Approved
on 30 January 2023

CUSTOMER SOFTWARE CERTIFICATION PROCEDURE
(the "Procedure")

Table of Contents

Procedure description and steps

This Procedure has been designed to comply with clauses 6.20 and 6.30 – 6.34 of the Know Your Customer Policy (the "Policy").

The following terms are used in this Procedure:

- ASTS means the trading and clearing system of the FX market, the Precious Metals Market, the Securities Market, and the Money Market. ASTS includes both the ASTS system and the following generations of the trading and clearing platforms ASTS+ and Rebus, and also modules which work on the mentioned markets in separate trading modes, such as FX OTC and FX RFS.
- LC means the liquidity consumer, active counterparty in trade execution.
- LP means the liquidity provider, passive counterparty in trade execution.
- SPECTRA means the Derivatives Market trading and clearing system. SPECTRA includes both the SPECTRA system and additional modules such as OTC system of Derivatives market.
- Customer software – software used for connecting to the Moscow Exchange hardware-software complex systems by the program protocols provided by the Moscow Exchange.
- Client – an individual or a legal entity who applies for the certification of the software developed by himself and which uses MOEX programming protocol.
- User of the customer software (the "User") means any legal entity for which Moscow Exchange provides services on the basis of an agreement for integrated technological service (ITS).
- Rules means any trading rules and/or clearing (settlement) rules and/or other inner documents of the market operator and/or clearing house which set out how orders are entered and filled, and the trades are cleared and settled.
- Software and Hardware Suite (SHS) - set of software and hardware means of the Moscow Exchange used for trading, settlement and provision of other client services.

Any other terms and definitions used in this Procedure have meanings ascribed to them by the Policy and Moscow Exchange's documents defining the technical access to the SHS.

1. General provisions
    - The Exchange provides testing scenarios for connecting customer software to the SPECTRA and ASTS trading and clearing systems (TCS), and also to other subsystems operating in the Exchange SHS.

    - To be certified a customer software must meet requirements given in clause 2 hereof and in any case must perform any mechanisms described in the Moscow Exchange Technical Center connectivity requirements for customer software (the "Requirements").

2. Customer software requirements

- A log file must be maintained to include all operations made with such software and hardware while interacting with the SHS, and results of such operations (entering/withdrawing orders, executing trades, receiving market data, etc.)

- Procedures for recovery cases must be implemented:

    - Interruption of a network connection to the TCS,
    - Software restart during the trading day,
    - the TCS restart which requires data reloading,
    - the TCS restart which does not require data reloading,
    - switch to the reserve and backup TCS access servers (may be done manually)

- The User must have administration and monitoring functions available in software intended for client operations (for broker systems)
- Names of tables, transactions and fields in the software must provide one-to-one mapping to the terms used in the documents of Moscow Exchange and NAMEX (Joint-Stock Company "National Mercantile Exchange") regulating trades in the relevant market, and in the documents of the Exchange and the National Clearing Centre (CCP NCC) regulating the clearing in the relevant market, and also in the documents of the Moscow Exchange regulating the technical access to the SHS.

- If the User or his client indicates a specific sub-system of the SHS at order entry then the software must:
    - ❖ Identify that order as being prepared for entry to a relevant SHS sub-system.
    - ❖ Provide that the order is sent only to a relevant sub-system to be included in the unified order book of only one sub-system to lead to a deal in that SHS sub-system provided that the order meets the requirements set out in the Rules.

- A software intended to interact with ASTS via native ASTS Bridge protocol must provide a limiting option for the frequency of data requests to the SHS. If the customer software is not configured to limit frequency of data requests (colocation or use of personal Gateway servers) that requirement is not applied, but the customer software will be certified only for use with such configurations.
- Requirements not specified for a certain market/trading and clearing system, are to be applied to any customer software interacting with any trading and clearing system.

3. Applying for certification. General certification procedure.
    - A software developer should send documents and information stated in Appendix 3 to the Policy to help@moex.com
    - Request connection to test environments by filling in a form at MOEX web site. Contact help@moex.com for assistance.
    - When being certified for connection to SPECTRA trading and clearing system via CGate API the Exchange will send an additional inquiry form.

    - Activation of logging in the software when using native protocols:

SPECTRA	ASTS
For CGate API: Default logging level must be set in router logging configuration file.	For ASTS Bridge: Configure the external system to log client transactions in the bridge by setting logging = 2,1 for MTEConnect function (or contact the developer to find out how such logs are created)

- Testing the customer software in a mode mirroring the production use in accordance with the following scenario (the testing date should be agreed with the MOEX support team).

SPECTRA	ASTS
The application must trade in the test environment throughout a trading day including the interday and evening clearing sessions, and after-hours trading session and send all commands specified in the Exchange inquiry form.	The application must perform a standard operation cycle provided by the system in the production environment during 5-10 minutes. When using ASTS Bridge the frequency of data requests must be available for tuning and must be changed per requests of the support team.

- Further certification procedure of the customer software runs in accordance with Know Your Customer Policy.

