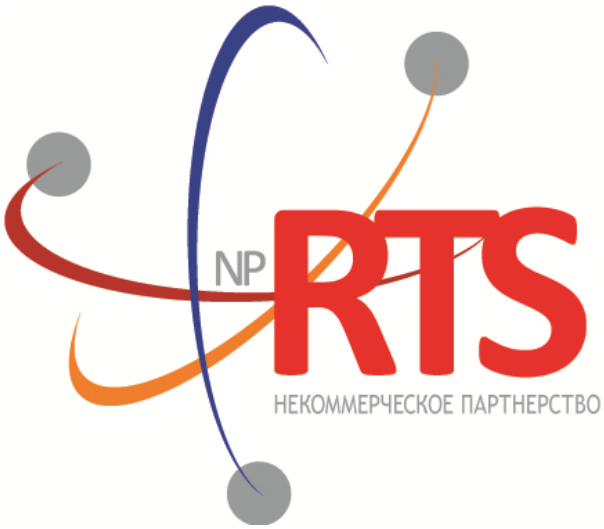


ADVANCED
TRADING & CLEARING
PLATFORM

# Native Protocol Risk Management Gateway

**Interface version 38**

**Document version 1.15.4**

**23 May 2022**

# Revision history

**Version 1.15.4 May 23, 2022**

1. Added description in Section 5.2.3.
2. Improved description in document.

**Version 1.15.3 December 21, 2021**

1. Updated the list of liquidity pools in Section 3.6 Liquidity pool identifiers.
2. Clarified description of the market_id field in the instrument component.
3. Clarified description of the topic_firstseq field in the TopicReject message in Section 5.1.15.

**Version 1.15.2 October 25, 2021**

Fixed a bug in the name of the segregation_type field (instead of segredation_type) in the ClearingAccount message.

**Version 1.15.1 October 1, 2021**

1. Added security type "Exchange traded bond" to the security_type field in the Bond message.
2. Added the table of correspondence between designations of instrument categories in the trading system and Interfax.

**Version 1.15.0 August 4, 2021**

1. The categoryProhibition field is added to the Client message in the section Topic of trading member references.
2. The category field is added to the Instrument message in the section Topic of instrument references.
3. Added error code 1405 in Appendix A.
4. Updated the list of liquidity pools in Section 3.6 Liquidity pool identifiers.

**Version 1.14.2 February 4, 2021**

Description of the Section 4.8 was updated — added description of keys from the SysProperties message.

**Version 1.14.1 December 24, 2020**

Description of the security_type field in the Issue message was updated.

**Version 1.14.0 September 2, 2020**

1. Added Section 5.2.3 describing YieldConversionRequest and YieldConversionReport messages.
2. Description of Section 5.1.6 was updated.

**Version 1.13.8 June 2, 2020**

Added warning about inexpedience using the parameters of a specific trading mode for setting up the trading system to the description of TradeModes message.

**Version 1.13.7 April 17, 2020**

1. Description about the trading system's behavior in case a client uses other, than standard method of datafeeds processing was added to Section 1.3.
2. The allowed range of values for topic_seq and topic_seqend parameter has been changed in Section 5.1.12.

**Version 1.13.6 January 24, 2020**

1. Added Section 5.1.4 describing client's send rate limit of the session level messages.
2. Added Section 5.2.1 describing client's send rate limit of the application level messages.

**Version 1.13.5 October 07, 2019**

Added description of system information topic SysProperties.

**Version 1.13.4 September 20, 2019**

Added liquidity pool identifier for Hong-Kong Exchange (1019).

**Version 1.13.3 July 25, 2019 года**

Section 5.1.3 renamed to "Keeping section in active state". Description of active session state maintenance is updated.

**Version 1.13.2 February 1, 2019**

Added value 4 (MemberTariff) to the fee_schema field of the Instrument message.

**Version 1.13.1 December 14, 2018**

1. Document structure was changed.
2. Terminology related to topic data transmission was changed.
3. The name of Trade message of topic of clearing transactions and transfers was changed to ClearingTrade .
4. Description of key fields in topic messages was added.
5. Name of topic_header component was changed to header.

**Version 1.13.0 November 3, 2017**

1. The "Service overview" section has been added.
2. The "Changing risk parameters" section has been removed.
3. The msgid field value changed for the TradeModes message.
4. The over_the_counter field added to the TradeModes message.
5. The msgid field value changed for the Instrument message.
6. The borrowing_status field added to the Instrument message.
7. The trading_status field of the TradingInstrumentStatus message renamed to status.
8. Terminology changes.
9. Error codes added.

**Version 1.12.0 November 30, 2016**

1. The markets field is added to the Period component.
2. The msgid value changed in the Instrument message.
3. New fields (order_id, exch_orderid, exec_market, and dir) are added and the msgid value is changed in the ClearingTrade message.

**Version 1.11.0 March 23, 2016**

The Market message is added to the Instruments topic.

**Version 1.10.1 February 16, 2016**

Restrictions on LimitRequest are clarified.

# Table of Contents

# List of Tables

# 1. Service overview

## 1.1. Data topics

The risk management gateway provides access to data about trading members and enables management of clients' limits.

The gateway currently provides the following topics:

1. Clearing trades and transfers.
2. Clearing positions.
3. Members' funds.
4. Risk rates.
5. Risk parameters.
6. Trading members’ references.
7. Instrument references.
8. System information.

Messages of each topic are numbered consecutively in the `topic_seq` field. The numbering of messages sent to client may be discontinuous as client receives data in accordance with login access rights.

## 1.2. Broadcast modes

Topics can broadcast data in two modes — **snapshot** and/or **snapshot with subsequent updates**.

A snapshot is aggregation of all current data, e.g. clearing positions list, transmitted at a specified frequency.

Updates are separate messages generated and transmitted to the client when an event occurs.

During a period of inactivity in an update feed the system sends a `Heartbeat` to acknowledge connection. If messages are not transmitted for a longer period, there is either a transmission delay or absence of connection.

## 1.3. Algorithm of receiving and processing topic data

If you want to connect to a topic with snapshots and updates, it is recommended to connect in mode of snapshot with subsequent updates. First, you should receive a complete snapshot, then start recording incoming updates. If an update has been lost, it can be requested by `ResendRequest`. If messages' recovery takes significant amount of time, it is recommended to request the snapshot instead of attempting to recover lost updates.

When snapshot is complete you should record the updates. Updates can replace or replenish earlier data, depending on the topic. For topics with replacement there are identifiers of updated data - `keys`. The `keys` are fields values of topic messages and are indicated in header of tables in section 4.

Table 1. Features of snapshot and updates

| Topic | Update | | Snapshot |
|---|---|---|---|
| | Replenishment | Replacement | |
| Clearing trades and transfers<br>Risk rates | ✓ | | Messages history since the start of the trading day |
| Clearing positions<br>Members' funds<br>Risk parameters | ✓ | ✓ | An aggregation of all current data |
| Instruments<br>Trading members’ references | | ✓ | |

## 1.3.1. Example for Clearing trades and transfers topic

*The trading system may deny execution of client's request with an error message if the client's requests do not follow the algorithm specified below.*

An updates from Clearing trades and transfers **replenish** earlier data.

1. Send the `TopicRequest` message with `Topic=Trades.Trade` and `mode=1` to the gateway.
2. The `TopicRequest` will result in the following message sequence:
   - `TopicReport` (seq=0, status=1, marker=0 (START), topic_lastseq=100, topic_lastseqsent=0);
   - `ClearingTrade` (seq=1, topic_seq=11);
   - `ClearingTrade` (seq=2, topic_seq=57);
   - `ClearingTrade` (seq=3, topic_seq=32);
   - `ClearingTrade` (seq=4, topic_seq=90);
   - `TopicReport` (seq=0, status=1, marker=2 (SLICE_END), topic_lastseq=100, topic_lastseqsent=100).

   The `ClearingTrade` messages have gaps between `topic_seq` values, because the `Heartbeat` messages were received between `ClearingTrade` messages.
3. Wait updates `ClearingTrade` of topic. For example, you received the following updates:
   - `ClearingTrade` (seq=5, topic_seq=110);
   - `ClearingTrade` (seq=6, topic_seq=117);

   Add received updates to the end of snapshot.

## 1.3.2. Example for Clearing positions topic

*The trading system may deny execution of client's request with an error message if the client's requests do not follow the algorithm specified below.*

An updates from Clearing positions topic **replace** earlier data.

1. Send the `TopicRequest` message with `Topic=Pos.PositionUpdate`, `mode=1` to the gateway.
2. The `TopicRequest` will result in the following message sequence:
   - `TopicReport` (seq=0, status=1, marker=0 (START), topic_lastseq=567, topic_lastseqsent=0);
   - `PositionUpdate` (seq=1, topic_seq=424, entity=entity1, balance_id=1000);
   - `PositionUpdate` (seq=2, topic_seq=318, entity=entity2, balance_id=1000);
   - `PositionUpdate` (seq=3, topic_seq=342, entity=entity1, balance_id=1001);
   - `PositionUpdate` (seq=4, topic_seq=383, entity=entity3, balance_id=1001);
   - `TopicReport` (seq=0, status=1, marker=2 (SLICE_END), topic_lastseq=567, topic_lastseqsent=567).

   The `PositionUpdate` messages have gaps between `topic_seq` values, because the `Heartbeat` messages were received between `PositionUpdate` messages.
3. Wait updates `PositionUpdate` of topic. For example, you received the following updates:
   - `PositionUpdate` (seq=5, topic_seq=581, entity=entity1, balance_id=1001);
   - `PositionUpdate` (seq=6, topic_seq=601, entity=entity1, balance_id=1000);
   - `PositionUpdate` (seq=7, topic_seq=594, entity=entity3, balance_id=1001);
4. Compare keys values of snapshot and each updates with number `topic_seq>topic_lastseqsent` (the keys of Clearing positions topic are `entity` and `balance_id`):
   - If the values are equal (updates with `seq=6`, `seq=7`), you should replace the snapshot message with an update.
   - If the values aren't equal (updates with `seq=5`, `seq=8`), you should replenish the snapshot with an update.

# 2. Interaction with gateway

## 2.1. Data request

To request data, the client should send TopicRequest to the trading system gateway with topic identifier `topic`, range of requested messages `topic_seq`, `topic_seqend` and `mode` of data receipt.

In response to valid request, the client will receive notification TopicReport and after that should expect data messages. At the end of snapshot transmission, the client will receive `TopicReport`.

If a request contains invalid values, is duplicate or cannot be executed, it will be rejected by TopicReject.

*If you want to request a new topic, wait until you have received all messages, related to the previous topic request, to avoid network overload.*

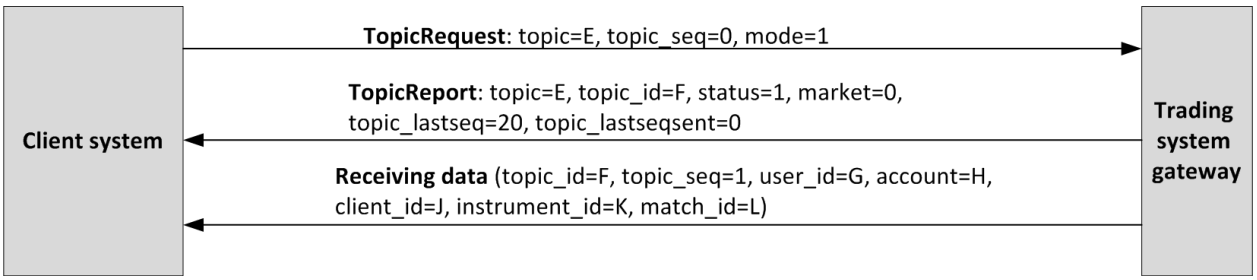


Figure 1. Request and receive data

## 2.2. Updates canceling and resuming

To stop receiving updates, the client should send TopicCancel to the trading system gateway specifying topic identifier `topic` or `topic_id`.

In response to valid request, the client will receive notification TopicReport and updates will be canceled; client may continue receiving messages with data for some time after notification.

If request contains invalid values or cannot be executed, it will be rejected by message TopicReject.

Updates are automatically canceled at disconnection.

