# National Commodity and Derivatives Exchange

## Post Trade Gateway

## FIX 5.0 SP2 Specification

| | |
|---|---|
| **Version** | 4.01 |
| **Release Date** | 06th Oct 2017 |

# 1 DOCUMENT CONTROL

## 1.1 Table of Contents

## 1.2 Revision History

| Date | Ver | Sections | Description |
|---|---|---|---|
| 04 Mar 2015 | 1.00 | | Initial Draft |
| 15 Nov 2015 | 2.00 | | Added length |
| 31 May 2016 | 3.00 | 1.5 | New abbreviations added |
| | | 3.1.3.5, 7.5.2, 7.8.2 | Multi-leg Order related details, addition of new tags and fixing of few tags lengths. |
| 20 June 2017 | 3.01 | 7.5.2 | Updated the TCR message with Options related fields.<br>Following Tags added.<br>201 : PutOrCall<br>202 : StrikePrice<br>1194 : ExerciseStyle<br>32022 : LastOptPx<br>1188 : Volatility |
| 06th Oct 2017 | 4.01 | 4.1.2 | Introduced the section on NNF ID. Updated messages where the Order Entry Operator ID is used. |

## 1.3 References

FIXT 1.1 Specification

FIX 5.0 (Service Pack 2) Specification

## 1.4 Exclusions

The given functionality is currently not supported by NCDEX

| Sr. NO | Functionality | |
|---|---|---|
| 1 | Order Types | Market |
| | | Market If Touched |
| | | Trailing Stop |
| | | Trailing Stop Limit |
| | | Reserve |
| | | Minimum Fill |

| | | |
|---|---|---|
| | | |
| | | Named |
| | | At the Open (OPG) |
| | | Fill or Kill(FOK) |
| | | At the Close (ATC) |
| | | Good For Auction(GFA) |
| | | Good Till Time (GTT) |
| | | Closing Price Cross |
| | | |
| 2 | Give-up from CTCL Front end | |
| | | |
| 3<br>4<br>5 | Negotiated Trades (EFP)<br>Spread Instrument<br>Strategies | Stop<br>Stop_Limit<br>BF<br>CD |

## 1.5 Definitions, Acronyms and Abbreviations

**Client** A participant or service bureau connected to the post trade gateway.

**CP Code** A Custodian Participant Code assigned to a Client who wishes to clear through a member other than his trading member's clearing member

**FIX** Version 5.0 (Service Pack 2) of the Financial Information Exchange Protocol.

**FIX Connection** A bi-directional stream of ordered messages between the client and server within a particular login. A FIX connection ends when the client logs out or if the TCP/IP connection is terminated.

**FIX Session** A bi-directional stream of ordered messages between the client and server within a continuous sequence number series. A single FIX session can exist across multiple FIX connections.

**FIXT** Version 1.1 of the Financial Information Exchange Session Protocol.

**NCDEX** National Commodities and Derivatives Exchange

**NNF ID** This should be a fifteen digit ID with the following breakup.

a. First 3 digits will be the identity code for distinguishing orders i.e. whether the orderis program generated (Automated Trading) or manually (CTCL Workstations) entered. Program generated (Automated trading) or manually entered should send the ATS/CTCLvendor code respectively.

e g. If an order is generated through CTCL workstation, they should enter the CTCL vendor code and if an order is generated thought ATS (also a CTCL workstation), they should enter the ATS vendor code.
b. Next 6 digits will be unique approved person code. (first 3 digits will be branch code and last 3 digits will be CTCL terminal Id)

c. Next 6 digits will be the Pin code.

**Order** Executable interest which for an order.

**Order Book** Each instrument is traded across multiple separate and distinct order books (e.g. regular, etc.). Each Trade Capture Report includes an indication of the instrument and order book to which it relates.

**Query-Based Client** A client configured not to receive real-time updates of eligible trades. Such a client is expected to request for the details of trades as and when they are needed.

**Real-Time Client** A client configured to receive eligible trades immediately after they are executed.

**Server** The post trade gateway of NCDEX.

**Service Bureau** A third party (e.g. technology vendor) providing technical access and connectivity to the server for a participant.

**Trade Report** A Trade Capture Report message submitted by a client to cancel or amend an on-book trade.

**Trading Gateway** The interface of NCDEX that allows participants and service bureaus to send and manage their trading interest.

**Trading Mnemonic** Each trade report must include the trading mnemonic it is submitted under. Trading privileges are assigned at the level of trading mnemonics.

# 2 OVERVIEW

NCDEX offers an interface to its post trade system that permits participants to perform the activities outlined below.

(i) Receive real-time updates on executed trades

(ii) Receive information on executed trades via a query-based service. This service is available for participants that do not require a real-time trade feed and to facilitate a recovery after a failure.

(iii) Request the correction of an on-book trade

The interface is a point-to-point service based on the technology and industry standards TCP/IP, FIXT and FIX. The session and application event models and messages are based on versions1.1 and 5.0 (Service Pack 2) of the FIXT and FIX protocols respectively. Please refer to Section 7.2 for the instances where the server varies from the FIX protocol.

The encryption of messages between the client and server is not supported.

The exchange requires that the vendor/exchange’s member undergo a conformance test upon completing development of the interface. The vendor/exchange’s member must contact the exchange to schedule an appropriate period for testing.

The vendor/exchange’s member may contact the CTCL division of the exchange to seek clarification

CTCL Team (Technology Department)

National Commodity & Derivatives Exchange Limited,
Akruti Corporate Park, 1st Floor, Near G.E.Garden, L.B.S. Marg,
Kanjurmarg (West), Mumbai - 400 078
Phone (022) 66406717/66406609-12
Fax (022) 66473201/66406899
Email: CTCL@ncdex.com

# 3 SERVICE DESCRIPTION

## 3.1 Trade Feed

Clients may receive real-time information on the trades executed on NCDEX along with notifications of any trade cancellation or correction. A query-based service is available for participant applications that do not require a real-time trade feed.

The details of trades executed on previous trading days are not available via this service.

### 3.1.1 Connection Configuration

A participant connection will be configured by NCDEX to be eligible to receive all of its trades. A clearing firm connection will be configured to be eligible to receive the trades it will clear. If required, a firm or clearing firm connection could be configured to only be eligible to receive trades for selected trading mnemonics and instruments. A clearing firm can be configured to receive trades of specific clients for whom the clearing firm handles the give –ups. Each connection will be configured as either a real-time client or a query-based client.

For the purpose of redundancy, the service supports the configuration of multiple post trade connections for the receipt of the same information on the activity of the selected firms/mnemonics.

#### 3.1.1.1 Real-Time Connections

A real-time client will receive the details of each eligible trade immediately after it is executed. Please refer to Sections 3.1.2, 6.4 and 6.5 for a description of how the trades executed during the time a real-time client is disconnected from the server may be recovered.

#### 3.1.1.2 Query-Based Service

A query-based client will not receive any real-time notifications of its eligible trades. Such clients are expected to request the server for the details of trades as and when they are needed as outlined in Section 3.1.2.

### 3.1.2 Trade Capture Report Requests

A client may use the Trade Capture Report Request message to request the details of all eligible trades or those that meet certain criteria. The server will respond with a Trade Capture Report Request Ack to indicate, via the TradeRequestStatus (750) and TradeRequestResult (749) fields, whether the request is successful or not.

The total number of Trade Capture Report Requests that a client may send is limited to *<25>* each day[1]. However, a client may request NCDEX to reset its request count. This feature is intended to help manage an emergency situation and should not be relied upon as a normal practice.

If a request is accepted, the number of Trade Capture Reports that will be sent in response to the request will be indicated in the TotNumTradeReports (748) field of the Trade Capture Report Request Ack. The server will transmit the requested Trade Capture Reports immediately after the Trade Capture Report Request Ack. Each Trade Capture Report will include the TradeRequestID (568) of the request it is sent in response to. The last Trade Capture Report will include a LastRptRequested (912) of Last Message (Y).

If a request is rejected, the reason will be specified in the field TradeRequestResult (749) of the Trade Capture Report Request Ack.

[1]

The Trade Capture Report Request message can only be used to request a snapshot of the current eligible trades. It cannot be used to subscribe to Trade Capture Reports.

#### 3.1.2.1 Request for All Trades

The Trade Capture Report Request should include a TradeRequestType (569) of All Trades (0) if the client wishes to request the details of all eligible trades.

#### 3.1.2.2 Request for Selected Trades

The Trade Capture Report Request should include a TradeRequestType (569) of Trades Matching Specified Criteria (1) if the client wishes to request the details of eligible trades for a selected set of instruments or for a specified party, trade type or order.

The Symbol (55) or SecurityAltID (455) field of the Trade Capture Report Request may be used if the request relates to a single instrument. While the Symbol (55) will relate to the instrument's trading identifier, the SecurityAltID (455) will refer to its identification number (i.e. ISIN).

The UnderlyingSymbol (311) field of the message may also be used in the case of derivative instrument. Alternately, the CFICode (461), SecurityType (167), SecuritySubType (762) and ProductComplex (1227) fields of the Trade Capture Report Request may be used if the request relates to a particular group of instruments. Please refer to Section 8 for a list of valid values for these fields.

The OrderBook (30001) field may be used if the request relates to a particular order book (e.g. regular, etc.).

The MatchType (574) field of the Trade Capture Report Request may be used if the request relates to particular types of trades (e.g. those executed during continuous trading).

The client may also use the PartyID block of the message to indicate the parties (i.e. executing firm, clearing firm, entering unit, etc.), if any, the request relates to. A Trade Capture Report Request may include multiple PartyIDs.

In addition, the Side (54) field of the Trade Capture Report Request may be used if the client only wishes trades for a particular side (i.e. buy or sell).

The client may also use the ClOrdID (11) or OrderID (37) field if the request is for trades related to a specific order.

The ExecType (150) field of the message may be used if the request is limited to cancelled trades, corrected trades or trades that have not been cancelled or corrected.

If a particular Trade Capture Report Request contains multiple criteria (e.g. symbol and side), the server will treat it as a request for trades that match all of the specified criteria. If none of the trades the client is eligible to receive match the specified criteria, the server will reject the request with a TradeRequestResult (749) of Cannot Match Selection Criteria (100).

If a user submits TradeRequestType (569) outside the above specified criteria but within the FIX supported range (i.e. 2,3,4) it will be rejected with a Trade Capture Report Request Ack with TradeRequestResult (749) set to 8 (TradeRequestType not supported).

If a user submits TradeRequestType (569) outside the FIX supported range (i.e. 0,1,2,3,4) it will be rejected with a session reject where SessionRejectReason (373) is 5 (Value out of range for this tag).

### 3.1.3 Trade Information

The Trade Capture Report message is utilised by the server to transmit the details of each confirmed trade. A separate Trade Capture Report will be sent for each side of a trade. These messages will include a TradeHandlingInstr (1123) of Trade Confirmation (0), a TradeReportType (856) of Submit (0), an ExecType (150) of Trade (F) and a MatchStatus (573) of Matched (0). The TradeReportTransType (487) will be New (0).

Each message will contain both basic and value added information on the trade (e.g. price, quantity, consideration, settlement date, etc.), the instrument (e.g. ISIN, underlying, expiration date, etc.) and the parties (e.g. executing firm, clearing firm, etc.). It will also contain information related to the computation of execution fees (e.g. maker or taker, trade type, order book, etc.).

### 3.1.3.1 Party Identifiers

| ID | Description | PartyRole (452) |
|---|---|---|
| Executing Firm | The trading firm the executed order was submitted under. | 1 |
| Entering Unit | The trader group of the firm the executed order was submitted under. | 58 |
| Trading Mnemonic | The mnemonic of the firm the executed order was submitted under. | 53 |
| Clearing Firm | The firm through which the trade will clear. | 4 |
| Contra Firm | The firm on the contra-side of an on book trade. Populated only if the trade is against a user from the same firm as Executing Firm.<br>Identifier of the counterparty trading firm in the case of a EFP trade. | 17 |
| Contra Trading Mnemonic | The mnemonic under which the order for the contra-side of the trade was entered (i.e. Trading Mnemonic of the contra-side of an on book trade). Populated only if the trade is against a user from the same firm as Executing Firm. | 37 |
| Order Entry Operator ID | The NNF ID specified on a client initiated message. A reply to a client initiated message will contain the original NNF ID | 44 |

The Account (1) field will be used to identify the investor account on whose behalf the trade was executed. The account type (e.g. client or house) is specified in the AccountType (581) field while the AllocAccount (79) field is used to specify the CP Code.

