FINANCIAL INFORMATION EXCHANGE PROTOCOL (FIX)

Version 5.0 Service Pack 1

VOLUME 4 – FIX APPLICATION MESSAGES: ORDERS AND EXECUTIONS (TRADE)

March 2008

DISCLAIMER

THE INFORMATION CONTAINED HEREIN AND THE FINANCIAL INFORMATION EXCHANGE PROTOCOL (COLLECTIVELY, THE "FIX PROTOCOL") ARE PROVIDED "AS IS" AND NO PERSON OR ENTITY ASSOCIATED WITH THE FIX PROTOCOL MAKES ANY REPRESENTATION OR WARRANTY, EXPRESS OR IMPLIED, AS TO THE FIX PROTOCOL (OR THE RESULTS TO BE OBTAINED BY THE USE THEREOF) OR ANY OTHER MATTER AND EACH SUCH PERSON AND ENTITY SPECIFICALLY DISCLAIMS ANY WARRANTY OF ORIGINALITY, ACCURACY, COMPLETENESS, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. SUCH PERSONS AND ENTITIES DO NOT WARRANT THAT THE FIX PROTOCOL WILL CONFORM TO ANY DESCRIPTION THEREOF OR BE FREE OF ERRORS. THE ENTIRE RISK OF ANY USE OF THE FIX PROTOCOL IS ASSUMED BY THE USER.

NO PERSON OR ENTITY ASSOCIATED WITH THE FIX PROTOCOL SHALL HAVE ANY LIABILITY FOR DAMAGES OF ANY KIND ARISING IN ANY MANNER OUT OF OR IN CONNECTION WITH ANY USER'S USE OF (OR ANY INABILITY TO USE) THE FIX PROTOCOL, WHETHER DIRECT, INDIRECT, INCIDENTAL, SPECIAL OR CONSEQUENTIAL (INCLUDING, WITHOUT LIMITATION, LOSS OF DATA, LOSS OF USE, CLAIMS OF THIRD PARTIES OR LOST PROFITS OR REVENUES OR OTHER ECONOMIC LOSS), WHETHER IN TORT (INCLUDING NEGLIGENCE AND STRICT LIABILITY), CONTRACT OR OTHERWISE, WHETHER OR NOT ANY SUCH PERSON OR ENTITY HAS BEEN ADVISED OF, OR OTHERWISE MIGHT HAVE ANTICIPATED THE POSSIBILITY OF, SUCH DAMAGES.

No proprietary or ownership interest of any kind is granted with respect to the FIX Protocol (or any rights therein) except as expressly set out in FIX Protocol Limited's Copyright and Acceptable Use Policy.

REPRODUCTION

FIX Protocol Limited grants permission to print in hard copy form or reproduce the FIX Protocol specification in its entirety provided that the duplicated pages retain the “Copyright FIX Protocol Limited” statement at the bottom of the page.

Portions of the FIX Protocol specification may be extracted or cited in other documents (such as a document which describes one’s implementation of the FIX Protocol) provided that one reference the origin of the FIX Protocol specification (***http://www.fixprotocol.org***) and that the specification itself is “Copyright FIX Protocol Limited”.

FIX Protocol Limited claims no intellectual property over one’s implementation (programming code) of an application which implements the behavior and details from the FIX Protocol specification.

Contents – Volume 4

DISCLAIMER **2**

REPRODUCTION **2**

FIX APPLICATION MESSAGES: ORDERS AND EXECUTIONS (TRADE) **5**

CATEGORY: SINGLE/GENERAL ORDER HANDLING **6**

NEW ORDER - SINGLE 6
EXECUTION REPORTS 12
Use of the Execution Report for Multileg Instruments: *27*
DON'T KNOW TRADE (DK) 29
EXECUTION REPORT ACKNOWLEDGEMENT 31
Using the Execution Report Ack *32*
Using the Execution Report Ack with DK Trade *35*
ORDER CANCEL/REPLACE REQUEST (A.K.A. ORDER MODIFICATION REQUEST) 36
ORDER CANCEL REQUEST 43
ORDER CANCEL REJECT 45
ORDER STATUS REQUEST 47
ORDER STATE CHANGE MATRICES 50
A Vanilla *55*
B Cancel *57*
C Cancel/Replace quantity changes *62*
D Cancel/Replace sequencing and chaining *68*
E Unsolicited/Reinstatement *75*
F Order Reject *78*
G Status *81*
H GT *83*
I TimeInForce *87*
J Execution Cancels/Corrects *88*
K Trading Halt *91*
L Miscellaneous *92*
ORDER HANDLING AND INSTRUCTION SEMANTICS 95
London SETS Order Types Matrix *95*
Asia/Pacific Regional Order Handling *95*
Japanese Exchange Price Conditions *96*
Euronext and Similar Exchange Price Conditions *96*
Handling Instructions (HandlInst) field *96*
Pegged Orders *97*
Discretionary Pricing *101*
"Target Strategy" Orders *101*
"Reserve Quantity" Orders *101*
Triggering Instructions *105*
Time In Force (TIF) *108*
Booking Instructions Specified at Time of Order *108*
ORDERCAPACITY AND ORDERRESTRICTIONS (FORMERLY RULE80A) USAGE BY MARKET 110
EXAMPLE USAGE OF PARTYROLE="INVESTOR ID" 114
Format of the Party ID field (PartyRole="Investor ID") *114*
Example Representations of Orders *114*

CATEGORY: ORDER MASS HANDLING **116**

ORDER MASS CANCEL REQUEST 116

ORDER MASS CANCEL REPORT 119
ORDER MASS STATUS REQUEST 122
ORDER MASS ACTION REQUEST 124
ORDER MASS ACTION REPORT 126

CATEGORY: CROSS ORDERS 128

BACKGROUND 128
PRIORITIZATION OF A SIDE OF A CROSS ORDER 128
CLASSIFICATION OF CROSS TRADES 128
EXECUTION REPORTING FOR CROSS ORDERS 128
CROSS ORDER HANDLING RULES 129
Acknowledgement of a Cross Order 129
Message Flow for cross order with CrossType=1 with only one side of the order provided 130
Message Flow for cross order with CrossType=1 when both sides of the cross order provided 131
Message Flow for cross order with CrossType=2 132
Message Flow for cross order with CrossType=3 133
Message Flow for cross order with CrossType=4 134
NEW ORDER - CROSS 135
CROSS ORDER CANCEL/REPLACE REQUEST (A.K.A. CROSS ORDER MODIFICATION REQUEST) 140
CROSS ORDER CANCEL REQUEST 146
CROSS ORDER CHANGE MATRICES 149
Cross Type 1 149
Cross Type 2 152
Cross Type 3 153
Cross Type 4 154

CATEGORY: MULTILEG ORDERS (SWAPS, OPTION STRATEGIES, ETC) 155

BACKGROUND 155
Predefined Multileg Security Model (FIX 4.2) (Model 1) 155
Enhanced Predefined Security Model (Model 2) 156
Product Definition Model using New Order - Multileg Message (Model 3) 157
Single Message Model (Model 4) 157
Messages Used for Multileg Trading 158
Multileg Pricing Methods 158
NEW ORDER - MULTILEG 159
MULTILEG ORDER CANCEL REPLACE REQUEST (A.K.A MULTILEGORDER MODIFICATION REQUEST) 165

CATEGORY: LIST/PROGRAM/BASKET TRADING 171

BID REQUEST 171
BID RESPONSE 174
NEW ORDER - LIST 176
LIST STRIKE PRICE 182
LIST STATUS 184
LIST EXECUTE 186
LIST CANCEL REQUEST 187
LIST STATUS REQUEST 188
FRAGMENTATION FOR LIST ORDER MESSAGES 189
PROGRAM/BASKET/LIST TRADING 190
Overview 190
Message Flow Diagrams 191
CONTINGENT ORDERS 198
Overview 198
Types of Contigent Orders 198

FIX APPLICATION MESSAGES: ORDERS AND EXECUTIONS (TRADE)

"Orders and Executions" (or "Trade") messaging is characterized as messages which are used to place or amend orders and communicate the results and status of orders.

The specific FIX "Orders and Executions" (or "Trade") messaging categories are:

1. SINGLE/GENERAL ORDER HANDLING
2. ORDER MASS HANDLING
3. CROSS ORDER HANDLING
4. MULTILEG ORDER HANDLING
5. LIST/PROGRAM/BASKET TRADING

Descriptions and formats of the specific FIX "Orders and Executions" (or "Trade") application messages follow.

CATEGORY: SINGLE/GENERAL ORDER HANDLING

See Volume 7 – PRODUCT: FIXED INCOME for usage guidance in using general order handling messages for Fixed Income trading.

New Order - Single

The new order message type is used by institutions wishing to electronically submit securities and forex orders to a broker for execution.

The New Order message type may also be used by institutions or retail intermediaries wishing to electronically submit Collective Investment Vehicle (CIV) orders to a broker or fund manager for execution.

See VOLUME 7 - "PRODUCT: COLLECTIVE INVESTMENT VEHICLES"

Orders can be submitted with special handling instructions and execution instructions. Handling instructions refer to how the broker should handle the order on its trading floor (see HandlInst field). Execution instructions contain explicit directions as to how the order should be executed (see ExecInst field).

New Order messages received with the PossResend flag set in the header should be validated by ClOrdID. Implementations should also consider checking order parameters (side, symbol, quantity, etc.) to determine if the order had been previously submitted. PossResends previously received should be acknowledged back to the client via an Execution - Status message. PossResends not previously received should be processed as a new order and acknowledged via an Execution - New message.

The value specified in the TransactTime field should allow the receiver of the order to apply business rules to determine if the order is potentially "stale" (e.g. in the event that there have been communication problems).To support forex accommodation trades, two fields, ForexReq and SettlCurrency, are included in the message. To request a broker to execute a forex trade in conjunction with the securities trade, the institution would set the ForexReq = Y and SettlCurrency = “intended settlement currency”. The broker would then execute a forex trade from the execution currency to the settlement currency and report the results via the execution message in the SettlCurrAmt and SettlCurrency fields.

See VOLUME 7 - "PRODUCT: FOREIGN EXCHANGE" section for more detailed usage notes specific to Foreign Exchange trading.

Orders involving or requiring Pre-Trade Allocation consist of the following steps:

- Buyside sends a New Order request message specifying one or more AllocAccount and AllocQty values within the repeating group designated by NoAllocs.
- Sellside sends Execution Report messages for the “New” and resulting fills.
- Post-Trade Allocation messaging takes place

To “take” an IOI (or Quote) from an ECN or exchange and not display the order on the book, the New Order message should contain the TimeInForce field with ImmediateOrCancel and an OrdType field with Previously Indicated (or Previously Quoted).

See “***Order State Change Matrices***”

The format for the new order message is as follows:

New Order - Single

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = D
11	ClOrdID	Y	Unique identifier of the order as assigned by institution or by the intermediary (CIV term, not a hub/service bureau) with closest association with the investor.
526	SecondaryClOrdID	N	
583	ClOrdLinkID	N	
component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
229	TradeOriginationDate	N	
75	TradeDate	N	
1	Account	N	
660	AcctIDSource	N	
581	AccountType	N	Type of account associated with the order (Origin)
589	DayBookingInst	N	
590	BookingUnit	N	
591	PreallocMethod	N	
70	AllocID	N	Used to assign an overall allocation id to the block of preallocations
Start of Component block, expanded in line < PreAllocGrp >			
78	NoAllocs	N	Number of repeating groups for pre-trade allocation
→ 79	AllocAccount	N	Required if NoAllocs > 0. Must be first field in repeating group.
→ 661	AllocAcctIDSource	N	
→ 736	AllocSettlCurrency	N	
→ 467	IndividualAllocID	N	
→ component block <NestedParties>		N	Insert here the set of "Nested Parties" (firm identification "nested" within additional repeating group) fields defined in "Common Components of Application Messages" Used for NestedPartyRole=Clearing Firm
→ 80	AllocQty	N	
End of Component block, expanded in line < PreAllocGrp >			
63	SettlType	N	
64	SettlDate	N	Takes precedence over SettlType value and conditionally required/omitted for specific SettlType values.
544	CashMargin	N	

635	ClearingFeeIndicator		N	
21	HandlInst		N	
18	ExecInst		N	Can contain multiple instructions, space delimited. If OrdType=P, exactly one of the following values (ExecInst = L, R, M, P, O, T, W, a, d) must be specified.
110	MinQty		N	
1089	MatchIncrement		N	
1090	MaxPriceLevels		N	
component block <DisplayInstruction>			N	
111	MaxFloor		N	(Deprecated in FIX.5.0)
100	ExDestination		N	
1133	ExDestinationIDSource		N	
Start of Component block, expanded in line < TrdgSesGrp >				
386	NoTradingSessions		N	Specifies the number of repeating TradingSessionIDs
→	336	TradingSessionID	N	Required if NoTradingSessions is > 0.
→	625	TradingSessionSubID	N	
End of Component block, expanded in line < TrdgSesGrp >				
81	ProcessCode		N	Used to identify soft trades at order entry.
component block <Instrument>			Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
component block <FinancingDetails>			N	Insert here the set of "FinancingDetails" (symbology) fields defined in "Common Components of Application Messages"
Start of Component block, expanded in line < UndInstrmtGrp >				
711	NoUnderlyings		N	Number of underlyings
→	component block <UnderlyingInstrument>		N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >				
140	PrevClosePx		N	Useful for verifying security identification
54	Side		Y	
114	LocateReqd		N	Required for short sell orders
60	TransactTime		Y	Time this order request was initiated/released by the trader, trading system, or intermediary.
component block <Stipulations>			N	Insert here the set of "Stipulations" (repeating group of Fixed Income stipulations) fields defined in "Common Components of Application Messages"

854	QtyType	N	
component block <OrderQtyData>		Y	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages"
40	OrdType	Y	
423	PriceType	N	
44	Price	N	Required for limit OrdTypes. For F/X orders, should be the "all-in" rate (spot rate adjusted for forward points). Can be used to specify a limit price for a pegged order, previously indicated, etc.
1092	PriceProtectionScope	N	
99	StopPx	N	Required for OrdType = "Stop" or OrdType = "Stop limit".
component block <TriggeringInstruction>		N	Insert here the set of "TriggeringInstruction" fields defined in "common components of application messages"
component block <SpreadOrBenchmarkCurveData>		N	Insert here the set of "SpreadOrBenchmarkCurveData" (Fixed Income spread or benchmark curve) fields defined in "Common Components of Application Messages"
component block <YieldData>		N	Insert here the set of "YieldData" (yield-related) fields defined in "Common Components of Application Messages"
15	Currency	N	
376	ComplianceID	N	
377	SolicitedFlag	N	
23	IOIID	N	Required for Previously Indicated Orders (OrdType=E)
117	QuoteID	N	Required for Previously Quoted Orders (OrdType=D)
59	TimeInForce	N	Absence of this field indicates Day order
168	EffectiveTime	N	Can specify the time at which the order should be considered valid
432	ExpireDate	N	Conditionally required if TimeInForce = GTD and ExpireTime is not specified.
126	ExpireTime	N	Conditionally required if TimeInForce = GTD and ExpireDate is not specified.
427	GTBookingInst	N	States whether executions are booked out or accumulated on a partially filled GT order
component block <CommissionData>		N	Insert here the set of "CommissionData" fields defined in "Common Components of Application Messages"
528	OrderCapacity	N	
529	OrderRestrictions	N	
1091	PreTradeAnonymity	N	

582	CustOrderCapacity		N	
121	ForexReq		N	Indicates that broker is requested to execute a Forex accommodation trade in conjunction with the security trade.
120	SettlCurrency		N	Required if ForexReq = Y.
775	BookingType		N	Method for booking out this order. Used when notifying a broker that an order to be settled by that broker is to be booked out as an OTC derivative (e.g. CFD or similar). Absence of this field implies regular booking.
58	Text		N	
354	EncodedTextLen		N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText		N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
193	SettlDate2		N	(Deprecated in FIX.5.0)Can be used with OrdType = "Forex - Swap" to specify the "value date" for the future portion of a F/X swap.
192	OrderQty2		N	(Deprecated in FIX.5.0)Can be used with OrdType = "Forex - Swap" to specify the order quantity for the future portion of a F/X swap.
640	Price2		N	(Deprecated in FIX.5.0)Can be used with OrdType = "Forex - Swap" to specify the price for the future portion of a F/X swap which is also a limit order. For F/X orders, should be the "all-in" rate (spot rate adjusted for forward points).
77	PositionEffect		N	For use in derivatives omnibus accounting
203	CoveredOrUncovered		N	For use with derivatives, such as options
210	MaxShow		N	(Deprecated in FIX.5.0)
component block <PegInstructions>			N	Insert here the set of "PegInstruction" fields defined in "Common Components of Application Messages"
component block <DiscretionInstructions>			N	Insert here the set of "DiscretionInstruction" fields defined in "Common Components of Application Messages"
847	TargetStrategy		N	The target strategy of the order
Start of Component block, expanded in line < StrategyParametersGrp >				
957	NoStrategyParameters		N	Indicates number of strategy parameters
→	958	StrategyParameterName	N	Name of parameter
→	959	StrategyParameterType	N	Datatype of the parameter.
→	960	StrategyParameterValu	N	Value of the parameter

		e		
End of Component block, expanded in line < StrategyParametersGrp >				
848	TargetStrategyParameters		N	(Deprecated in FIX.5.0)For further specification of the TargetStrategy
849	ParticipationRate		N	(Deprecated in FIX.5.0)Mandatory for a TargetStrategy=Participate order and specifies the target particpation rate. For other order types optionally specifies a volume limit (i.e. do not be more than this percent of the market volume)
480	CancellationRights		N	For CIV - Optional
481	MoneyLaunderingStatus		N	
513	RegistID		N	Reference to Registration Instructions message for this Order.
494	Designation		N	Supplementary registration information for this Order
1028	ManualOrderIndicator		N	
1029	CustDirectedOrder		N	
1030	ReceivedDeptID		N	
1031	CustOrderHandlingInst		N	
1032	OrderHandlingInstSource		N	
component block <TrdRegTimestamps>			N	
1080	RefOrderID		N	Required for counter-order selection / Hit / Take Orders. (OrdType = Q)
1081	RefOrderIDSource		N	Conditionally required if RefOrderID is specified.
StandardTrailer			Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element NewOrdSingle

Execution Reports

The execution report message is used to:

1. confirm the receipt of an order
2. confirm changes to an existing order (i.e. accept cancel and replace requests)
3. relay order status information
4. relay fill information on working orders
5. relay fill information on tradeable or restricted tradeable quotes
6. reject orders
7. report post-trade fees calculations associated with a trade

NOTE: Execution reports do not replace the end-of-day confirm. Execution reports are to be regarded only as replacements for the existing fill messages currently communicated via telephone.

NOTE: Individual Execution Reports are sent for each order on a New Order - List.

Each execution report contains two fields which are used to communicate both the current state of the order as understood by the broker (OrdStatus) and the purpose of the message (ExecType).

In an execution report the OrdStatus is used to convey the current state of the order. If an order simultaneously exists in more than one order state, the value with highest precedence is the value that is reported in the OrdStatus field. The order statuses are as follows (in highest to lowest precedence):

Precedence	OrdStatus	Description
11	Pending Cancel	Order with an Order Cancel Request pending, used to confirm receipt of an Order Cancel Request. DOES NOT INDICATE THAT THE ORDER HAS BEEN CANCELED.
10	Pending Replace	Order with an Order Cancel/Replace Request pending, used to confirm receipt of an Order Cancel/Replace Request. DOES NOT INDICATE THAT THE ORDER HAS BEEN REPLACED.
9	Done for Day	Order not, or partially, filled; no further executions forthcoming for the trading day
8	Calculated	Order has been completed for the day (either filled or done for day). Commission or currency settlement details have been calculated and reported in this execution message
7	Filled	Order completely filled, no remaining quantity
6	Stopped	Order has been stopped at the exchange. Used when guranteeing or protecting a price and quantity
5	Suspended	Order has been placed in suspended state at the request of the client.
4	Canceled	Canceled order with or without executions
4	Expired	Order has been canceled in broker's system due to time in force instructions. The only exceptions are Fill or Kill and Immediate or Cancel orders that have Canceled as terminal order state.

3	Partially Filled	Outstanding order with executions and remaining quantity
2	New	Outstanding order with no executions
2	Rejected	Order has been rejected by sell-side (broker, exchange, ECN). NOTE: An order can be rejected subsequent to order acknowledgment, i.e. an order can pass from New to Rejected status.
2	Pending New	Order has been received by sell-side's (broker, exchange, ECN) system but not yet accepted for execution. An execution message with this status will only be sent in response to a Status Request message.
1	Accepted for bidding	Order has been received and is being evaluated for pricing. It is anticipated that this status will only be used with the "Disclosed" BidType List Order Trading model.

The ExecType is used to identify the purpose of the execution report message. To transmit a change in OrdStatus for an order, the broker (sell side) should send an Execution Report with the new OrdStatus value in both the ExecType AND the OrdStatus fields to signify this message is changing the state of the order. The only exception to this rule is that when rejecting a cancel or cancel/replace request the CancelReject message is used both to reject the request and to communicate the current OrdStatus. An ExecType of Pending Cancel or Pending Replace is used to indicate that a cancel or cancel/replace request is being processed. An ExecType of Canceled or Replace is used to indicate that the cancel or cancel/replace request has been successfully processed.

Execution information (e.g. new partial fill or complete fill) should not be communicated in the same report as one which communicates other state changes (such as pending cancel, pending replace, canceled, replaced, accepted, done for day etc).

Any fills which occur and need to be communicated to the customer while an order is "pending" and waiting to achieve a new state (e.g. via a Order Cancel Replace (aka Order Modification) Request) must contain the "original" (current order prior to state change request) order parameters (i.e. ClOrdID, OrderQty, Price, etc). These fills will cause the CumQty and AvgPx to be updated. An order cannot be considered replaced until it has been explicitly accepted and confirmed to have reached the replaced status via an execution report with ExecType = 'Replace', at which time the effect of the replacement (ClOrdID, new quantity or limit price etc) will be seen.

Requests to cancel or cancel/replace an order are only acted upon when there is an outstanding order quantity. Requests to replace the OrderQty to a level less than the CumQty will be interpreted by the broker as requests to stop executing the order. Requests to change price on a filled order will be rejected (see Order Cancel Reject message type). The OrderQty, CumQty, LeavesQty, and AvgPx fields should be calculated to reflect the cumulative result of all versions of an order. For example, if partially filled order A were replaced by order B, the OrderQty, CumQty, LeavesQty, and AvgPx on order B's fills should represent the cumulative result of order A plus those on order B.

The general rule is: OrderQty = CumQty + LeavesQty.

There can be exceptions to this rule when ExecType and/or OrdStatus are Canceled, DoneForTheDay (e.g. on a day order), Expired, Calculated, or Rejected in which case the order is no longer active and LeavesQty could be 0.

Communication of information about a new fill is via the Execution report with ExecType = Trade. Execution Reports with ExecType = Trade Cancel or Trade Correct are used to cancel or correct a previously modified execution report as follows:

- The ExecType of Trade Cancel applies at the execution level and is used to cancel an execution which has been reported in error. The canceled execution will be identified in the ExecRefID field. Note: ExecType of Trade Cancel should not be used to cancel a previous ExecutionRpt with ExecType of Trade Cancel (i.e. cannot cancel a cancel).
- The ExecType of Trade Correct applies at the execution level and is used to modify an incorrectly reported fill. The incorrect execution will be identified in the ExecRefID field. If a single execution is corrected more than once, ExecRefID should refer to the ExecID of the last corrected ExecutionRpt (same convention as ClOrdID and OrigClOrdID). To correct an ExecutionRpt which was previously canceled, an ExecutionRpt with ExecType=Trade should be sent (i.e. cannot send ExecType=Trade Correct for an ExecutionRpt with ExecType=Trade Cancel). *Note: Data reported in the CumQty, LeavesQty, and AvgPx fields represent the status of the order as of the time of the correction, not as of the time of the originally reported execution.*

An ExecType of Order Status indicates that the execution messages contains no new information, only summary information regarding order status. It is used, for example, in response to an Order Status request message

See "Order State Change Matrices" for examples of key state changes, processing of cancel and cancel/replace requests, and for execution cancel/corrects.

An ExecutionRpt with ExecType = Restated represents an ExecutionRpt sent by the sellside communicating a change in the order or a restatement of the order's parameters without an electronic request from the customer. ExecRestatementReason must be set. This is used for GT orders and corporate actions (see below), changes communicated verbally to the sellside either due to normal business practices or as an emergency measure when electronic systems are not available, repricing of orders by the sellside (such as making Sell Short orders compliant with uptick / downtick rules), or other reasons (Broker option). ExecRestatementReason can also be used to communicate unsolicited cancels.

The field ClOrdID is provided for institutions or buy-side brokers or intermediaries to affix an identification number to an order to coincide with internal systems. The OrderID field is populated with the sell-side broker-generated order number (or fund manager-generated order number for CIVs). Unlike ClOrdID/OrigClOrdID which requires a chaining through Cancel/Replaces and Cancels, OrderID and SecondaryOrderID are not required to change through changes to an order.

The underlying business assumption of orders that can trade over multiple days, such as GTC and Good Till Date orders expiring on a future trading date (henceforth referred to as GT orders) is that a GT order that is not fully executed and has not been canceled and has not expired on a given day remains good for the broker to execute the following day. Note that the concept of "day" is determined by the market convention, which will be security specific. At the end of each trading day, once the order is no longer subject to execution, the broker may optionally send an Execution Report with ExecType=Done for Day(3). When the ExpireDate or ExpireTime of a Good Till Date order is reached, or a GTC order reaches a maximum age, the order is considered expired and the broker may optionally send an Execution Report with ExecType and OrdStatus=Expired(C).

In handling GT orders, the OrderQty, CumQty and AvgPx fields will represent the entirety of the order over all days. The fields DayOrderQty, DayCumQty, and DayAvgPx can be used on days following the day of the first trade on a GT order. Prior to the start of business each day, for all GT orders that have partial fills on previous days, DayCumQty and DayAvgPx are set to zero, and DayOrderQty becomes the LeavesQty. The following relationship holds: DayOrderQty = OrderQty – (CumQty – DayCumQty). Since (CumQty – DayCumQty)

represents the volume traded on all previous days, DayOrderQty = OrderQty – Volume traded on all previous days. Note that when changing the quantity of an order, both OrderQty and DayOrderQty will change. Requests to change or cancel an order will be made in terms of the total quantity for the order, not the quantity open today. For example, on an order where OrderQty=10000 and 2000 shares trade during the previous days, a request to change OrderQty to 15000 will mean that 13000 shares will be open. See "***Order State Change Matrices***" for examples of canceling and changing GT orders partially filled on previous days.

A Cancel on an execution (trade bust, ExecType = Trade Cancel) happening the same day of the trade will result in CumQty and DayCumQty each decreasing by the quantity busted, and LeavesQty increasing by the quantity busted. OrderQty and DayOrderQty will remain unchanged. If the business rules allow for a trade bust to be reported on a later date than the trade being busted, the OrderQty and DayCumQty will remain unchanged, the LeavesQty and DayOrderQty will increase by the quantity busted, and the CumQty will decrease by the quantity busted.

If bilaterally agreed between counterparties, a broker may wish to transmit a list of all open GT orders, permitting reconciliation of the open orders. Typically this transmission may occur at the end of the trading day or at the start of the following trading day. There is no expected response to such retransmission; in the event of a reconciliation problem this should be resolved manually or via the DK message. Assuming no corporate actions have occurred, the broker will send an Execution Report with ExecType = Restated (D) and ExecRestatementReason = GT renewal / restatement (no corporate action) (1) for each open GT order. These Execution Reports may have DayCumQty and DayAvgPx restated to zero, and DayOrderQty restated to LeavesQty if the transmission occurs at the start of the following business day. The broker has the option of changing the OrderID and SecondaryOrderID fields, or leaving them unchanged. If they are changed, then the buy-side should use these new ID fields when sending Order Cancel Request, Order Cancel/Replace Request, and Order Status Request messages.

In the case of a corporate action resulting in the adjustment of an open GT order, the broker will send an Execution Report with ExecType = Restated (D) and ExecRestatementReason = GT Corporate action (0) with the order's state after the corporate action adjustment. In the case of stock splits, OrderQty, CumQty, AvgPx, and LeavesQty will be adjusted to reflect the order's state in terms of current quantity (e.g. shares), not pre-split quantity (e.g. shares). See "***Order State Change Matrices***" for examples of GT order restatement with and without a corporate action.

CIV orders to be executed by the fund manager do not use the TimeInForce field and only a subset of OrdStatus values are expected to be used. ***See VOLUME 7 - "PRODUCT: COLLECTIVE INVESTMENT VEHICLES"*** for the CIV-specific OrdStatus values.

The Execution Report message is also used for multileg instrument. See "***Use of the Execution Report for Multileg Instruments***" for multileg-specific details.

The execution report message format is as follows:

Execution Report

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = 8
component block <ApplicationSequenceControl>		N	For use in drop copy applications. NOT FOR USE in transactional applications.

