SIMBA SPECTRA protocol specification

version 2.2.2

Moscow 2022

Table of Contents

History of changes

Date	Version	Changes
02.11.2022	2.2.2	Changes applied: • In section 4.1.11. DiscreteAuction (msg id=13), type of the 'UnderlyingSymbol' field was changed from String25 to VarChar.
14.10.2022	2.2.0	Changes applied: • New section '2.3.6. Repeating group dimensions'. • New section '2.4.7. Repeating field groups'. • New message 'DiscreteAuction (msg id=13)' - Parameters of assigned opening auctions. • The possibility of fragmentation for messages 'BestPrices (msg id=3)' was added (see "1.4.6. Incremental message fragmentation"). • Now the 'Best Prices (msg id=3)' messages are send in a separate packages(s) at the beginning of the matching transaction. • Section '4.2.1.2. The BestPrices, OrderUpdate, OrderExecution messages in one packet' was deleted. • New flags in the 'MDFlags' field of the 'OrderUpdate (msg id=5)' message: • 0x1000000000000000 - Book-or-cancel order (Passive only). • 0x4000000000000000 - Sign of an order/trade during a discrete auction. • In the 'OrderExecution (msg id=6)' message, new value in the 'MDFlags' field: 0x4000000000000000 - Sign of an trade during a discrete auction. • In the 'SecurityDefinition (msg id=12)' and 'SecurityStatus (msg id=9)' messages, new values in the 'SecurityTradingStatus' field: • '119' - the opening auction for the instrument started, you can place and delete orders for this instrument. • '121' - the opening auction for this instrument completed. • Some changes were made in section '6. Message schema'.
06.04.2022	2.0.0	Changes applied: • The 'SecurityDefinition (msg id=8)' message ID was changed from 8 to 12. • Added new types of fields 'DoubleNULL' and 'NegativePrices'. • New fields were added to 'SecurityDefinition (msg id=12)' message: • 'ValuationMethod' - Specifies the type of valuation method applied: 'FUT' - futures-style mark-to-market; 'EQTY' - equity-style • 'RiskFreeRate' - Risk free interest rate • 'FixedSpotDiscount' - The sum of the discounted values of the declared cash flows • 'ProjectedSpotDiscount' - The sum of the discounted values of the projected cash flows • 'SettlCurrency' - Settlement currency • 'NegativePrices' - Negative prices eligibility • 'UnderlyingBoard' - Underlying board code • 'DerivativeContractMultiplier' - Coefficient indicating the volume of the underlying asset in the contract quote and strikes of option series • In 'SecurityDefinition (msg id=12)' message, change for 'CFICode' field: • value 'OCEFPS' and 'OPEFPS' were removed; • description for values 'OCAFPS' and 'OPAFPS' was changed; • new values were added: • OCESCS - European cash-settled equity option Call

Date	Version	Changes
		• OPESCS - European cash-settled equity option Put • In 'SecurityDefinition (msg id=12)' message, transmitted values for the 'UnderlyingSymbol' field was changed. • In 'OrderUpdate (msg id=5)' message new value for 'MDFlags' field: 0x200000000000000 - Passive synthetic order. • In section '4.1.6. SecurityDefinition (msg id=12)', the description of the field 'UnderlyingFutureID' has been changed. • Some changes were made to section '6. Message schema'. The version of the message schema ('version' attribute in the sbe:messageSchema element) was changed from 0 to 1.
25.03.2022	1.2.0	Changes applied: • New CFI Code for Futures on Moscow Exchange: 'JFTXCC' - Daily futures contract with automatic prolongation (CFD - Contract for difference, Cash Settled) was added.
26.10.2021	1.0.4	Changes applied: • In OrderExecution (msg id=6) message: • The 'MDEntryPx' field type was changed to Decimal5NULL. • The 'MDEntrySize' field type was changed to Int64NULL. • Section '4.1.4. OrderExecution (msg id=6)' was split into two subsections: • '4.1.4.1. Matching an order into a trade'; • '4.1.4.2. Technical trades'. • New section "4.2.9. Building the order-book of active orders". • The 'minValue' and 'maxValue' attributes was added to the message schema in the definitions of decimal types Decimal5NULL and Decimal2NULL for 'mantissa'.
12.10.2021	1.0.3	Changes applied: • The explanations were added to the section '4.2. Trading interaction scenarios'. • Section '4.2.1. Add an order, trade and the best sell price update' was split into two subsections: • '4.2.1.1. The BestPrices message separately in the first package'; • '4.2.1.2. The BestPrices, OrderUpdate, OrderExecution messages in one packet
27.09.2021	1.0.2	Changes applied: • In the 'OrderUpdate (msg id=5)' message, the value '1' (Change) of the 'MDUpdateAction' field is no longer used. • The description of the 'MDEntrySize' field has been changed in the 'Order Execution (msg id=6)' message.
26.08.2021	1.0.1	Changes applied: • The new section 1.4.3. 'Messages in streams' was added. • In the 'OrderUpdate (msg id=5)' and 'OrderExecution (msg id=6)' messages, the value 'J '(EmptyBook) of the 'MDEntryType' field is no longer used, but it was left in the specification and is used in the 'OrderBookSnapshot' message (msg id = 7) to indicate an empty order book for an instrument.
29.06.2021	1.0.0	Changes applied: • Gateway and protocol was renamed to "SIMBA SPECTRA". • The SIMBA SPECTRA gateway does not broadcast technical clearing trades and matching orders (IOCs) that did not execute to trades. • Packet format was changed. The 'Incremental' and 'Snapshot' package formats are different. • The 'EmptyBook (msg id=4)' message was added. • The 'OrderLogUpdate' message was deleted. The 'OrderUpdate' and 'OrderExecution' messages are used instead. • The 'Revision' field was removed from message schemas.

Date	Version	Changes
		• The 'nullValue' attribute was removed from the message schema in the definitions of integer types. • In the message schema, the value of the 'package' attribute was changed from 'mktdata' to 'simba_spectra'. • New value 'PossDupFlag' for MsgFlagsSet.
19.04.2021	0.6.0	Changes applied: • Changed order of fields in 'Market Data Packet Header'. • 'BestPrices' message: • Removed 'RptSeq' field • 'ExchangeTradingSessionID' field type changed to uInt32. • The 'SecurityID' field type was changed to Int32, and the field itself was placed at the end of the message. • In the 'OrderLogUpdate' message, the 'SecurityID' field type is changed to Int32NULL. • In the 'OrderBookSnapshot', 'SecurityStatus', and 'SecurityDefinitionUpdateReport' messages, the 'SecurityID' field type is changed to Int32. • 'SecurityDefinition' message: • The type of the 'SecurityID' and 'LegSecurityID' fields has been changed to Int32. • The type of the 'UnderlyingSecurityID' and 'UnderlyingFutureID' fields has been changed to Int32NULL.

1. Introduction

1.1. Document Scope

This document describes SIMBA SPECTRA market data feed for the MOEX Derivatives Market and specifies the presentation, session, and application levels of the protocol. This document does not address any administrative, network connectivity or informational security aspects.

1.2. Target audience

The target audience is primarily business-analytics, system architects, and software developers implementing SIMBA SPECTRA market data connectors for the Derivatives Market.

1.3. Terms and definitions

This document contains the following terms, definitions and acronyms:

Term	Definition
FOL	Full Order Log
SBE	Simple Binary Encoding
NBP	New Best Prices (best bid price & best ask price)
UDP	User Datagram Protocol

1.4. SIMBA SPECTRA Gateway general description

SIMBA SPECTRA Gateway is a low-latency market data feed for the Derivatives Market and disseminates the Full Order Log, but unlike FAST FOL, the SIMBA SPECTRA gateway does not broadcast technical trades and IOC orders that did not execute in trades. It will also provide the new resulting best prices in the very beginning of each transaction.

The gateway broadcasts market data as SBE encoded FIX messages over UDP multicast.

1.4.1. Data streams

Main streams

SIMBA SPECTRA Gateway broadcasts online order log updates as two identical streams: 'Incremental Feed A' and 'Incremental Feed B'. Each stream is broadcasted via its own multicast group to mitigate the UDP’s unreliable nature. The 'BestPrices (msg id=3)', 'EmptyBook (msg id=4)', 'OrderUpdate (msg id=5)', 'OrderExecution (msg id=6)', 'Heartbeat (msg id=1)', 'SequenceReset (msg id=2)' messages are broadcasted in 'Incremental' stream. Messages are packed in the 'Incremental' packets. (see sec. '2.3.1. Incremental packet format').

Recovery streams

SIMBA SPECTRA Gateway broadcasts active orders snapshot in a loop as two identical streams: 'Snapshot Feed A' and 'Snapshot Feed B'. Each stream is broadcasted via its own multicast group to mitigate the UDP’s unreliable nature. To reduce bandwidth consumption data transmission rate is limited on the server side. The 'OrderBookSnapshot (msg id=7)', 'Heartbeat (msg id=1)', 'SequenceReset (msg id=2)' are transmitted in 'Snapshot' stream. Messages are packed in the 'Snapshot' packets (see sec. '2.3.2. Snapshot packet format').

TCP Replay Service

The gateway provides a full recovery service, which provides historical data for the entire current trading session via a TCP connection. The client sends the 'Logon (msg id=1000)', 'Logout(msg id=1001)', 'MarketDataRequest(msg id=1002)' messages to the service. Messages are packed in the 'Snapshot' packets (see sec. '2.3.2. Snapshot packet format').

1.4.2. Instrument data

SIMBA SPECTRA Gateway provides a service for publishing instruments, in which trading session status and descriptions of trading instruments are sent in the 'SecurityDefinition' (msg id = 12) and 'TradingSessionStatus' (msg id = 11) FIX messages form, encoded in SBE format. One message contains a description of one financial instrument. SIMBA SPECTRA Gateway broadcasts current trading session status, security definitions and their current trading status in a loop as two identical streams: 'Instrument Replay Feed A' and 'Instrument Replay Feed B'. Each stream is broadcasted via its own multicast group to mitigate the UDP’s unreliable nature. To reduce bandwidth consumption data transmission rate is limited on the server side.

SIMBA SPECTRA Gateway broadcasts instrument and session status changes as two identical streams: 'Instrument Incremental Feed A' and 'Instrument Incremental Feed B'. Each stream is broadcasted via its own multicast group to mitigate the UDP’s unreliable nature. To reduce bandwidth consumption data transmission rate is limited on the server side.

For client convenience data for different instrument types are published on separate multicast groups. Currently there are two groups:

- FUT-INFO - futures, calendar spreads, collateral futures for equity options;
- OPT-INFO - options and volatility information.

In the 'FUT-INFO Instrument Incremental' and 'OPT-INFO Instrument Incremental' groups, all messages have the format described in section "2.3.1. Incremental packet format".

In the 'FUT-INFO Instrument Replay' and 'OPT-INFO Instrument Replay' groups, all messages have the format described in section "2.3.2. Snapshot packet format".

1.4.3. Messages in streams

This section describes which messages are transmitted in each data stream.

Stream name	Stream type	Message name
ORDERS-LOG	Incremental	Heartbeat (msg id=1) SequenceReset (msg id=2) BestPrices (msg id=3) EmptyBook (msg id=4) OrderUpdate (msg id=5) OrderExecution (msg id=6)
ORDERS-LOG	Snapshot	Heartbeat (msg id=1) SequenceReset (msg id=2) OrderBookSnapshot (msg id=7)
ORDERS-LOG	Historical Replay	**From client to gateway:** Logon (msg id=1000) Logout (msg id=1001) MarketDataRequest (msg id=1002) **From gateway to client:** Heartbeat (msg id=1) BestPrices (msg id=3) EmptyBook (msg id=4) OrderUpdate (msg id=5) OrderExecution (msg id=6)
FUT-INFO	Instrument Replay	Heartbeat (msg id=1) SequenceReset (msg id=2) SecurityDefinition (msg id=12) TradingSessionStatus (msg id=11) DiscreteAuction (msg id=13)
FUT-INFO	Instrument Incremental	Heartbeat (msg id=1) SequenceReset (msg id=2) SecurityStatus (msg id=9) TradingSessionStatus (msg id=11) DiscreteAuction (msg id=13)
OPT-INFO	Instrument Replay	Heartbeat (msg id=1) SequenceReset (msg id=2) SecurityDefinition (msg id=12) TradingSessionStatus (msg id=11)
OPT-INFO	Instrument Incremental	Heartbeat (msg id=1) SequenceReset (msg id=2)

Stream name	Stream type	Message name
		SecurityStatus (msg id=9) SecurityDefinitionUpdateReport (msg id=10) TradingSessionStatus (msg id=11)

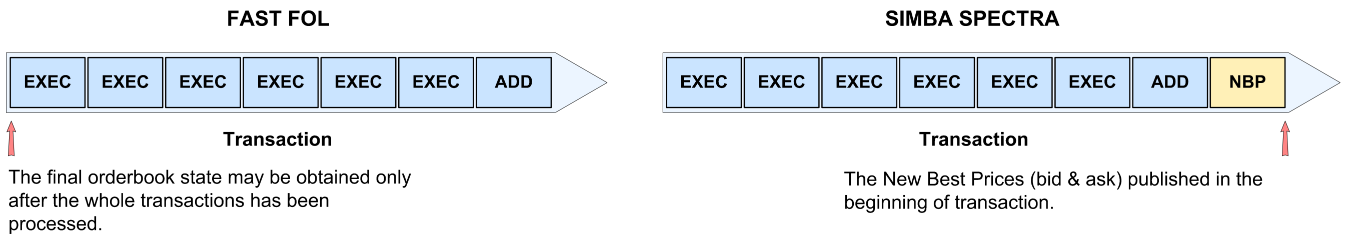
1.4.4. New Best Prices (NBP)

Every new order that reaches the matching engine produces a “transaction” – a list of order book modifications caused by that new order. A transaction is atomic and published only after all the order book modifications has been done.

FAST FOL Gateway (public market data feed) consecutively publishes order book events, which must be processed one by one to find the new final order book state or estimate the price shift.

When an aggressive order hits multiple passive orders, the resulting transaction may be quite long (hundreds of events). In such situations private execution reports may reveal valuable information well before it can be obtained through the public feed.

The new best prices bid&ask published in the very beginning of a transaction will give participants equal opportunities to estimate the magnitude of price movement.


FAST FOL
EXEC
EXEC
EXEC
EXEC
EXEC
EXEC
ADD
Transaction
The final orderbook state may be obtained only after the whole transactions has been processed.
SIMBA SPECTRA
EXEC
EXEC
EXEC
EXEC
EXEC
EXEC
ADD
NBP
Transaction
The New Best Prices (bid & ask) published in the beginning of transaction.

Pic. 1. New Best Prices

The new best buy and sell prices are published as the 'BestPrices' message (msg id = 3) on the 'Incremental Feed A' and 'Incremental Feed B'.

Important notes on NBP:

- NBP is published only for those transaction, where order entry or replacement resulted in a trade and best price was changed.
- NBP is omitted for negotiated orders.
- Implied synthetic liquidity is not shown.
- If multiple orderbooks have been involved in Synthetic Matching, the resulting transaction will have the BesttPrices (msg id = 3) message with NBP for each affected orderbook.
- If a transaction leads to an empty orderbook (or its either side), the corresponding flag is set in the 'BPFlags' field (Tag = 22000) in the 'Best Prices' incremental message (msg id = 3) (see sec. '4.1.1. BestPrices (msg id=3)').
- NBP publication is performed in a separate packages(s) before the package(s) of publication of the orders changes list.