Appendix 1. Certification tests for Native APIs

Plaza II

When any customer software with CGate API is certified, functions specified in the additional inquiry form sent by the Exchange under Appendix 1 are verified. Answers given in the inquiry form allow estimating the following software parameters:

Connecting to Plaza-II

1. The application's URL connection initialization is correct.
   a. The Client specifies URL-connections in the inquiry form
   b. URL accuracy and compliance with the inquiry form are verified by the client application log.
2. The application does not break connection rules in the multithreading environment
   a. The Client specifies number of threads and how publishers are distributed by threads and connections.
   b. Connections from different threads and compliance with the inquiry form are verified by the client application log.
3. The application activates the polling function at necessary frequency.
   a. The application stays connected with the router during five minutes in case of inactive incoming and outgoing replication streams
4. The application connects well to an enabled router authenticated in the Plaza 2 network.
   a. The application successfully connects and switches to take data streams and send commands after the router is logged in.
5. The application connects to an enabled router without connection to the Plaza 2 and detects the fact of the disconnection.
   a. The application connects successfully to the router and waits for network connection.
6. The application detects connection to a Plaza 2 router.
   a. Upon being connected to a router, the active application detects connection to Plaza 2 network and switches to take data streams and send commands.
7. The application detects disconnection with Plaza 2 network.
   a. The application detects broken connection to Plaza 2 and switches to a standby mode. Data streams and commands are suspended for that time.
8. The application detects broken connection to a router.
   a. The application detects broken connection to the router and switches to a standby mode. Data streams and commands are suspended for that time.

Receiving replicas

1. The application's URL connection initialization and subscriptions opening are correct.
   a. The Client specifies URLs used for subscriptions in the inquiry form.
   b. URL accuracy and compliance with the inquiry form are verified by the client application log.

2. The application does not break subscription rules in the multithreading environment
    a. The Client specifies number of subscriptions and how they are distributed by streams and connections.
    b. Subscriptions from different threads and compliance with the inquiry form are verified by the client application log.
3. The client-side application indicates valid schemes at initialization.
4. The server-side application processes correctly compatible changes in the server circuit
    a. Extension of the server circuit through new fields and tables does not result in application failures.
5. The server-side application detects accurately any incompatible changes in the server circuit and reports them after diagnostics.
    a. The application detects correctly whether any fields and tables have been deleted or field types have been changed, and does not behaves indefinitely if any of these has taken place.
6. The application detects broken connection and reopens threads correctly.
    a. The application verifies the manner for every of incoming threads to be reopened.
7. The application designed to get the full ORDERS_LOG is capable to process no less than 100 000 messages per sec.
    a. A data stream is fed at a specific frequency to verify whether the application processes it without delays.
8. The application must process correctly the messages:
    a. ClearDeletedMessage – the message about the data range deletion in the specified table
    b. LifeNumMessage – the message about the life number change of the data scheme

Sending commands

1. The application's URL connection initialization and publishers opening are correct.
    a. The Client specifies publishers' URLs in the inquiry form.
    b. URL accuracy and compliance with the inquiry form are verified by the client application log.
2. The application does not break publisher rules in the multithreading environment.
    a. The Client specifies number of threads and how the publishers are distributed by threads and connections.
    b. Publishers from different threads and compliance with the inquiry form are
    verified by the client application log.
3. The application includes a configurable rate control mechanism applied in sending commands.
    a. The application is configured to send messages as frequent as specified to verify any overshoot.
4. The application indicates a valid scheme when sending commands.
5. The application processes correctly reply messages and processing timeouts.

    a. The application must not behave indefinitely upon experiencing a reply message timeout.
6. The application processes correctly type 99 and type 100 messages.
    a. The application must not behave indefinitely upon receiving type 99 and type 100 messages.
7. The application detects broken connection and reopens publishers accurately.

ASTS Bridge

The following items are validated in using ASTS Bridge:

1. Correctness of data requests (opening and refreshing tables).
2. Valid disconnection from the trading and clearing system after the end of the work.
3. Absence of any errors at order entry raised by the transaction invalid generation.
4. Use of the latest bridge interface version at the moment of certification.
5. The application must initiate password change in interacting with ASTS. This is a mandatory item for successful certification.

Appendix 2. Scenario for customer software connected to SPECTRA FIX services

Test 1. Connection initialization and disconnection

Test 1-1. Session initialization

The client must get connected and introduce itself to the system by entering its **SenderCompID** obtained from the Exchange.
The client-side software must send **Logon** and **HeartBeat** messages. This is an automatic test without user actions required.

Test 1-2. Logging out

The user must log out correctly and log in again.