37	OrderID	Y	OrderID is required to be unique for each chain of orders.
198	SecondaryOrderID	N	Can be used to provide order id used by exchange or executing system.
526	SecondaryClOrdID	N	In the case of quotes can be mapped to: - QuoteID(117) of a single Quote - QuoteEntryID(299) of a Mass Quote.
527	SecondaryExecID	N	
11	ClOrdID	N	Required when referring to orders that where electronically submitted over FIX or otherwise assigned a ClOrdID(11). In the case of quotes can be mapped to: - QuoteMsgID(1166) of a single Quote - QuoteID(117) of a Mass Quote.
41	OrigClOrdID	N	Conditionally required for response to a Cancel or Cancel/Replace request (ExecType=PendingCancel, Replace, or Canceled) when referring to orders that where electronically submitted over FIX or otherwise assigned a ClOrdID(11). ClOrdID of the previous accepted order (NOT the initial order of the day) when canceling or replacing an order.
583	ClOrdLinkID	N	
693	QuoteRespID	N	Required if responding to a QuoteResponse message. Echo back the Initiator's value specified in the message.
790	OrdStatusReqID	N	Required if responding to and if provided on the Order Status Request message. Echo back the value provided by the requester.
584	MassStatusReqID	N	Required if responding to a Order Mass Status Request. Echo back the value provided by the requester.
961	HostCrossID	N	Host assigned entity ID that can be used to reference all components of a cross; sides + strategy + legs
911	TotNumReports	N	Can be used when responding to an Order Mass Status Request to identify the total number of Execution Reports which will be returned.
912	LastRptRequested	N	Can be used when responding to an Order Mass Status Request to indicate that this is the last Execution Reports which will be returned as a result of the request.
component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
229	TradeOriginationDate	N	
Start of Component block, expanded in line < ContraGrp >			

382	NoContraBrokers		N	Number of ContraBrokers repeating group instances.
→	375	ContraBroker	N	First field in repeating group. Required if NoContraBrokers > 0.
→	337	ContraTrader	N	
→	437	ContraTradeQty	N	
→	438	ContraTradeTime	N	
→	655	ContraLegRefID	N	
End of Component block, expanded in line < ContraGrp >				
66	ListID		N	Required for executions against orders which were submitted as part of a list.
548	CrossID		N	CrossID for the replacement order
551	OrigCrossID		N	Must match original cross order. Same order chaining mechanism as ClOrdID/OrigClOrdID with single order Cancel/Replace.
549	CrossType		N	
880	TrdMatchID		N	
17	ExecID		Y	Unique identifier of execution message as assigned by sell-side (broker, exchange, ECN) (will be 0 (zero) forExecType=I (Order Status)).
19	ExecRefID		N	Required for Trade Cancel and Trade Correct ExecType messages
150	ExecType		Y	Describes the purpose of the execution report.
39	OrdStatus		Y	Describes the current state of a CHAIN of orders, same scope as OrderQty, CumQty, LeavesQty, and AvgPx
636	WorkingIndicator		N	For optional use with OrdStatus = 0 (New)
103	OrdRejReason		N	For optional use with ExecType = 8 (Rejected)
378	ExecRestatementReason		N	Required for ExecType = D (Restated).
1	Account		N	Required for executions against electronically submitted orders which were assigned an account by the institution or intermediary
660	AcctIDSource		N	
581	AccountType		N	Specifies type of account
589	DayBookingInst		N	
590	BookingUnit		N	
591	PreallocMethod		N	
70	AllocID		N	
Start of Component block, expanded in line < PreAllocGrp >				
78	NoAllocs		N	Number of repeating groups for pre-trade allocation

→	79	AllocAccount	N	Required if NoAllocs > 0. Must be first field in repeating group.
→	661	AllocAcctIDSource	N	
→	736	AllocSettlCurrency	N	
→	467	IndividualAllocID	N	
→	component block <NestedParties>		N	Insert here the set of "Nested Parties" (firm identification "nested" within additional repeating group) fields defined in "Common Components of Application Messages" Used for NestedPartyRole=Clearing Firm
→	80	AllocQty	N	
End of Component block, expanded in line < PreAllocGrp >				
63	SettlType		N	
64	SettlDate		N	Takes precedence over SettlType value and conditionally required/omitted for specific SettlType values.
574	MatchType		N	
1115	OrderCategory		N	
544	CashMargin		N	
635	ClearingFeeIndicator		N	
component block <Instrument>			Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
component block <FinancingDetails>			N	Insert here the set of "FinancingDetails" (symbology) fields defined in "Common Components of Application Messages"
Start of Component block, expanded in line < UndInstrmtGrp >				
711	NoUnderlyings		N	Number of underlyings
→	component block <UnderlyingInstrument>		N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >				
54	Side		Y	
component block <Stipulations>			N	Insert here the set of "Stipulations" (repeating group of Fixed Income stipulations) fields defined in "Common Components of Application Messages"
854	QtyType		N	
component block <OrderQtyData>			N	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages" **IMPORTANT NOTE: OrderQty field is required for Single Instrument Orders unless rejecting or acknowledging an order for a CashOrderQty or PercentOrder. **

1093	LotType		N	
40	OrdType		N	
423	PriceType		N	
44	Price		N	Required if specified on the order
1092	PriceProtectionScope		N	
99	StopPx		N	Required if specified on the order
component block <TriggeringInstruction>			N	Insert here the set of "TriggeringInstruction" fields defined in "common components of application messages"
component block <PegInstructions>			N	Insert here the set of "PegInstruction" fields defined in "Common Components of Application Messages"
component block <DiscretionInstructions>			N	Insert here the set of "DiscretionInstruction" fields defined in "Common Components of Application Messages"
839	PeggedPrice		N	The current price the order is pegged at
1095	PeggedRefPrice		N	The reference price of a pegged order.
845	DiscretionPrice		N	The current discretionary price of the order
847	TargetStrategy		N	The target strategy of the order
Start of Component block, expanded in line < StrategyParametersGrp >				
957	NoStrategyParameters		N	Indicates number of strategy parameters
→	958	StrategyParameterName	N	Name of parameter
→	959	StrategyParameterType	N	Datatype of the parameter.
→	960	StrategyParameterValue	N	Value of the parameter
End of Component block, expanded in line < StrategyParametersGrp >				
848	TargetStrategyParameters		N	(Deprecated in FIX.5.0)For further specification of the TargetStrategy
849	ParticipationRate		N	(Deprecated in FIX.5.0)Mandatory for a TargetStrategy=Participate order and specifies the target particpation rate. For other order types optionally specifies a volume limit (i.e. do not be more than this percent of the market volume)
850	TargetStrategyPerformance		N	For communication of the performance of the order versus the target strategy
15	Currency		N	
376	ComplianceID		N	

377	SolicitedFlag	N	
59	TimeInForce	N	Absence of this field indicates Day order
168	EffectiveTime	N	Time specified on the order at which the order should be considered valid
432	ExpireDate	N	Conditionally required if TimeInForce = GTD and ExpireTime is not specified.
126	ExpireTime	N	Conditionally required if TimeInForce = GTD and ExpireDate is not specified.
18	ExecInst	N	Can contain multiple instructions, space delimited.
1057	AggressorIndicator	N	
528	OrderCapacity	N	
529	OrderRestrictions	N	
1091	PreTradeAnonymity	N	
582	CustOrderCapacity	N	
32	LastQty	N	Quantity (e.g. shares) bought/sold on this (last) fill. Required if ExecType = Trade or Trade Correct. If ExecType=Stopped, represents the quantity stopped/guaranteed/protected for.
1056	CalculatedCcyLastQty	N	Used for FX trades to express the quantity or amount of the other side of the currency. Conditionally required if ExecType = Trade or Trade Correct and is an FX trade.
1071	LastSwapPoints	N	Optionally used when ExecType = Trade or Trade Correct and is a FX Swap trade. Used to express the swap points for the swap trade event.
652	UnderlyingLastQty	N	
31	LastPx	N	Price of this (last) fill. Required if ExecType = Trade or Trade Correct. Should represent the "all-in" (LastSpotRate + LastForwardPoints) rate for F/X orders.). If ExecType=Stopped, represents the price stopped/guaranteed/protected at. Not required for FX Swap when ExecType = Trade or Trade Correct as there is no "all-in" rate that applies to both legs of the FX Swap.
651	UnderlyingLastPx	N	
669	LastParPx	N	Last price expressed in percent-of-par. Conditionally required for Fixed Income trades when LastPx is expressed in Yield, Spread, Discount or any other price type that is not percent-of-par.
194	LastSpotRate	N	Applicable for F/X orders
195	LastForwardPoints	N	Applicable for F/X orders

30	LastMkt		N	If ExecType = Trade (F), indicates the market where the trade was executed. If ExecType = New (0), indicates the market where the order was routed.
336	TradingSessionID		N	
625	TradingSessionSubID		N	
943	TimeBracket		N	
29	LastCapacity		N	
151	LeavesQty		Y	Quantity open for further execution. If the OrdStatus is Canceled, DoneForTheDay, Expired, Calculated, or Rejected (in which case the order is no longer active) then LeavesQty could be 0, otherwise LeavesQty = OrderQty - CumQty.
14	CumQty		Y	Currently executed quantity for chain of orders.
6	AvgPx		N	Not required for markets where average price is not calculated by the market. Conditionally required otherwise.
424	DayOrderQty		N	For GT orders on days following the day of the first trade.
425	DayCumQty		N	For GT orders on days following the day of the first trade.
426	DayAvgPx		N	For GT orders on days following the day of the first trade.
1361	TotNoFills		N	Used to support fragmentation. Sum of NoFills across all messages with the same ExecID.
893	LastFragment		N	Indicates whether this is the last fragment in a sequence of message fragments. Only required where message has been fragmented.
Start of Component block, expanded in line < FillsGrp >				
1362	NoFills		N	Specifies the number of partial fills included in this Execution Report
→	1363	FillExecID	N	Unique identifier of execution as assigned by sell-side (broker, exchange, ECN). Must not overlap ExecID(17). Required if NoFills > 0
→	1364	FillPx	N	Price of this partial fill. Conditionally required if NoFills > 0. Refer to LastPx(31).
→	1365	FillQty	N	Quantity (e.g. shares) bought/sold on this partial fill. Required if NoFills > 0.
→	component block <NestedParties4>		N	Contraparty information
End of Component block, expanded in line < FillsGrp >				
427	GTBookingInst		N	States whether executions are booked out or accumulated on a partially filled GT order

75	TradeDate	N	Used when reporting other than current day trades.
60	TransactTime	N	Time the transaction represented by this ExecutionReport occurred
113	ReportToExch	N	
component block <CommissionData>		N	Insert here the set of "CommissionData" fields defined in "Common Components of Application Messages" Note: On a fill/partial fill messages, it represents value for that fill/partial fill. On ExecType=Calculated, it represents cumulative value for the order. Monetary commission values are expressed in the currency reflected by the Currency field.
component block <SpreadOrBenchmarkCurveData>		N	Insert here the set of "SpreadOrBenchmarkCurveData" (Fixed Income spread or benchmark curve) fields defined in "Common Components of Application Messages"
component block <YieldData>		N	Insert here the set of "YieldData" (yield-related) fields defined in "Common Components of Application Messages"
381	GrossTradeAmt	N	
157	NumDaysInterest	N	
230	ExDate	N	
158	AccruedInterestRate	N	
159	AccruedInterestAmt	N	
738	InterestAtMaturity	N	For fixed income products which pay lump-sum interest at maturity.
920	EndAccruedInterestAmt	N	For repurchase agreements the accrued interest on termination.
921	StartCash	N	For repurchase agreements the start (dirty) cash consideration
922	EndCash	N	For repurchase agreements the end (dirty) cash consideration
258	TradedFlatSwitch	N	
259	BasisFeatureDate	N	
260	BasisFeaturePrice	N	
238	Concession	N	
237	TotalTakedown	N	
118	NetMoney	N	Note: On a fill/partial fill messages, it represents value for that fill/partial fill, on ExecType=Calculated, it represents cumulative value for the order. Value expressed in the currency reflected by the Currency field.
119	SettlCurrAmt	N	Used to report results of forex accommodation trade

120	SettlCurrency	N	Used to report results of forex accommodation trade
155	SettlCurrFxRate	N	Foreign exchange rate used to compute SettlCurrAmt from Currency to SettlCurrency
156	SettlCurrFxRateCalc	N	Specifies whether the SettlCurrFxRate should be multiplied or divided
21	HandlInst	N	
110	MinQty	N	
1089	MatchIncrement	N	
1090	MaxPriceLevels	N	
component block <DisplayInstruction>		N	Insert here the set of "DisplayInstruction" fields defined in "common components of application messages"
111	MaxFloor	N	
77	PositionEffect	N	For use in derivatives omnibus accounting
210	MaxShow	N	(Deprecated in FIX.5.0)
775	BookingType	N	Method for booking out this order. Used when notifying a broker that an order to be settled by that broker is to be booked out as an OTC derivative (e.g. CFD or similar). Absence of this field implies regular booking.
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
193	SettlDate2	N	(Deprecated in FIX.5.0)Can be used with OrdType = "Forex - Swap" to specify the "value date" for the future portion of a F/X swap.
192	OrderQty2	N	(Deprecated in FIX.5.0)Can be used with OrdType = "Forex - Swap" to specify the order quantity for the future portion of a F/X swap.
641	LastForwardPoints2	N	Can be used with OrdType = "Forex - Swap" to specify the forward points (added to LastSpotRate) for the future portion of a F/X swap.
442	MultiLegReportingType	N	Default is a single security if not specified.
480	CancellationRights	N	For CIV - Optional
481	MoneyLaunderingStatus	N	
513	RegistID	N	Reference to Registration Instructions message for this Order.
494	Designation	N	Supplementary registration information for this Order
483	TransBkdTime	N	For CIV - Optional

515	ExecValuationPoint			N	For CIV - Optional
484	ExecPriceType			N	For CIV - Optional
485	ExecPriceAdjustment			N	For CIV - Optional
638	PriorityIndicator			N	
639	PriceImprovement			N	
851	LastLiquidityInd			N	Applicable only on OrdStatus of Partial or Filled.
Start of Component block, expanded in line < ContAmtGrp >					
518	NoContAmts			N	Number of contract details in this message (number of repeating groups to follow)
→	519	ContAmtType		N	Must be first field in the repeating group.
→	520	ContAmtValue		N	
→	521	ContAmtCurr		N	
End of Component block, expanded in line < ContAmtGrp >					
Start of Component block, expanded in line < InstrmtLegExecGrp >					
555	NoLegs			N	Number of legs Identifies a Multi-leg Execution if present and non-zero.
→	component block <InstrumentLeg>			N	Must be provided if Number of legs > 0
→	687	LegQty		N	
→	685	LegOrderQty		N	When reporting an Execution, LegOrderQty may be used on Execution Report to echo back original LegOrderQty submission. This field should be used to specify OrderQty at the leg level rather than LegQty (deprecated).
→	690	LegSwapType		N	Instead of LegQty – requests that the sellside calculate LegQty based on opposite Leg
→	component block <LegStipulations>			N	
→	1366	LegAllocID		N	
→	*Start of Component block, expanded in line < LegPreAllocGrp >*				
→	670	NoLegAllocs		N	
→	→	671	LegAllocAccount	N	
→	→	672	LegIndividualAllocID	N	
→	→	component block <NestedParties2>		N	
→	→	673	LegAllocQty	N	

→	→	674	LegAllocAcctIDSource	N	
→	→	1367	LegAllocSettlCurrency	N	
→	*End of Component block, expanded in line < LegPreAllocGrp >*				
→	564	LegPositionEffect		N	Provide if the PositionEffect for the leg is different from that specified for the overall multileg security
→	565	LegCoveredOrUncovered		N	Provide if the CoveredOrUncovered for the leg is different from that specified for the overall multileg security.
→	component block <NestedParties3>			N	
→	654	LegRefID		N	Used to identify a specific leg.
→	587	LegSettlType		N	
→	588	LegSettlDate		N	Takes precedence over LegSettlType value and conditionally required/omitted for specific LegSettlType values.
→	637	LegLastPx		N	Used to report the execution price assigned to the leg of the multileg instrument
→	675	LegSettlCurrency		N	
→	1073	LegLastForwardPoints		N	
→	1074	LegCalculatedCcyLastQty		N	
→	1075	LegGrossTradeAmt		N	For FX Futures can be used to express the notional value of a trade when LegLastQty and other quantity fields are expressed in terms of number of contracts - LegContractMultiplier (231) is required in this case.
→	1379	LegVolatility		N	
→	1381	LegDividendYield		N	
→	1383	LegCurrencyRatio		N	
→	1384	LegExecInst		N	
→	1418	LegLastQty		N	
End of Component block, expanded in line < InstrmtLegExecGrp >					
797	CopyMsgIndicator			N	
Start of Component block, expanded in line < MiscFeesGrp >					
136	NoMiscFees			N	Required if any miscellaneous fees are reported. Indicates number of repeating entries. Repeating group. ** Nested Repeating Group follows **
→	137	MiscFeeAmt		N	Required if NoMiscFees > 0

→	138	MiscFeeCurr	N	
→	139	MiscFeeType	N	Required if NoMiscFees > 0
→	891	MiscFeeBasis	N	
End of Component block, expanded in line < MiscFeesGrp >				
1380	DividendYield		N	
1028	ManualOrderIndicator		N	
1029	CustDirectedOrder		N	
1030	ReceivedDeptID		N	
1031	CustOrderHandlingInst		N	
1032	OrderHandlingInstSource		N	
component block <TrdRegTimestamps>			N	
1188	Volatility		N	
1189	TimeToExpiration		N	
1190	RiskFreeRate		N	
811	PriceDelta		N	
StandardTrailer			Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element ExctnRpt

Use of the Execution Report for Multileg Instruments:

The Execution Report has been expanded to include an optional repeating group of instruments legs. The instrument leg repeating group will not be used to track order status (LeavesQty, CumQty, etc.). The instrument leg repeating group can be used to report:

- Each leg of a multileg instrument – this provides a method for data enrichment for productized multileg instruments that can be identified on orders using only the Instrument block.
- The user supplied per leg information for Party block, PositionEffect, CoveredUncovered
- To report the price specified by the user on the order.
- Reporting of last sales price per leg, settlement type, and settlement date.

The multileg repeat group cannot be used to report the following:

- fill quantity per leg
- order status per leg

There are three different ways strategies can be traded on markets.

1. As a product identified by an Instrument block in which all legs of a multileg instrument are traded atomically in the ratio quantities specified for leg where contraparties to the trade are also apportioned per the ratio quantities defined per leg. (Note this method applies to strategies that are or will be productized in the securities definition table)
2. As a product identified by an Instrument block in which all legs of a multileg instrument are traded, but they are traded against individual legs - likely resulting in contraparty trading quantities not corresponding to the ratio quantities. (Note this method applies to strategies that are or will be productized in the securities definition table)
3. As individual legs (legging in). (Note this method applies to strategies that are **not** and will **not** be productized in the securities definition table)

Multileg Instruments that are traded atomically and contraparties to the trade being assigned by ratio quantity can be reported by strategy by setting the MultilegReportType (442) field to 3. The OrdQty, LeavesQty, CumQty, AvgPx apply to the overall strategy. Quantities of each individual leg are calculated by multiplying the quantity field for the strategy quantity * the LegRatioQty.

Multileg Instruments that are not traded atomically (because they execute against orders and quotes for individual leg securities or they are traded on an open outcry environment by leg) can:

- Report fills by overall strategy and legs in a single Execution report, where instrument identification is in the Instrument Block and the leg instrument identification is in the Instrument Leg Block. The MultilegReportType field is 3. The OrdQty, LeavesQty, CumQty, AvgPx always apply to the strategy. Reporting **must** be done within the context of the strategy (ie: fills and partial fills are reported within the ratio quantities defined by the legs) even though contraparties have traded against individual legs and **perhaps not** within the ratio quantities defined by the legs. The LegRefID and ContraLegRefID are used to associate specific contra trade quantities against a leg with a specific contra party; or
- Counterparties can choose to send a summary Execution Report for the overall multileg instrument (MultilegReportType of 3) once the multileg order has been filled or partially filled, and then separately report details of each leg in separate Execution Reports. (MultilegReportType of 2). The OrdQty, LeavesQty, CumQty, AvgPx always apply to the **strategy**. Reporting **must** be done within the context of the strategy (ie:

fills and partial fills are reported within the ratio quantities defined by the legs) even though contraparties have traded against individual legs and **perhaps not** within the ratio quantities defined by the legs.

- The summary Execution Report is within the context of the strategy. Instrument identification is in the Instrument Block. This summary report does not contain leg information nor contraparty information. For ExecTypes = Pending New and New only the summary execution report should be sent.
- The separate Execution Report for each leg is within the context of a single leg of the strategy. Leg instrument identification is in the Instrument Leg Block. These reports contain the contraparty information for each leg.
- The ExecType of each separate leg report should be the same as the ExecType stated in the summary Exeeution Report; or
- Counterparties can choose to report fills by leg (without a summary Execution Report for the overall strategy). The MultilegReportType field is 2. Reporting **should** be done within the context of the strategy (ie: fills and partial fills are reported within the ratio quantities defined by the legs) even though contraparties have traded against individual legs and **perhaps not** within the ratio quantities defined by the legs.
- The Execution Report for each leg is within the context of a single leg of the strategy. Leg instrument identification is in the Instrument Leg Block. These reports contain the contraparty information for each leg. The OrdQty, LeavesQty, CumQty, AvgPx always apply to the strategy. Because a summary Exeeution Report is not being sent, ExecType = Pending New and New will also have to be reported by leg.
- If reporting of leg fills is **not** done within the context of the strategy, leg instrument identification and details should be **promoted to the Instrument Block**. Also, the OrdQty, LeavesQty, CumQty, AvgPx then apply to the individual leg. The MultilegReportType remains 2. … Always refer to the customs and practices of specific marketplaces to determine whether a specific marketplace permits reporting fills that are not within the context of the strategy and under what conditions such reporting is may be allowed.

Don’t Know Trade (DK)

The Don’t Know Trade (DK) message notifies a trading partner that an electronically received execution has been rejected. This message can be thought of as an execution reject message.

This message has special utility when dealing with one-way execution reporting. If the initial Order Acknowledgment message (Execution Report with LastQty=0 and OrdStatus=New) does not match an existing order this message can be used to notify the broker of a potential problem order.

Note that the decision to DK an execution lies with the institution. Some of the mismatches listed in the DKReason field may be acceptable and will not require a DK messages to be generated.

The Don’t Know Trade (DK) format is as follows:

Don’t Know Trade (DK)

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = Q
37	OrderID	Y	Broker Order ID as identified on problem execution
198	SecondaryOrderID	N	
17	ExecID	Y	Execution ID of problem execution
127	DKReason	Y	
component block <Instrument>		Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
Start of Component block, expanded in line < UndInstrmtGrp >			
711	NoUnderlyings	N	Number of underlyings
→	component block <UnderlyingInstrument>	N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >			
Start of Component block, expanded in line < InstrmtLegGrp >			
555	NoLegs	N	Number of legs
→	component block <InstrumentLeg>	N	Must be provided if Number of legs > 0
End of Component block, expanded in line < InstrmtLegGrp >			
54	Side	Y	
component block <OrderQtyData>		Y	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages"
32	LastQty	N	Required if specified on the ExecutionRpt
31	LastPx	N	Required if specified on the ExecutionRpt
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.

355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element DKTrd

Execution Report Acknowledgement

The Execution Report Acknowledgement message is an optional message that provides dual functionality to notify a trading partner that an electronically received execution has either been accepted or rejected (DK'd).

The DK portion of this message does not replace the existing DK Trade message for users who have already implemented the DK Trade message. For users who have not implemented the DK Trade message, through this single message they will be able to accept and DK an execution report. Users who wish to continue to use the DK Trade but also want a means to explicitly accept an execution report can also use this message to accept the execution report.

Execution Report Acknowledgement

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = BN
37	OrderID	Y	
198	SecondaryOrderID	N	
11	ClOrdID	N	Conditionally required if the Execution Report message contains a ClOrdID.
1036	ExecAckStatus	Y	Indicates the status of the execution acknowledgement. The "received, not yet processed" is an optional intermediary status that can be used to notify the counterparty that the Execution Report has been received.
17	ExecID	Y	The ExecID of the Execution Report being acknowledged.
127	DKReason	N	Conditionally required when ExecAckStatus = 2 (Don't know / Rejected).
component block <Instrument>		Y	
Start of Component block, expanded in line < UndInstrmtGrp >			
711	NoUnderlyings	N	Number of underlyings
→	component block <UnderlyingInstrument>	N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >			
Start of Component block, expanded in line < InstrmtLegGrp >			
555	NoLegs	N	Number of legs
→	component block <InstrumentLeg>	N	Must be provided if Number of legs > 0
End of Component block, expanded in line < InstrmtLegGrp >			
54	Side	Y	
component block <OrderQtyData>		Y	
32	LastQty	N	Conditionally required if specified on the Execution Report

31	LastPx	N	Conditionally Required if specified on the Execution Report
423	PriceType	N	Conditionally required if specified on the Execution Report
669	LastParPx	N	Conditionally required if specified on the Execution Report
14	CumQty	N	Conditionally required if specified on the Execution Report
6	AvgPx	N	Conditionally required if specified on the Execution Report
58	Text	N	Conditionally required if DKReason = "other"
354	EncodedTextLen	N	
355	EncodedText	N	
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element ExecutionAcknowledgement

Using the Execution Report Ack

This message is an **optional** message used by the Initiator to explicitly acknowledge the execution information conveyed in the Execution Report message sent from the Respondent. By sending this message the Initiator is notifying the Respondent that the Initiator agrees to the terms or details in the Execution Report-Trade message. This allows Initiators who do not implement the FIX Allocation message set (which acts as an implict acceptance of the executed order) or who sends FIX Allocations at a much later time to notify the Respondent (e.g. sell-side) that the execution details are accepted by the Initiator (e.g. buy-side).

Additionally, firms who conduct transactions via phone and receive one-way execution reporting via FIX would also benefit from this message. Once the Initiator (e.g. buy-side) receives the FIX Execution Report with the terms of the trade the Initiator can explictly agree to and accept the trade by sending this message.

In the "acceptance" mode the Execution Report Ack can be used as a response to Execution Report of Trade-partial fill, Trade-full fill, Trade-done for day, and Trade Correct.

The diagrams below illustrates the various example flows that this message can be used in.

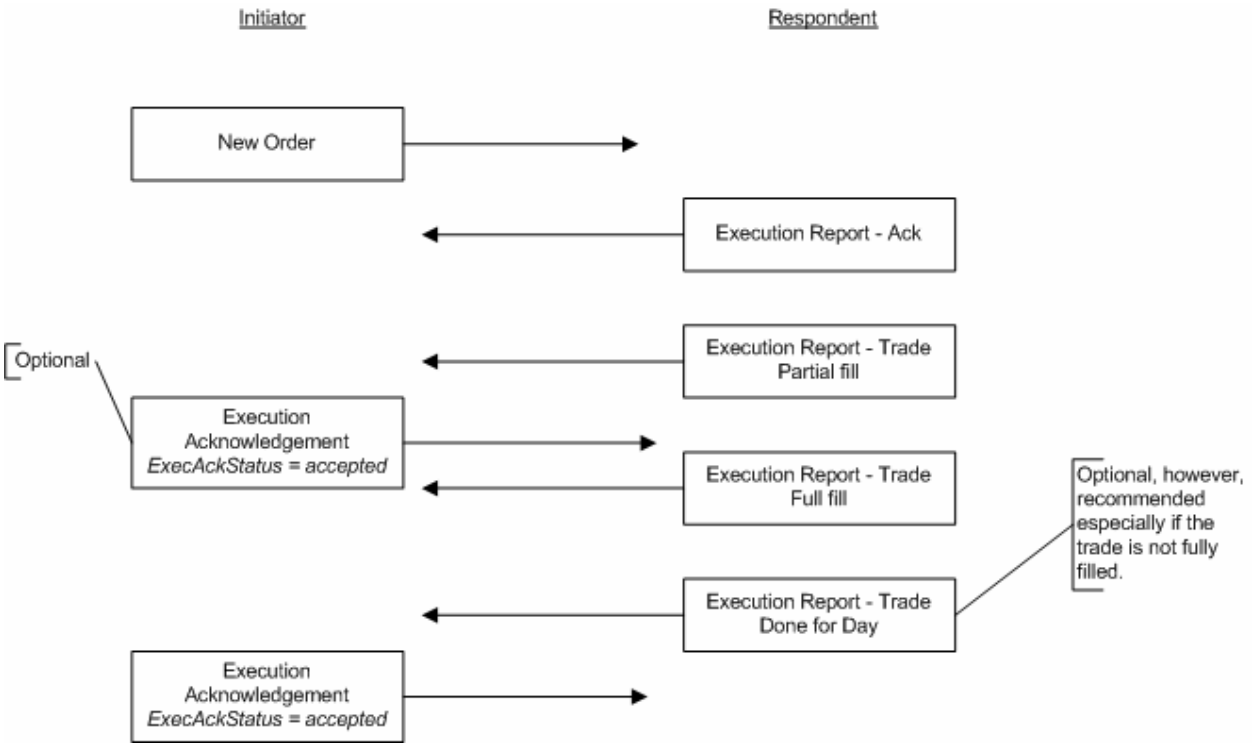


Figure 1: Execution Report Acknowledgement with Order/Execution

Figure 1 shows a flow whereby the order to be executed is placed via FIX. Depending on the asset type (e.g. equities, fixed income or FX) the order may result in a full fill or multiple partial fills until the order is filled. In some cases the Respondent may not be able to fully fill the order. The Execution Report Acknowledgement's primary function is to allow the Initiator to convey agreement to the execution details, thus sending it after the Execution Report indicating a full fill or "done for day" is most appropriate. However, if the Initiator wishes to send the Execution Report Acknowledgement after each partial fill, this should be agreed upon with the Respondent.

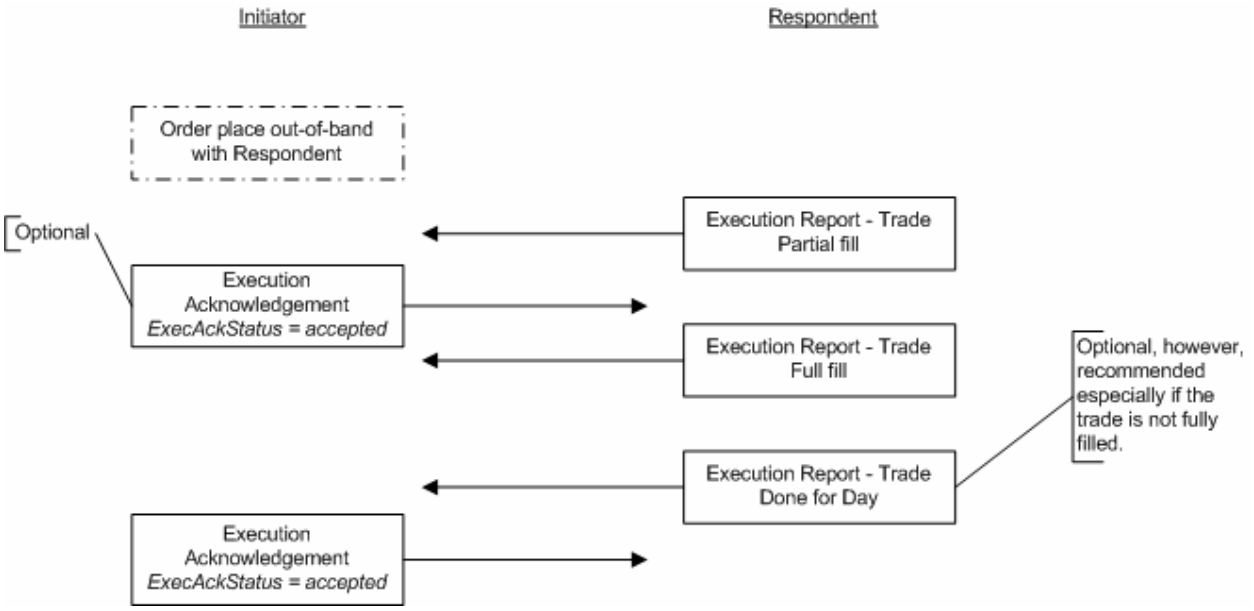


Figure 2: Execution Report Acknowledgement without Order via FIX

The message flow in Figure 2 is identical to Figure 1 with the exception that the Initiator has placed the order with the Respondent via means other than FIX. This could be via phone or via a trading platform's user interface.

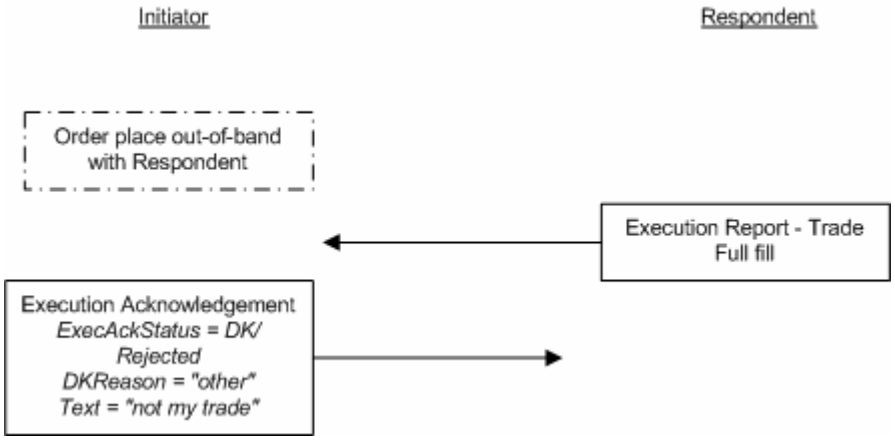


Figure 3: Execution Report Acknowledgement as a DK

Figure 3 illustrations the use of the Execution Report Acknowledgement being used in the "DK" mode to DK an execution report the Initiator does not accept or recognize.

