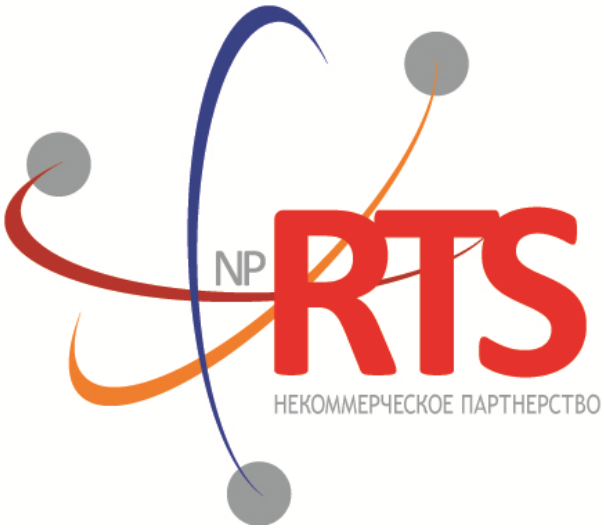

ADVANCED
TRADING & CLEARING
PLATFORM

Native Protocol Risk Management Gateway

System version 1.7

Interface version 36

Document version 1.13.0

27 February 2018

Revision history

Version 1.13.0 November 3, 2017

1. The msgid field value changed for the TradeModes message.
2. The over_the_counter field added to the TradeModes message.
3. The msgid field value changed for the Instrument message.
4. The borrowing_status field added to the Instrument message.
5. The trading_status field of the TradingInstrumentStatus message renamed to status.
6. Terminology changes.
7. Error codes added.

Version 1.12.0 November 30, 2016

1. New field markets 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) added and the msgid value changed in the Trade message.

Version 1.11.0 March 23, 2016

The Market message is added to the Instruments channel.

Version 1.10.1 February 16, 2016

Restrictions on LimitRequest are clarified.

Table of Contents

List of Tables

1. Service overview

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.

A user must utilize the Discovery service to connect to the risk management gateway. After the connection has been established, gateway data is available via topic subscriptions.

2. Interacting with gateway

2.1. Subscriptions

There are two subscription types—**logging** and **non-logging**. A logging stream transmits data, which is relevant, regardless of the time of receipt, such as list of transactions. A non-logging stream transmits the current status of constantly updated data.

A subscription is available in one of two modes—**snapshot** and **snapshot with subsequent updates**. For logged data, snapshot is the entire message history since the start of the trading day. For non-logging data, snapshot is aggregate of current status of all indicators. Updates are separate messages generated and transmitted to the client when an event occurs.

Messages of each stream have continuous numbering `topic_seq`. As client receives data in accordance with login access rights, numbering of messages sent to client may be discontinuous.

A subscription to streams is established by the client requests `TopicRequest` and `TopicCancel` and notifications `TopicReport` and `TopicReject`.

2.1.1. Subscription request

In order to get a subscription, the client should send `TopicRequest` to trading platform gateway with stream identifier `topic` and `mode` of data receipt.

The range of requested messages depends on the mode and type of the stream (by way of setting the first and last requested numbers). The client may indicate the lower range limit in the field `topic_seq` or the upper range limit in the field `topic_seqend` while requesting logging data. The client cannot specify the messages range while requesting for non-logging data.

In response to a valid request, the client will receive `TopicReport` and should expect messages with subscription data.

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

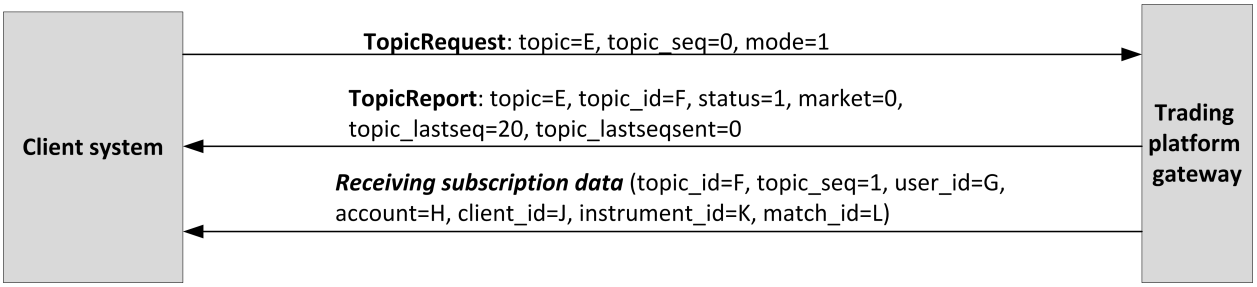

Figure 1. Request for subscription and data

2.1.2. Subscription canceling and resuming subscription

In order to stop receiving messages with subscription data, the client should send the subscription cancel request `TopicCancel` to trading platform gateway specifying stream identifier `topic` or `topic_id`.

In response to valid request, the client will receive notification `TopicReport` and subscription 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`.

All subscriptions are to be automatically canceled at disconnection.