### 3.1.3.2 Trade, Execution and Order Identifiers

| ID | Description |
|---|---|
| Trade Report ID | The TradeReportID (571) of each Trade Capture Report is unique across trading days. The Trade Capture Reports published to report the two sides of a trade will contain different TradeReportIDs. A Trade Capture Report published to notify a client of a trade cancellation or correction includes the TradeReportID of the message that was published to report the trade in the TradeReportRefID (572) field. |
| Trade ID | The Trade Capture Reports published to report the two sides of a particular trade will contain the same TradeID (1003). Trade IDs are unique across trading days. A Trade Capture Report published to notify a client of a trade cancellation or correction includes the TradeID (1003) of the trade. |
| Secondary Trade ID | The Trade Capture Reports published to report the two sides of a particular trade will contain the same SecondaryTradeID (1040). Secondary Trade IDs are unique only for the current trading day. A Trade Capture Report published to notify a client of a trade cancellation or correction includes the SecondaryTradeID (1040) of the trade. |
| Execution ID | A Trade Capture Report will contain the Execution ID of the Execution Report message sent by the Trading Gateway to report an execution or the cancellation or correction of an execution to the firm that submitted the order. This Execution ID will be specified in the SideExecID (1427) field of the Trade Capture Report.<br>The Execution Reports published to report the two sides of an execution will contain different Execution IDs which are unique across trading days. |
| Order IDs | The matching system‘s order identification number for the executed order will be included in the OrderID (37) field of the Trade Capture Report.<br>Order IDs are unique across trading days. In terms of the FIX protocol, unlike ClOrdID (11) which requires a chaining through Cancel/Replace Requests and Cancel Requests, the OrderID (37) of an order will remain constant throughout its life. |
| Client Order IDs | In the case of orders, the ClOrdID (11) included in the Trade Capture Report will be that specified when the order was submitted or last updated. An order's ClOrdID (11) will be updated each time an Order Cancel/Replace Request or an Order Cancel Request is accepted. |
| Trade Link ID | The Trade Capture Reports published for a single transaction (e.g. the execution of an incoming order against multiple passive orders, etc.) will contain the same TradeLinkID (820). Trade Link IDs are unique across trading days. A Trade Capture Report published to notify a client of a trade cancellation or correction includes the TradeLinkID (820) of the trade. |

### 3.1.3.3 Trade Type

An indication of whether the trade was executed on the order book will be specified in the TrdType (828) field of a Trade Capture Report.

In the case of a Give up trade, field TrdSubType (829) will contain the value 61 (Give Up Trade)

#### 3.1.3.4 Information for Billing

Each Trade Capture Report will specify the methodology under which a trade was executed (e.g. continuous trading,.), the type of interest (e.g. order,), the order book (e.g. regular) and whether an order was a maker or taker of liquidity via the MatchType (574), OrderCategory (1115), OrderBook (30001) and SideLiquidityInd (1444) fields respectively.

Trade Capture Reports relating to EFP trades will not include a SideLiquidityInd (1444).

#### 3.1.3.5 Strategies

The LastPx (31) of trades for multi-legged instruments may contain negative prices.

A trade for a strategy will involve a Trade Capture Report for the multi-legged instrument as well as separate Trade Capture Reports for each of the associated leg instruments. The field MultiLegReportingType (442) should be used to determine whether a particular message relates to the multi-legged instrument or a leg instrument. For every strategy trade, the Trade Capture Report for the multi-legged instrument as well as the leg instruments should have the same SideLiquidityInd (1444) and MatchType (574) between them.

The field MultiLegReportingType (442) will communicate a ‘Multi-leg order on leg instruments’ (9) on the Trade Capture Report message, if the trade is being communicated due to the execution of the legs of a multi-leg order.

While the ClOrdID (11) of a Trade Capture Report for a leg trade will be the same as the ClOrdID (11) of the executed order for the multi-legged instrument, the OrderID (37) will not.

### 3.1.4 Trade Cancellations and Corrections

NCDEX may cancel or correct a trade. The server will transmit Trade Capture Reports to the relevant clients to notify them of a trade cancellation or correction. The affected trade will be identified in the TradeID (1003) and TradeReportRefID (572) fields. The TradeHandlingInstr (1123) of the message will be Trade Confirmation (0).

In the case of a trade cancellation, ExecType (150) will be Trade Cancel (H) and MatchStatus (573) will be Unmatched (1). The TradeReportType (856) will be Trade Break (7) since the trade is cancelled by market operations. The TradeReportTransType (487) will be Cancel (1).

In the case of a trade correction, TradeReportTransType (487) will be Replace (2), ExecType (150) will be Trade Correct (G) and TradeReportType (856) will be  Addendum (4).

## 3.2 Trade Cancellation

The market operation of NCDEX can cancel trades. If a market operations user cancels a trade, such a cancellation will be notified to both the parties of the trade. The server will transmit a Trade Capture Report to each of the two counterparties to confirm the cancellation

Each message will include an ExecType (150) of Trade Cancel (H) and a TradeReportType (856) of Trade Report Cancel (6). The trade being cancelled will be identified via the TradeID (1003) and the TradeReportRefID (572) fields. The message will include a TradeHandlingInstr (1123) of TradeConfirmation (0), a TradeReportTransType (487) of Cancel (1) and a MatchStatus (573) of Unmatched (1).

## 3.3 Trade Amendment

This section outlines the event flow for the amendment of trades by participants. Please refer to Section 10.1 for related process flow diagrams.

NCDEX currently only permits participants to append a Client ID (Account (1)), AccountType (581) and CP Code (AllocAccount (79)) of an on-book trade. A participant should submit its amendment request via a Trade Capture Report. The server will use a Trade Capture Report Ack to acknowledge or reject the request. A Trade Capture Report will be transmitted by the server if the amendment request is successful.

Further, the trade amendment is required to submitted during a defined starting and ending time allocated for submitting amendment requests.

### 3.3.1 Submitting an Amendment Request

A client should submit the amendment request via a Trade Capture Report. The message should include the identifier assigned to the trade by NCDEX in the TradeID (1003) field. It should also include the side of the trade the client represents. The message should include a TradeReportType (856) of Addendum (4), a TradeReportTransType (487) of New (0), the identity of the instrument and the identity of the Client ID, Account or CP Code. Clients can optionally specify a TradeReportID (571) in the request.

The new Client ID should be mentioned as Account (1) with AccountType (581) as Client (1) or House (3).

### 3.3.2 Acknowledgement of the Amendment Request

An amendment request will be acknowledged by the server via a Trade Capture Report Ack. The request being acknowledged will be identified via the TradeID (1003) and Side (54) fields. The TrdRptStatus (939) will indicate whether the request is Accepted (0) or Rejected (1). If a request is rejected, the reason for the rejection will be specified in the TradeReportRejectReason (751) field. The MatchStatus (573) will be Matched (0) (unless an invalid TradeID (1003) was specified in the amendment request). The TransactTime (60) will contain the time the message was generated by the server.

If the client specified a TradeReportID (571) on the amendment request, the acknowledgement will also contain the same TradeReportID (571).

The field ‘AllocAccount’ (79) within the NoSides (552) block will be communicate the CP Code.

The new Client ID will be given as Account (1) with AccountType (581) as Client (1) or House (3) as specified in the amendment request.

For a give-up trade executed for a client account, the new Client ID will be mentioned as Account (1) and its corresponding CP Code in AllocAcocunt (79) with AccountType (581) as Client (1).

For a give-up trade executed for a house account, the new CP Code will be mentioned in the AllocAccount (79) field with AccountType (581) as House (3).

A Client ID amendment request rejection will be communicated with TrdRptStatus (939) as Rejected (1) and MatchStatus (573) as Matched (0). Further, it will also contain TradeReportRejectReason (751) with the corresponding rejection code and the Text (58) with the respective rejection reason.

### 3.3.3 Confirmation of the Amendment

The server will transmit a Trade Capture Report to confirm the trade amendment.

The message will include an ExecType (150) of Trade Correct (G) and a TradeReportType (856) of Addendum (4). The trade being amended will be identified via the TradeID (1003) and TradeReportRefID (572) fields. The message will include a TradeHandlingInstr (1123) of Trade Confirmation (0), a TradeReportTransType (487) of Replace (2) and a MatchStatus (573) of Matched (0). It will also include the updated investor account. TradeReportID(571) of the confirmation Trade Capture Report will not refer to the TradeReportID(571) that the client submitted.

Post amendment, the Trade Capture Report will indicate the Client ID prior to the amendment in PreAccount (22005) while the CP Codes prior to the amendment will be indicated in PreAllocAccount (22004)

## 3.4 Timestamps and Dates

The timestamps SendingTime (52), OrigSendingTime (122) and TransactTime (60) should be in UTC and in the YYYYMMDD-HH:MM:SS.sss format.

All dates (i.e. SettlDate (64), MaturityDate (541) and IssueDate (225)) should be in the YYYYMMDD format and specified in the local date for the server (i.e. not in UTC)).

## 3.5 Repeating Groups (Components/Component Block)

If a repeating group is used in a message, the NoXXX field (for example NoPartyIDs field in the trading party repeating group) should be specified first before the repeating group starts. This is applicable for both the messages generated by the client and the server.

The messages generated by the server will have the fields within a repeating group in order.

The messages generated by a client should have the first field in a repeating group in order. If the first field in a repeating group is in order, a message generated by a client will be accepted; else the message will be rejected.

If the same FIX tag is repeated with different values in the client generated message outside of a repeating group, the server takes the value in the last tag. The server will not reject such messages.

However, if a client initiated Logon message contains repeated tags; the server may not acknowledge the login request and will not send any reply. If client initiated other administrative messages or application messages contain repeated tags outside component blocks, such requests will be rejected by the server.

# 4 CONNECTIVITY

## 4.1 CompIDs

The CompID of each client must be registered with *NCDEX* before FIX communications can begin. A single client may have multiple connections to the server (i.e. multiple FIX sessions, each with its own CompID).

The CompID of the server will be *PTGW*. The messages sent to the server should contain the CompID assigned to the client in the field SenderCompID (49) and PTGW***Error! Reference source not found.***in the field TargetCompID (56). The messages sent from the server to the client will contain ***Error! Reference source not found.*** in the field SenderCompID (49) and the CompID assigned to the client in the field TargetCompID (56).

### 4.1.1 Passwords[2]

Each new CompID will be assigned a password on registration. Clients are encouraged to change the password to one of their choosing via the Logon message. The acceptance of a login request indicates that the new password has been accepted. The new password will, if accepted, be effective for subsequent logins.

In terms of the password policy of NCDEX, the password of each CompID should be changed at least every *<30>* days. If not, the password will expire and the client will be unable to login to the server. In such a case, the client should contact NCDEX to have its password reset. The SessionStatus (1409) of the server's Logon message will be Password Due to Expire (2) for the last *<5>* days of a password's validity period.

### 4.1.2 NNF ID

Clients connecting to NCDEX trading systems via FIX will have connections from either a NEAT front end system or any other front end.

When submitting a message, irrespective of whether the client is connecting via a Non-NEAT front end or not, the Non-NEAT Frontend ID has to be communicated to the server (mandatory). Hence, the PartyRole (452) of the following messages will contain the role Order Entry Operator ID (44). NNF ID should only contain numeric values (integers) and the maximum length should be 15 characters.

(a) Trade Capture Report Request (35=AD)
(b) Trade Capture Report Request (35=AE)

If the NNF ID is not submitted via Order Entry Operator ID (44) of Party Role (452) field or is not in the correct format as stipulated above, in any of the client initiated messages (i.e. Trade Capture Report (AE) and Trade Capture Report Request (AD)), such requests will be rejected with a Trade Capture Report Ack (AR) message consisting of a reject code and a reason as follows.

| Reject Code | Reject Reason |
|---|---|
| 166836 | Order Entry Operator ID is not submitted or format invalid |

[2] Delete this section if password validation is disabled.

## 4.2 Production IP Addresses and Ports

The IP address of each client must be registered with NCDEX before FIX communications can begin.

## 4.3 Failover and Recovery

The server has been designed with fault tolerance and disaster recovery technology that ensures that trading may continue in the unlikely event of a process or site outage.

If the client is unexpectedly disconnected from the server, it should attempt to re-connect to primary site within a few seconds. The client should only attempt to connect to the secondary IP address and port if so requested by NCDEX.

If there were Trade Capture Reports sent by the client which could not be processed due to a failure and if the client did not receive any responses for them before getting disconnected from the server, when the client logs back in, the system will transmit a Business Message Reject for each such Trade Capture Report to notify the client that the Trade Capture Reports were not processed by the system. The Business Message Reject message will not include RefTagID(371). It will include RefSeqNum(45), RefMsgType(372), BusinessRejectReason(380) as 4 and Text(58) as "Application not Available". The client is expected to retransmit required Trade Capture Reports in such situations. In certain (not all) scenarios, the Business Message Reject may include the PossResend (97) flag

# 5 FIX CONNECTIONS AND SESSIONS

## 5.1 Establishing a FIX Connection

FIX connections and sessions between the client and server are maintained as specified in the FIX protocol.

Each client will use the assigned IP address and port to establish a TCP/IP session with the server. The client will initiate a FIX session at the start of each trading day by sending the Logon message. The client will identify itself using the SenderCompID (49) field. The server will validate the CompID, password and IP address of the client.

The server will break the TCP/IP connection if messages are received before the exchange of Logons.

System can be configured in such a way that the test request at logon is either disabled or enabled[3]. A test request will not be sent along with the logon reply if the test request switch is set to its default “Off” mode. Depending on the System Configuration, the client’s logon message will be responded in two ways:

If during a logon of a SenderCompID, the server receives a second connection attempt via the same TCP/IP connection while a valid FIX session is already underway for that same SenderCompID, the server will immediately break the TCP/IP connection with the client without sending any messages. If the server receives another connection attempt from the same SenderCompID, while a session is already established, the connection attempt will be rejected via a Reject message without breaking the existing TCP/IP connection with the client. In both cases the server will increment the next inbound message sequence number expected from the client as well as its own outbound message sequence number.

### 5.1.1 Test Request at logon Enabled

Once the client is authenticated, the server will respond with a Logon message, followed by a Test Request. The Logon message will confirm the logon status and the Test Request’s purpose is to sync the Sequence numbers before sending any Missed Messages if any. The SessionStatus (1409) of this message will be Session Active (0). If the client’s Logon message included the field NewPassword (925) and the client is authenticated, the SessionStatus (1409) of the Logon sent by the server will be Session Active (0). The client must wait for the server’s Logon before sending additional messages. If additional messages are received from the client before the exchange of Logon messages, the TCP/IP connection with the client will be disconnected.

A successful logon response will be followed by a Test Request message. If the client responds to the Test Request with a Heartbeat message containing the appropriate Test Request ID and message sequence number, the server can start transmitting the missed messages or new messages in the Gateway.