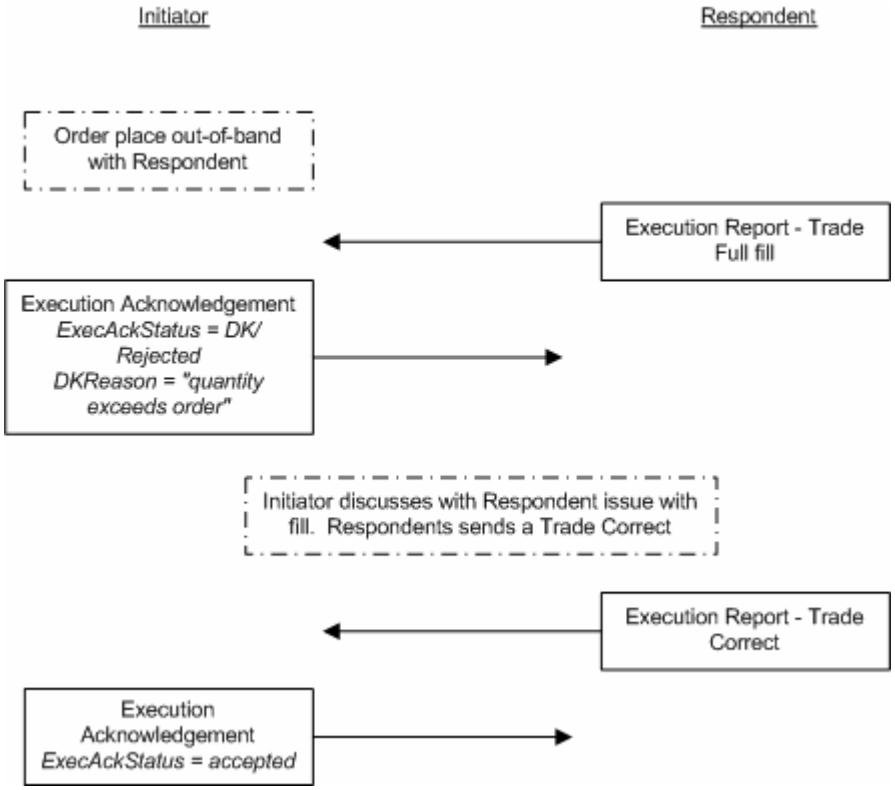


Figure 4: Execution Report Acknowledgement with Trade Correction

Figure 4 illustratins an example where an order was placed out of band, but the Initiator had to "DK" the execution report due to filled quantity mismatch. The Initiator resolves the issues with the Respondent via voice (or email) and the Respondent sends an Execution Report - Trade Correct. The Initiator accepts the change with the Execution Report Acknowledgement.

Using the Execution Report Ack with DK Trade

The Execution Report Acknowledgement can also be used in conjunction with users who have already implemented the DK Trade message but would like to make use of the Execution Report Acknowledgement's "accept" mode to explicitly notify the Respondent that the execution details are accepted.

Figure 5 below illustrations an example where by a voice order was executed and the fill was reported via FIX. The first instance the Initiator has DK the fill. Another order was placed via phone and this time the Initiator accepts the fill from the Resondent.

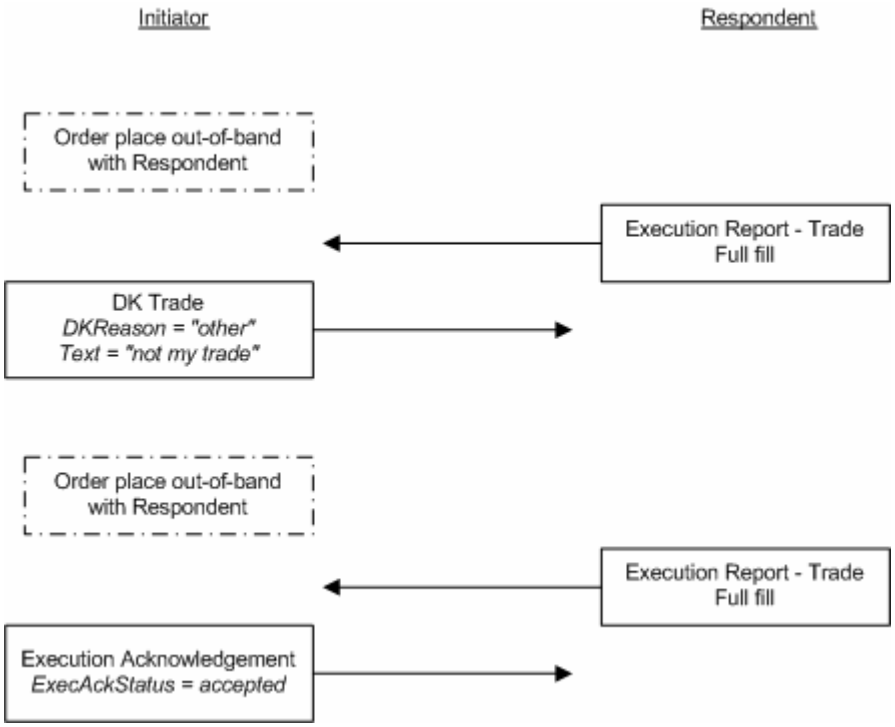


Figure 5: Execution Report Acknowledgement in conjunction with a DK Trade

Order Cancel/Replace Request (a.k.a. Order Modification Request)

The order cancel/replace request is used to change the parameters of an existing order.

Do not use this message to cancel the remaining quantity of an outstanding order, use the Order Cancel Request message for this purpose.

Cancel/Replace will be used to change any valid attribute of an open order (i.e. reduce/increase quantity, change limit price, change instructions, etc.), Subject to agreement between counterparties, it can be used to re-open a filled order by increasing OrderQty.

An immediate response to this message is required. It is recommended that an ExecutionRpt with ExecType=Pending Replace be sent unless the Order Cancel/Replace Request can be immediately accepted (ExecutionRpt with ExecType=Replace) or rejected (Order Cancel Reject message).

The Cancel/Replace request will only be accepted if the order can successfully be pulled back from the exchange floor without executing. Requests which cannot be processed will be rejected using the Cancel Reject message. The Cancel Reject message should provide the ClOrdID and OrigClOrdID values which were specified on the Cancel/Replace Request message for identification.

Note that while it is necessary for the ClOrdID to change and be unique, the broker's OrderID field does not necessarily have to change as a result of the Cancel/Replace request.

The protocol supports the chaining of multiple cancel/replace requests, though trading counterparties may not support this functionality. Care should be taken if the order sender wishes to send a cancel/replace request when there is one or more cancel/replaces which have not been accepted or rejected – in general:

- The order sender should chain client order ids on an 'optimistic' basis, i.e. set the OrigClOrdID to the last non rejected ClOrdID sent
- The order receiver should chain client order ids on a 'pessimistic' basis, i.e. set the OrigClOrdID on execution reports that convey the receipt or succesful application of a cancel/replace and Order Cancel Reject messages to be the last 'accepted' ClOrdID (See "***Order State Change Matrices***" for examples of this)

In the event that the order sender wants to chain order cancel/replaces rapidly then they should ensure that each replace request contains the full details of the order as they would now like it to be. For example if an attempt is made to change the limit price and then an immediate request to change the quantity is issued then if the desired behaviour is that both the limit price and quantity should be changed then the second request should include the revised limit price (in case the first replace request is rejected).

All of the application-level fields in the original order should be retransmitted with the original values in the Order Cancel/Replace Request, except the fields that are being changed. Any field may be changed with this message except those in the <Instrument> component block and limited changes to the Side field (noted below), however, buy-side firms should note that sell-side firms may further restrict which fields they allow to change; hence bilateral agreement is required. For example, some sell-side firms may not allow fields such as Side, SettlDate, etc. to change. Sell-side firms should validate the Order Cancel/Replace Request to ensure that the client is not requesting a change for a field that the sell-side cannot change; in this case the sell-side should send a Cancel Reject message with CxlRejReason = 2 (Broker/Exchange Option).

When modifying ExecInst values in a replacement order, it is necessary to re-declare all ExecInst in the replacement order. ExecInst values will not be carried forward from the original order to the replacement unless re-declared.

The format of the Order Cancel/Replace Request message is:

Order Cancel/Replace Request (a.k.a. Order Modification Request)

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = G
37	OrderID	N	Unique identifier of most recent order as assigned by sell-side (broker, exchange, ECN).
component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
229	TradeOriginationDate	N	
75	TradeDate	N	
41	OrigClOrdID	N	ClOrdID(11) of the previous non rejected order (NOT the initial order of the day) when canceling or replacing an order. Required when referring to orders that where electronically submitted over FIX or otherwise assigned a ClOrdID
11	ClOrdID	Y	Unique identifier of replacement order as assigned by institution or by the intermediary with closest association with the investor.. Note that this identifier will be used in ClOrdID field of the Cancel Reject message if the replacement request is rejected.
526	SecondaryClOrdID	N	
583	ClOrdLinkID	N	
66	ListID	N	Required for List Orders
586	OrigOrdModTime	N	TransactTime of the last state change that occurred to the original order
1	Account	N	
660	AcctIDSource	N	
581	AccountType	N	
589	DayBookingInst	N	
590	BookingUnit	N	
591	PreallocMethod	N	
70	AllocID	N	Used to assign an overall allocation id to the block of preallocations
Start of Component block, expanded in line < PreAllocGrp >			
78	NoAllocs	N	Number of repeating groups for pre-trade allocation
→	79 AllocAccount	N	Required if NoAllocs > 0. Must be first field in repeating group.
→	661 AllocAcctIDSource	N	

→	736	AllocSettlCurrency	N	
→	467	IndividualAllocID	N	
→	component block <NestedParties>		N	Insert here the set of "Nested Parties" (firm identification "nested" within additional repeating group) fields defined in "Common Components of Application Messages" Used for NestedPartyRole=Clearing Firm
→	80	AllocQty	N	
End of Component block, expanded in line < PreAllocGrp >				
63	SettlType		N	
64	SettlDate		N	Takes precedence over SettlType value and conditionally required/omitted for specific SettlType values.
544	CashMargin		N	
635	ClearingFeeIndicator		N	
21	HandlInst		N	
18	ExecInst		N	Can contain multiple instructions, space delimited. Replacement order must be created with new parameters (i.e. original order values will not be brought forward to replacement order unless redefined within this message).
110	MinQty		N	
1089	MatchIncrement		N	
1090	MaxPriceLevels		N	
component block <DisplayInstruction>			N	Insert here the set of "DisplayInstruction" fields defined in "common components of application messages"
111	MaxFloor		N	(Deprecated in FIX.5.0)
100	ExDestination		N	
1133	ExDestinationIDSource		N	
Start of Component block, expanded in line < TrdgSesGrp >				
386	NoTradingSessions		N	Specifies the number of repeating TradingSessionIDs
→	336	TradingSessionID	N	Required if NoTradingSessions is > 0.
→	625	TradingSessionSubID	N	
End of Component block, expanded in line < TrdgSesGrp >				
component block <Instrument>			Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages" Must match original order
component block <FinancingDetails>			N	Insert here the set of "FinancingDetails" (symbology) fields defined in "Common Components of Application Messages"

			Must match original order
Start of Component block, expanded in line < UndInstrmtGrp >			
711	NoUnderlyings	N	Number of underlyings
→	component block <UnderlyingInstrument>	N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >			
54	Side	Y	Should match original order's side, however, if bilaterally agreed to the following groups could potentially be interchanged: Buy and Buy Minus Sell, Sell Plus, Sell Short, and Sell Short Exempt Cross, Cross Short, and Cross Short Exempt
60	TransactTime	Y	Time this order request was initiated/released by the trader or trading system.
854	QtyType	N	
component block <OrderQtyData>		Y	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages" Note: OrderQty value should be the "Total Intended Order Quantity" (including the amount already executed for this chain of orders)
40	OrdType	Y	
423	PriceType	N	
44	Price	N	Required for limit OrdTypes. For F/X orders, should be the "all-in" rate (spot rate adjusted for forward points). Can be used to specify a limit price for a pegged order, previously indicated, etc.
1092	PriceProtectionScope	N	
99	StopPx	N	Required for OrdType = "Stop" or OrdType = "Stop limit".
component block <TriggeringInstruction>		N	Insert here the set of "TriggeringInstruction" fields defined in "common components of application messages"
component block <SpreadOrBenchmarkCurveData>		N	Insert here the set of "SpreadOrBenchmarkCurveData" (Fixed Income spread or benchmark curve) fields defined in "Common Components of Application Messages"
component block <YieldData>		N	Insert here the set of "YieldData" (yield-related) fields defined in "Common Components of Application Messages"
component block <PegInstructions>		N	Insert here the set of "PegInstruction" fields defined in "Common Components of Application Messages"
component block		N	Insert here the set of "DiscretionInstruction" fields

<DiscretionInstructions>				defined in "Common Components of Application Messages"
847	TargetStrategy		N	The target strategy of the order
Start of Component block, expanded in line < StrategyParametersGrp >				
957	NoStrategyParameters		N	Indicates number of strategy parameters
→	958	StrategyParameterName	N	Name of parameter
→	959	StrategyParameterType	N	Datatype of the parameter.
→	960	StrategyParameterValue	N	Value of the parameter
End of Component block, expanded in line < StrategyParametersGrp >				
848	TargetStrategyParameters		N	(Deprecated in FIX.5.0)For further specification of the TargetStrategy
849	ParticipationRate		N	(Deprecated in FIX.5.0)Mandatory for a TargetStrategy=Participate order and specifies the target particpation rate. For other order types optionally specifies a volume limit (i.e. do not be more than this percent of the market volume)
376	ComplianceID		N	
377	SolicitedFlag		N	
15	Currency		N	Must match original order.
59	TimeInForce		N	Absence of this field indicates Day order
168	EffectiveTime		N	Can specify the time at which the order should be considered valid
432	ExpireDate		N	Conditionally required if TimeInForce = GTD and ExpireTime is not specified.
126	ExpireTime		N	Conditionally required if TimeInForce = GTD and ExpireDate is not specified.
427	GTBookingInst		N	States whether executions are booked out or accumulated on a partially filled GT order
component block <CommissionData>			N	Insert here the set of "CommissionData" fields defined in "Common Components of Application Messages"
528	OrderCapacity		N	
529	OrderRestrictions		N	
1091	PreTradeAnonymity		N	
582	CustOrderCapacity		N	
121	ForexReq		N	Indicates that broker is requested to execute a Forex accommodation trade in conjunction with the security

			trade.
120	SettlCurrency	N	Required if ForexReq = Y.
775	BookingType	N	Method for booking out this order. Used when notifying a broker that an order to be settled by that broker is to be booked out as an OTC derivative (e.g. CFD or similar). Absence of this field implies regular booking.
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
193	SettlDate2	N	(Deprecated in FIX.5.0)Can be used with OrdType = "Forex - Swap" to specify the "value date" for the future portion of a F/X swap.
192	OrderQty2	N	(Deprecated in FIX.5.0)Can be used with OrdType = "Forex - Swap" to specify the order quantity for the future portion of a F/X swap.
640	Price2	N	(Deprecated in FIX.5.0)Can be used with OrdType = "Forex - Swap" to specify the price for the future portion of a F/X swap.
77	PositionEffect	N	For use in derivatives omnibus accounting
203	CoveredOrUncovered	N	For use with derivatives, such as options
210	MaxShow	N	(Deprecated in FIX.5.0)
114	LocateReqd	N	Required for short sell orders
480	CancellationRights	N	For CIV - Optional
481	MoneyLaunderingStatus	N	
513	RegistID	N	Reference to Registration Instructions message for this Order.
494	Designation	N	Supplementary registration information for this Order
1028	ManualOrderIndicator	N	
1029	CustDirectedOrder	N	
1030	ReceivedDeptID	N	
1031	CustOrderHandlingInst	N	
1032	OrderHandlingInstSource	N	
component block <TrdRegTimestamps>		N	
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details

Refer to FIXML element OrdCxlRplcReq

Order Cancel Request

The order cancel request message requests the cancelation of **all** of the remaining quantity of an existing order. Note that the Order Cancel/Replace Request should be used to partially cancel (reduce) an order).

The request will only be accepted if the order can successfully be pulled back from the exchange floor without executing.

A cancel request is assigned a ClOrdID and is treated as a separate entity. If rejected, the ClOrdID of the cancel request will be sent in the Cancel Reject message, as well as the ClOrdID of the actual order in the OrigClOrdID field. The ClOrdID assigned to the cancel request must be unique amongst the ClOrdID assigned to regular orders and replacement orders.

An immediate response to this message is required. It is recommended that an ExecutionRpt with ExecType=Pending Cancel be sent unless the Order Cancel Request can be immediately accepted (ExecutionRpt with ExecType=Canceled) or rejected (Order Cancel Reject message).

The format of the cancel request message is:

Order Cancel Request

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = F
41	OrigClOrdID	N	ClOrdID(11) of the previous non-rejected order (NOT the initial order of the day) when canceling or replacing an order. Required when referring to orders that where electronically submitted over FIX or otherwise assigned a ClOrdID
37	OrderID	N	Unique identifier of most recent order as assigned by sell-side (broker, exchange, ECN).
11	ClOrdID	Y	Unique ID of cancel request as assigned by the institution.
526	SecondaryClOrdID	N	
583	ClOrdLinkID	N	
66	ListID	N	Required for List Orders
586	OrigOrdModTime	N	
1	Account	N	
660	AcctIDSource	N	
581	AccountType	N	
component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
component block <Instrument>		Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
component block <FinancingDetails>		N	Insert here the set of "FinancingDetails" (symbology)

			fields defined in "Common Components of Application Messages" Must match original order
Start of Component block, expanded in line < UndInstrmtGrp >			
711	NoUnderlyings	N	Number of underlyings
→	component block <UnderlyingInstrument>	N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >			
54	Side	Y	
60	TransactTime	Y	Time this order request was initiated/released by the trader or trading system.
component block <OrderQtyData>		Y	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages" Note: OrderQty = CumQty + LeavesQty (see exceptions above)
376	ComplianceID	N	
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element OrdCxlReq

Order Cancel Reject

The order cancel reject message is issued by the broker upon receipt of a cancel request or cancel/replace request message which cannot be honored. Requests to change price or decrease quantity are executed only when an outstanding quantity exists. Filled orders cannot be changed (i.e quantity reduced or price change. However, the broker/sellside may support increasing the order quantity on a currently filled order).

When rejecting a Cancel/Replace Request (or Cancel Request), the Cancel Reject message should provide the ClOrdID which was specified on the Cancel/Replace Request (or Cancel Request) message for identification, and the OrigClOrdId should be that of the last accepted order (except in the case of CxlRejReason = "Unknown Order".

When rejecting an Order Mass Cancel Request, the ClOrdID should be set to the ClOrdID value of the Order Mass Cancel Request. OrigClOrdID is not specified for a rejected Order Mass Cancel Requests

The execution message responds to accepted cancel request and cancel/replace request messages.

The order cancel reject message format is as follows:

Order Cancel Reject

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = 9
37	OrderID	Y	If CxlRejReason="Unknown order", specify "NONE".
198	SecondaryOrderID	N	Can be used to provide order id used by exchange or executing system.
526	SecondaryClOrdID	N	
11	ClOrdID	Y	Unique order id assigned by institution or by the intermediary with closest association with the investor. to the cancel request or to the replacement order.
583	ClOrdLinkID	N	
41	OrigClOrdID	N	ClOrdID(11) which could not be canceled/replaced. ClOrdID of the previous accepted order (NOT the initial order of the day) when canceling or replacing an order. Required when referring to orders that where electronically submitted over FIX or otherwise assigned a ClOrdID.
39	OrdStatus	Y	OrdStatus value after this cancel reject is applied. If CxlRejReason = "Unknown Order", specify Rejected.
636	WorkingIndicator	N	For optional use with OrdStatus = 0 (New)
586	OrigOrdModTime	N	
66	ListID	N	Required for rejects against orders which were submitted as part of a list.
1	Account	N	

660	AcctIDSource	N	
581	AccountType	N	
229	TradeOriginationDate	N	
75	TradeDate	N	
60	TransactTime	N	
434	CxlRejResponseTo	Y	
102	CxlRejReason	N	
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details

Refer to FIXML element OrdCxlRej

Order Status Request

The order status request message is used by the institution to generate an order status message back from the broker.

(See "***Order State Change Matrices***" for examples of usage of this message, including how to respond to a status request for an unknown order.)

The format of the order status request message is:

Order Status Request

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = H
37	OrderID	N	Conditionally required if ClOrdID(11) is not provided. Either OrderID or ClOrdID must be provided.
11	ClOrdID	N	The ClOrdID of the order whose status is being requested. Conditionally required if the OrderID(37) is not provided. Either OrderID or ClOrdID must be provided.
526	SecondaryClOrdID	N	
583	ClOrdLinkID	N	
component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
790	OrdStatusReqID	N	Optional, can be used to uniquely identify a specific Order Status Request message. Echoed back on Execution Report if provided.
1	Account	N	
660	AcctIDSource	N	
component block <Instrument>		Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
component block <FinancingDetails>		N	Insert here the set of "FinancingDetails" (symbology) fields defined in "Common Components of Application Messages" Must match original order
Start of Component block, expanded in line < UndInstrmtGrp >			
711	NoUnderlyings	N	Number of underlyings
→	component block <UnderlyingInstrument>	N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >			

54	Side	Y	
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element OrdStatReq

Order State Change Matrices

(formerly known as "Appendix D")

The following matrices are included to clarify the sequence of messages and the status of orders involved in the submission and processing of new orders, executions, cancel requests, cancel/replace requests and order status requests. The matrices have been arranged in groups as follows:

A Vanilla

Ref	Description	Old Reference(4.3 Errata)
A.1.a	Filled order	1
A.1.b	Part-filled day order, done for day	2

B Cancel

Ref	Description	Old Reference (4.3 Errata)
B.1.a	Cancel request issued for a zero-filled order	3
B.1.b	Cancel request issued for a part-filled order – executions occur whilst cancel request is active	4
B.1.c	Cancel request issued for an order that becomes filled before cancel request can be accepted	5
B.1.d	Cancel request issued for an order that has not yet been acknowledged	6
B.1.e	Cancel request issued for an order that has not yet been acknowledged – the acknowledgment and the cancel request 'cross'	7
B.1.f	Cancel request issued for an unknown order	7a

C Cancel/Replace quantity changes

C.1 Replace to increase quantity

Ref	Description	Old Reference (4.3 Errata)
C.1.a	Zero-filled order, cancel/replace request issued to increase order qty	8
C.1.b	Part-filled order, followed by cancel/replace request to increase order qty, execution occurs whilst order is pending replace	9
C.1.c	Filled order, followed by cancel/replace request to increase order quantity	10

C.2 Replace not for quantity change

Ref	Description	Old Reference (4.3 Errata)
C.2.a	Cancel/replace request (not for quantity change) is rejected as a fill has occurred	11

C.3 Replace to decrease quantity

Ref	Description	Old Reference (4.3 Errata)
C.3.a	Cancel/replace request sent whilst execution is being reported – the requested order qty exceeds the cum qty. Order is replaced then filled	12
C.3.b	Cancel/replace request sent whilst execution is being reported – the requested order qty equals the cum qty – order qty is amended to cum qty	13
C.3.c	Cancel/replace request sent whilst execution is being reported – the requested order qty is below cum qty – order qty is amended to cum qty	14

D Cancel/Replace sequencing and chaining

D.1 Sequencing

Ref	Description	Old Reference (4.3 Errata)
D.1.a	One cancel/replace request is issued which is accepted – another one is issued which is also accepted	15
D.1.b	One cancel/replace request is issued which is rejected before order becomes pending replace – then another one is issued which is accepted	16
D.1.c	One cancel/replace request is issued which is rejected after it is in pending replace – then another one is issued which is accepted	17

D.2 Chaining

Ref	Description	Old Reference (4.3 Errata)
D.2.a	One cancel/replace request is issued followed immediately by another – broker processes sequentially	18
D.2.b	One cancel/replace request is issued followed immediately by another – broker processes pending replaces before replaces	19
D.2.c	One cancel/replace request is issued followed immediately by another – both are rejected	20
D.2.d	One cancel/replace request is issued followed immediately by another – broker rejects the second as order is pending replace	21

E Unsolicited/Reinstatement

Ref	Description	Old Reference (4.3 Errata)
E.1.a	Telephoned order	22
E.1.b	Unsolicited cancel of a part-filled order	23
E.1.c	Unsolicited replacement of a part-filled order	24
E.1.d	Unsolicited reduction of order quantity by sell side (e.g. for US ECNs to communicate Nasdaq SelectNet declines)	25
E.1.e	Unsolicited cancel of 'cancel if not best' order	
E.1.f	Order is sent to exchange, held waiting for activation and then activated	

F Order Reject

Ref	Description	Old Reference (4.3 Errata)
F.1.a	Order rejected due to duplicate ClOrdID	26
F.1.b	Poss resend and duplicate ClOrdID	27
F.1.c	Order rejected because the order has already been verbally submitted	28

G Status

Ref	Description	Old Reference (4.3 Errata)
G.1.a	Order status request rejected for unknown order	29
G.1.b	Transmitting a CMS-style "Nothing Done" in response to a status request	30
G.1.c	Order sent, immediately followed by a status request. Subsequent status requests sent during life of order	31

H GT

Ref	Description	Old Reference (4.3 Errata)
H.1.a	GTC order partially filled, restated (renewed) and partially filled the following day	32
H.1.b	GTC order with partial fill, a 2:1 stock split then a partial fill and fill the following day	33
H.1.c	GTC order partially filled, restated(renewed) and canceled the following day	34
H.1.d	GTC order partially filled, restated(renewed) followed by replace request to increase quantity	35

I TimeInForce

Ref	Description	Old Reference (4.3 Errata)
I.1.a	Fill or Kill order cannot be filled	36
I.1.b	Immediate or Cancel order that cannot be immediately hit	37

J Execution Cancels/Corrects

Ref	Description	Old Reference (4.3 Errata)
J.1.a	Filled order, followed by correction and cancellation of executions	38
J.1.b	A canceled order followed by a busted execution and a new execution	39
J.1.c	GTC order partially filled, restated (renewed) and partially filled the following day, with corrections of quantity on both executions	40
J.1.d	Part-filled order Done for day followed by trade correction and bust	41

K Trading Halt

Ref	Description	Old Reference (4.3 Errata)
K.1.a	Trading Halt – Reinstate	43
K.1.b	Trading Halt – Cancel	44

L Miscellaneous

Ref	Description	Old Reference (4.3 Errata)
L.1.a	Transmitting a guarantee of execution prior to execution (Stopped/Guarantee)	42
L.1.b	Use of CashOrderQty	

The Table below shows which state transitions have been illustrated by the matrices in this Appendix (marked with an asterisk). The row represents the current value of OrdStatus and the column represents the next value as reported back to the buy-side via an execution report or order cancel reject message. Next to each OrdStatus value is its precedence – this is used when the order exists in a number of states simultaneously to determine the value that should be reported back. Note that absence of a scenario should not necessarily be interpreted as meaning that the state transition is not allowed:

OrdStatus (precedence value)	New (2)	Partially Filled (4)	Filled (8)	Done For Day (10)	Pending Cancel (12)	Pending Replace (11)		Canceled (5)	Rejected (2)	Stopped (7)
Pending New (2)	*								*	
New (2)	*	*	*	*	*	*			*	*
Partially Filled (4)		*	*	*	*	*		*		
Filled (8)		*	*			*				
Done for Day (10)		*								
Pending Cancel (12)	*	*	*		*			*		
Pending Replace (11)	*	*	*			*		*		
Canceled (5)										
Rejected (2)										
Stopped (7)		*								

How to read the Order State Change Matrices:

- The ‘Execution Report’ message is referred to simply as ‘Execution’
- The ‘Order Cancel/Replace Request’ and ‘Order Cancel Request’ messages are referred to as ‘Replace Request’ and ‘Cancel Request’ respectively
- The shaded rows represent messages sent from buy-side to the sell-side
- In general where two lines of a matrix share the same time, this means either
    - that there are two possible paths (e.g. a request is accepted or rejected) – in this case the first row of the two possible paths is the reject case which is italicized. The non-italicized row is the path that is continued by the remainder of the matrix
    - that two messages are being sent at the same time but in different directions such that the messages cross on the connection (e.g. a cancel request is sent at the same time as the sell-side is sending an execution) – in this case both lines have bold text
- For scenarios involving cancel requests or cancel/replace requests ‘ X’ refers to the original order, ‘Y’ refers to the cancel/replacing order. A similar convention is used for corrections or cancels to executions

A Vanilla

A.1.a - Filled order

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sales*
2		Execution(X)	New	New	10000	0	10000	0	
3		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by trader/exchange*
3		Execution(X)	Trade	Partially Filled	10000	2000	8000	2000	Execution of 2000
4		Execution(X)	Trade	Partially Filled	10000	3000	7000	1000	Execution of 1000
5		Execution(X)	Trade	Filled	10000	10000	0	7000	Execution of 7000

A.1.b – Part-filled day order, done for day

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	2000	8000	2000	Execution of 2000
4		Execution(X)	Trade	Partially Filled	10000	3000	7000	1000	Execution of 1000
5		Execution(X)	Done for Day	Done for Day	10000	3000	0	0	Assuming day order. See other examples which cover GT orders

B Cancel

B.1.a – Cancel request issued for a zero-filled order

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected*
2		Execution(X)	New	New	10000	0	10000	0	
3	Cancel Request(Y,X)				10000				
4		*Cancel Reject (Y,X)*		*New*					*If rejected by salesperson*
4		Execution (Y,X)	Pending Cancel	Pending Cancel	10000	0	10000	0	Aknowledge the cancel request
5		*Cancel Reject (Y,X)*		*New*					*If rejected by trader/exchange*
5		Execution (Y,X)	Canceled	Canceled	10000	0	0	0	Confirm that order has been canceled

B.1.b – Cancel request issued for a part-filled order – executions occur whilst cancel request is active

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	2000	8000	2000	Execution for 2000
4	**Cancel Request(Y,X)**				**10000**				
4		**Execution(X)**	**Trade**	**Partially Filled**	**10000**	**5000**	**5000**	**3000**	**Execution for 3000. This execution passes the cancel request on the connection**
5		*Cancel Reject (Y,X)*		*Partially Filled*					*If request is rejected*
5		Execution (Y,X)	Pending Cancel	Pending Cancel	10000	5000	5000	0	'Pending cancel' order status takes precedence over 'partially filled' order status
6		Execution(X)	Trade	Pending Cancel	10000	6000	4000	1000	Execution for 1000 whilst order is pending cancel – 'pending cancel' order status takes precedence over 'partially filled' order status.
7		*Cancel Reject (Y,X)*		*Partially Filled*					*If request is rejected*
7		Execution (Y,X)	Canceled	Canceled	10000	6000	0	0	'Canceled' order status takes precedence over 'partially filled' order status

B.1.c – Cancel request issued for an order that becomes filled before cancel request can be accepted

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	2000	8000	2000	Execution for 2000
4	**Cancel Request(Y,X)**				**10000**				
4		**Execution(X)**	**Trade**	**Partially Filled**	**10000**	**5000**	**5000**	**3000**	**Execution for 3000. This execution passes the cancel request on the connection**
5		*Cancel Reject (Y,X)*		*Partially Filled*					*If request is rejected*
5		Execution (Y,X)	Pending Cancel	Pending Cancel	10000	5000	5000	0	'Pending cancel' order status takes precedence over 'partially filled' order status
6		Execution(X)	Trade	Pending Cancel	10000	10000	0	5000	Execution for 5000 whilst order is pending cancel. 'Pending cancel' order status takes precedence over 'filled' order status
7		Cancel Reject (Y,X)		Filled					Cancel request rejected – CxlRejectReason = 0 (too late to cancel)

B.1.d – Cancel request issued for an order that has not yet been acknowledged

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2	Cancel Request(Y,X)				10000				Order sender immediately wishes to cancel the order
3		Execution (Y,X)	Pending Cancel	Pending Cancel	10000	0	10000	0	OrigClOrd set to X even though X has not yet been 'accepted'.
4		Execution (X)	New	New	10000	0	10000	0	Order accepted before cancel request is processed.
5		Execution (Y,X)	Canceled	Canceled	10000	0	0	0	Order canceled.
6	New Order(A)				5000				
7	Cancel Request(B,A)				5000				Order sender immediately wishes to cancel the order
8		Execution (B,A)	Pending Cancel	Pending Cancel	5000	0	5000	0	OrigClOrd set to A even though A has not yet been 'accepted'.
9		Execution (B,A)	Canceled	Canceled	5000	0	0	0	Order canceled before it is accepted. Note OrigClOrdID set to A even though A has not yet been accepted

B.1.e – Cancel request issued for an order that has not yet been acknowledged – the acknowledgment and the cancel request 'cross'

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2	**Cancel Request(Y,X)**				**10000**				**Order sender immediately wishes to cancel the order**
2		**Execution (X)**	**New**	**New**	**10000**	**0**	**10000**	**0**	**This message crosses the Cancel request**
3		Execution (Y,X)	Pending Cancel	Pending Cancel	10000	0	10000	0	
4		Execution (Y,X)	Canceled	Canceled	10000	0	0	0	Order canceled.