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

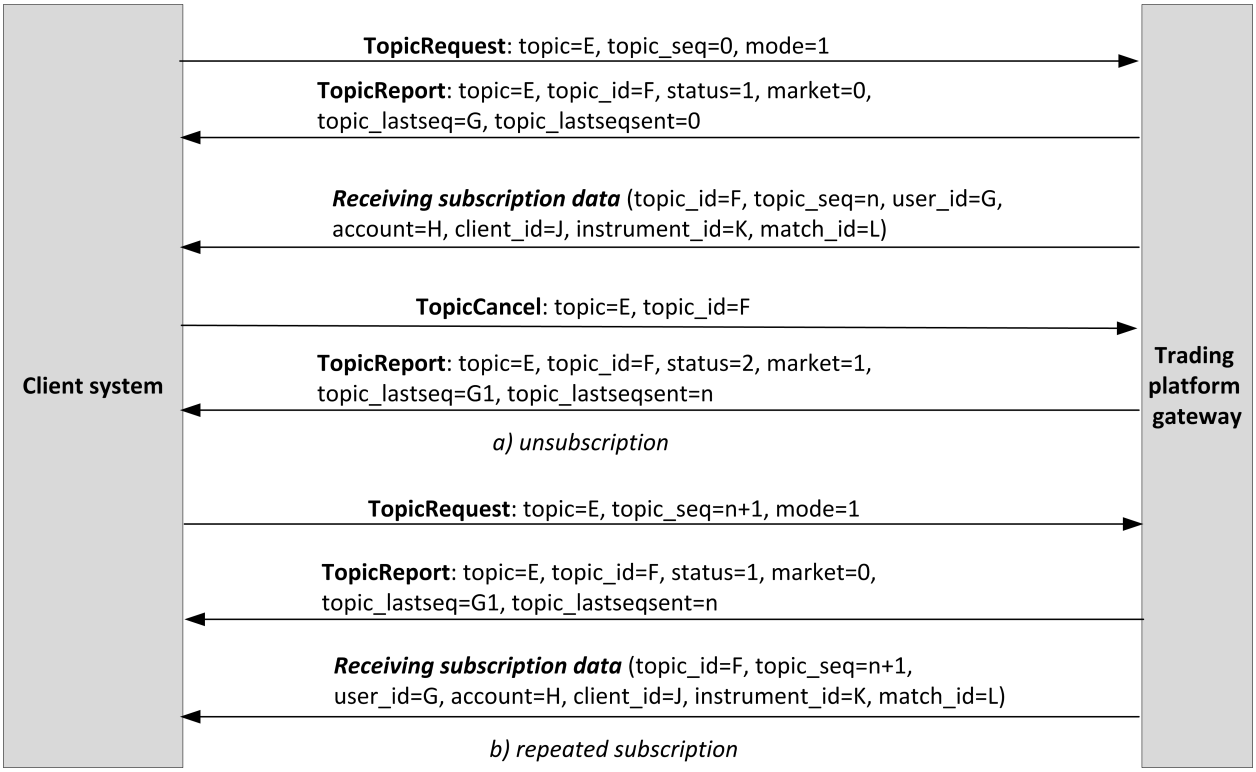

Figure 2. Subscription canceling and resuming subscription

2.2. Limit change

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

Limit can be increased or decreased for a client code, a group of client codes, an analytic clearing account or a clearing account. Client should specify the amount of limit change in the field `amount`.

A `LimitRequest` containing invalid field values will be rejected by `RejectReport`. In response to a valid request, the trading platform will send `LimitReport` to the client.

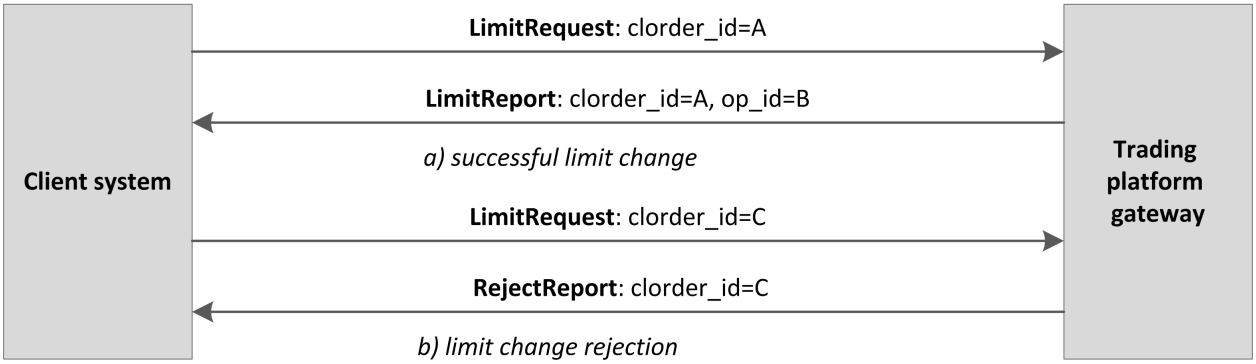

Figure 3. Request for limit change

3. Topics

3.1. Stream of clearing transactions and transfers

The topic of trades and transfers is logging with the `Transfer` and `Trade` messages broadcast. The topic ID is `topic=Trades`.

The `Trades.Transfer` and `Trades.Trade` topic are also available, and they transmit only a respective 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 1. Format of message `Transfer`: msgid=802, size=117

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	user_id	ascii16	Order sender 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 `Trade` message (on message processing please refer to section 4.4).