The client would not receive any responses to application messages sent until sequence numbers are synchronized by responding to the Test Request via Heartbeat or Resend Request message. However, these messages will be processed by the system.

If the client ignores the Test Request because the sequence number in the message is higher than the expected sequence number, the Client is expected to send a Resend Request asking for the missed messages. After responding to the Resend Request the FIX Gateway would send another Test Request to make sure both the client and server is in sync before sending out any missed or new application messages.

If the client sends a Resend Request before the FIX Gateway send a Test Request, then the FIX Gateway will serve the Resend Request first. After responding to the Resend

[3] This is done via the process configuration CLIENT SESSION CONFIRMATION.

Request the FIX Gateway would send a Test Request to make sure both the client and server are in sync before sending out any missed or new application messages.

When the client sends a logon with a sequence number higher than expected by the FIX Gateway, the FIX gateway will send a Resend Request followed by the Test Request. The client is expected to serve the Resend Request and respond to the Server's Test Request via a Heartbeat message.

If a logon attempt fails because of an invalid SenderCompID, invalid TargetCompID or invalid IP address, invalid password or not having the appropriate privileges to login to the gateway the server will break the TCP/IP connection with the client without sending a Logout or Reject message. If the server receives a second connection attempt from the same TCP/IP while a valid FIX session is already underway for that same SenderCompID the system will reject the second attempt with a reject message while maintaining the original connection. If the server receives a second connection attempt from a different TCP/IP while a valid FIX session is already underway for that same SenderCompID, the server will break the TCP/IP connection for the second attempt without sending a Logout or Reject message. As the logon attempt failed, the server will not increment the next inbound message sequence number expected from the client.

If a logon attempt fails because of an expired password, a locked CompID or if logins are not currently permitted, the server will send a Logout message and then break the TCP/IP connection with the client. The server will increment the next inbound message sequence number expected from the client as well as its own outbound message sequence number.

If a logon attempt fails because of a session level failure (e.g. due to invalid EncryptMethod or DefaultApplVerID…etc) the inbound sequence number and the outbound sequence number both will not be incremented**.** In this scenario the message sequence number 1 will be sent with the **Error! Reference source not found.** message.

However if a session level failure occurs due to a message sent by a client which contains a sequence number that is less than what is expected and the PossDupFlag (43) is not set to "Y", then the server will send a Logout message and terminate the FIX connection. In this scenario the inbound sequence number will not be incremented but the outbound sequence number will be incremented.

## 5.2 Maintaining a FIX Session

### 5.2.1 Message Sequence Numbers

As outlined in the FIX protocol, the client and server will each maintain a separate and independent set of incoming and outgoing message sequence numbers. Sequence numbers should be initialized to 1 (one) at the start of the FIX session and be incremented throughout the session.

Monitoring sequence numbers will enable parties to identify and react to missed messages and to gracefully synchronize applications when reconnecting during a FIX session.

If any message sent by the client contains a sequence number that is less than what is expected and the PossDupFlag (43) is not set to "Y", the server will send a Logout message and terminate the FIX connection. The Logout message will contain the next expected sequence number in the Text (58) field.

If the server receives a message that cannot be processed (malformed message) it will not respond to that message and will not increment the sequence number it maintains. In such a scenario, when the next readable message is received by the server it will detect a sequence gap between the client and server. The server will send a Resend Request to the client requesting for messages from the sequence number the server is maintaining. If the client does not correct the malformed message to a readable one, the above event model will be repeated until there is no sequence gap.

A FIX session will not continue to the next trading day. The server will initialize its sequence numbers at the start of each day. The client is expected to employ the same logic.

### 5.2.2 Heartbeats

The client and server will use the Heartbeat message to exercise the communication line during periods of inactivity and to verify that the interfaces at each end are available. The heartbeat interval will be the HeartBtInt (108) specified in the client’s Logon message.

The server will send a Heartbeat message anytime it has not transmitted a message for the heartbeat interval. The client is expected to employ the same logic.

If the server detects inactivity for a period longer than the heartbeat interval plus a reasonable transmission time, it will send a Test Request message to force a Heartbeat from the client. If a response to the Test Request is not received by a reasonable transmission time, the server will send a Logout and break the TCP/IP connection. The client is expected to employ similar logic if inactivity is detected on the part of the server.

### 5.2.3 Increasing Expected Sequence Number

The client or server may use the Sequence Reset message in Gap Fill mode if it wishes to increase the expected incoming sequence number of the other party.

The client or server may also use the Sequence Reset message in Sequence Reset mode if it wishes to increase the expected incoming sequence number of the other party. The MsgSeqNum (34) in the header of such a message will be ignored. The Sequence Reset mode should only be used recover from an emergency situation. It should not be relied upon as a regular practice.

## 5.3 Terminating a FIX Connection

The client is expected to terminate each FIX connection at the end of each trading day before the server shuts down. The client will terminate a connection by sending the Logout message. The server will respond with a Logout message to confirm the termination. The client will then break the TCP/IP connection with the server. As recommended in the FIXT protocol, clients are advised to transmit a Test Request, to force a Heartbeat from the server, before initiating the logout process.

All open TCP/IP connections will be terminated by the server when it shuts down (a Logout message will not be sent).Under exceptional circumstances the server may initiate the termination of a connection during the trading day by sending the Logout message. The server will terminate the TCP/IP connection (a Logout will not be sent) if the number of messages that are buffered for a client exceeds *<1,000>*.

If, during the exchange of Logout messages, the client or sever detects a sequence gap, it should send a Resend Request.

## 5.4 Re-Establishing a FIX Session

If a FIX connection is terminated during the trading day it may be re-established via an exchange of Logon messages. Once the FIX session is re-established, the message sequence numbers will continue from the last message successfully transmitted prior to the termination.

### 5.4.1 Test Request at logon Enabled

Once the client is authenticated, the server will respond with a Logon message, followed by a Test Request. The Logon message will confirm the logon status and the Test Request's purpose is to sync the Sequence numbers before sending any Missed Messages if any. The SessionStatus (1409) of this message will be Session Active (0). If the client's Logon message included the field NewPassword (925) and the client is authenticated, the SessionStatus (1409) of the Logon sent by the server will be Session Active (0).

If the client responds to the Test Request with a Heartbeat message containing the appropriate Test Request ID and message sequence number, the server can start transmitting the missed messages or new messages in the Gateway. The client would not receive any responses to application messages sent until sequence numbers are synchronized by responding to the Test Request via Heartbeat or Resend Request message. However, these messages will be processed by the system. If the client does not respond to the Test Request during the heartbeat interval, the gateway will disconnect the client.

If the client ignores the Test Request because the sequence number in the message is higher than the expected sequence number, the client is expected to send a Resend Request asking for the missed messages. After responding to the Resend Request the FIX Gateway would send another Test Request to make sure both the client and server is in sync before sending out any missed or new application messages.

If the client sends a Resend Request before the FIX Gateway send a Test Request, then the FIX Gateway will serve the Resend Request first. After responding to the Resend Request the FIX Gateway would send a Test Request to make sure both the client and server are in sync before sending out any missed or new application messages.

When the client sends a logon with a sequence number higher than expected by the FIX Gateway, the FIX gateway will send a Resend Request followed by the Test Request. The client is expected to serve the Resend Request and respond to the Server's Test Request via a Heartbeat message.

Once the FIX session is re-established successfully, the message sequence numbers will continue from the last message successfully transmitted prior to the termination.

### 5.4.2 Resetting Sequence Numbers: Starting a New FIX Session

#### 5.4.2.1 Reset Initiated by the Client

If the client requires both parties to initialize (i.e. reset to 1) sequence numbers, it may use the ResetSeqNumFlag (141) field of the Logon message. The server will respond with a Logon message with the ResetSeqNumFlag (141) field set to "Y" to confirm the initialization of sequence numbers. In such cases, if the MsgSeqNo (34) of the Logon message is not reset to 1, the server will break the TCP/IP connection after sending a Logout. It will contain the SessionStatus (1409) as "101" (Logout due to session level failure) and an indication the rejection in the Text(58) field.

A client may also manually inform market operations that it would like the server to initialize its sequence numbers prior to the client's next login attempt.

These features are intended to help a client manage an emergency situation. Initializing sequence numbers on a re-login should not be relied upon as a regular practice.

#### 5.4.2.2 Reset Initiated by the Server

The server has been designed with fault tolerance and disaster recovery technology that should ensure that the server retains its incoming and outgoing message sequence numbers for each client in the unlikely event of an outage.

However, clients are required to support a manual request by NCDEX to initialize sequence numbers prior to the next login attempt.

# 6 RECOVERY

## 6.1 Resend Requests

The client may use the Resend Request message to recover any lost messages. As outlined in the FIX protocol, this message may be used in one of three modes:

(i) To request a single message. The BeginSeqNo (7) and EndSeqNo (16) should be the same.

(ii) To request a specific range of messages. The BeginSeqNo (7) should be the first message of the range and the EndSeqNo (16) should be the last of the range.

(iii) To request all messages after a particular message. The BeginSeqNo (7) should be the sequence number immediately after that of the last processed message and the EndSeqNo (16) should be zero (0).

The server caches the last *<1,000>* messages transmitted to each CompID. Clients are unable to use a Resend Request to recover messages not in the server's cache. If the client requests for a range of messages that have sequence numbers falling outside the cache size, a Sequence Reset message in Gap Fill mode will be sent for the missing messages and will send the available messages as per the request after that.

## 6.2 Possible Duplicates

The server handles possible duplicates according to the FIX protocol. The client and server will use the PossDupFlag (43) field to indicate that a message may have been previously transmitted with the same MsgSeqNum (34).

## 6.3 Possible Resends

The server may, in the case of real-time clients and in the circumstances outlined in Section 6.4, use the PossResend (97) field to indicate that a Trade Capture Report may have already been sent under a different MsgSeqNum (34). The client should validate the TradeReportID (571) of such a message against that of previous Trade Capture Reports received from the server during the current trading day.

If a Trade Capture Report with same TradeReportID (571) had been processed, the resent Trade Capture Report should be ignored. If the same TradeReportID (571) had not been processed, the message should be processed.

The server does not handle possible resends for client-initiated messages and ignores the value in the PossResend (97) field of such messages.

## 6.4 Transmission of Missed Messages

The Trade Capture Reports generated during a period when a real-time client is disconnected from the server will be sent to the client when it next reconnects. In the unlikely event the disconnection was due to an outage of the server, all such messages will include a PossResend (97) of "Y". The Trade Capture Report Acks generated when the client was disconnected will not be resent.

## 6.5 Application Sequencing and Recovery

The server supports the application sequencing and recovery features of FIX. A real-time client may use the Application Message Request to recover missed trades in scenarios such as the following:

(i) Trades are missed due to a late connection or disconnection during the day[4].

(ii) Session level recovery via a Resend Request is unavailable due to a sequence number reset initiated by the client or server.

(iii) All or some of the trades transmitted by the server during the current day are lost due to a failure at the client site.

This feature does not apply to query-based clients. An Application Message Request from such a client will be rejected by the server.

### 6.5.1 Application Sequencing by Server

The post trade system consists of a series of parallel partitions each of which services an exclusive set of instruments.

Each Trade Capture Report transmitted by the server will include the identity of the partition that generated the trade and the partition's internal sequence number for the trade in the fields ApplID (1180) and ApplSeqNum (1181) respectively. As the partitions operate in parallel and employ the same application sequencing scheme, an ApplSeqNum (1181) is only unique per ApplID (1180). The ApplSeqNum of each ApplID is initialized to "1" at the start of each trading day.

As a client will only receive a subset of the trades processed by each partition, the field ApplLastSeqNum (1350) is also included in each Trade Capture Report. This field will contain the ApplSeqNum of the last Trade Capture Report generated for client. This will enable clients to distinguish deliberate sequence gaps from application errors by comparing the value of ApplLastSeqNum (1350) to the ApplSeqNum (1181) of the last received Trade Capture Report from the same ApplID (1180). AppLastSeqNum (1350) is not included in the first Trade Capture Report transmitted to a client.  The ApplLastSeqNum (1350) field is not included in Trade Capture Reports sent in response to a Trade Capture Report Request.

### 6.5.2 Detecting an Application Sequence Gap

A client can detect a dropped message by comparing the ApplLastSeqNum (1350) of each new Trade Capture Report against the ApplSeqNum (1181) of the last trade received from the same ApplID (1180).

In the case of a reconnection, the client can either wait for the next Trade Capture Report to determine whether trades have been missed or issue a request for the most current ApplSeqNum for each ApplID.

#### 6.5.2.1.1 Requesting the Latest ApplSeqNum

The client may use the Application Message Request to request the latest ApplSeqNum for one or more ApplIDs. The ApplReqType (1347) of the message should be Request for Last ApplSeqNum (2).

#### 6.5.2.1.2 Response to Request for Latest ApplSeqNum

The server will respond to the Application Message Request with an Application Message Request Ack. If the request was unsuccessful for a particular ApplID, the reason will be specified in the field ApplResponseError (1354). In the case of a successful request, the ApplSeqNum of the last trade generated for the client by each ApplID will be specified in the field RefApplLastSeqNum (1357).

[4] Delete this line if the feature to send clients missed messages on login is enabled.

### 6.5.3 Requesting Retransmission of Missed Trades

The client may use the Application Message Request to recover any lost trades. The ApplReqType (1347) of the message should be Retransmission of Application Messages (0). The message may be used in one of four modes:

(i) To request a single trade. The ApplBegSeqNum (1182) and ApplEndSeqNum (1183) should be the same.

