Irish Stock Exchange

ISE OBOE Release 1.0

FIX Gateway Specification

Version:	1.0.1
Interface Version:	OBOE-Engine_1.0-1

Publication Date 12th September 2017

Release Date 4th December 2017

FOREWORD BY THE IRISH STOCK EXCHANGE

ISE OBOE is the new MiFID-II compliant system for the reporting of off order book on exchange trades in ISE securities by member firms of the Irish Stock Exchange (ISE).

This version 1.0 of the FIX Gateway for the ISE OBOE system supersedes the preliminary version which was published in May. Members should use this version for their programming. Further changes to the specification are not planned. However if a change is identified it will be communicated to members as soon as possible.

Contents

1 Introduction

The ISE is launching a new MiFID II-compliant trade reporting system for the reporting of off order book on exchange deals in ISE instruments. The new system, 'ISE OBOE', will provide ISE member firms with the full suite of required MiFID II functionality for on exchange trade reporting. Existing ISE customisations, such as the specific ISE trade flags, will also be delivered together with some service enhancements.

The system is being provided as part of the ISE's successful strategic technology partnership with Deutsche Börse AG ('DBAG') with the new system powered by the Deutsche Börse 7 Market Technology. The core processing system for the service is the ISE OBOE MiFID II Engine, which will be used for the dissemination of data to the market based on straight-through processing of the member data provided through a FIX interface or a Web-based GUI.

ISE OBOE will launch on **4th December 2017**, which will ensure market readiness in advance of MiFID II implementation.

1.1 Purpose

The purpose of this document is to provide a functional and technical description of the FIX interface for the ISE OBOE system.

The description of the GUI functionality is detailed in a separate document.

1.2 Supported FIX Versions

Only FIX protocol versions 4.2 and 4.4 are supported.

The interface is a point-to-point service based on the technology and industry standards TCP/IP, FIX and FIX Session Protocol. The session and application event models and messages are based on versions 4.2 and 4.4 of the FIX protocol and are adapted to the MMT (Market Model Typology) standard.

FIXML will not be supported.

Following a FIX Protocol Limited (FPL) recommendation to use standard fields from higher versions as the primary solution before using user-defined fields, Deutsche Börse applies the following design rules for support of functionality currently not provided in the corresponding FIX version:

- Fields reserved for internal use (Tag numbers 10000 - 19999) are not used.
- Standard fields of the supported FIX versions that only became part of the standard message in a higher version are used.
- FIX fields of higher versions are only added to standard messages if no standard field for the required functionality is available in the supported FIX versions.

Characters in ASCII range 32-126 are allowed.

Leading and trailing spaces in string fields will be ignored and removed for the processing.

1.3 FIX Gap Analysis

Some enhancements are required for the implementation of the FIX interface for ISE OBOE.

For convenience part of the functionality will be implemented with features and field definitions that may differ from the FIX standard. It might be considered to integrate these changes in the FIX standard in the future.

Component <*TrdRegTimestamps*> enhanced by **adding:**

- Field *TrdRegTimestampSubType (28748)*

1.4 Readership

Detailed message formats and fields are provided and an outline is given for connection to the ISE OBOE FIX Gateway.

Requirements, assumptions, message layouts and the common message flows will be discussed.

All customers planning to use the FIX interface should thoroughly read the whole document.

Throughout this document the term “client” means a member firm of the ISE which has been approved by the ISE to use the ISE OBOE service provided by the Deutsche Börse.

1.5 Change Log

Date	Version	Description
11.05.2017	1.0	Creation
12.09.2017	1.0.1	Support of microseconds for the fields SendingTime (52) and OrigSendingTime (122)

2 Service Description

2.1 Overview of Supported Message Types

2.1.1 Administrative and Technical Messages

Message	Type	Description
Heartbeat	0	The Heartbeat message may be used by the client and the FIX Gateway to monitor the status of the communication link during periods of inactivity.
Test Request	1	The Test Request message is used to trigger a heartbeat message from the opposing application.
Resend Request	2	The Resend Request message is used by the client and the FIX Gateway to initiate the retransmission of messages in a recovery scenario.
Reject	3	The Reject message is used by the FIX Gateway when a message is received but cannot be properly processed due to a session-level rule violation.
Sequence Reset	4	The Sequence Reset message has two modes: Gap Fill mode is used in response to a Resend Request message when one or more messages must be skipped over. Reset mode specifies an arbitrarily higher new sequence number.
Logout	5	The Logout message initiates or confirms the termination of a FIX session. It is also used by the FIX Gateway to reject the FIX session logon.
Logon	A	The Logon message allows the client to connect to the FIX Gateway. It is also used by the FIX Gateway to confirm the logon.
Business Message Reject	j	The Business Message Reject message indicates that an application message has been rejected.
Trading Session Status	h	The Trading Session Status message informs about session related events.

2.1.2 Application Messages

Message	Type	Description
User Trade Capture Report	UAE	The User Trade Capture Report (UAE) message can be used by the clients to report trades (post-trade). The message will be also used to send information to the clients related to trade messages (GUI entries, status updates related to the different steps of a deferred publication). Only for FIX 4.2.
Trade Capture Report	AE	The Trade Capture Report (AE) message can be used by the clients to report trades (post-trade). The message will be also used to send information to the clients related to trade messages (GUI entries, status updates related to the different steps of a deferred publication). Only for FIX 4.4.
User Trade Capture Report Ack	UAR	The User Trade Capture Report Ack (UAR) message is the response to a User Trade Capture Report (UAE) request sent by the client for trade reporting (post-trade). Only for FIX 4.2.
Trade Capture Report Ack	AR	The Trade Capture Report Ack (AR) message is the response to a Trade Capture Report (AE) request sent by the client for trade reporting (post-trade). Only for FIX 4.4.

Additional information about the usage of the different application messages can be found in the next chapter, under “2.2.2 Scenarios at Application Level”.

2.2 Scenarios

2.2.1 Scenarios at Session Level

Scenario	Incoming Message	Outgoing Message
Technical Requests		
Logon	Logon (A)	Positive: Logon (A) Negative: Logout (5)
Logon with password change	Logon (A)	Positive: Logon (A) Negative: Logout (5)
Logout	Logout (5)	Logout (5)
System Generated Logout		
Logout	-	Logout (5)
General Rejections (possible for every request)		
Session level reject	All Request Messages	Reject (3)
General rejection through FIX Gateway	All Request Messages	Business Message Reject (j)

2.2.2 Scenarios at Application Level

2.2.2.1 Processing of FIX requests

Scenario	Incoming Message	Outgoing Message
Trade Reporting		
Enter, Amendment or Cancellation Trade Report	User / Trade Capture Report (UAE/AE)	User / Trade Capture Report Ack (UAR/AR) In case of a successful Amendment two response messages will be generated by the ISE OBOE MiFID II Engine and forwarded to the FIX client: - First message (cancellation of the original report) with TrdRptStatus (939) = 6 (Pending Replace), Source (5177) = OBOE_ENGINE - Second message (amendment) with TrdRptStatus (939) = 0 (Accepted)

2.2.2.2 Input via external interface (GUI)

Scenario	Incoming Message	Outgoing Message
Trade Reporting		
Enter, Amendment or Cancellation Trade Report	-	User / Trade Capture Report (UAE/AE) InputSource (979) = GUI For Amendments two messages will be generated by the ISE OBOE MiFID II Engine and forwarded to the FIX client: - First message (cancellation of the original report) with TrdRptStatus (939) = 6 (Pending Replace), Source (5177) = OBOE_ENGINE - Second message (amendment) with TrdRptStatus (939) = 0 (Accepted)

2.2.2.3 Status updates

Status Updates are sent in case there is a status change in the processing of the submitted transaction related to the deferred publication (see scenarios below).