Table 2. Format of message `Trade`: msgid=814, dynamic length

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	user_id	ascii16	Order sender 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	Parameters depending on the market. Values: • `0x1` (eUserLastRec): last transaction message: last report on trades executed within a single transaction; • `0x2` (eMMObligations): indicator of market maker executing the obligations at internal exchange, to be assigned to visible limit orders; • `0x4` (eNoMMTrade): indictator of order by market maker that not to be matched with another market maker's order; • `0x8` (ePresettlement): pre-delivery trade; • `0x10` (eExternalActivity): transaction executed through external interfaces; • `0x20` (eDelivery): delivery trade; • `0x40` (eDeliverySwapGood): transfer of a bona fide participant during delivery; • `0x80` (eDeliverySwapBad): transfer of a mala fide participant during delivery; • `0x100` (eDeliveryDonorTrade): delivery transfer of participant with donor involvement; • `0x200` (eNoSystem): negotiated trade indicator; • `0x2000` (eIgnoreDynamicLimits): ignoring dynamic limits; • `0x40000` (eLimitedMargin): a sign of limited security; • `0x100000` (eClientPartialExecute): partial execution of address order sent by the client; • `0x200000` (eHaltPeriodOrder): marker of issuing an order during a suspension period; • `0x400000` (eOverTheCounter): marker of an order or a deal with over-the-counter instrument
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 the 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 in the trading system
197	exch_orderid	ascii20	Child order ID at liquity pool
217	exec_market	int2	Liquidity pool of execution (please refer to section 4.5.4)

Offset	Field	Datatype	Description
219	dir	int1	Side. Values: • `1` (Buy): buy; • `2` (Sell): sell
220	deals_offset	int2	Offset of the first `deals` entry from the beggining 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 beggining 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 beggining 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 beggining 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

3.1.1. 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 `Trade` message.

3.2. Topic of clearing positions

Clearing positions topic is non-logging with the `PositionUpdate` message broadcast. The topic ID is `topic=Pos.PositionUpdate`.

Positions are counted for several entities: client codes, groups of client codes, analytic clearing accounts, and clearing accounts. The type of entity will be specified in the `entity_type` field with the identifier of a specific entity in `entity_id`. A special parameter can be set in the `flags` field.

Since a clearing is executed without breaks in trading, operations related to the closed session may appear after the start of a new trading session. Such operations change only the value of the `clear_amount` field.

Table 3. Format of message `PositionUpdate`: msgid=851, dynamic length

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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 last session closing. May vary during the next 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 beggining 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 oblibation

3.3. Topic of funds

The funds topic is non-logging with the `FundsUpdate` message broadcast. The topic ID is `topic=Funds.FundsUpdate`.

Table 4. Format of message `FundsUpdate`: msgid=852, size=79

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	entity	[account_entity]	Component specifying entity
43	free	decn	Available funds
52	reserve	decn	Reserved funds
61	current	decn	Current funds
70	income	decn	Income

3.4. Topic of margin rates

The topic of margin rates is logging and consists only of updates, with the `RiskRates` broadcast. The topic ID is `topic=RiskRates`.

Table 5. Format of message `RiskRates`: msgid=810, size=78

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	balance_id	int8	Balance instrument identifier
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

3.5. Stream of risk parameters

The topic of risk parameters is non-logging, with the `RiskParams` message broadcast. The topic ID is `topic=RiskParams`.

Table 6. Format of message `RiskParams`: msgid=860, dynamic length

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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 beggining of this field
47	params_count	int2	Number of the `params` group entries
	> params	[topic_risk_param]	List of parameters

Table 7. 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

Field	Datatype	Description
result	decn	Current value

3.6. Topic of trading member references

The topic of trading member references broadcast the following messages:

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

Data transmitted in the topic is restricted by the login access permissions.

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

All the reference topics are non-logging.

The `User` message conveys properties of a login for connecting to the trading platform gateway (on message processing please refer to section 4.4).

Table 8. Format of message `User`: msgid=911, dynamic length

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	user_id	ascii16	Login, client gateway ID
38	type	int2	Login type. Values: • `1` (Clearing): clearing member; • `2` (WildCard): trading member; • `3` (Tag): login of group of clients marked by tag; • `4` (Group): login of groups of client codes; • `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: • `0` (No): no access; • `1` (Yes): has access
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 beggining of this field

Offset	Field	Datatype	Description
115	clearing_account_count	int2	Number of the `clearing_account` group entries
117	otccodes_offset	int2	Offset of the first `otccodes` entry from the beggining of this field
119	otccodes_count	int2	Number of the `otccodes` group entries
121	login_flags	int8	Login parameters. Values: • `0x1` (IS_ACTIVE): active login; • `0x8` (USE_ANY_GW): can ignore the list of permitted gateways; • `0x10` (USE_ANY_ACCOUNT): can use any clearing account of the member; • `0x20` (LEVEL_CM): level of clearing member; • `0x40` (LEVEL_TM): level of trading member; • `0x80` (LEVEL_CG): level of client group; • `0x100` (LEVEL_CLIENT): client code level; • `0x200` (LEVEL_TCA): level of clearing account; • `0x400` (IS_CM_OPERATOR): operator of clearing member; • `0x800` (IS_TM_OPERATOR): operator of the trading member; • `0x2000` (IS_SUSPENDED): login suspended by client's command
129	rights_flags	int8	Login permissions. Values: • `0x1` (M_TRADE): issue trading orders; • `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 external trading venues available through the trading platform.

Table 9. Format of message `OTCCode`: msgid=902, size=242

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	code	ascii16	Code for negotiated trading
38	market_id	int2	Liquidity pool ID (please refer to section 4.5.4
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 accounts, including links to clearing accounts in liquidity pools (on message processing please refer to section 4.4).