Test 2. Interaction in trading

Test 2-1. New futures order

On a basic level, an order is accepted by the system, and then the following steps are performed:

Step	Sender	Note
1	Client	The client sends a **limit** order to **buy** a test futures <**out of any futures available in the testing environment**> at a price **<within the limits>** with **<the Client's account>** used. The order size is 5 contracts.
2	System	The system sends a message based on a business logic

Test 2-2. New long-term order to sell futures

A GTD order is placed:

Step	Sender	Note
1	Client	The client sends a **limit** order to **sell** a test futures **<out of any futures available in the testing environment>** at a price **<within the limits>** with **<the Client's account>** used. The order size is 6 contracts. The order is **Good Till Date** order. The expiration date is not specified.
2	System	The system sends a message based on a business logic

Test 2-3. Working order cancellation

The order placed in test 2-2 is cancelled by the Client for OrigClOrdID:

Step	Sender	Note
1	Client	The client initiates the cancellation of a working order for OrigClOrdID received (ClOrdID of the previously entered order) (selling six contracts)
2	System	The system sends a message based on a business logic

Test 2-4. New option order

Step	Sender	Note
1	Client	The client enters a **limit** order to **buy** an option **<out of any options available in the testing environment>**.The order size is one contract from **<the Client's account>.**
2	System	The system sends a message based on a business logic

Test 2-5. Order replacement

The client sends a long-term order to buy an option and change the order parameters for OrigClOrdID

Step	Sender	Note
1	Client	The client enters a **limit** order to buy an option **<out of any options available in the testing environment>**. The order size is **10** contracts from **<the Client's account>**. The order is **Good Till Date** order. The expiration date is not fixed.
2	System	The system sends a message based on a business logic
3	Client	The client changes the price of the previously entered order for **OrigClOrdID** received (**ClOrdID** of the previously entered order). The newly entered price must differ from the previous one.
4	System	The system sends a message based on a business logic

Test 2-6. Placement of a multi-leg instrument order *

*applied only to clients wishing to trade multi-leg instruments, for the other clients this paragraph is not mandatory

The client enters an order in a multi-leg instrument

Step	Sender	Note
1	Client	The client enters a **limit** order to buy a multi-leg instrument **<out of any instruments available in the testing environment>**. The order size is **10** contracts. The **"client account"** is used.
2	System	The system sends a message based on a business logic

Test 2-7. Order status request

Receiving status for the order entered in tests 2-5

Step	Sender	Note
1	Client	The client requests the status of the order entered in test 2-5 for **ClOrdID** received
2	System	The system sends a message based on a business logic

Test 2-8. Order mass cancellation

Cancelling all available orders accepted during the testing. Orders in multi-leg instruments cannot be cancelled with this message.

Step	Sender	Note
1	Client	The client requests mass order cancellation by indicating **<the Client's account>** and relevant **MassCancelRequestType=8** and **MarketSegmentID=F**
2	System	The system sends a message based on business logic. Only **futures** orders are cancelled
3	Client	The client requests mass order cancellation by indicating **<the Client's account>** and relevant **MassCancelRequestType=8** and **MarketSegmentID=O**
4	System	The system sends a message based on business logic. Only **option** orders are cancelled

Appendix 3. Scenario for customer software connected to SPECTRA DropCopy FIX services

The Drop Copy service sends **execution** reports in two formats:

a. only deals
b. orders and deals

The client may choose the format. Clients wishing to get only trades are subject to tests **1, 2-7, 2-8, 2-9.** Other clients should perform all the tests. Tests 2-6 and 2-9 are applied only to clients wishing to trade multi-leg instruments. Trading operations in orders may be performed via both FIX Gate of the Derivatives Market and any other services available on the Derivatives market. Before testing, please make sure that your Drop Copy ID is linked to a trading section for which reports are expected.

Test 1. Connection initialization and disconnection

Test 1-1. Session initialization

The client must get connected and introduce itself to the system by entering its **SenderCompID** obtained from the Exchange.
The client-side software must send **Logon** and **HeartBeat** messages. This is an automatic test without actions required on the user side.

Test 1-2. Session logout

The user must log out correctly and log in again.

Test 2. Interaction in trading

Test 2-1. New futures order

On a basic level, an order is accepted by the system, and then the following steps are performed:

Step	Sender	Note
1	Client	The client sends a **limit** order to **buy** a test **futures <out of any futures available in the testing environment>** at a price **<within the limits>** with **<the Client's account>** used. The order size is **5** contracts.
2	System	Relevant ExecutionReport is awaited from the Drop Copy service

Test 2-2. New long-term order to sell futures

A GTD order is placed:

Step	Sender	Note
1	Client	The client sends a **limit** order to **sell** a test **futures <out of any futures available in the testing environment>** at a price **<within the limits>** with **<the Client's account>** used. The order size is **6** contracts. The order is **Good Till Date** order. The expiration date is not specified.
2	System	Relevant ExecutionReport is awaited from the Drop Copy service

Test 2-3. Working order cancellation

The order placed in test 2-2 is cancelled by the Client:

Step	Sender	Note
1	Client	The client initiates the cancellation of a working order (selling **6** contracts)
2	System	Relevant ExecutionReport is awaited from the Drop Copy service

Test 2-4. New option order

Step	Sender	Note
1	Client	The client enters a **limit** order to **buy** an option **<out of any options available in the testing environment>**. The order size is **1** contract from **<the Client's account>.**
2	System	Relevant ExecutionReport is awaited from the Drop Copy service

Test 2-5. Order replacement

The client sends a long-term order to buy an option and then change the order parameters

Step	Sender	Note
1	Client	The client enters a **limit** order to **buy** an option **<out of any options available in the testing environment>**. The order size is **10** contracts from **<the Client's account>**. The order is **Good Till Date** order. The expiration date is not fixed.