(ii) To request a specific range of trades. The ApplBegSeqNum (1182) should be the first trade of the range and the ApplEndSeqNum (1183) should be the last of the range.

(iii) To request all trades after a particular trade. The ApplBegSeqNum (1182) should be the application sequence number immediately after that of the last processed trade and the ApplEndSeqNum (1183) should be zero (0).

(iv) To request all trades for the day. The ApplBegSeqNum (1182) should be one (1) and the ApplEndSeqNum (1183) should be zero (0).

In all cases, the client should identify the partition to which the request relates via the field RefApplID (1355).

### 6.5.4 Response to a Trade Retransmission Request

The server will respond to the Application Message Request with an Application Message Request Ack to indicate whether the retransmission request is successful or not. If the request was unsuccessful for a particular ApplID, the reason will be specified in the field ApplResponseError (1354).

In the case of a successful retransmission request, the server will resend the requested Trade Capture Reports and Trade Capture Report Acks immediately after the Application Message Request Ack. Each Trade Capture Report and Trade Capture Report Ack will include an ApplResendFlag (1352) of "Y" to indicate that it is resent in response to an Application Message Request. The resent messages will not include the field ApplLastSeqNum (1350).

### 6.5.5 Disconnection Prior to Completion of Retransmission

If the FIX connection is terminated prior to the completion of the Trade Capture Report retransmission, the client should submit a new Application Message Request once it reconnects to the server.

# 7 MESSAGE FORMATS

This section provides details on the header and trailer, the seven administrative messages and six application messages utilized by the post trade gateway. Any client-initiated message not included in this section will be rejected by the server via a Reject or Business Message Reject.

## 7.1 Supported Message Types

### 7.1.1 Administrative Messages

All administrative messages may be initiated by either the client or the server.

| Message | MsgType | Usage |
|---|---|---|
| Heartbeat | 0 | Allows the client and server to exercise the communication line during periods of inactivity and verify that the interfaces at each end are available. |
| Logon | A | Allows the client and server to establish a FIX session. |
| Test Request | 1 | Allows the client or server to request a response from the other party if inactivity is detected. |
| Resend Request | 2 | Allows for the recovery of messages lost during a malfunction of the communications layers. |
| Reject | 3 | Used to reject a message that does not comply with FIXT. |
| Sequence Reset | 4 | Allows the client or server to increase the expected incoming sequence number of the other party. |
| Logout | 5 | Allows the client and server to terminate a FIX session. |

### 7.1.2 Application Messages: Trade Feed

#### 7.1.2.1 Client-Initiated

| Message | MsgType | Usage |
|---|---|---|
| Trade Capture Report Request | AD | Allows a client to request for all eligible trades or those that meet certain criteria. |
| Application Message Request | BW | Allows a real-time client to request one of the following:<br>Retransmission of missed trades<br>Latest ApplSeqNum of each ApplID |

#### 7.1.2.2 Server-Initiated

| Message | MsgType | Usage |
|---|---|---|
| Trade Capture Report | AE | Indicates one of the following:<br>Trade<br>Trade cancellation<br>Trade correction |
| Trade Capture Report Request Ack | AQ | Indicates whether a request for trade capture reports is successful or not. |
| Application Message Request Ack | BX | Indicates whether a request to retransmit trades or for the latest ApplSeqNum is successful or not. |

### 7.1.3 Application Messages: Cancellation of On-Book Trades

#### 7.1.3.1 Server-Initiated

| Message | MsgType | Usage |
|---|---|---|
| Trade Capture Report | AE | Indicates one of the following:.<br>(i) On-book trade cancelled |

### 7.1.4 Application Messages: Other

#### 7.1.4.1 Server-Initiated

| Message | MsgType | Usage |
|---|---|---|
| Business Message Reject | j | Indicates that an application message sent by the client could not be processed. |

## 7.2 Variations from the FIX Protocol

The server conforms to the FIX protocol except as follows:

(i) The Trade Capture Report message includes the custom fields PublicationDelay (30000) and OrderBook (30001). The data type of these fields is Int (i.e. integer).

(ii) The TradeRequestResult (749) field of the Trade Capture Report Request Ack message includes custom values 100 and 200 which indicate 'Cannot Match Selection Criteria' and 'Request Limit for Day Reached' respectively.

(iii) The Trade Capture Report and Trade Capture Report Request Ack messages include the field OrderSource (30004).The data types of this field is String.

(iv) The Trade Capture Report Request message includes the field MatchType (574).

(v) The Logon (A) message includes the ConnectionType (22001) and WorkstationType (22002). The data types of the fields are 'Enums'.

(vi) The Trade Capture Report message includes fields PreAccount (22005) and PreAllocAccount (22004). The data types of the fields are String.

(vii) The Trade Capture Report message includes the field InternalRefFlag (22009). The data type of the field is Integer.

(viii) The field TrdSubType (829) on Trade Capture Report and Trade Capture Report Ack messages contains the custom value 61 which indicates a Give-up Trade.

(ix) The field AllocType (626) on Trade Capture Report and Trade Capture Report Ack messages contains the custom value 17 which indicates a Give-up.

(x) The field AllocType (626) on Trade Capture Report and Trade Capture Report Ack messages contains the custom value 18 which indicates a Take-up.

(xi) The field PartyRole (452) on Trade Capture Report message contains the custom value 97 which indicates a Give-up Clearing Firm.

(xii) The field PartyRole (452) on Trade Capture Report message contains the custom value 98 which indicates a Take-up Clearing Firm.

(xiii) The field PartyRole (452) on Trade Capture Report message contains the custom value 200 which indicates an Obsolete Take-up Clearing Firm.

## 7.3 Message Header and Trailer

### 7.3.1 Message Header

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Length</th><th>Description</th></tr>
<tr><td>8</td><td>BeginString</td><td>Y</td><td>N/A</td><td>FIXT.1.1</td></tr>
<tr><td>9</td><td>BodyLength</td><td>Y</td><td>N/A</td><td>Number of characters after this field up to and including the delimiter immediately preceding the CheckSum.</td></tr>
<tr><td>35</td><td>MsgType</td><td>Y</td><td>I32</td><td>Message type.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>Heartbeat</td></tr>
<tr><td>1</td><td>Test Request</td></tr>
<tr><td>2</td><td>Resend Request</td></tr>
<tr><td>3</td><td>Reject</td></tr>
<tr><td>4</td><td>Sequence Reset</td></tr>
<tr><td>5</td><td>Logout</td></tr>
<tr><td>A</td><td>Logon</td></tr>
<tr><td>AD</td><td>Trade Capture Report Request</td></tr>
<tr><td>AE</td><td>Trade Capture Report</td></tr>
<tr><td>AQ</td><td>Trade Capture Report Request Ack</td></tr>
<tr><td>AR</td><td>Trade Capture Report Ack</td></tr>
<tr><td>BW</td><td>Application Message Request</td></tr>
<tr><td>BX</td><td>Application Message Request Ack</td></tr>
<tr><td>j</td><td>Business Message Reject</td></tr>
</table>
</td></tr>
<tr><td>49</td><td>SenderCompID</td><td>Y</td><td>30</td><td>CompID of the party sending the message.</td></tr>
<tr><td>56</td><td>TargetCompID</td><td>Y</td><td>30</td><td>CompID of the party the message is sent to.</td></tr>
<tr><td>34</td><td>MsgSeqNum</td><td>Y</td><td>I32</td><td>The sequence number of the message.</td></tr>
<tr><td>43</td><td>PossDupFlag</td><td>N</td><td>1</td><td>Whether the message was previously transmitted under the same MsgSeqNum (34). Absence of this field is interpreted as Original Transmission (N).
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>Y</td><td>Possible Duplicate</td></tr>
<tr><td>N</td><td>Original Transmission</td></tr>
</table>
</td></tr>
</table>

| | | | | |
|---|---|---|---|---|
| 97 | PossResend | N | 1 | Whether the message was previously transmitted under a different MsgSeqNum (34). Absence of this field is interpreted as Original Transmission (N).<br>**Value** **Meaning**<br>Y Possible Resend<br>N Original Transmission |
| 52 | SendingTime | Y | 30 | Time the message was transmitted. |
| 122 | OrigSendingTime | N | 30 | Time the message was originally transmitted. If the original time is not available, this should be the same value as SendingTime (52). Required if PossDupFlag (43) is Possible Duplicate (Y). |
| 1128 | ApplVerID | N | 1 | Version of FIX used in the message. Required if the message is generated by the server.<br>**Value** **Meaning**<br>9 FIX50SP2 |

### 7.3.2 Message Trailer

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| 10 | CheckSum | Y | 30 | |

## 7.4 Administrative Messages

### 7.4.1 Logon

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | A = Logon |
| **Message Body** | | | | |
| 98 | EncryptMethod | Y | 1 | Method of encryption.<br>**Value** **Meaning**<br>0 None |
| 108 | HeartBtInt | Y | UI32 | Indicates the heartbeat interval in seconds. |
| 141 | ResetSeqNum Flag | N | 1 | Indicates whether the client and server should reset sequence numbers. Absence of this field is interpreted as Do Not Reset Sequence Numbers (N).<br>**Value** **Meaning**<br>Y Reset Sequence Numbers<br>N Do Not Reset Sequence Numbers |
| 554 | Password | N | 30 | Password assigned to the CompID. Required if the message is generated by the client. |
| 925 | NewPassword | N | 30 | New password for the CompID. |
| 1409 | SessionStatus | N | UI32 | Status of the FIX session. Required if the message is generated by the server.<br>**Value** **Meaning**<br>0 Session Active<br>2 Password Due to Expire |

| | | | | |
|---|---|---|---|---|
| 1137 | DefaultApplVerID | | 1 | Default version of FIX messages used in this session.<br>**Value** **Meaning**<br>9 FIX50SP2 |
| **Standard Trailer** | | | | |

### 7.4.2 Logout

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | 5 = Logout |
| **Message Body** | | | | |
| 1409 | SessionStatus | N | UI32 | Status of the FIX session. Required if the message is generated by the server.<br>**Value** **Meaning**<br>4 Session logout complete<br>6 Account locked<br>7 Logons are not allowed at this time<br>8 Password expired <Delete if password validation is disabled><br>100 Other<br>101 Logout due to session level failure<br>102 Logout by market operations |
| 58 | Text | N | 60 | Text specifying reason for the logout. |
| **Standard Trailer** | | | | |

### 7.4.3 Heartbeat

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | 0 = Heartbeat |
| **Message Body** | | | | |
| 112 | TestReqID | N | N/A | Required if the heartbeat is a response to a Test Request. The value in this field should echo the TestReqID (112) received in the Test Request. |
| **Standard Trailer** | | | | |

### 7.4.4 Test Request

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | 1 = Test Request |
| **Message Body** | | | | |
| 112 | TestReqID | Y | N/A | Identifier for the request. |
| **Standard Trailer** | | | | |

### 7.4.5 Resend Request

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | 2 = Resend Request |
| **Message Body** | | | | |
| 7 | BeginSeqNo | Y | I32 | Sequence number of first message in range. |
| 16 | EndSeqNo | Y | I32 | Sequence number of last message in range. |
| **Standard Trailer** | | | | |

### 7.4.6 Reject

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | 3 = Reject |
| **Message Body** | | | | |
| 45 | RefSeqNum | Y | I32 | MsgSeqNum (34) of the rejected message. |
| 371 | RefTagID | N | I32 | If a message is rejected due to an issue with a particular field its tag number will be indicated. |
| 372 | RefMsgType | N | 2 | MsgType (35) of the rejected message. |
| 373 | SessionReject Reason | N | I32 | Code specifying the reason for the reject. Please refer to Section 9.1 for a list of reject codes. |
| 58 | Text | N | 60 | Text specifying the reason for the rejection. |
| **Standard Trailer** | | | | |

### 7.4.7 Sequence Reset

| Tag | Field Name | Req | Description | |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | 4 = Sequence Reset |
| **Message Body** | | | | |
| 36 | NewSeqNo | Y | I32 | Sequence number of the next message to be transmitted. |
| 123 | GapFillFlag | N | 1 | The mode in which the message is being used. Absence of this field is interpreted as Sequence Reset (N).<br>**Value** **Meaning**<br>Y Gap Fill<br>N Sequence Reset |
| **Standard Trailer** | | | | |