B.1.f – Cancel request issued for an unknown order

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	Cancel Request(Y,X)				10000				
2		Cancel Reject (Y,X)		Rejected					Cancel request rejected with reject reason of "Unknown Order", OrdStatus is "Rejected" and OrderID is "NONE"

NOTE: It is important to note that rejecting a cancel request for an unknown OrigClOrdID does not cause the sell-side to consume the OrigClOrdID used in the Cancel Request.

C Cancel/Replace quantity changes

C.1 Replace to increase quantity

C.1.a – Zero-filled order, cancel/replace request issued to increase order qty

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3	Replace Request(Y,X)				11000				Request to increase order qty to 11000
4		*Cancel Reject (Y,X)*		*New*					*If request is rejected by salesperson*
4		Execution (Y,X)	Pending Replace	Pending Replace	10000	0	10000	0	Acknowledge the Replace request
5		*Cancel Reject (Y,X)*		*New*					*If rejected by trader/exchange*
5		Execution (Y,X)	Replace	New	11000	0	11000	0	Confirm order has been replaced
6		Execution (Y)	Trade	Partially Filled	11000	1000	10000	1000	Execution for 1000. Use Y as the new ClOrdID.
7		Execution (Y)	Trade	Partially Filled	11000	3000	8000	2000	Execution for 2000

C.1.b – Part-filled order, followed by cancel/replace request to increase order qty, execution occurs whilst order is pending replace

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	Replace Request(Y,X)				12000				Request increase in order quantity to 12000
5		*Cancel Reject (Y,X)*		*Partially Filled*					*If request is rejected*
5		Execution (Y,X)	Pending Replace	Pending Replace	10000	1000	9000	0	'Pending replace' order status takes precedence over 'partially filled' order status
6		Execution(X)	Trade	Pending Replace	10000	1100	8900	100	Execution for 100 before cancel/replace request is dealt with
7		*Cancel Reject (Y,X)*		*Partially Filled*					*If request is rejected*
7		Execution (Y,X)	Replace	Partially Filled	12000	1100	10900	0	Confirm replace has been accepted
8		Execution(Y)	Trade	Filled	12000	12000	0	10900	Execution for 10900

C.1.c – Filled order, followed by cancel/replace request to increase order quantity

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Filled	10000	10000	0	10000	Execution for 10000
4	Replace Request(Y,X)				12000				Request increase in order quantity to 12000
5		*Cancel Reject (Y,X)*		*Filled*					*If request is rejected*
5		Execution (Y,X)	Pending Replace	Pending Replace	10000	10000	0	0	'Pending replace' order status takes precedence over 'partially filled' order status
6		*Cancel Reject (Y,X)*		*Filled*					*If request is rejected*
6		Execution (Y,X)	Replace	Partially Filled	12000	10000	2000	0	. Confirm order has been replaced
7		Execution(Y)	Trade	Filled	12000	12000	0	2000	Execution for 2000

C.2 Replace not for quantity change

C.2.a – Cancel/replace request (not for quantity change) is rejected as a fill has occurred

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	**Replace Request(Y,X)**				**10000**				**Assume in this scenario that client does not wish to increase qty (e.g. client wants to amend limit price)**
4		**Execution (X)**	**Trade**	**Filled**	**10000**	**10000**	**0**	**9000**	**Execution for 9000 – the replace request message and this execution report pass each other on the connection**
5		Cancel Reject (Y,X)		Filled					CxlRejectReason = 0 (too late to cancel)

C.3 Replace to decrease quantity

C.3.a – Cancel/replace request sent whilst execution is being reported – the requested order qty exceeds the cum qty. Order is replaced then filled

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	**Replace Request(Y,X)**				**8000**				**Request a decrease order quantity to 8000 (leaving 7000 open)**
4		**Execution(X)**	**Trade**	**Partially Filled**	**10000**	**1500**	**8500**	**500**	**Execution for 500 sent. Replace request and this execution report pass each other on the connection**
5		*Cancel Reject (Y,X)*		*Partially Filled*					*If request is rejected by salesperson*
5		Execution (Y,X)	Pending Replace	Pending Replace	10000	1500	8500	0	'Pending replace' order status takes precedence over 'partially filled' order status
6		Execution(X)	Trade	Pending Replace	10000	1600	8400	100	Execution for 100 occurs before cancel/replace request is accepted
7		*Cancel Reject (Y,X)*		*Partially Filled*					*If request is rejected by trader/exchange*
7		Execution (Y,X)	Replace	Partially Filled	8000	1600	6400	0	Replace is accepted as requested order qty exceeds cum qty
8		Execution (Y)	Trade	Filled	8000	8000	0	6400	Execution for 6400.

C.3.b – Cancel/replace request sent whilst execution is being reported – the requested order qty equals the cum qty – order qty is amended to cum qty

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3	**Replace Request(Y,X)**				**7000**				**Client wishes to amend order qty to 7000**
3		**Execution(X)**	**Trade**	**Partially Filled**	**10000**	**7000**	**3000**	**7000**	**Execution for 7000 - the replace message and this execution report pass each other on the connection**
4		Execution (Y,X)	Replace	Filled	7000	7000	0	0	The replace request is interpreted as requiring the balance of the order to be canceled – the 'filled' order status takes precedence over 'canceled'.

C.3.c – Cancel/replace request sent whilst execution is being reported – the requested order qty is below cum qty – order qty is amended to cum qty

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3	**Replace Request(Y,X)**				**7000**				**Client wishes to amend order qty to 7000**
3		**Execution(X)**	**Trade**	**Partially Filled**	**10000**	**8000**	**2000**	**8000**	**Execution for 8000 - the replace message and this execution report pass each other on the connection**
4		Execution (Y,X)	Replace	Filled	8000	8000	0	0	The replace request is interpreted as requiring the balance of the order to be canceled – the 'filled' order status takes precedence over 'canceled'.

D Cancel/Replace sequencing and chaining

D.1 Sequencing

D.1.a – One cancel/replace request is issued which is accepted – another one is issued which is also accepted

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	Replace Request(Y,X)				8000				Request decrease in order quantity to 8000, leaving 7000 open
5		Execution (Y,X)	Pending Replace	Pending Replace	10000	1000	9000	0	'Pending replace' order status takes precedence over 'partially filled' order status
6		Execution(X)	Trade	Pending Replace	10000	1500	8500	500	Execution for 500
7		Execution (Y,X)	Replace	Partially Filled	8000	1500	6500	0	
8		Execution (Y)	Trade	Partially Filled	8000	3500	4500	2000	Execution for 2000
9	Replace Request(Z,Y)				6000				Request decrease in order quantity to 6000, leaving 2500 open
10		Execution (Z,Y)	Pending Replace	Pending Replace	8000	3500	4500	0	
11		Execution(Y)	Trade	Pending Replace	8000	4000	4000	500	Execution for 500
12		Execution (Z,Y)	Replace	Partially Filled	6000	4000	2000	0	
13		Execution(Z)	Trade	Filled	6000	6000	0	2000	Execution for 2000

D.1.b – One cancel/replace request is issued which is rejected before order becomes pending replace – then another one is issued which is accepted

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	Replace Request(Y,X)				8000				Request decrease in order quantity to 8000, leaving 7000 open
5		Cancel Reject (Y,X)		Partially Filled					Request is rejected
6		Execution(X)	Trade	Partially Filled	10000	1500	8500	500	Execution for 500
7		Execution(X)	Trade	Partially Filled	10000	3500	6500	2000	Execution for 2000
8	Replace Request(Z,X)				6000				Request decrease in order quantity to 6000, leaving 2500 open. Note that OrigClOrdID = X , as this is the last non rejected ClOrdID
9		Execution (Z,X)	Pending Replace	Pending Replace	10000	3500	6500	0	Note that OrigClOrdID = X
10		Execution (Z,X)	Replace	Partially Filled	6000	3500	2500	0	Note that OrigClOrdID = X
11		Execution(Z)	Trade	Partially Filled	6000	5000	1000	1500	Execution for 1500

D.1.c - One cancel/replace request is issued which is rejected after it is in pending replace – then another one is issued which is accepted

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	Replace Request(Y,X)				8000				Request decrease in order quantity to 8000, leaving 7000 open
5		Execution (Y,X)	Pending Replace	Pending Replace	10000	1000	9000	0	
6		Execution(X)	Trade	Pending Replace	10000	1500	8500	500	Execution for 500. 'Pending replace' order status takes precedence over 'partially filled' order status
7		Cancel Reject (Y,X)		Partially Filled					Request is rejected (e.g. by trader/exchange)
8		Execution(X)	Trade	Partially Filled	10000	3500	6500	2000	Execution for 2000
9	Replace Request(Z,X)				6000				Request decrease in order quantity to 6000, leaving 2500 open. Note that OrigClOrdID = X as this is the last non rejected ClOrdID
10		Execution (Z,X)	Pending Replace	Pending Replace	10000	3500	6500	0	
11		*Cancel Reject (Z,X)*		*Partially Filled*					*If request is rejected (e.g. by trader/exchange)*
11		Execution (Z,X)	Replace	Partially Filled	6000	3500	2500	0	
12		Execution(Z)	Trade	Partially Filled	6000	5000	1000	1500	Execution for 1500

D.2 Chaining

D.2.a – One cancel/replace request is issued followed immediately by another – broker processes sequentially

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	Replace Request(Y,X)				8000				Request decrease in order quantity to 8000, leaving 7000 open
5	Replace Request(Z,Y)				7000				Request decrease in order quantity to 7000, leaving 6000 open. Note OrigClOrdID set to last non rejected ClOrdID i.e. Y (on an 'optimistic' basis)
6		Execution (Y,X)	Pending Replace	Pending Replace	10000	1000	9000	0	Broker processes Replace (Y,X) first
7		Execution (Y,X)	Replace	Partially Filled	8000	1000	7000	0	Broker processes Replace (Y,X) first
8		Execution (Z,Y)	Pending Replace	Pending Replace	8000	1000	7000	0	Broker then processes Replace (Z,Y). Note OrigClOrdID set to last accepted ClOrdID i.e. Y
9		Execution (Z,Y)	Replace	Partially Filled	7000	1000	6000	0	Broker then processes Replace (Z,Y)
10		Execution(Z)	Trade	Filled	7000	7000	0	6000	Execution for 6000

D.2.b – One cancel/replace request is issued followed immediately by another – broker processes pending replaces before replaces

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	Replace Request(Y,X)				8000				Request decrease in order quantity to 8000, leaving 7000 open
5	Replace Request(Z,Y)				7000				Request decrease in order quantity to 7000, leaving 6000 open. Note OrigClOrdID set to last non rejected ClOrdID i.e. Y
6		Execution (Y,X)	Pending Replace	Pending Replace	10000	1000	9000	0	Broker processes Replace (Y,X) first
7		Execution (Z,X)	Pending Replace	Pending Replace	8000	1000	7000	0	Broker then processes Replace (Z,Y). Note OrigClOrdID set to last accepted ClOrdID i.e. X
8		Execution (Y,X)	Replace	Pending Replace	8000	1000	7000	0	Broker processes Replace (Y,X) first Note OrigClOrdID set to last accepted ClOrdID i.e. X. OrdStatus of Pending Replace takes precedence over Partially Filled
9		Execution (Z,Y)	Replace	Partially Filled	7000	1000	6000	0	Broker then processes Replace (Z,Y) Note OrigClOrdID set to last accepted ClOrdID i.e. Y
10		Execution(Z)	Trade	Filled	7000	7000	0	6000	Execution for 6000

D.2.c – One cancel/replace request is issued followed immediately by another – both are rejected

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	Replace Request(Y,X)				8000				Request decrease in order quantity to 8000, leaving 7000 open
5	Replace Request(Z,Y)				7000				Request decrease in order quantity to 7000, leaving 6000 open. Note OrigCOrdID set to last non rejected ClOrdID i.e. Y
6		Execution (Y,X)	Pending Replace	Pending Replace	10000	1000	9000	0	Broker processes Replace (Y,X) first
7		Cancel Reject (Y,X)		Partially Filled					Broker rejects first replace request Note OrigClOrdID set to last accepted ClOrdID i.e. X
8		Execution (Z,X)	Pending Replace	Pending Replace	10000	1000	9000	0	Broker then processes Replace (Z,Y). Note OrigClOrdID set to last accepted ClOrdID i.e. X
9		Cancel Reject (Z,X)		Partially Filled					Broker then rejects second replace request Note OrigClOrdID set to last accepted ClOrdID i.e. X
10		Execution(X)	Trade	Partially Filled	10000	7000	3000	6000	Execution for 6000

D.2.d – One cancel/replace request is issued followed immediately by another – broker rejects the second as order is pending replace

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	Replace Request(Y,X)				8000				Request decrease in order quantity to 8000, leaving 7000 open
5	Replace Request(Z,Y)				7000				Request decrease in order quantity to 7000, leaving 6000 open Note OrigCOrdID set to last non rejected ClOrdID i.e. Y
6		Execution (Y,X)	Pending Replace	Pending Replace	10000	1000	9000	0	
7		Cancel Reject (Z,X)		Pending Replace					Rejected because broker does not support processing of order cancel replace request whilst order is pending cancel. CxlRejReason = 'Order already in pending cancel or pending replace status' OrigClOrdID set to last accepted ClOrdID i.e. X
8		Execution (Y,X)	Replace	Partially Filled	8000	1000	7000	0	
9		Execution (Y)	Trade	Partially Filled	8000	3000	5000	2000	Execution for 2000

This matrix illustrates the case where the broker/order receiver does not support multiple outstanding order cancel or order cancel/replace requests

E Unsolicited/Reinstatement

E.1.a – Telephoned order

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1									Order for 10000 phoned to broker
2		Execution	New	New	10000	0	0	0	Confirm that the broker has accepted the order – note that broker does not need to capture a ClOrdID
3		Execution	Trade	Partially Filled	10000	2000	8000	2000	Execution of 2000
4		Execution	Trade	Partially Filled	10000	3000	7000	1000	Execution of 1000
5		Execution	Trade	Filled	10000	10000	0	7000	Execution of 7000

E.1.b – Unsolicited cancel of a part-filled order

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4									Broker verbally agrees to cancel order
5		Execution(X)	Canceled	Canceled	10000	1000	0	0	Broker signifies that order has been canceled - ExecRestatementReason = Verbal change

This scenario might occur if the buy-side has not implemented order cancel requests or alternatively there is an electronic communication problem at the point that the buy-side wishes to send a cancel request.

E.1.c – Unsolicited replacement of a part-filled order

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3									Broker verbally agrees to increase order quantity to 11000
4		Execution(X)	Restated	New	11000	0	11000	0	Broker signifies that order has been replaced ExecRestatementReason = Verbal
5		Execution(X)	Trade	Partially Filled	11000	1000	10000	1000	Execution for 1000
6									Broker verbally agrees to increase order quantity to 12000
7		Execution(X)	Restated	Partially Filled	12000	1000	11000	0	Broker signifies that order has been replaced ExecRestatementReason = Verbal change

This scenario might occur if the buy-side has not implemented order cancel/replace requests or alternatively there is an electronic communication problem at the point that the buy-side wishes to send a cancel replace request

E.1.d - Unsolicited reduction of order quantity by sell side (e.g. for US ECNs to communicate Nasdaq SelectNet declines)

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Restated	New	9000	0	9000	0	ExecRestatementReason=”Partial Decline of OrderQty”
4		Execution(X)	Trade	Filled	9000	9000	0	9000	

E.1.e - Unsolicited cancel of a 'cancel if not best' order

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Price**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000	56				ExecInst = Z (Cancel if Not Best)
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*56*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN) (e.g. if the order book is at 56.1-57.1 prior to this order)*
2		Execution(X)	New	New	10000	56	0	10000	0	Order accepted as order book was 55.9-56.9 prior to this order. Order book is now 56.0-56.9
3		Execution(X)	Canceled	Canceled	10000	56	0	0	0	Order book moves to 56.1-57.0. Order is no longer best bid/offer so is canceled with ExecRestatementReason ="Canceled, Not Best"

E.1.f - Order is sent to exchange, held waiting for activation and then activated

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10 000				Entry of a stop (OrdType = 3), stop limit (OrdType = 4), At the Close (TimeInForce = 7), etc, order. I.e. an order that is held off the book wainting for activation subject to specified conditions.
2		*Execution(X)*	*Rejected*	*Rejected*	*10 000*	*0*	*0*	*0*	*If order is rejected by trader / exchange*
2		Execution(X)	New	New	10 000	0	10 000	0	Trader / exchange acknowledge receipt of order but does not enter it into the book at activation conditions are not met (**WorkingIndicator = N**). Note that if the conditions are met on entry, this WorkingIndicator is not sent.
3		Execution(X)	Triggered by (Trading System)	New	10 000	0	10 000	0	Activation conditions are reached. Trader / exchange acknowledge that order is put on book (**WorkingIndicator = Y**). Note that this message can be implicit in situations where there is an immediate fill or partial fill (as any state other than New / Pending New means the order has been added to the book and is working).

4		Execution(X)	Trade	Partially Filled	10 000	2 000	8 000	2 000	Execution of 2000
5		Execution(X)	Trade	Filled	10 000	10 000	0	8 000	Execution of 8000

F Order Reject

F.1.a– Order rejected due to duplicate ClOrdID

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	New Order(X)				15000				Order submitted with the same order id .
5		Execution(X)	Rejected	Partially Filled	10000	1000	9000	0	OrdRejReason = duplicate order. Note combining a reject of the order for 15000 with a status on the first order for 10000 (which is partially filled)

F.1.b - Poss resend and duplicate ClOrdID

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		Execution(X)	New	New	10000	0	10000	0	
3	New Order(X)				10000				PossResend=Y
4		Execution(X)	Order Status	New	10000	0	10000		Because order X has already been received, confirm back the current state of the order. Last Qty not required when ExecType = Order Status
5	New Order(X)				20000				PossResend=N or not set
6		Execution(X)	Rejected	New	10000	0	10000		OrdRejReason = duplicate order. Note combining a reject of the second order for 20000 with a status on the first order for 10000.
7	New Order(Y)				15000				PossResend=Y
8		Execution(Y)	New	New	15000	0	15000	0	Because order Y has not been received before, confirm back as a new order.

F.1.c - Order rejected because the order has already been verbally submitted

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				Order for 10000 sent electronically
2									Order passed verbally as there is communication problem and order does not arrive. The verbally passed order starts getting executed
3		Execution(X)	Rejected	Rejected	10000	0	0	0	Order finally arrives and is detected as a duplicate of a verbal order and is therefore rejected. OrdRejReason = duplicate of a verbal order

Note that the sell-side may employ a number of mechanisms to detect that the electronic order is potentially a duplicate of a verbally passed order, e.g. :

- Checking the possdup flag on the order message header
- Checking the incoming order details against other orders from the same client (e.g. side, quantity)
- Looking at the transact time on the order as a guide to ‘staleness’

G Status

G.1.a - Order status request rejected for unknown order

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4	Status Request (Y)								
5		Execution(Y)	Order Status	Rejected	0	0	0		OrdRejReason = unknown order LastQty not required when ExecType=Order Status

G.1.b – Transmitting a CMS-style "Nothing Done" in response to a status request

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3	Status Request(X)								
4		Execution(X)	Order Status	New	10000	0	10000	0	Text="Nothing Done"

G.1.c - Order sent, immediately followed by a status request. Subsequent status requests sent during life of order

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2	Status Request (X)								
3		Execution(X)	Order Status	Pending New	10000	0	10000		Sent in response to status request. LastQty not required when ExecType=Order Status
4		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected*
4		Execution(X)	New	New	10000	0	10000	0	
5	Status Request (X)								
6		Execution(X)	Order Status	New	10000	0	10000		Sent in response to status request.
7		Execution(X)	Trade	Partially Filled	10000	2000	8000	2000	Execution for 2000
8	Status Request (X)								
9		Execution(X)	Order Status	Partially Filled	10000	2000	8000		Sent in response to status request
10		Execution(X)	Trade	Filled	10000	10000	0	8000	Execution for 8000
11	Status Request (X)								
12		Execution(X)	Order Status	Filled	10000	10000	0		Sent in response to status request
13	Replace Request(Y,X)				12000				Request to increase order qty
14		Execution (Y,X)	Pending Replace	Pending Replace	10000	10000	0	0	
15		Execution (Y,X)	Replace	Partially Filled	12000	10000	2000	0	
16	Status Request (X)								

17		Execution (Y,X)	Order Status	Partially Filled	12000	10000	2000		Sent in response to status request. Note reference to X to allow tie back of execution report to status request
18	Status Request (Y)								
19		Execution(Y)	Order Status	Partially Filled	12000	10000	2000		Sent in response to status request

H GT

H.1.a - GTC order partially filled, restated (renewed) and partially filled the following day

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**Ord Status**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Day Order Qty**	**Day Cum Qty**	**Comment**
Day 1,1	New Order(X)				10000						
Day 1,2		Execution(X)	New	New	10000	0	10000	0			
Day 1,3		Execution(X)	Trade	Partially Filled	10000	2000	8000	2000			Execution for 2000
Day 1,4		Execution(X)	Done for Day	Done for Day	10000	2000	8000	0			Optional at end of trading day
Day 2,1		Execution(X)	Restated	Partially Filled	10000	2000	8000	0	8000	0	ExecRestatementReason = GTC renewal/restatement (no change) – optionally sent the following morning
Day 2,2		Execution(X)	Trade	Partially Filled	10000	3000	7000	1000	8000	1000	Execution for 1000

H.1.b - GTC order with partial fill, a 2:1 stock split then a partial fill and fill the following day

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**Ord Status**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Day Order Qty**	**Day Cum Qty**	**Comment**
Day 1,1	New Order(X)				10000						
Day 1,2		Execution(X)	New	New	10000	0	10000	0			
Day 1,3		Execution(X)	Trade	Partially Filled	10000	2000	8000	2000			Execution for 2000 @ 50
Day 1,4		Execution(X)	Done for Day	Done for Day	10000	2000	8000	0			Optional at end of trading day
Day 2,1		Execution(X)	Restated	Partially Filled	20000	4000	16000	0	16000	0	Sent the following morning after the split ExecRestatementReason = GTC corporate action. AvgPx=25, DayAvgPx=0
Day 2,2		Execution(X)	Trade	Partially Filled	20000	9000	11000	5000	16000	5000	Execution for 5000
Day 2,3		Execution(X)	Trade	Filled	20000	20000	0	11000	16000	16000	Execution for 11000

H.1.c - GTC order partially filled, restated(renewed) and canceled the following day

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**Ord Status**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Day Order Qty**	**Day Cum Qty**	**Comment**
Day 1,1	New Order(X)				10000						
Day 1,2		Execution(X)	New	New	10000	0	10000	0			
Day 1,3		Execution(X)	Trade	Partially Filled	10000	2000	8000	2000			Execution for 2000
Day 1,4		Execution(X)	Done for Day	Done for Day	10000	2000	8000	0			Optional at end of trading day
Day 2,1		Execution(X)	Restated	Partially Filled	10000	2000	8000	0	8000	0	ExecRestatementReason = GTC renewal/restatement (no change) – optionally sent the following morning
Day 2,2	Cancel Request (Y,X)				10000						
Day 2,3		*Cancel Reject (Y,X)*		*Partially Filled*							*If rejected by salesperson*
Day 2,3		Execution (Y,X)	Pending Cancel	Pending Cancel	10000	2000	8000	0	8000	0	
Day 2,4		*Cancel Reject (Y,X)*		*Partially Filled*							*If rejected by trader/exchange*
Day 2,4		Execution (Y,X)	Canceled	Canceled	10000	2000	0	0	8000	0	

H.1.d - GTC order partially filled, restated(renewed) followed by replace request to increase quantity

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**Ord Status**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Day Order Qty**	**Day Cum Qty**	**Comment**
Day 1,1	New Order(X)				10000						
Day 1,2		Execution(X)	New	New	10000	0	10000	0			
Day 1,3		Execution(X)	Trade	Partially Filled	10000	2000	8000	2000			Execution for 2000
Day 1,4		Execution(X)	Done for Day	Done for Day	10000	2000	8000	0			Optional at end of trading day
Day 2,1		Execution(X)	Restated	Partially Filled	10000	2000	8000	0	8000	0	ExecRestatementReason = GTC renewal/restatement (no change) – optionally sent the following morning
Day 2,2	Replace Request(Y,X)				15000						Increasing qty
Day 2,3		*Cancel Reject (Y,X)*		*Partially Filled*							*If rejected by salesperson*
Day 2,3		Execution (Y,X)	Pending Replace	Pending Replace	10000	2000	8000	0	8000	0	
Day 2,4		Execution (X)	Trade	Pending Replace	10000	3000	7000	1000	8000	1000	Execution for 1000
Day 2,5		*Cancel Reject (Y,X)*		*Partially Filled*							*If rejected by trader/exchange*
Day 2,5		Execution (Y,X)	Replace	Partially Filled	15000	3000	12000	0	13000	1000	

I TimeInForce

I.1.a – Fill or Kill order cannot be filled

Time	Message Received (ClOrdID, OrigClOrdID)	Message Sent (ClOrdID, OrigClOrdID)	Exec Type	OrdStatus	Order Qty	Cum Qty	Leaves Qty	Last Qty	Comment
1	New Order(X)				10000				Order is FOK
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Canceled	Canceled	10000	0	0	0	If order cannot be immediately filled

I.1.b – Immediate or Cancel order that cannot be immediately hit

Time	Message Received (ClOrdID, OrigClOrdID)	Message Sent (ClOrdID, OrigClOrdID)	Exec Type	OrdStatus	Order Qty	Cum Qty	Leaves Qty	Last Qty	Comment
1	New Order(X)				10000				Order is IOC
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	1000	Execution for 1000
4		Execution(X)	Canceled	Canceled	10000	1000	0	0	If order cannot be immediately hit

J Execution Cancels/Corrects

J.1.a – Filled order, followed by correction and cancellation of executions

Time	Message Received (ClOrdID, OrigClOrdID)	Message Sent (ClOrdID, OrigClOrdID)	Exec Type	OrdStatus	Order Qty	Cum Qty	Leaves Qty	AvgPx	Last Qty	Last Px	ExecId (ExecRefID)	Comment
1	New Order(X)				10000							
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*		*0*		*A*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	0		B	
3		Execution(X)	Trade	Partially Filled	10000	1000	9000	100	1000	100	C	Execution for 1000 @ 100
4		Execution(X)	Trade	Filled	10000	10000	0	109	9000	110	D	Execution for 9000 @ 110
5		Execution(X)	Trade Cancel	Partially Filled	10000	9000	1000	110	0	0	E ©	Cancel execution for 1000.
6		Execution(X)	Trade Correct	Partially Filled	10000	9000	1000	100	9000	100	F (D)	Correct price on execution for 9000 to 100.
7		Execution(X)	Trade	Filled	10000	10000	0	102	1000	120	G	Execution for 1000 @ 120
8		Execution(X)	Trade Correct	Filled	10000	10000	0	120	9000	120	H(F)	Correct price on execution for 9000 to 120
9	Replace Request (Y,X)				12000							Request to increase order qty
10		Execution (Y,X)	Pending Replace	Pending Replace	10000	10000	0	120	0	0	I	
11		Execution (Y,X)	Replace	Partially Filled	12000	10000	2000	120	0	0	J	
12		Execution(Y)	Trade Correct	Partially Filled	12000	10500	1500	120	9500	120	K(H)	Correct execution of 9000 @ 120 to 9500 @ 120

J.1.b - A canceled order followed by a busted execution and a new execution

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**Ord Status**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**ExecID (ExecRefID)**	**Comment**
1	New Order(X)				10000					
2		Execution(X)	New	New	10000	0	10000	0	A	
3		Execution(X)	Trade	Partially Filled	10000	5000	5000	5000	B	LastPx=50
4	Cancel Request(Y,X)				10000					
5		Execution (Y,X)	Pending Cancel	Pending Cancel	10000	5000	5000	0	C	
6		Execution (Y,X)	Canceled	Canceled	10000	5000	0	0	D	
7		Execution(X)	Trade Cancel	Canceled	10000	0	0	0	E(B)	Cancel of the execution. ‘Canceled’ order status takes precedence over ‘New’
8		Execution(X)	Trade	Canceled	10000	4000	0	4000	F	Fill for 4000. LastPx=51

J.1.c - GTC order partially filled, restated (renewed) and partially filled the following day, with corrections of quantity on both executions

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**Ord Status**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Day Order Qty**	**Day Cum Qty**	**ExecID (ExecRefID)**	**Comment**
Day 1,1	New Order(X)				10000							
Day 1,2		Execution(X)	New	New	10000	0	10000	0			A	
Day 1,3		Execution(X)	Trade	Partially Filled	10000	2000	8000	2000			B	Execution for 2000
Day 1,4		Execution(X)	Done for Day	Done for Day	10000	2000	8000	0			C	Optional at end of trading day
Day 2,1		Execution(X)	Restated	Partially Filled	10000	2000	8000	0	8000	0	D	ExecRestatementReason = GTC renewal/restatement (no change) – optionally sent the following morning
Day 2,2		Execution(X)	Trade	Partially Filled	10000	3000	7000	1000	8000	1000	E	Execution for 1000
Day 2,3		Execution(X)	Trade Correct	Partially Filled	10000	2500	7500	1500	8500	1000	F (B)	Correct quantity on previous day's execution from 2000 to 1500
Day 2,4		Execution(X)	Trade Correct	Partially Filled	10000	2000	8000	500	8500	500	G (E)	Correct quantity on today's execution from 1000 to 500

J.1.d – Part-filled order Done for day followed by trade correction and bust

Time	Message Received (ClOrdID, OrigClOrdID)	Message Sent (ClOrdID, OrigClOrdID)	Exec Type	Ord Status	Order Qty	Cum Qty	Leaves Qty	Last Qty	ExecID (ExecRefID)	Comment
1	New Order(X)				10000					
2		Execution(X)	New	New	10000	0	10000	0	A	
3		Execution(X)	Trade	Partially Filled	10000	5000	5000	5000	B	LastPx=50
4		Execution(X)	Done for Day	Done for day	10000	5000	0	0	C	Done for day message sent
5		Execution(X)	Trade Correct	Done for day	10000	4000	0	4000	D (B)	Correct quantity on execution to 4000. LastPx = 50
6		Execution(X)	Trade Cancel	Done for day	10000	0	0	0	E (D)	Done for Day OrdStatus takes precedence

K Trading Halt

K.1.a – Trading Halt – Reinstate

Time	Message Received (ClOrdID, OrigClOrdID)	Message Sent (ClOrdID, OrigClOrdID)	Exec Type	OrdStatus	Order Qty	Cum Qty	Leaves Qty	Last Qty	Comment
1	New Order(X)				10000				ExecInst set to reinstate on trading halt
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3									Trading halt established
4									Trading halt lifted
5		Execution(X)	Trade	Filled	10000	10000	0	10000	

K.1.b – Trading Halt – Cancel

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				ExecInst set to cancel on trading halt
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3									Trading halt established
4		Execution	Canceled	Canceled	10000	0	0	0	Order canceled due to trading halt. ExecRestatementReason = Canceled due to trading halt

L Miscellaneous

L.1.a– Transmitting a guarantee of execution prior to execution

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Comment**
1	New Order(X)				10000				
2		*Execution(X)*	*Rejected*	*Rejected*	*10000*	*0*	*0*	*0*	*If order is rejected by sell-side (broker, exchange, ECN)*
2		Execution(X)	New	New	10000	0	10000	0	
3		Execution(X)	Stopped	Stopped	10000	0	10000	1000	Text=”You are guaranteed to buy 1000 at 50.10”; LastPx=50.10. This is similar to the concept of a ‘protected’ trade. Not actually reporting a trade, so Exectype = Stopped
4		Execution(X)	Trade	Stopped	10000	1000	9000	1000	LastPx=50 * executed price is better than guaranteed

L.1.b- Use of CashOrderQty

Time	**Message Received** (ClOrdID, OrigClOrdID)	**Message Sent** (ClOrdID, OrigClOrdID)	**Exec Type**	**OrdStatus**	**Order Qty**	**Cash Order Qty**	**Cum Qty**	**Leaves Qty**	**Last Qty**	**Last Px**	**Last Qty**	**Comment**
1	New Order(X)					10000						Currency=EUR. A buy order to invest 10,000 EUR.
2		*Execution(X)*	*Rejected*	*Rejected*		10000	*0*	*0*			*0*	*If order is rejected*
2		Execution(X)	New	New	500	10000	0	500			0	Assuming product has a unit price of 20 EUR at time of order receipt
3		Execution(X)	Trade	Partially Filled	500	10000	200	300	200	20.1	200	Execution of 200 @20.1 (i.e. does not have to be at the 'conversion price' of 20_
4		Execution(X)	Trade	Filled	500	10000	500	0	300	20.2	300	Execution of 300 @20.2 (i.e. does not have to be at the 'conversion price' of 20_

Order Handling and Instruction Semantics

London SETS Order Types Matrix

The table below presents the representation of the London Stock Exchange Trading System (SETS) order types in the FIX protocol:

LSE Order Type	OrdType	TimeInForce	ExpireTime	Price	Comment
At Best	1	3	n/a	No	
Fill or Kill - no limit price	1	4	n/a	No	
Fill or Kill - limit price	2	4	n/a	Yes	
Limit - day	2	n/a, 0	n/a	Yes	
Limit - good until	2	6	Good Till Date	Yes	
Execute and Eliminate	2	3	n/a	Yes	
Market Orders - day	1	n/a, 0	N/a	No	SETS Release 3.1 Only
Market Orders -good until	1	6	Good Till Date	No	SETS Release 3.1 Only

Asia/Pacific Regional Order Handling

The following table identifies how to represent via FIX the commonly used and understood order handling instructions within the Asia/Pacific region.