Status Updates will be delivered via *User / Trade Capture Reports (UAE/AE)* with *MessageEventSource (1011) = 100 (Actual publication after deferral)* and *InputSource (979) = OBOE_ENGINE*.

Scenarios	Incoming Message	Outgoing Message
Trade Reporting		
Actual publication after standard deferral	-	User / Trade Capture Report (UAE/AE) InputSource (979) = OBOE_ENGINE MessageEventSource (1011) = 100 (Actual publication after deferral)

2.2.2.4 System availability

Scenario	Incoming Message	Outgoing Message
ISE OBOE MiFID II Engine Connection Availability		
ISE OBOE MiFID II Engine Connection available	-	Trading Session Status (h) (Open)
ISE OBOE MiFID II Engine Connection unavailable	-	Trading Session Status (h) (Halted)
System Shutdown		
System Shutdown initiated	-	Trading Session Status (h) (Pre-closed)
System Shutdown finished	-	Trading Session Status (h) (Closed)
Logout	-	Logout (5)

2.3 Reporting Functionality

2.3.1 Trade Reporting

2.3.1.1 Message identification criteria

Action Flags

The trade reporting processing depends on the action flag set in the trade messages. Possible action flags are *ENTR* (entry), *CANC* (cancellation) and *AMND* (amendment).

These can be set in the FIX messages using different values of the field *TradeReportTransType (487):*

- *ENTR* (entry) — *TradeReportTransType (487) = 0 (New)*
- *CANC* (cancellation) — *TradeReportTransType (487) = 1 (Cancel)*
- *AMND* (amendment) — *TradeReportTransType (487) = 2 (Replace)*

Identification of a Trade

Following field will be used for the identification of a trade:

- *TradeID (1003)* — Unique trade identification number assigned by the ISE OBOE MiFID II Engine when a trade entry is processed successfully. Its value will not change after an amendment.

When a report is entered the client can define an own identifier (client reference) in the field *FirmTradeID (1041)*. If the entry is successful a unique trade identification number is assigned by the ISE OBOE MiFID II Engine and delivered in the field *TradeID (1003)*.

In case of amendments and cancellations, the *TradeID (1003)* must be used in the FIX requests to identify the trade.

The usage of the client reference (*FirmTradeID (1041)*) as identifier for amendments and cancellations will not be supported.

Amendments

Amendments are processed in two steps:

- In a first step the existing report is cancelled.
- In a second step an amendment record with the new data, but maintaining the previous trade identification number (*TradeID (1003)*), is generated.

The FIX messages generated by the FIX Gateway for a successful amendment will be identified as follows:

Cancel of the existing report	*TradeReportTransType (487)* *TrdRptStatus (939)*	*= 2 (Replace)* *= 6 (Pending Replace)*
Amendment record with new data	*TradeReportTransType (487)* *TrdRptStatus (939)*	*= 2 (Replace)* *= 0 (Accepted)*

The following table summarizes the usage of the fields used as identification criteria for the supported messages:

Scenario	TradeReportTransType (487)	TradeID (1003)
User / Trade Capture Report (UAE/AE)		
Enter	0 (New)	FIX Requests Not set Enter via GUI / Status Updates TradeID generated by the ISE OBOE MiFID II Engine
Cancellation	1 (Cancel)	FIX Requests TradeID from report entry (mandatory) Cancellations via GUI / Status Updates TradeID from report entry
Amendment	2 (Replace)	FIX Requests TradeID from report entry (mandatory) Amendments via GUI / Status Updates TradeID from report entry
User / Trade Capture Report Ack (UAR/AR)		
Enter	0 (New)	TradeID generated by the ISE OBOE MiFID II Engine. Not set for rejections.
Cancellation	1 (Cancel)	TradeID from request
Amendment	2 (Replace)	TradeID from request

2.3.1.2 Additional message identifiers and processing flags

All messages contain the field *TradeReportID (571)*, which is filled with a unique value.

The FIX client must send the field *TradeReportID (571)* in all the FIX requests. The uniqueness must be ensured by the FIX client, there is no check within the FIX Gateway. The field *TradeReportID (571)* in the FIX request message is returned back with the FIX response message.

For messages related to GUI entries and status updates a unique value for the field *TradeReportID (571)* will be assigned by the FIX Gateway.

The client can set the field *ExchangeSpecialInstructions (1139)* of the component <*TrdCapRptSidGrp*> to publish a trade immediately regardless of the result of the deferral calculation:

ExchangeSpecialInstructions (1139)	Description
NODEF	Publish a trade immediately regardless of the result of the deferral calculation.

The result of the FIX request processing is delivered in the FIX response message in the field *TrdRptStatus (939)*. Following values are possible:

- *0 (Accepted)* Request successfully processed. Additional details (Information, Warning, Alert) can be delivered in the component <*StatusInformationGrp*>.
- *1 (Rejected)* Request rejected. Error information delivered in *TradeReportRejectReason (751), RejectText (1328)* and *Source (5177).*
- *4, 5, 6 (Pending)* Request processing pending, final confirmation still open (see details in "2.3.2 Pending Messages").

FIX messages sent to the client for reports successfully processed will contain information about the publication status in the field *TradePublishIndicator (1390)*:

- *1 = Publish Trade* Trade was published without deferral
- *2 = Deferred Publication* Trade publication deferred

2.3.1.3 Message identifiers for status updates

If the trade publication is deferred additional messages (status updates) will be generated after the actual publication of the report.

Each status update message will be a *User / Trade Capture Report (UAE/AE)* with the data of the original report and following attributes for the identification of the message:

- InputSource (979) = OBOE_ENGINE
- MessageEventSource (1011) = 100 (Actual publication after deferral)
- TradePublishIndicator (1390) = 2 (Deferred Publication)
- <publication date and time> = Publication date and time

2.3.2 Pending Messages

2.3.2.1 Identification of pending messages

The FIX Gateway will generate pending messages in different scenarios (see next chapters).

