the last message in the message group for the instrument. Allowable values: <ul style="list-style-type: none"> • 1 – the last message
7944	RouteFirst	Y	Indicates whether the message is the first of the ones generated for the given instrument or not. Allowable values: <ul style="list-style-type: none"> • 1 – the first message
5842	ExchangeTradingSessionId	Y	Trading session ID.
48	SecurityId	Y	Instrument numeric code.
22	SecurityIdSource	C	8 - Exchange Symbol
268	NoMDEntries	Y	Number of 'MDEntry' records in the current message.
=>269	MDEntryType	Y	Record type: <ul style="list-style-type: none"> • J - Empty Book
=>278	MDEntryID	N	Is missing
=>272	MDEntryDate	N	Record last change date.
=>273	MDEntryTime	Y	Record last change time. Format: HHMMSSssssssssss.
=>270	MDEntryPx	N	Is missing
=>271	MDEntrySize	N	Is missing
=>1003	TradeID	N	Is missing
=>20017	MDFlags	N	Is missing

6. TCP Recovery (Historical Replay) service limitations

The following limitations are applied to the TCP Recovery service, in order to lower the load:

Parameter	Value	Details
Maximum active connections, per market, per instance, per IP address	2	You can establish no more than indicated number active TCP connection from single IP address. An attempt to make more connections will be rejected
Maximum connections count, per market, per instance, per day, per IP address	1000	You can make no more than indicated number of tcp connections per IP address per day. Extra connection attempts will be rejected
Maximum number of messages to request	1000	TCP replay request is rejected if a number of requested messages is greater than indicated value
Marketdata request timeout, seconds	1	Connection is terminated with logout message if marketdata request is not received within indicated number of seconds since logon message. TCP session is terminated if no confirming logout is received after server-side logout.