1.4.5. Message numbering

Each channel (a set of a pair of 'Incremental Feed' order streams, a pair of 'Snapshot Feed' recovery streams and the TCP Replay service) has its own counter, from which the next number is taken when sending each packet. Each sent packet increments the counter value by '1'.

The counter is reset to '1' once a day during the technical break (see sec. '4.2.7. Data cleanup and message sequence reset').

1.4.6. Incremental message fragmentation

In order to prevent UDP packets from exceeding MTU size of 1500 bytes (typical for Ethernet networks), messages are fragmented into several parts. Fragmentation is carried out for packets formed from messages 'BestPrices', 'OrderUpdate', 'OrderExecution' and 'DiscreteAuction'.

If the transaction is more than one UDP packet, then the gateway sends it in several packet-fragments, marking such packets with the 'LastFragment' = 0 flag (the 'MsgFlags' field in the packet header see sec. '2.3.3. Market Data Packet Header'). If the entire transaction fits in one packet, then the gateway sends such a packet with the 'LastFragment' = 1 flag. The last packet-fragment of the transaction is marked with the same sign.

1.4.7. Snapshot message fragmentation

The first snapshot packet for the instrument with the 'OrderBookSnapshot' message (msg id = 7) is marked with the 'StartOfSnapshot' = 1 flag (the 'MsgFlags' field in the packet header see sec. '2.3.3. Market Data Packet Header'), and the last fragment packet with the

'OrderBookSnapshot' message (msg id = 7) is marked with the 'EndOfSnapshot' = 1 flag. If the entire snapshot fits into one packet, then the packet is marked with two flags at once: 'StartOfSnapshot' = 1 and 'EndOfSnapshot' = 1. Two flags are necessary so that in the event of packet loss outside of a snapshot, it is possible to collect a snapshot for specific instrument without waiting for a repeated round of snapshot broadcasting.

1.4.8. Recovery and late join

In case of packet loss or late connection to trading, SIMBA SPECTRA Gateway provides several mechanisms for data recovery:

- Recover missing data from recovery streams ('Snapshot Feed'). This way can be used to receive a large amount of lost data and to connect after the start of trading (see sec. '4.2.5. Late join and data recovery from the Snapshot Feed').
- Request in a separate TCP session of messages replays, which were previously sent to the 'Incremental' multicast group (TCP Replay service). This recovery method has some limitations (see sec. '4.2.6. Using TCP Replay service for data recovery') and can be used to receive a small amount of data only.

1.4.9. SIMBA SPECTRA protocol

The SIMBA SPECTRA protocol is based on FIX Simple Binary Encoding (https://www.fixtrading.org/standards/sbe-online); it is expected that users have already got some information about this protocol. SIMBA SPECTRA protocol consists of presentation, session and application layers.

2. Presentation layer

2.1. FIX syntax

Types and structure of messages, names and types of fields are used from the FIX standard: http://fiximate.fixtrading.org/.

2.2. SBE format

The Simple Binary Encoding standard version 1 is used to encode messages:https://www.fixtrading.org/standards/sbe-online/.

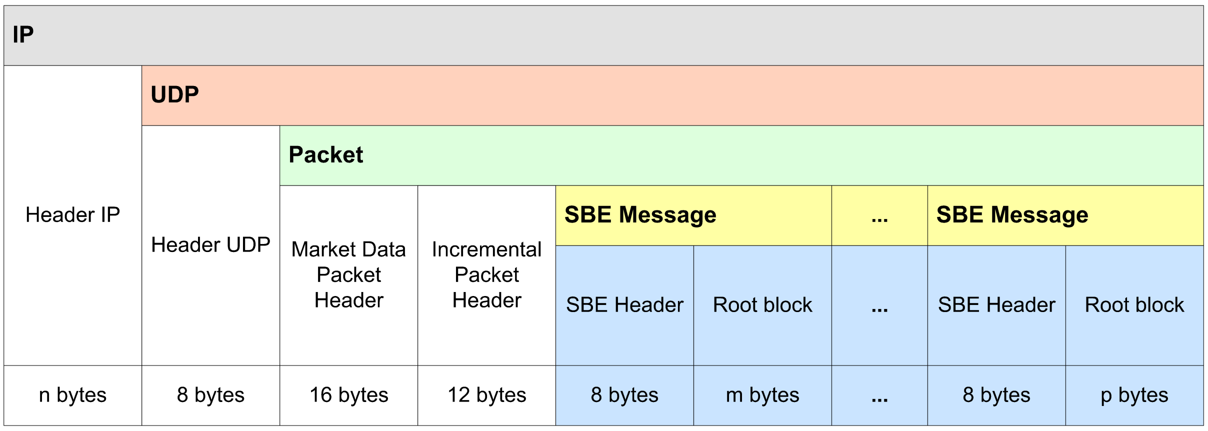
2.3. Packages structure

Data is transmitted in streams as packets.

2.3.1. Incremental packet format

Packet consists of the following parts:

- Market Data Packet Header.
- Incremental Packet Header.
- One or more SBE messages, each message consists of the following parts:
  - SBE Header.
  - FIX message in SBE format.

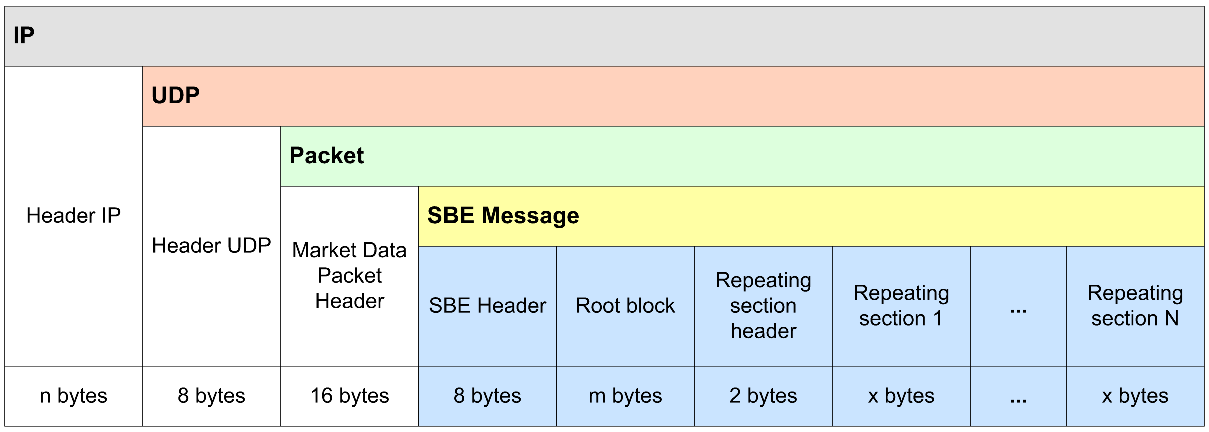


Pic. 2. Incremental packet format

2.3.2. Snapshot packet format

Packet consists of the following parts:

- Market Data Packet Header.
- SBE Header.
- FIX message in SBE format.



Pic. 3. Snapshot packet format

2.3.3. Market Data Packet Header

Market Data Packet Header contains packet sending time, packet serial number, packet size, and flags field. Byte order of encoding is little-endian, meaning that the least significant byte is serialized first on the wire.

Field	Type and size	Number of bytes	Mandatory	Details
MsgSeqNum	uInt32	4	Y	Package secuence number. A unique number is given to each packet sent. Each channel (a set of a pair of order streams, a pair of recovery streams and the TCP Replay service) has its own counter, from which the next number is taken when sending each packet. Each packet sent increases the counter value by 1. The counter is reset to 1 once a day.
MsgSize	uInt16	2	Y	The length of the entire message in bytes, including the length of this packet header.
MsgFlags	uInt16	2	Y	Message flags: • 0x1 - message fragmentation flag (LastFragment): 0 - it is not the last fragment of a fragmented message; 1 - last fragment of a fragmented message or the message is not fragmented. • 0x2 - flag of the first message in the snapshot for the instrument (StartOfSnapshot); • 0x4 - flag of the last message in the snapshot for the instrument (EndOfSnapshot); • 0x8 - flag of the 'IncrementalPacket': 0 - flag of the 'Snapshot' packet, 1 - flag of the 'Incremental' packet. • 0x10 - flag PossDupFlag: 0 - flag of broadcasting online updates, 1 - flag of broadcasting full order-books in the form of Incremental packages.
SendingTime	uInt64	8	Y	UTC time when the packet was sent by the gateway. In nanoseconds with Unix epoch, UTC timezone.

2.3.4. Incremental Packet Header

Incremental Packet Header contains the start time of transaction processing in matching and the trading session identifier. Byte order of encoding is little-endian, meaning that the least significant byte is serialized first on the wire.

Field	Type and size	Number of bytes	Mandatory	Details
TransactTime	uInt64	8	Y	UTC time of the beginning of transaction processing in matching. In nanoseconds in Unix epoch, UTC timezone.
ExchangeTradingSessionID	uInt32	4	N	The trading session identifier.

2.3.5. SBE Header

SBE Header contains size, message template ID, message schema ID, message schema version.

Field	Type and size	Number of bytes	Mandatory	Details
BlockLength	uInt16	2	Y	The root part length of the message in bytes. It does not include the SBE message header and the repeating 'NoMDEntries' field group.
TemplateID	uInt16	2	Y	Message template identifier.
SchemaID	uInt16	2	Y	Message schema identifier.
Version	uInt16	2	Y	Message schema version.

2.3.6. Repeating group dimensions

The header of a repeating group of fields. Repeating group dimensions contains the size of the field group and the number of field groups.

Field	Type and size	Number of bytes	Mandatory	Details
blockLength	uInt16	2	Y	Field group size.
numInGroup	uInt8	1	Y	Number of field groups.

2.4. Data types

Within the protocol, the following data types are used:

2.4.1. Integer

```
<type name="uInt8" primitiveType="uint8"/>

<type name="uInt8NULL" presence="optional" primitiveType="uint8"/>

<type name="uInt32" primitiveType="uint32"/>

<type name="uInt32NULL" presence="optional" primitiveType="uint32"/>

<type name="uInt64" primitiveType="uint64"/>

<type name="uInt64NULL" presence="optional" primitiveType="uint64"/>

<type name="Int32" primitiveType="int32"/>

<type name="Int32NULL" presence="optional" primitiveType="int32"/>

<type name="Int64" primitiveType="int64"/>

<type name="Int64NULL" presence="optional" primitiveType="int64"/>
```

2.4.2. Decimal

```
<composite name="Decimal5" description="Price type in Spectra" semanticType="Price">
  <type name="mantissa" description="mantissa" primitiveType="int64"/>
  <type name="exponent" description="exponent" presence="constant" primitiveType="int8">-5</type>
</composite>

<composite name="Decimal5NULL" description="Price type in Spectra" semanticType="Price">
   <type name="mantissa" description="mantissa" presence="optional" minValue="-9223372036854775808"
            maxValue="9223372036854775806" nullValue="9223372036854775807" primitiveType="int64"/>
   <type name="exponent" description="exponent" presence="constant" primitiveType="int8">-5</type>
</composite>

<composite name="Decimal2NULL" description="Price type in Spectra" semanticType="Price">
   <type name="mantissa" description="mantissa" presence="optional" minValue="-9223372036854775808"
            maxValue="9223372036854775806" nullValue="9223372036854775807" primitiveType="int64"/>
   <type name="exponent" description="exponent" presence="constant" primitiveType="int8">-2</type>
</composite>
```

2.4.3. String

```
<type name="Char" primitiveType="char"/>

<type name="String3" length="3" primitiveType="char"/>
```

```
<type name="String4" length="4" primitiveType="char"/>

<type name="String6" length="6" primitiveType="char"/>

<type name="String25" length="25" primitiveType="char"/>

<type name="String31" length="31" primitiveType="char"/>

<type name="String256" length="256" primitiveType="char"/>

<type name="DoubleNULL" presence="optional" primitiveType="double"/>

<type name="SecurityIDSource" presence="constant" length="1" primitiveType="char">8</type>

<type name="MarketID" presence="constant" length="4" primitiveType="char">MOEX</type>

<composite name="Utf8String" description="Variable-length UTF-8 string">
    <type name="length" primitiveType="uint16" semanticType="Length"/>
    <type name="varData" length="0" primitiveType="uint8" semanticType="data" characterEncoding="UTF-8"/>

</composite><composite name="VarString" description="Variable-length ASCII string">
    <type name="length" primitiveType="uint16" semanticType="Length"/>
    <type name="varData" length="0" primitiveType="uint8" semanticType="data" characterEncoding="US-ASCII"/
</composite>
```

2.4.4. Floating point

```
<type name="DoubleNULL" presence="optional" primitiveType="double"/>
```

2.4.5. Enumerations

```
<enum name="MDUpdateAction" encodingType="uInt8">
  <validValue name="New" description="New"      >0</validValue>
  <validValue name="Change" description="Change">1</validValue>
  <validValue name="Delete" description="Delete">2</validValue>
</enum>

<enum name="MDEntryType" encodingType="Char">
  <validValue name="Bid" description="Bid"             >0</validValue>
  <validValue name="Offer" description="Offer"         >1</validValue>
  <validValue name="EmptyBook" description="Empty Book">J</validValue>
</enum>

<enum name="SecurityAltIDSource" encodingType="Char">
  <validValue name="ISIN" description="ISIN"                     >4</validValue>
  <validValue name="ExchangeSymbol" description="Exchange symbol">8</validValue>
</enum>

<enum name="SecurityTradingStatus" encodingType="uInt8NULL">
  <validValue name="TradingHalt" description="Trading halt"   >2</validValue>
  <validValue name="ReadyToTrade" description="Ready to trade">17</validValue>
  <validValue name="NotAvailableForTrading"
              description="Not available for trading"         >18</validValue>
  <validValue name="NotTradedOnThisMarket"
              description="Not traded on this market"         >19</validValue>
  <validValue name="PreOpen" description="Pre-open"           >21</validValue>
  <validValue name="DiscreteAuctionOpen"
              description="Discrete auction started"          >119</validValue>
  <validValue name="DiscreteAuctionClose"
              description="Discrete auction ended"            >121</validValue>
</enum>

<enum name="TradingSessionID" encodingType="uInt8NULL">
  <validValue name="Day" description="Day session"        >1</validValue>
  <validValue name="Morning" description="Morning session">3</validValue>
  <validValue name="Evening" description="Evening session">5</validValue>
</enum>

<enum name="MarketSegmentID" encodingType="Char">
  <validValue name="Derivatives" description="Derivatives">D</validValue>
</enum>
```

```
<enum name="TradSesStatus" encodingType="uInt8">
  <validValue name="Halted" description="Session paused"    >1</validValue>
  <validValue name="Open" description="Session started"     >2</validValue>
  <validValue name="Closed" description="Session ended"     >3</validValue>
  <validValue name="PreOpen" description="Session initiated">4</validValue>
</enum>

<enum name="TradSesEvent" encodingType="uInt8NULL">
  <validValue name="TradingResumes" description="Trading resumed after intraday
            clearing session"                                      >0</validValue>
  <validValue name="ChangeOfTradingSession" description="Start and end of
            trading session"                                       >1</validValue>
  <validValue name="ChangeOfTradingStatus"  description="Trading session
            status change"                                         >3</validValue>
</enum>

<enum name="NegativePrices" encodingType="uInt8">
  <validValue name="NotEligible" description="Futures prices, price limits and options
            strikes are limited to be positive only"               >0</validValue>
  <validValue name="Eligible" description="Futures prices and options strikes are
            not limited"                                           >1</validValue>
</enum>
```