Pending responses will be identified as follows:

- The FIX Gateway will send a *User / Trade Capture Report Ack (UAR/AR)* with *TrdRptStatus (939) = 4 (Pending New), 5 (Pending Cancel)* or *6 (Pending Replace)*.
- Depending on the scenario the field *Source (5177)* in the response message will contain *FIX_GATEWAY* or *OBOE_ENGINE*.

For trade amendments via GUI pending messages will be also generated. These will be identified as follows:

- The FIX Gateway will send a *User / Trade Capture Report (UAE/AE)* with *TrdRptStatus (939) = 6 (Pending Replace)*.

2.3.2.2 Trade amendments

As described in "2.3.1.1 Message identification criteria" a trade amendment is processed in two steps:

- In a first step, the existing report is cancelled.
- In a second step an amendment record with the new data, but maintaining the previous trade identification number (*TradeID (1003)*), is generated.

For the first step a pending message will be generated. For requests entered via FIX this will be a *User / Trade Capture Report Ack (UAR/AR)*, for amendments via GUI a *User / Trade Capture Report (UAE/AE)*.

In both cases, the generated message will contain following fields*:*

- *TradeReportTransType (487) = 2 (Replace)*
- *TrdRptStatus (939) = 6 (Pending Replace)*

The message *User / Trade Capture Report Ack (UAR/AR)* will contain additionally *Source (5177) = OBOE_ENGINE* to distinguish this pending response from pending responses generated by the FIX Gateway.

2.3.2.3 Request status unknown

A pending FIX response will be generated when the status of a FIX request is unknown. For example:

- If no response from the ISE OBOE MiFID II Engine is received within a certain time (currently one minute).
- In recovery situations for FIX requests with *PossDup* = *"Y"*, for which the report status cannot be determined by the FIX Gateway.

In all these cases the pending messages will contain *Source (5177)* = *FIX_GATEWAY*.

If a pending message indicating that the status of a FIX request is unknown is received, clients are requested to check the status of the report via GUI.

2.3.3 Instrument Identification

For the Instrument identification the component <*Instrument*> will be used.

The International Securities Identification Number (ISIN) must be used as instrument identifier.

The field *SecurityIDSource (22)* must contain the value *4 (ISIN)* and the ISIN will be set in the field *SecurityID (48).*

2.3.4 Party Identification

The *Parties* component block will be used to describe all parties participating in a transaction. For each party a separate occurrence of the repeating group will be set up.

2.4 Drop Copy Functionality for GUI Entries

The reception of drop copy messages via FIX for data entered via GUI is provided as an optional feature of the FIX sessions.

When the client chooses the drop copy feature for a FIX session in the Member Section, the information about successful GUI entries for the corresponding Legal Entity Identifier (LEI), including entries generated from a file upload, will be sent to the related FIX session.

3 Connectivity and Session Parameters

As per definition by the FIX Protocol standard, a FIX session is defined as a bi-directional stream of ordered messages between two parties within a continuous sequence number.

A FIX session will be initiated by the participant and maintained between the participant and the FIX Gateway over the course of a business day.

3.1 Session Concept

FIX sessions can be requested via Member Section. For each FIX session a *SenderCompID (49)* (unique client identifier) and a *Password (554)* is assigned on registration. A participant may have multiple FIX sessions (connections to the FIX Gateway).

For security reasons a *Password (554)* must be specified in the *Logon (A)* message. The initial password assigned for each FIX session should be changed during the first logon by specifying *NewPassword (925)* in the *Logon (A)* message.

All messages sent to the FIX Gateway must contain the assigned unique client identifier of the FIX session in the field *SenderCompID (49)* and "*OBOE_ENGINE*" in the field *TargetCompID (56)*.

All messages sent by the FIX Gateway to the client will contain "*OBOE_ENGINE*" in the field *SenderCompID (49)* and the assigned unique client identifier of the FIX session in the field *TargetCompID (56)*.

3.2 Network Authentication

The FIX Gateway will validate the subnet from where the FIX session is initiated during session logon.

The FIX session logon (*Logon (A)* message) will be rejected by the FIX Gateway if the subnet cannot be authenticated.

Participants are allowed to initiate/resume their FIX sessions from alternate locations, e.g., a backup site or disaster recovery location.

The FIX Gateway permits the setup of up to four IP subnet addresses via the Member Section.

3.3 Session Logon

The *Logon (A)* message authenticates a FIX session and establishes a connection to the FIX Gateway.

This message must be the first one sent by the client and will be validated by the FIX Gateway. A successful logon will initiate a FIX session.

The FIX Gateway does not support encryption. *EncryptMethod (98)* must therefore be set to *"0"*.

As an additional safeguard measure, the *TestMessageIndicator (464)* is used to indicate whether a FIX session to be initiated will be used for Simulation or Production purposes. The FIX Gateway will reject a *Logon (A)* message in the event that the *TestMessageIndicator (464)* value does not match the target environment.

3.4 IP Addresses and Ports

The FIX connection between a client infrastructure and the FIX Gateway service is established via a TCP/IP connection.

The service comprises the access to a FIX Gateway in each environment (Simulation and Production) with dedicated IP addresses and port numbers.

For each FIX session, an individual port number is assigned for each environment and communicated to the client together with the FIX Gateway IP address to connect to the respective environment.

The client is free to define its own source addresses as long as they match one of the IP subnet addresses entered during the registration of the FIX session.

3.5 Failover

The FIX Gateway service features a redundant setup of all components to provide a high level of availability and fault tolerance.

The failover functionality in FIX Gateway is implemented via a cluster solution. In case of failure customer sessions will be disconnected and the backup system will be activated automatically.

After a failover the clients can connect to the FIX Gateway using the individual connection parameters (IP address and a port number) assigned during the registration of the FIX session.

4 Session Layer

4.1 Sequence Number

All FIX messages are identified by a unique sequence number.

Sequence numbers are reset by the FIX Gateway during down time after the end of each business day. The same behaviour is expected for the FIX engine on the client side.

Sequence numbers sent by the client which are behind sequence expected will trigger a logout and TCP connection close by the FIX Gateway.

Sequence numbers ahead of sequence will trigger a message recovery by the FIX Gateway via the *Resend Request (2)* message.

4.2 Logon

The *Logon (A)* message is the first message the client needs to send after the TCP connection has been established. No encryption is supported by the FIX Gateway.

As the first message for the day the client should send a *Logon (A)* message with sequence number 1.

A FIX session is identified by the fields *SenderCompID (49)*, *TargetCompID (56)* and *BeginString (8)* in the message header.

TestMessageIndicator (464) and *Password (554)* are validated during the session logon. If validation fails, the FIX Gateway will send a *Logout (5)* message specifying the reason for the rejection followed by the termination of the TCP connection.

Note: if validation during session logon has failed, the sequence number will not be reset.

In the event of an intra-day restart the *Logon (A)* response message may provide a sequence number higher than expected by the client. This would indicate that messages were missed. The client should send a *Resend Request (2)* message to trigger retransmission of the missed messages.