Table 10. Format of message `ClearingAccount`: msgid=903, dynamic length

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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: • `0` (No): virtual clearing account; • `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: • `0` (No): not under trust management; • `1` (Yes): under trust management
258	is_own_asset	int1	Own clearing account flag. Values: • `0` (No): not own clearing account; • `1` (Yes): own clearing account
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
299	segredation_type	int2	Method of available funds accounting. Values: • `0` (Custom): regular; • `1` (Private): dedicated; • `2` (Separate): isolated
301	exchange_accounts_offset	int2	Offset of the first `exchange_accounts` entry from the beggining 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 11. Format of message `Member`: msgid=904, size=259

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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: • `0` (Clearing): clearing member; • `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 4.4).

Table 12. Format of message `Client`: msgid=905, dynamic length

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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: • `0` (No): not under trust management; • `1` (Yes): under trust management
241	is_own_asset	int1	Own clearing account flag. Values: • `0` (No): not own clearing account; • `1` (Yes): own clearing account
242	has_client_group	int1	Indicatior of belonging to a group of client codes. Values: • `0` (No): does not belongs to a group; • `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 beggining of this field

Offset	Field	Datatype	Description
261	exchange_clients_count	int2	Number of the `exchange_clients` group entries
263	tag_offset	int2	Offset of the first `tag` entry from the beggining of this field
265	tag_count	int2	Number of the `tag` group entries
267	individual_investment_account	int1	Flag of individual investment account. Values: • `0` (No): non-individual investment account; • `1` (Yes): individual investment account
	> 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 4.4).

Table 13. Format of message `ClientGroup`: msgid=906, dynamic length

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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: • `0` (No): not under trust management; • `1` (Yes): under trust management
241	is_own_asset	int1	Own clearing account flag. Values: • `0` (No): not own clearing account; • `1` (Yes): own clearing account
242	tag_offset	int2	Offset of the first `tag` entry from the beggining of this field
244	tag_count	int2	Number of the `tag` group entries
	> tag	char15+1	Group of client codes marked by tag

3.7. Topic of instrument references

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

- `Currency` balance instrument,
- `Issue` balance instrument,
- `Spot` balance instrument,

- `Futures` balance instrument,
- `Bond` balance instrument,
- `TradeModes`,
- `Markets` liquidity pools,
- trading `Instrument`.

The parent topic broadcasts the `TradingInstrumentsStatus` notification and updates of price limit changes for orders. 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 subscribe to a child topic of only one reference. Such child thread has its own numbering `topic_seq`, and its identifier `topic` is as follows `Instruments.Instrument`.

All the reference topics are non-logging.

Table 14. Format of message `Currency`: msgid=931, size=278

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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: • `0` (REAL): Real; • `1` (TEST): Test

Table 15. Format of message `Issue`: msgid=932, size=486

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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

Offset	Field	Datatype	Description
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
443	security_type	int1	Security type. Values: • `1` (OrdinaryShare): ordinary share; • `2` (PreferredShare): preferred share; • `5` (ETF): security of foreign exchange traded fund; • `6` (RDR): Russian depositary receipt; • `7` (ADR): American depositary receipt; • `8` (GDR): global depositary receipt; • `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: • `0` (REAL): Real; • `1` (TEST): Test

Table 16. Format of message `Spot`: msgid=933, size=293

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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

Offset	Field	Datatype	Description
292	is_test	int1	Flag of test instrument. Values: • `0` (REAL): Real; • `1` (TEST): Test

Table 17. Format of message `Futures`: msgid=934, size=292

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	balance_id	int4	Balance instrument ID
26	code	char32+1	Futures 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	date_expire	time8m	Expiration date
286	underlying_id	int4	Underlying instrument ID
290	exec_type	int1	Futures type. Values: • `0` (FuturesThroughSpot): futures through spot; • `1` (FuturesCashSettlement): cash-settled futures
291	is_test	int1	Flag of test instrument. Values: • `0` (REAL): Real; • `1` (TEST): Test

Table 18. Format of message `Bond`: msgid=935, dynamic length

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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

Offset	Field	Datatype	Description
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 beggining 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: • `1` (GovernmentBond): government bond; • `2` (MunicipalBond): municipal bond; • `3` (CentralBankBond): Central bank bond; • `4` (CorporateBond): corporate bond; • `5` (FinancialInstitutionBond): financial institution bond
456	issue_date	time8m	Date of issue
464	quotation_list	char32+1	Quotation list
497	is_test	int1	Flag of test instrument. Values: • `0` (REAL): Real; • `1` (TEST): Test
	> coupon_payment	[coupon_payment]	Schedule of coupon payments

Table 19. Format of message `TradeModes`: msgid=942, size=222

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	trade_mode_id	int2	Trade mode ID
24	name	char64+1	Name of trading mode in English
89	name_ru	char128+1	Name of trading mode in Russian

Offset	Field	Datatype	Description
218	is_address	int1	Negotiated trading flag in trading mode. Values: • 0 (No): non-negotiated; • 1 (Yes): negotiated
219	is_multileg	int1	Multi-leg trade indicator. Values: • 0 (No): sigle-leg; • 1 (Yes): multi-leg
220	is_ext_close	int1	Closing auction indicator. Values: • 0 (No): not traded at closing auction; • 1 (Yes): traded at closing auction
221	over_the_counter	int1	Over-the-counter trade mode indicator. Values: • 0 (No): not present; • 1 (Yes): present

Table 20. Format of message `Market`: msgid=936, size=220

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	market_id	int4	Market ID
26	desc	char64+1	Full name in English
91	desc_ru	char128+1	Full name in Russian

Table 21. Format of message `Instrument`: msgid=973, dynamic length

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic 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