After updates canceling, the client may request updates again, sending TopicRequest and specifying the `topic_seq` number to be subsequent number of the last received message.

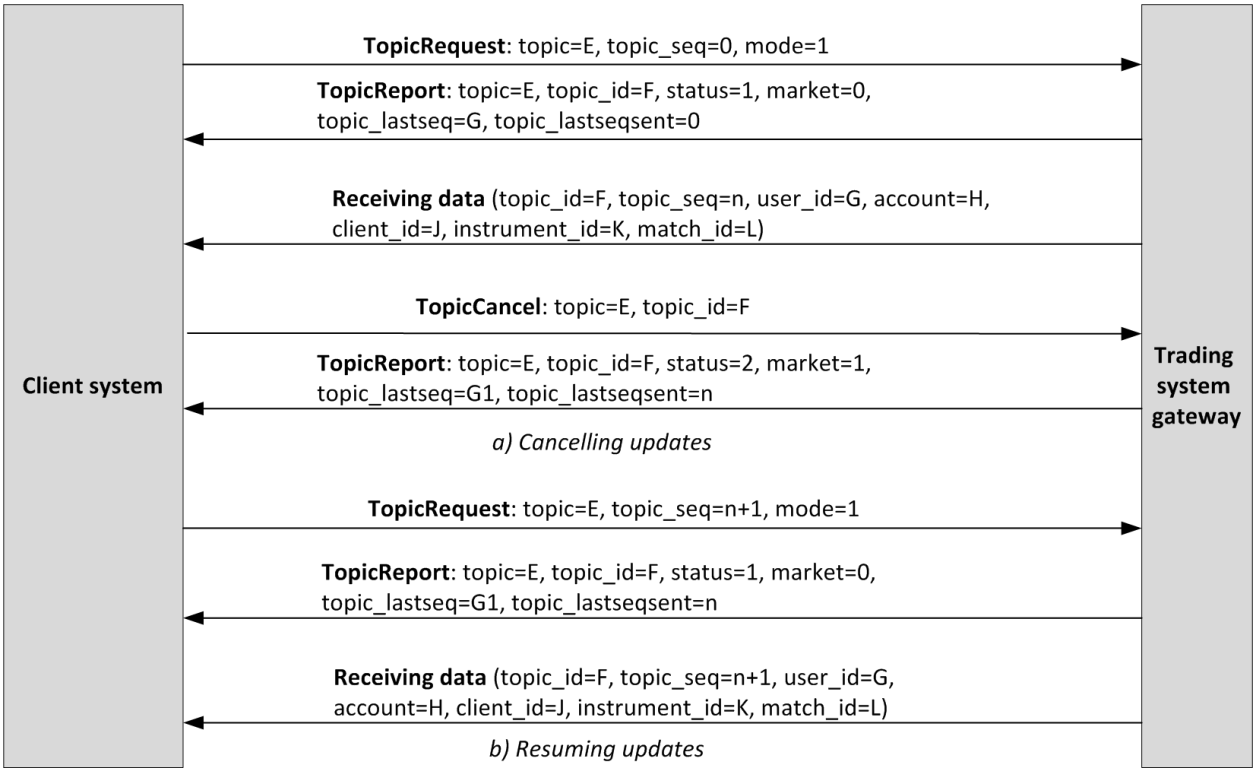


Figure 2. Canceling and resuming updates

# 2.3. Limit change

To change client instrument limits, the client should send LimitRequest to the trading system gateway. The request should contain the identifier of balance instrument, which limit must be changed, in the `balance_id` field.

In response to a valid request, the trading system will send LimitReport to the client.

A `LimitRequest` containing invalid field values will be rejected by RejectReport.

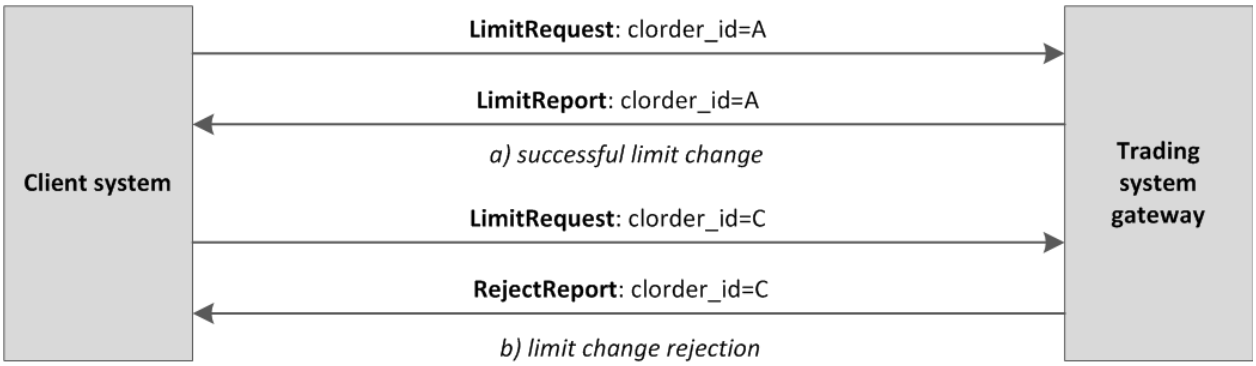


Figure 3. Request for limit change

# 3. Protocol overview

## 3.1. Data types

The trading system uses little-endian byte order (same as in x86 processor); the client shall use same.

`asciiN` is an alphanumeric string of *N*-byte length; the unused part should be filled with zero bytes.

`charN+1` is a UFT-8 encoded string of *N+1*-byte length. The last byte is the end of line character and so the available length is *N*; the unused part should be filled with zero bytes.

`dec2` is an eight-byte integer representing a fraction multiplied by $10^2$.

`dec8` is an eight-byte integer representing a fraction multiplied by $10^8$.

`decn` is a nine-byte sequence; the first eight bytes are an integer representing a fraction multiplied by $10^n$ and the last byte is *n*. Its value should be within the range from 0 to 8.

`intN` is an *N*-byte integer.

`time4` is a four-byte integer representing the Unix time in seconds, i.e. the number of seconds since 1 January 1970.

`time8n` is an eight-byte integer representing the Unix time in nanoseconds, i.e. the number of nanoseconds since 1 January 1970.

`time8m` is an eight-byte integer representing the Unix time in milliseconds, i.e. the number of milliseconds since 1 January 1970. If a field of this datatype conveys a date, the value part representing hours, minutes, seconds and milliseconds should be neglected, i.e. that is to use an integer value (rounded down) of division by 86 400 000.

## 3.2. Message format

A native protocol message is a sequence of field values in a strict order. Each message starts with the `frame` header; this three-field component includes message size, message type, and sequence number. The message size is the length of the whole message, except for the frame header, in bytes. The size is constant for all message types which don not include any repeating component or field.

A message is transmitted in a network packet as a sequence of bytes.

## 3.3. Common components of messages

Table 2. Format of component `frame`: length 12 bytes

| Field | Datatype | Description |
|---|---|---|
| size | int2 | Message length in bytes, excluding the `frame` header |
| msgid | int2 | Message type |
| seq | int8 | Application message sequence number |

Table 3. Format of component `instrument`: length 6 bytes

| Field | Datatype | Description |
|---|---|---|
| market_id | int2 | Liquidity pool ID. For a description of values, refer to Section 3.6 |
| instrument_id | int4 | Trading instrument ID |

Table 4. Format of component `user_header`: length 20 bytes

| Field | Datatype | Description |
|---|---|---|
| clorder_id | ascii20 | Client order ID |

Table 5. Format of component `gate_header`: length 46 bytes

| Field | Datatype | Description |
|---|---|---|
| system_time | time8n | Client request processing time |
| source_id | int2 | Message source (for values please refer to section 3.5) |
| clorder_id | ascii20 | Client order ID |
| user_id | ascii16 | Login, client gateway ID |

Table 6. Format of component `header`: length 22 bytes

| Field | Datatype | Description |
|---|---|---|
| topic_id | int4 | Numerical ID of topic |
| topic_seq | int8 | Message sequence number in topic |
| system_time | time8n | Message generation time |
| source_id | int2 | Message source (for values please refer to section 3.5) |

Table 7. Format of component `account`: length 36 bytes

| Field | Datatype | Description |
|---|---|---|
| member_id | int4 | Trading member ID |
| account | ascii16 | Clearing account ID |
| client_id | ascii16 | Client code ID |

Table 8. Format of component `account_entity`: length 21 bytes

| Field | Datatype | Description |
|---|---|---|
| member_id | int4 | Trading member ID |
| entity_id | ascii16 | Entity ID |
| entity_type | int1 | Type of entity. Values:<br>• `0` (CLIENT): client code;<br>• `1` (CLIENT_GROUP): group of client codes;<br>• `2` (CLEAR_ACC): analytical clearing account;<br>• `3` (PRINCIPAL_CLEAR_ACC): clearing account;<br>• `4` (TRADE_MEMBER): trading member |

Table 9. Format of component `deal`: length 20 bytes

| Field | Datatype | Description |
|---|---|---|
| deal_price | dec8 | Trade price |
| deal_id | int8 | Trade ID assigned by liquidity pool |
| amount | int4 | Trade volume |

Table 10. Format of component `otccodes`: length 32 bytes

| Field | Datatype | Description |
|---|---|---|
| initiator_party | ascii16 | Negotiated order sender ID |
| ctrparty | ascii16 | Negotiated order recipient ID |

Table 11. Format of component `clr_deal`: length 85 bytes

| Field | Datatype | Description |
|---|---|---|
| deal_id | int8 | Trade ID |
| clr_deal_id | int8 | Clearing trade ID |
| traded_balance_id | int8 | Trading balance instrument ID |
| measuring_balance_id | int8 | Balance instrument ID for measuring price |
| clr_deal_price | dec8 | Clearing trade price |
| amount | decn | Clearing trade volume |
| volume | decn | Volume of trade, part one, in settlement currency |
| dir | int1 | Side of trade. Values:<br>• `1` (Buy): deposit;<br>• `2` (Sell): withdrawal |
| fee | decn | Trade fee. The number can contain up to ten decimals |
| accr_interest | decn | Accrued coupon income |
| flags | int8 | Flags depending on the market |

Table 12. Format of component `clr_repo_deal`: length 126 bytes

| Field | Datatype | Description |
|---|---|---|
| deal_id | int8 | Trade ID |
| clr_deal_id | int8 | Clearing trade ID |
| traded_balance_id1 | int8 | Trading balance instrument ID. 1st part of the repo |
| measuring_balance_id1 | int8 | Balance instrument ID for measuring price. 1st part of the repo |
| traded_balance_id_back | int8 | Trading balance instrument ID. 2nd part of the repo |
| measuring_balance_id_back | int8 | Balance instrument ID for measuring price. 2nd part of the repo |
| repo_rate | dec8 | Repo rate |
| price | dec8 | Price of repo trade, part one |
| amount | decn | Volume of trade in asset units |
| volume | decn | Volume of trade, part one, in settlement currency |
| buyback_volume | decn | Buyback volume |

| Field | Datatype | Description |
|---|---|---|
| buyback_price | dec8 | Buyback price |
| dir | int1 | Side of trade. Values:<br>• `1` (Buy): deposit;<br>• `2` (Sell): withdrawal |
| fee | decn | Trade fee. The number can contain up to ten decimals |
| accr_interest | decn | Accrued coupon income |
| flags | int8 | Flags depending on the market |

Table 13. Format of component `coupon_payment`: length 16 bytes

| Field | Datatype | Description |
|---|---|---|
| date | time8m | Date of payment |
| value | dec8 | Amount of payment |

Table 14. Format of component `ExchangeAccount`: length 36 bytes

| Field | Datatype | Description |
|---|---|---|
| market_id | int2 | Liquidity pool ID (for a description of values, refer to Section 3.6) |
| type | int2 | Method to identify a clearing account. Values:<br>• `1` (Standart): standard;<br>• `2` (External): extended, with the use of `codeExtra` |
| account | ascii16 | Clearing account ID at liquidity pool |
| code_extra | ascii16 | Additional ID of clearing account at liquidity pool |