Logon requests with *ResetSeqNumFlag (141)* set to *"Y"* will trigger a reset of sequence numbers at the participant side only. The FIX Gateway's sequence numbering will remain unchanged. Thus the customer is able to access all messages disseminated by the FIX Gateway.

If a FIX session is successfully logged on subsequent *Logon (A)* messages will be discarded.

4.3 Logout

The *Logout (5)* message is used by the client to gracefully close the FIX session. Messages need to be processed normally by the client until the FIX Gateway sends the logout confirmation.

The FIX Gateway will also send a *Logout (5)* message if validation fails for a FIX session logon. The reason for the rejection is specified in *SessionStatus (1409)*. The *Logout (5)* message is followed by a close of the TCP connection.

4.4 Reject

Session level rejects are used by the FIX Gateway to indicate violations of the session protocol, missing fields or invalid values.

4.5 Heartbeat

The *HeartBtInt (108)* has to be specified by the client during the FIX session logon.

A *Heartbeat (0)* message should be sent by the client if no other message has been processed during the defined *HeartBtInt (108)* interval.

4.6 Test Request

A *Test Request (1)* message should be sent if no in-sequence message has been received for more than the heartbeat interval. If no in-sequence message is received after that for more than the heartbeat interval, the TCP connection should be closed.

4.7 Resend Request

A *Resend Request (2)* message initiates the retransmission of missed messages and can be used if a sequence number gap has been detected. A *Resend Request (2)* message needs to be processed even if it is ahead of sequence.

The *PossDupFlag (43)* field set to "Y" in the Message Header indicates that a FIX engine is repeating transmission of already sent content (including *MsgSeqNum (34)*).

The FIX Gateway supports open or closed sequence range in a *Resend Request (2)* message (an open range is indicated by sequence number zero as the *EndSeqNo (16)*).

Note: No Gap Fill messages should be sent by the client during the resend series for application messages. Application messages should always be re-transmitted since the FIX Gateway requires all missed application messages for the purpose of reconciliation.

4.8 Sequence Reset

Two types of *Sequence Reset (4)* message are supported: Gap Fill mode and Reset mode.

4.8.1 Gap Fill Mode

This type of *Sequence Reset (4)* message is the response to a *Resend Request (2)* message.

Gap Fill mode is indicated by the field *GapFillFlag (123) = "Y"*.

All gap fill messages should have *PossDupFlag (43) = "Y"* in the Message Header.

Gap Fill mode should only be used for administrative messages.

4.8.2 Reset Mode

This type of *Sequence Reset (4)* message may be used by the client in emergency scenarios where all means of automatic recovery are lost (e.g. in case of an unrecoverable application failure).

Reset mode is indicated if *GapFillFlag (123) = "N"* or if the field is omitted.

After a Reset mode has been triggered, the *Test Request (1)* message should be used by the participant to verify that the requested reset has been accepted by the FIX Gateway.

4.9 Recovery

When a client reconnects after a FIX session disconnection during the same business day, two different scenarios can be identified as a reason for the outage: outage on the client side and outage on FIX Gateway side.

4.9.1 Outage on the Client Side

After resuming the FIX session, the client may have missed some messages from the FIX Gateway. In this case, the sequence number of the next message received from the FIX Gateway will be ahead of the last *MsgSeqNum (34)* stored on the client side.

The client should send a *Resend Request (2)* message in order to trigger all missed messages during the outage.

The FIX Gateway will return all potentially missed messages with *PossDupFlag (43)* = *"Y"* to indicate that a message may have been previously transmitted with the same *MsgSeqNum (34)*.

4.9.2 Outage on the FIX Gateway Side

After reconnection of the FIX session, the FIX Gateway may receive a sequence number higher than the one expected and sends a *Resend Request (2)* message to the client.

The client should resend all potentially missed messages with *PossDupFlag (43)* = *"Y"* to indicate that a message may have been previously transmitted with the same *MsgSeqNum (34)*. The FIX Gateway will send responses to previously transmitted messages with *PossResend (97)* = *"Y"*.

No Gap Fill messages should be sent by the client during the resend series for application messages. Application messages should always be re-transmitted since the FIX Gateway requires all missed application messages for the purpose of reconciliation.

4.10 System Availability

4.10.1 ISE OBOE MiFID II Engine Connection Availability

The FIX Gateway offers an interface for the reporting of trades, but the reporting will be only possible if the connection to the ISE OBOE MiFID II Engine is available.

Report messages will be rejected by the FIX Gateway if the connection to the ISE OBOE MiFID II Engine is not available.

The FIX Gateway will generate *Trading Session Status (h)* messages to inform the customers about the availability of the connection to the ISE OBOE MiFID II Engine. Customers will be informed about the current status after each FIX logon and additionally *Trading Session Status (h)* messages will be also generated by each status change:

- The availability of the connection to the ISE OBOE MiFID II Engine will be indicated by a *Trading Session Status (h)* message with *TradSesStatus (340)* set to *2 = "Open"*.
- The unavailability of the connection to the ISE OBOE MiFID II Engine will be indicated by a *Trading Session Status (h)* message with *TradSesStatus (340)* set to *1 = "Halted".*

4.10.2 System Shutdown

From the customer's point of view an incoming *Trading Session Status (h)* message where *TradSesStatus (340)* is set to *5 = "Pre-Close"* indicates that a system shutdown has been initiated by the FIX Gateway.

No more incoming requests will be accepted, but outgoing messages waiting for delivery will still be transmitted.

The ISE OBOE MiFID II Engine may also generate further messages, which will also be distributed.

As soon as the ISE OBOE MiFID II Engine has finished to do so, the FIX Gateway will disseminate a *Trading Session Status (h)* message where status is set to *3 = "Closed"* and – after a short delay – a *Logout (5)* message for every active connection will be sent. Then all sessions/connections will be terminated unconditionally.

5 Message Formats

This chapter provides details about the messages used by the FIX Gateway.

The structure of the header and trailer as well as details on the fields and components used in application messages will be described in the next chapters.

The column “R” (Required) in the tables of the next chapters describes if the related field or component is mandatory (“M”) or optional (“O”).

Messages sent by clients not listed in this section are rejected by the server via a *Reject (3)* message or *BusinessMessageReject (j)* message.

5.1 Message Header and Trailer

5.1.1 Message Standard Header

Tag	Field Name	R	Description
<Begin StandardHeader>			Standard FIX message header.
8	BeginString	M	*String* Identifies beginning of new message and protocol version. Valid Values and Description: FIX.4.4 = Version 4.4 FIX.4.2 = Version 4.2 Must always be the first field in the header structure.
9	BodyLength	M	*Length* Message length, in bytes, forward to the CheckSum field. Must always be the second field in the header structure.
35	MsgType	M	*String* Defines the message type. Must conform to the set of valid messages types. Must always be the third field in the header structure.
34	MsgSeqNum	M	*SeqNum* Integer message sequence number.
43	PossDupFlag	O	*Boolean* Indicates possible retransmission of message with this sequence number. Valid Values and Description: N = Original transmission Y = Possible duplicate
49	SenderCompID	M	*String* Assigned identifier of the party sending the message. Will be “OBOE_ENGINE” for messages sent to client.