Offset	Field	Datatype	Description
257	type	char3+1	Trading instrument type: • f: futures; • t: T+N; • o: option; • r: repo; • pr: related trades; • sw: swap; • c: calendar spread; • sf: spot-futures spread; • dvp: delivery versus payment
261	auction_dir	int1	Type of auction. Values: • `0` (Direct): direct auction; • `1` (Inverse): inverse auction
262	price_increment	dec8	Price increment
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: • `0` (NoScalping): no scalping; • `1` (Custom): custom scalping; • `2` (InverseScalping): inverse scalping
284	fee_schema	int1	Fee scheme. Values: • `1` (MakerTakerSpot): maker-taker for spot; • `2` (MakerTakerFutures): maker-taker for futures; • `3` (REPO): repo
285	fee_rate_offset	int2	Offset of the first `fee_rate` entry from the beggining 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 beggining 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 beggining 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

Offset	Field	Datatype	Description
330	is_test	int1	Flag of test instrument. Values: • 0 (REAL): Real; • 1 (TEST): Test
331	te_id	int2	Trading engine ID
333	be_mode	int1	Best execution mode. Values: • 0 (External): external trades; • 1 (Internal): internal trades at external prices
334	borrowing_status	int1	Short selling availability for the instrument. Values: • 1 (HARD_TO_BORROW): Short selling unavailable; • 2 (EASY_TO_BORROW): Short selling available
	> fee_rate	dec8	Fee rate
	> periods	[Period]	Group of trading periods (such as trading session) for instrument
	> exchange_instrument	[ExchangeInstrument]	Group specifying trading instruments at liquidity pools

In this version of the trading platform, 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 .

Table 22. Format of message `TradingInstrumentStatus`: msgid=2031, size=96

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	instrument	[instrument]	Component specifying trading instrument

Offset	Field	Datatype	Description
28	status	int1	Current status of trading instrument. Values: • `2` (HALT): trading is halted; • `17` (TRADING): instrument is traded; • `18` (NO_TRADING): trading finished or not started yet; • `102` (CLOSE): instrument is traded at closing auction; • `103` (CLOSE_PERIOD): close period trading; • `107` (DISCRETE_AUCTION): instrument is traded at discrete auction; • `118` (OPEN): instrument is traded at opening auction; • `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 23. Format of message `TradingInstrumentLimits`: msgid=2032, size=42

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

Table 24. Format of message `BorrowingStatus`: msgid=2033, size=27

Offset	Field	Datatype	Description
	[frame]	[frame]	Session header
0	[topic_header]	[topic_header]	Topic header
22	instrument_id	int4	Trading instrument identifier
26	borrowing_status	int1	Short selling availability for the instrument. Values: • `1` (HARD_TO_BORROW): Short selling unavailable; • `2` (EASY_TO_BORROW): Short selling available

4. Native protocol specification

4.1. Datatypes

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.

4.2. 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 *Network Connectivity*.

After establishing connection with the Discovery service, the client should send the `Hello` message. The IP address of the client must be authorized for the specified login (user ID); otherwise, the connection request will be rejected. The message contains the session header `frame` (for more details refer to section 4.3.1).

Table 25. Format of request `Hello: msgid=1, size=32, seq=0`

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 group `addresses`; the number of group records will be specified in the field `addresses_count` (for more details on processing of repeating groups please see section 4.4). The group includes fields `type` (gateway attribute) and `addresses` (host address and gateway port). Gateway attributes may combine.

For some time after the trading system response, the gateway will expect the client's login connection to the specified address. In case of failure, the client should 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 26. Format of response `Report: msgid=2, seq=0`, dynamic length

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

Offset	Field	Datatype	Description
0	status	int2	Request status. Values: • `0` (success), • `1` (reject due to invalid login/password)
2	reason	char127+1	Textual description
130	addresses_off-set	int2	Offset of the first underlying entry from the beginning of this field. Value: `4`
132	address-es_count	int2	Number of `addresses` group entries
	> [addresses]	[address-es]	Addresses list

Table 27. Format of component `addresses`: size 52 bytes

Field	Datatype	Description
type	int2	Gateway attributes, bit mask. Values: • `0x0` (No); • `0x1` (Trading); • `0x2` (Drop copy); • `0x4` (Risk management); • `0x8` (Dictionaries); • `0x10` (Market data); • `0x4000` (Backup)
ver	int1	Protocol version
pad0	int1	Reserved field, filled with zero bytes
address	char47+1	Address of host and gateway port

4.3. General session layer

4.3.1. Message generation and transmission

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

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

Table 28. 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

4.3.2. 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 server 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 29. 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: • `0` (no): sequence numbers continue; • `1` (yes): sequence numbers reset
33	heartbeat_ms	int4	`Heartbeat` frequency in milliseconds

Table 30. 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. 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

4.3.3. Heartbeats

The client and the gateway exchange `Heartbeat` messages to monitor the connection status. Heartbeat is 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 inactivity for a period longer than the specified interval, the server will break the connection. The client is expected to do the same, if inactivity is detected on the part of the server.

Table 31. Format of message `HeartBeat`: msgid=8103, size=0

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

4.3.4. 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.

4.3.5. Message sequence number reset

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 32. 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

4.3.6. Message resend request

If the client receives from the server a message with the number higher than expected, the client should either reset the counter or request missing messages from the server by `ResendRequest`.

Messages sent during the current and previous trading days are available for client resend requests. If the client forcefully resets message numbering (`reset_seq=1` in `Login`), a request for resending messages sent prior to this reset is not possible.