Table 15. Format of component `ExchangeClient`: length 18 bytes

| Field | Datatype | Description |
|---|---|---|
| market_id | int2 | Liquidity pool ID (for a description of values, refer to Section 3.6) |
| client_name | ascii16 | Client code ID at liquidity pool |

Table 16. Format of component `ExchangeInstrument`: length 61 bytes

| Field | Datatype | Description |
|---|---|---|
| instrument | [instrument] | Component specifying trading instrument |
| code_group | char16+1 | Market section |
| code | char16+1 | Instrument ticker |
| code_extra | char16+1 | Instrument code |
| status | [instrument_status] | Current status of trading instrument |

Table 17. Format of component `extra_data`: length 11 bytes

| Field | Datatype | Description |
|---|---|---|
| type | int2 | Parameter type |
| value | decn | Parameter value |

Table 18. Format of component `instrument_status`: length 4 bytes

| Field | Datatype | Description |
|---|---|---|
| trading_status | int1 | Current status of trading instrument. Values:<br>• `2` (HALT): trading is halted;<br>• `17` (TRADING): trading in progress;<br>• `18` (NO_TRADING): no trading;<br>• `102` (CLOSE): trading during closing auction;<br>• `103` (CLOSE_PERIOD): trading during close period;<br>• `107` (DISCRETE_AUCTION): trading during discrete auction;<br>• `118` (OPEN): trading during opening auction;<br>• `120` (FIXED_PRICE_AUCTION): trading at closing auction price |
| suspend_status | int1 | Reserved field. To be filled with null byte |
| routing_status | int1 | Reserved field. To be filled with null byte |
| reason | int1 | Reserved field. To be filled with null byte |

Table 19. Format of component `t_OTCCode`: length 18 bytes

| Field | Datatype | Description |
|---|---|---|
| code | ascii16 | Negotiated trading ID |
| market_id | int2 | Liquidity pool ID (for a description of values, refer to Section 3.6) |

Table 20. Format of component `Period`: length 30 bytes

| Field | Datatype | Description |
|---|---|---|
| start | time8m | Start timestamp |
| finish | time8m | End timestamp |
| mode | int2 | Type of auction. Values:<br>• `0` (ProRata): pro rata two-way anonymous auction;<br>• `1` (Parity): parity two-way anonymous auction;<br>• `2` (TimePriority): time priority anonymous auction;<br>• `3` (Address): negotiated trading;<br>• `4` (OpenAuction): opening auction;<br>• `5` (CloseAuction): closing auction;<br>• `6` (NoTrade): no trading;<br>• `7` (ExtClose): closing auction at liquidity pool |
| currency_id | int4 | Currency ID of traded instrument |

| Field | Datatype | Description |
|---|---|---|
| underlying_offset | int2 | Offset of the first `underlying` entry from the beginning of this field |
| underlying_count | int2 | Number of the `underlying` group entries |
| markets_offset | int2 | Offset of the first `markets` entry from the beginning of this field |
| markets_count | int2 | Number of the `markets` group entries |
| > underlying | [Underlying] | Component for specifying the lot volume of a trading instrument within a period |
| > markets | int2 | List of available liquidity pools (for a description of values, refer to section 3.6) |

Table 21. Format of component `transfer`: length 43 bytes

| Field | Datatype | Description |
|---|---|---|
| transfer_id | int8 | Transfer ID assigned by trading system |
| balance_id | int8 | Balance instrument ID |
| sess_id | int4 | Clearing session ID |
| clearing_id | int4 | Clearing ID preceding the transfer |
| dir | int1 | Side of transfer. Values:<br>• `1` (Buy): deposit;<br>• `2` (Sell): withdrawal |
| transfer_type | int1 | Type of transfer. Values:<br>• `1` (Trade): trade result;<br>• `2` (LimitChange): change in the amount of positions |
| flags | int8 | Parameters of transfer. Values:<br>• `0x100` (not to verify a non-increase in arrears (“hard” withdrawal));<br>• `0x200` (transfer by member’s command);<br>• `0x1000` (deposit-withdrawal by client’s order);<br>• `0x2000` (deposit-withdrawal by administrator’s order);<br>• `0x4000` (transfer generated in the process of clearing);<br>• `0x1000000` (applied for clearing account);<br>• `0x4000000` (applied to client code);<br>• `0x8000000` (applied for group of client codes);<br>• `0x10000000` (applied for analytical clearing account) |
| amount | decn | Volume of transfer |

Table 22. Format of component `Underlying`: length 15 bytes

| Field | Datatype | Description |
|---|---|---|
| balance_id | int4 | Balance instrument ID |

| Field | Datatype | Description |
|---|---|---|
| qty | decn | Number of balance instrument units |
| flags | int2 | Flags field. Values:<br>• `0x1` (CORP_DUE_BILL): additional liability in connection with corporate event;<br>• `0x2` (CORP_CORRECTION): liability adjustment by clearing center in connection with corporate event;<br>• `0x4` (CORP_INCOME_RETURN): transfer of income in connection with corporate event;<br>• `0x8` (PRINCIPAL_OBLIGATION): principal liability flag |

# 3.4. Repetitive components and fields

Several message types contain one or more repeating components or fields which may have an arbitrary number of entries. One message may include multiple repetitive components and fields. All same-type repetitive components has a constant length.

A repeating component or field is always preceded by the two fields — `offset` and `count`. The `count` field specifies the number of entries. The `offset` field indicates an offset in bytes of first entry from the beginning of this very field; its value is no less than 4.

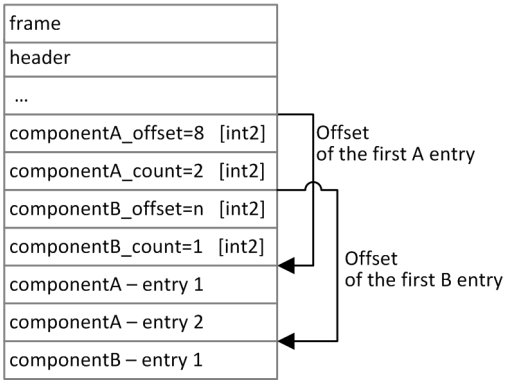


Figure 4. Template of a message with two repeating components

A repeating component may include another repeating component or field. In this case each entry refers to its own set of the embedded entries.

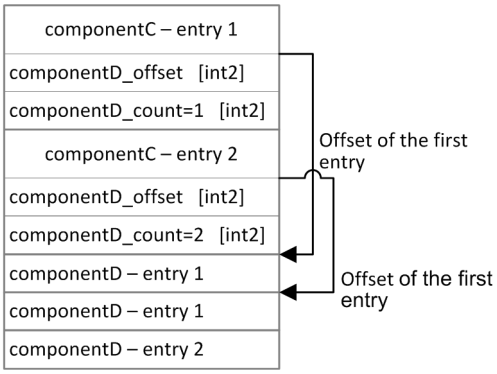


Figure 5. Template of embedded components

# 3.5. `Source_id` values

Field `source_id` is in the header gate_header; the field specifies the module transmitting message to gateway for sending it to client.

Table 23. `Source_id` values to be returned to client

| Range | Description |
|---|---|
| 100–199 | Trading system gateway |
| 200–249 | Clearing House risk parameter verification modules |
| 250–259 | Matching modules |
| 300–499 | Modules of generation and calculation of market data |
| 500–549 | Routing modules |
| 1000–1099 | Liquidity pools identifiers |

# 3.6. Liquidity pool identifiers

Liquidity pools' identifiers may be in fields `market`, `markets`, `market_id`, `source_id` and `exec_market` .

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

`1001` (TRADSYS) — all available liquidity pools

`1000` (SPB) — liquidity pool of SPB Exchange

`1010` (MOEX_FOND) — liquidity pool of Moscow Exchange

`1015` (IB) — execution at United States liquidity pools

`1017` (LSE) — liquidity pool of LSE

`1019` (SEHK) — liquidity pool of SEHK

`1021` (XETRA) — liquidity pool of german equities

# 4. Topics

## 4.1. Topic of clearing transactions and transfers

*Snapshot is an entire message history since the start of the trading day. Updates **replenish** earlier data.*

The topic of trades and transfers broadcasts the `Transfer` and `ClearingTrade` messages. The topic ID is `topic=Trades`.

The `Trades.Transfer` and `Trades.Trade` topics are also available, transmitting only corresponding message. These topic have their own numbering `topic_seq`. Subsequent versions of the system will not have these child topics.

Data on executed transfers is transmitted in the `Transfer` message.

Table 24. Format of message `Transfer`: msgid=802, size=117

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | user_id | ascii16 | Initiator login |
| 38 | account | [account] | Component specifying trading member, clearing account, and client code |
| 74 | transfer | [transfer] | Description component of executed transfer |

Data on executed clearing trade is transmitted in the `ClearingTrade` message (on message processing please refer to section 3.4).

Table 25. Format of message `ClearingTrade`: msgid=814, dynamic length

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | user_id | ascii16 | Initiator login |
| 38 | account | [account] | Component specifying trading member, clearing account, and client code |
| 74 | instrument | [instrument] | Component specifying trading instrument |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 80 | flags | int8 | Market dependent parameters. Values:<br>• `0x1` (eUserLastRec): last transaction message: last report on trades executed within a single transaction;<br>• `0x2` (eMMObligations): indicator of market maker executing the obligations at internal exchange, to be assigned to visible limit orders;<br>• `0x4` (eNoMMTrade): indicator of order by market maker that not to be matched with another market maker's order;<br>• `0x8` (ePresettlement): pre-delivery trade;<br>• `0x10` (eExternalActivity): transaction executed through external interfaces;<br>• `0x20` (eDelivery): delivery trade;<br>• `0x40` (eDeliverySwapGood): transfer of a bona fide participant during delivery;<br>• `0x80` (eDeliverySwapBad): transfer of a mala fide participant during delivery;<br>• `0x100` (eDeliveryDonorTrade): delivery transfer of participant with donor involvement;<br>• `0x200` (eNoSystem): negotiated trade indicator;<br>• `0x2000` (eIgnoreDynamicLimits): ignoring dynamic limits;<br>• `0x100000` (eClientPartialExecute): partial execution of address order sent by the client;<br>• `0x200000` (eHaltPeriodOrder): marker of issuing an order during a suspension period;<br>• `0x400000` (eOverTheCounter): marker of an order or a deal with over-the-counter instrument;<br>• `0x800000` (eRFQ): request for quote mode indicator |
| 88 | price | dec8 | Price |
| 96 | price_extra | dec8 | Additional price |
| 104 | parties | [otccodes] | Component specifying parties in negotiated order |
| 136 | amount_rest | int4 | Balance after the trades specified in this report |
| 140 | comment | char23+1 | Comments |
| 164 | extra_ref | ascii12 | Additional identifier of order |
| 176 | extra1 | char4+1 | Additional field 1 |
| 181 | match_id | int8 | Trade match ID |
| 189 | order_id | int8 | Order ID assigned by the trading system |
| 197 | exch_orderid | ascii20 | Order ID assigned by liquidity pool |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 217 | exec_market | int2 | Liquidity pool of execution (for a description of values, refer to Section 3.6) |
| 219 | dir | int1 | Side. Values:<br>• `1` (Buy): buy;<br>• `2` (Sell): sell |
| 220 | deals_offset | int2 | Offset of the first `deals` entry from the beginning of this field |
| 222 | deals_count | int2 | Number of the `deals` group entries |
| 224 | clr_deals_offset | int2 | Offset of the first `clr_deals` entry from the beginning of this field |
| 226 | clr_deals_count | int2 | Number of the `clr_deals` group entries |
| 228 | clr_repo_deals_offset | int2 | Offset of the first `clr_repo_deals` entry from the beginning of this field |
| 230 | clr_repo_deals_count | int2 | Number of the `clr_repo_deals` group entries |
| 232 | transfers_offset | int2 | Offset of the first `transfers` entry from the beginning of this field |
| 234 | transfers_count | int2 | Number of the `transfers` group entries |
| | > deals | [deal] | List of trades |
| | > clr_deals | [clr_deal] | List of clearing trades |
| | > clr_repo_deals | [clr_repo_deal] | List of repo trades |
| | > transfers | [transfer] | List of transfers as result of the trade |

### 4.1.1. Delivery

An obligation of asset delivery is represented as a spot instrument position. An execution date is assigned to each spot instrument. A spot position can be executed in the process of delivery by either of two ways:

1. Converting a spot position into a position in cash assets such as stocks, bonds, or foreign currency. Converting a position into cash assets is done by a transfer in the direction opposite to the current day execution and in the direction opposite to change of balance instrument position in such assets as stocks, bonds, or currencies.
2. Transfer of an obligation not secured by cash assets to the next trading day. A transfer is executed by automatic generation of negotiated repo orders sent from the login of the clearing member to the Clearing center. As a result of the orders execution, a trade with repo instrument is generated with execution on the next trading day. Obligations are transferred only for a main clearing accounts.