Asia/Pacific Dealer Instruction	OrdType	ExecInst	Other Fields
Careful Discretion	1 (Market)	4 (Over the Day)	
Market	1 (Market)	5 (Held)	
Trader Discretion	1 (Market)	1 (Not Held)	
Hong Kong SE – Regular Limit Order	2 (Limit)	b = Strict Limit (No Price Improvement)	Price = xxx
Hong Kong SE – Special Limit	2 (Limit)		TimeInForce = Immediate Or Cancel Price = xxx
HongKongSE – Enhanced Limit	2 (Limit)*	d = Peg to Limit Price	Price = xxx PegType = fixed PegOffsetType = tick PegOffsetValue = -1 (buy) 1(sell)

			PegOffsetLimitType = or worse

*** note that strictly speaking this order type is both 'Limit' and 'Pegged' but set OrdType = limit**

If only OrdType 2 (Limit) is used with no ExecInst specified, the order will be traded as Limit Or Better. Sell-side firms will trade the order with the best possible tick up to the Limit price in the Hong Kong Stock Exchange.

Japanese Exchange Price Conditions

The following table identifies how to represent via FIX the price conditions implemented by Japanese exchanges.

Japanese Exchange Price Condition	OrdType	ExecInst	PegOffsetValue	TimeInForce
Current price limit	P (Pegged)	P (Market Peg)		
Preferred price limit	P (Pegged)	P (Market Peg)	+1 (or –1)	
Market with Leftover Limit	K (Market with Leftover As Limit)			
Market Fill with Leftover Kill	1 (Market)			3 (Immediate or Cancel (IOC))

Euronext and Similar Exchange Price Conditions

The following table identifies how to represent via FIX the price conditions implemented by the Euronext and Similar exchanges.

Euronext and Similar Exchange Price Condition	OrdType
A tout prix (At All Price)	1 (Market)
Au prix du marche ()	J (Market with Leftover As Limit)

Handling Instructions (HandlInst) field

The following identifies the meaning and expected usage of the HandlInst (Handling Instructions) field. This field has been required on the New Order messages since the inception of FIX. Usage of this field may vary by market and by broker. Buy side and sell side firms should confirm their mutual understanding of the usage and implementation of HandlInst.

1 = Automated execution order, private, no Broker	Order is systematically routed to the market place, usually to an exchange or ECN or market maker, for execution. It is expected that no broker intervention is required to

intervention	accept or forward the order into the market. Notes: • Private does not mean broker cannot see buy side order flow. In many markets, the Broker has the legal requirement to monitor customer order flow and be responsible for those orders. • Buy side firm may be expected to supply the symbology required by the market. • Broker may require certain optional fields, such as ExDestination and/or Currency. • Implies an immediate reject will be sent if order cannot be forwarded immediately into the market.
2 = Automated execution order, public, Broker intervention OK	Broker may stop order from flowing immediately into the market place. This would typically be done, if the broker can cross this order against another order to provide price improvement and / or liquidity. If Broker does not choose to stop this order, it will automatically flow into the market for execution.
3 = Manual order, best execution	Order is routed to appropriate sell side broker who then accepts responsibility for the order. This should operate as though the buy side firm called the order into their broker. Notes: • Different than “not held”. • Does not imply “call first” (an ExecInst value).

Pegged Orders

The following are all pegging PegPriceType[1] values used when OrdType=P to specify the type of pegged order represented. Note that these fields cannot be combined; only one may be specified on a pegged order.

1 = Last peg (last sale)

2 = Mid-price peg (midprice of inside quote)

3 = Opening peg

4 = Market peg

5 = Primary peg (primary market - buy at bid/sell at offer)

7 = Peg to VWAP

8 = Trailing Stop Peg

9 = Peg to Limit Price

A pegged order acts like a limit order, except that the limit price is set relative to another price, such as the last sale price, midpoint price, opening price, bid, offer, or VWAP (Volume Weighted Average Price). A primary peg order

[1] Versions FIX 4.4 and prior used ExecInst to define the type of peg.

is priced relative to the bid if buying, the offer if selling. A market peg order is priced relative to the offer if buying, the bid if selling.

Pegs can be fixed (that is they are calculated when the order is received) or floating, in which case they fluctuate according to movements in the reference price (using the PegMoveType field). The PegOffsetType field can be used to specify whether the desired offset is being expressed as a price, in basis points, in ticks or in price tiers/levels. For example a primary pegged buy order with PegOffsetValue = -0.01, PegMoveType = Fixed (1), and PegOffsetType = Price (0) will have a fixed price equal to the bid less 0.01. The same order with a PegOffsetType = Ticks (2) and a PegOffsetValue = -1 will have a fixed price equal to the bid less one tick. To specify that a buy order is to float on the third best price level set the PegOffsetType = Price Tier/Level (3), ExecInst = Primary Peg (R), PegMoveType = Floating (0) and PegOffsetValue = -2 (i.e. 2 below the best bid). PegRoundingDirection can be used to specify, in the event that the calculated price is not a valid tick size, whether the price should be rounded aggressively or passively.

When calculating the peg price, the reference price can be obtained from more than one liquidity pool as specified by the PegScope field. For example a PegScope = national excluding local will use a reference price based on all liquidity pools except the one in which the order resides. Another possibility is to peg to a specific security using PegSymbol, PegSecurityID and PegSecurityIDSource and/or PegSecurityDesc fields.

Prior FIX specifications defined ExecInst = Fixed Peg to Local best bid or offer at time of order (T). This must now be expressed as a pegged order with PegPriceType = Primary Peg (5), PegMoveType = Fixed (1), and PegScope = Local (1).

In the absence of the PegOffsetValue field, or when PegOffsetValue = 0, the price of the pegged order follows the referenced quantity exactly. Note that the PegOffsetValue is always ‘added’ to the reference value. PegMoveType will default to float if not specified.

Some systems allow pegged orders to be specified with a Price field. In such cases the OrdType should be specified as ‘pegged’. In this case, the Price field serves to put a limit on how far the pegged value can move. For instance, if the bid for a stock is 50, the offer is 50.10, the order is a primary peg to sell, PegOffsetValue = -0.02, and Price = 45, the order will be priced to sell at the offer + (-0.02) or 50.08. If the offer falls, the order's price will fall such that it is always 0.02 less than the offer. However, once the order's price hits 45 (the limit specified in the Price field) it can fall no further.

A pegged order with PegPriceType = 8, a trailing stop peg, behaves differently. It requires PegOffsetValue, which must be positive when buying and negative when selling. A trailing stop peg represents a stop order whose price can fluctuate relative to the last sale price. Initially, the stop is placed at the last sale price + PegOffsetValue. The stop price will move like a last peg so that the stop price is the last sale price + PegOffsetValue, with one exception: if buying, the fluctuating stop price cannot increase, and if selling, the stop price cannot decrease. For example, a security trades at $10.00, and a trailing stop peg order to sell with PegOffsetValue = -0.10 is placed. The pegged stop price will rest at $9.90. The security rises in price to $10.20, and the stop similarly rises to $10.10. The security price falls to $10.15, but the trailing stop holds its price at $10.10. The security's price keeps falling, and when it reaches $10.10, the stop order is triggered and the security is sold. Trailing stop pegs are incompatible with PegMoveType = Fixed (1).

Although best practice is not to restate orders when the price of a floating pegged orders changes, some system need the option to do such restatements periodically or based on other events (e.g. when a trailing stop peg reaches its stop price). In those cases the PeggedRefPrice field can be used to relay the reference price. Note that the Price field is used for any limit (cap/floor) price and the PeggedPrice tag for the display price of the order. In cases where the only reason for the restatement is a change in price, the “Peg Refresh” value can be used as the ExecRestatementReason. Note that the fields changing should be the PeggedPrice and the PeggedRefPrice.

The following table shows the mapping between the deprecated ExecInst values and the new PegPriceType:

OrdType (retained)	**ExecInst (deprecated values)**	**PegPriceType (added tag)**
P = Pegged	L = Last peg (last sale)	1 = Last peg (last sale)
P = Pegged	M = Mid-price peg (midprice of inside quote)	2 = Mid-price peg (midprice of inside quote)

P = Pegged	O = Opening peg	3 = Opening peg
P = Pegged	P = Market peg	4 = Market peg
P = Pegged	R = Primary peg (primary market - buy at bid/sell at offer)	5 = Primary peg (primary market - buy at bid/sell at offer)
P = Pegged	W = Peg to VWAP	7 = Peg to VWAP
P = Pegged	a = Trailing Stop Peg	8 = Trailing Stop Peg
P = Pegged	d = Peg to Limit Price	9 = Peg to Limit Price

Peg Instruction Examples

Fixed Peg with limit that is not exceeded

	Message	Peg Price Type (1094)	Peg Offset Type (835)	Peg Offset Value (211)	Peg Move Type (835)	Peg Scope (840)	Side (2)	Price (44)	Pegged Price (839)	Pegged RefPrice (1095)	Comment
1	User sends New Order single pegged to -1 tick below last local market sale with a limit on 10	Last Sale (1)	Ticks (2)	1	Fixed (1)	Local (1)	Sell	10.00	N/A	N/A	
2	Exchange issues Execution Report informing the user that order is in the book	Last Sale (1)	Ticks (2)	1	Fixed (1)	Local (1)	Sell	10.00	10.09	10.00	Assuming last sale is 10:10 and penny ticks. As the peg is fixed the initial limit price and other peg instructions can now be dropped

Fixed Peg with limit that is exceeded

	Message	Peg Price Type (1094)	Peg Offset Type (835)	Peg Offset Value (211)	Peg Move Type (835)	Peg Scope (840)	Side (2)	Price (44)	Pegged Price (839)	Pegged Ref Price (1095)	Comment
1	User sends New Order single pegged to -1 tick below last local market sale with a limit on 10	Last Sale (1)	Ticks (2)	1	Fixed (1)	Local (1)	Sell	10.00	N/A	N/A	
2	Exchange issues Execution Report informing the user that order is in the book	Last Sale (1)	Ticks (2)	1	Fixed (1)	Local (1)	Sell	10.00	10.00	9.90	Assuming last sale is 9:90. As a pegged price is below the limit, the order is fixed at the limit

Floating Peg with limit that is not exceeded

	Message	Peg Price Type (1094)	Peg Offset Type (835)	Peg Offset Value (211)	Peg Move Type (835)	Peg Scope (840)	Side (2)	Price (44)	Pegged Price (839)	Pegged Ref Price (1095)	Comment
1	User sends New Order single pegged to -1 tick below last local market sale with a limit on 10	Last Sale (1)	Ticks (2)	1	Floating (0)	Local (1)	Sell	10.00	N/A	N/A	
2	Exchange issues Execution Report informing the user that order is in the book	Last Sale (1)	Ticks (2)	1	Floating (0)	Local (1)	Sell	10.00	10.09	10.10	Assuming last sale is 10:10 and penny ticks.
3	Exchange optionally issues Execution Report informing the user that order is repegged	Last Sale (1)	Ticks (2)	1	Floating (0)	Local (1)	Sell	10.00	10.04	10.05	Assuming a new last sale occured at 10.05
4	Exchange issues complete fill at the new price. (Tag values do not contribute to the example and are not shown)										

Floating Peg wiht limit that is exceeded

	Message	Peg Price Type (1094)	Peg Offset Type (835)	Peg Offset Value (211)	Peg Move Type (835)	Peg Scope (840)	Side (2)	Price (44)	Pegged Price (839)	Pegged Ref Price (1095)	Comment
1	User sends New Order single pegged to -1 tick below last local market sale with a limit on 10	Last Sale (1)	Ticks (2)	1	Floating (0)	Local (1)	Sell	10.00	N/A	N/A	
2	Exchange issues Execution Report informing the user that order is in the book	Last Sale (1)	Ticks (2)	1	Fixed (1)	Local (1)	Sell	10.00	10.00	9.90	Assuming last sale is 9:90. As a pegged price is below the limit, the order is assigned the limit price
3	Exchange optionally issues Execution	Last Sale (1)	Ticks (2)	1	Floating (0)	Local (1)	Sell	10.00	10.04	10.05	Assuming a new last sale occured at

	Report informing the user that order is repegged										10.05. As the peg is now above the limit, the order is repegged.
4	Exchange issues complete fill at the new price. (Tag values do not contribute to the example and are not shown)										

Discretionary Pricing

The presence of DiscretionInst on an order indicates that the trader wishes to display one price but will accept trades at another price. For example, a sell order with OrdType = Limit (2), Price=50.00, DiscretionInst = Related to displayed price (0) and DiscretionOffsetValue = -0.25 means that the order should be displayed as an offer for 50.00, but will match any bid >= 49.75. Discretionary pricing can also be used when pegging an order - for example to indicate that a buy order is to be displayed as pegged to the bid minus 0.25, but can be matched by anything <= the offer, set OrdType = Pegged (P), ExecInst = Primary Peg (R), PegOffsetValue = -0.25, DiscretionInst = Related to market price (1) and DiscretionOffsetValue = 0.

Discretionary prices can be pegged to reference values in the same way as displayed prices (see above)

"Target Strategy" Orders

The presence of an ExecInst=e (lower case E), work to target strategy, indicates that the order is to be worked to try to achieve the specified target in the TargetStrategy field, typically by slicing the order into the market, either manually or via an algorithm . The start and end times during which the order is to be worked can be communicated using the EffectiveTime and ExpireTime fields respectively or through using TradingSessionIDs

For example, to indicate that the receiver of the order should try to work the order to achieve the volume weighted average price, set TargetStrategy = VWAP. A Participate order is one where the sender of the order wants the order to be worked such that the execution profile of the order is the specified percentage of the volume profile in the market. The target participation rate is communicated via the ParticipationRate field .

Where appropriate the performance versus the target can be communicated back to the originator of the order by use of the TargetStrategyPerformance field on the Execution Report. The use of this field will depend on the strategy. For a VWAP order this would be the VWAP price for the appropriate time period (taking into account any limit price on the order and excluding/including off order book trades as per the market convention). For Participate orders this field can be used to convey the actual % of volume in the appropriate time period that this executed volume represents. For Minimise Market Impact orders this may be utilised to give an estimate of the order's market impact in basis points, etc.

More complex parameters can be specified in the TargetStrategyParameters field.

"Reserve Quantity" Orders

Reserve orders allow users to hide the full size of their order and thereby potentially limit its influence on prices.

DisplayQty: Traditionally used to indicate reserve quantity. To indicate a single level of reserve quantity,DisplayQty should be used.

SecondaryDisplayQty: Used when two levels of reserve quantities are needed, e.g. one level displayed to the world (DisplayQty) and another displayed to subscribers of their ECN (SecondaryDisplayQty.) In other words, DisplayQty <= SecondaryDisplayQty <= OrderQty.

- One may place an order for 100,000 shares (OrderQty), only want 1000 shares shown to NASDAQ at any one time (DisplayQty), but will allow other subscribers of that ECN to see 5000 shares (SecondaryDisplayQty).

Additional reserve order features are supported through the following fields:

- DisplayWhen - Determines when the order is refilled:
  - Immediate = For each partial fill.
  - Exhaust = When the displayed quantity is completely filled.
- DisplayMethod - Determines what quantity should be displayed:
  - Initial = The same size as initially displayed (DisplayQty)
  - New = A value separate from the initially displayed size in order to limit the possibility of identifying the order as a reserve order (RefreshQty)
  - Random = A random value, supported through the following fields:
    - DisplayLowQty specifying the lowest value to display.
    - DisplayHighQty specifying the highest value to display.
    - DisplayMinIncr. Specifying the increment used when randomizing the new quantity to display. In some instances – for example a security that trades in round lots only – a minimum increment for the display quantity will be implied. But in many cases, the order entry side may wish a higher increment.

Reserve Quantity Order Examples

Refresh Immediate using Initial Display Quantity

Message	Order Qty	Leaves Qty	Display Qty	Display When	Display Method	Display LowQty	Display HighQty	Refresh Qty	Comment
New order	1000		100	1	1				
Execution Report (New)	1000	1000	100	1	1				
Execution Report (Partial Fill)	1000	750	100	1	1				Fill for 250
Execution Report (Partial Fill)	1000	50	50	1	1				Fill for 700
Execution Report (Filled)	1000	0	0	1	1				Fill for 50

Refresh Immediate using New Display Quantity

Message	Order Qty	Leaves Qty	Display Qty	Display When	Display Method	Display LowQty	Display HighQty	Refresh Qty	Comment
New order	1000		100	1	2			200	
Execution Report (New)	1000	1000	100	1	2			200	

Execution Report (Partial Fill)	1000	750	200	1	2			200	Fill for 250
Execution Report (Partial Fill)	1000	50	50	1	2			200	Fill for 700
Execution Report (Filled)	1000	0	0	1	2			200	Fill for 50

Refresh Immediately using Random Display Quantity

Message	Order Qty	Leaves Qty	Display Qty	Display When	Display Method	Display LowQty	Display HighQty	RefreshQty	Comment
New order	1000		100	1	3	100	200		
Execution Report (New)	1000	1000	100	1	3	100	200		
Execution Report (Partial Fill)	1000	750	150	1	3	100	200		Fill for 250, refresh size randomized
Execution Report (Partial Fill)	1000	50	50	1	3	100	200		Fill for 700
Execution Report (Filled)	1000	0	0	1	3	100	200		Fill for 50

Refresh when Display Quantity is Exhausted using Initial Display Quantity

Message	Order Qty	Leaves Qty	Display Qty	Display When	Display Method	Display LowQty	Display HighQty	Refresh Qty	Comment
New order	1000		100	2	1				
Execution Report (New)	1000	1000	100	2	1			(100)	
Execution Report (Partial Fill)	1000	950	50	2	1			(100)	Fill for 50
Execution Report (Partial Fill)	1000	900	100	2	1			(100)	Fill for 50
….									Subsequent fills,

Execution Report (Partial Fill)	1000	50	50	2	1			(100)	totaling 850
Execution Report (Filled)	1000	0	0	2	1			(100)	Fill for 50

Refresh when Display Quantity is Exhausted using New Display Quantity

Message	Order Qty	Leaves Qty	Display Qty	Display When	Display Method	Display LowQty	Display HighQty	Refresh Qty	Comment
New order	1000		100	2	2				
Execution Report (New)	1000	1000	100	2	2			200	
Execution Report (Partial Fill)	1000	950	50	2	2			200	Fill for 50
Execution Report (Partial Fill)	1000	900	200	2	2			200	Fill for 50
….									Subsequent fills, totaling 850
Execution Report (Partial Fill)	1000	50	50	2	2			200	
Execution Report (Filled)	1000	0	0	2	2			200	Fill for 50

Refresh when Display Quantity is Exhausted using Random Display Quantity

Message	Order Qty	Leaves Qty	Display Qty	Display When	Display Method	Display LowQty	Display HighQty	Refresh Qty	Comment
New order	1000		100	2	3	100	200		
Execution Report (New)	1000	1000	100	2	3	100	200		
Execution Report (Partial Fill)	1000	950	50	2	3	100	200		Fill for 50
Execution Report (Partial Fill)	1000	900	150	2	3	100	200		Fill for 50, refresh size randomized

….									Subsequent fills, totaling 850
Execution Report (Partial Fill)	1000	50	50	2	3	100	200		
Execution Report (Filled)	1000	0	0	2	3	100	200		Fill for 50

Triggering Instructions

In order to support increasingly complex, predefined and automatic, order modifications, the <Triggering Instructions> component block can be used. FIX 4.4, included:

- Stop and stop limit conditions, i.e. activating an order when a certain price is reached.
- Good at the close orders, i.e. activating an order when the closing trading sessions is reached.
- Reserve orders, i.e. hiding the full quantity of an order and instead showing a smaller part of the quantity at any time.
- Market to limit conditions, i.e. changing a market price order to a limit order once it is partially filled (or when it is entered into the book if the market does not allow market orders to sit on the book).

Some markets have more extensive triggering functionality than previously supported in FIX, including e.g.:

- Market Stop / Limit Stop / Market Stop Limit / Limit Stop Limit, i.e. orders that reside as tradable in the book, but will change when the stop price is reached.
- Defining a quantity change to be activated when the trigger is hit. A Limit Stop order, residing in the book for 100@10 could thereby be automatically changed to e.g. 200@market when the stop price is reached.
- Defining that the trigger should react on price changes in another security. In this way, e.g. an options stop order could be triggered off price changes in the underlying. An order can also be cancelled when a specified price is reached.
- Defining what type of price, independent on the side of the order, should trigger the action, e.g. Best Offer, Best Bid, Last Sale or Best Mid. The price definition could also include whether to use local market prices, national ones or even global prices.
- Specifying which direction a price change must have, e.g. rising (or falling) prices only.

Using the <Triggering Instructions> component block the following fields are available:

- TriggerType - determines what should trigger the change (Partial Execution; Specified Trading Session; Next Auction; Price Movement)
- TriggerAction - determines what action to take (Activate; Modify; Cancel)
- TriggerPrice - a specified limit price to validate against price movements – the trigger hits when the price is reached. The TriggerPrice is very similar to the current StopPx tag.
- A security (if not the one of the order) whose price movements should be tracked:
    - TriggerSymbol
    - TriggerSecurityID
    - TriggerSecurityIDSource
    - TriggerSecurityDesc

- TriggerPriceType - determines what type of price should be tracked for price movements (Best Offer; Last Sale; Best Bid; Best Bid or Last Sale; Best Offer or Last Sale; Best Mid).
- TriggerPriceTypeScope determining the scope of the price (None, Local, National, Global).
- TriggerPriceDirection used to specify if the trigger should hit only on rising (Up) or falling prices (Down). If unspecified, the trigger will hit in both directions.
- A specification of the price the order should have after the trigger hit (provided the price should change at all):
    - TriggerNewPrice specifying a new Limit Price to be assigned to the order.
    - TriggerOrderType specifying the order type, e.g. that the order is changed from a limit to a market order.
- TriggerNewQty specifying a new Quantity to be assigned to the order after the trigger hit.
- A defined trading session when the trigger hits:
    - TriggerTradingSessionID.
    - TriggerTradingSessionSubID.

Trigger Instruction Examples

The following examples illustrate the toolbox provided by the Triggering Instruction component block. In order to make the examples easy to relate to, reasonably common trigger actions have used. You may note that many of them are supported in earlier versions of FIX using other mechanisms. The examples are focused on showing the use of the TriggerType and TriggerAction tag values.

Stop Orders

Vanilla "Stop" Order Trigger

Tag	Field Name	Value	Value Description	Comment
1100	TriggerType	4	Price Movement	
1101	TriggerAction	1	Activate	
1102	Trigger Price	10.00		
1107	TriggerPriceType	2	Last Trade	
40	OrdType	1	Market	

Vanilla "Stop Limit" Order Trigger

Tag	Field Name	Value	Value Description	Comment
1100	TriggerType	4	Price Movement	
1101	TriggerAction	1	Activate	
1102	Trigger Price	10.00		
1107	TriggerPriceType	2	Last Trade	
1110	TriggerNewPrice	10.00		
40	OrdType	2	Limit	

"Limit Stop" Order Trigger

A limit sell order with a "high" price that is displayed in the book. The order is converted to a market order if prices have being rising up through the stop price and then are decling back through it.

Tag	Field Name	Value	Value Description	Comment
1100	TriggerType	4	Price Movement	
1101	TriggerAction	1	Modify	
1102	Trigger Price	10.00		
1107	TriggerPriceType	2	Last Trade	
1109	TriggerPriceDirection	Down	Triggers if the price of the specified type goes DOWN to or through the specified Trigger Price.	
1111	TriggerOrdType	1	Market	
54	Side	2	Sell	
40	OrdType	2	Limit	
44	Price	11.00		

Trading Session Triggers

Vanilla "At the Close" Trigger

Tag	Field Name	Value	Value Description	Comment
1100	TriggerType	2	Specified Trading Session	
1101	TriggerAction	1	Activate	
1113	TriggerTradingSessionID	5	Closing Auction	TradingSessionID's are bilaterally agreed
40	OrdType	1	Market	

Vanilla "Funari" Trigger

A Limit Day Order where an unexecuted portion is handled as Market On Close

Tag	Field Name	Value	Value Description	Comment
1100	TriggerType	2	Specified Trading Session	
1101	TriggerAction	1	Modify	
1113	TriggerTradingSessionID	5	Closing Auction	TradingSessionID's are bilaterally agreed
1111	TriggerOrdType	1	Market	
40	OrdType	2	Limit	

Vanilla "Next Auction" Trigger

Defines an order that is activated for when the next call auction is initiated. Especially relevant when continuous call auctions are performed in the market.

Tag	Field Name	Value	Value Description	Comment
1100	TriggerType	3	Next Auction	
1101	TriggerAction	1	Activate	
40	OrdType	1	Market	

Pre-Defined Order Modifications

Vanilla "Market with Leftover as Limit" Trigger

A market order that, if partially executed, is converted to a limit order at the last executed price.

Tag	Field Name	Value	Value Description	Comment
1100	TriggerType	1	Partial Execution	
1101	TriggerAction	2	Modify	
1111	TriggerOrdType	2	Limit	
40	OrdType	1	Market	

Cancel Order if certain Price is reached

Tag	Field Name	Value	Value Description	Comment
1100	TriggerType	4	Price Movement	
1101	TriggerAction	3	Cancel	
1102	Trigger Price	8.00		
1107	TriggerPriceType	2	Last Trade	
54	Side	2	Sell	
40	OrdType	2	Limit	
44	Price	10.00		

Time In Force (TIF)

When TIF=0 (DAY) is used in conjunction with TradingSessionID, the Time In Force of DAY means the order is good for the duration of the specified session. This will accommodate trading platforms where the specified trading session may span more than a calendar day (e.g. specified session starts at 8 p.m. and ends next day at 2 p.m.).

Booking Instructions Specified at Time of Order

The following table identifies the effect of booking instructions provided on the order.

DayBookingInst	BookingUnit	Effect (end-result)

"auto"	"each partial"	Each partial can be booked as soon as the partial is reported to the client
"auto"	"whole order"	The order can be booked as soon as it is filled (or part-filled and Done For Day)
"auto"	"aggregation"	The order can be booked as soon as it is filled (or part-filled and Done For Day)
"speak first"	"each partial"	Do not book after reporting a fill without discussion
"speak first"	"whole order"	Do not book order when filled (or part-filled when Done For Day) but discuss
"speak first"	"aggregation"	Do not book the aggregate until verbally agreed

OrderCapacity and OrderRestrictions (formerly Rule80A) Usage by Market

The Rule80A field was deprecated in FIX 4.3 and replaced by the combination of a new OrderCapacity field and OrderRestrictions field. See Volume 6: "Appendix 6-E - Deprecated Features and Supported Approach"

The term Rule80A is a very US market-specific term. Other world markets need to convey similar information, however, often a subset of the US values. In addition the deprecated Rule80A field's values both "overloaded" the field with various combinations of order capacity and associated order restrictions, and the Rule80A field as structured (modeled after CMS and SEC Rule 11Ac1-1/4) made it both difficult to understand and difficult to convey the various order capacities. This section documents the market-specific usage of the OrderCapacity and OrderRestrictions fields.

United States Listed Equity Markets:

Rule80A's values and usage details are documented in SEC Rule11Ac1-1/4. Note the purpose behind the rule is to restrict prices from rising or falling too fast providing more stability in the market. See Investments by Sharpe, 6th edition p. 50. Indicates the order type upon which exchange Rule 80A is applied.

The following values are valid and applicable when using FIX to communicate with the New York Stock Exchange (NYSE) or other US listed equity exchanges per the SuperDOT Notification document. The values and usage details when used for US trading are documented in SEC Rule11Ac1-1/4.