2.4.6. Bitmasks

```
<set name="BPFlagsSet" encodingType="uInt8">
  <choice name="BidEmptyBook" description="Empty bid book"    >0</choice>
  <choice name="OfferEmptyBook" description="Empty offer book">1</choice>
</set>

<set name="MDFlagsSet" encodingType="uInt64">
  <choice name="Day" description="Orders and Trades: Day order"                      >0</choice>
  <choice name="IOC" description="Orders and Trades: IOC order"                      >1</choice>
  <choice name="NonQuote" description="Orders and Trades: Non quote entry"           >2</choice>
  <choice name="EndOfTransaction" description="Orders and Trades: The end of matching
                                              transaction"                           >12</choice>
  <choice name="SecondLeg" description="Trades: Second leg of multileg trade"        >14</choice>
  <choice name="FOK" description="Orders: FOK order"                                 >19</choice>
  <choice name="Replace" description="Orders:The record results from replacing the order">20</choice>
  <choice name="Cancel" description="Orders:The record results from cancelling the order">21</choice>
  <choice name="MassCancel" description="Orders: The record results from mass cancelling">22</choice>
  <choice name="Negotiated" description="Trades: Negotiated trade"                   >26</choice>
  <choice name="MultiLeg" description="Trades: Multileg trade"                       >27</choice>
  <choice name="CrossTrade" description="Orders: Flag of cancelling the left balance of the order
                                        because of a cross-trade"                    >29</choice>
  <choice name="COD" description="Orders: The record results from cancelling an order via
                                 'Cancel on Disconnect' service"                     >32</choice>
  <choice name="ActiveSide" description="Trades: Flag of aggressive side"            >41</choice>
  <choice name="PassiveSide" description="Trades: Flag of passive side"              >42</choice>
  <choice name="Synthetic" description="Orders and Trades: Flag of the synthetic order"  >45</choice>
  <choice name="RFS" description="Orders and Trades: RFS is the source of entry"     >46</choice>
  <choice name="SyntheticPassive" description="Orders: Flag of the passive synthetic
                                              order"                                 >57</choice>
  <choice name="BOC" description="Orders: Book or Cancel order"                      >60</choice>
  <choice name="DuringDiscreteAuction" description="Orders and Trades: The record formed in the
                                                   process of discrete auction"      >62</choice>
</set>

<set name="FlagsSet" encodingType="uInt64">
  <choice name="EveningOrMorningSession" description="Trading in the
                                                    evening or morning session"      >0</choice>
  <choice name="AnonymousTrading" description="Anonymous trading"                    >4</choice>
  <choice name="PrivateTrading" description="Private trading"                        >5</choice>
  <choice name="DaySession" description="Trading in the day session"                >6</choice>
  <choice name="MultiLeg" description="MultiLeg instrument"                          >8</choice>
  <choice name="Collateral" description="Collateral instrument"                      >18</choice>
  <choice name="IntradayExercise" description="Exercise in the intraday clearing session">19</choice>
</set>

<set name="MsgFlagsSet" encodingType="uint16">
  <choice name="LastFragment" description="Message fragmentation flag"               >0</choice>
  <choice name="StartOfSnapshot" description="Flag of the first message in the snapshot
```

```
                                              for the instrument"                        >1</choice>
  <choice name="EndOfSnapshot" description="Flag of the last message in the snapshot
                                            for the instrument"                          >2</choice>
  <choice name="IncrementalPacket" description="Incremental packet flag"                 >3</choice>
  <choice name="PossDupFlag" description="Flag of the order book retransmission
                                          in the incremental stream"                     >4</choice>
</set>
```

2.4.7. Repeating field groups

```
<composite name="groupSize" description="Repeating group dimensions" semanticType="NumInGroup">
   <type name="blockLength" primitiveType="uint16"/>
   <type name="numInGroup" primitiveType="uint8"/>
</composite>
```