Trades of obligations transfer have `flags=0x20` in the `ClearingTrade` message.

## 4.2. Topic of clearing positions

*Snapshot is aggregation of all current data. Updates **replenish** and/or **replace** earlier data.*

Clearing positions topic broadcasts the `PositionUpdate` message. The topic ID is `topic=Pos.PositionUpdate`.

The position at clearing in the `clear_amount` field can be changed durind the trading session. This can be result of the transfer or delay of data from the previous session.

Table 26. Format of message `PositionUpdate`: msgid=851, dynamic length, keys=entity, balance_id, extra_key

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | entity | [account_entity] | Component specifying trading member, clearing account, and client code |
| 43 | balance_id | int8 | Balance instrument ID |
| 51 | extra_key | int8 | Additional ID |
| 59 | last_session_id | int4 | Last clearing session ID |
| 63 | last_clearing_id | int4 | Last clearing ID |
| 67 | clear_amount | decn | The position at clearing result. May vary during the trading session |
| 76 | amount_buy | decn | Number of the balance instrument lots in current session's buy trades |
| 85 | value_buy | decn | Sum total in all buy trades of current session |
| 94 | amount_sell | decn | Number of the balance instrument lots in current session's sell trades |
| 103 | value_sell | decn | Sum total in all sell trades of current session |
| 112 | last_transfer_id | int8 | ID of the last transfer of clearing instrument that changed the balance |
| 120 | extra_data_offset | int2 | Offset of the first `extra_data` entry from the beginning of this field |
| 122 | extra_data_count | int2 | Number of the `extra_data` group entries |
| | > extra_data | [extra_data] | Set of blocks with additional parameters of the instrument obligation |

# 4.3. Topic of funds

*Snapshot is aggregation of all current data. Updates **replenish** and/or **replace** earlier data.*

The funds topic broadcasts the `FundsUpdate` message. The topic ID is `topic=Funds.FundsUpdate`.

Table 27. Format of message `FundsUpdate`: msgid=852, size=79, keys=entity

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | entity | [account_entity] | Component specifying entity |
| 43 | free | decn | Available funds |
| 52 | reserve | decn | Reserved funds |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 61 | current | decn | Current funds |
| 70 | income | decn | Income |

# 4.4. Topic of margin rates

*Snapshot is an entire message history since the start of the trading day. Updates **replenish** earlier data.*

The topic of margin rates broadcasts the `RiskRates` message. The topic ID is `topic=RiskRates`.

Table 28. Format of message `RiskRates`: msgid=810, size=78

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | balance_id | int8 | Balance instrument ID |
| 30 | currency_id | int8 | Currency ID for risk rate calculations |
| 38 | last_session_id | int4 | Current session ID |
| 42 | last_clearing_id | int4 | Last clearing ID |
| 46 | time | time8m | Time of rates formation |
| 54 | price | dec8 | Price for risk rate calculations |
| 62 | rate_down | dec8 | Downward risk rate |
| 70 | rate_up | dec8 | Upward risk rate |

# 4.5. Topic of risk parameters

*Snapshot is aggregation of all current data. Updates **replenish** and/or **replace** earlier data.*

The topic of risk parameters broadcasts the `RiskParams` message. The topic ID is `topic=RiskParams`.

Table 29. Format of message `RiskParams`: msgid=860, dynamic length, keys=entity

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | entity | [account_entity] | Portfolio of transfer |
| 43 | reserved | int2 | Reserved field |
| 45 | params_offset | int2 | Offset of the first `params` entry from the beginning of this field |
| 47 | params_count | int2 | Number of the `params` group entries |
| | > params | [topic_risk_param] | List of parameters |

Table 30. Format of component `topic_risk_param`: length 20 bytes

| Field | Datatype | Description |
|---|---|---|
| type | int2 | Parameter. Value: `1` (CheckFunds): Check limits: value: 1. - enabled, 0. - off |
| reserved | decn | Reserved |
| result | decn | Current value |

# 4.6. Topic of trading member references

*Snapshot is aggregation of all current data. Updates **replace** earlier data.*

The topic of trading member references broadcasts the following messages:

- logins for connecting to the trading system gateways (User message),
- codes for negotiated trading (OTCCode message),
- clearing accounts (ClearingAccount message),
- trading and clearing members (Member message),
- client codes (Client message),
- groups of client codes (ClientGroup message).

Data transmitted in the topic is limited by the requesting login access permissions.

The topic ID is `topic=Participants`. Besides, the client may also connect to a separate child topic. Such child topic has its own numbering `topic_seq`, and its identifier is as follows `Participants.User`.

The `User` message conveys properties of a login for connecting to the trading system gateway (on message processing please refer to section 3.4).

Table 31. Format of message `User`: msgid=911, dynamic length, keys=user_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | user_id | ascii16 | Login, client gateway ID |
| 38 | type | int2 | Login type. Values:<br>• `1` (Clearing): clearing member;<br>• `2` (WildCard): trading member;<br>• `3` (Tag): login of group of clients marked by tag;<br>• `4` (Group): login of groups of client codes;<br>• `5` (Client): client code login |
| 40 | member_id | int8 | Trading member ID |
| 48 | main_clearing_account | ascii16 | Default clearing account |
| 64 | use_any_account | int1 | Right of access to all clearing accounts of the trading member. Values:<br>• `0` (No): no access;<br>• `1` (Yes): has access |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 65 | client_code | ascii16 | Client code ID. Filled when `type=5` |
| 81 | client_group | ascii16 | Group of client codes. Filled when `type=4` |
| 97 | tags | char15+1 | Client codes and/or groups of client codes marked by tag. Filled when `type=3` |
| 113 | clearing_account_offset | int2 | Offset of the first `clearing_account` entry from the beginning of this field |
| 115 | clearing_account_count | int2 | Number of the `clearing_account` group entries |
| 117 | otccodes_offset | int2 | Offset of the first `otccodes` entry from the beginning of this field |
| 119 | otccodes_count | int2 | Number of the `otccodes` group entries |
| 121 | login_flags | int8 | Login parameters. Values:<br>• `0x1` (IS_ACTIVE): active login;<br>• `0x8` (USE_ANY_GW): can ignore the list of permitted gateways;<br>• `0x10` (USE_ANY_ACCOUNT): can use any clearing account of the member;<br>• `0x20` (LEVEL_CM): level of clearing member;<br>• `0x40` (LEVEL_TM): level of trading member;<br>• `0x80` (LEVEL_CG): level of client group;<br>• `0x100` (LEVEL_CLIENT): client code level;<br>• `0x200` (LEVEL_TCA): level of clearing account;<br>• `0x400` (IS_CM_OPERATOR): operator of clearing member;<br>• `0x800` (IS_TM_OPERATOR): operator of the trading member;<br>• `0x2000` (IS_SUSPENDED): login suspended by client's command |
| 129 | rights_flags | int8 | Login permissions. Values:<br>• `0x1` (M_TRADE): issue trading orders;<br>• `0x800` (CAN_IGNORE_DYNAMIC_LIMITS): ignore the dynamic limits test |
|  | > clearing_account | ascii16 | Clearing account |
|  | > otccodes | [t_OTCCode] | List of identifiers for negotiated trading |

The `OTCCode` message contains information about a code for negotiated trading at liquidity pools available in the trading system.

Table 32. Format of message `OTCCode`: msgid=902, size=242, keys=code

| Offset | Field | Datatype | Description |
|---|---|---|---|
|  | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 22 | code | ascii16 | Code for negotiated trading |
| 38 | market_id | int2 | Liquidity pool ID (for a description of values, refer to Section 3.6 |
| 40 | desc | char64+1 | Full name in English |
| 105 | desc_ru | char128+1 | Full name in Russian |
| 234 | member_id | int8 | Member ID holding the registered code |

The `ClearingAccount` message conveys properties of clearing account, including links to clearing accounts in liquidity pools (on message processing please refer to section 3.4).

Table 33. Format of message `ClearingAccount`: msgid=903, dynamic length, keys=code, clearing_member_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | code | ascii16 | Clearing account ID |
| 38 | clearing_member_id | int8 | Clearing member ID |
| 46 | desc | char64+1 | Full name in English |
| 111 | desc_ru | char128+1 | Full name in Russian |
| 240 | is_principal | int1 | Virtual clearing account flag. Values:<br>• `0` (No): virtual clearing account;<br>• `1` (Yes): regular clearing account |
| 241 | parent_clear_account | ascii16 | Trading member ID who is transacting through the trading and clearing account. Filled when `is_virtual=1` |
| 257 | is_trusted_asset | int1 | Trust management flag. Values:<br>• `0` (No): not under trust management;<br>• `1` (Yes): under trust management |
| 258 | is_own_asset | int1 | Own clearing account flag. Values:<br>• `0` (No): clearing account of trading member's clients;<br>• `1` (Yes): clearing account of trading member |
| 259 | trade_member_id | int8 | Trading member ID who is transacting through the clearing account |
| 267 | default_client | ascii16 | Default client code |
| 283 | default_client_extra | ascii16 | Additional default client code |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 299 | segregation_type | int2 | Method of available funds accounting. Values:<br>• `0` (Custom): regular;<br>• `1` (Private): dedicated;<br>• `2` (Separate): isolated |
| 301 | exchange_accounts_offset | int2 | Offset of the first `exchange_accounts` entry from the beginning of this field |
| 303 | exchange_accounts_count | int2 | Number of the `exchange_accounts` group entries |
| | > exchange_accounts | [ExchangeAccount] | List of clearing accounts at liquidity pools |

The `Member` message contains properties of the trading or clearing member.

Table 34. Format of message `Member`: msgid=904, size=259, keys=member_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | member_id | int8 | ID of trading or clearing member |
| 30 | member_code | char32+1 | Unique symbol code |
| 63 | member_type | int2 | Type of member. Values:<br>• `0` (Clearing): clearing member;<br>• `1` (Trade): trading member |
| 65 | name | char64+1 | Full name in English |
| 130 | name_ru | char128+1 | Full name in Russian |

The `Client` message conveys properties of client code including links to a client code ID in liquidity pools (on message processing please refer to section 3.4).

Table 35. Format of message `Client`: msgid=905, dynamic length, keys=code, trade_member_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | code | ascii16 | Client code ID |
| 38 | trade_member_id | int8 | Trading member ID |
| 46 | name | char64+1 | Full name in English |
| 111 | name_ru | char128+1 | Full name in Russian |
| 240 | is_trust_asset | int1 | Trust management flag. Values:<br>• `0` (No): not under trust management;<br>• `1` (Yes): under trust management |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 241 | is_own_asset | int1 | Own client code flag. Values:<br>• `0` (No): trading member's client code;<br>• `1` (Yes): trading member code |
| 242 | has_client_group | int1 | Indicatior of belonging to a group of client codes. Values:<br>• `0` (No): does not belongs to a group;<br>• `1` (Yes): belongs to a group |
| 243 | client_group_id | ascii16 | Group of client codes. Indicated when `has_client_group=1` |
| 259 | exchange_clients_offset | int2 | Offset of the first `exchange_clients` entry from the beginning of this field |
| 261 | exchange_clients_count | int2 | Number of the `exchange_clients` group entries |
| 263 | tag_offset | int2 | Offset of the first `tag` entry from the beginning of this field |
| 265 | tag_count | int2 | Number of the `tag` group entries |
| 267 | individual_investment_account | int1 | Flag of individual investment account. Values:<br>• `0` (No): non-individual investment account;<br>• `1` (Yes): individual investment account |
| 268 | categoryProhibition | int4 | Prohibition bit mask by instrument category. Values:<br>• `0x1` (UNQUALIFIED_CLIENT_PROHIBITION);<br>• `0x2` (FOREIGNSECURITY_CLIENT_PROHIBITION);<br>• `0x4` (FOREIGNETF_CLIENT_PROHIBITION);<br>• `0x8` (UNQUOTRUSECURITY_CLIENT_PROHIBITION);<br>• `0x10` (DERIVATIVES_CLIENT_PROHIBITION);<br>• `0x20` (UNRATEDRUBOND_CLIENT_PROHIBITION);<br>• `0x40` (FOREIGNBOND_CLIENT_PROHIBITION);<br>• `0x80` (STRUCTEDBOND_CLIENT_PROHIBITION);<br>• `0x100` (STRUCTEDINCOMEBOND_CLIENT_PROHIBITION);<br>• `0x200` (REPO_CLIENT_PROHIBITION);<br>• `0x400` (CLOSEDFUND_CLIENT_PROHIBITION);<br>• `0x800` (DELISTED_CLIENT_PROHIBITION) |
| | > exchange_clients | [ExchangeClient] | Link to a client code at liquidity pool |
| | > tag | char15+1 | Client code marked by tag |

The `ClientGroup` message contains the description of a group of client codes (on message processing please refer to section 3.4).