With regards to OrderCapacity and OrderRestirctions field usage in the United States Listed Equity Markets, the following table provides a cross-reference of former Rule80A values to FIX supported values

Rule80A Value		OrderCapacity (528)		OrderRestrictions (529) Note datatype: MultipleValueString		Side (54)	
A	Agency single order	A	Agency				
B	Short exempt transaction (refer to A type)	A	Agency			6 or A	Sell short exempt or Cross short exempt
C	Proprietary, Non-Algorithmic Programing Trade (non-index arbitrage)	P	Principal	1 3 D	Program Trade Non-Index Arbitrage Non-algorithmic		
D	Program Order, index arb, for Member firm/org	P	Principal	1 2	Program Trade Index Arbitrage		
E	Short Exempt Transaction for Principal (was incorrectly	P	Principal			6 or A	Sell short exempt or Cross short exempt

	identified in the FIX spec as "Registered Equity Market Maker trades")						
F	Short exempt transaction (refer to W type)	W	Agent for Other Member			6 or A	Sell short exempt or Cross short exempt
H	Short exempt transaction (refer to I type)	I	Individual			6 or A	Sell short exempt or Cross short exempt
I	Individual Investor, single order	I	Individual				
J	Proprietary, Algorithmic Program Trade (non-index arbitrage)	P	Principal	1 3 E	Program Trade Non-Index Arbitrage Algorithmic		
K	Agency, Algorithmic Program Trade (non-index arbitrage)	I or A or W	Individual or Agency or Agent for other member	1 3 E	Program Trade Non-Index Arbitrage Algorithmic		
L	Short exempt transaction for member competing market-maker affiliated with the firm clearing the trade (refer to P and O types)	P	Principal	4	Competing Market Maker	6 or A	Sell short exempt or Cross short exempt
M	Program Order, index arb, agent for other member	W	Agent for Other Member	1 2	Program Trade Index Arbitrage		
N	Agent for other member, Non-algorithmic Porgram Trade (non-index arbitrage)	W	Agent for Other Member	1 3 D	Program Trade Non-Index Arbitrage Non-algorithmic		
O	Proprietary transactions for competing market-maker that is affiliated with the clearing member (was incorrectly identified in the FIX spec as "Competing dealer trades")	P	Principal	4	Competing Market Maker		
P	Principal	P	Principal				
R	Transactions for the account of a non-member competing market maker (was incorrectly identified in the FIX spec as	A	Agency	4	Competing Market Maker		

	"Competing dealer trades")						
S	Specialist trades	P	Principal	5	Acting as Market Maker or Specialist in the security		
T	Transactions for the account of an unaffiliated member's competing market maker (was incorrectly identified in the FIX spec as "Competing dealer trades")	W	Agent for Other Member	5	Acting as Market Maker or Specialist in the security		
U	Agency, Index Arbitrage	A or I	Agency or Individual	1 2	Program Trade Index Arbitrage		
W	All other orders as agent for other member	W	Agent for Other Member				
X	Short exempt transaction for member competing market-maker not affiliated with the firm clearing the trade (refer to W and T types)	W	Agent for Other Member	4	Competing Market Maker	6 or A	Sell short exempt or Cross short exempt
Y	Agency, Non-Algorithmic Program Trade (non-index arbitrage)	A or I	Agency or Individual	1 3 D	Program Trade Non-Index Arbitrage Non-algorithmic		
Z	Short exempt transaction for non-member competing market-maker (refer to A and R types)	A	Agency	4	Competing Market Maker	6 or A	Sell short exempt or Cross short exempt

Japanese Equity Markets:

OrderCapacity is used to specify whether order is Agency or Principal.

Valid values:
A = Agency single order
P = Principal

Other Markets:

All or a subset of the OrderCapacity and OrderRestrictions field values defined in the field reference may

be applicable for other markets. Future markets will be included in this section as they are defined and brought forward to the FIX Technical Committee.

Example Usage of PartyRole="Investor ID"

Two fields, PartyID and PartyIDSource, facilitate the passing of exchange required ID's when PartyRole="Investor ID". At present, regulatory requirements require the exchange's Investor ID be provided when orders are placed in Taiwan, China (Shenzhen and Shanghai), and Korea. At present, India, Malaysia and Poland have regulatory requirements requiring the exchange's Investor ID be provided post-trade.

When placing an order on behalf of multiple distinct Investor ID values, then multiple orders will be sent at the same time with each representing a single Investor ID. For example, three funds/sub-accounts in Taiwan would result in three Investor ID values and thus would also result in three orders.

Note that the Investor ID value is not the same as the customer's AllocAccount field nor is there necessarily a one-to-one mapping between AllocAccount and Investor ID. In addition, in Korea one Investor ID value may represent multiple accounts. Thus, account pre-trade allocation (See Volume 5: "Example Usage of Allocations") can still take place in addition to PartyRole="Investor ID", PartyID, and PartyIDSource usage.

Format of the Party ID field (PartyRole="Investor ID")

PartyIDSource	Format of PartyID (Investor ID) value
Korean Investor ID	Single digit to six digits
Taiwanese Qualified Foreign Investor ID QFII / FID	Eight digits
Taiwanese Trading Account	Seven digits
Malaysian Central Depository (MCD) number	Fifteen digits
Chinese B Share (Shenzhen and Shanghai)	Nine digits
Note: All Investor ID values above should be provided in PartyID as numeric only (i.e. exclude alpha-numeric characters such as dashes).	

Example Representations of Orders

Symbol	Quantity	Side	OrdType	PartyIDSource	PartyID (Investor ID)	Comments
00660.KS	1000	Buy	Market	1	3452	Korean ID provided
00660.KS	3000	Buy	Market	1	232	Different Korean ID provided
2330.TW	3000	Sell	Market	2	90567878	QFII/FID given and Sell-side derives Trading Account based on QFII/FID and Buy-side Client ID
2330.TW	2000	Sell	Limit	3	9901234	Trading Account given
STAR.KL	1000	Sell	Market	4	4567895624658667	MCD given for T+0, pre- or post-trade

KOREA

Buy 10,000 Hyundai Elec for 3 funds/sub-accounts sharing same ID

Symbol	Quantity	Side	OrdType	PartyIDSourc	PartyID (Investor ID)
00660.KS	10,000	Buy	Market	1	3452

Buy 10,000 Hyundai Elec for 3 funds/sub-accounts sharing 2 ID's (sent as 2 orders)

Symbol	Quantity	Side	OrdType	PartyIDSourc	PartyID (Investor ID)	AllocAcc	AllocQty
00660.KS	4000	Buy	Market	1	3452	B56-78	4000
00660.KS	6000	Buy	Market	1	56	B56-48	2000
						C24-67	4000

Note: AllocAccount and AllocQty are optional and are not a substitute for PartyID (Investor ID) value (nor used to lookup PartyID (Investor ID)).

TAIWAN

Buy 12,000 TSMC for 3 funds/sub-accounts for 4000 each (sent as 3 orders)

Symbol	Quantity	Side	OrdType	PartyIDSourc	PartyID (Investor ID)
2330.TW	4000	Buy	Market	3	9903327
2330.TW	4000	Buy	Market	3	9925562
2330.TW	4000	Buy	Market	3	9903562

Additional Notes:

- Any change to the PartyID (Investor ID) post submission must be made through the allocation message – you cannot amend PartyID (Investor ID).

- If PartyIDSource and PartyID (Investor ID) provided are not valid for PartyRole="Investor ID", the sell-side will send an Execution Report with OrdRejReason of "Invalid Investor ID".

- These fields are not to be used to determine the routing of an order to an Exchange (value of PartyID is not a substitute for ExDestination).

CATEGORY: ORDER MASS HANDLING

Order Mass Cancel Request

The order mass cancel request message requests the cancelation of **all** of the remaining quantity of a group of orders matching criteria specified within the request. NOTE: This message can only be used to cancel order messages (reduce the full quantity).

An order mass cancel request is assigned a ClOrdID and is treated as a separate entity. The order mass cancel request is acknowledged using an Order Mass Cancel Report. The Order Mass Cancel Report will contain the ClOrdID that was specified on the Order Mass Cancel Request. The ClOrdID assigned to the cancel request must be unique amongst the ClOrdID assigned to regular orders, replacement orders, cancel requests, and order mass cancel requests.

An immediate response to this message is required. It is recommended that an ExecutionRpt with ExecType=Pending Cancel be sent unless the Order Mass Cancel Request can be immediately accepted (ExecutionRpt with ExecType=Canceled) or rejected (Order Cancel Reject message).

Specifying order cancellation criteria is specified using the MassCancelRequestType field:

Field Value	Description	Explanation
1	Cancel orders for a security	Cancel orders that match the security identification block, all fields required to uniquely qualify the security should be specified.
2	Cancel orders for an Underlying security	Cancel orders that match the underlying security identification block, all fields required to uniquely identify the underlying security should be populated
3	Cancel orders for a Product	Cancel orders for a specific type of Product (high-level security classification). Only Product should be specified
4	Cancel orders for a CFICode	Cancel orders for a specific type of CFICode (security classification). Only CFICode should be specified
5	Cancel orders for a SecurityType	Cancel orders for a specific type of security. Only SecurityType should be specified
6	Cancel orders for a trading session	Cancel orders for a specific trading session, TradingSessionID must be specified.
7	Cancel all orders	Cancel all orders for the firm identified using this FIX connection
8	Cancel orders for a market	Cancel all orders for a specific market. MarketID must be specified.
9	Cancel orders for a market segment	Cancel all orders for a specific market segment. MarketSegmentID must be specified.
A	Cancel orders for a security group	Cancel all orders for a specific security group. SecurityGroup must be specified.

Example uses of MassCancelRequestType with Qualifiers:

Cancel for a Symbol

Cancel for an underlying

Cancel orders on one side of a market for a symbol

Cancel orders for a specific option series

Cancel all orders

Cancel all orders on one side of a market

Cancel all money market orders

Cancel all common stock orders

Cancel all orders for a trading session

Cancel all orders for a trading session on one side of a market

Cancel all orders for a trading session for an underlying on one side of a market

The format of the Order Mass Cancel Request message is:

Order Mass Cancel Request

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = q (lowercase Q)
11	ClOrdID	Y	Unique ID of Order Mass Cancel Request as assigned by the institution.
526	SecondaryClOrdID	N	
530	MassCancelRequestType	Y	Specifies the type of cancellation requested
336	TradingSessionID	N	Trading Session in which orders are to be canceled
625	TradingSessionSubID	N	
component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "common components of application messages"
component block <Instrument>		N	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
component block <UnderlyingInstrument>		N	Insert here the set of "UnderlyingInstrument" (underlying symbology) fields defined in "Common Components of Application Messages"
1301	MarketID	N	Required for MassCancelRequestType = 8 (Cancel orders for a market)
1300	MarketSegmentID	N	Required for MassCancelRequestType = 9 (Cancel orders for a market segment)
54	Side	N	Optional qualifier used to indicate the side of the market for which orders are to be canceled. Absence of this field indicates that orders are to be canceled regardless of side.

60	TransactTime	Y	Time this order request was initiated/released by the trader or trading system.
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element OrdMassCxlReq

Order Mass Cancel Report

The Order Mass Cancel Report is used to acknowledge an Order Mass Cancel Request. Note that each affected order that is canceled is acknowledged with a separate Execution Report or Order Cancel Reject message.

Order Mass Cancel Report

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = r (lowercase R)
11	ClOrdID	N	ClOrdID provided on the Order Mass Cancel Request. Unavailable in case of an unsolicited report, such as after a trading halt or a corporate action requiring the deletion of outstanding orders.
526	SecondaryClOrdID	N	
37	OrderID	Y	(Deprecated in FIX.5.0SP1)Unique Identifier for the Order Mass Cancel Request assigned by the recipient of the Order Mass Cancel Request.
1369	MassActionReportID	Y	Unique Identifier for the Order Mass Cancel Report assigned by the recipient of the Order Mass Cancel Request
198	SecondaryOrderID	N	(Deprecated in FIX.5.0SP1)Secondary Order ID assigned by the recipient of the Order Mass Cancel Request.
530	MassCancelRequestType	Y	Order Mass Cancel Request Type accepted by the system
531	MassCancelResponse	Y	Indicates the action taken by the counterparty order handling system as a result of the Cancel Request 0 - Indicates Order Mass Cancel Request was rejected.
532	MassCancelRejectReason	N	Indicates why Order Mass Cancel Request was rejected Required if MassCancelResponse = 0
533	TotalAffectedOrders	N	Optional field used to indicate the total number of orders affected by the Order Mass Cancel Request
Start of Component block, expanded in line < AffectedOrdGrp >			
534	NoAffectedOrders	N	Optional field used to indicate the number of order identifiers for orders affected by the Order Mass Cancel Request. Must be followed with OrigClOrdID as the next field
→ 41	OrigClOrdID	N	Required if NoAffectedOrders > 0 and must be the first repeating field in the group. Indicates the client order id of an order affected by this request. If order(s) were manually delivered (or otherwise not delivered over FIX and not assigned a ClOrdID) this field should contain string "MANUAL".
→ 535	AffectedOrderID	N	Contains the OrderID assigned by the counterparty of an affected order. Not required as part of the repeating group if OrigClOrdID(41) has a value other than "MANUAL".

→	536	AffectedSecondaryOrderID	N	Contains the SecondaryOrderID assigned by the counterparty of an affected order. Not required as part of the repeating group
End of Component block, expanded in line < AffectedOrdGrp >				
Start of Component block, expanded in line < NotAffectedOrdersGrp >				
1370	NoNotAffectedOrders		N	Optional field used to indicate the number of order identifiers for orders not affected by the request. Must be followed with OrigClOrdID(41) as the next field.
→	1372	NotAffOrigClOrdID	N	Required if NoNotAffectedOrders(1370) > 0 and must be the first repeating field in the group. Indicates the client order id of an order not affected by the request. If order(s) were manually delivered (or otherwise not delivered over FIX and not assigned a ClOrdID) this field should contain string "MANUAL".
→	1371	NotAffectedOrderID	N	Contains the OrderID assigned by the counterparty of an unaffected order. Not required as part of the repeating group if NotAffOrigClOrdID(1372) has a value other than "MANUAL".
End of Component block, expanded in line < NotAffectedOrdersGrp >				
336	TradingSessionID		N	Trading Session in which orders are to be canceled
625	TradingSessionSubID		N	
component block <Parties>			N	Insert here the set of "Parties" (firm identification) fields defined in "common components of application messages"
component block <Instrument>			N	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
component block <UnderlyingInstrument>			N	Insert here the set of "UnderlyingInstrument" (underlying symbology) fields defined in "Common Components of Application Messages"
1301	MarketID		N	
1300	MarketSegmentID		N	
54	Side		N	Side of the market specified on the Order Mass Cancel Request
60	TransactTime		N	Time this report was initiated/released by the sells-side (broker, exchange, ECN) or sell-side executing system.
58	Text		N	
354	EncodedTextLen		N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText		N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer			Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element OrdMassCxlRpt

Order Mass Status Request

The order mass status request message requests the status for orders matching criteria specified within the request.

A mass status request is assigned a ClOrdID and is treated as a separate entity.

ExecutionReports with ExecType="Order Status" are returned for all orders matching the criteria provided on the request.

Specifying order selection criteria is specified using the MassStatusReqType field:

Field Value	Description	Explanation
1	Status for all orders for a security	Return status on orders that match the security identification block, all fields required to uniquely qualify the security should be specified.
2	Status for all orders for an Underlying security	Return status on orders that match the underlying security identification block, all fields required to uniquely identify the underlying security should be populated
3	Status for all orders for a Product	Return status on orders for a specific type of Product (high-level security classification), Only Product should be specified
4	Status for all orders for a CFICode	Return status on orders for a specific type of CFICode (security classification), Only CFICode should be specified
5	Status for all orders for a SecurityType	Return status on orders for a specific type of security, Only SecurityType should be specified
6	Status for all orders for a trading session	Return status on orders for a specific trading session, TradingSessionID must be specified.
7	Status for all orders	Return status on all orders for the firm identified using this FIX connection
8	Status for order belonging to a PartyID	Status all orders belonging to a PartyID
9	Status for all orders for an Account	Status for orders for an account.

Example uses of MassStatusReqType with Qualifiers:

- Status for a Symbol
- Status for an underlying
- Status orders on one side of a market for a symbol
- Status orders for a specific option series
- Status all orders
- Status all orders on one side of a market
- Status all money market orders
- Status all common stock orders
- Status all orders for a trading session
- Status all orders for a trading session on one side of a market
- Status all orders for a trading session for an underlying on one side of a market

- Status all orders belonging to a PartyID.
- Status all orders belonging to an Account

The format of the Order Mass Status Request message is:

Order Mass Status Request

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = AF
584	MassStatusReqID	Y	Unique ID of mass status request as assigned by the institution.
585	MassStatusReqType	Y	Specifies the scope of the mass status request
component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
1	Account	N	Account
660	AcctIDSource	N	
336	TradingSessionID	N	Trading Session
625	TradingSessionSubID	N	
component block <Instrument>		N	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
component block <UnderlyingInstrument>		N	Insert here the set of "UnderlyingInstrument" (underlying symbology) fields defined in "Common Components of Application Messages"
54	Side	N	Optional qualifier used to indicate the side of the market for which orders will be returned.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element OrdMassStatReq

Order Mass Action Request

The Order Mass Action Request message can be used to request the suspension or release of a group of orders that match the criteria specified within the request. This is equivalent to individual Order Cancel Replace Requests for each order with or without adding "S" to the ExecInst values. It can also be used for mass order cancellation.

An Order Mass Action Request is assigned a ClOrdID and is treated as a separate entity. The Order Mass Action Request is acknowledged using an Order Mass Action Report. The Order Mass Action Report will contain the ClOrdID that was specified on the Order Mass Action Request. The ClOrdID assigned to the suspension or release request must be unique amongst the ClOrdID assigned to regular orders, replacement orders, cancel requests, etc.

An immediate response to this message is required. It is recommended that an Execution Report with ExecType=Pending Replace (or Pending Cancel if used for mass cancellation) be sent unless the Order Mass Action Request can be immediately accepted (Execution Report with ExecType=Replaced or Canceled).

Specifying filtering criteria is done using the MassActionType field.

Order Mass Action Request

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = CA
11	ClOrdID	Y	Unique ID of Order Mass Action Request as assigned by the institution.
526	SecondaryClOrdID	N	
1373	MassActionType	Y	Specifies the type of action requested
1374	MassActionScope	Y	Specifies the scope of the action
1301	MarketID	N	MarketID for which orders are to be affected
1300	MarketSegmentID	N	MarketSegmentID for which orders are to be affected
336	TradingSessionID	N	Trading Session in which orders are to be affected
625	TradingSessionSubID	N	
component block <Parties>		N	
component block <Instrument>		N	
component block <UnderlyingInstrument>		N	
54	Side	N	
60	TransactTime	Y	
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element OrdMassActReq

Order Mass Action Report

The Order Mass Action Report is used to acknowledge an Order Mass Action Request. Note that each affected order that is suspended or released or canceled is acknowledged with a separate Execution Report for each order.

Order Mass Action Report

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = BZ
11	ClOrdID	N	ClOrdID provided on the Order Mass Action Request.
526	SecondaryClOrdID	N	
1369	MassActionReportID	Y	Unique Identifier for the Order Mass Action Report
1373	MassActionType	Y	Order Mass Action Request Type accepted by the system
1374	MassActionScope	Y	Specifies the scope of the action
1375	MassActionResponse	Y	Indicates the action taken by the counterparty order handling system as a result of the Action Request 0 - Indicates Order Mass Action Request was rejected.
1376	MassActionRejectReason	N	Indicates why Order Mass Action Request was rejected Required if MassActionResponse = 0
533	TotalAffectedOrders	N	Optional field used to indicate the total number of orders affected by the Order Mass Action Request
Start of Component block, expanded in line < AffectedOrdGrp >			
534	NoAffectedOrders	N	Optional field used to indicate the number of order identifiers for orders affected by the Order Mass Cancel Request. Must be followed with OrigClOrdID as the next field
→	41 OrigClOrdID	N	Required if NoAffectedOrders > 0 and must be the first repeating field in the group. Indicates the client order id of an order affected by this request. If order(s) were manually delivered (or otherwise not delivered over FIX and not assigned a ClOrdID) this field should contain string "MANUAL".
→	535 AffectedOrderID	N	Contains the OrderID assigned by the counterparty of an affected order. Not required as part of the repeating group if OrigClOrdID(41) has a value other than "MANUAL".
→	536 AffectedSecondaryOrderID	N	Contains the SecondaryOrderID assigned by the counterparty of an affected order. Not required as part of the repeating group
End of Component block, expanded in line < AffectedOrdGrp >			
Start of Component block, expanded in line < NotAffectedOrdersGrp >			
1370	NoNotAffectedOrders	N	Optional field used to indicate the number of order identifiers for orders not affected by the request. Must be followed with OrigClOrdID(41) as the next field.

→	1372	NotAffOrigClOrdID	N	Required if NoNotAffectedOrders(1370) > 0 and must be the first repeating field in the group. Indicates the client order id of an order not affected by the request. If order(s) were manually delivered (or otherwise not delivered over FIX and not assigned a ClOrdID) this field should contain string "MANUAL".
→	1371	NotAffectedOrderID	N	Contains the OrderID assigned by the counterparty of an unaffected order. Not required as part of the repeating group if NotAffOrigClOrdID(1372) has a value other than "MANUAL".
End of Component block, expanded in line < NotAffectedOrdersGrp >				
1301	MarketID		N	MarketID for which orders are to be affected
1300	MarketSegmentID		N	MarketSegmentID for which orders are to be affected
336	TradingSessionID		N	TradingSessionID for which orders are to be affected
625	TradingSessionSubID		N	TradingSessionSubID for which orders are to be affected
component block <Parties>			N	
component block <Instrument>			N	
component block <UnderlyingInstrument>			N	
54	Side		N	Side of the market specified on the Order Mass Action Request
60	TransactTime		N	Time this report was initiated/released by the sells-side (broker, exchange, ECN) or sell-side executing system.
58	Text		N	
354	EncodedTextLen		N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText		N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer			Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element OrdMassActRpt

CATEGORY: CROSS ORDERS

Background

FIX provides support for a cross order using Side[54]=Cross on a New Order Single Message. For many markets the New Order – Single does not provide enough information about the counterparties of a trade to meet regulatory and post-trade requirements. Markets that find the use of a New Order – Single Message with Side[54]=Cross adequate for cross trading – can continue to use this implementation. When additional information regarding the counterparty to the cross trade is required – the Cross Order message should be used.

The Japanese Exchange Working Group proposed the creation of a Cross Order message that would elaborate both counterparties involved in the cross for a security. Companion Cross Order Cancel Replace Requests and Cross Order Cancel Requests were also proposed.

Prioritization of a side of a cross order

Some markets permit one side or the other of the cross order to be prioritized for execution. A new field, CrossPrioritization[550] indicates which side of the cross order will be prioritized for execution. The definition of prioritization is left to the market. In some markets the prioritized side will be guaranteed execution. In other markets, prioritization means that the prioritized side will be applied to the market first.

Classification of cross trades

Four types of cross trades have been identified.

1. Cross Trade that is executed completely or not. Both sides are treated in the same manner. This is equivalent to Fill or Kill type behavior, where the cross order meets the crossing criteria – within the market and is executed or it is rejected.
2. Cross Trade that is executed partially and the rest is canceled. One side is fully executed, the other side is partially executed with the remainder being canceled. This is equivalent to an Immediate or Cancel on the other side. Note: The CrossPrioritization[550] field is used to indicate which side should fully execute in this scenario.
3. Cross trade that is partially executed with the unfilled portions remaining active. One side of the cross is fully executed as denoted with the CrossPrioritization[550] field, but the unfilled portion remains active.
4. Cross trade is executed with existing orders with the same price. In the case other orders exist with the same price, the quantity of the Cross is executed against the existing orders and quotes, the remainder of the cross is executed against the other side of the cross. The two sides potentially have different quantities. The use of CrossPrioritization[550] field is used to indicate which side of the cross will be executed against the existing market.

Execution Reporting for cross orders

Execution reporting for cross orders is done by side. Execution Reports are sent using the ClOrdID[11], OrigClOrdID[41], OrderID[37] of the side of the cross order being reported. The CrossID[548] and the CrossType[549] fields are provided as optional fields on the Execution Report. The CrossID[548] must be provided on the Execution Reports for sides of Cross Orders.

Cross Order Handling Rules

If one side of the cross order is invalid then the entire cross order is rejected. Markets should not accept one side of the cross order without accepting the other side.

The CrossType[549] field defines the proper processing of cross orders once the cross order has been accepted.

The order state changes for each leg are reported independently using separate Execution Reports for each side.

Acknowledgement of a Cross Order

The following shows typical message flows for the acknowledgement of cross orders.

	Broker		Market
1	**Sends Cross Order**	→	Receives Cross Order and processes
2			**OPTIONAL: Market conditionally accepts or fully rejects the order (This optional state change is assumed for all remaining examples and will not be elaborated to save space).**
		←	Sends *Execution Report* Side 1 ClOrdID(1) OrdStatus=Pending New or OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field. *If one side is rejected then the entire cross order is rejected.*
		←	If the Cross Order contains two sides: Sends *Execution Report* Side 2 ClOrdID(2) OrdStatus=Pending New or OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.
3			**Order is accepted or rejected by market**
		←	Sends *Execution Report* Side 1 ClOrdID(1) OrdStatus=New or OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field. *If one side is rejected then the entire cross order is rejected.*
		←	If the Cross Order contains two sides: Sends *Execution Report* Side 2 ClOrdID(2) OrdStatus=New or OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.

Message Flow for cross order with CrossType=1 with only one side of the order provided

In the case where the broker is crossing the order and no further identification is required as part of the order – the cross order can contain one leg. The cross order is executed fully or canceled.

	Broker		Market
1	**Sends Cross Order with one side only**	→	Receives Cross Order and processes
2			**Order Accepted or Rejected by Market**
		←	Sends Execution Report for ClOrdID(1) OrdStatus=New or OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.
3			**Order fully executed or is canceled by market**
		←	Sends Execution Report for ClOrdID(1) OrdStatus=FILL or OrdStatus=CANCEL. The reason for the cancel should be specified in the Text[58] field.

Message Flow for cross order with CrossType=1 when both sides of the cross order provided

In this example two sides of the cross are explicitly provided on the cross order. The cross order is executed fully or canceled.

	Broker		Market
1	**Sends Cross Order with both sides provided**	→	Receives Cross Order and processes
2a			**Order Accepted by Market**
		←	Sends Execution Report for Side 1 ClOrdID(1) OrdStatus=New
		←	Sends Execution Report for Side 2 ClOrdID(2) OrdStatus=New
2b			**Order Rejected by Market**
		←	Sends Execution Report for Side 1 ClOrdID(1) OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.
		←	Sends Execution Report for Side 2 ClOrdID(2) OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.
3a			**Order is fully executed by market**
		←	Sends Execution Report for ClOrdID(1) OrdStatus=FILLED
		←	Sends Execution Report for ClOrdID(2) OrdStatus=FILLED
3b			**Order is canceled by market**
		←	Sends Execution Report for ClOrdID(1) OrdStatus=FILLED or OrdStatus=CANCELED. The reason for the cancel should be specified in the Text[58] field.
		←	Sends Execution Report for ClOrdID(2) OrdStatus=FILLLED or OrdStatus=CANCELED. The reason for the cancel should be specified in the Text[58] field.

Message Flow for cross order with CrossType=2

In the following example the cross order contains a buy and sell order. The buy side is prioritized and in the case of CrossType=2 will be fully executed. In the following example – the sell side is not fully executed – the balance being canceled.

	Broker		Market
1	**Sends Cross Order with CrossType=2 and CrossPrioritization = Buy Side**	→	Receives Cross Order and processes
2a			**Order Accepted by Market**
		←	Sends Execution Report for Buy Side 1 ClOrdID(1) OrdStatus=New
		←	Sends Execution Report for Sell Side 2 ClOrdID(2) OrdStatus=New
2b			**Order Rejected by Market**
		←	Sends Execution Report for Buy Side 1 ClOrdID(1) OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.
		←	Sends Execution Report for Sell Side 2 ClOrdID(2) OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.
3			**Buy Side of Cross Order is partially filled**
		←	Sends Execution Report for Buy Side ClOrdID(1) OrdStatus=PARTIALLY FILLED
4			**Cross Order is partially crossed with FILLED status of Buy Side**
		←	Sends Execution Report for Buy Side ClOrdID(1) OrdStatus=FILLED
		←	Sends Execution Report for Sell Side ClOrdID(2) OrdStatus=PARTIALLY FILLED
5			**Remaining quantity of Sell Side is canceled by the market automatically**
		←	Sends Execution Report for Sell Side ClOrdID(2) OrdStatus=CANCELED. The reason for the cancel should be specified in the Text[58] field.

Message Flow for cross order with CrossType=3

In this scenario – the cross order is executed with the buy side prioritized. The buy side is fully executed. The remaining part of the Sell side remains active and is eventually filled or canceled.

	Broker		Market
1	**Sends Cross Order with CrossType=3 and CrossPrioritization = Buy Side**	→	Receives Cross Order and processes
2a			**Order Accepted by Market**
		←	Sends Execution Report for Buy Side 1 ClOrdID(1) OrdStatus=New
		←	Sends Execution Report for Sell Side 2 ClOrdID(2) OrdStatus=New
2b			**Order Rejected by Market**
		←	Sends Execution Report for Buy Side 1 ClOrdID(1) OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.
		←	Sends Execution Report for Sell Side 2 ClOrdID(2) OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.
3			**Buy Side of Cross Order is partially filled**
		←	Sends Execution Report for Buy Side ClOrdID(1) OrdStatus=PARTIALLY FILLED
4			**Cross Order is partially crossed with FILLED status of Buy Side**
		←	Sends Execution Report for Buy Side ClOrdID(1) OrdStatus=FILLED
		←	Sends Execution Report for Sell Side ClOrdID(2) OrdStatus=PARTIALLY FILLED
5a			**Remaining quantity of Sell Side remains active and is later filled**
		←	Sends Execution Report for Sell Side ClOrdID(2) OrdStatus=FILLED
5b			**Remaining quantity of Sell Side that remains active is canceled by request**
	Order Cancel Request submitted after cross order completes to cancel remaining unfilled portion of sell side	→	Order Cancel Replace is applied to the market.
		←	Sends Execution Report for Sell Side ClOrdID(2) OrdStatus= Pending Cancel
		←	Sends Execution Report for Sell Side ClOrdID(2) OrdStatus=Canceled

Message Flow for cross order with CrossType=4

In the following example the buy order is prioritized. The buy side will trade against orders in the book at the same price. The sell side of the cross will trade with the remaining quantity of the buy side. The sell side will be filled at a lower quantity than the buy side that executed against existing orders. NOTE: It is possible for the sell side to be filled with no quantity – if sufficient sell orders exist in the book.

	Broker		Market
1	**Sends Cross Order with CrossType=4 and CrossPrioritization = Buy Side**	→	Receives Cross Order and processes
2a			**Order Accepted by Market**
		←	Sends Execution Report for Buy Side 1 ClOrdID(1) OrdStatus=New
		←	Sends Execution Report for Sell Side 2 ClOrdID(2) OrdStatus=New
2b			**Order Rejected by Market**
		←	Sends Execution Report for Buy Side 1 ClOrdID(1) OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.
		←	Sends Execution Report for Sell Side 2 ClOrdID(2) OrdStatus=Rejected. The reason for the reject should be specified in the Text[58] field.
3			**Buy side of theCross Order is partially crossed with sell orders in the book**
		←	Sends Execution Report for Buy Side ClOrdID(1) OrdStatus=PARTIALLY FILLED
4			**Cross Order is completed when remaining Buy Side quantity is filled against the Sell Side of the cross**
		←	Sends Execution Report for Buy Side ClOrdID(1) OrdStatus=FILLED
		←	Sends Execution Report for Sell Side ClOrdID(2) OrdStatus=FILLED even though the filled quantity of the sell side < filled quantity on buy side.

New Order - Cross

Used to submit a cross order into a market. The cross order contains two order sides (a buy and a sell). The cross order is identified by its CrossID.