## 7.5 Application Messages: Trade Feed

### 7.5.1 Client-Initiated

#### 7.5.1.1 Trade Capture Report Request

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | AD = Trade Capture Report Request |
| **Message Body** | | | | |
| 568 | TradeRequestID | Y | 30 | Client specified unique identifier of the request. |
| 569 | TradeRequestType | Y | UI32 | Type of request.<br>**Value** **Meaning**<br>0 All Trades<br>1 Trades Matching Specified Criteria |
| 150 | ExecType | N | 1 | Type of trade report requested.<br>**Value** **Meaning**<br>F Trade<br>G Trade Correct |
| 574 | MatchType | N | I32 | Point in matching process trade was matched.<br>**Value** **Meaning**<br>4 Continuous Trading |
| 55 | Symbol | N | 30 | Unique identifier of the instrument. |
| 454 | NoSecurityAltID | N | I32 | If present, value in this field should always be “1”. |
| ➡ | 455 SecurityAltID | N | 30 | Identification number of the instrument. |
| ➡ | 456 SecurityAltID Source | N | 1 | Type of instrument identification used. Required if SecurityAltID (455) is specified.<br>**Value** **Meaning**<br>4 ISIN |
| 30001 | OrderBook | Y | I32 | Identifier of the order book.<br>**Value** **Meaning**<br>1 Regular |
| 461 | CFICode | N | 6 | Indicates the instrument type. Please refer to Section 8.1 for the valid CFI codes. |
| 167 | SecurityType | N | 10 | Indicates the instrument type. Please refer to Section 8.2 for the valid types. |
| 762 | SecuritySubType | N | 8 | Indicates the type of multi-legged instrument. Please refer to Section 8.3 for valid sub types. |
| 1227 | ProductComplex | N | 30 | Indicates the segment. Please refer to Section 8.4 for the valid groups. |

| | | | | | |
|---|---|---|---|---|---|
| 711 | NoUnderlyings | | N | 1 | Number of underlyings. If present, the value in this field should always be “1”. |
| ➡ | 311 | N | | 30 | Symbol of the underlying instrument. Required if NoUnderlyings (711) is specified. |
| 54 | Side | | N | 1 | Side of the executed order.<br>**Value** **Meaning**<br>1 Buy<br>2 Sell |
| 11 | ClOrdID | | N | 20 | Identifier of the executed order as specified by the entering firm. |
| 37 | OrderID | | N | 12 | Identifier of the executed order as specified by matching system. |
| 30004 | OrderSource | | N | 24 | Free form text (remarks) of up to 24 characters |
| 453 | NoPartyIDs | | Y | I32 | Number of party identifiers. (At least one block required to denote Order Entry Operator ID) |
| ➡ | 448 | PartyID | Y | 30 | Identifier of the party. |
| ➡ | 447 | PartyIDSource | Y | 1 | **Value** **Meaning**<br>D Proprietary/Custom Code |
| ➡ | 452 | PartyRole | Y | I32 | Role of the specified PartyID (448).<br>**Value** **Meaning**<br>1 Executing Firm<br>4 Clearing Firm<br>44 Order Entry Operator ID (Mandatory)<br>53 Trader Mnemonic<br>58 Entering Unit |
| **Standard Trailer** | | | | | |

### 7.5.1.2 Application Message Request

| Tag | Field Name | | Req | Length | Description |
|---|---|---|---|---|---|
| **Standard Header** | | | | | |
| 35 | MsgType | | Y | I32 | BW = Application Message Request |
| **Message Body** | | | | | |
| 1346 | ApplReqID | | Y | I32 | Client specified unique identifier of the request. |
| 1347 | ApplReqType | | Y | I32 | Type of request.<br>**Value** **Meaning**<br>0 Retransmission of Application Messages<br>2 Request for Last ApplSeqNum |
| 1351 | NoApplIDs | | Y | I32 | Number of ApplIDs to which the request relates. |
| ➡ | 1355 | RefApplID | Y | I32 | Identifier of the partition. |
| ➡ | 1182 | ApplBeg SeqNum | N | I32 | Application sequence number of first message in range to be resent. Required if ApplReqType (1347) is Retransmission of Application Messages (0). |
| ➡ | 1183 | ApplEnd SeqNum | N | I32 | Application sequence number of last message in range to be resent. Required if ApplReqType (1347) is Retransmission of Application Messages (0). |
| **Standard Trailer** | | | | | |

### 7.5.2 Server-Initiated

#### 7.5.2.1 Trade Capture Report

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | AE = Trade Capture Report |
| **Message Body** | | | | |
| 1180 | ApplID | Y | I32 | Identifier of the partition. |
| 1181 | ApplSeqNum | Y | I32 | Partition’s sequence number for trade. |
| 1350 | ApplLastSeqNum | N | I32 | ApplSeqNum of last trade generated for client. Required if ApplResendFlag (1352) is “N” and TradeRequestID (568) is not present. |
| 1352 | ApplResendFlag | N | 1 | Whether the message is sent in response to an Application Message Request. Absence of this field is interpreted as Original Transmission (N).<br>**Value** **Meaning**<br>Y Response to Application Message Request<br>N Original Transmission |
| 568 | TradeRequestID | N | 30 | Identifier of the Trade Capture Report Request the message is sent in response to. |
| 912 | LastRptRequested | N | 1 | Indicates the last message sent in response to a Trade Capture Report Request.<br>**Value** **Meaning**<br>Y Last Message |
| 571 | TradeReportID | N | 10 | Identifier of the message. |
| 1003 | TradeID | Y | 10 | Identifier of the trade. |
| 1040 | SecondaryTradeID | N | 7 | The numeric identifier of the trade. |
| 820 | TradeLinkID | Y | 10 | Identifier of the transaction in which the trade was executed. |
| 1041 | FirmTradeID | N | 11 | Identifier assigned to an EFP trade by the counterparties. Required if TrdType (828) is EFP Trade (1). |
| 572 | TradeReportRefID | N | 21 | Reference to trade being cancelled or corrected or pre-released. Required if TradeReportTransType (487) is Cancel (1), Replace (2) or Release (3). |
| 1123 | TradeHandlingInstr | Y | 1 | Handling instructions to client.<br>**Value** **Meaning**<br>0 Trade Confirmation |

| | | | | |
|---|---|---|---|---|
| 856 | TradeReportType | Y | I32 | Type of trade report.<br>**Value** **Meaning**<br>0 Submit<br>4 Addendum<br>5 No/Was<br>6 Trade Report Cancel<br>7 Trade Break |
| 150 | ExecType | Y | 1 | Type of execution being reported.<br>**Value** **Meaning**<br>F Trade<br>G Trade Correct<br>H Trade Cancel |
| 487 | TradeReportTrans Type | Y | 10 | Type of transaction being reported.<br>**Value** **Meaning**<br>0 New<br>1 Cancel<br>2 Replace<br>3 Release |
| 573 | MatchStatus | Y | 5 | Status of the trade.<br>**Value** **Meaning**<br>0 Matched<br>1 Unmatched |
| 828 | TrdType | N | I32 | Type of trade. Absence of this field should be interpreted as Regular Trade (0).<br>**Value** **Meaning**<br>0 Regular Trade<br>1 EFP Trade |
| 829 | TrdSubType | N | I32 | **Value** **Meaning**<br>61 Give-up Trade |
| 60 | TransactTime | Y | 21 | Time trade, cancellation or correction occurred. |
| 32 | LastQty | Y | 20,8 | Traded quantity. |
| 31 | LastPx | Y | 20,8 | Traded price. |

| | | | | | |
|---|---|---|---|---|---|
| 15 | Currency | | N | 30 | Currency attached to the instrument |
| 64 | SettlDate | | Y | 10 | Date on which the trade will settle. |
| 574 | MatchType | | Y | I32 | Point in matching process trade was matched.<br>**Value** **Meaning**<br>4 Continuous Trading |
| 55 | Symbol | | N | 30 | Identifier of the instrument. |
| 454 | NoSecurityAltID | | N | I32 | If present, value in this field will always be "1". |
| ➡ | 455 | SecurityAltID | N | 30 | Identification number of the instrument. |
| ➡ | 456 | SecurityAltID Source | N | 1 | Type of instrument identification used. Required if SecurityAltID (455) is specified.<br>**Value** **Meaning**<br>4 ISIN |
| 30001 | OrderBook | | Y | I32 | Identifier of the order book.<br>**Value** **Meaning**<br>1 Regular<br>4 EFP Trade |
| 461 | CFICode | | N | 6 | Indicates the instrument type. Please refer to Section 8.1 for the valid CFI codes. |
| 167 | SecurityType | | N | 10 | Indicates the instrument type. Please refer to Section 8.2 for the valid types. |
| 762 | SecuritySubType | | N | 8 | Indicates the type of multi-legged instrument. Please refer to Section 8.3 for the valid sub types. Required for multi-legged instruments. |
| 1227 | ProductComplex | | N | 30 | Segment the instrument belongs to. Please refer to Section 8.4 for the valid groups. |
| 541 | MaturityDate | | N | 10 | Date an instrument expires. Required for futures, options and fixed income instruments. |
| 711 | NoUnderlyings | | N | 1 | Number of underlyings for the instrument. If present, the value in this field will always be "1". Required for futures and options instruments. |
| ➡ | 311 | Underlying Symbol | N | 30 | Symbol of the underlying instrument. Required if NoUnderlyings (711) is specified. |

| | | | | | |
|---|---|---|---|---|---|
| 201 | PutOrCall | | N | 1 | Required for options instruments.<br>**Value** **Meaning**<br>0 Put<br>1 Call |
| 202 | StrikePrice | | N | 20,8 | Strike price of an options instrument. Required for options instruments. |
| 1193 | SettlMethod | | N | 1 | Type of settlement. Required for futures and options instruments.<br>**Value** **Meaning**<br>0 Cash<br>1 Physical |
| 1194 | ExerciseStyle | | N | 1 | Type of exercise. Required for options instruments.<br>**Value** **Meaning**<br>0 European<br>1 American |
| 231 | ContractMultiplier | | N | 8 | Multiplier of a derivatives contract. Required for futures and options instruments. |
| 442 | MultiLegReporting Type | | N | I32 | Instrument type report is generated for. Absence should be interpreted as Single Instrument (1).<br>**Value** **Meaning**<br>1 Single Instrument<br>2 Leg of Multi-Leg Instrument<br>3 Multi-Leg Instrument<br>9 Multi-leg order on leg instruments |
| 552 | NoSides | | Y | 1 | Number of sides which will always be "1". |
| ➡ | 54 | Side | Y | 1 | Side of the executed order.<br>**Value** **Meaning**<br>1 Buy<br>2 Sell |
| ➡ | 1427 | SideExecID | N | 12 | Identifier of the execution received by the order. |
| ➡ | 453 | NoPartyIDs | Y | I32 | Number of party identifiers. |

|  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|
| ➡ | ➡ | 448 | PartyID | Y | 30 | Identifier of the party. |
| ➡ | ➡ | 447 | PartyID Source | Y | 1 | **Value** **Meaning**<br>D Proprietary/Custom Code |
| ➡ | ➡ | 452 | Party Role | Y | I32 | Role of the specified PartyID (448).<br>**Value** **Meaning**<br>1 Executing Firm<br>4 Clearing Firm<br>17 Contra Firm<br>37 Contra Trading Mnemonic<br>44 Order Entry Operator ID<br>53 Trading Mnemonic<br>97 Give-up Clearing Firm<br>98 Take-up Clearing Firm<br>200 Obsolete Take-up Clearing Firm |
| ➡ | 1 | Account |  | N | 32 | Identifier of the investor account. |
| ➡ | 22005 | PreAccount |  | N | 1 | Investor account of the trade prior to a Client ID amendment. |
| ➡ | 581 | AccountType |  | N | I32 | Type of the investor account.<br>**Value** **Meaning**<br>1 Client<br>3 House |
| ➡ | 79 | AllocAccount |  | N | 30 | Sub-account mnemonic. The CP Code of the Client ID will be mentioned here. |
| ➡ | 22004 | PreAllocAccount |  | N | 30 | The CP Code of the Client ID prior to the trade amendment will be mentioned here. |
| ➡ | 1115 | OrderCategory |  | Y | 1 | Type of interest behind trade.<br>**Value** **Meaning**<br>1 Order |
| ➡ | 1444 | SideLiquidityInd |  | N | UI32 | Whether the order added or removed liquidity.<br>**Value** **Meaning**<br>1 Added Liquidity<br>2 Removed Liquidity |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
| ➡ | 37 | OrderID | N | 12 | Identifier of the executed order as specified by the matching system. |
| ➡ | 11 | ClOrdID | N | 20 | Identifier of the executed order as specified by the entering firm. |
| ➡ | 1093 | LotType | Y | 1 | Lot type of the order.<br>**Value** **Meaning**<br>2 Round Lot<br>3 Block Lot |
| ➡ | 30004 | OrderSource | N | 24 | Free form text specified for the order. Text (remarks) is of up to 24 characters. |
| 32022 | LastOptPx |  | N | 20,8 | Converted price of the executed volatility of the options instrument. |
| 1188 | Volatility |  | N | 20,8 | Converted volatility of the executed price of the options instrument. |
| 70 | AllocID |  | N | 10 | Unique identifier of the give-up trade<br>Required if the TCR is communicating a give-up trade |
| 72 | RefAllocID |  | N | 10 | Identifies the original AllocID (70)<br>To be used with AllocTransType (71) as Replace (1) |
| 71 | AllocTransType |  | N | 4 | Communicates if the give-up or take-up is new or cancelled<br>**Value** **Meaning**<br>0 New<br>1 Replace<br>2 Cancel |
| 626 | AllocType |  | N | 4 | Indicates if the TCR is communicating a give-up or a take-up<br>**Value** **Meaning**<br>17 Give-up<br>18 Take-up |

<table>
<tr><td>87</td><td>AllocStatus</td><td>N</td><td>4</td><td>Communicates the status of the give-up or take-up process
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>0</td><td>Accepted</td></tr>
<tr><td>6</td><td>Alloc Pending</td></tr>
<tr><td>5</td><td>Rejected</td></tr>
<tr><td>9</td><td>Claimed</td></tr>
<tr><td>10</td><td>Refused</td></tr>
<tr><td>12</td><td>Cancelled</td></tr>
</table></td></tr>
<tr><td>1840</td><td>TradeAllocStatus</td><td>N</td><td>4</td><td>Communicates the type of the response received by the RMS
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>2</td><td>Cleared</td></tr>
<tr><td>3</td><td>Rejected</td></tr>
</table></td></tr>
<tr><td>220069</td><td>InternalRefFlag</td><td>Y</td><td>15</td><td>Internal reference flag</td></tr>
<tr><td colspan="5"><b>Standard Trailer</b></td></tr>
</table>