Table 36. Format of message `ClientGroup`: msgid=906, dynamic length, keys=code, trade_member_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | code | ascii16 | Client codes group ID |
| 38 | trade_member_id | int8 | Trading member ID |
| 46 | name | char64+1 | Full name in English |
| 111 | name_ru | char128+1 | Full name in Russian |
| 240 | is_trusted_asset | int1 | Trust management flag. Values:<br>• `0` (No): not under trust management;<br>• `1` (Yes): under trust management |
| 241 | is_own_asset | int1 | Own client group flag. Values:<br>• `0` (No): group of trading member's client codes;<br>• `1` (Yes): group of trading member codes |
| 242 | tag_offset | int2 | Offset of the first `tag` entry from the beginning of this field |
| 244 | tag_count | int2 | Number of the `tag` group entries |
| | > tag | char15+1 | Group of client codes marked by tag |

# 4.7. Topic of instrument references

*Snapshot is aggregation of all current data. Updates **replace** earlier data.*

The topic of instrument references transmits data on instruments and trading modes:

- balance instrument Currency,
- balance instrument Issue,
- balance instrument Spot,
- balance instrument Bond,
- accrued coupon income (BondAccruedInterest),
- trading modes (TradeModes),
- liquidity pools (Market),
- trading instrument (Instrument).

The topic of instrument references broadcasts the TradingInstrumentStatus notifications about changes in the status of a trading instrument and the TradingInstrumentLimits updates about changes in the price limits for orders for a trading instrument. The BorrowingStatus message is sent, when short selling availability of an instrument has changed.

The topic ID is `topic=Instruments`. Besides, the client may also connect to a separate child topic. Such child topic has its own numbering `topic_seq`, and its identifier `topic` is as follows `Instruments.Instrument`.

Table 37. Format of message `Currency`: msgid=931, size=278, keys=balance_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | balance_id | int4 | Balance instrument ID |
| 26 | code | char32+1 | Currency code |
| 59 | desc | char64+1 | Full name of currency in English |
| 124 | desc_ru | char128+1 | Full name of currency in Russian |
| 253 | section | char8+1 | Market section |
| 262 | min_volume | dec8 | Minimum volume of asset |
| 270 | cfi_code | char6+1 | CFI code |
| 277 | is_test | int1 | Flag of test instrument. Values:<br>• `0` (REAL): Real;<br>• `1` (TEST): Test |

Table 38. Format of message `Issue`: msgid=932, size=486, keys=balance_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | balance_id | int4 | Balance instrument ID |
| 26 | code | char32+1 | Instrument ticker |
| 59 | desc | char64+1 | Full name of stock in English |
| 124 | desc_ru | char128+1 | Full name of stock in Russian |
| 253 | section | char8+1 | Market section |
| 262 | min_volume | dec8 | Minimum volume of lot |
| 270 | isin | char32+1 | ISIN |
| 303 | cfi_code | char6+1 | CFI code |
| 310 | reg_num | char32+1 | Registration number |
| 343 | issuer_name | char64+1 | Name of issuer or management company (for stakes) |
| 408 | issuer_country | char8+1 | Issuer country |
| 417 | face_value | dec8 | Face value |
| 425 | face_value_currency | char8+1 | Face value currency |
| 434 | total_amount | decn | Total amount of issue |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 443 | security_type | int1 | Security type. Values:<br>• 1 (OrdinaryShare): ordinary share or REIT;<br>• 2 (PreferredShare): preferred share;<br>• 3 (OpenEndedMutualFund): open-end mutual fund;<br>• 4 (ClosedEndMutualFund): closed-end mutual fund;<br>• 5 (ETF): security of foreign exchange traded fund;<br>• 6 (RDR): Russian depositary receipt;<br>• 7 (ADR): American depositary receipt;<br>• 8 (GDR): global depositary receipt;<br>• 9 (IntervalMutualFund): share of mutual fund |
| 444 | issue_date | time8m | Issue or registration date |
| 452 | quotation_list | char32+1 | Quotation list |
| 485 | is_test | int1 | Flag of test instrument. Values:<br>• 0 (REAL): Real;<br>• 1 (TEST): Test |

Table 39. Format of message `Spot`: msgid=933, size=293, keys=balance_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | balance_id | int4 | Balance instrument ID |
| 26 | code | char32+1 | Spot code |
| 59 | desc | char64+1 | Full name in English |
| 124 | desc_ru | char128+1 | Full name in Russian |
| 253 | section | char8+1 | Market section |
| 262 | lot | int8 | Lot volume in balance instrument units (instrument ID specified in `underlying_id`) |
| 270 | date_exec | time8m | Execution date |
| 278 | shift | int2 | Shift of execution date from today |
| 280 | underlying_id | int4 | Underlying instrument ID |
| 284 | accrued_interest | dec8 | Accrued interest as of the delivery date |
| 292 | is_test | int1 | Flag of test instrument. Values:<br>• 0 (REAL): Real;<br>• 1 (TEST): Test |

Table 40. Format of message `Bond`: msgid=935, dynamic length, keys=balance_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | balance_id | int4 | Balance instrument ID |
| 26 | code | char32+1 | Bond code |
| 59 | desc | char64+1 | Full name in English |
| 124 | desc_ru | char128+1 | Full name in Russian |
| 253 | section | char8+1 | Market section |
| 262 | min_volume | dec8 | Minimum volume of lot |
| 270 | isin | char32+1 | ISIN |
| 303 | cfi_code | char6+1 | CFI code |
| 310 | date_maturity | time8m | Maturity date |
| 318 | coupon_payment_offset | int2 | Offset of the first `coupon_payment` entry from the beginning of this field |
| 320 | coupon_payment_count | int2 | Number of the `coupon_payment` group entries |
| 322 | reg_num | char32+1 | Registration number of bond issue |
| 355 | issuer_name | char64+1 | Name of issuer or management company (for stakes) |
| 420 | issuer_country | char8+1 | Issuer country |
| 429 | face_value | dec8 | Face value |
| 437 | face_value_currency | char8+1 | Face value currency |
| 446 | issue_amount | decn | Total amount of issue |
| 455 | security_type | int1 | Security type. Values:<br>• `1` (GovernmentBond): government bond;<br>• `2` (MunicipalBond): municipal bond;<br>• `3` (CentralBankBond): Central bank bond;<br>• `4` (CorporateBond): corporate bond;<br>• `5` (FinancialInstitutionBond): financial institution bond;<br>• `6` (ExchangeTradedBond): exchange traded bond |
| 456 | issue_date | time8m | Date of issue |
| 464 | quotation_list | char32+1 | Quotation list |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 497 | is_test | int1 | Flag of test instrument. Values:<br>• 0 (REAL): Real;<br>• 1 (TEST): Test |
| | > coupon_payment | [coupon_payment] | Schedule of coupon payments |

Table 41. Format of message `BondAccruedInterest`: msgid=937, dynamic length, keys=balance_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | balance_id | int4 | Balance instrument (bond) ID |
| 26 | accrued_interest_offset | int2 | Offset of the first `accrued_interest` entry from the beginning of this field |
| 28 | accrued_interest_count | int2 | Number of the `accrued_interest` group entries |
| | > accrued_interest | [coupon_payment] | Coupon payment schedule |

*The list of trading modes, transmitted via the `TradeModes` messages, is the subject to modification. It is not recommended to use parameters of a specific trading mode for setting up the trading system.*

Table 42. Format of message `TradeModes`: msgid=942, size=222, keys=trade_mode_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | trade_mode_id | int2 | Trade mode ID |
| 24 | name | char64+1 | Name of trade mode in English |
| 89 | name_ru | char128+1 | Name of trade mode in Russian |
| 218 | is_address | int1 | Negotiated trading flag in trade mode. Values:<br>• 0 (No): non-negotiated;<br>• 1 (Yes): negotiated |
| 219 | is_multileg | int1 | Multi-leg trade indicator. Values:<br>• 0 (No): sigle-leg;<br>• 1 (Yes): multi-leg |
| 220 | is_ext_close | int1 | Closing auction indicator. Values:<br>• 0 (No): not traded at closing auction;<br>• 1 (Yes): traded at closing auction |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 221 | over_the_counter | int1 | Over-the-counter trade mode indicator. Values:<br>• `0` (No): not present;<br>• `1` (Yes): present |

Table 43. Format of message `Market`: msgid=936, size=220, keys=market_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | market_id | int4 | Liquidity pool ID (for a description of values, refer to Section 3.6) |
| 26 | desc | char64+1 | Full name of market in English |
| 91 | desc_ru | char128+1 | Full name of market in Russian |

Table 44. Format of message `Instrument`: msgid=973, dynamic length, keys=instrument_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | instrument_id | int4 | Trading instrument ID |
| 26 | symbol | char32+1 | Symbolic instrument ID |
| 59 | desc | char64+1 | Full instrument name in English |
| 124 | desc_ru | char128+1 | Full instrument name in Russian |
| 253 | status | [instrument_status] | Current status of trading instrument |
| 257 | type | char3+1 | Trading instrument type:<br>• f: futures;<br>• t: T+N;<br>• o: option;<br>• r: repo;<br>• pr: related trades;<br>• sw: swap;<br>• c: calendar spread;<br>• sf: spot-futures spread;<br>• dvp: delivery versus payment |
| 261 | auction_dir | int1 | Type of auction. Values:<br>• `0` (Direct): direct auction;<br>• `1` (Inverse): inverse auction |
| 262 | price_increment | dec8 | Price increment |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 270 | step_price | dec8 | Step price |
| 278 | legs_count | int2 | Number of legs |
| 280 | trade_mode_id | int2 | Trading mode ID |
| 282 | scalping_type | int2 | Scalping type. Values:<br>• `0` (NoScalping): no scalping;<br>• `1` (Custom): custom scalping;<br>• `2` (InverseScalping): inverse scalping |
| 284 | fee_schema | int1 | Fee scheme. Values:<br>• `1` (MakerTakerSpot): maker-taker for spot;<br>• `2` (MakerTakerFutures): maker-taker for futures;<br>• `3` (REPO): repo;<br>• `4` (MemberTariff): maker-taker for spot at members |
| 285 | fee_rate_offset | int2 | Offset of the first `fee_rate` entry from the beginning of this field |
| 287 | fee_rate_count | int2 | Number of the `fee_rate` group entries |
| 289 | curr_price | char16+1 | Currency of the instrument price |
| 306 | periods_offset | int2 | Offset of the first `periods` entry from the beginning of this field |
| 308 | periods_count | int2 | Number of the `periods` group entries |
| 310 | exchange_instrument_offset | int2 | Offset of the first `exchange_instrument` entry from the beginning of this field |
| 312 | exchange_instrument_count | int2 | Number of the `exchange_instrument` group entries |
| 314 | limit_up | dec8 | Price limit up |
| 322 | limit_down | dec8 | Price limit down |
| 330 | is_test | int1 | Flag of test instrument. Values:<br>• `0` (REAL): Real;<br>• `1` (TEST): Test |
| 331 | te_id | int2 | Trading engine ID |
| 333 | be_mode | int1 | Best execution mode. Values:<br>• `0` (External): external trades;<br>• `1` (Internal): internal trades at external prices |
| 334 | borrowing_status | int1 | Short selling availability for the instrument. Values:<br>• `1` (HARD_TO_BORROW): short selling unavailable;<br>• `2` (EASY_TO_BORROW): short selling available |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 335 | category | int4 | Instrument category bitmask. Values:<br>• `0x1` (UNQUALIFIED_CLIENT_PROHIBITION);<br>• `0x2` (FOREIGNSECURITY_CLIENT_PROHIBITION);<br>• `0x4` (FOREIGNETF_CLIENT_PROHIBITION);<br>• `0x8` (UNQUOTRUSECURITY_CLIENT_PROHIBITION);<br>• `0x10` (DERIVATIVES_CLIENT_PROHIBITION);<br>• `0x20` (UNRATEDRUBOND_CLIENT_PROHIBITION);<br>• `0x40` (FOREIGNBOND_CLIENT_PROHIBITION);<br>• `0x80` (STRUCTEDBOND_CLIENT_PROHIBITION);<br>• `0x100` (STRUCTEDINCOMEBOND_CLIENT_PROHIBITION);<br>• `0x200` (REPO_CLIENT_PROHIBITION);<br>• `0x400` (CLOSEDFUND_CLIENT_PROHIBITION);<br>• `0x800` (DELISTED_CLIENT_PROHIBITION) |
| | > fee_rate | dec8 | Fee rate |
| | > periods | [Period] | Component of trading periods (such as trading session) for instrument |
| | > exchange_instrument | [ExchangeInstrument] | Component specifying trading instruments at liquidity pools |