New Order - Cross

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = s (lowercase S)
548	CrossID	Y	
549	CrossType	Y	
550	CrossPrioritization	Y	
component block <RootParties>		N	Insert here the set of "Root Parties" fields defined in "common components of application messages" Used for acting parties that applies to the whole message, not individual sides.
Start of Component block, expanded in line < SideCrossOrdModGrp >			
552	NoSides	Y	Must be 1 or 2 1 or 2 if CrossType=1 2 otherwise
→	54 Side	Y	
→	41 OrigClOrdID	N	Required when referring to orders that where electronically submitted over FIX or otherwise assigned a ClOrdID(11)
→	11 ClOrdID	Y	Unique identifier of the order as assigned by institution or by the intermediary with closest association with the investor.
→	526 SecondaryClOrdID	N	
→	583 ClOrdLinkID	N	
→	component block <Parties>	N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
→	229 TradeOriginationDate	N	
→	75 TradeDate	N	
→	1 Account	N	
→	660 AcctIDSource	N	
→	581 AccountType	N	
→	589 DayBookingInst	N	
→	590 BookingUnit	N	
→	591 PreallocMethod	N	
→	70 AllocID	N	Use to assign an identifier to the block of preallocations

→	*Start of Component block, expanded in line < PreAllocGrp >*				
→	78	NoAllocs		N	Number of repeating groups for pre-trade allocation
→	→	79	AllocAccount	N	Required if NoAllocs > 0. Must be first field in repeating group.
→	→	661	AllocAcctIDSource	N	
→	→	736	AllocSettlCurrency	N	
→	→	467	IndividualAllocID	N	
→	→	component block <NestedParties>		N	Insert here the set of "Nested Parties" (firm identification "nested" within additional repeating group) fields defined in "Common Components of Application Messages" Used for NestedPartyRole=Clearing Firm
→	→	80	AllocQty	N	
→	*End of Component block, expanded in line < PreAllocGrp >*				
→	854	QtyType		N	
→	component block <OrderQtyData>			Y	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages"
→	component block <CommissionData>			N	Insert here the set of "CommissionData" fields defined in "Common Components of Application Messages"
→	528	OrderCapacity		N	
→	529	OrderRestrictions		N	
→	1091	PreTradeAnonymity		N	
→	582	CustOrderCapacity		N	
→	121	ForexReq		N	Indicates that broker is requested to execute a Forex accommodation trade in conjunction with the security trade.
→	120	SettlCurrency		N	Required if ForexReq = Y.
→	775	BookingType		N	Method for booking out this order. Used when notifying a broker that an order to be settled by that broker is to be booked out as an OTC derivative (e.g. CFD or similar). Absence of this field implies regular booking.
→	58	Text		N	
→	354	EncodedTextLen		N	Must be set if EncodedText field is specified and must immediately precede it.
→	355	EncodedText		N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
→	77	PositionEffect		N	For use in derivatives omnibus accounting
→	203	CoveredOrUncovered		N	For use with derivatives, such as options

→	544	CashMargin	N	
→	635	ClearingFeeIndicator	N	
→	377	SolicitedFlag	N	
→	659	SideComplianceID	N	
→	962	SideTimeInForce	N	Specifies how long the order as specified in the side stays in effect. Absence of this field indicates Day order.
End of Component block, expanded in line < SideCrossOrdModGrp >				
component block <Instrument>			Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
Start of Component block, expanded in line < UndInstrmtGrp >				
711	NoUnderlyings		N	Number of underlyings
→	component block <UnderlyingInstrument>		N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >				
Start of Component block, expanded in line < InstrmtLegGrp >				
555	NoLegs		N	Number of legs
→	component block <InstrumentLeg>		N	Must be provided if Number of legs > 0
End of Component block, expanded in line < InstrmtLegGrp >				
63	SettlType		N	
64	SettlDate		N	Takes precedence over SettlType value and conditionally required/omitted for specific SettlType values.
21	HandlInst		N	
18	ExecInst		N	Can contain multiple instructions, space delimited. If OrdType=P, exactly one of the following values (ExecInst = L, R, M, P, O, T, or W) must be specified.
110	MinQty		N	
1089	MatchIncrement		N	
1090	MaxPriceLevels		N	
component block <DisplayInstruction>			N	Insert here the set of "DisplayInstruction" fields defined in "common components of application messages"
111	MaxFloor		N	(Deprecated in FIX.5.0)
100	ExDestination		N	
1133	ExDestinationIDSource		N	
Start of Component block, expanded in line < TrdgSesGrp >				
386	NoTradingSessions		N	Specifies the number of repeating TradingSessionIDs
→	336	TradingSessionID	N	Required if NoTradingSessions is > 0.

→	625	TradingSessionSubID	N	
End of Component block, expanded in line < TrdgSesGrp >				
81	ProcessCode		N	Used to identify soft trades at order entry.
140	PrevClosePx		N	Useful for verifying security identification
114	LocateReqd		N	Required for short sell orders
60	TransactTime		Y	Time this order request was initiated/released by the trader, trading system, or intermediary.
483	TransBkdTime		N	A date and time stamp to indicate when this order was booked with the agent prior to submission to the VMU
component block <Stipulations>			N	Insert here the set of "Stipulations" (repeating group of Fixed Income stipulations) fields defined in "Common Components of Application Messages"
40	OrdType		Y	
423	PriceType		N	
44	Price		N	Required for limit OrdTypes. For F/X orders, should be the "all-in" rate (spot rate adjusted for forward points). Can be used to specify a limit price for a pegged order, previously indicated, etc.
1092	PriceProtectionScope		N	
99	StopPx		N	Required for OrdType = "Stop" or OrdType = "Stop limit".
component block <TriggeringInstruction>			N	Insert here the set of "TriggeringInstruction" fields defined in "common components of application messages"
component block <SpreadOrBenchmarkCurveData>			N	Insert here the set of "SpreadOrBenchmarkCurveData" (Fixed Income spread or benchmark curve) fields defined in "Common Components of Application Messages"
component block <YieldData>			N	Insert here the set of "YieldData" (yield-related) fields defined in "Common Components of Application Messages"
15	Currency		N	
376	ComplianceID		N	
23	IOIID		N	Required for Previously Indicated Orders (OrdType=E)
117	QuoteID		N	Required for Previously Quoted Orders (OrdType=D)
59	TimeInForce		N	Absence of this field indicates Day order
168	EffectiveTime		N	Can specify the time at which the order should be considered valid
432	ExpireDate		N	Conditionally required if TimeInForce = GTD and ExpireTime is not specified.
126	ExpireTime		N	Conditionally required if TimeInForce = GTD and ExpireDate is not specified.

427	GTBookingInst		N	States whether executions are booked out or accumulated on a partially filled GT order
210	MaxShow		N	(Deprecated in FIX.5.0)
component block <PegInstructions>			N	Insert here the set of "PegInstruction" fields defined in "Common Components of Application Messages"
component block <DiscretionInstructions>			N	Insert here the set of "DiscretionInstruction" fields defined in "Common Components of Application Messages"
847	TargetStrategy		N	The target strategy of the order
Start of Component block, expanded in line < StrategyParametersGrp >				
957	NoStrategyParameters		N	Indicates number of strategy parameters
→	958	StrategyParameterName	N	Name of parameter
→	959	StrategyParameterType	N	Datatype of the parameter.
→	960	StrategyParameterValue	N	Value of the parameter
End of Component block, expanded in line < StrategyParametersGrp >				
848	TargetStrategyParameters		N	(Deprecated in FIX.5.0)For further specification of the TargetStrategy
849	ParticipationRate		N	(Deprecated in FIX.5.0)Mandatory for a TargetStrategy=Participate order and specifies the target particpation rate. For other order types optionally specifies a volume limit (i.e. do not be more than this percent of the market volume)
480	CancellationRights		N	For CIV - Optional
481	MoneyLaunderingStatus		N	
513	RegistID		N	Reference to Registration Instructions message for this Order.
494	Designation		N	Supplementary registration information for this Order
StandardTrailer			Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details

Refer to FIXML element NewOrdCrss

Cross Order Cancel/Replace Request (a.k.a. Cross Order Modification Request)

Used to modify a cross order previously submitted using the New Order - Cross message. See Order Cancel Replace Request for details concerning message usage.

Refer to the Order Cancel Replace Request (a.k.a. Order Modification Request) message for restrictions on what fields can be changed during a cancel replace.

The Cross Order-specific fields, CrossType (tag 549) and CrossPrioritization (tag 550), can not be modified using the Cross Order Cancel Replace Request.

Cross Order Cancel / Replace Request (a.k.a. Cross Order Modification Request)

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = t (lowercase T)
37	OrderID	N	Unique identifier of most recent order as assigned by sell-side (broker, exchange, ECN).
548	CrossID	Y	CrossID for the replacement order
551	OrigCrossID	Y	Must match the CrossID of the previous cross order. Same order chaining mechanism as ClOrdID/OrigClOrdID with single order Cancel/Replace.
961	HostCrossID	N	Host assigned entity ID that can be used to reference all components of a cross; sides + strategy + legs
549	CrossType	Y	
550	CrossPrioritization	Y	
component block <RootParties>		N	Insert here the set of "Root Parties" fields defined in "common components of application messages" Used for acting parties that applies to the whole message, not individual sides.
Start of Component block, expanded in line < SideCrossOrdModGrp >			
552	NoSides	Y	Must be 1 or 2 1 or 2 if CrossType=1 2 otherwise
→ 54	Side	Y	
→ 41	OrigClOrdID	N	Required when referring to orders that where electronically submitted over FIX or otherwise assigned a ClOrdID(11)
→ 11	ClOrdID	Y	Unique identifier of the order as assigned by institution or by the intermediary with closest association with the investor.
→ 526	SecondaryClOrdID	N	
→ 583	ClOrdLinkID	N	
→ component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"

→	229	TradeOriginationDate		N	
→	75	TradeDate		N	
→	1	Account		N	
→	660	AcctIDSource		N	
→	581	AccountType		N	
→	589	DayBookingInst		N	
→	590	BookingUnit		N	
→	591	PreallocMethod		N	
→	70	AllocID		N	Use to assign an identifier to the block of preallocations
→	*Start of Component block, expanded in line < PreAllocGrp >*				
→	78	NoAllocs		N	Number of repeating groups for pre-trade allocation
→	→	79	AllocAccount	N	Required if NoAllocs > 0. Must be first field in repeating group.
→	→	661	AllocAcctIDSource	N	
→	→	736	AllocSettlCurrency	N	
→	→	467	IndividualAllocID	N	
→	→	component block <NestedParties>		N	Insert here the set of "Nested Parties" (firm identification "nested" within additional repeating group) fields defined in "Common Components of Application Messages" Used for NestedPartyRole=Clearing Firm
→	→	80	AllocQty	N	
→	*End of Component block, expanded in line < PreAllocGrp >*				
→	854	QtyType		N	
→	component block <OrderQtyData>			Y	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages"
→	component block <CommissionData>			N	Insert here the set of "CommissionData" fields defined in "Common Components of Application Messages"
→	528	OrderCapacity		N	
→	529	OrderRestrictions		N	
→	1091	PreTradeAnonymity		N	
→	582	CustOrderCapacity		N	
→	121	ForexReq		N	Indicates that broker is requested to execute a Forex accommodation trade in conjunction with the security trade.
→	120	SettlCurrency		N	Required if ForexReq = Y.
→	775	BookingType		N	Method for booking out this order. Used when notifying

				a broker that an order to be settled by that broker is to be booked out as an OTC derivative (e.g. CFD or similar). Absence of this field implies regular booking.
→	58	Text	N	
→	354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
→	355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
→	77	PositionEffect	N	For use in derivatives omnibus accounting
→	203	CoveredOrUncovered	N	For use with derivatives, such as options
→	544	CashMargin	N	
→	635	ClearingFeeIndicator	N	
→	377	SolicitedFlag	N	
→	659	SideComplianceID	N	
→	962	SideTimeInForce	N	Specifies how long the order as specified in the side stays in effect. Absence of this field indicates Day order.
End of Component block, expanded in line < SideCrossOrdModGrp >				
component block <Instrument>			Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
Start of Component block, expanded in line < UndInstrmtGrp >				
711	NoUnderlyings		N	Number of underlyings
→	component block <UnderlyingInstrument>		N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >				
Start of Component block, expanded in line < InstrmtLegGrp >				
555	NoLegs		N	Number of legs
→	component block <InstrumentLeg>		N	Must be provided if Number of legs > 0
End of Component block, expanded in line < InstrmtLegGrp >				
63	SettlType		N	
64	SettlDate		N	Takes precedence over SettlType value and conditionally required/omitted for specific SettlType values.
21	HandlInst		N	
18	ExecInst		N	Can contain multiple instructions, space delimited. If OrdType=P, exactly one of the following values (ExecInst = L, R, M, P, O, T, or W) must be specified.
110	MinQty		N	
1089	MatchIncrement		N	

1090	MaxPriceLevels		N	
component block <DisplayInstruction>			N	Insert here the set of "DisplayInstruction" fields defined in "common components of application messages"
111	MaxFloor		N	(Deprecated in FIX.5.0)
100	ExDestination		N	
1133	ExDestinationIDSource		N	
Start of Component block, expanded in line < TrdgSesGrp >				
386	NoTradingSessions		N	Specifies the number of repeating TradingSessionIDs
→	336	TradingSessionID	N	Required if NoTradingSessions is > 0.
→	625	TradingSessionSubID	N	
End of Component block, expanded in line < TrdgSesGrp >				
81	ProcessCode		N	Used to identify soft trades at order entry.
140	PrevClosePx		N	Useful for verifying security identification
114	LocateReqd		N	Required for short sell orders
60	TransactTime		Y	Time this order request was initiated/released by the trader, trading system, or intermediary.
483	TransBkdTime		N	A date and time stamp to indicate when this order was booked with the agent prior to submission to the VMU
component block <Stipulations>			N	Insert here the set of "Stipulations" (repeating group of Fixed Income stipulations) fields defined in "Common Components of Application Messages"
40	OrdType		Y	
423	PriceType		N	
44	Price		N	Required for limit OrdTypes. For F/X orders, should be the "all-in" rate (spot rate adjusted for forward points). Can be used to specify a limit price for a pegged order, previously indicated, etc.
1092	PriceProtectionScope		N	
99	StopPx		N	Required for OrdType = "Stop" or OrdType = "Stop limit".
component block <TriggeringInstruction>			N	Insert here the set of "TriggeringInstruction" fields defined in "common components of application messages"
component block <SpreadOrBenchmarkCurveData>			N	Insert here the set of "SpreadOrBenchmarkCurveData" (Fixed Income spread or benchmark curve) fields defined in "Common Components of Application Messages"
component block <YieldData>			N	Insert here the set of "YieldData" (yield-related) fields defined in "Common Components of Application Messages"
15	Currency		N	
376	ComplianceID		N	

23	IOIID		N	Required for Previously Indicated Orders (OrdType=E)
117	QuoteID		N	Required for Previously Quoted Orders (OrdType=D)
59	TimeInForce		N	Absence of this field indicates Day order
168	EffectiveTime		N	Can specify the time at which the order should be considered valid
432	ExpireDate		N	Conditionally required if TimeInForce = GTD and ExpireTime is not specified.
126	ExpireTime		N	Conditionally required if TimeInForce = GTD and ExpireDate is not specified.
427	GTBookingInst		N	States whether executions are booked out or accumulated on a partially filled GT order
210	MaxShow		N	(Deprecated in FIX.5.0)
component block <PegInstructions>			N	Insert here the set of "PegInstruction" fields defined in "Common Components of Application Messages"
component block <DiscretionInstructions>			N	Insert here the set of "DiscretionInstruction" fields defined in "Common Components of Application Messages"
847	TargetStrategy		N	The target strategy of the order
Start of Component block, expanded in line < StrategyParametersGrp >				
957	NoStrategyParameters		N	Indicates number of strategy parameters
→	958	StrategyParameterName	N	Name of parameter
→	959	StrategyParameterType	N	Datatype of the parameter.
→	960	StrategyParameterValue	N	Value of the parameter
End of Component block, expanded in line < StrategyParametersGrp >				
848	TargetStrategyParameters		N	(Deprecated in FIX.5.0)For further specification of the TargetStrategy
849	ParticipationRate		N	(Deprecated in FIX.5.0)Mandatory for a TargetStrategy=Participate order and specifies the target particpation rate. For other order types optionally specifies a volume limit (i.e. do not be more than this percent of the market volume)
480	CancellationRights		N	For CIV - Optional
481	MoneyLaunderingStatus		N	
513	RegistID		N	Reference to Registration Instructions message for this Order.
494	Designation		N	Supplementary registration information for this Order
StandardTrailer			Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element CrssOrdCxlRplcReq

Cross Order Cancel Request

Used to fully cancel the remaining open quantity of a cross order.

Cross Order Cancel Request

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = u (lowercase U)
37	OrderID	N	Unique identifier of most recent order as assigned by sell-side (broker, exchange, ECN).
548	CrossID	Y	CrossID for the replacement order
551	OrigCrossID	Y	Must match the CrossID of previous cross order. Same order chaining mechanism as ClOrdID/OrigClOrdID with single order Cancel/Replace.
961	HostCrossID	N	Host assigned entity ID that can be used to reference all components of a cross; sides + strategy + legs
549	CrossType	Y	
550	CrossPrioritization	Y	
component block <RootParties>		N	Insert here the set of "Root Parties" fields defined in "common components of application messages" Used for acting parties that applies to the whole message, not individual sides.
Start of Component block, expanded in line < SideCrossOrdCxlGrp >			
552	NoSides	Y	Must be 1 or 2
→	54 Side	Y	
→	41 OrigClOrdID	N	Required when referring to orders that where electronically submitted over FIX or otherwise assigned a ClOrdID(11).
→	11 ClOrdID	Y	Unique identifier of the order as assigned by institution or by the intermediary with closest association with the investor.
→	526 SecondaryClOrdID	N	
→	583 ClOrdLinkID	N	
→	586 OrigOrdModTime	N	
→	component block <Parties>	N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
→	229 TradeOriginationDate	N	
→	75 TradeDate	N	
→	component block <OrderQtyData>	Y	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages"
→	376 ComplianceID	N	

→	58	Text	N	
→	354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
→	355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
End of Component block, expanded in line < SideCrossOrdCxlGrp >				
component block <Instrument>			Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
Start of Component block, expanded in line < UndInstrmtGrp >				
711	NoUnderlyings		N	Number of underlyings
→	component block <UnderlyingInstrument>		N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >				
Start of Component block, expanded in line < InstrmtLegGrp >				
555	NoLegs		N	Number of legs
→	component block <InstrumentLeg>		N	Must be provided if Number of legs > 0
End of Component block, expanded in line < InstrmtLegGrp >				
60	TransactTime		Y	Time this order request was initiated/released by the trader, trading system, or intermediary.
StandardTrailer			Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element CrssOrdCxlReq

Cross Order Change Matrices

Cross Type 1

Scenario-1: Cancel (Before Exchange Crossing Session Time)

Time	Message Received	Message Sent	No Sides	Cross ID	OrigCrossID	Buy			Sell			OrdStatus	Comment
						ClOrdID	OrigClOrdID	Qty	ClOrdID	OrigClOrdID	Qty		
1	New Order - Cross		2	X		10		9000	20		9000		
2		Execution(Buy)		X		10						New	
		Execution(Sell)		X					20			New	
3	Cross Order Cancel		2	Y	X	11	10	9000	21	20	9000		
4		Execution(Buy)		Y	X	11	10					Pending Cancel	
		Execution(Sell)		Y	X				21	20		Pending Cancel	
5		Execution(Buy)		Y	X	11	10					Canceled	
		Execution(Sell)		Y	X				21	20		Canceled	

Scenario-2: Replace (Before Exchange Crossing Session Time)

Time	Message Received	Message Sent	No Sides	Cross ID	OrigCrossID	Buy			Sell			OrdStatus	Comment
						ClOrdID	OrigClOrdID	Qty	ClOrdID	OrigClOrdID	Qty		
1	New Order - Cross		2	X		10		9000	20		9000		
2		Execution(Buy)		X		10						New	
		Execution(Sell)		X					20			New	
3	Cross Cancel/Replace		2	Y	X	11	10	9000	21	20	9000		
4		Execution(Buy)		Y	X	11	10					Pending Replaced	
		Execution(Sell)		Y	X				21	20		Pending Replaced	
5		Execution(Buy)		Y	X	11	10					Replaced	

		Execution(Sell)		Y	X				21	20		Replaced	
6	Cross Cancel/Replace		1	Z	Y	12	11	9000			9000		
7		Execution(Buy)		Z	Y	12	11					Pending Replaced	
8		Execution(Buy)		Z	Y	12	11					Replaced	
9	Exchange Crossing Session Time												
10		Execution(Buy)		Z		12	11					Filled	Cross trade is performed.
		Execution(Sell)		Z					21	20		Filled	

Scenario-3: In case that there are no orders in the book of the market, a New Order - Cross is submitted. (During market hours)

Time	Message Received	Message Sent	No Sides	Cross ID	OrjgCrossID	Buy			Sell			OrdStatus	Comment
						ClOrdID	OrjgClOrdID	Qty	ClOrdID	OrjgClOrdID	Qty		
1						Exchange Session Time							
2	New Order - Cross		2	X		10		9000	20		9000		
3		Execution(Buy)		X		10						New	There is no order in the book.
		Execution(Sell)		X					20			New	
4		Execution(Buy)		X		10						Filled	
		Execution(Sell)		X					20			Filled	

Scenario-4: In case that there are no orders in the book, a New Order - Cross is submitted. (During market hours)

Time	Message Received	Message Sent	No Sides	Cross ID	OrjgCrossID	Buy			Sell			OrdStatus	Comment
						ClOrdID	OrjgClOrdID	Qty	ClOrdID	OrjgClOrdID	Qty		
1						Exchange Session Time							
2	New Order - Cross		2	X		10		9000	20		9000		
3		Execution(Buy)		X		10						Rejected	There was an order in the book.
		Execution(Sell)		X					20			Rejected	

Cross Type 2

Scenario-1: There are no orders in the book

Time	Message Received	Message Sent	No Sides	Cross ID	OrigCrossID	Buy			Sell			OrdStatus	Comment
						ClOrdID	OrigClOrdID	Qty	ClOrdID	OrigClOrdID	Qty		
1	New Order - Cross		2	X		10		9000	20		9000		
2		Execution(Buy)		X		10						New	
		Execution(Sell)		X					20			New	
3		Execution(Buy)		X		10						Filled	
		Execution(Sell)		X					20			Filled	

Scenario-2: There is an order in the book

Time	Message Received	Message Sent	No Sides	Cross ID	OrigCrossID	Buy			Sell			OrdStatus	Comment
						ClOrdID	OrigClOrdID	Qty	ClOrdID	OrigClOrdID	Qty		
1	New Order - Cross		2	X		10		9000	20		9000		
2		Execution(Buy)		X		10						New	
		Execution(Sell)		X					20			New	
3		Execution(Buy)		X		10		5000 (LastShares)				Partial Filled	There was a sell order(Qty=5000) in the book.
4		Execution(Buy)		X		10		4000 (LastShares)				Filled	
		Execution(Sell)		X					20		4000 (LastShares)	Partial Filled	
5		Execution(Sell)		X					20			Canceled	Remaining order is canceled.

Cross Type 3

Time	Message Received	Message Sent	No Sides	Cross ID	OrjgCrossID	Buy			Sell			OrdStatus	Comment
						ClOrdID	OrjgClOrdID	Qty	ClOrdID	OrjgClOrdID	Qty		
1	New Order - Cross		2	X		10		9000	20		9000		
2		Execution(Buy)		X		10						New	
		Execution(Sell)		X					20			New	
3		Execution(Buy)		X		10		5000 (LastShares)				Partial Filled	There is a sell order(Qty=5000) in the book.
4		Execution(Buy)		X		10		4000 (LastShares)				Filled	
		Execution(Sell)		X					20		4000 (LastShares)	Partial Filled	
5	Cross Cancel/Replace		1	Y	X				21	20	8000		
6		Execution(Sell)		Y	X				21	20		Pending Replaced	
7		Execution(Sell)		Y	X				21	20		Replaced	
8	Cross Cancel/Replace		1	Z	Y				22	21	4000		
9		Cancel Reject		Z	Y				22	21			Replace request is rejected
10	Cross Cancel Request		1	W	Y				23	21	8000		
11		Execution(Sell)		W	Y				23	21		Pending Canceled	
12		Execution(Sell)		W	Y				23	21		Canceled	

Cross Type 4

Time	Message Received	Message Sent	No Sides	Cross ID	Orig Cross ID	Buy			Sell			OrdStatus	Comment
						ClOrdID	OrigClOrdID	Qty	ClOrdID	OrigClOrdID	Qty		
1	New Order - Cross		2	X		10		14000	20		9000		There is a sell order(Qty=5000) in the book.
2		Execution(Buy)		X		10						New	
		Execution(Sell)		X					20			New	
3		Execution(Buy)		X		10		5000 (LastShares)				Partial Filled	
4		Execution(Buy)		X		10		9000 (LastShares)				Filled	
		Execution(Sell)		X					20		9000 (LastShares)	Filled	

CATEGORY: MULTILEG ORDERS (SWAPS, OPTION STRATEGIES, ETC)

Background

A multileg security is made up of multiple securities that are traded atomically. Swaps, option strategies, futures spreads, are a few examples of multileg securities. This requirement that all legs be traded in the quantities that they make up the multlileg security is the important distinction between a multileg order and a list order.

Two generalized approaches to trading multileg securities are supported by FIX. The first approach involves a market maintaining multileg securities as separate products for which markets can be created. This "product approach" is often used in electronic trading systems. The second approach is to trade the multileg security as a group of separate securities – as is commonly done today in open outcry markets.

The multileg order can be traded using one of the following trading models using FIX. The first three models are variations on the multileg security as a separate tradeable product. The last models permits trading of multileg securities in environments where the multileg securities are not productized.

Predefined Multileg Security Model (FIX 4.2) (Model 1)

In this model a Security Definition Request for the security is sent to the counterparty that defines the multileg security and the legs. The counterparty accepts the security definition with an acknowledging Security Definition message. The initiating counterparty can then send a New Order – Single message that specifies just the multileg instrument without the legs.

	Counterparty 1 – **Interested in trading a multileg instrument**		**Counterparty 2 or Market**
1	**Sends Security Definition Request that defined Multileg Security**	→	Receives Security Definition Request – determines if multileg security has already been defined. If so – return identification of the multileg security – otherwise create the multileg security and return its identification.
2a	*Create the order*	←	**Reply with Security Definition for multileg security with identification identical to that of the request**
2b	*Create the order*	←	**Reply with Security Definition for multileg security with identification different from that of the request**
2c	Exception Handling for failed Security Definition Request	←	**Reply with Security Definition rejecting the security request**
3	**Send *New Order - Multileg* for security identification provide in Security Definition Request (The Instrument Leg component block is not provided)**	→	***Accepts order for processing***
4a		←	If MultilegReportTypeRequest =0 or =1 or if market rules require reporting by multileg security: **Send *Execution Report* for the overall multileg security (MultilegReportType=2)**

4b		←	If MultliegReportTypeRequest =1 or =2 or if market rules require reporting by multileg security **Send *Execution Reports* for each instrument leg defined previously for the multileg security (MultilegReportType=3)**

Enhanced Predefined Security Model (Model 2)

In the enhanced model – the multileg security is still defined as a product using the Security Definition message. However, the instrument legs are elaborated on the order to provide clearing information per leg, such as LegPositionEffect, LegCoveredOrUncovered, and within <NestedParties> information such as ClearingFirm for the leg, etc.

	Counterparty 1 – **Interested in trading a multileg instrument**		**Counterparty 2 or Market**
1	**Sends Security Definition Request that defined Multileg Security**	→	Receives Security Definition Request – determines if multileg security has already been defined. If so – return identification of the multileg security – otherwise create the multileg security and return its identification.
2a	*Create the order*	←	**Reply with Security Definition for multileg security with identification identical to that of the request**
2b	*Create the order*	←	**Reply with Security Definition for multileg security with identification different from that of the request**
2c	Exception Handling for failed Security Definition Request	←	**Reply with Security Definition rejecting the security request**
3	**Send *New Order - Multileg* for security identification provide in Security Definition Request. Includes Leg Instrument Block**	→	***Accepts order for processing***
4a		←	If MultilegReportTypeRequest =0 or =1 or if market rules require reporting by multileg security: **Send *Execution Report* for the overall multileg security (MultilegReportType=2)**
4b		←	If MultliegReportTypeRequest =1 or =2 or if market rules require reporting by multileg security **Send *Execution Reports* for each instrument leg defined previously for the multileg security (MultilegReportType=3)**

Product Definition Model using New Order - Multileg Message (Model 3)

In this approach the Multileg Security is defined using the New Order - Multileg message. However, the market or counterparty still creates or maintains a product definition for the multileg security upon receipt of the New Order - Multileg.

	Counterparty 1 – **Interested in trading a multileg instrument**		**Counterparty 2 or Market**
1	**Send New Order - Multileg that includes the multileg security definition in the Leg Instrument Block**	→	*Accepts order for processing* A product is defined or identified for the multileg security. If the multileg security is not a valid product in the market – the order is rejected. The order is rejected using an Execution Report – indicating an invalid product was encountered.
2a		←	If MultilegReportTypeRequest =0 or =1 or if market rules require reporting by multileg security: **Send *Execution Report* for the overall multileg security (MultilegReportType=2)**
2b		←	If MultliegReportTypeRequest =1 or =2 or if market rules require reporting by multileg security **Send *Execution Reports* for each instrument leg defined previously for the multileg security (MultilegReportType=3)**

Single Message Model (Model 4)

No product definition is used (Likely will be used by open outcry markets that do not have a product definition service). The message flow is the same as model 3 – the difference being that the counterparty or market receiving the order does not create nor maintain product information for the multileg security – most likely the multileg security is simply distributed to the market.

	Counterparty 1 – **Interested in trading a multileg instrument**		**Counterparty 2 or Market**

1	**Send New Order - Multileg that includes the multileg security definition in the Leg Instrument Block**	→	*Accepts order for processing* The multileg security information is distributed to the market. No product definition takes place. If the multileg security is not a valid multileg strategy in the market – the order is rejected. The order is rejected using an Execution Report – indicating an invalid product was encountered.
2a		←	If MultilegReportTypeRequest =0 or =1 or if market rules require reporting by multileg security: **Send *Execution Report* for the overall multileg security (MultilegReportType=2)**
2b		←	If MultliegReportTypeRequest =1 or =2 or if market rules require reporting by multileg security **Send *Execution Reports* for each instrument leg defined previously for the multileg security (MultilegReportType=3)**

Messages Used for Multileg Trading

Order Entry

Use the New Order - Multileg (MsgType=AB) message to submit a multileg order to a market place.

Execution Reports for Multileg Orders

The Execution Report (MsgType=8) has been modified to report the order status of Multileg Orders.

Modification of a Multileg Order

Use the Multileg Order Cancel Replace Request (a.k.a MultilegOrder Modification Request) (MsgType=AC) to modify a Multileg Order.

Cancellation of a Multileg Order

Multileg orders are canceled using the Order Cancel Request (MsgType = F). The entire multileg order is cancelled by OrderID (tag #37) or ClOrdID (tag# 11). The ability to cancel one leg of a multileg order is not supported in FIX 4.3 and above.

Multileg Pricing Methods

Multileg orders may be submitted using different pricing schemes.

1. Prior to FIX 5.0 SP1 LegPrice (566) was used to specify an anchor price for a leg as part of the definition or creation of a multileg strategy. Price (44) would not be specified if LegPrice is specified.
2. .... [FIXME: need more content describing the use of MultilegPriceMethod field and what relation Price field has to all this also]

New Order - Multileg

The New Order - Multileg is provided to submit orders for securities that are made up of multiple securities, known as legs.