2	System	The system sends a message based on business logic.
3	Client	The client changes the price of the previously entered order. The newly entered price must differ from the previous one.
4	System	Relevant ExecutionReport is awaited from the Drop Copy service

Test 2-6. Placement of a multi-leg instrument order*

*applied only to clients wishing to trade multi-leg instruments, for the other clients this paragraph is not mandatory

Step	Sender	Note
1	Client	The client enters a **limit** order to **buy** a multi-leg instrument <**out of any instruments available in the testing environment>.** The order size is **10** contracts. The **"client account"** is used.
2	System	Relevant ExecutionReport is awaited from the Drop Copy service

Test 2-7. Partial filling

Step	Sender	Note
1	Client	The client enters a **limit** order to **buy/sell futures** to fill the order **in part**
2	System	Relevant ExecutionReport is awaited from the Drop Copy service

Test 2-8. Full filling

Step	Sender	Note
1	Client	The client enters a **limit** order to **buy/sell futures** to fill the order **in full**
2	System	Relevant ExecutionReport is awaited from the Drop Copy service

Test 2-9. Trades in a multi-leg instrument*

*applied only to clients wishing to trade multi-leg instruments, for the other clients this paragraph is not mandatory

Step	Sender	Note
1	Client	Perform tests 2-7 and 2-8 for a **multi-leg** instrument
2	System	Relevant ExecutionReport is awaited from the Drop Copy service

Appendix 4. Scenario for customer software connected to SPECTRA OTCGate FIX services (RFS service)

Test 1. Connection initialization and disconnection

Test 1-1. Session initialization

The client must get connected and introduce itself to the system by entering its **SenderCompID** obtained from the Exchange.
The client-side software must send **Logon** and **HeartBeat** messages. This is an automatic test without actions required on the user side.

Test 1-2. Session logout

The user must log out correctly and log in again.

Test 2. Interaction in trading

Test 2-1. LP to add quote

Step	Sender	Note
0		**Bidirectional** quote stream must be opened for a test i**nstrument <out of any instruments available in the testing environment>**
1	Client LP	The client sends a **buy quote** and a **sell quote** for a test **instrument** at a price **<within the limits>** with **<the Client's account>** used.
2	System	The system sends a message based on business logic.

Test 2-2. LP to add a buy quote

Step	Sender	Note
0		**Unidirectional** quote stream to **sell** must be opened for a test instrument **<out of any instruments available in the testing environment>**
1	**Client LP**	The client sends a **buy quote** for a test instrument **<out of any futures available in the testing environment>** at a price **<within the limits>** with **<the Client's account>** used.
2	System	The system sends a message based on business logic.

Test 2-3. LP to add a sell quote

Step	Sender	Note
0		**Unidirectional** quote stream to **buy** must be opened for a test instrument **<out of any instruments available in the testing environment>**
1	**Client LP**	The client sends a **sell quote** for a test instrument **<out of any futures available in the testing environment>** at a price **<within the limits>** with **<the Client's account>** used.
2	System	The system sends a message based on business logic.

Test 2-4. LP to move their quote

Step	Sender	Note
0		**Unidirectional** quote stream to **buy** must be opened for a test instrument **<out of any instruments available in the testing environment>**
1	**Client LP**	The client adds a **sell quote** for a test instrument **<out of any futures available in the testing environment>** at a price **<within the limits>** with **<the Client's account>** used.
2	System	The system sends a message based on business logic.
3	**Client LP**	The client moves their added **sell quote** for a test instrument **<out of any futures available in the testing environment>** at a different price **<within the limits>** with **<the Client's account>** used.
4	System	The system sends a message based on business logic.

Test 2-5. LP to delete their quote

The client sends a long-term order to buy an option and then change the order parameters

Step	Sender	Note
0		**Unidirectional** quote stream to **buy** must be opened for a test instrument **<out of any instruments available in the testing environment>**
1	**Client LP**	The client adds a **sell quote** for a test instrument **<out of any futures available in the testing environment>** at a price **<within the limits>** with **<the Client's account>** used.
2	System	The system sends a message based on business logic.

3	Client	The client deletes their added **sell quote.**
4	System	The system sends a message based on business logic.

Test 2-6. LP to perform 'mass cancel' operation under quotes

Step	Sender	Note
0		**Unidirectional** quote stream to **buy** must be opened for a test instrument **<out of any instruments available in the testing environment>**
1	**Client LP**	The client adds 3 **sell quotes** for a test instrument **<out of any futures available in the testing environment>** at a price **<within the limits>** with **<the Client's account>** used.
2	System	The system sends a message based on business logic.
3	Client	The client deletes their added **sell quote.**
4	System	The system sends a message based on business logic.

Test 2-7. LP to confirm an indicative trade

Step	Sender	Note
0		**Unidirectional** quote stream to **buy** must be opened for a test instrument **<out of any instruments available in the testing environment>** with enabled Last Look step.
1	**Client LP**	The client adds a **sell quote** for a test instrument **<out of any futures available in the testing environment>** at a price **<within the limits>** with **<the Client's account>** used. "Last look" step must be set. QuoteType(537) = 0.
2	System	The system sends a message based on business logic.
3		Passive counterparty (Liquidity consumer) should confirm indicative trade.
4	System	The system sends a message based on business logic.
5	**Client LP**	Confirm command should be issued specifying QuoteReqID(131), ExecID(17), OrderID(31) fields by the client – liquidity provider.
6	System	The system sends a message based on business logic.