In this version of the trading system, the `fee_rate` group has five entries. The group has the following sequence of entries:

1. Minimum fee rate, in instrument currency.
2. Fee rate for pre-delivery trades, in instrument currency.
3. Taker fee rate depending on fee scheme: portion of trade volume in price currency for shares; amount of price currency per contract for derivatives; portion of the first leg value multiplied by repo duration for repo.
4. Maker fee rate depending on fee scheme: portion of trade volume in price currency for shares; amount of price currency per contract for derivatives; portion of the first leg value multiplied by repo duration for repo.
5. Accuracy.

Values of third and fourth records are based on the mechanism of fee calculation specified in the `fee_schema` field.

The `category` field indicates the instrument category in the trading system in accordance with designations adopted on the SPB Exchange. The correspondence between designations of instrument categories in the trading system and Interfax is shown in the table below.

Table 45. Correspondence between designations of instrument categories in the trading system and Interfax

| Bitmask | Instrument category in the trading system | Instrument category in Interfax |
|---|---|---|
| 0x1 | UNQUALIFIED_CLIENT_PROHIBITION | 0 |
| 0x2 | FOREIGNSECURITY_CLIENT_PROHIBITION | 10 |
| 0x4 | FOREIGNETF_CLIENT_PROHIBITION | 11 |
| 0x8 | UNQUOTRUSECURITY_CLIENT_PROHIBITION | 9 |
| 0x10 | DERIVATIVES_CLIENT_PROHIBITION | — |

| Bitmask | Instrument category in the trading system | Instrument category in Interfax |
|---|---|---|
| 0x20 | UNRATEDRUBOND_CLIENT_PROHIBITION | 6 |
| 0x40 | FOREIGNBOND_CLIENT_PROHIBITION | 7 |
| 0x80 | STRUCTEDBOND_CLIENT_PROHIBITION | 4 |
| 0x100 | STRUCTEDINCOMEBOND_CLIENT_PROHIBITION | 8 |
| 0x200 | REPO_CLIENT_PROHIBITION | — |
| 0x400 | CLOSEDFUND_CLIENT_PROHIBITION | 5 |
| 0x800 | DELISTED_CLIENT_PROHIBITION | — |

Table 46. Format of message `TradingInstrumentStatus`: msgid=2031, size=96, keys=instrument

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | instrument | [instrument] | Component specifying trading instrument |
| 28 | trading_status | int1 | Status of trading instrument. Values:<br>• `2` (HALT): trading is halted;<br>• `17` (TRADING): trading in progress;<br>• `18` (NO_TRADING): no trading;<br>• `102` (CLOSE): trading during closing auction;<br>• `103` (CLOSE_PERIOD): trading during close period;<br>• `107` (DISCRETE_AUCTION): trading during discrete auction;<br>• `118` (OPEN): trading during opening auction;<br>• `120` (FIXED_PRICE_AUCTION): trading at closing auction price |
| 29 | reserved | char2+1 | Reserved field. To be filled with null byte |
| 32 | comment | char63+1 | Comments |

Table 47. Format of message `TradingInstrumentLimits`: msgid=2032, size=42, keys=instrument_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | instrument_id | int4 | Trading instrument ID |
| 26 | limit_up | dec8 | Price limit up |
| 34 | limit_down | dec8 | Price limit down |

Table 48. Format of message `BorrowingStatus`: msgid=2033, size=27, keys=instrument_id

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | instrument_id | int4 | Trading instrument ID |
| 26 | borrowing_status | int1 | Short selling availability for the instrument. Values:<br>• `1` (HARD_TO_BORROW): Short selling unavailable;<br>• `2` (EASY_TO_BORROW): Short selling available |

# 4.8. System information topic

*Snapshot is aggregation of all current data. Updates **replace** earlier data.*

The topic of parameters depending on the system state. The topic ID is `topic=SysProperties`. The topic transmits the `SysProperties` messages.

Table 49. Format of message `SysProperties`: msgid=864, size=30, keys=key

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [header] | [header] | Header |
| 22 | key | int4 | Key. Values: 1 and 2 |
| 26 | data_offset | int2 | Offset of the first byte from the beginning of the `data` field |
| 28 | data_count | int2 | Byte length of the `data` field |
| | > data | char | Key-dependent data |

The `data` field contains data that depends on the key value in the `key` field.

Table 50. Data dependence on key value

| Key | Data description |
|---|---|
| 1 | Time of day when Day, IOC, FOK orders are no longer accepted, HH:MM:SS, UTC |
| 2 | Time of day when morning session finishes, HH:MM:SS, UTC |

# 5. Protocol specification

## 5.1. Session layer

### 5.1.1. Discovery service

The Discovery service provides a host address for client connections to the trading system gateway. The client should request the service for address allocation each time before connecting to the gateway. Upon receipt of response, the client should disconnect from the login server and connect to a gateway through the received address.

For the address for accessing the Discovery service, please refer to document *Network Connectivity*.

After establishing connection with the Discovery service, the client should send the `Hello` message. The message contains the session header `frame` (for more details, refer to Section 3.2). The client should specify login and password, and the IP address of the client must be authorized for the specified login (user ID).

Table 51. Format of message `Hello`: msgid=1, size=32

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | login | ascii16 | Login |
| 16 | password | ascii16 | Password |

In response to request, the server sends the `Report` message. If this message has `status`=0, the message contains repetitive component `Report_Address`; the number of component records will be specified in the field `addresses_count` (for more details on processing of repeating groups, refer to Section 3.4). The component includes fields `type` (gateway attribute) and `address` (host address and gateway port). Gateway attributes may combine.

After the trading system responds, the gateway will expect the client's login connection to the specified address. In case of failure, the client is recommended to make two additional connection attempts with an interval of half a second. If the login is invalid or blocked, the server response will contain `status=1`.

Table 52. Format of message `Report`: msgid=2, dynamic length

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | status | int2 | Request status. Values:<br>• `0` (Success): success;<br>• `1` (Fail): reject due to invalid login/password |
| 2 | reason | char127+1 | Textual description |
| 130 | addresses_offset | int2 | Offset of the first `addresses` entry from the beginning of this field |
| 132 | addresses_count | int2 | Number of the `addresses` group entries |
| | > addresses | [Report_Address] | Address list |

Table 53. Format of component `Report_Address`: length 52 bytes

| Field | Datatype | Description |
|---|---|---|
| type | int2 | Gateway attributes, bit mask. Values:<br>• `0x1` (Transaction): trading;<br>• `0x2` (DropCopy): drop-copy;<br>• `0x4` (Risk): risk management;<br>• `0x8` (Dictionary): dictionaries;<br>• `0x10` (MarketData): market data recovery;<br>• `0x4000` (Backup): backup |
| ver | int1 | Interface version |
| pad0 | int1 | Reserved field, filled with zero bytes |
| address | char47+1 | Address of host and gateway port |

## 5.1.2. Session initialization

A session is established over a network connection between the client's system and the gateway of the trading system.

Once connection is established, the client can send the `Login` message to initiate a session. The message includes the user ID and the password. The system validates the authentication parameters and answers with the `Logon` message and so the session is active. Upon receipt of a malformed `Login` message or invalid login/password, the server breaks the connection.

A login may have a single concurrent session. If the server detects a second connection attempt via the same login while a valid session is already underway, the server will respond with `Reject`.

Table 54. Format of message `Login`: msgid=8001, size=37

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | login | ascii16 | Login |
| 16 | password | ascii16 | Password |
| 32 | reset_seq | int1 | Reset sequence numbers indicator. Values:<br>• `0` (no): sequence numbers continue;<br>• `1` (yes): sequence numbers reset |
| 33 | heartbeat_ms | int4 | `Heartbeat` frequency in milliseconds |

Table 55. Format of message `Logon`: msgid=8101, size=24

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | last_seq | int8 | Last application message available to client.<br>If altered form the last received message, `ResendRequest` is to be sent |
| 8 | expected_seq | int8 | Next application message expected from client |
| 16 | system_id | ascii8 | Deployment ID |

## 5.1.3. Keeping session in active state

The client and the gateway must exchange `Heartbeat` messages to maintain session in active state. Heartbeat must be sent, if no session or application message has been sent within the heartbeat interval.

When initiating a session, the client sets the heartbeat interval in the field `heartbeat_ms` of the `Login` message.

If the server detects that the client has not sent any messages, including the `Heartbeat` messages, for a period longer than the specified interval, the system will break the connection. The client is expected to do the same, if inactivity is detected on the part of the server.

Table 56. Format of message `Heartbeat`: msgid=8103, size=0

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |

## 5.1.4. Send rate limit for session messages

The rate at which a client can send session level messages into the system is limited. When client's send rate exceeds the limit, the system terminates the user session.

## 5.1.5. Message numbers

All application messages have a unique number throughout the trading day. Messages by each session side (the client and the gateway) are sequentially numbered with positive integers starting with 1. This allows to request and resend messages lost in case of unexpected disconnection.

Sequence numbers are not assigned to session messages — the `seq` value is always 0.

In order to maintain sequential numbering of messages, at session initialization the gateway provides two key values in its `Logon` message — the number of the last message sent (`last_seq`) and the expected number of the following message (`expected_seq`).

The gateway accumulates messages addressed to the client even when no connection established. If the `last_seq` filed is greater than the last message received during the previous session, the client should request not received messages via the `ResendRequest`.

If the message number differs from the expected one, the gateway terminates the connection. After disconnection, the client should reconnect by addressing the Discovery service and restore the number of messages according to the values obtained in the `Logon` message from the gateway. The gateway never initiates a change in numbering when receiving a message with the number higher than expected.

The trading system supports continuous message numbering between trading sessions, including trading days. The client should set `reset_seq=1` in message `Login` at session initialization to reset numbering.

## 5.1.6. Message resend request

If the client's system has not been connected to the gateway for some time, the gateway may accumulate messages intended for the client, but not received by him. In order to be convinced of the presence of such messages, it is necessary to compare the `seq` number of the last received message with the `last_seq` number in the `Logon` message. If the numbers are different, the client should use the `ResendRequest` message to retrieve the missed messages.

The client can request missed messages sent during the current and previous trading days. If the client forcefully resets the message numbering (`reset_seq = 1` in the `Login` message), then the request for missed messages which were sent prior to this reset is not possible.