Tag	Field Name	R	Description
52	SendingTime	M	*UTCTimestamp* Time of message transmission. Supplied by the sending system. Following formats are supported: - YYYYMMDD-HH:MM:SS - YYYYMMDD-HH:MM:SS.sss - YYYYMMDD-HH:MM:SS.ssssss
56	TargetCompID	M	*String* Assigned identifier of the party receiving the message. Must be “OBOE_ENGINE” for messages sent by the client.
97	PossResend	O	*Boolean* Indicates that message may contain information that has been sent under another sequence number. Valid Value and Description: N = Original transmission Y = Possible resend Requests with PossResend (97) = “Y” (Possible Resend) will be rejected.
122	OrigSendingTime	O	*UTCTimestamp* Required if PossDupFlag (43) = “Y”. Following formats are supported: - YYYYMMDD-HH:MM:SS - YYYYMMDD-HH:MM:SS.sss - YYYYMMDD-HH:MM:SS.ssssss The FIX Gateway ignores the OrigSendingTime (122) in all message types.
369	LastMsgSeqNumProcessed	O	*SeqNum* The last MsgSeqNum (34) value received by the FIX engine and processed by downstream application. Can be specified on every message sent. Useful for detecting a backlog with a counterparty. This field will be ignored by the FIX Gateway.
<End StandardHeader>			

5.1.2 Message Standard Trailer

Tag	Field Name	R	Description
<Begin StandardTrailer>			Standard FIX message trailer.
10	CheckSum	M	*String* – 3 Bytes. Three byte, simple checksum.
<End StandardTrailer>			

5.2 Administrative and Technical Messages

5.2.1 Logon

The *Logon (A)* message allows the client to connect to the FIX Gateway. It is also used by the FIX Gateway to confirm the logon.

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = A (Logon)
98	EncryptMethod	M	*Integer* Method of encryption. Valid Value and Description: 0 = None/other Will be validated and then be ignored.
108	HeartBtInt	M	*Integer* Heartbeat interval (seconds).
141	ResetSeqNumFlag	O	*Boolean* Used to indicate whether the both sides of the FIX session should reset sequence numbers. Valid Values and Description: N = No Y = Yes, reset sequence numbers
383	MaxMessageSize	O	*Length* Maximum number of bytes supported for a single message. Will be validated and then be ignored.
464	TestMessageIndicator	O	*Boolean* Indicates that this FIX session will be sending and receiving “test” vs. “production” messages. Valid Values and Description: N = False (Production) Y = True (Simulation) This field is required in the messages sent to the FIX Gateway.
554	Password	O	*String* Password. This field is required in the messages sent to the FIX Gateway.
789	NextExpectedMsgSeqNum	O	*SeqNum* Next expected MsgSeqNum value to be received. Will be validated and then be ignored.
925	NewPassword	O	*String* New password or passphrase.
1408	DefaultCstmApplVerID	O	*String* Most recent version number of the FIX Gateway interface.
<StandardTrailer>			

5.2.2 Logout

The *Logout (5)* message initiates or confirms the termination of a FIX session. It is also used by the FIX Gateway to reject the FIX session.

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = 5 (Logout)
58	Text	O	*String* Additional explanation why the message was sent.
1409	SessionStatus	O	*Integer* Session status. Valid Values and Description: 4 = Session logout complete 5 = Invalid user name or password
<StandardTrailer>			

5.2.3 Heartbeat

The *Heartbeat (0)* message may be used by the client and the FIX Gateway to monitor the status of the communication link during periods of inactivity.

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = 0 (Heartbeat)
112	TestReqID	O	*String* Identifier included in the Test Request (1) message. Required if the heartbeat is a response to a Test Request (1) message.
<StandardTrailer>			

5.2.4 Test Request

The *Test Request (1)* message is used to trigger a heartbeat message from the opposing application.

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = 1 (Test Request)
112	TestReqID	M	*String* Identifier included in the Test Request (1) message. Required in the Heartbeat (0) message if the heartbeat is a response to a Test Request (1) message.
<StandardTrailer>			

5.2.5 Resend Request

The *Resend Request (2)* message is used by the client and the FIX Gateway to initiate the retransmission of messages in a recovery scenario.

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = 2 (Resend Request)
7	BeginSeqNo	M	*SeqNum* Message sequence number of first message in range to be resent.
16	EndSeqNo	M	*SeqNum* Message sequence number of last message in range to be resent.
<StandardTrailer>			

5.2.6 Sequence Reset

The *Sequence Reset (4)* message has two modes: Gap Fill mode is used in response to a *Resend Request (2)* message when one or more messages must be skipped over. Reset mode specifies an arbitrarily higher new sequence number after an unrecoverable application failure.

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = 4 (Sequence Reset)
36	NewSeqNo	M	*SeqNum* New sequence number.
123	GapFillFlag	O	*Boolean* Indicates that the Sequence Reset (4) message is replacing administrative or application messages which will not be resent. Valid Values and Description: N = Sequence Reset, Ignore Msg Seq Num Y = Gap Fill Message, Msg Seq Num Filed Valid
<StandardTrailer>			

5.2.7 Reject

The *Reject (3)* message is used by the FIX Gateway when a message is received but cannot be properly processed due to a session-level rule violation.

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = 3 (Reject)
45	RefSeqNum	M	*SeqNum* MsgSeqNum of rejected message.
58	Text	O	*String* Error text.
371	RefTagID	O	*Integer* Tag number of the FIX field being referenced.
372	RefMsgType	O	*String* MsgType (35) of the FIX message being referenced.
373	SessionRejectReason	O	*Integer* Reject reason. Reason for a session-level Reject message. Valid Values and Description: 0 = Invalid tag number 1 = Required tag missing 2 = Tag not defined for this message type 3 = Undefined tag 4 = Tag specified without a value 5 = Value is incorrect for this tag 6 = Incorrect data format for value 7 = Decryption problem 8 = Signature problem 9 = CompID problem 10 = SendingTime accuracy problem 11 = Invalid MsgType 12 = XML Validation Error 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 17 = Non “data” value includes field delimiter 18 = Invalid/Unsupported Application Version 99 = Other
5177	Source	O	*String* Will identify the component providing the reject information. Valid Value and Description: FIX_GATEWAY = FIX Gateway
5555	ReturnCode	O	*Integer* Error code.
<StandardTrailer>			

5.2.8 Business Message Reject

The *Business Message Reject (j)* indicates that an application message has been rejected.