Appendix 5. Scenario for customer software connected to ASTS FIX services

ASTS FIX (MFIX Transactional)

The following is validated:

- Order entry/withdrawal; absence of errors at order entry caused by the transaction invalid generation;
- Proper use of tag 11 ClOrdID and associated <Parties group> (valid order entry from a client account with the pertinent client code);
- Proper use of Resend Request after the session restoring;
- The password change transaction (an opportunity to enter a new password in a specific tag of the Logon message) and the language change of the trading system messages (specifying the language in the special tag of the Logon message).

FX market OTC FIX (OTCT, OTCF, CPCL)

The following is validated:

- Order entry; absence of errors at order entry caused by the transaction invalid generation;
- Proper use of tag 11 ClOrdID and associated <Parties group> (valid order entry from a client account with the pertinent client code);
- Proper use of <Parties group> to explicit statement of trade counterparty on the CPCL board;
- Proper use of Resend Request after the session restoring;
- The password change transaction (an opportunity to enter a new password in a specific tag of the Logon message) and the language change of the trading system messages (specifying the language in the special tag of the Logon message).

FX market RFS platform FIX (RFSM)

The following is validated:

- Change language of the trading system messages with Logon ('A');
- Request available instruments list with message Security List Request ('X');
- Operation with subscription/unsubscribing for the information about acting RFS auctions with message RFQ Request ('AH');
- Auction initiation with message Quote Request ('R');
- Cancellation of the initiated auction with message Quote Response ('AJ');
- Rejection to participate in the auction with message Quote Request Reject ('AG');
- Send quote to the auction with message Quote ('S');
- Quote cancellation with message Quote Cancel ('Z');
- Agreement with a quote with message Quote Response ('AJ').

Appendix 6. Scenario for customer software connected to Standardized OTC derivatives FIX services (SPFI FIX)

Test 1-1. Open session

Client should authenticate at the test Standardized OTC Derivatives market server

Step	Sender	Note
1	**Client**	Send a Logon (A) message with user id and password provided by the Exchange
2	System	System will respond with Logon (A) message in case of successful authentication or Logout (5) in case of failure

Test 1-2. Password change

Client should change the password of their FIX login

Step	Sender	Note
1	**Client**	Send a Logon (A) message, using a login and password provided by the Exchange and specifying the new password in NewPassword (925) tag
2	System	System will respond with Logon (A) message in case of successful authentication or Logout (5) in case of failure. Password change status can be seen in SessionStatus (1409) tag

Test 1-3. Session closure

Client should successfully disconnect from the system and connect again (test 1-1)

Step	Sender	Note
1	**Client**	Send a Logout (5) message
2	System	System will respond with a Logout (5) message. Status 4 in SessionStatus (1409) tag – successful session termination

Test 2-1. Instrument list request

Client should send a request for obtaining a full list of standard instruments (order book tickers)

Step	Sender	Note
1	**Client**	Send a SecurityListRequest (x) message without specifying an instrument
2	System	System will respond with a SecurityList (y) message containing a list of traded tickers in Symbol (55) tags

Test 2-2. Instrument description request with a specific ticker

Client should send a request for obtaining an instrument description with a specified ticker from a list received in test 2-1

Step	Sender	Note
1	**Client**	Send a SecurityListRequest (x) message with specifying an instrument ticker in Symbol (55) tag
2	System	System will respond with a SecurityList (y) message with a description of specified ticker in FpML format in SecurityXML (1185) tag

Test 3-1. Market data request for a ticker

Client should send a request for receiving market data for an instrument with specified ticker from a list received in test 2-1

Step	Sender	Note
1	**Client**	Send a MarketDataRequest (V) message with specifying an instrument ticker in Symbol (55) tag
2	System	System will respond with MarketDataSnapshotFullRefresh (W) message with order prices and volume in an order book for requested ticker

Test 4-1. Order placement

Client sends an order into system with a ticker of a traded instrument

Step	Sender	Note
1	**Client**	Send a NewOrderSingle (D) message with specifying an instrument ticker
2	System	System will respond with a message or a set of messages, based on FIX protocol logic (ExecutionReport (8) with OrdStatus (39)=0 if the order is accepted, ExecutionReport (8) with OrdStatus (39)=2 if the order is filled or ExecutionReport (8) with OrdStatus (39)=8 if the order is rejected)

Test 4-2. Order cancelation

Client sends a cancelation transaction for cancelation of a previously placed order

Step	Sender	Note
1	**Client**	Send a OrderCancelRequest (F) message with an identifier of a previously placed order in OrderID (37) tag
2	System	System will respond with a message or a set of messages, based on FIX protocol logic (ExecutionReport (8) with OrdStatus (39)=4 if order is cancelled or OrderCancelReject (9) if cancellation was not successful)