The `ResendRequest` message must contain the number of the first message in the `from_seq` field and the number of the last message in the `till_seq` field within the requested messages range. Possible request parameters are listed below:

1. `from_seq=n, till_seq=m` — request for messages from *n* to *m* but not exceeding the maximum available number.
2. `from_seq=0, till_seq=n` — request for messages from the lowest number available to *n* but not exceeding the maximum available number.
3. `from_seq=n, till_seq=0` — request for messages from *n* to the last number available but not exceeding the maximum available number.

4. `from_seq=0, till_seq=0` — request for all available messages but not exceeding the maximum available number.

The number of requested messages in one request cannot exceed the specified value (for more details please see document *Network Connectivity*, section 1.3). To request more messages, the client should send multiple consecutive `ResendRequest` messages.

Table 57. Format of message `ResendRequest`: msgid=8005, size=16

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | from_seq | int8 | First requested message |
| 8 | till_seq | int8 | Last requested message |

In response to a correctly formed request, the trading system will transmit requested messages, preceding the sending by the `ResendReport` message with the `ACK` status. When messages are complete, the gateway will send the `ResendReport` message with status `MORE` or `FINISH`. The `MORE` status means that the number of the last message within the requested messages range is less than the number of the last message sent by the gateway. That is, there are messages that are not included in the request output. They could have been generated during the request execution, or the number of messages in one request exceeded the specified value. In this case, another `Resend Request` message should be made.

If the recovery of missed messages is performed by means of several consecutive `Resend Request` messages, each subsequent request should be performed after receiving all messages of the previous request. Otherwise, it will be rejected by the `ResendReport` message with the `DUPLICATE_REQUEST` status.

When connecting for the first time in the current trading day, it is recommended to use a request with parameters `from_seq = -1, till_seq = 0`. If, after sending, the gateway returns the `ResendReport` message with the `MORE` status, you should send another request, indicating in the `from_seq` field a number one more than the last one forwarded message, and `till_seq = 0`.

To recover missed messages after reconnection, you must send a request with the parameters `from_seq = n, till_seq = s`, where *n* is the number of the last received message before the connection was terminated plus one, and *s* is the number of the last message available to the client (`last_seq` field) received in the `Logon` message. If, after sending, the gateway returns the `ResendReport` message with the `MORE` status and the client has not yet received messages with the specified numbers, another request should be sent, indicating in the `from_seq` field a number one more than that of the last forwarded message, and `till_seq = s`.

*The ResendRequest is processed by the gateway in parallel with the sending of current messages. That is, the client can receive both missed messages and messages sent after connecting. The client system must independently restore the correct order of received messages based on their seq numbers.*

Table 58. Format of message `ResendReport`: msgid=8105, size=2

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | status | int2 | Request status. Values:<br>• `0` (ACK): gateway is ready to respond to a request;<br>• `1` (MORE): gateway executed the query and still has data for client;<br>• `2` (FINISH): all available data sent to the client;<br>• `3` (DUPLICATE_REQUEST): server busy with the previous `ResendRequest`;<br>• `4` (UNAVAILABLE): recovery service unavailable |

## 5.1.7. Message numbers reset by the client

The client may change the number of expected message at the gateway. For this purpose, the client should send `SequenceReset` specifying next message number in the `next_seq` field. At that, the new number shall not be less than the current value at the gateway.

Table 59. Format of message `SequenceReset`: msgid=8004, size=8

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | next_seq | int8 | Next sequence number expected from client |

## 5.1.8. Message numbers reset by the trading system

In response to `ResendRequest`, the trading system may also send the `GapFill` request to change the number of message expected by the client. The trading system sends `GapFill` to the client to skip update of topic.

Table 60. Format of message `GapFill`: msgid=8106, size=8

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | next_seq | int8 | Next sequence number to be expected by the recipient |

## 5.1.9. Session termination

The server or the client sends `Logout` to terminate the session and expects the other party to disconnect.

Table 61. Format of message `Logout`: msgid=8002, size=16

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | login | ascii16 | Login, client gateway ID |

## 5.1.10. Message rejection

If the client's message is either malformed or contains invalid values, the system rejects such message and responds with `Reject`. The `ref_msgid` field specifies message type, `ref_seq` contains the application level message number or has 0 for session message, fields `reason` and `message` contain, correspondingly, code of rejection reason and its description.

Table 62. Format of message `Reject`: msgid=8102, size=45

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | ref_seq | int8 | Sequence number of rejected message |
| 8 | ref_msgid | int2 | Type of rejected message |
| 10 | reason | int2 | Code of rejection reason |
| 12 | message | char32+1 | Rejection parameters or textual description |

## 5.1.11. Disconnection

System disconnects when receiving message:

- with unknown value of `msgid`,

- with a `size` incorrect for the specified message type,
- with a `seq` number other than expected.

## 5.1.12. Data request

To request data, client should send `TopicRequest` to the trading system gateway specifying `topic` ID and `mode` (snapshot or snapshot and updates). The client does not have to fill the `clorder_id` field.

The client can specify the range of requested messages through `topic_seq` and `topic_seqend` fields:

- `topic_seq=n, topic_seqend=m` — request for messages from *n* to *m*.
- `topic_seq=0, topic_seqend=n` — request for messages from the lowest number available to *n*.
- `topic_seq=n, topic_seqend=0` — request for messages from *n* to the last number available.
- `topic_seq=0, topic_seqend=0` — request for all available messages.

When making an initial request for Clearing trades and transfers and Risk rates topics, the client should specify 0 in `topic_seq` and `topic_seqend` fields. And in a repeating request, value of the `topic_seq` field should be one more than value of the `topic_lastseqsent` field in the last received `TopicReport`. If `TopicReport` is not recieved, value of the `topic_seq` field should be one more than that of the last message received.

When requesting for Clearing positions, Members' funds, Risk parameters, Trading members' references and Instrument references topics, the client should specify 0 in `topic_seq` and `topic_seqend` fields.

If a request can be processed, the client will receive the TopicReport message and after that should expect data messages. In case of requesting a topics snapshot for Clearing positions, Members' funds, Risk parameters, Trading members' references and Instrument references, the client can receive updates along with the requested messages. After data transfer is completed, the client will also receive `TopicReport`.

If a request is incorrect or cannot be processed, the client will receive the TopicReject message.

For more details about interaction with gateway, refer to Section 2.1.

*If you want to request a new topic, wait until you have received all messages, related to the previous topic request, to avoid network overload.*

Table 63. Format of message `TopicRequest`: msgid=301, size=101

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [user_header] | [user_header] | Standard header |
| 20 | topic | ascii64 | Topic ID |
| 84 | topic_seq | int8 | First number of requested messages |
| 92 | topic_seqend | int8 | Last number of requested messages |
| 100 | mode | int1 | Broadcast mode. Values:<br>• `0` (DATA_SLICE): snapshot;<br>• `1` (SUBSCRIBE): snapshot and subsequent updates |

## 5.1.13. Updates canceling

To stop receiving updates the client should send `TopicCancel` to the trading system gateway specifying one or both topic identifiers — `topic` and `topic_id`.

In case of successful request processing, the updates will be canceled and the client will receive the TopicReport message with `status=2`. Still for some time after the report client may continue receiving messages with data.

If a request is incorrect or cannot be executed, the client will receive the TopicReject message.

For more details about interaction with gateway, refer to Section 2.2.

Table 64. Format of message `TopicCancel`: msgid=302, size=88

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [user_header] | [user_header] | Standard header |
| 20 | topic | ascii64 | Topic ID |
| 84 | topic_id | int4 | Numerical ID of topic |

## 5.1.14. Report on executing request

The client will receive the `TopicReport` message in the following cases:

- successful execution of the data request TopicRequest;
- successful execution of request to cancel updates TopicCancel;
- completion of snapshot transmission.

The message includes reference fields `topic_lastseq` (the number of the last message generated in the topic) and `topic_lastseqsent` (the number of the last message sent to the client).

Table 65. Format of message `TopicReport`: msgid=401, size=134

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [gate_header] | [gate_header] | Standard header |
| 46 | topic | ascii64 | Topic ID |
| 110 | topic_id | int4 | Numerical topic ID |
| 114 | status | int2 | Status of data transfer. Values:<br>• `0` (DATA_SLICE): snapshot transfer;<br>• `1` (ADD_SUBSCRIBE): snapshot transfer with updates;<br>• `2` (DEL_SUBSCRIBE): cancellation of updates |
| 116 | marker | int2 | Indicator of start and finish of data transfer. Values:<br>• `0` (START): start of data transfer;<br>• `1` (END): end of the data transfer;<br>• `2` (SLICE_END): snapshot transfer completed |
| 118 | topic_lastseq | int8 | Number of the last message generated in the topic |
| 126 | topic_lastseqsent | int8 | Number of the last message sent to the client |

## 5.1.15. Report on rejecting request

If the client's request is incorrect or cannot be processed, the client will receive the TopicReject message. The reason for rejection is specified in the `reason` field.

The `TopicReject` message includes reference fields `topic_lastseq` (the number of the last message generated in the topic) and `topic_lastseqsent` (the number of the last message sent to the client).

For streams with `Trades` and `IQ` identifiers (refer to document *Network Connectivity*) in the `TopicReject` messages, the number in the `topic_firstseq` field is the same as the number of the first available message.

Table 66. Format of message `TopicReject`: msgid=402, size=142

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [gate_header] | [gate_header] | Standard header |
| 46 | topic | ascii64 | Topic ID |
| 110 | topic_id | int4 | Numerical topic ID |
| 114 | status | int2 | Status of data transfer. Values:<br>• `0` (DATA_SLICE): snapshot transfer;<br>• `1` (ACTIVE): snapshot transfer with updates;<br>• `2` (INACTIVE): no data transfer |
| 116 | reason | int2 | Reason for rejection. Values:<br>• `1` (BAD_TOPIC): invalid topic identifier;<br>• `2` (ALREADY_SUBSCRIBED): transfer in progress already;<br>• `3` (NOT_SUBSCRIBED): transfer was not requested;<br>• `4` (DATA_NOT_AVAILABLE): data not available;<br>• `5` (DUPLICATE_REQUEST): repeated request;<br>• `6` (BAD_SEQ): invalid message number in the topic;<br>• `7` (BAD_MODE): invalid mode |
| 118 | topic_firstseq | int8 | Number of the first message since the beginning of the trading day |
| 126 | topic_lastseq | int8 | Number of the last message generated in the topic |
| 134 | topic_lastseqsent | int8 | Number of the last message sent to the client |

# 5.2. Application layer

## 5.2.1. Send rate limit for client requests

The rate at which a client can send requests into the system is limited. There are two limits:

1. When the first threshold of send rate is reached, the system starts declining the application level requests and transmits the report on rejecting request with reason "Number of messages exceeded limit".
2. When the second threshold of send rate is reached, the system terminates the user session.

## 5.2.2. Changing client limits

### 5.2.2.1. Request for limit change

The client should send the `LimitRequest` message to the trading system gateway to change client instrument limits. The `LimitRequest` message can only be sent by the login levels LEVEL_TM and/or LEVEL_CG.

It should contain the client order identifier `clorder_id`, unique during the trading day for each login, and the balance instrument identifier `balance_id`, which limit is to be changed (available for balance instruments: `Currency`, `Issue` and `Bond`).

A limit can be set for several entities: a client code, a group of client codes, a clearing account or an analytic clearing account. The type of entity should be specified in the `entity_type` field, the identifier of a specific entity should be specified in the `entity_id` field.

The parameter of limit change should be set in the `flags` field.

Limit can be decreased or increased by a value specified in the `amount` field.

In response to a valid request, the trading system will send the LimitReport message to the client. A request, containing invalid data, will be rejected by the RejectReport message. For more details about an interaction with gateway, refer to Section 2.3.