2.5. Message schema

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="sbe_schema.xsl" type="text/xsl"?>
<sbe:messageSchema package="moex_spectra_simba" byteOrder="littleEndian" id="19780" version="1"
 semanticVersion="FIX5SP2" description="20201005"
 xmlns:sbe="http://fixprotocol.io/2016/sbe"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://fixprotocol.io/2016/sbe sbe.xsd">
</sbe:messageSchema>
```

Schema attributes:

Attribute	Details	Value
id	Schema unique ID	
version	Schema version	
package	Schema name or category	"moex_spectra_simba"
byteOrder	Byte order in fields	"littleEndian"

The order of fields in tables in this specification may not correspond to the order of fields in message descriptions in the xml-schema of the SIMBA SPECTRA protocol. You have to be guided by the xml-schema of the SIMBA SPECTRA protocol when parsing SIMBA messages.

3. Session layer

3.1. Supported messages

- **Logon (msg id=1000)** - Initiates and confirms a session with the TCP Replay service to request missed packets.
- **Logout (msg id=1001)** - Initiates and confirms the end of the TCP Replay service session.
- **Heartbeat (msg id=1)** - SIMBA SPECTRA Gateway sends this message if there are no other messages on the stream for 30 seconds.
- **SequenceReset (msg id=2)** - Reset message numbers.

Below, there are details on the message fields. Each field contains the following attributes:

- **Tag** - field unique ID;
- **Field** - field name;
- **Mandatory** - defines whether 'nullValue' is a valid value or not:
  - **Y** - the field is mandatory, i.e. 'nullValue' will not be transmitted;
  - **N** - the field non-mandatory, i.e. 'nullValue' may be transmitted;
  - **C** - the field contains a non-'nullValue' value subject to a certain condition.
- **Type** - field type;
- **Details** - field's detailed description.

3.1.1. Logon (msg id=1000)

Initiates and confirms a session with the TCP Replay service to request missed packets.

A client message to the SIMBA SPECTRA Gateway initiating a session with TCP Replay service to request missed packets. A message from SIMBA SPECTRA Gateway to the client confirming the establishment of a session with the TCP Replay service.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		

3.1.2. Logout (msg id=1001)

A message from SIMBA SPECTRA Gateway to initiate the end of session with TCP Replay service.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
58	Text	Y	String256	Free format text string. May contain the reason for the session ending.

A client message confirming the end of the session with the TCP Replay service.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
58	Text	Y	String256	Free format text string. May contain the confirmation text of the session end.

3.1.3. Heartbeat (msg id=1)

SIMBA SPECTRA Gateway sends this message if there are no other messages on the stream for 30 seconds.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		

3.1.4. SequenceReset (msg id=2)

Reset message numbers.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
36	NewSeqNo	Y	uInt32	New sequence number.

4. Application layer

4.1. Supported messages

- **BestPrices (msg id=3)** - A message for publishing the best prices (best bid & best ask) at the beginning of each transaction.
- **EmptyBook (msg id=4)** - Clearing data in the gateway.
- **OrderUpdate (msg id=5)** - New order message or delete order message.
- **OrderExecution (msg id=6)** - Order execution message.
- **OrderBookSnapshot (msg id=7)** - Active orders snapshot.
- **SecurityDefinition (msg id=12)** - Instrument information.
- **SecurityStatus (msg id=9)** - Changing the status, price limits or collateral volume for the instrument.
- **SecurityDefinitionUpdateReport (msg id=10)** - Volatility and theoretical prices of options.
- **TradingSessionStatus (msg id=11)** - Changing the trading session state.
- **MarketDataRequest (msg id=1002)** - Request for missing packets via TCP Replay service.
- **DiscreteAuction (msg id=13)** - Parameters of assigned opening auctions.

4.1.1. BestPrices (msg id=3)

A message for publishing the best prices (best bid & best ask) at the beginning of each transaction.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
268	NoMDEntries	Y	groupSize	Number of entries in 'BestPrices' message.
=>645	MktBidPx	N	Decimal5NULL	Best buy price.
=>646	MktOfferPx	N	Decimal5NULL	Best sell price.
=>22000	BPFlags	Y	BPFlagsSet	Empty order-book flag. The field is a bit mask: • 0x1 - Empty order-book to buy • 0x2 - Empty order-book to sell
=>48	SecurityID	Y	Int32	Instrument numeric code.

4.1.2. EmptyBook (msg id=4)

'EmptyBook' message means clearing data in the gateway. When a client receives 'EmptyBook' message, in which the 'Incremental Packet Header' field has 'ExchangeTradingSessionID' is equal NULL, he must clear all orders on his side, ignore all 'OrderUpdate' updates (msg id = 5) with numbers greater than the number from the 'LastMsgSeqNumProcessed' field from the 'EmptyBook' message (msg id = 4), and wait for the arrival of all orders available in the trading system after cleaning. If the field 'LastMsgSeqNumProcessed field' = NULL, it means that all previously sent 'OrderUpdate' updates (msg id = 5) are valid.

When a client receives 'EmptyBook' messages, in which the 'Incremental Packet Header' field has 'ExchangeTradingSessionID' not equal NULL, he must clear all orders on his side for a given trading session and wait for the arrival of orders re-placed in a new trading session.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
369	LastMsgSeqNumProcessed	N	uInt32NULL	Sequence number of the last valid Incremental feed packet.

4.1.3. OrderUpdate (msg id=5)

New order message or delete order message.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
278	MDEntryID	Y	Int64	Order ID
270	MDEntryPx	Y	Decimal5	Order price.
271	MDEntrySize	Y	Int64	Order volume.

Tag	Field	Manda-tory	Type	Details
20017	MDFlags	Y	MDFlagsSet	Order type. The field is a bit mask: • 0x1 - Day • 0x2 - IOC • 0x4 - OTC • 0x1000 - End of transaction bit • 0x80000 - Fill-or-Kill • 0x100000 - The entry is the result of the order move • 0x200000 - The entry is the result of the order cancel • 0x400000 - The entry is the result of the orders mass cancel • 0x4000000 - Negotiated order • 0x8000000 - Multi-leg order • 0x20000000 - Sign of order deletion due to a cross-trade • 0x100000000 - The entry is the result of the orders cancel by "Cancel on Disconnect" service • 0x200000000000 - Synthetic order • 0x400000000000 - RFS order • 0x200000000000000 - Passive synthetic order • 0x1000000000000000 - Book-or-cancel order (Passive only) • 0x4000000000000000 - Sign of an order/trade during the opening auction
48	SecurityID	Y	Int32	Instrument numeric code.
83	RptSeq	Y	uInt32	Incremental refresh sequence number.
279	MDUpdateAction	Y	MDUpdateAction	Incremental refresh type: • '0' - New • '2' - Delete
269	MDEntryType	Y	MDEntryType	Record type: • '0' - Bid • '1' - Ask

4.1.4. OrderExecution (msg id=6)

4.1.4.1. Matching an order into a trade

Partial and full execution of orders (trade).

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
278	MDEntryID	Y	Int64	Order ID
270	MDEntryPx	Y	Decimal5NULL	Order price.
271	MDEntrySize	Y	Int64NULL	Remaining quantity in the order.
31	LastPx	Y	Decimal5	Trade price.
32	LastQty	Y	Int64	Trade volume.
1003	TradeID	Y	Int64	Trade ID.
20017	MDFlags	Y	MDFlagsSet	Trade type. The field is a bit mask:

Tag	Field	Manda-tory	Type	Details
				• 0x1 - Trade by quote order (Day) • 0x2 - Trade by matching order (IOC) • 0x4 - OTC-trades, including negotiated, multy-leg trades, RFS-trades • 0x1000 - End of transaction bit • 0x80000 - Trade by Fill-or-kill order • 0x4000000 - Negotiated trade • 0x8000000 - Multi-leg trade. For multi-leg operations • 0x20000000000 - The active side in the trade. The order that led to the trade when added to the order-book • 0x40000000000 - The passive side in the trade. Order from the order-book participating in the trade • 0x200000000000 - Trade by synthetic order • 0x400000000000 - RFS trade • 0x4000000000000000 - Sign of an trade during the opening auction
48	SecurityID	Y	Int32	Instrument numeric code.
83	RptSeq	Y	uInt32	Incremental refresh sequence number.
279	MDUpdateAction	Y	MDUpdateAction	Incremental refresh type: • '1' - Change • '2' - Delete
269	MDEntryType	Y	MDEntryType	Record type: • '0' - Bid • '1' - Ask

4.1.4.2. Technical trades

Trades on the legs of calendar spreads.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
278	MDEntryID	Y	Int64	Order ID
270	MDEntryPx	N	Decimal5NULL	Not present
271	MDEntrySize	N	Int64NULL	Not present
31	LastPx	Y	Decimal5	Trade price.
32	LastQty	Y	Int64	Trade volume.
1003	TradeID	Y	Int64	Trade ID.
20017	MDFlags	Y	MDFlagsSet	Trade type. The field is a bit mask: • 0x4 - OTC-trades, negotiated, multy-leg trades, RFS-trades • 0x1000 - End of transaction bit • 0x4000 - Sign of operation on the second leg of the multi-leg instrument • 0x8000000 - Multi-leg trade • 0x400000000000 - RFS trade
48	SecurityID	Y	Int32	Instrument numeric code.
83	RptSeq	Y	uInt32	Incremental refresh sequence number.

Tag	Field	Mandatory	Type	Details
279	MDUpdateAction	Y	MDUpdateAction	Incremental refresh type: • '0' - New
269	MDEntryType	Y	MDEntryType	Record type: • '0' - Bid • '1' - Ask

4.1.5. OrderBookSnapshot (msg id=7)

Order book.

If the snapshot of active orders for the instrument is empty before the start of trading, then it is not transmitted in the stream. If the snapshot of active orders for the instrument becomes empty during trading, then it is transmitted as a message with MDEntryType = J (EmptyBook).

Tag	Field	Mandatory	Type	Details
<messageHeader>		Y		
48	SecurityID	Y	Int32	Instrument numeric code.
369	LastMsgSeqNumProcessed	Y	uInt32	The 'MsgSeqNum' of the last message sent into incremental feed at the time of the current snapshot generation.
83	RptSeq	Y	uInt32	The 'RptSeq' number of the last incremental update included in the current market data snapshot for instrument.
5842	ExchangeTradingSessionID	Y	uInt32	Trading session ID.
268	NoMDEntries	Y	groupSize	Number of 'MDEntry' records in the current message.
=>278	MDEntryID	N	Int64NULL	Order ID.
=>60	TransactTime	Y	uInt64	The start time of the event processing. UNIX time in nanoseconds, according to UTC.
=>270	MDEntryPx	N	Decimal5NULL	Order price..
=>271	MDEntrySize	N	Int64NULL	Order volume.
=>1003	TradeID	N	Int64NULL	Trade ID.
=>20017	MDFlags	Y	MDFlagsSet	Order or trade type. The field is a bit mask: • 0x1 - Day
=>269	MDEntryType	Y	MDEntryType	Record type: • '0' - Bid • '1' - Ask • 'J' - Empty Book

4.1.6. SecurityDefinition (msg id=12)

Instrument information.

Tag	Field	Mandatory	Type	Details
<messageHeader>		Y		
911	TotNumReports	Y	uInt32	Total messages number in the current list.
55	Symbol	N	String25	Symbol code of the instrument.
48	SecurityID	Y	Int32	Instrument unique ID. ID uniqueness is guaranteed within the market segment specified by the field 'MarketSegmentId'.
22	SecurityIdSource	C	SecurityIDSource	Identifies class or source of tag 48-SecurityID value.
455	SecurityAltID*	N	String25	Instrument symbol code.
456	SecurityAltIDSource*	N	SecurityAltIDSource	Class for SecurityAltID (455): • '8' - Exchange Symbol

Tag	Field	Manda-tory	Type	Details
				• '4' - ISIN number
167	SecurityType	N	String4	Multileg type - 'MLEG' — calendar spread.
461	CFICode	N	String6	Financial instrument class according to ISO-10962. Valid values are shown in the table below.
202	StrikePrice	N	Decimal5NULL	Strike price, for options.
231	ContractMultiplier	N	Int32NULL	Units of underlying asset in instrument.
326	SecurityTradingStatus*	N	SecurityTradingStatus	Instrument trading status: • '21' - session initiated • '17' - session started • '2' - session paused • '18' - session ended • '19' - not traded on this market • '119' - the opening auction for the instrument started, you can place and delete orders for this instrument • '121' - the opening auction for this instrument completed
15	Currency	N	String3	Currency: • RUB - roubles • USD - US dollars • XXX - percent points
1301	MarketId*	Y	MarketID	Exchange MIC: • 'MOEX' - Moscow Exchange
1300	MarketSegmentId*	N	MarketSegmentID	Market segments. Valid values are shown in the table below.
336	TradingSessionId	N*	TradingSessionID	Trading session type: • '1' - main session • '3' - early session • '5' - evening session
5842	ExchangeTradingSessionID	N	Int32NULL	Trading session ID.
5678	Volatility*	N	Decimal5NULL	Option volatility.
1149	HighLimitPx*	N	Decimal5NULL	Upper price limit. Futures and calendar spreads only.
1148	LowLimitPx*	N	Decimal5NULL	Lower price limit. Futures and calendar spreads only.
969	MinPriceIncrement	N	Decimal5NULL	Minimum price step.
1146	MinPriceIncrementAmount	N	Decimal5NULL	Price step cost.
20002	InitialMarginOnBuy*	N	Decimal2NULL	• futures - buyer initial margin • options - underlying initial margin for buying futures-style option
20000	InitialMarginOnSell*	N	Decimal2NULL	• futures - seller initial margin • options - underlying initial margin one uncovered position
20001	InitialMarginSyntetic*	N	Decimal2NULL	Underlying initial margin for one covered position (RUB). Options only.
20006	TheorPrice*	N	Decimal5NULL	Option theoretical price.
20007	TheorPriceLimit*	N	Decimal5NULL	Option theoretical price (limits adjusted).
879	UnderlyingQty	N	Decimal5NULL	Security nominal value.
318	UnderlyingCurrency	N	String3	Code of currency of the security nominal value.

Tag	Field	Manda-tory	Type	Details
541	MaturityDate	N	uInt32NULL	Instrument settlement date (format: YYYYMMDD). Futures only. The 'MaturityDate' field contains only the date.
1079	MaturityTime	N	uInt32NULL	Instrument settlement time (format: HHmmSSsss). Futures only. The 'MaturityTime' field always contains beginning of the day.
20008	Flags*	N	FlagsSet	Instrument flags: • '0x1' - Sign of trading in the evening or morning session; • '0x10' - Sign of anonymous trading; • '0x20' - Sign of non-anonymous trading; • '0x40' - Sign of trading in the main session; • '0x100' - Sign of multileg-instrument; • '0x40000' - Collateral instrument; • '0x80000' - Exercise in the intraday clearing session;
20040	MinPriceIncrementAmountCurr	N	Decimal5NULL	Value of the minimum increment in foreign currency.
20041	SettlPriceOpen	N	Decimal5NULL	Settlement price at the start of the session.
1197	ValuationMethod	N	String4	Specifies the type of valuation method applied: • 'FUT' - futures-style mark-to-market • 'EQTY' - equity-style
1190	RiskFreeRate	N	DoubleNULL	Risk free interest rate.
20042	FixedSpotDiscount	N	DoubleNULL	The sum of the discounted values of the declared cash flows.
20043	ProjectedSpotDiscount	N	DoubleNULL	The sum of the discounted values of the projected cash flows.
120	SettlCurrency	N	String3	Settlement currency.
20044	NegativePrices	Y	NegativePrices	Negative prices eligibility: • '0' - Futures prices, price limits and options strikes are limited to be positive only • '1' - Futures prices and options strikes are not limited
1266	DerivativeContractMultiplier	N	Int32NULL	Coefficient indicating the volume of the underlying asset in the contract quote and strikes of option series. Broadcast for options only.
1141	NoMDFeedTypes	N*	groupSize	Number of feed types.
=>1022	MDFeedType	N	String25	Feed type.
=>264	MarketDepth	N	uInt32NULL	Order-book depth.
=>1021	MDBookType	N	uInt32NULL	Order-book type: • '1' - Top of Book • '2' - Price Depth
711	NoUnderlyings	N	groupSize	Number of underlyings
=>311	UnderlyingSymbol	N	String25	Valid field values: • for futures and calendar spreads - underlying asset code is broadcast; • for options on futures - futures code is broadcast; • for equity options - SECCODE code from ASTS gateway is broadcast for getting a market data of a share. The combination of SECCODE + SECBOARD should be considered as a separate instrument with separate quotes and tables of deals and orders.

Tag	Field	Manda-tory	Type	Details
=>20045	UnderlyingBoard	N	String4	Underlying board code: • for futures and calendar spreads - 'nullValue' is broadcast; • for options on futures - 'nullValue' is broadcast; • for equity options - SECBOARD trading mode ID from ASTS gateway is broadcast. The combination of SECCODE + SECBOARD should be considered as a separate instrument with separate quotes and tables of deals and orders.
=>309	UnderlyingSecurityID	N	Int32NULL	Futures instrument ID.
=>2620	UnderlyingFutureID	N	Int32NULL	Identifier of the option series 'option_series_id'. Broadcast for options only.
555	NoLegs	N	groupSize	Nymber of legs
=>600	LegSymbol	N	String25	Multileg instrument's individual security's Symbol.
=>602	LegSecurityID	N	Int32	Multileg instrument's individual security's SecurityID.
=>623	LegRatioQty	N	Int32	Quantity ratio. Value of field ‘LegRatioQty’ indicates both amount and direction of a multi leg instrument, i.e. if the field ‘LegRatioQty’ contains a value greater than 0, then the multi leg instrument has the same direction as the multi leg order, while a value less than 0 indicates a direction of this multi leg instrument opposite to that of the multi leg order. The absolute value of the field ‘LegRatioQty’ multiplied by multi leg instrument amount in the order allows to obtain the instrument amount value for field ‘LegSymbol’.
870	NoInstrAttrib	N	groupSize	=0
864	NoEvents	N	groupSize	• '2' - for futures • '4' - for options
=>865 =>866 =>1145	EventType EventDate EventTime	N	Int32 uInt32 uInt64	EventType=7. Instrument trading end date.
=>865 =>866 =>1145	EventType EventDate EventTime	N	Int32 uInt32 uInt64	EventType=5. Instrument trading start date.
=>865 =>866 =>1145	EventType EventDate EventTime	N	Int32 uInt32 uInt64	EventType=100. Instrument exercise start date.
=>865 =>866 =>1145	EventType EventDate EventTime	N	Int32 uInt32 uInt64	EventType=101. Instrument exercise end date.
107	SecurityDesc	N	Utf8String	Instrument name.
20005	QuotationList	N	VarString	Quotation list.

Symbol '*****' - sign differs from the standard FIX protocol.

Table 1. Moscow Exchange 'MarketSegmentID' values

MarketId	MarketSegmentId	CFICode	Securi-tyType	Details
MOEX	D	FXXXSX FFXCSX FCXCSX		Futures: • 'FXXXSX' - undefined type of futures contract (Standardized Unknown Future, Unknown delivery)

MarketId	MarketSegmentId	CFICode	SecurityType	Details
		FXXCSX FFXPSX FCXPSX FXXPSX JFTXCC		• 'FFXCSX' - Cash-settled Futures on the stock and money sections of the market (Standardized Financial Future, Cash delivery) • 'FCXCSX' - Cash-settled Futures on the commodity and NAMEX sections of the market (Standardized Commodity Future, Cash delivery) • 'FXXCSX' - Cash-settled Futures otherwise (Standardized Unknown Future, Cash delivery) • 'FFXPSX' - Deliverable Futures on the stock and money sections of the market (Standardized Financial Future, Physical delivery) • 'FCXPSX' - Deliverable Futures on the commodity and NAMEX sections of the market (Standardized Commodity Future, Physical delivery) • 'FXXPSX' - Deliverable Futures otherwise (Standardized Unknown Future, Physical delivery) • 'JFTXCC' - Daily Futures with automatic prolongation (Contract for difference, Cash Settled)
MOEX	D	FMXXSX	MLEG	Calendar spreads
MOEX	D	OCAFPS OPAFPS OCESCS OPESCS		Options: • 'OCAFPS' - American deliverable futures option Call • 'OPAFPS' - American deliverable futures option Put • 'OCESCS' - European cash-settled equity option Call • 'OPESCS' - European cash-settled equity option Put

Table 2. Decoding characters from the 'CFICode' field

Char 1 *Category*	Char 2 *Group*	Char 3 *Scheme*	Char 4 *Underlying Asset*	Char 5 *Delivery*	Char 6 *Stdized/Non-Std*
O=Options	**C**=Call **P**=Put	**A**=American **E**=European	**S**=Stock-Equities **F**=Futures	**P**=Physical **C**=Cash	**S**=Standardized terms (maturity date, strike price, contract size)

4.1.7. SecurityStatus (msg id=9)

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

Tag	Field	Mandatory	Type	Details
<messageHeader>		Y		
48	SecurityID	Y	Int32	Instrument numerical code.
22	SecurityIdSource	C	SecurityIDSource	Identifies class or source of tag 48-SecurityID value.
55	Symbol	N	String25	Symbol code of the instrument.
326	SecurityTradingStatus*	N	SecurityTradingStatus	Instrument trading status: • '21' - session initiated • '17' - session started • '2' - session paused • '18' - session ended • '19' - not traded on this market • '119' - the opening auction for the instrument started, you can place and delete orders for this instrument • '121' - the opening auction for this instrument completed
1148	LowLimitPx*	N	Decimal5NULL	Lower price limit. Futures and calendar spreads only.
1149	HighLimitPx*	N	Decimal5NULL	Upper price limit. Futures and calendar spreads only.
20002	InitialMarginOnBuy*	N	Decimal2NULL	• futures - buyer initial margin

Tag	Field	Manda-tory	Type	Details
				• options - underlying initial margin for buying futures-style option
20000	InitialMarginOnSell*	N	Decimal2NULL	• futures - seller initial margin • options - underlying initial margin one uncovered position
20001	InitialMarginSyntetic*	N	Decimal2NULL	Underlying initial margin for one covered position (RUB). Options only.

Symbol '*****' - sign differs from the standard FIX protocol.

4.1.8. SecurityDefinitionUpdateReport (msg id=10)

Volatility and theoretical prices of options.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
48	SecurityID	Y	Int32	Instrument numerical code.
22	SecurityIdSource	C	SecurityIDSource	Identifies class or source of tag 48-SecurityID value.
5678	Volatility*	N	Decimal5NULL	Option volatility.
20006	TheorPrice*	N	Decimal5NULL	Option theoretical price.
20007	TheorPriceLimit*	N	Decimal5NULL	Option theoretical price (limits adjusted).

Symbol '*****' - sign differs from the standard FIX protocol.

4.1.9. TradingSessionStatus (msg id=11)

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

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
342	TradSesOpenTime	Y	uInt64	Trading session open date and time.
344	TradSesCloseTime	Y	uInt64	Trading session close date and time.
5840	TradSesIntermClearingStartTime*	N	uInt64NULL	Intraday clearing session start date and time.
5841	TradSesIntermClearingEndTime*	N	uInt64NULL	Intraday clearing session end date and time.
336	TradingSessionId	Y	TradingSessionID	Trading session type: • '1' - Day session • '3' - Morning session • '5' - Evening session
5842	ExchangeTradingSessionID	N	uInt32NULL	Trading session ID.
340	TradSesStatus	Y	TradSesStatus	Trading session state • '4' - Session initiated • '2' - Session started • '1' - Session paused • '3' - Session ended
1301	MarketId*	N*	MarketID	Exchange MIC: • 'MOEX' - Moscow Exchange
1300	MarketSegmentId	N*	MarketSegmentID	Market segments: • 'D' - Derivatives
1368	TradSesEvent	N	TradSesEvent	Trading session events: • '0' - Trading resumed after intraday clearing session

Tag	Field	Manda-tory	Type	Details
				• '1' - Start and end of trading session • '3' - Trading session status change

Symbol '*' - sign differs from the standard FIX protocol.

4.1.10. MarketDataRequest (msg id=1002)

Request for missing packets via TCP Replay service.

Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
1182	ApplBegSeqNum	Y	uInt32	Sequence number of the first requested message
1183	ApplEndSeqNum	Y	uInt32	Sequence number of the last requested message

4.1.11. DiscreteAuction (msg id=13)

Parameters of assigned opening auctions.

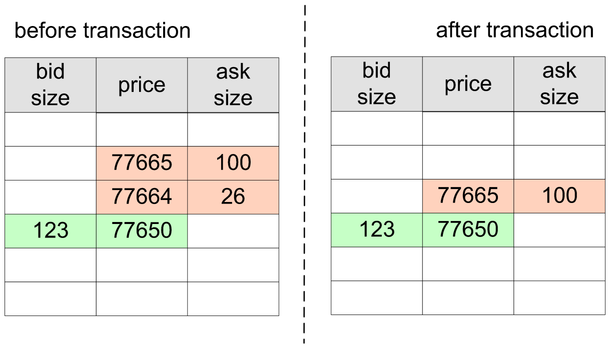
Tag	Field	Manda-tory	Type	Details
<messageHeader>		Y		
342	TradSesOpenTime	Y	uInt64	Date and time of the opening auction start.
20046	TradSesCloseTimeFrom	Y	uInt64	Date and time of the beginning of the time interval during which the opening auction will be completed.
20047	TradSesCloseTimeTill	Y	uInt64	Date and time of the end of the time interval during which the opening auction will be completed.
21002	AuctionID	Y	Int64	Opening auction ID.
5842	ExchangeTradingSessionID	Y	Int32	Trading session number.
20048	EventIDOpen	Y	Int32	ID of the synchro event about the start of the opening auction.
20049	EventIDClose	Y	Int32	ID of the synchro event about the end of the opening auction.
711	NoUnderlyings	N	groupSize	Number of blocks.
=>311	UnderlyingSymbol	N	VarChar	The underlying asset code of the instruments assigned to the opening auction.

4.2. Trading interaction scenarios

The scripts in this section show the possible variants for the messages layout by packages. The 'Best Prices' message is always published separately in the first package(s), and the 'Order Update', 'OrderExecution' messages in subsequent packages.

4.2.1. Add an order, trade and the best sell price update

An order is added to the system, a trade is made and the best sell price is updated.



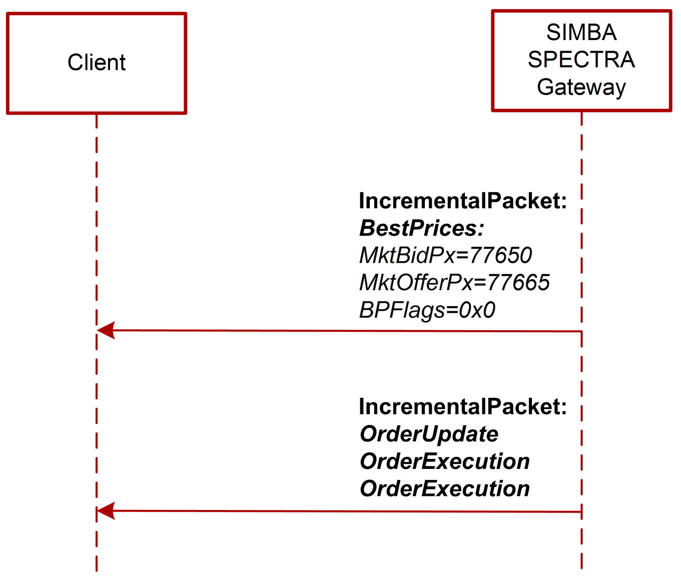
before transaction

bid size	price	ask size
	77665	100
	77664	26
123	77650	

after transaction

bid size	price	ask size
	77665	100
123	77650	

Pic. 4. Order-book state



Pic. 5. Diagram. Best selling price update.

With the first packet, SIMBA SPECTRA Gateway sends the 'BestPrices' message, where the best buy (unchanged) and sell (new) prices are in the 'MktBidPx' and 'MktOfferPx' fields.

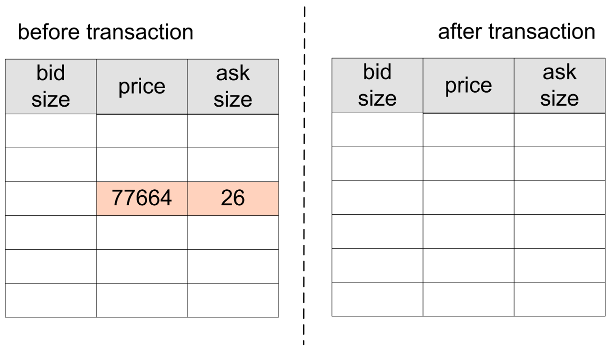
```
{ Packet header:
 MsgSeqNum=105805 MsgSize=N MsgFlags={ LastFragment:0 StartOfSnapshot:0 EndOfSnapshot:0
 IncrementalPacket:1 } SendingTime=20201014070029621
}
{ Incremental packet header:
 TransactTime[60]=70029621508252 ExchangeTradingSessionID[5842]=6144
}
{ SBE Header:
 BlockLength=M TemplateID=3 SchemaID=1 Version=1
}
{ SBE Message:
  Sequence: NoMDEntries[268] = 1 {
   [0]:  SecurityID[48]=1439162 MktBidPx[645]=77650 MktOfferPx[646]=77665 BPFlags[22000]=0x0
}
```

With the second packet, SIMBA SPECTRA Gateway sends an 'OrderUpdate' message and two 'OrderExecution' messages.

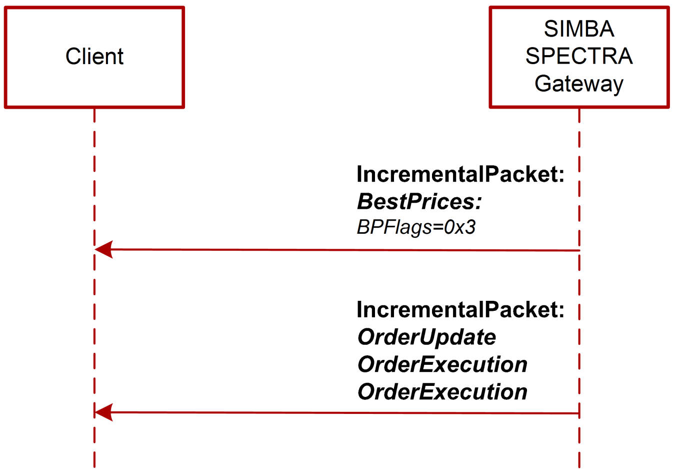
```
{ Packet header:
 MsgSeqNum=105806 MsgSize=N MsgFlags={ LastFragment:1 StartOfSnapshot:0 EndOfSnapshot:0
 IncrementalPacket:1 } SendingTime=20201014070029621
}
{ Incremental packet header:
 TransactTime[60]=70029621508252 ExchangeTradingSessionID[5842]=6144
}
{ SBE Header: BlockLength=M TemplateID=5 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=0 MDEntryID[278]=1892945606659163299 SecurityID[48]=1439162
 RptSeq[83]=60142 MDEntryPx[270]=77667 MDEntrySize[271]=26 MDFlags[20017]=0x2
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=0 MDEntryID[278]=1892945606659163299 SecurityID[48]=1439162
 RptSeq[83]=60144 MDEntryPx[270]=77667 LastPx[31]=77664 LastQty[32]=26
 TradeID[1003]=1892945606658296055 MDFlags[20017]=0x20000000002
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=1 MDEntryID[278]=1892945606659163300 SecurityID[48]=1439162
 RptSeq[83]=60145 MDEntryPx[270]=77664 LastPx[31]=77664 LastQty[32]=26
 TradeID[1003]=1892945606658296055 MDFlags[20017]=0x40000000001
}
```

4.2.2. Add an order, trade and order-book becomes empty

An order is added to the system, a trade is made and the order-book becomes empty.



Pic. 6. Order-book state



Pic. 7. Diagram. Updating - empty order-book

With the first packet, SIMBA SPECTRA Gateway sends the 'BestPrices' message. There are no orders left in the order-book, so the 'MktBidPx' and 'MktOfferPx' fields are missing in the message. Here and below, we mean by the field missing that it contains the value 'nullValue'. Value field 'BPFlags' = 0x3 in the message indicates that the order-book is empty.

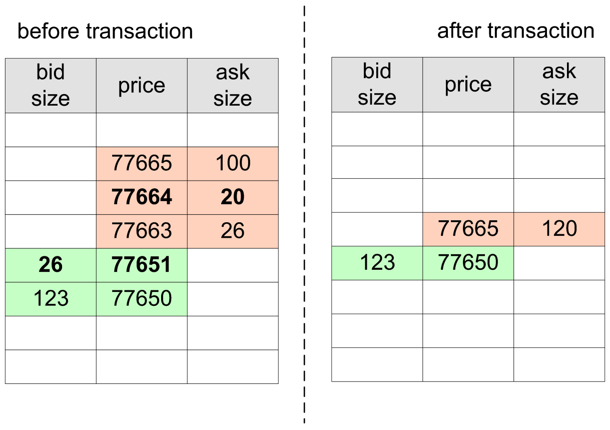
```
{ Packet header:
 MsgSeqNum=105805 MsgSize=N MsgFlags={ LastFragment:0 StartOfSnapshot:0 EndOfSnapshot:0
 IncrementalPacket:1 } SendingTime=20201014070029621
}
{ Incremental packet header:
 TransactTime[60]=70029621508252 ExchangeTradingSessionID[5842]=6144
}
{ SBE Header:
 BlockLength=M TemplateID=3 SchemaID=1 Version=1
}
{ SBE Message:
  Sequence: NoMDEntries[268] = 1 {
   [0]:  SecurityID[48]=1439162 BPFlags[22000]=0x3
}
```

With the second packet, SIMBA SPECTRA Gateway sends an 'OrderUpdate' message and two 'OrderExecution' messages.