Test 5-1. Request for trades accepted for clearing

Client sends a request for trades list

Step	Sender	Note
1	**Client**	Send a TradeCaptureReportRequest (AD) message with specifying a TradeRequestType (569)=1 parameter
2	System	System will respond with a message or a set of messages, based on API logic (TradeCaptureReportRequestAck, TradeCaptureReport)

Appendix 7. Scenario for customer software connected to SPECTRA TWIME services

The client must perform all the scenarios below and send (on the same day) the log files of the application to help@moex.com, strictly in format: Message name (tag1 = value1, tag2 = value2=,)

Example: NewOrderSingle (blockLength=46, templateId=6000, schemaId=19781, version=2, ClOrdID=102, ExpireDate=18446744073709551615L, Price=100000, SecurityID=347990, ClOrdLinkID=7895424, OrderQty=5, TimeInForce=0, Side=1, CheckLimit=1, Account='AAAA')

Test 1. Connection initialization and disconnection

Test 1-1. Session initialization

The client must get connected to the system, by entering its **TWIME** login obtained from the Exchange.

The client-side software must send **Establish** message and at least 1 **HeartBeat** message **(**Sequence with NextSeqNo= null). This is an automatic test without user actions required.

Test 1-2. Logging out

The user must log out correctly in 1 minute and log in again.

Test 2. Interaction in trading

Test 2-1. New futures order

Order with validity «Day». On a basic level, an order is accepted by the system, and then the following steps are performed:

Step	Sender	Note
1	Client	The client sends a **limit** order to **buy** a futures <**out of any futures available in the testing environment**> with **<the Client's account>** used. The order size is 5 contracts. **Order type is Day <0>**
2	System	The system sends a message based on a business logic

Test 2-2. New long-term order to buy futures

A GTD order is placed:

Step	Sender	Note

1	Client	The client sends a **limit** order to **buy** a **futures <out of any futures available in the testing environment>** at a price **<within the limits>** with **<the Client's account>** used. The order size is **6** contracts. The order is **GTD=<6>**. The expiration date is any different from the current one.
2	System	The system sends a message based on a business logic

Test 2-3. Working quote cancellation

The order placed in test 2-1 or 2-2 is cancelled by the Client for OrderID::

Step	Sender	Note
1	Client	The client initiates the cancellation of a working quote for OrderID received (selling **6** contracts)
2	System	The system sends a message based on business logic.

Test 2-4. New option order

Client sends an order to buy an option:

Step	Sender	Note
1	Client	The client sends a **limit** order to **buy** an option **<out of any options available in the testing environment>** The order size is **1** contract from **<the Client's account>.** Order type is **Day <0>**
2	System	The system sends a message based on business logic.

Test 2-5. Order replacement

Client sends an order to buy an option and replace the working order:

Step	Sender	Note
1	Client	The client enters a **limit** order to **buy** an option **<out of any options available in the testing environment>**. The order size is **10** contracts from **<the Client's account>. Order type is Day <0>**
2	System	The system sends a message based on business logic
3	Client	The client changes the price of the previously entered order for OrderID. The newly entered price must differ from the previous one.
4	System	The system sends a message based on business logic

Test 2-6. Placement of a multi-leg instrument order *:

*applied only to clients wishing to trade multi-leg instruments, for the other clients this paragraph is not mandatory

Step	Sender	Note
1	Client	The client enters a **limit** order to **buy (NewOrderMultileg)** a multi-leg instrument **<out of any available in the testing environment>**. The order size is **10** contracts. The **"client account"** is used.
2	System	The system sends a message based on a business logic

Test 2-7. Order mass cancellation

Cancelling all available orders accepted during the testing.
Orders in multi-leg instruments cannot be cancelled with this message.

Step	Sender	Note
1	Client	The client requests mass order cancellation by indicating **<Account>** and relevant Side=<89> and SecurityType=<0>
2	System	The system sends a message based on business logic. Only **futures** orders are cancelled
3	Client	The client requests mass order cancellation by indicating **<Account>** and relevant Side=<89> and SecurityType=<1>
4	System	The system sends a message based on business logic. Only **option** orders are cancelled

Test 2-8. Message retransmission request

Request for resending the last 5 messages:

Step	Sender	Note
1	Client	The client sends a **RetransmitRequest** message with a serial number =N-5, where N – number of the last message received.
2	System	The system sends a Retransmission message and re-requested messages.

Test 3. Missed messages mass recovery

Test 3-1. Session initialization

The client must get connected to the Recovery service, by entering its **TWIME** login obtained from the Exchange.

The client-side software must send **Establish** and **HeartBeat** messages **(**Sequence with NextSeqNo= null). This is an automatic test without user actions required.

Test 3-2. Retransmission request of a large number of messages

Step	Sender	Note
1	Client	The client sends a **RetransmitRequest** message with a serial number =1, and the quantity of the messages is not more than N-1. N – the number of the next system message
2	System	The system sends a Retransmission message and re-requested messages.

* the retransmission limits may be specified from the tech support.

Test 4. Request to NCC for verification an initial margin within a BF

Test 4-1 Request to NCC for verification an initial margin within a BF

The request is designed to cancel orders by a clearing participant for positions insecurity elimination. With this request in case of negative free limit within a broker firm (FreeMoney < 0) all the active orders of the client within the broker firm are cancelled.