### 7.5.2.2 Trade Capture Report Request Ack

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | AQ = Trade Capture Report Request Ack |
| **Message Body** | | | | |
| 568 | TradeRequestID | Y | 30 | Identifier of the request being acknowledged. |
| 569 | TradeRequestType | Y | UI32 | Value specified in the request. |
| 750 | TradeRequestStatus | Y | UI32 | Whether the request is accepted or rejected.<br>**Value** **Meaning**<br>0 Accepted<br>2 Rejected |
| 749 | TradeRequestResult | Y | UI32 | Reason the request is rejected.<br>**Value** **Meaning**<br>0 Successful<br>8 TradeRequestType not supported<br>9 Not Authorized<br>100 Cannot Match Selection Criteria<br>200 Request Limit for Day Reached |
| 748 | TotNumTradeReports | N | UI32 | Number of Trade Capture Reports that will be sent in response to the request. Required if TradeRequestStatus (750) is Accepted (0). |
| 30004 | OrderSource | N | 24 | Free form text (remarks) of up to 24 characters. |
| **Standard Trailer** | | | | |

### 7.5.2.3 Application Message Request Ack

| Tag | Field Name | | Req | Length | Description |
|---|---|---|---|---|---|
| **Standard Header** | | | | | |
| 35 | MsgType | | Y | I32 | BX = Application Message Request Ack |
| **Message Body** | | | | | |
| 1353 | ApplResponseID | | Y | I32 | Server specified identifier of the acknowledgement. |
| 1346 | ApplReqID | | Y | I32 | Identifier of the request being acknowledged. |
| 1347 | ApplReqType | | Y | I32 | Value specified in the request being acknowledged. |
| 1351 | NoApplIDs | | Y | I32 | Number of ApplIDs to which the request relates. |
| ➡ | 1355 | RefApplID | Y | I32 | Identifier of the partition. |
| ➡ | 1182 | ApplBeg SeqNum | N | I32 | Application sequence number of first message in range to be resent. Required if ApplReqType (1347) is Retransmission of Application Messages (0). |
| ➡ | 1183 | ApplEnd SeqNum | N | I32 | Application sequence number of last message in range to be resent. Required if ApplReqType (1347) is Retransmission of Application Messages (0). |
| ➡ | 1357 | RefAppl LastSeq Num | N | I32 | ApplSeqNum of the last trade generated for the client. Required if ApplReqType (1347) is Request for Last ApplSeqNum (2) and ApplResponseError (1354) is not specified. |
| ➡ | 1354 | Appl Response Error | N | I32 | Reason request is rejected.<br>**Value** **Meaning**<br>0 ApplID Does Not Exist<br>1 Requested Trades are Not Available<br>2 Client Not Authorised |
| 58 | Text | | N | 60 | Text specifying the reason for the rejection.If requests for multiple partitions are rejected, the reason for the rejection at the last partition will be specified. |
| **Standard Trailer** | | | | | |

## 7.6 Application Messages: Cancellation of On-Book Trades

### 7.6.1 Server-Initiated

#### 7.6.1.1 Trade Capture Report

Please refer to Section 7.5.2.1

## 7.7 Application Messages: Trade Amendment

### 7.7.1 Client-Initiated

#### 7.7.1.1 Trade Capture Report

| Tag | Field Name | | Req | Length | Description |
|---|---|---|---|---|---|
| **Standard Header** | | | | | |
| 35 | MsgType | | Y | I32 | AE = Trade Capture Report |
| **Message Body** | | | | | |
| 571 | TradeReportID | | N | 10 | Optional client identifier. |
| 1003 | TradeID | | Y | 10 | Server-assigned identifier of the trade. |
| 856 | TradeReportType | | Y | I32 | Type of trade report.<br>**Value** **Meaning**<br>4 Addendum |
| 487 | TradeReportTrans Type | | Y | 10 | Type of request.<br>**Value** **Meaning**<br>0 New |
| 55 | Symbol | | Y | 30 | Identifier of the instrument. |
| 60 | TransactTime | | Y | 21 | Time the message was generated by the client. |
| 552 | NoSides | | Y | 1 | Number of sides. The value in this field should always be "1". |
| ➡ | 54 | Side | Y | 1 | Side of the trade.<br>**Value** **Meaning**<br>1 Buy<br>2 Sell |
| ➡ | 1 | Account | Y | 32 | Identifier of the investor account. |
| ➡ | 581 | AccountType | Y | I32 | Type of the investor account.<br>**Value** **Meaning**<br>1 Client<br>3 House |
| ➡ | 79 | AllocAccount | N | 30 | Sub-account mnemonic. The CP Code of the Client ID will be mentioned here. |
| ➡ | 30004 | OrderSource | N | 24 | Free form text (remarks) of up to 24 characters. |
| 22009 | InternalRefFlag | | Y | 15 | Internal reference flag |
| 453 | NoPartyIDs | | Y | I32 | Number of party identifiers. (At least one block required to denote Order Entry Operator ID) |
| ➡ | 448 | PartyID | Y | 30 | Identifier of the party. |

| | | | | | |
|---|---|---|---|---|---|
| ➡ | 447 | PartyIDSource | Y | 1 | **Value** **Meaning**<br>D Proprietary/Custom Code |
| ➡ | 452 | PartyRole | Y | I32 | Role of the specified PartyID (448).<br>**Value** **Meaning**<br>44 Order Entry Operator ID (Mandatory) |
| **Standard Trailer** | | | | | |

### 7.7.2 Server-Initiated

#### 7.7.2.1 Trade Capture Report Ack

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | AR = Trade Capture Report Ack |
| **Message Body** | | | | |
| 571 | TradeReportID | N | 10 | Value submitted in amendment request. |
| 1003 | TradeID | N | 10 | Value submitted in amendment request. |
| 939 | TrdRptStatus | Y | I32 | Whether the request is accepted or rejected.<br>**Value** **Meaning**<br>0 Accepted<br>1 Rejected |
| 751 | TradeReportReject Reason | N | I32 | Code specifying the reason for rejection. Please refer to Section 9.2 for a list of reject codes. Required if TrdRptStatus (939) is Rejected (1). |
| 58 | Text | N | 60 | Text specifying the reason for rejection. Required if TrdRptStatus (939) is Rejected (1). |
| 856 | TradeReportType | Y | I32 | Value submitted in amendment request. |
| 573 | MatchStatus | Y | 1 | Status of the trade.<br>**Value** **Meaning**<br>0 Matched |
| 55 | Symbol | Y | 30 | Identifier of the instrument. |
| 60 | TransactTime | Y | 21 | Time the message was generated by the server. |
| 552 | NoSides | Y | 1 | Number of sides. The value in this field will always be “1”. |
| ➡ | 54 Side | Y | 1 | Value submitted in amendment request. |
| ➡ | 1 Account | Y | 32 | Identifier of the investor account. |
| ➡ | 581 AccountType | Y | I32 | Type of the investor account.<br>**Value** **Meaning**<br>1 Client<br>3 House |
| ➡ | 79 AllocAccount | N | 30 | Sub-account mnemonic. The CP Code of the Client ID will be mentioned here. |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
| ➡ | 30004 | OrderSource | N | 24 | Free form text (remarks) of up to 24 characters. |
| 22009 | InternalRefFlag |  | Y | 15 | Internal reference flag |
| **Standard Trailer** |  |  |  |  |  |

### 7.7.2.2 Trade Capture Report

Please refer to Section 7.5.2.1.

# 7.8 Application Messages: Give Ups and Take Ups

## 7.8.1 Client-Initiated

### 7.8.1.1 Trade Capture Report

| Tag | Field Name | | | Req | Length | Description |
|---|---|---|---|---|---|---|
| **Standard Header** | | | | | | |
| 35 | MsgType | | | Y | I32 | AE = Trade Capture Report |
| **Message Body** | | | | | | |
| 571 | TradeReportID | | | N | 10 | Identifier of the message. |
| 1003 | TradeID | | | Y | 10 | Identifier of the trade. |
| 856 | TradeReportType | | | Y | I32 | Type of trade report.<br>**Value** **Meaning**<br>0 Submit |
| 487 | TradeReportTrans Type | | | Y | 10 | Type of transaction being reported.<br>**Value** **Meaning**<br>0 New |
| 829 | TrdSubType | | | N | I32 | **Value** **Meaning**<br>61 Give-up Trade |
| 60 | TransactTime | | | Y | 21 | Time trade, cancellation or correction occurred. |
| 55 | Symbol | | | N | 30 | Identifier of the instrument. |
| 552 | NoSides | | | Y | 1 | Number of sides which will always be “1”. |
| ➡ | 54 | Side | | Y | 1 | Side of the executed order.<br>**Value** **Meaning**<br>1 Buy<br>2 Sell |
| ➡ | 453 | NoPartyIDs | | Y | I32 | Number of party identifiers. (At least one block required to denote Order Entry Operator ID) |
| ➡ | ➡ | 448 | PartyID | Y | 30 | Identifier of the party. |
| ➡ | ➡ | 447 | PartyID Source | Y | 1 | **Value** **Meaning**<br>D Proprietary/Custom Code |
| ➡ | ➡ | 452 | Party Role | Y | I32 | Role of the specified PartyID (448).<br>**Value** **Meaning**<br>98 Take-up Clearing Firm<br>44 Order Entry Operator ID (Mandatory) |
| ➡ | 1 | Account | | N | 32 | Identifier of the investor account. |

| | | | | | |
|---|---|---|---|---|---|
| ➡ | 581 | AccountType | N | I32 | Type of the investor account.<br>**Value** **Meaning**<br>1 Client<br>3 House |
| ➡ | 79 | AllocAccount | N | 30 | Sub-account mnemonic. The CP Code of the Client ID will be mentioned here. |
| ➡ | 30004 | OrderSource | N | 24 | Free form text specified for the order. Text (remarks) is of up to 24 characters. |
| 70 | AllocID | | N | 10 | Unique identifier of the give-up trade<br>Required if the TCR is communicating a give-up trade |
| 626 | AllocType | | N | 4 | Indicates if the TCR is communicating a give-up or a take-up<br>**Value** **Meaning**<br>17 Give-up<br>18 Take-up |
| 87 | AllocStatus | | N | 4 | Communicates the status of the give-up or take-up process<br>**Value** **Meaning**<br>9 Claimed<br>10 Refused |
| **Standard Trailer** | | | | | |

### 7.8.2 Server-Initiated

#### 7.8.2.1 Trade Capture Report Ack

| Tag | Field Name | Req | Length | Description |
|---|---|---|---|---|
| **Standard Header** | | | | |
| 35 | MsgType | Y | I32 | AR = Trade Capture Report Ack |
| **Message Body** | | | | |
| 571 | TradeReportID | N | 10 | Value submitted in amendment request. |
| 1003 | TradeID | N | 10 | Value submitted in amendment request. |
| 939 | TrdRptStatus | Y | 1 | Whether the request is accepted or rejected.<br>**Value** **Meaning**<br>0 Accepted<br>1 Rejected<br>4 Pending New |
| 751 | TradeReportReject Reason | N | I32 | Code specifying the reason for rejection. Please refer to Section 9.2 for a list of reject codes. Required if TrdRptStatus (939) is Rejected (1). |
| 58 | Text | N | 60 | Text specifying the reason for rejection. Required if TrdRptStatus (939) is Rejected (1). |
| 856 | TradeReportType | Y | I32 | Value submitted in amendment request. |
| 829 | TrdSubType | N | I32 | **Value** **Meaning**<br>61 Give-up Trade |
| 573 | MatchStatus | Y | 1 | Status of the trade.<br>**Value** **Meaning**<br>0 Matched |
| 55 | Symbol | Y | 30 | Identifier of the instrument. |
| 60 | TransactTime | Y | 21 | Time the message was generated by the server. |
| 552 | NoSides | Y | I32 | Number of sides. The value in this field will always be “1”. |
| ➡ | 54 Side | Y | 1 | Value submitted in amendment request. |
| ➡ | 1 Account | Y | 32 | Identifier of the investor account. |

| | | | | | |
|---|---|---|---|---|---|
| ➡ | 581 | AccountType | Y | I32 | Type of the investor account.<br>**Value** **Meaning**<br>1 Client<br>3 House |
| ➡ | 79 | AllocAccount | N | 30 | Sub-account mnemonic. The CP Code of the Client ID will be mentioned here. |
| ➡ | 30004 | OrderSource | N | 24 | Free form text (remarks) of up to 24 characters. |
| 70 | AllocID | | N | 10 | Unique identifier of the give-up trade<br>Required if the TCR Ack is communicating a give-up trade |
| 626 | AllocType | | N | 4 | Indicates if the TCR is communicating a give-up or a take-up<br>**Value** **Meaning**<br>17 Give-up<br>18 Take-up |
| 87 | AllocStatus | | N | 4 | Communicates the status of the give-up or take-up process<br>**Value** **Meaning**<br>9 Claimed<br>10 Refused |
| 71 | AllocTransType | | N | 4 | Communicates whether the give-up or the take-up is amended<br>**Value** **Meaning**<br>0 New<br>1 Replace |
| 22009 | InternalRefFlag | | Y | 15 | Communicates the Token ID of the instrument |
| **Standard Trailer** | | | | | |