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = j (Business Message Reject)
45	RefSeqNum	O	*SeqNum* MsgSeqNum of rejected message.
58	Text	O	*String* Error text.
372	RefMsgType	M	*String* MsgType (35) of the FIX message being referenced.
379	BusinessRejectRefID	O	*String* ID of the message being referenced. TradeReportID (571) from request for rejections of post-trade requests.
380	BusinessRejectReason	M	*Integer* Code to identify reason for a Business Message Reject. Valid Values and Description: 0 = Other 1 = Unknown ID 3 = Unsupported Message Type 4 = Application not available 5 = Conditionally required field missing 6 = Not authorized
5177	Source	O	*String* Will identify the component generating the return code. Valid Value and Description: FIX_GATEWAY = FIX Gateway
5555	ReturnCode	O	*Integer* Error code.
<StandardTrailer>			

5.2.9 Trading Session Status

The *Trading Session Status (h)* message informs about session related events.

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = h (Trading Session Status)
58	Text	O	*String* Text.
60	TransactTime	O	*UTCTimestamp* Transaction time. The FIX Gateway will set the value in following format: - YYYYMMDD-HH:MM:SS.sss
336	TradingSessionID	M	*String* Identifier for Trading Session. Valid Value and Description: 1 = Day
340	TradSesStatus	M	*Integer* State of the trading session. Valid Values and Description: 1 = Halted 2 = Open 3 = Closed 5 = Pre-Close
<StandardTrailer>			

5.3 Application Messages: Trade Reporting

5.3.1 User / Trade Capture Report (UAE/AE)

The *User / Trade Capture Report (UAE/AE)* message can be used by the clients to report trades (post-trade).

The message will be also used to send information to the clients related to trade messages (GUI entries and status updates).

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = UAE/AE (User / Trade Capture Report)
<Instrument>		M	Financial Instrument.
<StatusInformationGrp>		O	Status information (Information, Warning, Alert). Not allowed in messages from client.
<TrdRegPublicationGrp>		O	Publication Reasons. Not allowed if the flag “ISE Ordinary Trade” (OT) is set (PreviouslyReported (570) = “Y”).
<Begin TrdCapRptSideGrp>		M	Side-specific information items of a trade capture report message.
552	NoSides	M	*NumInGroup* Number of Side repeating group instances. Valid Value and Description: 1 = One Side
54	Side	M	*Char* Side. Valid Values and Description: 1 = Buy 2 = Sell
<Begin Parties>		M	Party Information.
<entering firm>		M	Member LEI. The LEI of the owner of the FIX session.
<executing firm>		M	Executing LEI. Owner of the trade.
<entering trader>		M	User ID The trader/user of the ISE member submitting the trade report. Must be filled with a valid user ID of the entering ISE member, as setup in the Member Portal.
<counterparty>		M	Counterparty Description of the counterparty. The field can contain a LEI or a natural person.
<End Parties>			
15	Currency	M	*Currency* Identifies currency used for price. Denomination Currency according to the ISO 4217 standard.

Tag	Field Name	R	Description
29	LastCapacity	M	*Char* ISE Trade Capacity flag. Valid Values and Description: 1 = Deal on own account (DEAL) 2 = Any other capacity (AOTC)
1139	ExchangeSpecialInstruc-tions	O	*String* The field can be used to publish a trade immedi-ately, regardless of the result of the deferral calcu-lation. Valid Values and Description: NODEF = Skip check for deferred publication
<End TrdCapRptSideGrp>			
<Begin TrdRegTimestamps>		M	Timestamps.
<trade date and time>		M	Trade date and time when the trade was executed by the involved parties. It must not be in the future, but there are no limits to the past. In case of an amendment trade date and time must be identical to the initial trade report message.
<publication date and time>		O	Publication date and time. The field will be set for GUI entries that have been published without deferral (TradePublishIndicator (1390) = “1”) and for status updates related to the deferred publication (TradePublishIndicator (1390) = “2”, MessageEventSource (1011) set). Not allowed in messages from client.
<preliminary publication date and time>		O	Preliminary publication date and time. In case of a deferred publication, this field contains the preliminary publication date and time (result of the deferral calculation). Not allowed in messages from client.
<End TrdRegTimestamps>			
<Begin TradePriceConditionGrp>		O	Trade Price Conditions.
<trade flag sdiv>		O	Trade flag information. See “5.3.3 Input Trade Flags”.
<End TradePriceConditionGrp>			
30	LastMkt	M	*Exchange* Identification of the ISE market segment for which the trade was reported (segment MIC). Valid Values and Description: XMSM = Main Securities Market XESM = Enterprise Securities Market XATL = Atlantic Securities Market

Tag	Field Name	R	Description
31	LastPx	M	*Price* Price of the instrument within a report trade (> 0 (zero)). 4 decimal places to be maintained system-wide. Up to 20 numerical digits with a decimal separator.
32	LastQty	M	*Qty* Quantity. Number of units of the financial instrument of the trade report (>0 (zero)). Up to 20 numerical digits with a decimal separator.
487	TradeReportTransType	M	*Integer* Identifies Trade Report message transaction type. Valid Values and Description: 0 = New (ENTR) 1 = Cancel (CANC) 2 = Replace (AMND)
570	PreviouslyReported	M	*Boolean* ISE Pre-Trade Transparency Flag identifier for “ISE Ordinary Trade” (OT). Valid Values and Description: Y = Trade meets the requirements of trade flag OT (ISE Ordinary Trade) N = Trade has not been flagged as OT (i.e. not an ISE Ordinary Trade) The value “Y” is not allowed if the field TrdRegPublicationReason (2670) in the group <TrdRegTimestamps> is set. See “5.3.3 Input Trade Flags”.
571	TradeReportID	M	*String* Unique identifier of Trade Capture Report. For FIX requests the uniqueness must be ensured by the customer and will not be verified by the FIX Gateway. Will be set by the FIX Gateway for messages entered via external interface (GUI) and for status updates.
828	TrdType	M	*Integer* Type of trade. Valid Values and Description: 0 = Regular trade
829	TrdSubType	O	*Integer* Further qualification of the trade type. Valid Value and Description: 37 = Crossed Trade (ACTX) See “5.3.3 Input Trade Flags”.

Tag	Field Name	R	Description
855	SecondaryTrdType	O	*Integer* Additional TrdType (828) assigned to a trade. Valid Value and Description: 64 = Benchmark (BENC) See “5.3.3 Input Trade Flags”.
939	TrdRptStatus	O	*Integer* Trade Report Status. Valid Values and Description: 0 = Accepted 6 = Pending Replace Messages from client The field is not required, but if it is set only the value 0 (Accepted) is allowed Messages to client The field will be always set in the messages to the client: - The value 6 (Pending Replace) will be set in combination with TradeReportTransType (487) = 2 (Replace) to identify the delete of the existing report data during the processing of an amendment entered via GUI. - In all other cases the value 0 (Accepted) will be set. Additional details (Information, Warning, Alert) can be contained in the component <StatusInformationGrp>.
979	InputSource	O	*String* Originating source of the message. Valid Values and Description: GUI = Web GUI OBOE_ENGINE = OBOE Engine Not allowed in messages from client.
1003	TradeID	O	*String* Unique ID assigned by the ISE OBOE MiFID II Engine. Messages from client Not allowed for trade entry. For amendments and cancellations the field is mandatory and is used as trade identifier. Messages to client Trade identification number generated by the ISE OBOE MiFID II Engine. The TradeID (1003) will be generated after a successful trade entry. Its value will not change after an amendment.