```
{ Packet header:
 MsgSeqNum=105806 MsgSize=N MsgFlags={ LastFragment:1 StartOfSnapshot:0 EndOfSnapshot:0
 IncrementalPacket:1 } SendingTime=20201014070029621
}
{ Incremental packet header:
 TransactTime[60]=70029621508252 ExchangeTradingSessionID[5842]=6144
}
{ SBE Header:
 BlockLength=M TemplateID=4 SchemaID=1 Version=1
}
{ SBE Header: BlockLength=M TemplateID=5 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=0 MDEntryID[278]=1892945606659163299 SecurityID[48]=1439162
 RptSeq[83]=60142 MDEntryPx[270]=77667 MDEntrySize[271]=26 MDFlags[20017]=0x2
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=0 MDEntryID[278]=1892945606659163299 SecurityID[48]=1439162
 RptSeq[83]=60144 MDEntryPx[270]=77667 LastPx[31]=77664 LastQty[32]=26
 TradeID[1003]=1892945606658296055 MDFlags[20017]=0x20000000002 Revision[20018]=188696152606
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=1 MDEntryID[278]=1892945606659163300 SecurityID[48]=1439162
 RptSeq[83]=60145 MDEntryPx[270]=77664 LastPx[31]=77664 LastQty[32]=26
 TradeID[1003]=1892945606658296055 MDFlags[20017]=0x40000000001 Revision[20018]=188696152607
}
```

4.2.3. Move pair of orders, trade and the best prices update

For example, there is one order on each price levels - Ask: 77664 and Bid: 77651, and those orders are pair moved. In a matching transaction, the orders at the Ask: 77664 and Bid: 77651 price levels are pair moved, a trade is made on the moved Bid: 77651 order, and the best buy and sell prices are updated.



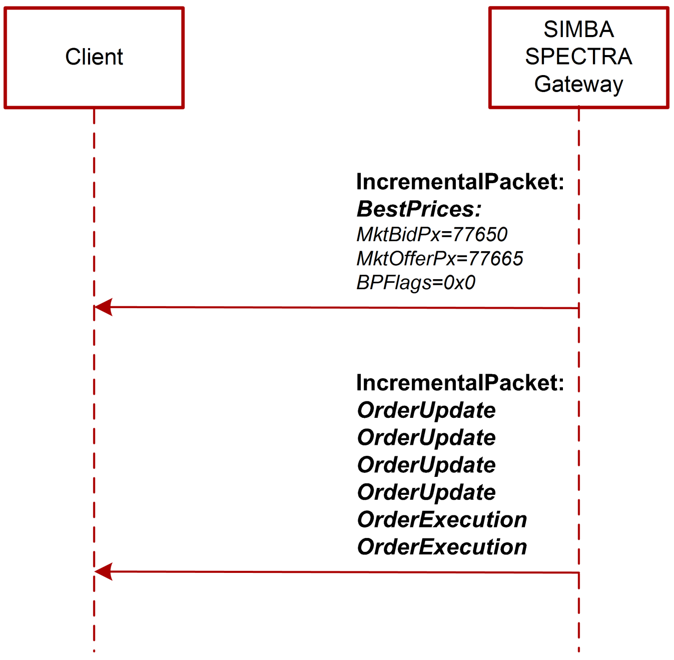
before transaction

bid size	price	ask size
	77665	100
	77664	**20**
	77663	26
26	**77651**	
123	77650	

after transaction

bid size	price	ask size
	77665	120
123	77650	

Pic. 8. Order-book state



Pic. 9. Diagram. Update - pair move

With the first packet, SIMBA SPECTRA Gateway sends the 'BestPrices' message with updated best prices.

```
{ Packet header:
 MsgSeqNum=105805 MsgSize=N MsgFlags={ LastFragment:0 StartOfSnapshot:0 EndOfSnapshot:0
 IncrementalPacket:1 } SendingTime=20201014070029621
}
{ Incremental packet header:
 TransactTime[60]=70029621508252 ExchangeTradingSessionID[5842]=6144
}
{ SBE Header:
 BlockLength=M TemplateID=3 SchemaID=1 Version=1
}
{ SBE Message:
  Sequence: NoMDEntries[268] = 1 {
   [0]:  SecurityID[48]=1439162 MktBidPx[645]=77650 MktOfferPx[646]=77665 BPFlags[22000]=0x0
}
```

With the second packet, SIMBA SPECTRA Gateway sends:

- four 'OrderUpdate' messages - a couple of messages (deleting / adding an order) for each moved order;
- two 'OrderExecution' messages - a trade.

```
{ Packet header:
 MsgSeqNum=105806 MsgSize=N MsgFlags={ LastFragment:1 StartOfSnapshot:0 EndOfSnapshot:0
 IncrementalPacket:1 } SendingTime=20201014070029621
}
{ Incremental packet header:
 TransactTime[60]=70029621508252 ExchangeTradingSessionID[5842]=6144
}
{ SBE Header:
 BlockLength=M TemplateID=4 SchemaID=1 Version=1
}
{ SBE Header: BlockLength=M TemplateID=5 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=1 MDEntryID[278]=1892945606659163287 SecurityID[48]=1439162
 RptSeq[83]=60140 MDEntryPx[270]=77664 MDEntrySize[271]=20 MDFlags[20017]=0x100001
}
{ SBE Header: BlockLength=M TemplateID=5 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=1 MDEntryID[278]=1892945606659163301 SecurityID[48]=1439162
 RptSeq[83]=60141 MDEntryPx[270]=77665 MDEntrySize[271]=20 MDFlags[20017]=0x100001
}
{ SBE Header: BlockLength=M TemplateID=5 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=0 MDEntryID[278]=1892945606659163299 SecurityID[48]=1439162
 RptSeq[83]=60142 MDEntryPx[270]=77651 MDEntrySize[271]=26 MDFlags[20017]=0x100001
}
{ SBE Header: BlockLength=M TemplateID=5 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=0 MDEntryID[278]=1892945606659163302 SecurityID[48]=1439162
 RptSeq[83]=60143 MDEntryPx[270]=77663 MDEntrySize[271]=26 MDFlags[20017]=0x100001
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=0 MDEntryID[278]=1892945606659163302 SecurityID[48]=1439162
 RptSeq[83]=60144 MDEntryPx[270]=77663 LastPx[31]=77663 LastQty[32]=26
 TradeID[1003]=1892945606658296055 MDFlags[20017]=0x20000000001
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=1 MDEntryID[278]=1892945606659163300 SecurityID[48]=1439162
 RptSeq[83]=60145 MDEntryPx[270]=77663 LastPx[31]=77663 LastQty[32]=26
 TradeID[1003]=1892945606658296055 MDFlags[20017]=0x40000000001
}
```

4.2.4. Add an order, trade and price changes for three different instruments as a result of synthetic matching

There are order-books of three different instruments participating in the synthetic match, RTS-03.21-06.21 (calendar spread), RTS-03.21 (near futures), RTS-06.21 (far futures). The order-book status is shown in the figure below. The 'bid size' and 'ask size' fields contain total liquidity (natural + synthetic), the 'bid size synth' and 'ask size synth' fields contain synthetic liquidity.

before transaction

RTS-03.21 (SecurityID=1)				
bid size synth	bid size	price	ask size	ask size synth
0	10	87500		

RTS-06.21 (SecurityID=2)				
bid size synth	bid size	price	ask size	ask size synth
		88550	20	0

RTS-03.21-0.6.21 (SecurityID=3)				
bid size synth	bid size	price	ask size	ask size synth
		1050	15	10

Pic. 10. Order-book state before the transaction

In SIMBA SPECTRA Gateway, the best prices are shown excluding synthetic liquidity, and for our case they will look like this:

	MktBidPx	**MktOfferPx**
RTS-03.21 (SecurityID = 1)	87500	-
RTS-06.21 (SecurityID = 2)	-	88550
RTS-03.21-06.21 (SecurityID = 3)	-	1050

In a matching transaction, the buy order at price 1050 on 20 contracts for RTS-03.21-06.21 is added to order-book for this instrument, the trade is made and order-books for all three instruments: RTS-03.21, RTS-06.21, RTS-03.21-06.21, change.

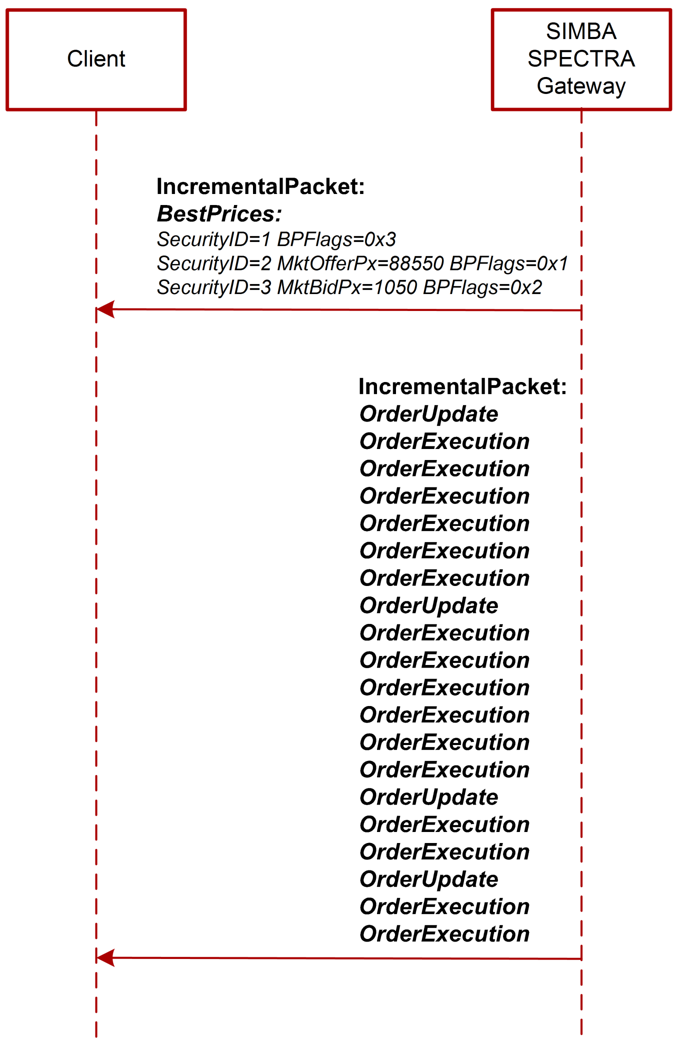
after transaction

RTS-03.21 (SecurityID=1)				
bid size synth	bid size	price	ask size	ask size synth
		87500	5	5

RTS-06.21 (SecurityID=2)				
bid size synth	bid size	price	ask size	ask size synth
		88550	10	0

RTS-03.21-0.6.21 (SecurityID=3)				
bid size synth	bid size	price	ask size	ask size synth
0	5	1050		

Pic. 11. Order-book state after the transaction



Pic. 12. Diagram. Update for three instruments

With the first packet, SIMBA SPECTRA Gateway sends the 'BestPrices' message with updated best prices for three instruments.