### 7.8.2.2 Trade Capture Report

Please refer to section 7.5.2.1

## 7.9 Application Messages: Other

### 7.9.1 Server-Initiated

#### 7.9.1.1 Business Message Reject

<table>
<tr><th>Tag</th><th>Field Name</th><th>Req</th><th>Length</th><th>Description</th></tr>
<tr><td colspan="5"><b>Standard Header</b></td></tr>
<tr><td>35</td><td>MsgType</td><td>Y</td><td>I32</td><td>j = Business Message Reject</td></tr>
<tr><td colspan="5"><b>Message Body</b></td></tr>
<tr><td>45</td><td>RefSeqNum</td><td>Y</td><td>I32</td><td>MsgSeqNum (34) of the rejected message.</td></tr>
<tr><td>372</td><td>RefMsgType</td><td>Y</td><td>2</td><td>MsgType (35) of the rejected message.</td></tr>
<tr><td>371</td><td>RefTagID</td><td>N</td><td>I32</td><td>If a message is rejected due to an issue with a particular field its tag number will be indicated.</td></tr>
<tr><td>379</td><td>BusinessRejectRefID</td><td>N</td><td>I32</td><td>Client specified identifier of the rejected message if it is available.
<table>
<tr><th>Message</th><th>Identifier</th></tr>
<tr><td>TradeCaptureReport</td><td>FirmTradeID(1041)</td></tr>
<tr><td>TradeCaptureReportRequest</td><td>TradeRequestID(568</td></tr>
<tr><td>ApplicationMessageRequest</td><td>ApplReqID(1346)</td></tr>
</table></td></tr>
<tr><td>380</td><td>BusinessReject Reason</td><td>Y</td><td>I32</td><td>Code specifying the reason for the reject. Please refer to Section 9.3 for a list of reject codes.</td></tr>
<tr><td>58</td><td>Text</td><td>N</td><td>60</td><td>Text specifying the reason for the rejection.</td></tr>
<tr><td colspan="5"><b>Standard Trailer</b></td></tr>
</table>

# 8 INSTRUCTION CLASSIFICATION

## 8.1 CFI Codes

| CFI Code | Description |
|---|---|
| *F* | *Future* |
| *FM* | *Futures Strategy* |

## 8.2 Security Types

| Type | Description |
|---|---|
| *FUT* | *Future* |
| *MLEG* | *Multi-Leg Instrument* |

## 8.3 Security Sub Types

| Sub Type | Description |
|---|---|
| *CS* | *Calendar Spread* |

## 8.4 Segments

| **CFI Code** | Description |
|---|---|
| *Main* | *Main Board* |
| *Secondary* | *Secondary Board* |
| *Default* | *Default Board* |
| *FU* | *Futures* |
| *OP* | *Options* |
| *CS* | *Calendar Spreads* |

# 9 REJECT CODES

## 9.1 Reject

| Session Reject Reason | Meaning |
|---|---|
| 1 | Required tag missing[5] |
| 2 | Tag not defined for this message type6 |
| 4 | Tag specified without a value |
| 5 | Value is incorrect (out of range) for this tag |
| 6 | Incorrect data format for value |
| 9 | CompID problem |
| 11 | Invalid MsgType[7] |
| 13 | Tag appears more than once |
| 14 | Tag specified out of required order |
| 15 | Repeating group fields out of order |
| 16 | Incorrect NumInGroup count for repeating group |
| 18 | Invalid or unsupported application version |
| 99 | Other |

## 9.2 Trade Capture Report Ack

| Trade Report Reject Reason | Meaning |
|---|---|
| 1 | Unknown RootPartyID or PartyID |
| 2 | Unknown instrument |
| 3 | Not authorised to submit trade reports |
| 4 | Invalid TrdType (828) or TrdSubType (829) |
| 99 | Other |

Please refer to the *NCDEX - Reject Codes and Reasons_v1.00.doc* for the list of reject codes and meanings specific to NCDEX.

[5] This reject reason is sent when all the required tags for the message are not present in a message that is recognized by the gateway.The exception to this is when SendingTime and OrigSendingTime tags are not present in a message. In these two cases, the system responds with reject reason ‘Required SendingTime field missing’ and ‘Conditionally required OrigSendingTime field missing’ respectively.

[6]Delete this reject reason if the configuration to reject unknown fields in application messages is disabled.

[7] This reject reason is sent when a message that is not defined in the FIX data dictionary is received by the gateway

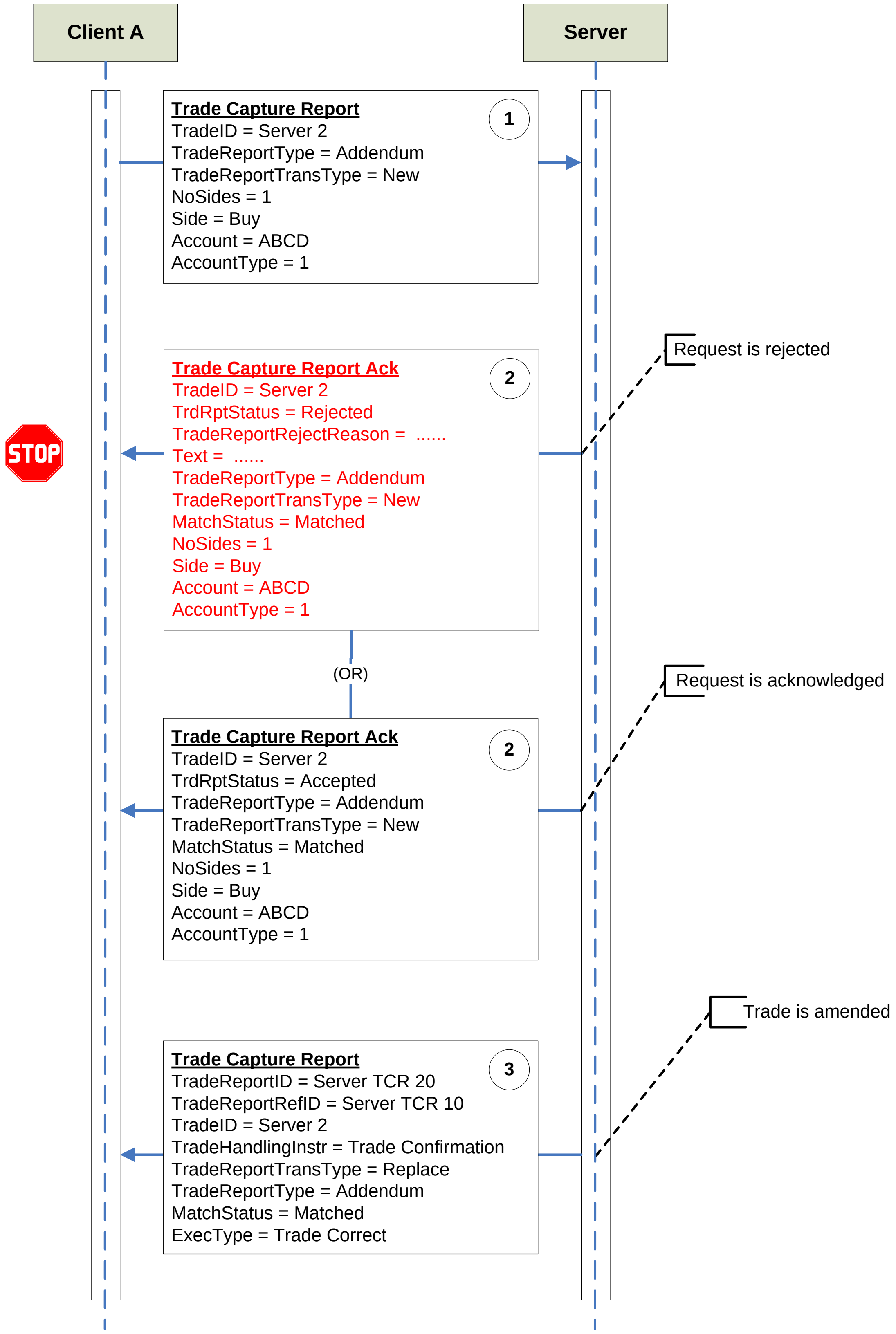
## 9.3 Business Message Reject

| Business Reject Reason | Meaning |
|---|---|
| 3 | Unsupported message type[8] |
| 4 | Application not available |
| 5 | Conditionally required field missing |
| 30 | Session not in sync |

[8] This reject reason is sent when the received message is not defined as a valid message for the Post Trade Gateway

# 10 PROCESS FLOWS

## 10.1 Trade Amend



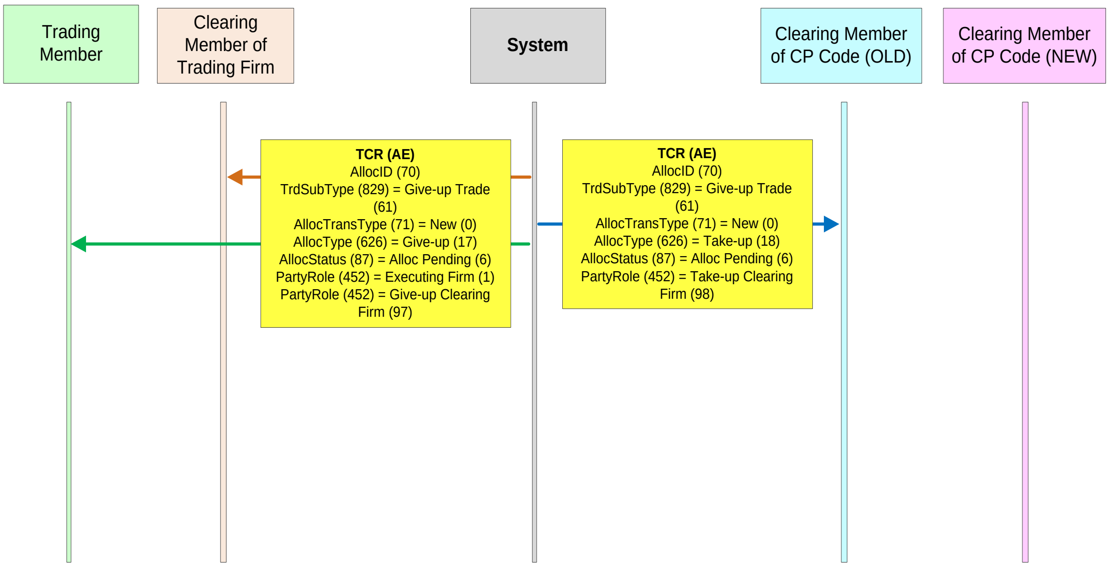
**Trade Capture Report**
TradeID = Server 2
TradeReportType = Addendum
TradeReportTransType = New
NoSides = 1
Side = Buy
Account = ABCD
AccountType = 1

**Trade Capture Report Ack**
TradeID = Server 2
TrdRptStatus = Rejected
TradeReportRejectReason = ......
Text = ......
TradeReportType = Addendum
TradeReportTransType = New
MatchStatus = Matched
NoSides = 1
Side = Buy
Account = ABCD
AccountType = 1

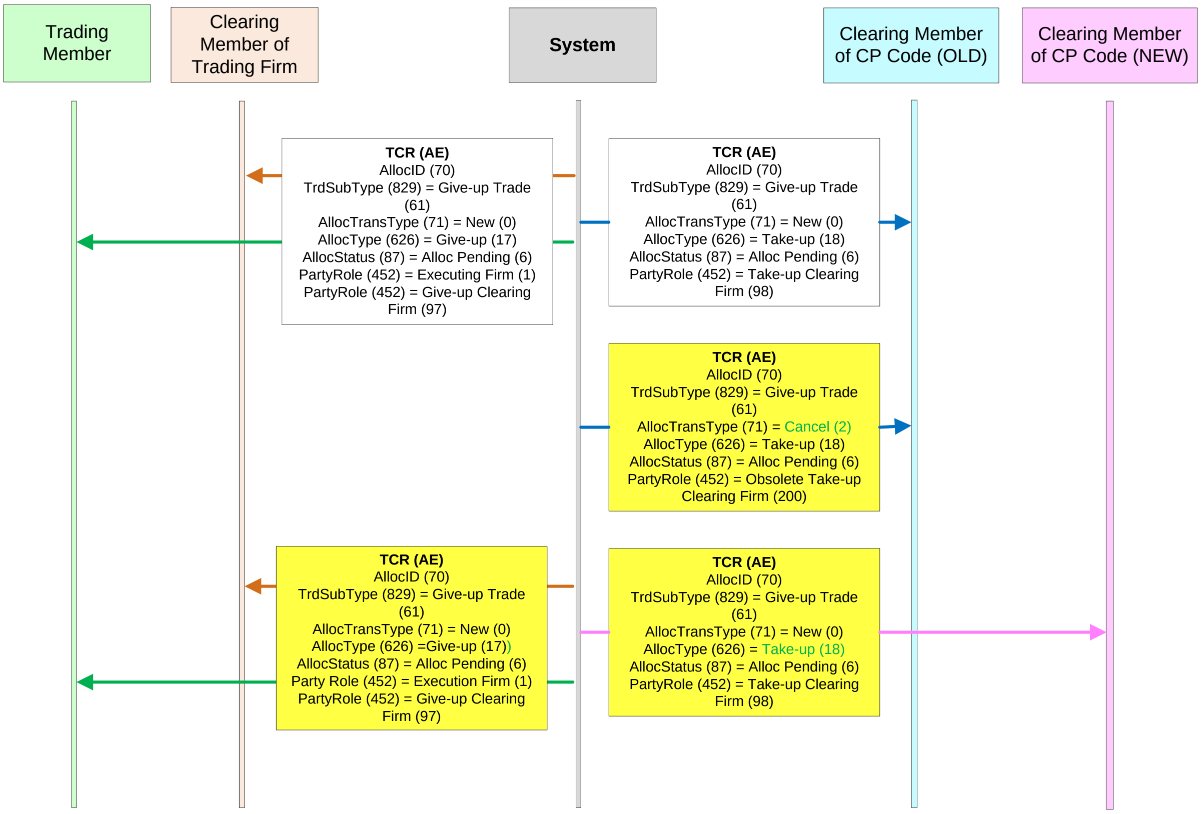
**Trade Capture Report Ack**
TradeID = Server 2
TrdRptStatus = Accepted
TradeReportType = Addendum
TradeReportTransType = New
MatchStatus = Matched
NoSides = 1
Side = Buy
Account = ABCD
AccountType = 1