Table 67. Format of message `LimitRequest`: msgid=501, size=67

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [user_header] | [user_header] | Standard header |
| 20 | balance_id | int8 | Balance instrument ID |
| 28 | entity | [account_entity] | Component specifying entity of limit |
| 49 | mode | int1 | Limit changing mode. Values:<br>• `1` (Enrolment): deposit;<br>• `2` (Withdrawal): withdrawal |
| 50 | flags | int8 | Bit mask of limit change. Values:<br>• `0x100` (FORCED_UPDATE): not to verify a non-increase in initial margin (IM) arrears (“hard” withdrawal);<br>• `0x400` (FORCED_ASSET_UPDATE): not to verify asset presence in case of withdrawal |
| 58 | amount | decn | Volume of limit change |

## 5.2.2.2. Report on changing limit

After limit is changed as result of the LimitRequest message, the client will receive a report on changing limit `LimitReport`. The report contains a new limit value in the `amount_rest` field.

Table 68. Format of message `LimitReport`: msgid=601, size=102

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [gate_header] | [gate_header] | Standard header |
| 46 | balance_id | int8 | Balance instrument ID |
| 54 | entity | [account_entity] | Component specifying entity of limit |
| 75 | mode | int1 | Limit changing mode. Values:<br>• `1` (Enrolment): deposit;<br>• `2` (Withdrawal): withdrawal |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 76 | flags | int8 | Bitmask of limit change. Values:<br>• `0x100` (FORCED_UPDATE): not to verify a non-increase in initial margin (IM) arrears (“hard” withdrawal);<br>• `0x400` (FORCED_ASSET_UPDATE): not to verify asset presence in case of withdrawal |
| 84 | amount | decn | Volume of limit change |
| 93 | amount_rest | decn | Actual size of limit after operation |

### 5.2.2.3. Report on rejecting LimitRequest

The LimitRequest message containing incorrect values will be rejected by the `RejectReport` message. Reasons for rejection are specified in the `reason` field, and the `message` field may contain detailed description of rejection reasons or parameters.

Table 69. Format of message `RejectReport`: msgid=201, size=91

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [gate_header] | [gate_header] | Standard header |
| 46 | market | int2 | Liquidity pool rejecting client’s order (for a description of values, refer to Section 3.6) |
| 48 | reason | int2 | Code of rejection reason |
| 50 | message | char32+1 | Rejection code parameters or textual description of the rejection reason |
| 83 | extra_data0 | int8 | Reserved field. To be filled with null byte |

## 5.2.3. Conversion of price to yield

### 5.2.3.1. Request for conversion of price to yield

The client should send the `YieldConversionRequest` message to the trading system gateway to convert price to yield (or yield to price).

The request should contain the client order identifier `clorder_id`, unique during the trading day for each login, and the balance instrument identifier `instrument_id`, which the conversion is performed. The `market_id` field should contain 0.

The conversion can be performed in two directions: from price to yield and from yield to price (must be specified in the `conversion_dir` field). Yield type should be specified in the `yield_type` field.

The value for conversion should be specified in the `value` field.

In response to a valid request, client will receive the YieldConversionReport message containing the conversion result.

Table 70. Format of message `YieldConversionRequest`: msgid=514, size=36

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |

| Offset | Field | Datatype | Description |
|---|---|---|---|
| 0 | [user_header] | [user_header] | Standard header |
| 20 | instrument | [instrument] | Component specifying trading instrument. Value of the market_id field is 0 |
| 26 | conversion_dir | int1 | Direction of conversion. Values:<br>• 0 (YIELD_TO_PRICE): from yield to price;<br>• 1 (PRICE_TO_YIELD): from price to yield |
| 27 | yield_type | int1 | Type of yield. Values:<br>• 0 (YTM): effective yield to maturity;<br>• 1 (YnTM): nominal yield to maturity |
| 28 | value | dec8 | Value to convert |

## 5.2.3.2. Report on conversion of price to yield

The report contain the client order identifier `clorder_id`, the conversion result is specified in the `result` field.

Table 71. Format of message `YieldConversionReport`: msgid=614, size=70

| Offset | Field | Datatype | Description |
|---|---|---|---|
| | [frame] | [frame] | Session header |
| 0 | [gate_header] | [gate_header] | Standard header |
| 46 | instrument | [instrument] | Component specifying trading instrument |
| 52 | conversion_dir | int1 | Direction of conversion. Values:<br>• 0 (YIELD_TO_PRICE): from yield to price;<br>• 1 (PRICE_TO_YIELD): from price to yield |
| 53 | yield_type | int1 | Type of yield. Values:<br>• 0 (YTM): effective yield to maturity;<br>• 1 (YnTM): nominal yield to maturity |
| 54 | value | dec8 | Value to convert |
| 62 | result | dec8 | Result of conversion |

# Appendix A. Error codes

Table 72. 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. |
| 1116 | Invalid level. |

| Code | Description |
|---|---|
| 1117 | Invalid trade_mode. |
| 1200 | Invalid segment. |
| 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 rejected 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 reject this order. |

| Code | Description |
|---|---|
| 1311 | Order currently being replaced. |
| 1312 | Order sent before system crash, but received after recovery. |
| 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. |
| 1405 | No permissions to trade this instrument category. |
| 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. |

| Code | Description |
|---|---|
| 1721 | Incorrect instrument limit for limited margin. |
| 1722 | Incorrect order limit for limited margin. |
| 1723 | Incorrect extra limit for limited margin. |
| 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. |
| 1800 | Incorrect yield type specified. |
| 1801 | Incorrect yield conversion direction specified. |
| 1980 | Invalid stages in info field. |
| 2100 | Account does not belong to member_id. |

| Code | Description |
|---|---|
| 2200 | No permissions to submit trading instructions. |
| 2201 | Client group level prohibition is set. |
| 2202 | Trade member level prohibion is set. |
| 2203 | Clearing member prohibion is set. |
| 2204 | Trade administrator level prohibion is set. |
| 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. |
| 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. |
| 3005 | User has no permission to trade this instrument during the current trading period. |
| 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. |

| Code | Description |
|---|---|
| 4001 | The specified liquidity pool not available. |
| 4002 | Order forcedly routed to a liquidity pool after rejected 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. |
| 4101 | The order was rejected by an external platform. |
| 4200 | Invalid client for TCA registered at liquidity pool. |
| 4201 | Invalid TCA for liquidity pool. |
| 5000 | Invalid application message type. |
| 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 | Snapshot with updates has already been requested. |
| 5302 | Snapshot with updates has not been requested. |
| 5303 | Requested data not available. |
| 5304 | Another request processing in progress. |

| Code | Description |
|---|---|
| 5400 | Reset_seq indicated, but seqnums cannot be reset. |
| 5401 | Number of messages exceeded limit. |
| 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 1.10.0 December 24, 2015**

1. Added component extra_data to the message PositionUpdate.
2. The field amount_rest_extra is removed from message ClearingTrade, added components clr_repo_deals and transfers, and changed the field value msgid.
3. In message User added fields login_flags and rights_flags and changed the field value msgid.
4. In message Instrument added fields is_test, te_id, and be_mode, removed field reserved, and changed the field value msgid.
5. In component Underlying added field flags and changed the dimensions of field qty.
6. The composition of component clr_deal changed.

**Version 1.9.0 July 2, 2015**

The order of fields trade_mode_id and reserved changed in the message table Instrument.

**Version 1.8.0 June 19, 2015**

The format of message Instrument changed: size of field trade_mode_id reduced to 2 bytes and added by field reserved in front of it.

**Version 1.7.1 June 4, 2015**

The message header LimitReport corrected.

**Version 1.7.0 May 12, 2015**

1. Messages for changing risk parameters added.
2. Risk parameters stream added.
3. New error codes added to application A: DENY_CLIENT_ACCOUNT, BAD_SOR_EXCHANGE, BAD_TYPE, BAD_VALUE, AMBIGUOUS_TYPE, INSUFFIC_BLOCKED_ASSETS and error codes ranging from 8300 to 8325.

**Version 1.6.1 March 25, 2015**

Sequence of records in field [fee_rate] corrected.

**Version 1.6.0 February 20, 2015**

1. Field accrued_interest added to message Spot.
2. Field individual_retirement_account added to message Client.

**Version 1.5.0 February 11, 2015**

1. Message TradingInstrumentLimits added to instrument streams.
2. Fields limit_up and limit_down added to message Instrument.
3. Field is_ext_close added to message TradeModes.
4. Gateway mode when sending notification TopicReport corrected.
5. New extra_ref field added and size of field extra1 changed in the Trade message .
6. Fields reg_num, iissuer_name, issuer_country, face_value, face_value_currency, total_amount, security_type, issue_date, and quotation_list added to message Issue.
7. Fields reg_num, issuer_name, issuer_country, face_value, face_value_currency, issue_amount, security_type, issue_date и quotation_list added to message Bond.
8. Margin rates stream added (please refer to section 4.4).
9. Field maturity_date renamed to maturity in message Bond.
10. Errors 1115, 1315, 1316, 8103, 8104, 8105, 8106, and 8201 added to error codes table.

**Version 1.4.7 December 15, 2014**

Value 3 added in field entity_type for component account_entity.

**Version 1.4.6 November 28, 2014**

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

**Version 1.4.5 November 20, 2014**

1. New value added to field mode for component Period.
2. New values added to field flags.
3. Trades.Transfer and Trades. Trade streams are not recommended for use as they will be absent in subsequent versions of the system.

**Version 1.4.4 October 29, 2014**

1. Subsection Supply added to section "Flow of Clearing Transactions and Transfers".
2. List of values revised in fields type and scalping_type of message Instrument.
3. Instrument streams updated.

**Version 1.4.3 October 9, 2014**

Identifiers of message and stream of ClientGroup guides updated.

**Version 1.4.2 October 1, 2014**

1. Changed msgid in messages Currency, Issue, Bond, Futures, Spot, Instrument.
2. Message TradingInstrumentLimits added to instrument streams.
3. Size of field code changed in messages Currency, Issue, Spot, Futures, and Bond.
4. Size field symbol changed in message Instrument.
5. Component instrument_status added to component ExchangeInstrument.
6. Field status replaced by component instrument_status in message Instrument.
7. Field ver added to report Discovery service.
8. Gateway mode when resending messages corrected (please refer to section 5.1.6).
9. Size of field fee in component clr_deal corrected.

**Version 1.3 August 26, 2014**

1. Message Bond added to instrument streams.
2. Value msgid in the Trade message corrected.
3. Field buyback_amt deleted from component clr_deal, fields deal_amount and accr_interest addedt.
4. Field buyback_clr_id added to component deals.
5. Consistency of message numbering corrected (please refer to section 5.1.5).

**Version 1.2 July 31, 2014**

1. Datatype of field amount in messages LimitRequest and LimitReport corrected.
2. Datatype of field amount_rest in message LimitReport corrected.
3. Datatype of fields free, reserve, current, and income in message FundsUpdate corrected.
4. Size of field source_id in component t_OTCCode corrected.
5. Message CombinedCommodity added to instrument streams.

**Version 1.1 June 30, 2014**

1. Datatype of fields type and tags in message User corrected.
2. Size of fields source_id, desc, and desc_ru in message OTCCode corrected.
3. Field member_id added to message OTCCode.
4. Size of fields desc, desc_ru, and segregation_type in message ClearingAccount corrected.
5. Size of fields name and name_ru in message Member corrected.
6. Field member_code added to message Member.
7. Size of fields name and name_ru in message Client corrected.
8. Datatype of field tag in message Client corrected.
9. Size of fields name and name_ru in message ClientGroup corrected.
10. Datatype of field tag in message ClientGroup corrected.
11. Field cfi_code added to message Currency.
12. Datatype of fields code, desc, desc_ru, and section in message Currency corrected.
13. Datatype of fields code, desc, desc_ru, and section in message Issue corrected.

14. Fields isin and cfi_code added to message Issue.
15. Datatype of fields code, desc, desc_ru, and section in message Spot corrected.
16. Field cfi_code added to message Spot.
17. Datatype of fields code, desc, desc_ru, section, and exec_type in message Futures corrected.
18. Size of fields name and name_ru in message TradeModes corrected.
19. Datatype of fields symbol, desc, desc_ru, status, fee_schema, and curr_price in message Instrument corrected.
20. Fields desc and desc_ru deleted from component OTCCode.
21. Field member_id added to component OTCCode.
22. Size of field type in component ExchangeAccount corrected.
23. Datatype of type fields code_group, code, and code_extra in component ExchangeInstrument corrected.
24. Size of field type in component Period corrected.