The `ResendRequest` must specify the first message within requested messages range (`from_seq`) and the last message (`till_seq`). If the client uses `from_seq=0` and `till_seq=0`, the gateway will resend messages starting from the lowest number available. If the client uses field `till_seq=0`, the server will resend all messages of the current trading session starting from the number specified in field `from_seq`. All possible cases are listed hereinafter:

1. `from_seq=n, till_seq=m` (request for messages from *n* to *m*),
2. `from_seq=0, till_seq=n` (request for messages from the lowest number available to *n*),
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).
5. `from_seq=-1, till_seq=0` (request for all available messages for the current trading day but not exceeding the maximum available number).
6. `from_seq=-2, till_seq=0` (request for all available messages for the previous and current trading days but not exceeding the maximum available number; if messages for one of the trading days are not available, the trading system will return an error).

It is recommended to use the query `from_seq=0, till_seq=0` at the first connection after a long break. If after resending, the gateway returns `ResendReport` with the `MORE` status, the client should send another request specifying `from_seq` with the number following the last resent message and `till_seq=0`.

The number range for requested messages is not limitless (for more details please refer to *Network Connectivity*). If requiring more messages, the client should send several consecutive requests. Any new request sent prior to the resend

completion is to be rejected by the gateway with `ResendReport` indicating the `DUPLICATE_REQUEST` status. If the query is `till_seq=0`, the gateway is to transmit messages not exceeding its maximum number.

Table 33. 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 correct request, the trading system will transmit `ResendReport` indicating the `ACK` status and requested messages. Upon completion of transmission, the gateway will send `ResendReport` conveying `MORE` or `FINISH`. The status `MORE` indicates that the number of the last message within the range is less than the number of the last trading message sent by the gateway; that is, there are messages of application level not included in the request and they could have been generated during the request execution.

While resending, the server may also transmit new trading messages, so client should also expect message with a number exceeding the requested range.

Table 34. Format of message `ResendReport`: msgid=8105, size=2

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

4.3.7. Reset of expected sequence number

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 messages from stream of non-logging data.

Table 35. 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

4.3.8. Session termination

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

Table 36. Format of message `Logout`: msgid=8002, size=16

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

Offset	Field	Datatype	Description
0	login	ascii16	Login, client gateway ID

4.3.9. Message rejection

If the client's message is either malformed or contains invalid values, the server 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 37. 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

4.3.10. Disconnection

Server 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.

4.3.11. Subscription management

Subscription management messages do not belong to application level and have no message number `seq` assigned. Nevertheless, messages with subscription data have number `seq`.

Each message with the data generated by the trading platform in a subscription stream is assigned with the `topic_seq` number starting from 1. As the client receives messages in accordance with login access rights, numbering of messages sent to client will be discontinuous.

Subscription management messages have no `topic_seq` number.

4.3.11.1. Subscription request

In order to request an update subscription or one-time snapshot, the client should send `TopicRequest` to the trading platform gateway specifying `topic` ID and `mode` (snapshot or snapshot and updates). The client does not have to fill the `clorder_id` field.

As for logging data stream, the client may specify the first number of requested data `topic_seq`. For a non-logging stream, `topic_seq` is 0 in a request. When making an initial subscription request for logging stream, the client should use 0, and in a repeating request, the number shall be one more than that of the last message received.

When requesting a logging stream snapshot (`mode=0`), the client may request messages up to a definite number to be specified in the `topic_seqend` field.

When requesting a non-logging stream snapshot, the client (`mode=1`) should specify 0 value in `topic_seqend`.

If a request is incorrect or cannot be processed, the gateway will answer with `TopicReject`. Otherwise, the client will receive `TopicReport` and after that should expect topic data messages. In case of requesting a non-logging stream snapshot, such requested messages may be sent along with messages with updates.

Table 38. 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 code
84	topic_seq	int8	First number of requested topic data
92	topic_seqend	int8	Last number of requested topic data
100	mode	int1	Data request mode. Values: • `0` (DATA_SLICE): snapshot; • `1` (SUBSCRIBE): snapshot and subsequent updates

4.3.11.2. Unsubscription

The client shall send `TopicCancel` to the trading platform gateway, to stop receiving messages with subscription data, specifying one or both stream identifiers—`topic` and `topic_id`.

If a request is incorrect or cannot be executed, the gateway will respond with `TopicReject`. In case of successful request processing, the subscription will be canceled and the client will receive notification `TopicReport` with `status=2` and the subscription will be terminated; still for some time after notification client may continue receiving messages with data.

Table 39. 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 code
84	topic_id	int4	Numerical code of topic

4.3.11.3. Subscribe or unsubscribe rejection

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

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

Table 40. 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 code
110	topic_id	int4	Numerical code of topic

Offset	Field	Datatype	Description
114	status	int2	Subscription status. Values: • 0 (DATA_SLICE): snapshot; • 1 (ACTIVE): active subscription; • 2 (INACTIVE): inactive subscription
116	reason	int2	Reason for rejection. Values: • 1 (BAD_TOPIC): invalid topic identifier; • 2 (ALREADY_SUBSCRIBED): subscription is already active; • 3 (NOT_SUBSCRIBED): subscription not activated; • 4 (DATA_NOT_AVAILABLE): data not available; • 5 (DUPLICATE_REQUEST): repeated request; • 6 (BAD_SEQ): non-existent number in topic; • 7 (BAD_MODE): invalid mode
118	topic_firstseq	int8	Number of first available message
126	topic_lastseq	int8	Last number of topic data
134	topic_lastseqsent	int8	Last number of topic data sent to client

4.3.11.4. Subscription notification

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

- successful execution of a request for a new subscription;
- successful execution of request for removing a subscription;
- completion of snapshot transmission under a subscription.