```
{ Packet header:
 MsgSeqNum=105805 MsgSize=N MsgFlags={ LastFragment:0 StartOfSnapshot:0 EndOfSnapshot:0
 IncrementalPacket:1 }  SendingTime=20201014070029621
}
{ Incremental packet header:
 TransactTime[60]=152831685239757 ExchangeTradingSessionID[5842]=50091
}
{ SBE Header:
 BlockLength=M TemplateID=3 SchemaID=1 Version=1
}
{ SBE Message:
  Sequence: NoMDEntries[268] = 3 {
   [0]:  SecurityID[48]=1 BPFlags[22000]=0x3
   [2]:  SecurityID[48]=2 MktOfferPx[646]=88550 BPFlags[22000]=0x1
   [3]:  SecurityID[48]=3 MktBidPx[645]=1050 BPFlags[22000]=0x2
}
```

With the second packet, SIMBA SPECTRA Gateway sends the 'OrderUpdate' and 'OrderExecution' messages.

```
{ Packet header:
 MsgSeqNum=105806 MsgSize=N MsgFlags={ LastFragment:1 StartOfSnapshot:0 EndOfSnapshot:0
 IncrementalPacket:1 } SendingTime=20201014070029621
}
{ Incremental packet header:
 TransactTime[60]=152831685239757 ExchangeTradingSessionID[5842]=50091
}
{ SBE Header:
 BlockLength=M TemplateID=4 SchemaID=1 Version=1
}
{ SBE Header: BlockLength=M TemplateID=5 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=0 MDEntryID[278]=1923533655070736409 SecurityID[48]=3
 RptSeq[83]=16 MDEntryPx[270]=1050 MDEntrySize[271]=20 MDFlags[20017]=134217729
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=1 MDEntryID[278]=1923533655070736409 SecurityID[48]=1
 RptSeq[83]=15 LastPx[31]=87500 LastQty[32]=5
 TradeID[1003]=1923533655070736390 MDFlags[20017]=2199157473285
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=0 MDEntryID[278]=1923533655070736407 SecurityID[48]=1
 RptSeq[83]=16 LastPx[31]=87500 LastQty[32]=5
 TradeID[1003]=1923533655070736390 MDFlags[20017]=4398180728837
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=0 MDEntryID[278]=1923533655070736409 SecurityID[48]=2
 RptSeq[83]=20 LastPx[31]=88550 LastQty[32]=5
 TradeID[1003]=1923533655070736391 MDFlags[20017]=2199157489669
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=1 MDEntryID[278]=1923533655070736407 SecurityID[48]=2
 RptSeq[83]=21 LastPx[31]=88550 LastQty[32]=5
 TradeID[1003]=1923533655070736391 MDFlags[20017]=4398180745221
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=1 MDEntryType[269]=0 MDEntryID[278]=1923533655070736409 SecurityID[48]=3
 RptSeq[83]=17 MDEntryPx[270]=1050 MDEntrySize[271]=15 LastPx[31]=1050 LastQty[32]=5
 TradeID[1003]=1923533655070736392 MDFlags[20017]=2199157473281
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=1 MDEntryID[278]=1923533655070736407 SecurityID[48]=3
 RptSeq[83]=18 MDEntryPx[270]=1050 LastPx[31]=1050 LastQty[32]=5
 TradeID[1003]=1923533655070736392 MDFlags[20017]=4398180728833
}
{ SBE Header: BlockLength=M TemplateID=5 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=1 MDEntryID[278]=1923533655070736410 SecurityID[48]=3
 RptSeq[83]=19 MDEntryPx[270]=1050 MDEntrySize[271]=10 MDFlags[20017]=35184506306561
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=1 MDEntryID[278]=1923533655070736409 SecurityID[48]=1
 RptSeq[83]=17 LastPx[31]=87500 LastQty[32]=10
 TradeID[1003]=1923533655070736393 MDFlags[20017]=2199157473285
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=0 MDEntryID[278]=1923533655070736410 SecurityID[48]=1
 RptSeq[83]=18 LastPx[31]=87500 LastQty[32]=10
 TradeID[1003]=1923533655070736393 MDFlags[20017]=39582552817669
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=0 MDEntryID[278]=1923533655070736409 SecurityID[48]=2
 RptSeq[83]=22 LastPx[31]=88550 LastQty[32]=10
 TradeID[1003]=1923533655070736394 MDFlags[20017]=2199157489669
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=1 MDEntryID[278]=1923533655070736410 SecurityID[48]=2
 RptSeq[83]=23 LastPx[31]=88550 LastQty[32]=10
 TradeID[1003]=1923533655070736394 MDFlags[20017]=39582552834053
}
```

