Order Gate

Order Gate operates FIX protocol over TCP transport. FIX protocol is encoded via Simple Binary Encoding (or just SBE). Unless specified explicitly, specifications of these standards should be taken into account (particularly for terminology) for the rest of the chapter.

Since the gate's performance does matter, only statically-known types are used in message templates. Also, it is important to notice that this manual may list tags or constants that may differ from the FIX specification.

Types

The schema uses the following types:

- signed integers (int8, int16, int32, int64)
- unsigned integers (uint8, uint16, uint32, uint64)
- decimal (decimal9)
- fixed-sized ascii string (string4, string20)
- aliases (`DurationNanosecs`, `Timestamp`)
- enumerations (`TerminationCode`, `EstablishmentRejectCode`, ...)

Integers are encoded in little-endian, and the number after the type (e.g 16 in uint16) specifies the number of bits used for an integer. Here are some encoding examples:

Number	Type	Encoded (hex)
123	int8	`7b`
-123	int8	`85`
12345	int16	`39 30`
-12345	int16	`c7 cf`
12345678	int32	`4e 61 bc 00`
-12345678	int32	`b2 9e 43 ff`
1234567891234567	int64	`07 af 9b 3c d5 62 04 00`
-1234567891234567	int64	`f9 50 64 c3 2a 9d fb ff`
123	uint8	`7b`
12345	uint16	`39 30`
12345678	uint32	`4e 61 bc 00`
1234567891234567	uint64	`07 af 9b 3c d5 62 04 00`

Mantissa of a decimal is encoded as int64. The number after the type (e.g 9 in decimal9) specifies the fixed exponent with the minus sign. Here are some encoding examples:

Number	Type	Encoded (hex)
1	decimal9	`00 ca 9a 3b 00 00 00 00`
1234567.891234567	decimal9	`4e 16 13 39 f0 6c 04 00`
-1234567.891234567	decimal9	`f9 50 64 c3 2a 9d fb ff`

Fixed-sized ASCII strings are encoded as an array of ASCII characters (in range [0x20; 0x7f]), and the length of the array is specified at the end of the type name (e.g 4 in string4). Here are some encoding examples:

String	Type	Encoded (hex)
00000000	string4	`00 00 00 00`
ABCD	string4	`41 42 43 44`

An alias is basically an alternative name to another type whose encoding is already defined.

- `DurationNanosecs` is an alias for uint64
- `Timestamp` is an alias for uint64

All enumerations are encoded as uint8, thus forbids to have more than 256 variants. Its values are defined later.

All types have valid ranges and a Null magic value, which is being used for unset optional fields. The table for minimum, maximum, and null values for different types is specified below:

Type	Null Value	Min Value	Max Value
int8	-128	-127	127
int16	-32768	-32767	32767
int32	-2147483648	-2147483647	2147483647
int64	-9223372036854775808	-9223372036854775807	9223372036854775807
uint8	255	0	254
uint16	65535	0	65534
uint32	4294967295	0	4294967294
uint64	18446744073709551615	0	18446744073709551614

`Price`, `Amount` and `Leverage` types are decimals with constant exponent.

Composite type value is considered to be Null if the first field is Null.

Header

All messages can be split into two main groups: Session and Application layer messages. Session layer messages provide utility for connection maintenance, and application layer messages allow concrete trading actions.

Every message contains a fixed message header:

Field name	Type	Description
`blockLength`	uint16	message body length
`templateId`	uint16	message template identifier
`schemaId`	uint16	schema identifier
`version`	uint16	version number

The protocol described in this document uses the following constants:

- `schemaId` = 7172

Messages

Establish (templateId = 5000)

The message is used for creating a session for the current TCP connection

Tag	Field name	Required	Type	Description
20004	KeepaliveInterval	+	DurationNanosecs	Requested heartbeat interval
20005	LoginId	+	LoginId	Client login id

EstablishmentAck (templateId = 5001)

Confirmation of session creation, as a response to the Establish message

Tag	Field name	Required	Type	Description
20004	KeepaliveInterval	+	DurationNanosecs	Session heartbeat interval
20006	NextSeqNo	+	UInt64	Sequence number of the next message

EstablishmentReject (templateId = 5002)

Denial of session creation, as a response to the Establish message

Tag	Field name	Required	Type	Description
20007	EstablishmentRejectCode	+	EstablishmentRejectCode	

Terminate (templateId = 5003)

Terminates session

Tag	Field name	Required	Type	Description
20008	TerminationCode	+	TerminationCode	

RetransmitRequest (templateId = 5004)