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

Table 41. 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 code
110	topic_id	int4	Numerical code of topic
114	status	int2	Subscription mode. Values: • 0 (DATA_SLICE): snapshot; • 1 (ADD_SUBSCRIBE): new subscription; • 2 (DEL_SUBSCRIBE): cancellation of subscription
116	marker	int2	Indicator of start and finish of data transfer. Values: • 0 (START): start of data transfer; • 1 (END): end of the data transfer; • 2 (SLICE_END): snapshot transfer completed

Offset	Field	Datatype	Description
118	topic_lastseq	int8	Last number of message with topic data
126	topic_lastseqsent	int8	Last number of topic data sent to client under subscription

4.4. Processing messages with repetitive components and fields

Several message types contain one or more repeating groups 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 group entries. The `offset` field indicates an offset in bytes of first entry of the group from the beginning of this very field; its value is no less than 4.

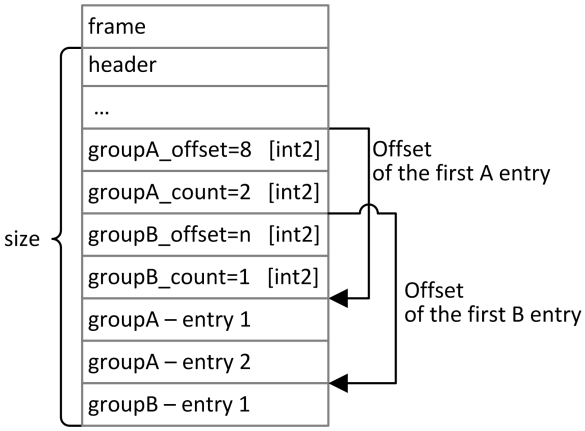

Figure 4. Template of a message with two repeating components

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

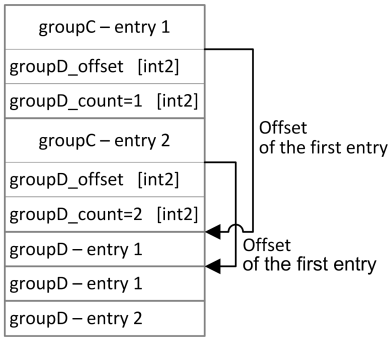

Figure 5. Template of embedded components

4.5. Application level

4.5.1. Changing client limits

4.5.1.1. Request for limit change

The client should send `LimitRequest` to the trading platform gateway to change client instrument limits.

The request should contain the identifier of a balance instrument, which limit is to be changed, in the `balance_id` field. (For balance and trading instruments please refer to *Trading Instruments Specification*).

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

Limit can be decreased or increased by the specified `amount`.

A request `LimitRequest`, containing invalid data, will be rejected by `RejectReport`. In response to a valid request, the trading platform will send `LimitReport` to the client.

Table 42. 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: • `1` (Enrolment): deposit; • `2` (Withdrawal): withdrawal
50	flags	int8	Bit mask of limit change. Values: • `0x100` (FORCED_UPDATE): not to verify a non-increase in initial margin (IM) arrears (“hard” withdrawal); • `0x400` (FORCED_ASSET_UPDATE): not to verify asset presence in case of withdrawal
58	amount	decn	Volume of limit change

4.5.1.2. Report on changing limit

After a change of limit in the trading platform as result of `LimitRequest`, the client will be sent a report on limit changing `LimitReport`, which contains a unique operation identifier `op_id` and a new limit value `amount_rest`.

Table 43. 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: • `1` (Enrolment): deposit; • `2` (Withdrawal): withdrawal

Offset	Field	Datatype	Description
76	flags	int8	Bitmask of limit change. Values: • `0x100` (FORCED_UPDATE): not to verify a non-increase in initial margin (IM) arrears (“hard” withdrawal); • `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

4.5.1.3. Report on rejection

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

Table 44. 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
48	reason	int2	Code of rejection reason
50	message	char32+1	Rejection code parameters and or its textual description
83	extra_data0	int8	Reserved field. To be filled with null byte

4.5.2. Formats of common components

Table 45. 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 46. Format of component `account_entity`: length 21 bytes

Field	Datatype	Description
member_id	int4	Trading member ID
entity_id	ascii16	Entity ID

Field	Datatype	Description
entity_type	int1	Type of entity. Values: • 0 (CLIENT): client code; • 1 (CLIENT_GROUP): group of client codes; • 2 (CLEAR_ACC): analytical clearing account; • 3 (PRINCIPAL_CLEAR_ACC): clearing account

Table 47. 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 4.5.3)
clorder_id	ascii20	Client order ID
user_id	ascii16	Login, client gateway ID

Table 48. Format of component `otccodes`: length 32 bytes

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

Table 49. Format of component `topic_header`: length 22 bytes

Field	Datatype	Description
topic_id	int4	Numerical code 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 4.5.3)

Table 50. Format of component `user_header`: length 20 bytes

Field	Datatype	Description
clorder_id	ascii20	Client order ID

Table 51. Format of component `clr_deal`: length 85 bytes

Field	Datatype	Description
deal_id	int8	Trade transaction ID
clr_deal_id	int8	Trade ID
traded_balance_id	int8	Trading balance instrument ID
measuring_balance_id	int8	Balance instrument measuring price ID