Tag	Field Name	R	Description
1011	MessageEventSource	O	*String* Used to identify the event that caused the creation of a status update message. Valid Value and Description: 100 = Actual publication after deferral Not allowed in messages from client.
1041	FirmTradeID	O	*String* Optional ID assigned by the customer (client reference).
1390	TradePublishIndicator	O	*Integer* Publish indicator. Valid Values and Description: 1 = Publish Trade 2 = Deferred Publication The value 1 (Publish Trade) will be set to indicate that trade was published without deferral. The value 2 (Deferred Publication) will be set to indicate that the publication was deferred. In this case the report will be published with the post-trade flag LRGS (post-trade large in scale deferral). Not allowed in messages from client.
5681	ExchangeTradeType	O	*String* ISE Trade flags. ISE Trade flags concatenated into a string of up to three values delimited by ";". Valid Values and Description: AH = After Hours Trading BB = Broker to Broker Trade CP = Connected Party Trade LT = Late Trade RP = Riskless Principal SS = Special Settlement See "5.3.3 Input Trade Flags".
<StandardTrailer>			

5.3.2 User / Trade Capture Report Ack (UAR/AR)

The *User / Trade Capture Report Ack (UAR/AR)* message is the response to a *User / Trade Capture Report (UAE/AE)* request sent by the client for trade reporting (post-trade).

Tag	Field Name	R	Description
<StandardHeader>		M	Tag 35 = UAR/AR (User / Trade Capture Report Ack)
<Instrument>		M	Financial Instrument.
<StatusInformationGrp>		O	Status information (Information, Warning, Alert) for a FIX message processed successfully.
<TrdRegPublicationGrp>		O	Publication Reasons. Will be copied from the request.
<Begin TrdRegTimestamps>		O	Timestamps.
<trade date and time>		O	Trade date and time when the trade was executed by the involved parties.
<publication date and time>		O	Publication date and time. The field will be only set if the report has been published without deferral (TradePublishIndicator (1390) = “1”).
<preliminary publication date and time>		O	Preliminary publication date and time. The field will be set in case of deferred publication (TradePublishIndicator (1390) = “2”)and contains the preliminary publication date and time (result of the deferral calculation).
<End TrdRegTimestamps>			
150	ExecType	M	*Char* Type of Execution being reported. Valid Value and Description: F = Trade
487	TradeReportTransType	M	*Integer* Identifies Trade Report message transaction type. Valid Values and Description: 0 = New (ENTR) 1 = Cancel (CANC) 2 = Replace (AMND) Will be copied from the request.
571	TradeReportID	M	*String* Unique identifier of the Trade Capture Report. Will be copied from the request.
751	TradeReportRejectReason	O	*Integer* Error code.

Tag	Field Name	R	Description
939	TrdRptStatus	M	*Integer* Trade Report Status. Valid Values and Description: 0 = Accepted 1 = Rejected 4 = Pending New 5 = Pending Cancel 6 = Pending Replace In case of TrdRptStatus = “0” (Accepted) additional details (Information, Warning, Alert) can be contained in the component <StatusInformationGrp>.
1003	TradeID	O	*String* Unique ID assigned by the ISE OBOE MiFID II Engine. Response to a trade entry request: The field will contain the trade identification number generated by the ISE OBOE MiFID II Engine. Will not be set for rejections. Response to amendment and cancellation requests: Value will be copied from the FIX request.
1041	FirmTradeID	O	*String* Optional ID assigned by the customer (client reference).
1328	RejectText	O	*String* Error text.
1390	TradePublishIndicator	O	*Integer* Publish indicator. Valid Values and Description: 1 = Publish Trade 2 = Deferred Publication The value 1 (Publish Trade) will be set to indicate that trade was published without deferral. The value 2 (Deferred Publication) will be set to indicate that the publication was deferred. In this case the report will be published with the post-trade flag LRGS (post-trade large in scale deferral).
5177	Source	O	*String* Will identify the component providing the reject or pending information. Valid Values and Description: FIX_GATEWAY = FIX Gateway OBOE_ENGINE = OBOE Engine
5681	ExchangeTradeType	O	*String* ISE Trade flags concatenated into a string of up to three values delimited by “;”.
<StandardTrailer>			

5.3.3 Input Trade Flags

Trade Flag	Trade Flag Name	Exchange-Trade-Type (5681)	Previously-Reported (570)	TrdSub-Type (829)	Secondary-TrdType (855)	Trade-Price-Condition (1839)	Last-Capacity (29)	TrdReg-Publication-Reason (2670)
BENC	Benchmark Trade				**64** (Benchmark)			
ACTX	Agency Cross Trade			**37** (Crossed Trade)				
SDIV	Special Dividend Trades (only for equity trades)					**13** (Special Dividend)		
DEAL	Deal on own account						**1** (Deal on own account)	
AOTC	Any other capacity						**2** (Any other capacity)	
NLIQ	Liquid instrument							**0** (liquid instrument)
OILQ	Illiquid instrument							**1** (Illiquid instrument)
PRIC	Conditions other than the current market price							**2** (Conditions other than the current market price)
LIS	Large in Scale							**6** (Large in Scale)
AH	After Hours Trading	**AH** (After Hours Trading)						
BB	Broker to Broker Trade	**BB** (Broker to Broker Trade)						
CP	Connected Party Trade	**CP** (Connected Party Trade)						

Trade Flag	Trade Flag Name	Exchange-Trade-Type (5681)	Previously-Reported (570)	TrdSub-Type (829)	Secondary-TrdType (855)	Trade-Price-Condition (1839)	Last-Capacity (29)	TrdReg-Publication-Reason (2670)
LT	Late Trade	**LT** (Late Trade)						
RP	Riskless Principal	**RP** (Riskless Principal)						
SS	Special Settlement	**SS** (Special Settlement)						
OT	ISE Ordinary Trade		**Y** (Trade meets the requirements of trade flag OT)					

It is possible to enter several flags simultaneously setting the different fields appropriately.

Empty fields in the table above will not be taken into account for the determination of the corresponding flag.

Example:
If the field *SecondaryTrdType (855)* is set to *64 (Benchmark)*, the flag *BENC* is set, independently of the values of the other fields. If the same message contains additionally the field *TrdSubType (829)* with the value *37 (Crossed Trade)*, the flag *ACTX* is also set.