```
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=1 MDEntryType[269]=0 MDEntryID[278]=1923533655070736409 SecurityID[48]=3
 RptSeq[83]=20 MDEntryPx[270]=1050 MDEntrySize[271]=5 LastPx[31]=1050 LastQty[32]=10
 TradeID[1003]=1923533655070736395 MDFlags[20017]=2199157473281
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=1 MDEntryID[278]=1923533655070736410 SecurityID[48]=3
 RptSeq[83]=21 MDEntryPx[270]=1050 LastPx[31]=1050 LastQty[32]=10
 TradeID[1003]=1923533655070736395 MDFlags[20017]=39582552817665
}
{ SBE Header: BlockLength=M TemplateID=5 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=0 MDEntryID[278]=1923533655070736411 SecurityID[48]=2
 RptSeq[83]=24 MDEntryPx[270]=88550 MDEntrySize[271]=10 MDFlags[20017]=35184372088833
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=0 MDEntryID[278]=1923533655070736411 SecurityID[48]=2
 RptSeq[83]=25 MDEntryPx[270]=88550 LastPx[31]=88550 LastQty[32]=10
 TradeID[1003]=1923533655070736396 MDFlags[20017]=37383395344385
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=1 MDEntryType[269]=1 MDEntryID[278]=1923533655070736408 SecurityID[48]=2
 RptSeq[83]=26 MDEntryPx[270]=88550 MDEntrySize[271]=10 LastPx[31]=88550 LastQty[32]=10
 TradeID[1003]=1923533655070736396 MDFlags[20017]=4398046511105
}
{ SBE Header: BlockLength=M TemplateID=5 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=0 MDEntryType[269]=1 MDEntryID[278]=1923533655070736412 SecurityID[48]=1
 RptSeq[83]=19 MDEntryPx[270]=87500 MDEntrySize[271]=10 MDFlags[20017]=35184372088833
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=1 MDEntryID[278]=1923533655070736412 SecurityID[48]=1
 RptSeq[83]=20 MDEntryPx[270]=87500 LastPx[31]=87500 LastQty[32]=10
 TradeID[1003]=1923533655070736397 MDFlags[20017]=37383395344385
}
{ SBE Header: BlockLength=N TemplateID=6 SchemaID=1 Version=1 }
{ SBE Message:
 MDUpdateAction[279]=2 MDEntryType[269]=0 MDEntryID[278]=1923533655070736406 SecurityID[48]=1
 RptSeq[83]=21 MDEntryPx[270]=87500 LastPx[31]=87500 LastQty[32]=10
 TradeID[1003]=1923533655070736397 MDFlags[20017]=4398046515201
}
```

4.2.5. Late join and data recovery from the Snapshot Feed

Data recovery from the Snapshot Feed can be used for retrieving a large amount of missed data, particularly in case of late join or disaster recovery.

The Snapshot Feed repeatedly disseminates the active orders snapshots. Message sequence number starts from 1 in each cycle therefore the cycle completes once one of the following messages again has the sequence number set to 1. In each 'OrderBookSnapshot' (msg id=7) message the value of the 'LastMsgSeqNumProcessed' (Tag=369) field corresponds to the value of the 'MsgSeqNum' field of the latest message published on the 'Incremental Feed' at the moment of taking the snapshot, and the 'RptSeq'(Tag=83) field contains the 'RptSeq' number of the latest incremental update on a per instrument basis. The 'MsgSeqNum' gaps allow general data loss detection, and the 'RptSeq' gaps shows the exact instruments affected by data loss.

Recovery Procedure:

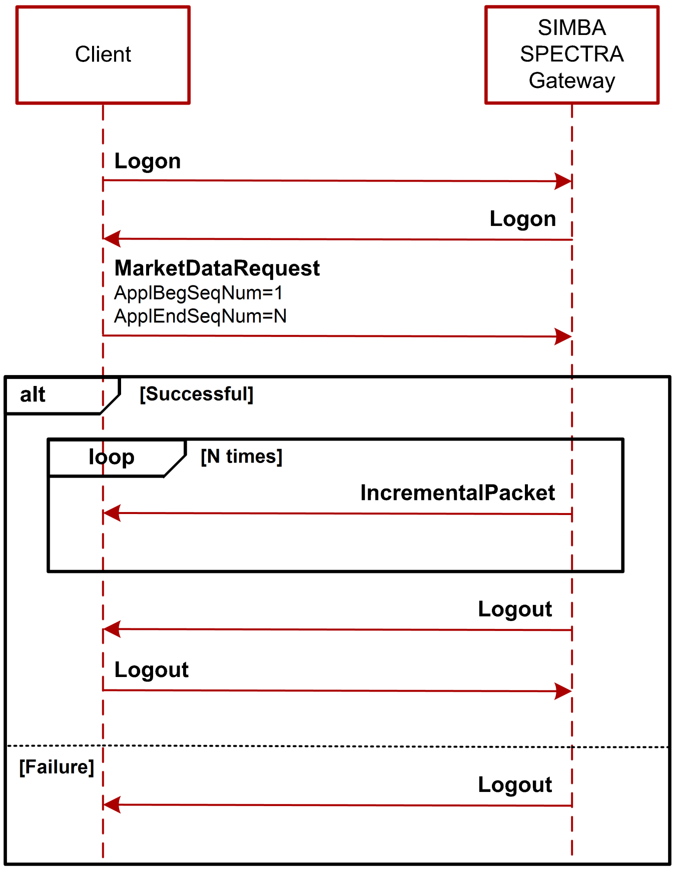
- While collecting the incremental updates, start listening to the Snapshot Feeds.
- Receive the latest snapshot.
- Merge the snapshot with the collected incremental updates as follows: for each instrument skip the incremental updates with 'MsgSeqNum' less or equal to 'LastMsgSeqNumProcessed', and then subsequently apply the rest of updates.
- Stop listening to the Snapshot Feed.

4.2.6. Using TCP Replay service for data recovery

The TCP Replay service allows clients to request a replay of messages already published on the Incremental Feed. This service is not a performance-based recovery option and should only be used to recover a relatively small number of missed messages and if any other option is not available.

Recovery scenario:

- The client opens a new TCP session and sends the 'Logon' (msg id=1000) message.
- The server confirms with the 'Logon' (msg id=1000) message.
- The client sends 'MarketDataRequest' (msg id=1002) specifying a range of the messages being requested.
- The server replays the requested messages.
- The server sends the 'Logout' (msg id=1001) message.
- The clilent confirms with the 'Logout' (msg id=1001) message.
- The server closes the TCP connection.



Pic. 13. Diagram. Data recovery with the TCP Replay service.

The messages 'Login' (msgid=1000), 'MarketDataRequest' (msg id=1002), 'Logout' (msg id=1001) have the format described in the section '2.3.2. Snapshot packet format'.

The TCP Replay service observe the following limits:

Parameter	Value	Details
Concurrent active TCP connections limit	2	New connections above the specified limit will be rejected.
Total number of TCP connection for a single IP address per day	1000	New connections above the specified limit will be rejected.
Maximum number of requested messages.	1000	Requests with the number of messages above the specified limit will be rejected
Client activity timeout	1	The TCP connection will be dropped if the specified timeout exceeded in the following cases:

Parameter	Value	Details
		• The client did not send the Logon message after opening TCP session. • The client did not send the MarketDataRequest message after the Logon message. • The client did not respond with the Logout message to the server's Logout message.

4.2.7. Data cleanup and message sequence reset

SIMBA SPECTRA Gateway clears data and resets message sequence every day during the technological break (00:00 - 05:50 UTC + 3, MSK time zone).

After cleaning and resetting the message sequence, the following messages are broadcast:

- Session-level message SequenceReset (msg id = 2) with the value "1" in the 'NewSeqNo' field. Upon receipt of this message, the client system has to set the message counter to 'NewSeqNo' and reset the 'RptSeq' update numbers to "1".
- EmptyBook message (msg id = 4) with field values: 'ExchangeTradingSessionID' = nullValue, 'LastMsgSeqNumProcessed' = 0. The client has to process this message as described below in '4.2.8. Processing the EmptyBook message'.

4.2.8. Processing the EmptyBook message

This section contains all possible variants of the EmptyBook message (msg id = 4) and description of the client reaction.

The client collects the book of active orders (order book):

Event	Field values	Client actions
Gateway start during technological break (00:00 – 05:50 UTC+3, MSK timezone)	ExchangeTradingSessionID = nullValue LastMsgSeqNumProcessed = 0	Cleans order books. Receives new order books in the form of OrderUpdate messages (msg id = 5) and then continues to update them with incremental messages.
Clearing	ExchangeTradingSessionID != nullValue LastMsgSeqNumProcessed = nullValue	Cleans order books. Receives new order books (multi-day orders, replaced in a new trading session) in the form of OrderUpdate messages (msg id = 5) and then continues to update them with incremental messages.
Recovering after a crash or unplanned event	ExchangeTradingSessionID = nullValue LastMsgSeqNumProcessed > 0	Cleans order books. Receives new order books in the form of OrderUpdate messages (msg id = 5) and then continues to update them with incremental messages.

The client collects the order log:

Event	Field values	Client actions
Gateway start during technological break (00:00 – 05:50 UTC+3, MSK timezone)	ExchangeTradingSessionID = nullValue LastMsgSeqNumProcessed = 0	Receives new order books in the form of OrderUpdate messages (msg id = 5) with the 'PossDupFlag' flag set in the 'MsgFlags' field and then continues to accumulate the log with incremental messages.
Clearing	ExchangeTradingSessionID != nullValue LastMsgSeqNumProcessed = nullValue	Receives multi-day orders replaced in a new trading session in the form of OrderUpdate messages (msg id = 5) and then continues to accumulate a log with incremental messages.
Recovering after a crash or unplanned event	ExchangeTradingSessionID = nullValue LastMsgSeqNumProcessed > 0	Ignores all previously received OrderUpdate messages (msg id = 5) with numbers greater than the number from the 'LastMsgSeqNumProcessed' field. Receives and processes new order books in the form of OrderUpdate messages (msg id = 5) with the 'PossDupFlag' flag set in the 'MsgFlags' field and then continues to accumulate the log with incremental messages.

4.2.9. Building the order-book of active orders

To build an order-book of active orders, the client has to process OrderUpdate and OrderExecution messages only, in which the 0x4 (NonQuote) flag is not set - OTC-order or OTC-trade.

5. Gateway configuration

The configuration file contains two sections describing incremental groups and two sections describing snapshot groups. The parameters are shown in the table:

Parameter	Default value	Example value	Details
type		Incremental	Multicast group type: • Incremental • Snapshot
protocol		UDP/IP	Transport protocol type.
src-ip		91.203.253.233	Source IP multicast group address.
ip		239.195.10.40	IP multicast group address.
port		44040	Multicast group port.
maxKbps	0	0	The speed upper limit (kilobits/ second) of data transmitting in the shaper group: • 0 - traffic shaping is not done for incremental groups • 1024 - shaping setting for snapshot groups
feed		A	Feed type: • A • B

The file contains one section describing the parameters of the TCP Replay service:

Parameter	Default value	Example value	Details
type		Historical Replay	Multicast group type.
protocol		TCP/IP	Transport protocol type.
ip		91.203.253.240	IP address for connecting to the TCP Replay service.
ip		91.203.253.240	Reserve IP address for connecting to the TCP Replay service.
port		7011	Port for connecting to the TCP Replay service.

6. Message schema

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="sbe_schema.xsl" type="text/xsl"?>
<sbe:messageSchema package="moex_spectra_simba" byteOrder="littleEndian" id="19780" version="1"
    semanticVersion="FIX5SP2" description="20201005" xmlns:sbe="http://fixprotocol.io/2016/sbe"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://fixprotocol.io/2016/sbe sbe.xsd">
    <types>
        <type name="uInt8" primitiveType="uint8"/>
        <type name="uInt8NULL" presence="optional" primitiveType="uint8"/>
        <type name="uInt32" primitiveType="uint32"/>
        <type name="uInt32NULL" presence="optional" primitiveType="uint32"/>
        <type name="uInt64" primitiveType="uint64"/>
        <type name="uInt64NULL" presence="optional" primitiveType="uint64"/>
        <type name="Int32" primitiveType="int32"/>
        <type name="Int32NULL" presence="optional" primitiveType="int32"/>
        <type name="Int64" primitiveType="int64"/>
        <type name="Int64NULL" presence="optional" primitiveType="int64"/>
        <type name="Char" primitiveType="char"/>
        <type name="String3" length="3" primitiveType="char"/>
        <type name="String4" length="4" primitiveType="char"/>
        <type name="String6" length="6" primitiveType="char"/>
        <type name="String25" length="25" primitiveType="char"/>
        <type name="String31" length="31" primitiveType="char"/>
        <type name="String256" length="256" primitiveType="char"/>
        <type name="DoubleNULL" presence="optional" primitiveType="double"/>
        <type name="SecurityIDSource" presence="constant" length="1"
              primitiveType="char">8</type>
        <type name="MarketID" presence="constant" length="4" primitiveType="char">MOEX</type>

        <set name="MsgFlagsSet" encodingType="uint16">
            <choice name="LastFragment"          description="Message fragmentation flag"
                    >0</choice>
            <choice name="StartOfSnapshot"       description="Flag of the first message in
                    the snapshot for the instrument"    >1</choice>
            <choice name="EndOfSnapshot"         description="Flag of the last message in
                    the snapshot for the instrument"     >2</choice>
            <choice name="IncrementalPacket"     description="Incremental packet flag"
                    >3</choice>
            <choice name="PossDupFlag"           description="Flag of the order book
                    retransmission in the incremental stream" >4</choice>
        </set>

        <composite name="MarketDataPacketHeader" description="Market Data Packet Header">
            <type name="MsgSeqNum"    primitiveType="uint32" description="Message sequence
                  number"/>
            <type name="MsgSize"      primitiveType="uint16" description="Message size
                  includes size of Market Data Packet Header"/>
            <ref  name="MsgFlags"     type="MsgFlagsSet"     />
            <type name="SendingTime"  primitiveType="uint64" description="Sending time in
                  number of nanoseconds since Unix epoch, UTC timezone"/>
        </composite>

        <composite name="IncrementalPacketHeader" description="Incremental Packet Header">
            <type name="TransactTime"             primitiveType="uint64" description="Start of
                event processing time in number of nanoseconds since Unix epoch, UTC timezone"/>
            <type name="ExchangeTradingSessionID"  primitiveType="uint32" presence="optional"
                  nullValue="4294967295" description="Trading session ID"/>
        </composite>

        <composite name="messageHeader" description="Template ID and length of message root">
            <type name="blockLength"    primitiveType="uint16"/>
            <type name="templateId"     primitiveType="uint16"/>
            <type name="schemaId"       primitiveType="uint16"/>
            <type name="version"        primitiveType="uint16"/>
        </composite>

        <composite name="groupSize" description="Repeating group dimensions"
                   semanticType="NumInGroup">
            <type name="blockLength" primitiveType="uint16"/>
            <type name="numInGroup" primitiveType="uint8"/>
        </composite>

        <composite name="Utf8String" description="Variable-length UTF-8 string">
            <type name="length" primitiveType="uint16" semanticType="Length"/>
            <type name="varData" length="0" primitiveType="uint8" semanticType="data"
                  characterEncoding="UTF-8"/>
        </composite>

        <composite name="VarString" description="Variable-length ASCII string">
            <type name="length" primitiveType="uint16" semanticType="Length"/>
            <type name="varData" length="0" primitiveType="uint8" semanticType="data"
                  characterEncoding="US-ASCII"/>
        </composite>

        <composite name="Decimal5" description="Price type in Spectra" semanticType="Price">
            <type name="mantissa" description="mantissa" primitiveType="int64"/>
            <type name="exponent" description="exponent" presence="constant"
                  primitiveType="int8">-5</type>
        </composite>

        <composite name="Decimal5NULL" description="Price type in Spectra" semanticType="Price">
            <type name="mantissa" description="mantissa" presence="optional"
                  minValue="-9223372036854775808" maxValue="9223372036854775806"
                  nullValue="9223372036854775807" primitiveType="int64"/>
            <type name="exponent" description="exponent" presence="constant"
                  primitiveType="int8">-5</type>
        </composite>

        <composite name="Decimal2NULL" description="Price type in Spectra" semanticType="Price">
            <type name="mantissa" description="mantissa" presence="optional"
                  minValue="-9223372036854775808" maxValue="9223372036854775806"
                  nullValue="9223372036854775807" primitiveType="int64"/>
            <type name="exponent" description="exponent" presence="constant"
                  primitiveType="int8">-2</type>
        </composite>

        <enum name="MDUpdateAction" encodingType="uInt8">
            <validValue name="New"    description="New"   >0</validValue>
            <validValue name="Change" description="Change">1</validValue>
            <validValue name="Delete" description="Delete">2</validValue>
        </enum>

        <enum name="MDEntryType" encodingType="Char">
            <validValue name="Bid"          description="Bid"        >0</validValue>
            <validValue name="Offer"        description="Offer"      >1</validValue>
            <validValue name="EmptyBook"    description="Empty Book">J</validValue>
        </enum>

        <enum name="SecurityAltIDSource" encodingType="Char">
            <validValue name="ISIN"           description="ISIN"           >4</validValue>
            <validValue name="ExchangeSymbol" description="Exchange symbol">8</validValue>
        </enum>

        <enum name="SecurityTradingStatus" encodingType="uInt8NULL">
            <validValue name="TradingHalt"            description="Trading halt"
                        >2</validValue>
            <validValue name="ReadyToTrade"           description="Ready to trade"
                        >17</validValue>
            <validValue name="NotAvailableForTrading" description="Not available for trading"
                        >18</validValue>
            <validValue name="NotTradedOnThisMarket"  description="Not traded on this market"
                        >19</validValue>
            <validValue name="PreOpen"                description="Pre-open"
                        >21</validValue>
            <validValue name="DiscreteAuctionOpen"    description="Discrete auction started"
                        >119</validValue>
            <validValue name="DiscreteAuctionClose"   description="Discrete auction ended"
                        >121</validValue>
       </enum>

        <enum name="TradingSessionID" encodingType="uInt8NULL">
```

```
            <validValue name="Day"     description="Day session"    >1</validValue>
            <validValue name="Morning" description="Morning session">3</validValue>
            <validValue name="Evening" description="Evening session">5</validValue>
        </enum>

        <enum name="MarketSegmentID" encodingType="Char">
            <validValue name="Derivatives" description="Derivatives">D</validValue>
        </enum>

        <enum name="TradSesStatus" encodingType="uInt8">
            <validValue name="Halted"               description="Session paused"
                        >1</validValue>
            <validValue name="Open"                 description="Session started"
                        >2</validValue>
            <validValue name="Closed"               description="Session ended"
                        >3</validValue>
            <validValue name="PreOpen"              description="Session initiated"
                        >4</validValue>
        </enum>

        <enum name="TradSesEvent" encodingType="uInt8NULL">
            <validValue name="TradingResumes"         description="Trading resumed after
                        intraday clearing session">0</validValue>
            <validValue name="ChangeOfTradingSession" description="Start and end of trading
                        session"               >1</validValue>
            <validValue name="ChangeOfTradingStatus"  description="Trading session status
                        change"                  >3</validValue>
        </enum>

        <enum name="NegativePrices" encodingType="uInt8">
            <validValue name="NotEligible" description="Futures prices, price limits and
                        options strikes are limited to be positive only">0</validValue>
            <validValue name="Eligible"    description="Futures prices and options strikes are
                        not limited"                              >1</validValue>
        </enum>

        <set name="BPFlagsSet" encodingType="uInt8">
            <choice name="BidEmptyBook"     description="Empty bid book"  >0</choice>
            <choice name="OfferEmptyBook"   description="Empty offer book">1</choice>
        </set>

        <set name="MDFlagsSet" encodingType="uInt64">
            <choice name="Day"                  description="Orders and Trades: Day order"
            <choice name="IOC"                  description="Orders and Trades: IOC order"
            <choice name="NonQuote"             description="Orders and Trades: Non quote
                    entry"                                                  >2</choice>
            <choice name="EndOfTransaction"     description="Orders and Trades: The end of
                    matching transaction"                                  >12</choice>
            <choice name="SecondLeg"            description="Trades: Second leg of multileg
                    trade"                                                 >14</choice>
            <choice name="FOK"                  description="Orders: FOK order"
            <choice name="Replace"              description="Orders: The record results from
                    replacing the order"                                  >20</choice>
            <choice name="Cancel"               description="Orders: The record results from
                    cancelling the order"                                 >21</choice>
            <choice name="MassCancel"           description="Orders: The record results from
                    mass cancelling"                                      >22</choice>
            <choice name="Negotiated"           description="Trades: Negotiated trade"
                                                                          >26</choice>
            <choice name="MultiLeg"             description="Trades: Multileg trade"
                                                                          >27</choice>
            <choice name="CrossTrade"           description="Orders: Flag of cancelling the
                    left balance of the order because of a cross-trade"      >29</choice>
            <choice name="COD"                  description="Orders: The record results from
                     cancelling an order via 'Cancel on Disconnect' service" >32</choice>
            <choice name="ActiveSide"           description="Trades: Flag of aggressive side"
                                                                          >41</choice>
            <choice name="PassiveSide"          description="Trades: Flag of passive side"
                                                                          >42</choice>