Field	Datatype	Description
clr_deal_price	dec8	Trade price
amount	decn	Trade volume
volume	decn	Volume of trade, part one, in settlement currency
dir	int1	Side of trade. Values: • `1` (Buy): deposit; • `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 52. Format of component `clr_repo_deal`: length 126 bytes

Field	Datatype	Description
deal_id	int8	Trade transaction ID
clr_deal_id	int8	Trade ID
traded_balance_id1	int8	Trading balance instrument ID. 1st part of the repo
measuring_balance_id1	int8	Balance instrument measuring price ID. 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 measuring price ID. 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_amount	decn	Buyback amount
buyback_price	dec8	Buyback price
dir	int1	Direction of transaction. Values: • `1` (Buy): deposit; • `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 53. Format of component `coupon_payment`: length 16 bytes

Field	Datatype	Description
date	time8m	Date of payment

Field	Datatype	Description
value	dec8	Amount of payment

Table 54. 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 55. Format of component `ExchangeAccount`: length 36 bytes

Field	Datatype	Description
market_id	int2	Liquidity pool ID (please refer to section 4.5.4)
type	int2	Method to identify a clearing account. Values: • `1` (Standart): standard; • `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 56. Format of component `ExchangeClient`: length 18 bytes

Field	Datatype	Description
market_id	int2	Liquidity pool ID (please refer to section 4.5.4)
client_name	ascii16	Client code ID at liquidity pool

Table 57. 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 58. Format of component `extra_data`: length 11 bytes

Field	Datatype	Description
type	int2	Parameter type
value	decn	Parameter value

Table 59. Format of component `instrument`: length 6 bytes

Field	Datatype	Description
market_id	int2	Liquidity pool ID (please refer to section 4.5.4)
instrument_id	int4	Trading instrument ID

Table 60. Format of component `instrument_status`: length 4 bytes

Field	Datatype	Description
trading_status	int1	Current status of trading instrument. Values: • `2` (HALT): trading is halted; • `17` (TRADING): instrument is traded; • `18` (NO_TRADING): trading finished or not started yet; • `102` (CLOSE): instrument is traded at closing auction; • `103` (CLOSE_PERIOD): close period trading; • `107` (DISCRETE_AUCTION): instrument is traded at discrete auction; • `118` (OPEN): instrument is traded at opening auction; • `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 61. Format of component `t_OTCCode`: length 18 bytes

Field	Datatype	Description
code	ascii16	Negotiated trading ID
market_id	int2	Liquidity pool ID (please refer to section 4.5.4)

Table 62. Format of component `Period`: length 30 bytes

Field	Datatype	Description
start	time8m	Start timestamp
finish	time8m	End timestamp
mode	int2	Type of auction. Values: • `0` (ProRata): pro rata two-way anonymous auction; • `1` (Parity): parity two-way anonymous auction; • `2` (TimePriority): time priority anonymous auction; • `3` (Address): negotiated trading; • `4` (OpenAuction): opening auction; • `5` (CloseAuction): closing auction; • `6` (NoTrade): no trading; • `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 beggining of this field
underlying_count	int2	Number of the `underlying` group entries
markets_offset	int2	Offset of the first `markets` entry from the beggining of this field
markets_count	int2	Number of the `markets` group entries
> underlying	[Underlying]	Group for trading instrument lot volume specification within a period of time
> markets	int2	List of available liquidity pools (please refer to section 4.5.4)

Table 63. 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	Direction of transfer. Values: • `1` (Buy): deposit; • `2` (Sell): withdrawal
transfer_type	int1	Type of transfer. Values: • `1` (Trade): trade result; • `2` (LimitChange): limit change for client code or group of client codes
flags	int8	Parameters of transfer. Values: • `0x100` (not to verify a non-increase in arrears (“hard” withdrawal)); • `0x200` (transfer by member’s command); • `0x1000` (deposit-withdrawal by client’s order); • `0x2000` (deposit-withdrawal by administrator’s order); • `0x4000` (transfer generated in the process of clearing); • `0x1000000` (applied for clearing account); • `0x4000000` (applied to client code); • `0x8000000` (applied for group of client codes); • `0x10000000` (applied for analytical clearing account)
amount	decn	Volume of transfer

Table 64. 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: • `0x1` (CORP_DUE_BILL): additional liability in connection with corporate event; • `0x2` (CORP_CORRECTION): liability adjustment by clearing center in connection with corporate event; • `0x4` (CORP_INCOME_RETURN): transfer of income in connection with corporate event; • `0x8` (PRINCIPAL_OBLIGATION): principal liability flag

4.5.3. Values `source_id`

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

Table 65. Values `source_id` 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 pool identifiers

4.5.4. Liquidity pool identifiers

Liquidity pools' identifiers may be in fields `market` and `exec_market`.

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

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

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

`1010` — liquidity pool of Moscow Exchange.

`1015` — execution at United States liquidity pools.

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

`1030` — liquidity pool of NYSE.

`1031` — liquidity pool of ARCA.

`1032` — liquidity pool of NASDAQ.

`1033` — liquidity pool of BATS.

Appendix A. Error codes

Table 66. Error codes list

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

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

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

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

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

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

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

Appendix B. Revision History

Version 1.10.0 December 24, 2015

1. Added component extra_data to the message PositionUpdate.
2. The field amount_rest_extra is removed from message Trade, 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 field extra_ref added and size of field extra1 changed in message Trade.
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 3.4).
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 4.3.6).
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 message Trade 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 4.3.4).

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.