5.4 Component Blocks

5.4.1 <Instrument>

Instrument Identification.

Tag	Field Name	R	Description
<Begin Instrument>			Financial Instrument.
55	Symbol	M	*String* This field is not used by the FIX Gateway and will be set to “[N/A]”. Clients also should supply “[N/A]” in all requests
48	SecurityID	M	*String* Security identifier.
22	SecurityIDSource	M	*String* Identifies class or source of the instrument data. Valid Values and Description: 4 = ISIN number
<End Instrument>			

5.4.2 <StatusInformationGrp>

The *StatusInformationGrp* component block is used to specify additional information for a FIX message processed successfully in the form of “Information”, “Warning” or “Alert”.

A message can contain one or more status information entries.

Tag	Field Name	R	Description
<Begin StatusInformationGrp>			Status information for a FIX message processed successfully.
25104	NoStatusInformation	M	*NumInGroup* Counter for information in addition to status field.
25105	StatusInformationType	M	*Integer* Type of status information. Valid Values and Description: 1 = Information 2 = Warning 3 = Alert
25106	StatusInformationValue	M	*String* This field contains the text for the type contained in the field StatusInformationType (25105).
<End StatusInformationGrp>			

5.4.3 <TrdRegPublicationGrp>

The *TrdRegPublicationGrp* component block is used to express trade publication reasons.

Tag	Field Name	R	Description
<Begin TrdRegPublicationGrp>			Timestamps.
2668	NoTrdRegPublications	M	*NumInGroup* Number of publication reasons.
2669	TrdRegPublicationType	M	*Integer* Type of regulatory trade publication. Valid Values and Description: 1 = Post-trade deferral
2670	TrdRegPublicationReason	M	*Integer* ISE transparency flag. Additional reason for trade publication specified in TrdRegPublicationType (2669). Valid Values and Description: 0 = Liquid instrument (NLIQ) 1 = Illiquid instrument (OILQ) 2 = Conditions other than the current market price (PRIC) 6 = Large in Scale (LIS) See “5.3.3 Input Trade Flags”.
<End TrdRegPublicationGrp>			

5.4.4 <TrdRegTimestamps>

The *TrdRegTimestamps* component block is used to express timestamps that are required by regulatory agencies.

5.4.4.1 TrdRegTimestamps component block

Tag	Field Name	R	Description
<Begin TrdRegTimestamps>			Timestamps.
768	NoTrdRegTimestamps	M	*NumInGroup* Number of repeating groups of TrdRegTimestamps.
769	TrdRegTimestamp	M	*UTCTimestamp* Traded/Regulatory timestamp value. Following formats are supported in messages from client: - YYYYMMDD-HH:MM:SS - YYYYMMDD-HH:MM:SS.sss - YYYYMMDD-HH:MM:SS.ssssss The messages to the clients will always contain values in microseconds: - YYYYMMDD-HH:MM:SS.ssssss
770	TrdRegTimestampType	M	*Integer* Trading/Regulatory timestamp type. Valid Values and Description: 1 = Execution Time 11 = Publicly reported (enter) 12 = Public report updated (amend, cancel)
28748	TrdRegTimestampSub-Type	O	*Integer* Further qualification of a trading/regulatory timestamp type. Will be set in trade messages sent to the client together with TrdRegTimestampType (770) = 11 (Publicly reported) and 12 (Public report updated) for the delivery of preliminary publication timestamps. Not allowed in messages from client. Valid Values and Description: 1 = Preliminary
<End TrdRegTimestamps>			

5.4.4.2 TrdRegTimestamps entries

The following table contains the different entries of the component block *TrdRegTimestamps*, as referenced in the message definitions, with the corresponding combination of the fields *TrdRegTimestampType (770)* and *TrdRegTimestampSubType (28748)*.

TrdRegTimestamp	TrdRegTimestampType (770)	TrdRegTimestampSub-Type (28748)
<trade date and time>	1 = Execution Time	-
<publication date and time>	11 = Publicly reported (enter) 12 = Public report updated (amend, cancel)	-
<preliminary publication date and time>	11 = Publicly reported (enter) 12 = Public report updated (amend, cancel)	1 = Preliminary

5.4.5 <TradePriceConditionGrp>

The *TradePriceConditionGrp* component block is used to express price conditions associated with a trade that impact trade price.

The component will be used to enter the trade flag *SDIV (Special Dividend Trade)*. This will be referred to as <trade flag sdiv>. See also "5.3.3 Input Trade Flags".

Tag	Field Name	R	Description
<Begin TradePriceConditionGrp>			Trade Price Conditions.
1838	NoTradePriceConditions	M	*NumInGroup* Number of trade price conditions.
1839	TradePriceCondition	M	*Integer* Price conditions in effect at the time of the trade. Valid Values and Description: <trade flag sdiv> 13 = Special Dividend (SDIV)
<End TradePriceConditionGrp>			

5.4.6 <Parties>

5.4.6.1 Parties component block

The following structure of the party component block is used for FIX 4.2 and FIX 4.4.

Tag	Field Name	R	Description
<Begin Parties>			Party Information.
453	NoPartyIDs	M	*NumInGroup* Number of PartyID (448), PartyIDSource (447) and PartyRole (452) entries. Number of parties involved.
448	PartyID	M	*String* Party identifier.
447	PartyIDSource	M	*Char* Identifies class or source of the PartyID (448) value. Valid Value and Description: D = Proprietary custom code N = Legal Entity Identifier (LEI)
452	PartyRole	M	*String* Identifies the type role of the PartyID (448) specified. Valid Values and Description: 1 = Executing Firm 7 = Entering Firm 17 = Contra Firm 36 = Entering Trader
<End Parties>			

5.4.6.2 Supported parties

The <Parties> component block comprises all parties participating in a transaction. For each party a separate occurrence of the repeating group is set up.

The parties component block is referred to by <Parties>, the different entries will be shown in the following table.

Party	Content	Description	PartyRole (452)	PartyIDSource (447)
<entering firm>	Member Legal Entity Identifier (LEI)	Identifies the owner of the FIX session. Will contain the same value of the executing firm if the client reports own trades or a different value for delegated reporting, except for trade reports submitted by the Market Supervisor, which have a specific Member LEI of Market Supervisor.	7 = Entering Firm	N = Legal Entity Identifier (LEI)
<executing firm>	Executing Legal Entity Identifier (LEI)	The owner of the trades being reported. Can be used for delegated reporting.	1 = Executing Firm	N = Legal Entity Identifier (LEI)
<entering trader>	User ID	The trader/user of the ISE member submitting the trade report. Must be filled with a valid user ID of the entering ISE member, as setup in the Member Portal.	36 = Entering Trader	D = Proprietary custom code
<counterparty>	Counterparty	Description of the counterparty. The field can contain a LEI or a natural person, depending on the value of the field PartyIDSource (447).	17 = Contra Firm	N = Legal Entity Identifier (LEI) or D = Proprietary custom code