            <choice name="Synthetic"            description="Orders and Trades: Flag of the
                   synthetic order"                                       >45</choice>
            <choice name="RFS"                  description="Orders and Trades: RFS is the
```

```
                source of entry"                                        >46</choice>
        <choice name="SyntheticPassive"      description="Orders: Flag of the passive
              synthetic order"                                          >57</choice>
        <choice name="BOC"                   description="Orders: Book or Cancel order"
                                                                        >60</choice>
        <choice name="DuringDiscreteAuction" description="Orders and Trades: The record
                    formed in the process of discrete auction"          >62</choice>
    </set>

    <set name="FlagsSet" encodingType="uInt64">
        <choice name="EveningOrMorningSession" description="Trading in the evening or
                morning session">0</choice>
        <choice name="AnonymousTrading"        description="Anonymous trading"
                >4</choice>
        <choice name="PrivateTrading"          description="Private trading"
                >5</choice>
        <choice name="DaySession"              description="Trading in the day session"
                >6</choice>
        <choice name="MultiLeg"                description="MultiLeg instrument"
                >8</choice>
        <choice name="Collateral"              description="Collateral instrument"
                >18</choice>
        <choice name="IntradayExercise"        description="Exercise in the intraday
                clearing session">19</choice>
    </set>
</types>

<sbe:message name="Heartbeat" id="1" description="Heartbeat" semanticType="0"/>

<sbe:message name="SequenceReset" id="2" description="SequenceReset" semanticType="4">
    <field name="NewSeqNo"                     id="36"    type="uInt32"
           description="New sequence number"/>
</sbe:message>

<sbe:message name="BestPrices" id="3" semanticType="X">
    <group name="NoMDEntries"              id="268"   dimensionType="groupSize"
           description="Number of entries in Best Prices message">
        <field name="MktBidPx"             id="645"   type="Decimal5NULL"
               description="Best bid price"/>
        <field name="MktOfferPx"           id="646"   type="Decimal5NULL"
               description="Best offer price"/>
        <field name="BPFlags"              id="22000" type="BPFlagsSet"
               description="The field is a bit mask"/>
        <field name="SecurityID"           id="48"    type="Int32"
               description="Instrument numeric code"/>
    </group>
</sbe:message>

<sbe:message name="EmptyBook" id="4" semanticType="X">
    <field name="LastMsgSeqNumProcessed"   id="369"   type="uInt32NULL"
           description="Sequence number of the last valid Incremental feed packet"/>
</sbe:message>

<sbe:message name="OrderUpdate" id="5" semanticType="X">
    <field name="MDEntryID"                id="278"   type="Int64"
           description="Order ID"/>
    <field name="MDEntryPx"                id="270"   type="Decimal5"
           description="Order price"/>
    <field name="MDEntrySize"              id="271"   type="Int64"
           description="Market Data entry size"/>
    <field name="MDFlags"                  id="20017" type="MDFlagsSet"
           description="The field is a bit mask"/>
    <field name="SecurityID"               id="48"    type="Int32"
           description="Instrument numeric code"/>
    <field name="RptSeq"                   id="83"    type="uInt32"
           description="Market Data entry sequence number per instrument update"/>
    <field name="MDUpdateAction"           id="279"   type="MDUpdateAction"
           description="Market Data update action"/>
    <field name="MDEntryType"              id="269"   type="MDEntryType"
           description="Market Data entry type"/>
</sbe:message>
```

```
    <sbe:message name="OrderExecution" id="6" semanticType="X">
        <field name="MDEntryID"                id="278"   type="Int64"
               description="Order ID"/>
        <field name="MDEntryPx"                id="270"   type="Decimal5NULL"
               description="Order price"/>
        <field name="MDEntrySize"              id="271"   type="Int64NULL"
               description="Market Data entry size"/>
        <field name="LastPx"                   id="31"    type="Decimal5"
               description="Matched trade price"/>
        <field name="LastQty"                  id="32"    type="Int64"
               description="Trade volume"/>
        <field name="TradeID"                  id="1003"  type="Int64"
               description="Trade ID"/>
        <field name="MDFlags"                  id="20017" type="MDFlagsSet"
               description="The field is a bit mask"/>
        <field name="SecurityID"               id="48"    type="Int32"
               description="Instrument numeric code"/>
        <field name="RptSeq"                   id="83"    type="uInt32"
               description="Market Data entry sequence number per instrument update"/>
        <field name="MDUpdateAction"           id="279"   type="MDUpdateAction"
               description="Market Data update action"/>
        <field name="MDEntryType"              id="269"   type="MDEntryType"
               description="Market Data entry type"/>
    </sbe:message>

    <sbe:message name="OrderBookSnapshot" id="7" semanticType="W">
        <field name="SecurityID"                   id="48"    type="Int32"
               description="Instrument numeric code"/>
        <field name="LastMsgSeqNumProcessed"       id="369"   type="uInt32"
               description="Sequence number of the last Incremental feed packet processed.
               This value is used to synchronize the snapshot loop with the real-time feed"/>
        <field name="RptSeq"                       id="83"    type="uInt32"
               description="Market Data entry sequence number per instrument update"/>
        <field name="ExchangeTradingSessionID"     id="5842"  type="uInt32"
               description="Trading session ID"/>
        <group name="NoMDEntries"                  id="268"   dimensionType="groupSize"
               description="Number of entries in Market Data message">
            <field name="MDEntryID"                id="278"   type="Int64NULL"
                   description="Order ID"/>
            <field name="TransactTime"             id="60"    type="uInt64"
                   description="Start of event processing time in number of nanoseconds since
                   Unix epoch, UTC timezone"/>
            <field name="MDEntryPx"                id="270"   type="Decimal5NULL"
                   description="Order price"/>
            <field name="MDEntrySize"              id="271"   type="Int64NULL"
                   description="Market Data entry size"/>
            <field name="TradeID"                  id="1003"  type="Int64NULL"
                   description="Trade ID"/>
            <field name="MDFlags"                  id="20017" type="MDFlagsSet"
                   description="The field is a bit mask"/>
            <field name="MDEntryType"              id="269"   type="MDEntryType"
                   description="Market Data entry type"/>
        </group>
    </sbe:message>

    <sbe:message name="SecurityDefinition" id="12" semanticType="d">
        <field name="TotNumReports"                id="911"   type="uInt32"
               description="Total messages number in the current list"/>
        <field name="Symbol"                       id="55"    type="String25"
               description="Symbol code of the instrument"/>
        <field name="SecurityID"                   id="48"    type="Int32"
               description="Instrument numeric code"/>
        <field name="SecurityIDSource"             id="22"    type="SecurityIDSource"
               description="Identifies class or source of tag 48-SecurityID value"/>
        <field name="SecurityAltID"                id="455"   type="String25"
               description="Instrument symbol code"/>
        <field name="SecurityAltIDSource"          id="456"   type="SecurityAltIDSource"
               description="Class of tag 455-SecurityAltID"/>
        <field name="SecurityType"                 id="167"   type="String4"
               description="Multileg type"/>
        <field name="CFICode"                      id="461"   type="String6"
               description="Financial instrument class according to ISO-10962"/>
        <field name="StrikePrice"                  id="202"   type="Decimal5NULL"
              description="Strike price"/>
        <field name="ContractMultiplier"           id="231"   type="Int32NULL"
               description="Units of underlying asset in instrument"/>
        <field name="SecurityTradingStatus"        id="326"   type="SecurityTradingStatus"
               description="Identifies the trading status of instrument"/>
        <field name="Currency"                     id="15"    type="String3"
               description="Currency"/>
        <field name="MarketID"                     id="1301"  type="MarketID"
               description="Identifies the market"/>
        <field name="MarketSegmentID"              id="1300"  type="MarketSegmentID"
               description="Identifies the market segment"/>
        <field name="TradingSessionID"             id="336"   type="TradingSessionID"
               description="Trading session type"/>
        <field name="ExchangeTradingSessionID"     id="5842"  type="Int32NULL"
               description="Trading session ID"/>
        <field name="Volatility"                   id="5678"  type="Decimal5NULL"
               description="Option volatility"/>
        <field name="HighLimitPx"                  id="1149"  type="Decimal5NULL"
               description="Upper price limit"/>
        <field name="LowLimitPx"                   id="1148"  type="Decimal5NULL"
               description="Lower price limit"/>
        <field name="MinPriceIncrement"            id="969"   type="Decimal5NULL"
               description="Minimum price step"/>
        <field name="MinPriceIncrementAmount"      id="1146"  type="Decimal5NULL"
               description="Price step cost in RUB"/>
        <field name="InitialMarginOnBuy"           id="20002" type="Decimal2NULL"
               description="Initial margin"/>
        <field name="InitialMarginOnSell"          id="20000" type="Decimal2NULL"
               description="Initial margin"/>
        <field name="InitialMarginSyntetic"        id="20001" type="Decimal2NULL"
               description="Underlying collateral for one uncovered position (RUB)"/>
        <field name="TheorPrice"                   id="20006" type="Decimal5NULL"
               description="Option theoretical price"/>
        <field name="TheorPriceLimit"              id="20007" type="Decimal5NULL"
               description="Option theoretical price (limits adjusted)"/>
        <field name="UnderlyingQty"                id="879"   type="Decimal5NULL"
               description="Security nominal value"/>
        <field name="UnderlyingCurrency"           id="318"   type="String3"
               description="Code of currency of the security nominal value"/>
        <field name="MaturityDate"                 id="541"   type="uInt32NULL"
               description="Instrument settlement date"/>
        <field name="MaturityTime"                 id="1079"  type="uInt32NULL"
               description="Instrument settlement time"/>
        <field name="Flags"                        id="20008" type="FlagsSet"
               description="Flags of instrument"/>
        <field name="MinPriceIncrementAmountCurr"  id="20040" type="Decimal5NULL"
               description="Value of the minimum increment in foreign currency"/>
        <field name="SettlPriceOpen"               id="20041" type="Decimal5NULL"
               description="Settlement price at the start of the session"/>
        <field name="ValuationMethod"              id="1197"  type="String4"
               description="Specifies the type of valuation method applied"/>
        <field name="RiskFreeRate"                 id="1190"  type="DoubleNULL"
               description="Risk free interest rate"/>
        <field name="FixedSpotDiscount"            id="20042" type="DoubleNULL"
               description="The sum of the discounted values of the declared cash flows"/>
        <field name="ProjectedSpotDiscount"        id="20043" type="DoubleNULL"
               description="The sum of the discounted values of the projected cash flows"/>
        <field name="SettlCurrency"                id="120"   type="String3"
               description="Settlement currency"/>
        <field name="NegativePrices"               id="20044" type="NegativePrices"
               description="Negative prices eligibility"/>
        <field name="DerivativeContractMultiplier" id="1266"  type="Int32NULL"
               description="Coefficient indicating the volume of the underlying asset in the
               contract quote and strikes of option series"/>
        <group name="NoMDFeedTypes"                id="1141"  dimensionType="groupSize"
               description="Number of feed types">
            <field name="MDFeedType"               id="1022"  type="String25"
                   description="Feed type"/>
            <field name="MarketDepth"              id="264"   type="uInt32NULL"
                   description="Order-book depth"/>
            <field name="MDBookType"               id="1021"  type="uInt32NULL"
```

```
                 description="Order-book type"/>
      </group>
      <group name="NoUnderlyings"                id="711"   dimensionType="groupSize"
             description="Number of underlyings">
          <field name="UnderlyingSymbol"         id="311"   type="String25"
                 description="Underlying asset code"/>
          <field name="UnderlyingBoard"          id="20045" type="String4"
                 description="Underlying board code"/>
          <field name="UnderlyingSecurityID"     id="309"   type="Int32NULL"
                 description="Futures instrument ID"/>
          <field name="UnderlyingFutureID"       id="2620"  type="Int32NULL"
                 description="ID of the base futures instrument, applicable to options only"/>
      </group>
      <group name="NoLegs"                       id="555"   dimensionType="groupSize"
             description="Nymber of legs">
          <field name="LegSymbol"                id="600"   type="String25"
                 description="Multileg instrument's individual security's Symbol"/>
          <field name="LegSecurityID"            id="602"   type="Int32"
                 description="Multileg instrument's individual security's SecurityID"/>
          <field name="LegRatioQty"              id="623"   type="Int32"
                 description="The ratio of quantity for this individual leg relative to the
                 entire multileg security"/>
      </group>
      <group name="NoInstrAttrib"                id="870"   dimensionType="groupSize"
             description="Number of attributes">
          <field name="InstrAttribType"          id="871"   type="Int32"
                 description="Code to represent the type of instrument attribute"/>
          <field name="InstrAttribValue"         id="872"   type="String31"
                 description="Attribute value appropriate to tag 87-InstrAttribType"/>
      </group>
      <group name="NoEvents"                     id="864"   dimensionType="groupSize"
             description="Number of events">
          <field name="EventType"                id="865"   type="Int32"
                 description="Code to represent the type of event"/>
          <field name="EventDate"                id="866"   type="uInt32"
                 description="Date of event"/>
          <field name="EventTime"                id="1145"  type="uInt64"
                 description="Time of event"/>
      </group>
      <data name="SecurityDesc"                 id="107"   type="Utf8String"
            description="Instrument name"/>
      <data name="QuotationList"                id="20005" type="VarString"
            description="Quotation list"/>
  </sbe:message>

  <sbe:message name="SecurityStatus" id="9" semanticType="f">
      <field name="SecurityID"                   id="48"    type="Int32"
             description="Instrument numeric code"/>
      <field name="SecurityIDSource"             id="22"    type="SecurityIDSource"
             description="Identifies class or source of tag 48-SecurityID value"/>
      <field name="Symbol"                       id="55"    type="String25"
             description="Symbol code of the instrument"/>
      <field name="SecurityTradingStatus"        id="326"   type="SecurityTradingStatus"
             description="Identifies the trading status of instrument"/>
      <field name="HighLimitPx"                  id="1149"  type="Decimal5NULL"
             description="Upper price limit"/>
      <field name="LowLimitPx"                   id="1148"  type="Decimal5NULL"
             description="Lower price limit"/>
      <field name="InitialMarginOnBuy"           id="20002" type="Decimal2NULL"
             description="Initial margin"/>
      <field name="InitialMarginOnSell"          id="20000" type="Decimal2NULL"
             description="Initial margin"/>
      <field name="InitialMarginSyntetic"        id="20001" type="Decimal2NULL"
             description="Underlying collateral for one uncovered position (RUB)"/>
  </sbe:message>

  <sbe:message name="SecurityDefinitionUpdateReport" id="10" semanticType="BP">
      <field name="SecurityID"                   id="48"    type="Int32"
             description="Instrument numeric code"/>
      <field name="SecurityIDSource"             id="22"    type="SecurityIDSource"
             description="Identifies class or source of tag 48-SecurityID value"/>
      <field name="Volatility"                   id="5678"  type="Decimal5NULL"
```

```
               description="Option volatility"/>
        <field name="TheorPrice"                   id="20006" type="Decimal5NULL"
               description="Option theoretical price"/>
        <field name="TheorPriceLimit"              id="20007" type="Decimal5NULL"
               description="Option theoretical price (limits adjusted)"/>
    </sbe:message>

    <sbe:message name="TradingSessionStatus" id="11" semanticType="f">
        <field name="TradSesOpenTime"                id="342"   type="uInt64"
               description="Trading session open date and time"/>
        <field name="TradSesCloseTime"               id="344"   type="uInt64"
               description="Trading session close date and time"/>
        <field name="TradSesIntermClearingStartTime" id="5840"  type="uInt64NULL"
               description="Intraday clearing session start date and time"/>
        <field name="TradSesIntermClearingEndTime"   id="5841"  type="uInt64NULL"
               description="Intraday clearing session end date and time"/>
        <field name="TradingSessionID"               id="336"   type="TradingSessionID"
               description="Trading session type"/>
        <field name="ExchangeTradingSessionID"       id="5842"  type="Int32NULL"
               description="Trading session ID"/>
        <field name="TradSesStatus"                  id="340"   type="TradSesStatus"
               description="State of the trading session"/>
        <field name="MarketID"                       id="1301"  type="MarketID"
               description="Identifies the market"/>
        <field name="MarketSegmentID"                id="1300"  type="MarketSegmentID"
               description="Identifies the market segment"/>
        <field name="TradSesEvent"                   id="1368"  type="TradSesEvent"
               description="Identifies an event related to tag 340-TradSesStatus"/>
    </sbe:message>

    <sbe:message name="DiscreteAuction" id="13" semanticType="U1">
        <field name="TradSesOpenTime"                id="342"   type="uInt64"
               description="Discrete auction open date and time"/>
        <field name="TradSesCloseTimeFrom"           id="20046" type="uInt64"
               description="Discrete auction closing interval start date and time"/>
        <field name="TradSesCloseTimeTill"           id="20047" type="uInt64"
               description="Discrete auction closing interval end date and time"/>
        <field name="AuctionID"                      id="21002" type="Int64"
               description="Discrete auction ID"/>
        <field name="ExchangeTradingSessionID"       id="5842"  type="Int32"
               description="Trading session ID"/>
        <field name="EventIDOpen"                    id="20048" type="Int32"
               description="Discrete auction open event ID"/>
        <field name="EventIDClose"                   id="20049" type="Int32"
               description="Discrete auction close event ID"/>
        <group name="NoUnderlyings"                  id="711"   dimensionType="groupSize"
               description="Number of underlyings">
            <data name="UnderlyingSymbol"            id="311"   type="VarString"
               description="Underlying asset code"/>
        </group>
    </sbe:message>

    <sbe:message name="Logon" id="1000" semanticType="A"/>

    <sbe:message name="Logout" id="1001" semanticType="5">
        <field name="Text" id="58" type="String256"
               description="Free format text string. May include logout confirmation or reason
               for logout"/>
    </sbe:message>

    <sbe:message name="MarketDataRequest" id="1002" semanticType="V">
        <field name="ApplBegSeqNum"                id="1182"  type="uInt32"
               description="Sequence number of the first requested message"/>
        <field name="ApplEndSeqNum"                id="1183"  type="uInt32"
               description="Sequence number of the last requested message"/>
    </sbe:message>

</sbe:messageSchema>
```