Request for message retransmission

Tag	Field name	Required	Type	Description
20009	FromSeqNo	+	UInt64	Starting sequence number
20010	Count	+	UInt32	Number of messages to retrieve

Retransmission (templateId = 5005)

Message for notifying that the following Count messages are responses for RestransmitRequest

Tag	Field name	Required	Type	Description
20006	NextSeqNo	+	UInt64	First returned sequence number
20010	Count	+	UInt32	Number of messages

RetransmitReject (templateId = 5006)

Message for notifying that the following Count messages are responses for RestransmitRequest

Tag	Field name	Required	Type	Description
20011	Reason	+	RetransmitRejectCode	Reject reason

Sequence (templateId = 5007)

Heartbeat message. Client should set NextSeqNo to the nullValue. Server sets NextSeqNo to the next message sequence number (counted by application-level messages)

Tag	Field name	Required	Type	Description
20006	NextSeqNo	C	UInt64	Sequence number of the next message

FloodReject (templateId = 5008)

Message for notifying that message flooding has been detected

Tag	Field name	Required	Type	Description
11	RequestId	+	RequestId	Request id of the message leading to flood
20012	QueueSize	+	UInt32	Number of received messages for the past second
20013	PenaltyRemain	+	DurationNanosecs	Time left until end of block, in nanoseconds

MessageReject (templateId = 5009)

The message is sent in response to invalid client messages

Tag	Field name	Required	Type	Description
11	RequestId	+	RequestId	Request id of rejected message
371	RefTagId	C	UInt32	Invalid field identifier
373	Reason	+	MessageRejectReason	Reject reason code

OrderPlaceRequest (templateId = 6001)

Place new order

Tag	Field name	Required	Type	Description
20002	TraceId	C	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	Instrument identifier
44	Price	C	Price	Price (null for market order)
38	Size	+	Size	Order quantity
40	Type	+	OrderType	Order type
59	TimeInForce	+	TimeInForce	
54	Side	+	Side	

OrderCancelRequest (templateId = 6002)

Cancel previously placed order

Tag	Field name	Required	Type	Description
20002	TraceId	C	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
37	OrderId	+	OrderId	Order identifier

OrderReplaceRequest (templateId = 6003)

Replace previously placed order

Tag	Field name	Required	Type	Description
20002	TraceId	C	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
37	OrderId	+	OrderId	Order identifier to replace
44	Price	C	Price	New price (null for market order)
38	Size	C	Size	New size
40	Type	C	OrderType	Order type
59	TimeInForce	C	TimeInForce	

OrderMassCancelRequest (templateId = 6004)

Cancel multiple orders by filter

Tag	Field name	Required	Type	Description
20002	TraceId	C	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
1	AccountId	C	AccountId	Account identifier
48	InstrumentId	C	InstrumentId	Instrument identifier
54	Side	C	Side	

BalanceFetchRequest (templateId = 8000)

Request balance information

Tag	Field name	Required	Type	Description
20002	TraceId	C	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
1	AccountId	+	AccountId	Account identifier
15	Currency	+	Currency	

PositionFetchRequest (templateId = 8001)

Request position information

Tag	Field name	Required	Type	Description
20002	TraceId	C	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	Instrument identifier

RiskParamsFetchRequest (templateId = 8002)

Request to retrieve current risk parameters

Tag	Field name	Required	Type	Description
20002	TraceId	C	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	Instrument identifier

RiskParamsChangeRequest (templateId = 8003)

Request to change risk limit

Tag	Field name	Required	Type	Description
20002	TraceId	C	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	Instrument identifier
20014	MarginMode	C	MarginMode	
20015	Leverage	C	Leverage	Must be set when MarginMode is set to Isolated
20016	RiskLimit	C	Amount	

IsolatedMarginChangeRequest (templateId = 8004)

Request to change isolated margin

Tag	Field name	Required	Type	Description
20002	TraceId	C	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	Instrument identifier
20017	DeltaMargin	+	Amount	

OrderReport (templateId = 7000)

Notifies that an order was successfully placed as a response to OrderPlaceRequest

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	C	RequestId	Client request identifier
20003	TradingTimestamp	+	Timestamp	
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	Instrument identifier
37	OrderId	+	OrderId	
44	Price	C	Price	Order price (null for market order)
38	Size	+	Size	
40	Type	+	OrderType	
59	TimeInForce	+	TimeInForce	
54	Side	+	Side	

OrderPlaceReject (templateId = 7001)

Notifies that an order was rejected as a response to OrderPlaceRequest

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
103	Reason	+	OrderRejectReason	

OrderCancelReport (templateId = 7002)