Step	Sender	Note
1	Client	The client sends a OrderMassCancelByBFLimitRequest, with pointing **<Account>.**
2	System	The system sends a message based on business logic.

*the test is not mandatory. Request for Security Sufficiency verification can be sent only by CF or BF logins.

Optional tests

Test 5. Sending trading commands with bit 0x2 (NCC inquiry) in field ClientFlags

5.1 NewOrderSingle. Ordinary instrument order adding

Step	Sender	Note
1	Client	The client sends a **limit** order to **buy** a futures <**out of any futures available in the testing environment**>. The order size is 5 contracts from **<the Client's account>**. ClientFlags - 0x2
2	System	The system sends a message based on a business logic

5.2 NewOrderMultileg. Addition of a multi-leg instrument order

Step	Sender	Note
1	Client	The client enters a **limit** order to **buy (NewOrderMultileg)** a multi-leg instrument **<out of any available in the testing environment>**. The order size is **10** contracts. The **"client account"** is used. ClientFlags - 0x2
2	System	The system sends a message based on a business logic

5.3 OrderCancelRequest. Working order cancellation request

Step	Sender	Note
1	Client	The client initiates the cancellation of a working order indicating **<Account>.** ClientFlags - 0x2
2	System	The system sends a message based on business logic.

5.4 OrderReplaceRequest. Working order price/size change request

Step	Sender	Note
1	Client	The client initiates the change of a working order indicating **<Account>, Mode = <1>.** ClientFlags - 0x2
2	System	The system sends a message based on business logic.

Appendix 8. Scenario for customer software connected to TWIME ASTS services

For the necessary certification of TWIME ASTS connection it is required to demonstrate following scenarios. After all described test cases are passed, on the same trading day, it is required to send result log files of client's software to help@moex.com.

Please note, that actions on orders must be performed using a trading and clearing account of a client.

Test 1. Session establishment and termination scenario

Test 1-1. Session establishment

The client (initiator) sends a Logon message with SenderCompID (the identifier provided by MOEX) to establish connection with TWIME ASTS server (acceptor). The acceptor will authenticate the identity of the initiator by examining the Logon message.

The **Establish** (msg_id:6) message must be received from the software.

Test 1-2. Session termination

Client closes the session sending the **Terminate** (msg_id:4) message and then reestablish connection to the service.

Test 1-3. RetransmitRequest (msg_id:2) resend messages mechanism

Client establishes session and then swaps a few messages with the service. Client's software sends **RetransmitRequest** (msg_id:2) message with random number of messages specified.

Test 2. Password change

This test case designed to test software functions to operate **ChangePassword** (msg_id:9) message.

Client establishes the session and Client's software submits **ChangePassword** (msg_id:9) message with the new password specified. To pass the test client should terminate session normally and then establish the session using the new password.

Step	Sender	Note
1	Client	Client's software submits **ChangePassword** (msg_id:9) message with the new password specified.
2	System	The system sends **ChangePasswordAck** (msg_id:10).

Test 3. Trading operations

Following test cases are designed to test software's options for trading participate.

Test 3-1. Submit New order

Client submits new order using **NewOrderSingle** (msg_id:13) message. Trading board value and the security and other order parameters remain at the discretion of the client.

Step	Sender	Note
1	Client	Client's software submits **NewOrderSingle** (msg_id:13) message. Order parameters are not regulated.
2	System	The system sends **ExecutionReport** (msg_id:17) response message with ExecTypeNum (tag_id:150) field contains 0 value.

Test 3-2. Cancel order

To perform cancel scenario client submits limit order that not to be immediately executed. Then, the order should be canceled using **OrderCancelRequest** (msg_id:14) message.

Expected result:

Step	Sender	Note
1	Client	Client's software submits **OrderCancelRequest** (msg_id:14) message to cancel working limit order.
2	System	The system sends **ExecutionReport** (msg_id:17) response message with ExecTypeNum (tag_id:150) equal to value of 4 or 6.

Test 3-3. Amend order

To perform amend scenario client submits limit order that not to be immediately executed. To amend the order, client submits **OrderReplaceRequest** (msg_id:16) message. It is required to wait to message **ExecutionReport** (msg_id:17) having ExecTypeNum (tag_id:150) = 5.

Client should successfully cancel the new order with **OrderCancelRequest** (msg_id:14) message.

Following interaction scenario is expected:

Step	Sender	Note

1	Client	Client's software submits **NewOrderSingle** (msg_id:13) message to enter a **limit** order to **buy** a security <**out of any securities available in the testing environment**>. The quantity is 10 from **<the Client's account>**
2	System	The system sends **ExecutionReport** (msg_id:17) response message with ExecTypeNum (tag_id:150) field contains value of 0.
3	Client	The client changes the price of the previously entered order. The newly entered price must differ from the previous one. Client's software submits **OrderReplaceRequest** (msg_id:16) message.
4	System	The system sends **ExecutionReport** (msg_id:17) response message with ExecTypeNum (tag_id:150) field contains value of 5.
5	Client	Client's software submits **OrderCancelRequest** (msg_id:14) message.