**Trade Capture Report**
TradeReportID = Server TCR 20
TradeReportRefID = Server TCR 10
TradeID = Server 2
TradeHandlingInstr = Trade Confirmation
TradeReportTransType = Replace
TradeReportType = Addendum
MatchStatus = Matched
ExecType = Trade Correct

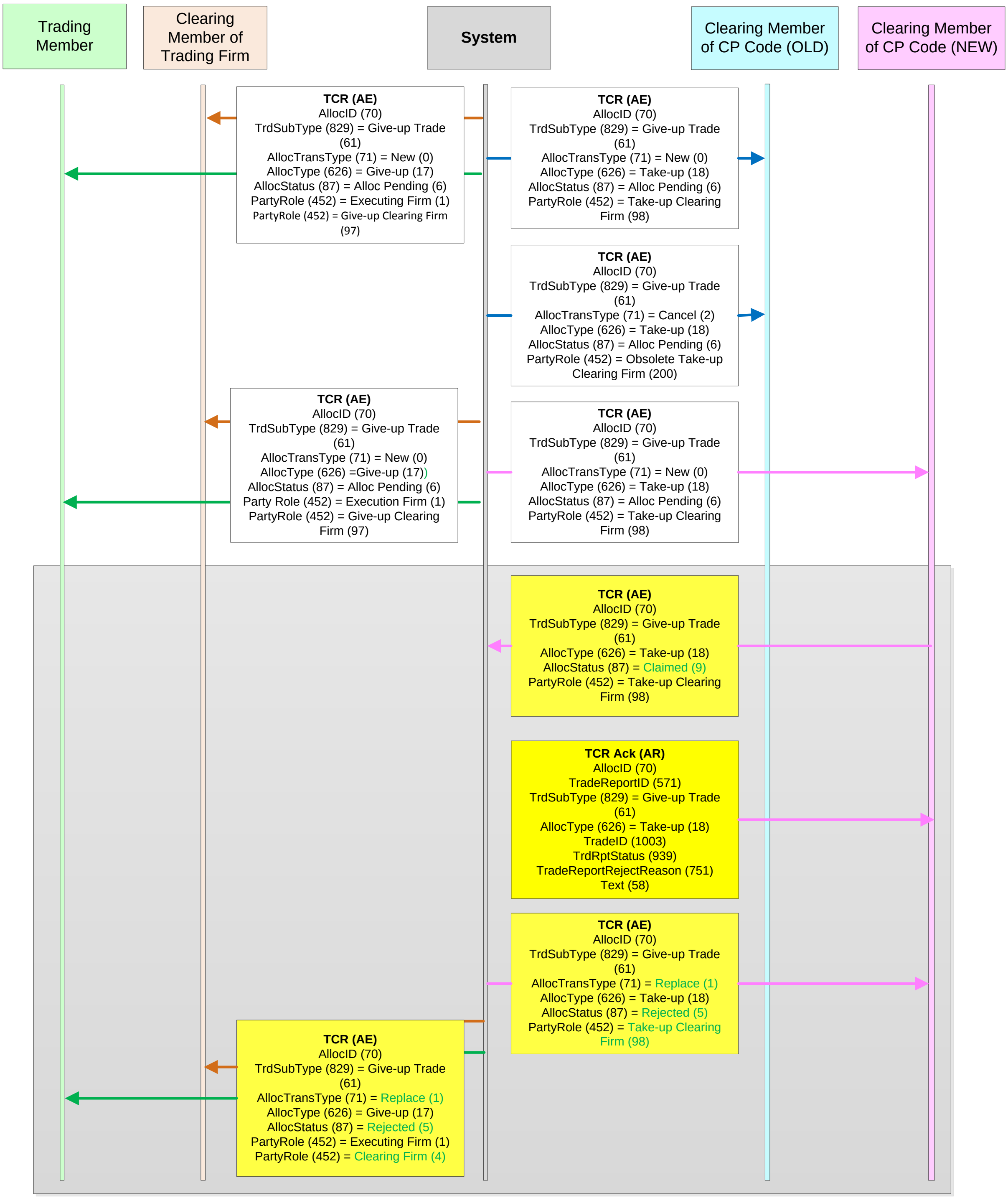
## 10.2 Give Up and Take Up Initiation


Trading Member
Clearing Member of Trading Firm
System
Clearing Member of CP Code (OLD)
Clearing Member of CP Code (NEW)
TCR (AE)
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Give-up (17)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Executing Firm (1)
PartyRole (452) = Give-up Clearing Firm (97)
TCR (AE)
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Take-up (18)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Take-up Clearing Firm (98)

## 10.3 Communicating a change in Take-Up Eligibility


Trading Member
Clearing Member of Trading Firm
System
Clearing Member of CP Code (OLD)
Clearing Member of CP Code (NEW)
TCR (AE)
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Give-up (17)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Executing Firm (1)
PartyRole (452) = Give-up Clearing Firm (97)
TCR (AE)
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Take-up (18)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Take-up Clearing Firm (98)
TCR (AE)
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = Cancel (2)
AllocType (626) = Take-up (18)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Obsolete Take-up Clearing Firm (200)
TCR (AE)
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) =Give-up (17))
AllocStatus (87) = Alloc Pending (6)
Party Role (452) = Execution Firm (1)
PartyRole (452) = Give-up Clearing Firm (97)
TCR (AE)
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Take-up (18)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Take-up Clearing Firm (98)

## 10.4 Take-up claim


Trading Member
Clearing Member of Trading Firm
System
Clearing Member of CP Code (OLD)
Clearing Member of CP Code (NEW)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Give-up (17)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Executing Firm (1)
PartyRole (452) = Give-up Clearing Firm (97)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Take-up (18)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Take-up Clearing Firm (98)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = Cancel (2)
AllocType (626) = Take-up (18)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Obsolete Take-up Clearing Firm (200)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) =Give-up (17))
AllocStatus (87) = Alloc Pending (6)
Party Role (452) = Execution Firm (1)
PartyRole (452) = Give-up Clearing Firm (97)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Take-up (18)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Take-up Clearing Firm (98)

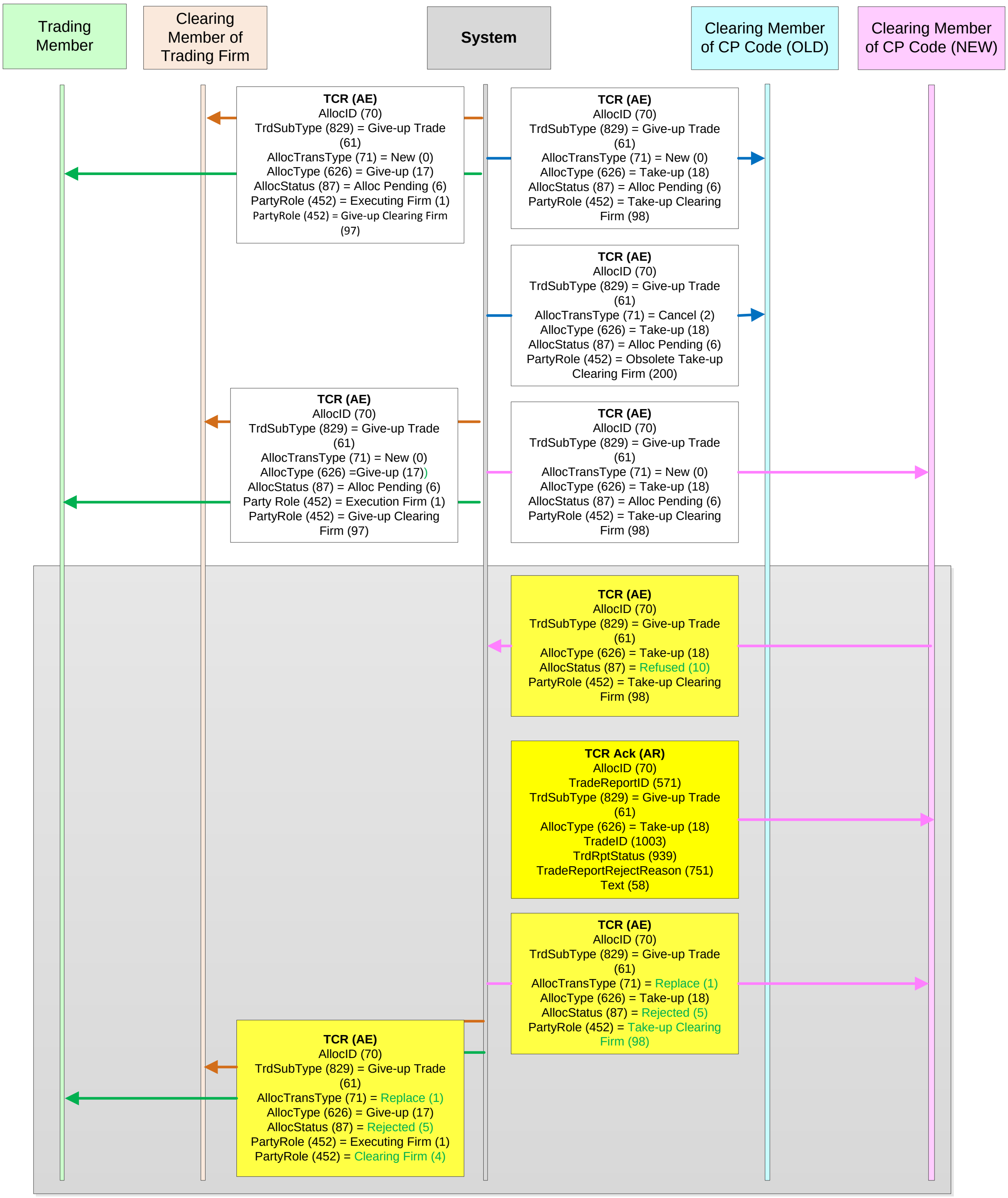
**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocType (626) = Take-up (18)
AllocStatus (87) = Claimed (9)
PartyRole (452) = Take-up Clearing Firm (98)

**TCR Ack (AR)**
AllocID (70)
TradeReportID (571)
TrdSubType (829) = Give-up Trade (61)
AllocType (626) = Take-up (18)
TradeID (1003)
TrdRptStatus (939)
TradeReportRejectReason (751)
Text (58)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = Replace (1)
AllocType (626) = Take-up (18)
AllocStatus (87) = Rejected (5)
PartyRole (452) = Take-up Clearing Firm (98)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = Replace (1)
AllocType (626) = Give-up (17)
AllocStatus (87) = Rejected (5)
PartyRole (452) = Executing Firm (1)
PartyRole (452) = Clearing Firm (4)

## 10.5 Take-up Refusal


Trading Member
Clearing Member of Trading Firm
System
Clearing Member of CP Code (OLD)
Clearing Member of CP Code (NEW)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Give-up (17)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Executing Firm (1)
PartyRole (452) = Give-up Clearing Firm (97)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Take-up (18)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Take-up Clearing Firm (98)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = Cancel (2)
AllocType (626) = Take-up (18)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Obsolete Take-up Clearing Firm (200)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) =Give-up (17))
AllocStatus (87) = Alloc Pending (6)
Party Role (452) = Execution Firm (1)
PartyRole (452) = Give-up Clearing Firm (97)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = New (0)
AllocType (626) = Take-up (18)
AllocStatus (87) = Alloc Pending (6)
PartyRole (452) = Take-up Clearing Firm (98)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocType (626) = Take-up (18)
AllocStatus (87) = Refused (10)
PartyRole (452) = Take-up Clearing Firm (98)

**TCR Ack (AR)**
AllocID (70)
TradeReportID (571)
TrdSubType (829) = Give-up Trade (61)
AllocType (626) = Take-up (18)
TradeID (1003)
TrdRptStatus (939)
TradeReportRejectReason (751)
Text (58)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = Replace (1)
AllocType (626) = Take-up (18)
AllocStatus (87) = Rejected (5)
PartyRole (452) = Take-up Clearing Firm (98)

**TCR (AE)**
AllocID (70)
TrdSubType (829) = Give-up Trade (61)
AllocTransType (71) = Replace (1)
AllocType (626) = Give-up (17)
AllocStatus (87) = Rejected (5)
PartyRole (452) = Executing Firm (1)
PartyRole (452) = Clearing Firm (4)

# 11 FUNCTIONAL AND IMPLEMENTATION LIMITATIONS

11.1.1.1 All the FIX gateways (FIX Trading, FIX Drop Copy and FIX Post Trade) currently use a common library. The system hence accepts all FIX messages defined for all three gateways, and cannot distinguish between them per gateway.

It will validate the incoming messages in the following sequence:

1. The system initially does a FIX library level validation
2. The system does a validation for required fields
3. The system finally does the Gateway level validation

11.1.1.2 Hence if a message is sent which does not comply with the specific gateway being used (but is defined in a different FIX gateway), it will validate the required fields. If any of the required tags are missing, it will give out a session reject with message "Required tag missing").

11.1.1.3 If a message is sent which does not comply with the specific gateway being used (but is defined in a different FIX gateway), it will validate the required fields. If all required fields are available, a gateway validation gives out a business reject message "Unsupported Message Type".

11.1.1.4 If a message is sent which does not comply with any of the FIX gateways used it will then give out a session reject message "Invalid Msg Type”.

11.1.1.5 If an undefined tag is sent along with any of the Administrative messages, the system will ignore the undefined tags and process the rest of the message. This is a limitation exists in the FIX library.

11.1.1.6 If Post Trade Gateway generates a Business Reject at a time the post trade process PTTPS is unavailable, required tag 45 (RefSeqNum) would not be returned. This is due to a technical limitation between PTTPS and Post Trade Gateway.