Notifies that an order was canceled as a response to OrderCancelRequest or for other reason

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	C	RequestId	Client request identifier
20003	TradingTimestamp	+	Timestamp	
37	OrderId	+	OrderId	
20018	Reason	+	CancelReason	

OrderCancelReject (templateId = 7003)

Notifies that an order cancel was rejected as a response to OrderCancelRequest

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
103	Reason	+	OrderRejectReason	

OrderReplaceReport (templateId = 7004)

Notifies that an order was successfully replaced as a response to OrderReplaceRequest

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	C	RequestId	Client request identifier
20003	TradingTimestamp	+	Timestamp	
37	NewOrderId	+	OrderId	
44	Price	C	Price	New price (null for market order)

Tag	Field name	Required	Type	Description
38	Size	+	Size	New size
20019	OldOrderId	+	OrderId	

OrderReplaceReject (templateId = 7005)

Reject for replaced order as a response to OrderReplaceRequest

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier
103	Reason	+	OrderRejectReason	

OrderMassCancelReport (templateId = 7006)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	+	RequestId	Client request identifier

OrderMassCancelReject (templateId = 7007)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	+	RequestId	
103	Reason	+	OrderRejectReason	

ExecutionReport (templateId = 7008)

Report for an order being matched

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
20003	TradingTimestamp	+	Timestamp	Trading system time
1003	TradeId	+	TradeId	Trade identifier
31	TradePrice	+	Price	Trade price
32	TradeSize	+	Size	Trade size
37	OrderId	+	OrderId	Public order identifier
38	OrderSize	+	Size	Remaining size of the order

EmptyBookReport (templateId = 7009)

Book was emptied

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
20003	TradingTimestamp	+	Timestamp	
48	InstrumentId	+	InstrumentId	Instrument identifier

TradingStatusReport (templateId = 7010)

Message for trading status events notification

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
20003	TradingTimestamp	+	Timestamp	
48	InstrumentId	+	InstrumentId	Instrument identifier
340	Status	+	TradingStatus	

TradeReport (templateId = 9000)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
20003	TradingTimestamp	+	Timestamp	
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	Instrument identifier
1003	TradeId	+	TradeId	
20020	Type	+	TradeType	
38	Size	+	Size	
31	Price	+	Price	
54	Side	+	Side	
20047	NotionalValue	+	Amount	
20021	TradingPnl	+	Amount	
20022	ExchangeFee	+	Amount	
20023	BrokerFee	+	Amount	

BalanceReport (templateId = 9001)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	C	RequestId	
1	AccountId	+	AccountId	Account identifier
15	Currency	+	Currency	
20024	Reason	+	BalanceReportReason	
20025	UnrealizedPnl	+	Amount	
20026	OrdersMargin	+	Amount	
20027	PositionsMargin	+	Amount	
20028	Wallet	+	Amount	
20029	Borrowed	+	Amount	
20030	Available	+	Amount	

BalanceFetchReject (templateId = 9002)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	+	RequestId	
20031	Reason	+	BalanceFetchRejectReason	

PositionReport (templateId = 9003)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	C	RequestId	
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	Instrument identifier
20032	Reason	+	PositionReportReason	
20033	Size	+	Size	
20034	EntryPrice	+	Price	
20035	PartialLiquidationPrice	C	Price	
20036	FullLiquidationPrice	C	Price	
20037	EntryNotionalValue	+	Amount	
20038	CurrentNotionalValue	+	Amount	
20025	UnrealizedPnl	+	Amount	
20039	Margin	+	Amount	
20040	MaxRemovableMargin	+	Amount	

PositionFetchReject (templateId = 9004)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	+	RequestId	
20041	Reason	+	PositionFetchRejectReason	

RiskParamsReport (templateId = 9005)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	C	RequestId	
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	Instrument identifier
20042	Reason	+	RiskParamsReportReason	
20014	MarginMode	+	MarginMode	
20015	Leverage	C	Leverage	
20016	RiskLimit	+	Amount	

RiskParamsChangeReject (templateId = 9006)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	+	RequestId	
20043	Reason	+	RiskParamsChangeRejectReason	

RiskParamsFetchReject (templateId = 9007)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	+	RequestId	
20044	Reason	+	RiskParamsFetchRejectReason	

IsolatedMarginChangeReport (templateId = 9008)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	C	RequestId	
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	

IsolatedMarginChangeReject (templateId = 9009)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
11	RequestId	+	RequestId	
20045	Reason	+	IsolatedMarginChangeRejectReason	

FundingReport (templateId = 9010)