Appendix 9. Scenario for customer software connected to MOEX WEB-API services

WEB-API Clearing terminal

Test 1-1. Receiving MOEX Passport Token

The client logs in at passport-test.moex.com/authenticate and get MOEX Passport Token in the response.

Step	Sender	Note
1	Client	The client sends a GET-request using Basic authentication with the **passport** user's credentials
2	System	The system responds with a cookie with MicexPassportCert, containing the MOEX Passport Token.

Test 1-2. Receiving API access token

The client sends a POST-request to play-api.moex.com with the **detached digital signature.**

Step	Sender	Note
1	Client	The client sends a POST-request indicating the **signature** - detached digital signature generated from MOEX Passport Token with **xpki1utl** (for GOST (state standard) certificates) or with **rpki** (for RSA certificates)
2	System	The system responds with the **access_token** – access token which must be sent during each API using

Test 2-1. Receiving markets list

The client sends a request to get the markets list

Step	Sender	Note
1	Client	The client sends a GET-request/markets
2	System	The system sends a list of the available markets

Test 2-2. Receiving market assets

The client sends a request to get assets available in the chosen market sector

Step	Sender	Note
1	Client	The client sends a GET-request/markets/ {Market}/assets with the market indication

2	System	The system responds with a message with available assets for the specified market

Test 2-3. Receiving settlement accounts

Step	Sender	Note
1	Client	The client sends a GET-request/scodes?Market={Market } with the market indication
2	System	The system responds with a message with available settlement accounts for the specified market

Test 2-4. Initial margin transfer request

The client sends a request transfer the initial margin from the settlement account to another settlement account

Step	Sender	Note
1	Client	The client sends a POST-request/asset-transfers with indicating the required data
2	System	The system sends a message about request acceptance (202 Accepted)

Test 2-5. Profile transfer request

The client sends a request to transfer profile from the settlement account to another settlement account

Step	Sender	Note
1	Client	The client sends a POST-request/profile-transfers with indicating the required data
2	System	The system sends a message about request acceptance (202 Accepted)

Test 2-6. Receiving the list of requests sent with the clearing terminal

The client sends a request to get the full requests list, sent through the clearing terminal

Step	Sender	Note
1	Client	The client sends a GET-request/reqs
2	System	The system sends a message with the list of requests, sent earlier with the clearing terminal

The following test are passed if the client is planning to work with the derivatives market

Test 2-7. Receiving broker accounts list

The client sends a request to get the list of his broker accounts

Step	Sender	Note
1	Client	The client sends a GET-request/brokers
2	System	The system sends a message with the list of available broker accounts

Test 2-8. Receiving broker accounts list

The client sends a request to get the list of his client accounts

Step	Sender	Note
1	Client	The client sends a GET-request/clr-clients
2	System	The system sends a message with the list of available client accounts

WEB-SPFI (standardized OTC derivatives market)

Test 1-1. Receiving MOEX Passport Token

The client must sign in to passport-test.moex.com/authenticate and get MOEX Passport Token as a response

Step	Sender	Note
1	Client	The client sends a GET-request, using Basic authentication with the **passport** user credentials
2	System	The system sends a cookie with MicexPassportCert, containing MOEX Passport Token.

Test 1-2. Receiving API access token

The client sends a POST-request to play-api.moex.com with a separated digital signature

Step	Sender	Note
1	Client	The client sends a POST-request, using indicating signature – separated digital signature formed of MOEX Passport Token with xpki1utl (for state standard certificates (GOST) or rpki (RSA certificates)

2	System	The system sends an access_token which must be sent in API request.

Test 2-1. Participants list request

The client sends a request to get the full list of participants

Step	Sender	Note
1	Client	The client sends a PartyDetailsListRequest with pointing RequestedPartyRole=1 parameter
2	System	The system sends a message PartyDetailsListReport with the requested data

Test 2-2. Clients settlement accounts request

The client sends a request to get the list of his settlement accounts

Step	Sender	Note
1	Client	The client sends a PartyDetailsListRequest with pointing RequestedPartyRole=24 parameter
2	System	The system sends a message PartyDetailsListReport with the requested data

Test 2-3. Clearing accepted deals request

The client sends a request to get the deals list

Step	Sender	Note
1	Client	The client sends a TradeCaptureReportRequest with pointing **TradeRequestType=1** parameter
2	System	The system sends a message/messages with API logic (TradeCaptureReportRequestAck, TradeCaptureReport)

Test 2-4. Request of the client's deals list waiting for the partner approve

The client sends a request to get the deals list waiting for the partner approve

Step	Sender	Note
1	Client	The client sends a TradeCaptureReportRequest with pointing **TradeRequestType=2 and PartyRole=1** parameter
2	System	The system sends a message/messages with API logic (TradeCaptureReportRequestAck, TradeCaptureReport)

Test 2-5. Request of the deals list issued to the client and waiting for the partner approve

The client sends a request to get the deals list issued to the client and waiting for the partner approve

Step	Sender	Note
1	Client	The client sends a TradeCaptureReportRequest with pointing **TradeRequestType=2 and PartyRole=17** parameter
2	System	The system sends a message/messages with API logic (TradeCaptureReportRequestAck, TradeCaptureReport)