Tag	Field name	Required	Type	Description
20001	SeqNo	+	UInt64	
20002	TraceId	+	TraceId	Trace identifier
1	AccountId	+	AccountId	Account identifier
48	InstrumentId	+	InstrumentId	Instrument identifier
20046	Amount	+	Amount	
20049	Rate (since v2)	+	Ratio	When the value is positive, longs pay shorts
20003	TradingTimestamp (since v2)	+	Timestamp	

Enumerations

TerminationCode:

Value	Name	Description
1	Request	Session finished by user request
2	InternalError	Internal server error
3	ReRequestOutOfBounds	Range specified in RetransmitRequest is out of bounds
4	ReRequestInProgress	Retransmit request is in progress
5	TooFastClient	Client is sending too many messages
6	TooSlowClient	Client does not retrieve messages from TCP socket
7	MissedHeartbeat	Client did not send any messages for KeepaliveInterval time
8	InvalidMessage	Message is incorrect in terms of protocol
9	InvalidSequenceNumber	Response to a Sequence message with incorrect SequenceNumber
10	ServerShutdown	Server is shutting down by schedule

EstablishmentRejectCode:

Value	Name	Description
1	AlreadyEstablished	Connection for that client is already established
2	LoginBlocked	Client was blocked
3	InvalidKeepaliveInterval	Heartbeat interval is outside of allowed range

Value	Name	Description
4	AccessDenied	Credentials (LoginId + IP) are incorrect
5	InternalError	Internal server error

RetransmitRejectCode:

Value	Name	Description
1	OutOfRange	NextSeqNo + Count is beyond the range of sequence numbers
2	RequestLimitExceeded	The message Count exceeds a local rule for maximum retransmission size

MessageRejectReason:

Value	Name	Description
1	InvalidValue	Field value is incorrect
2	SystemUnavailable	Trading system is unavailable
3	DuplicateRequestId	Request identifier is not unique
4	ConflictingValue	Some field (RefTagId) value conflicts with another
5	UnsupportedOperation	Operation is not supported
6	ForbiddenOperation	Login is not allowed to perform this operation

OrderRejectReason:

Value	Name	Description
1	InternalError	Internal system error
2	UnknownInstrument	Unknown instrument
3	UnknownAccount	Unknown account
4	UnknownOrder	Order not found in system
5	NothingToChange	All changeable fields are empty
6	MinPriceIncrementViolation	
7	TooManyOrders	
8	TradingStatus	
9	PriceBandsViolation	Price exceeds current price band
10	InsufficientFunds	
11	RiskLimitReached	Order exceeds risk limits
12	RiskParamsChanging	Risk params are changing
13	AccountBlocked	Account is blocked
14	ConflictingProperties	Conflicting order type and time in force values

OrderType:

Value	Name	Description
1	Limit	Limit order
2	PostOnly	Passive order
3	Market	Market order

TimeInForce:

Value	Name	Description
1	GTC	Good Til Canceled
2	IOC	Immediate Or Cancel
3	FOK	Fill Or Kill

Side:

Value	Name	Description
1	Buy	
2	Sell	

MarginMode:

Value	Name	Description
1	Cross	
2	Isolated	

TradeType:

Value	Name	Description
1	Regular	
2	Liquidation	
3	Deleverage	

PositionFetchRejectReason:

Value	Name	Description
1	InternalError	Internal system error
3	NoOpenPosition	No open position
4	UnknownInstrument	Unknown instrument
5	UnknownAccount	Unknown Account

BalanceFetchRejectReason:

Value	Name	Description
1	InternalError	Internal system error
3	UnknownCurrency	Unknown currency
4	UnknownAccount	Unknown Account

RiskParamsChangeRejectReason:

Value	Name	Description
1	InternalError	Internal system error
3	UnknownInstrument	Unknown instrument
4	UnknownAccount	Unknown Account
5	NothingToChange	All changeable fields are empty
6	MaxRiskLimitExceeded	Requested risk limit is too high
7	UnavailableLeverage	Leverage unavailable for risk limit
8	NotionalValueExceeded	Notional value exceeds requested risk limit
9	InsufficientFunds	

RiskParamsFetchRejectReason:

Value	Name	Description
1	InternalError	Internal system error
3	UnknownInstrument	Unknown instrument

Value	Name	Description
4	UnknownAccount	Unknown Account

RiskParamsReportReason:

Value	Name	Description
1	Request	User requested report
2	AccountLinked	
3	RiskParamsChanged	

IsolatedMarginChangeRejectReason:

Value	Name	Description
1	InternalError	Internal system error
3	UnknownInstrument	Unknown instrument
4	UnknownAccount	Unknown Account
5	InsufficientFunds	
6	InsufficientMargin	
7	NoOpenPosition	No open position for instrument
8	NoIsolatedMargin	

BalanceReportReason:

Value	Name	Description
1	Request	User requested report
2	AccountLinked	
3	RiskParamsChanged	
4	IsolatedMarginChanged	
5	MarketChanged	
6	OrderActivity	
7	Trading	
8	Funding	
9	Liquidation	
10	Deleverage	
100	Other	
101	Deposit	
102	Withdrawal	
103	Correction	
104	NonTradingFee	
105	InterAccountTransfer	
106	LoanIssuance	
107	LoanRepayment	
108	LoanInterest	

PositionReportReason:

Value	Name	Description
1	Request	User requested report
2	RiskParamsChanged	
3	IsolatedMarginChanged	
4	MarketChanged	
5	WalletBalanceUpdated	
6	Trading	
7	Funding	
8	Liquidation	

Value	Name	Description
9	Deleverage	

CancelReason:

Value	Name	Description
1	BookEmptied	Book emptied
2	OrderType	Passive only
3	TimeInForce	FOK, IOC
4	CrossTrade	Cross-trade
5	CancelRequest	Canceled by client
6	Liquidation	Canceled by liquidation mechanism

TradingStatus:

Value	Name	Description
1	Halted	
2	Normal	
3	CancelOnly	

Protocol flow

Session establishment and termination

Client *must* send `Establish` message to establish a connection. The server *must* reply with `EstablishmentAck` if parameters are correct. The session is now considered established. In case parameters are incorrect, or connection is already established, the server *must* reply with `EstablishmentReject` message stating the reason for the reject in `EstablishmentRejectCode` field.

Client *must* send `Terminate` message to close the session.

Monitoring session state

Client and gate should periodically exchange heartbeat messages (called `Sequence`). Client sets its heartbeat sending interval in `Establish` message and gate responds with its own interval in `EstablishmentAck`.

Server *must* send message once in its interval, but the message is not always heartbeat. In case the client doesn't send any message during its `KeepaliveInterval`, it is diconnected from the gate with `Terminate` message (reason `MissedHeartbeat`).

Client *must not* send more than a certain number of messages per second, otherwise it is disconnected with `Terminate` message (reason `TooFastClient`).

Message numbering

Client *must* support a sequence counter of incoming messages. Client receives its initial state in `EstablishmentAck` message. Every application level message after that should increment this counter.

Message retransmit

Client can request message retransmit using `Retransmit` and must specify the number of the first message to retransmit in the `FromSeqNo` field. The server must respond with `Retransmission` message with `count` field set to the number of messages that are going to be retransmitted. Retransmitted messages are sent after that. No new messages are allowed from the client until retransmission is complete.

Session restore

In case a connection is closed, Client should establish a new session and check the sequence number in `EstablishmentAck`. In case the received sequence number is greater than its own sequence number, Client should use `RetransmitRequest` to receive lost messages.

Tracing

All application level messages contain `TraceId`.

`TraceId` from responses and reports can be used when contacting the technical support in order to identify a problem response.

At the moment the gate ignores `TraceId` in requests and generates its own. The client can get generated `TraceId` from a corresponding response.

Application level messages

Placing orders

Client sends `OrderPlaceRequest` to place a new order. Gate responds with either `OrderReport` or `OrderPlaceReject`. `OrderReport` *may* be followed by another message depending on the value of `TimeInForce`:

TimeInForce	Messages
GTC	ExecutionReport (if filled)
IOC	ExecutionReport (if filled), OrderCancelReport (if not fulfilled)
FOK	ExecutionReport (if filled), OrderCancelReport (if not fulfilled)

Canceling orders

Client may cancel a successfully placed order using `OrderCancelRequest`. If the order cannot be found (e.g. already canceled, executed, or non-existent), `OrderCancelReject` is returned.

Canceling multiple orders

Client may cancel multiple orders at once using `OrderMassCancelRequest`. Client may request one or more filters for the request. Filters are additive, for example: if only `InstrumentId` field is set, it means that all orders with matching `InstrumentId` will be canceled. If both `InstrumentId` and `Side` are set, it means that all orders matching both `Side` and `InstrumentId` are canceled, etc. This gate will respond with `OrderCancelReport` for each successfully canceled order and `OrderMassCancelReport` at the end.

Replacing orders

An existing order can be replaced with an order with different price and quantity using `OrderReplaceRequest` message. If the order is already canceled or executed, `OrderReplaceReject` is returned.

Requesting limits

Client may request its limits by using `BalanceFetchRequest`. Gate responds with `BalanceReport` message.