The format for the new order message is as follows:

New Order - Multileg

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = AB
11	ClOrdID	Y	Unique identifier of the order as assigned by institution or by the intermediary with closest association with the investor.
526	SecondaryClOrdID	N	
583	ClOrdLinkID	N	
component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
229	TradeOriginationDate	N	
75	TradeDate	N	
1	Account	N	
660	AcctIDSource	N	
581	AccountType	N	
589	DayBookingInst	N	
590	BookingUnit	N	
591	PreallocMethod	N	
70	AllocID	N	Used to assign an identifier to the block of individual preallocations
Start of Component block, expanded in line < PreAllocMlegGrp >			
78	NoAllocs	N	Number of repeating groups for pre-trade allocation
→	79 AllocAccount	N	Required if NoAllocs > 0. Must be first field in repeating group.
→	661 AllocAcctIDSource	N	
→	736 AllocSettlCurrency	N	
→	467 IndividualAllocID	N	
→	component block <NestedParties3>	N	Insert here the set of "NestedParties3" (firm identification "nested" within additional repeating group) fields defined in "Common Components of Application Messages"
→	80 AllocQty	N	
End of Component block, expanded in line < PreAllocMlegGrp >			

63	SettlType		N	
64	SettlDate		N	Takes precedence over SettlType value and conditionally required/omitted for specific SettlType values.
544	CashMargin		N	
635	ClearingFeeIndicator		N	
21	HandlInst		N	
18	ExecInst		N	Can contain multiple instructions, space delimited. If OrdType=P, exactly one of the following values (ExecInst = L, R, M, P, O, T, or W) must be specified.
110	MinQty		N	
1089	MatchIncrement		N	
1090	MaxPriceLevels		N	
component block <DisplayInstruction>			N	Insert here the set of "ReserveInstruction" fields defined in "common components of application messages"
111	MaxFloor		N	(Deprecated in FIX.5.0)
100	ExDestination		N	
1133	ExDestinationIDSource		N	
Start of Component block, expanded in line < TrdgSesGrp >				
386	NoTradingSessions		N	Specifies the number of repeating TradingSessionIDs
→	336	TradingSessionID	N	Required if NoTradingSessions is > 0.
→	625	TradingSessionSubID	N	
End of Component block, expanded in line < TrdgSesGrp >				
81	ProcessCode		N	Used to identify soft trades at order entry.
54	Side		Y	Additional enumeration that indicates this is an order for a multileg order and that the sides are specified in the Instrument Leg component block.
component block <Instrument>			N	
Start of Component block, expanded in line < UndInstrmtGrp >				
711	NoUnderlyings		N	Number of underlyings
→	component block <UnderlyingInstrument>		N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >				
140	PrevClosePx		N	Useful for verifying security identification
1069	SwapPoints		N	For FX Swaps. Used to express the differential between the far leg's bid/offer and the near leg's bid/offer.
Start of Component block, expanded in line < LegOrdGrp >				
555	NoLegs		Y	Number of legs
→	component block		N	Must be provided if Number of legs > 0

	<InstrumentLeg>				
→	687	LegQty		N	
→	690	LegSwapType		N	
→	component block <LegStipulations>			N	
→	1366	LegAllocID		N	
→	*Start of Component block, expanded in line < LegPreAllocGrp >*				
→	670	NoLegAllocs		N	
→	→	671	LegAllocAccount	N	
→	→	672	LegIndividualAllocID	N	
→	→	component block <NestedParties2>		N	
→	→	673	LegAllocQty	N	
→	→	674	LegAllocAcctIDSource	N	
→	→	1367	LegAllocSettlCurrency	N	
→	*End of Component block, expanded in line < LegPreAllocGrp >*				
→	564	LegPositionEffect		N	Provide if the PositionEffect for the leg is different from that specified for the overall multileg security
→	565	LegCoveredOrUncovered		N	Provide if the CoveredOrUncovered for the leg is different from that specified for the overall multileg security.
→	component block <NestedParties>			N	Insert here the set of "Nested Parties" (firm identification "nested" within additional repeating group) fields defined in "Common Components of Application Messages" Used for NestedPartyRole=Leg Clearing Firm/Account, Leg Account/Account Type
→	654	LegRefID		N	Used to identify a specific leg.
→	587	LegSettlType		N	Refer to values for SettlType (63)
→	588	LegSettlDate		N	Refer to values for SettlDate (64)
→	675	LegSettlCurrency		N	
→	685	LegOrderQty		N	
→	1379	LegVolatility		N	
→	1381	LegDividendYield		N	
→	1383	LegCurrencyRatio		N	
→	1384	LegExecInst		N	
End of Component block, expanded in line < LegOrdGrp >					

114	LocateReqd	N	Required for short sell orders
60	TransactTime	Y	Time this order request was initiated/released by the trader, trading system, or intermediary.
854	QtyType	N	
component block <OrderQtyData>		N	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages" Conditionally required when the multileg order is not for a FX Swap, or any other swap transaction where having OrderQty is irrelevant as the amounts are expressed in the LegQty.
40	OrdType	Y	
1377	MultilegModel	N	
1378	MultilegPriceMethod	N	
423	PriceType	N	
44	Price	N	Required for limit OrdTypes. For F/X orders, should be the "all-in" rate (spot rate adjusted for forward points). Can be used to specify a limit price for a pegged order, previously indicated, etc.
1092	PriceProtectionScope	N	
99	StopPx	N	Required for OrdType = "Stop" or OrdType = "Stop limit".
component block <TriggeringInstruction>		N	Insert here the set of "TriggeringInstruction" fields defined in "common components of application messages"
15	Currency	N	
376	ComplianceID	N	
377	SolicitedFlag	N	
23	IOIID	N	Required for Previously Indicated Orders (OrdType=E)
117	QuoteID	N	Required for Previously Quoted Orders (OrdType=D)
1080	RefOrderID	N	Required for counter-order selection / Hit / Take Orders. (OrdType = Q)
1081	RefOrderIDSource	N	Conditionally required if RefOrderID is specified.
59	TimeInForce	N	Absence of this field indicates Day order
168	EffectiveTime	N	Can specify the time at which the order should be considered valid
432	ExpireDate	N	Conditionally required if TimeInForce = GTD and ExpireTime is not specified.
126	ExpireTime	N	Conditionally required if TimeInForce = GTD and ExpireDate is not specified.
427	GTBookingInst	N	States whether executions are booked out or accumulated on a partially filled GT order

component block <CommissionData>			N	Insert here the set of "CommissionData" fields defined in "Common Components of Application Messages"
528	OrderCapacity		N	
529	OrderRestrictions		N	
1091	PreTradeAnonymity		N	
582	CustOrderCapacity		N	
121	ForexReq		N	Indicates that broker is requested to execute a Forex accommodation trade in conjunction with the security trade.
120	SettlCurrency		N	Required if ForexReq = Y.
775	BookingType		N	Method for booking out this order. Used when notifying a broker that an order to be settled by that broker is to be booked out as an OTC derivative (e.g. CFD or similar). Absence of this field implies regular booking.
58	Text		N	
354	EncodedTextLen		N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText		N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
77	PositionEffect		N	For use in derivatives omnibus accounting
203	CoveredOrUncovered		N	For use with derivatives, such as options
210	MaxShow		N	(Deprecated in FIX.5.0)
component block <PegInstructions>			N	Insert here the set of "PegInstruction" fields defined in "Common Components of Application Messages"
component block <DiscretionInstructions>			N	Insert here the set of "DiscretionInstruction" fields defined in "Common Components of Application Messages"
847	TargetStrategy		N	The target strategy of the order
Start of Component block, expanded in line < StrategyParametersGrp >				
957	NoStrategyParameters		N	Indicates number of strategy parameters
→	958	StrategyParameterName	N	Name of parameter
→	959	StrategyParameterType	N	Datatype of the parameter.
→	960	StrategyParameterValue	N	Value of the parameter
End of Component block, expanded in line < StrategyParametersGrp >				
848	TargetStrategyParameters		N	(Deprecated in FIX.5.0)For further specification of the TargetStrategy
1190	RiskFreeRate		N	

849	ParticipationRate	N	(Deprecated in FIX.5.0)Mandatory for a TargetStrategy=Participate order and specifies the target particpation rate. For other order types optionally specifies a volume limit (i.e. do not be more than this percent of the market volume)
480	CancellationRights	N	For CIV - Optional
481	MoneyLaunderingStatus	N	
513	RegistID	N	Reference to Registration Instructions message for this Order.
494	Designation	N	Supplementary registration information for this Order
563	MultiLegRptTypeReq	N	Indicates the method of execution reporting requested by issuer of the order.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details

Refer to FIXML element NewOrdMleg

Multileg Order Cancel Replace Request (a.k.a MultilegOrder Modification Request)

Used to modify a multileg order previously submitted using the New Order - Multileg message. See Order Cancel Replace Request for details concerning message usage.

The format of the Multileg Order Cancel/Replace Request message is:

Multileg Order Cancel/Replace Request (a.k.a Multileg Order Modification Request)

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = AC
37	OrderID	N	Unique identifier of most recent order as assigned by sell-side (broker, exchange, ECN).
41	OrigClOrdID	N	ClOrdID of the previous order (NOT the initial order of the day) when canceling or replacing an order. Required when referring to orders that were electronically submitted over FIX or otherwise assigned a ClOrdID.
11	ClOrdID	N	Unique identifier of replacement order as assigned by institution or by the intermediary with closest association with the investor.. Note that this identifier will be used in ClOrdID field of the Cancel Reject message if the replacement request is rejected.
526	SecondaryClOrdID	N	
583	ClOrdLinkID	N	
586	OrigOrdModTime	N	
component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
229	TradeOriginationDate	N	
75	TradeDate	N	
1	Account	N	
660	AcctIDSource	N	
581	AccountType	N	
589	DayBookingInst	N	
590	BookingUnit	N	
591	PreallocMethod	N	
70	AllocID	N	Used to assign an identifier to the block of individual preallocations
Start of Component block, expanded in line < PreAllocMlegGrp >			
78	NoAllocs	N	Number of repeating groups for pre-trade allocation
→ 79	AllocAccount	N	Required if NoAllocs > 0. Must be first field in repeating group.

→	661	AllocAcctIDSource	N	
→	736	AllocSettlCurrency	N	
→	467	IndividualAllocID	N	
→	component block <NestedParties3>		N	Insert here the set of "NestedParties3" (firm identification "nested" within additional repeating group) fields defined in "Common Components of Application Messages"
→	80	AllocQty	N	
End of Component block, expanded in line < PreAllocMlegGrp >				
63	SettlType		N	
64	SettlDate		N	Takes precedence over SettlType value and conditionally required/omitted for specific SettlType values.
544	CashMargin		N	
635	ClearingFeeIndicator		N	
21	HandlInst		N	
18	ExecInst		N	Can contain multiple instructions, space delimited. If OrdType=P, exactly one of the following values (ExecInst = L, R, M, P, O, T, or W) must be specified.
110	MinQty		N	
1089	MatchIncrement		N	
1090	MaxPriceLevels		N	
component block <DisplayInstruction>			N	Insert here the set of "DisplayInstruction" fields defined in "common components of application messages"
111	MaxFloor		N	(Deprecated in FIX.5.0)
100	ExDestination		N	
1133	ExDestinationIDSource		N	
Start of Component block, expanded in line < TrdgSesGrp >				
386	NoTradingSessions		N	Specifies the number of repeating TradingSessionIDs
→	336	TradingSessionID	N	Required if NoTradingSessions is > 0.
→	625	TradingSessionSubID	N	
End of Component block, expanded in line < TrdgSesGrp >				
81	ProcessCode		N	Used to identify soft trades at order entry.
54	Side		Y	Additional enumeration that indicates this is an order for a multileg order and that the sides are specified in the Instrument Leg component block.
component block <Instrument>			N	
Start of Component block, expanded in line < UndInstrmtGrp >				
711	NoUnderlyings		N	Number of underlyings

→	component block <UnderlyingInstrument>			N	Must be provided if Number of underlyings > 0
End of Component block, expanded in line < UndInstrmtGrp >					
140	PrevClosePx			N	Useful for verifying security identification
1069	SwapPoints			N	
Start of Component block, expanded in line < LegOrdGrp >					
555	NoLegs			Y	Number of legs
→	component block <InstrumentLeg>			N	Must be provided if Number of legs > 0
→	687	LegQty		N	
→	690	LegSwapType		N	
→	component block <LegStipulations>			N	
→	1366	LegAllocID		N	
→	*Start of Component block, expanded in line < LegPreAllocGrp >*				
→	670	NoLegAllocs		N	
→	→	671	LegAllocAccount	N	
→	→	672	LegIndividualAllocID	N	
→	→	component block <NestedParties2>		N	
→	→	673	LegAllocQty	N	
→	→	674	LegAllocAcctIDSource	N	
→	→	1367	LegAllocSettlCurrency	N	
→	*End of Component block, expanded in line < LegPreAllocGrp >*				
→	564	LegPositionEffect		N	Provide if the PositionEffect for the leg is different from that specified for the overall multileg security
→	565	LegCoveredOrUncovered		N	Provide if the CoveredOrUncovered for the leg is different from that specified for the overall multileg security.
→	component block <NestedParties>			N	Insert here the set of "Nested Parties" (firm identification "nested" within additional repeating group) fields defined in "Common Components of Application Messages" Used for NestedPartyRole=Leg Clearing Firm/Account, Leg Account/Account Type
→	654	LegRefID		N	Used to identify a specific leg.
→	587	LegSettlType		N	Refer to values for SettlType (63)

→	588	LegSettlDate	N	Refer to values for SettlDate (64)
→	675	LegSettlCurrency	N	
→	685	LegOrderQty	N	
→	1379	LegVolatility	N	
→	1381	LegDividendYield	N	
→	1383	LegCurrencyRatio	N	
→	1384	LegExecInst	N	
End of Component block, expanded in line < LegOrdGrp >				
114	LocateReqd		N	Required for short sell orders
60	TransactTime		Y	Time this order request was initiated/released by the trader, trading system, or intermediary.
854	QtyType		N	
component block <OrderQtyData>			Y	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages"
40	OrdType		Y	
1377	MultilegModel		N	
1378	MultilegPriceMethod		N	
423	PriceType		N	
44	Price		N	Required for limit OrdTypes. For F/X orders, should be the "all-in" rate (spot rate adjusted for forward points). Can be used to specify a limit price for a pegged order, previously indicated, etc.
1092	PriceProtectionScope		N	
99	StopPx		N	Required for OrdType = "Stop" or OrdType = "Stop limit".
component block <TriggeringInstruction>			N	Insert here the set of "TriggeringInstruction" fields defined in "common components of application messages"
15	Currency		N	
376	ComplianceID		N	
377	SolicitedFlag		N	
23	IOIID		N	Required for Previously Indicated Orders (OrdType=E)
117	QuoteID		N	Required for Previously Quoted Orders (OrdType=D)
59	TimeInForce		N	Absence of this field indicates Day order
168	EffectiveTime		N	Can specify the time at which the order should be considered valid
432	ExpireDate		N	Conditionally required if TimeInForce = GTD and ExpireTime is not specified.
126	ExpireTime		N	Conditionally required if TimeInForce = GTD and

				ExpireDate is not specified.
427	GTBookingInst		N	States whether executions are booked out or accumulated on a partially filled GT order
component block <CommissionData>			N	Insert here the set of "CommissionData" fields defined in "Common Components of Application Messages"
528	OrderCapacity		N	
529	OrderRestrictions		N	
1091	PreTradeAnonymity		N	
582	CustOrderCapacity		N	
121	ForexReq		N	Indicates that broker is requested to execute a Forex accommodation trade in conjunction with the security trade.
120	SettlCurrency		N	Required if ForexReq = Y.
775	BookingType		N	Method for booking out this order. Used when notifying a broker that an order to be settled by that broker is to be booked out as an OTC derivative (e.g. CFD or similar). Absence of this field implies regular booking.
58	Text		N	
354	EncodedTextLen		N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText		N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
77	PositionEffect		N	For use in derivatives omnibus accounting
203	CoveredOrUncovered		N	For use with derivatives, such as options
210	MaxShow		N	(Deprecated in FIX.5.0)
component block <PegInstructions>			N	Insert here the set of "PegInstruction" fields defined in "Common Components of Application Messages"
component block <DiscretionInstructions>			N	Insert here the set of "DiscretionInstruction" fields defined in "Common Components of Application Messages"
847	TargetStrategy		N	The target strategy of the order
Start of Component block, expanded in line < StrategyParametersGrp >				
957	NoStrategyParameters		N	Indicates number of strategy parameters
→	958	StrategyParameterName	N	Name of parameter
→	959	StrategyParameterType	N	Datatype of the parameter.
→	960	StrategyParameterValue	N	Value of the parameter
End of Component block, expanded in line < StrategyParametersGrp >				

848	TargetStrategyParameters	N	(Deprecated in FIX.5.0)For further specification of the TargetStrategy
1190	RiskFreeRate	N	
849	ParticipationRate	N	(Deprecated in FIX.5.0)Mandatory for a TargetStrategy=Participate order and specifies the target particpation rate. For other order types optionally specifies a volume limit (i.e. do not be more than this percent of the market volume)
480	CancellationRights	N	For CIV - Optional
481	MoneyLaunderingStatus	N	
513	RegistID	N	Reference to Registration Instructions message for this Order.
494	Designation	N	Supplementary registration information for this Order
563	MultiLegRptTypeReq	N	Indicates the method of execution reporting requested by issuer of the order.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element MlegOrdCxlRplc

CATEGORY: LIST/PROGRAM/BASKET TRADING

The List/Program/Basket Trading message set is used for the trading of lists/programs/baskets of orders.

A subset of the List/Program/Basket Trading message set, New Order List and List Status, is also used to support contingent orders. Contingent orders include "one-cancels-other", "one-triggers-other", and "one-updates-other". See ***"Contingent Orders"*** for usage guidelines.

Bid Request

The BidRequest Message can be used in one of two ways depending on which market conventions are being followed.

In the “Non disclosed” convention (e.g. US/European model) the BidRequest message can be used to request a bid based on the sector, country, index and liquidity information contained within the message itself. In the “Non disclosed” convention the entry repeating group is used to define liquidity of the program. ***See " Program/Basket/List Trading" for an example.***

In the “Disclosed” convention (e.g. Japanese model) the BidRequest message can be used to request bids based on the ListOrderDetail messages sent in advance of BidRequest message. In the “Disclosed” convention the list repeating group is used to define which ListOrderDetail messages a bid is being sort for and the directions of the required bids.

The pair of fields SideValue1 and SideValue2 are used to show the monetary total value in either direction (buy or sell) of the transaction without revealing whether it is the buy-side institution’s intention to buy or sell.

The two repeating groups, NoEntries and NoBidComponents are mutually exclusive and a function of which bidding model is being used. If the “Non Disclosure” method is being used the portfolio of stocks being traded is described by a number of “bid descriptors” entries. If the “Disclosure” Method is being used the portfolio is fully disclosed, except for side, by a number of “list” entries enumerating the lists that list the stocks to be traded.

A BidRequest message with BidRequestTransType cancel may be used to indicate to sell side firms that they no longer need to store details of the BidRequest as they have either lost the bid or the List has been canceled.

The format for the Bid Request message is as follows:

Bid Request

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = k (lowercase)
390	BidID	N	Required to relate the bid response
391	ClientBidID	Y	
374	BidRequestTransType	Y	Identifies the Bid Request message transaction type
392	ListName	N	
393	TotNoRelatedSym	Y	
394	BidType	Y	e.g. "Non Disclosed", "Disclosed", No Bidding Process
395	NumTickets	N	Total number of tickets/allocations assuming fully executed
15	Currency	N	Used to represent the currency of monetary amounts.
396	SideValue1	N	Expressed in Currency

397	SideValue2		N	Expressed in Currency
Start of Component block, expanded in line < BidDescReqGrp >				
398	NoBidDescriptors		N	Used if BidType="Non Disclosed"
→	399	BidDescriptorType	N	Required if NoBidDescriptors > 0. Must be first field in repeating group.
→	400	BidDescriptor	N	
→	401	SideValueInd	N	Refers to the SideValue1 or SideValue2. These are used as opposed to Buy or Sell so that the basket can be quoted either way as Buy or Sell.
→	404	LiquidityValue	N	Value between LiquidityPctLow and LiquidityPctHigh in Currency
→	441	LiquidityNumSecurities	N	Number of Securites between LiquidityPctLow and LiquidityPctHigh in Currency
→	402	LiquidityPctLow	N	Liquidity indicator or lower limit if LiquidityNumSecurities > 1
→	403	LiquidityPctHigh	N	Upper liquidity indicator if LiquidityNumSecurities > 1
→	405	EFPTrackingError	N	Eg Used in EFP (Exchange For Physical) trades 12%
→	406	FairValue	N	Used in EFP trades
→	407	OutsideIndexPct	N	Used in EFP trades
→	408	ValueOfFutures	N	Used in EFP trades
End of Component block, expanded in line < BidDescReqGrp >				
Start of Component block, expanded in line < BidCompReqGrp >				
420	NoBidComponents		N	Used if BidType="Disclosed"
→	66	ListID	N	Required if NoBidComponents > 0. Must be first field in repeating group.
→	54	Side	N	When used in request for a "Disclosed" bid indicates that bid is required on assumption that SideValue1 is Buy or Sell. SideValue2 can be derived by inference.
→	336	TradingSessionID	N	Indicates off-exchange type activities for Detail.
→	625	TradingSessionSubID	N	
→	430	NetGrossInd	N	Indicates Net or Gross for selling Detail.
→	63	SettlType	N	
→	64	SettlDate	N	Takes precedence over SettlType value and conditionally required/omitted for specific SettlType values.
→	1	Account	N	
→	660	AcctIDSource	N	
End of Component block, expanded in line < BidCompReqGrp >				
409	LiquidityIndType		N	
410	WtAverageLiquidity		N	Overall weighted average liquidity expressed as a % of

			average daily volume
411	ExchangeForPhysical	N	
412	OutMainCntryUIndex	N	% value of stocks outside main country in Currency
413	CrossPercent	N	% of program that crosses in Currency
414	ProgRptReqs	N	
415	ProgPeriodInterval	N	Time in minutes between each ListStatus report sent by SellSide. Zero means don't send status.
416	IncTaxInd	N	Net/Gross
121	ForexReq	N	Is foreign exchange required
417	NumBidders	N	Indicates the total number of bidders on the list
75	TradeDate	N	
418	BidTradeType	Y	
419	BasisPxType	Y	
443	StrikeTime	N	Used when BasisPxType = "C"
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details

Refer to FIXML element BidReq

Bid Response

The Bid Response message can be used in one of two ways depending on which market conventions are being followed.

In the "Non disclosed" convention the Bid Response message can be used to supply a bid based on the sector, country, index and liquidity information contained within the corresponding bid request message. ***See "Program/Basket/List Trading" for an example.***

In the "Disclosed" convention the Bid Response message can be used to supply bids based on the List Order Detail messages sent in advance of the corresponding Bid Request message.

The format for the Bid Response message is as follows:

Bid Response

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = l (lowercase L)
390	BidID	N	
391	ClientBidID	N	
Start of Component block, expanded in line < BidCompRspGrp >			
420	NoBidComponents	Y	Number of bid repeating groups
→	component block <CommissionData>	Y	Insert here the set of "CommissionData" fields defined in "Common Components of Application Messages" First element of price. Required if NoBidComponents > 0.
→	66 ListID	N	
→	421 Country	N	ISO Country Code
→	54 Side	N	When used in response to a "Disclosed" request indicates whether SideValue1 is Buy or Sell. SideValue2 can be derived by inference.
→	44 Price	N	Second element of price
→	423 PriceType	N	
→	406 FairValue	N	The difference between the value of a future and the value of the underlying equities after allowing for the discounted cash flows associated with the underlying stocks (E.g. Dividends etc).
→	430 NetGrossInd	N	Net/Gross
→	63 SettlType	N	
→	64 SettlDate	N	Takes precedence over SettlType value and conditionally required/omitted for specific SettlType values.
→	336 TradingSessionID	N	
→	625 TradingSessionSubID	N	
→	58 Text	N	
→	354 EncodedTextLen	N	Must be set if EncodedText field is specified and must

				immediately precede it.
→	355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
End of Component block, expanded in line < BidCompRspGrp >				
StandardTrailer			Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element BidRsp

New Order - List

The NewOrderList Message can be used in one of two ways depending on which market conventions are being followed.

In the “Non disclosed” convention the New Order - List message is sent after the bidding process has been completed, by telephone or electronically. The New Order - List message enumerates the stocks, quantities, direction for the trade and may contain pre-allocation information.

This message may also be used as the first message for the transmission of a program trade where the bidding process has been done by means other than FIX. In this scenario the messages may either be used as a staging process, in which case the broker will start execution once either a ListExecute is received or for immediate execution, in which case the orders will be executed on receipt.

In the “Disclosed” convention the New Order - List message is sent before the bidding process is started, by telephone or electronically. The New Order - List message enumerates the stocks and quantities from the bidding process, and may contain pre-allocation information. The direction of the trade is disclosed after the bidding process is completed.

Where multiple waves of a program trade are submitted by an institution or retail intermediaries, as a series of separate lists, to a broker ClOrdLinkID may be used to link the orders together.

See "Program/Basket/List Trading" for examples.

The New Order – List message type may also be used by institutions or retail intermediaries wishing to electronically submit multiple Collective Investment Vehicle orders to a broker or fund manager for execution.

See VOLUME 7 - "PRODUCT: COLLECTIVE INVESTMENT VEHICLES"

The format for the New Order - List message is as follows:

New Order - List

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = E
66	ListID	Y	Must be unique, by customer, for the day
390	BidID	N	Should refer to an earlier program if bidding took place.
391	ClientBidID	N	
414	ProgRptReqs	N	
394	BidType	Y	e.g. Non Disclosed Model, Disclosed Model, No Bidding Process
415	ProgPeriodInterval	N	
480	CancellationRights	N	For CIV - Optional
481	MoneyLaunderingStatus	N	
513	RegistID	N	Reference to Registration Instructions message applicable to all Orders in this List.
433	ListExecInstType	N	Controls when execution should begin For CIV Orders indicates order of execution..
69	ListExecInst	N	Free-form text.
1385	ContingencyType	N	Used for contingency orders.

352	EncodedListExecInstLen		N	Must be set if EncodedListExecInst field is specified and must immediately precede it.
353	EncodedListExecInst		N	Encoded (non-ASCII characters) representation of the ListExecInst field in the encoded format specified via the MessageEncoding field.
765	AllowableOneSidednessPct		N	The maximum percentage that execution of one side of a program trade can exceed execution of the other.
766	AllowableOneSidednessValue		N	The maximum amount that execution of one side of a program trade can exceed execution of the other.
767	AllowableOneSidednessCurr		N	The currency that AllowableOneSidedness is expressed in if AllowableOneSidednessValue is used.
68	TotNoOrders		Y	Used to support fragmentation. Sum of NoOrders across all messages with the same ListID.
893	LastFragment		N	Indicates whether this is the last fragment in a sequence of message fragments. Only required where message has been fragmented.
component block <RootParties>			N	Insert here the set of "Root Parties" fields defined in "common components of application messages" Used for acting parties that applies to the whole message, not individual orders.
Start of Component block, expanded in line < ListOrdGrp >				
73	NoOrders		Y	Number of orders in this message (number of repeating groups to follow)
→	11	ClOrdID	Y	Must be the first field in the repeating group.
→	526	SecondaryClOrdID	N	
→	67	ListSeqNo	Y	Order number within the list
→	583	ClOrdLinkID	N	
→	160	SettlInstMode	N	
→	component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"
→	229	TradeOriginationDate	N	
→	75	TradeDate	N	
→	1	Account	N	
→	660	AcctIDSource	N	
→	581	AccountType	N	
→	589	DayBookingInst	N	
→	590	BookingUnit	N	
→	70	AllocID	N	Use to assign an ID to the block of individual preallocations
→	591	PreallocMethod	N	

→	*Start of Component block, expanded in line < PreAllocGrp >*				
→	78	NoAllocs		N	Number of repeating groups for pre-trade allocation
→	→	79	AllocAccount	N	Required if NoAllocs > 0. Must be first field in repeating group.
→	→	661	AllocAcctIDSource	N	
→	→	736	AllocSettlCurrency	N	
→	→	467	IndividualAllocID	N	
→	→	component block <NestedParties>		N	Insert here the set of "Nested Parties" (firm identification "nested" within additional repeating group) fields defined in "Common Components of Application Messages" Used for NestedPartyRole=Clearing Firm
→	→	80	AllocQty	N	
→	*End of Component block, expanded in line < PreAllocGrp >*				
→	63	SettlType		N	
→	64	SettlDate		N	Takes precedence over SettlType value and conditionally required/omitted for specific SettlType values.
→	544	CashMargin		N	
→	635	ClearingFeeIndicator		N	
→	21	HandlInst		N	
→	18	ExecInst		N	Can contain multiple instructions, space delimited. If OrdType=P, exactly one of the following values (ExecInst = L, R, M, P, O, T, or W) must be specified.
→	110	MinQty		N	
→	1089	MatchIncrement		N	
→	1090	MaxPriceLevels		N	
→	component block <DisplayInstruction>			N	Insert here the set of "DisplayInstruction" fields defined in "common components of application messages"
→	111	MaxFloor		N	
→	100	ExDestination		N	
→	1133	ExDestinationIDSource		N	
→	*Start of Component block, expanded in line < TrdgSesGrp >*				
→	386	NoTradingSessions		N	Specifies the number of repeating TradingSessionIDs
→	→	336	TradingSessionID	N	Required if NoTradingSessions is > 0.
→	→	625	TradingSessionSubID	N	

→	*End of Component block, expanded in line < TrdgSesGrp >*			
→	81	ProcessCode	N	
→	component block <Instrument>		Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
→	*Start of Component block, expanded in line < UndInstrmtGrp >*			
→	711	NoUnderlyings	N	Number of underlyings
→	→	component block <UnderlyingInstrument>	N	Must be provided if Number of underlyings > 0
→	*End of Component block, expanded in line < UndInstrmtGrp >*			
→	140	PrevClosePx	N	Useful for verifying security identification
→	54	Side	Y	Note: to indicate the side of SideValue1 or SideValue2, specify Side=Undisclosed and SideValueInd=either the SideValue1 or SideValue2 indicator.
→	401	SideValueInd	N	Refers to the SideValue1 or SideValue2. These are used as opposed to Buy or Sell so that the basket can be quoted either way as Buy or Sell.
→	114	LocateReqd	N	Required for short sell orders
→	60	TransactTime	N	
→	component block <Stipulations>		N	Insert here the set of "Stipulations" (repeating group of Fixed Income stipulations) fields defined in "Common Components of Application Messages"
→	854	QtyType	N	
→	component block <OrderQtyData>		Y	Insert here the set of "OrderQtyData" fields defined in "Common Components of Application Messages"
→	40	OrdType	N	
→	423	PriceType	N	
→	44	Price	N	
→	1092	PriceProtectionScope	N	
→	99	StopPx	N	
→	component block <TriggeringInstruction>		N	Insert here the set of "TriggeringInstruction" fields defined in "common components of application messages"
→	component block <SpreadOrBenchmarkCurveData>		N	Insert here the set of "SpreadOrBenchmarkCurveData" (Fixed Income spread or benchmark curve) fields defined in "Common Components of Application Messages"
→	component block <YieldData>		N	Insert here the set of "YieldData" (yield-related) fields defined in "Common Components of Application Messages"
→	15	Currency	N	

→	376	ComplianceID	N	
→	377	SolicitedFlag	N	
→	23	IOIID	N	Required for Previously Indicated Orders (OrdType=E)
→	117	QuoteID	N	Required for Previously Quoted Orders (OrdType=D)
→	1080	RefOrderID	N	Required for counter-order selection / Hit / Take Orders (OrdType = Q)
→	1081	RefOrderIDSource	N	Conditionally required if RefOrderID is specified.
→	59	TimeInForce	N	
→	168	EffectiveTime	N	
→	432	ExpireDate	N	Conditionally required if TimeInForce = GTD and ExpireTime is not specified.
→	126	ExpireTime	N	Conditionally required if TimeInForce = GTD and ExpireDate is not specified.
→	427	GTBookingInst	N	States whether executions are booked out or accumulated on a partially filled GT order
→	component block <CommissionData>		N	Insert here the set of "CommissionData" fields defined in "Common Components of Application Messages"
→	528	OrderCapacity	N	
→	529	OrderRestrictions	N	
→	1091	PreTradeAnonymity	N	
→	582	CustOrderCapacity	N	
→	121	ForexReq	N	
→	120	SettlCurrency	N	
→	775	BookingType	N	Method for booking out this order. Used when notifying a broker that an order to be settled by that broker is to be booked out as an OTC derivative (e.g. CFD or similar). Absence of this field implies regular booking.
→	58	Text	N	
→	354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
→	355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
→	193	SettlDate2	N	Can be used with OrdType = "Forex - Swap" to specify the "value date" for the future portion of a F/X swap.
→	192	OrderQty2	N	Can be used with OrdType = "Forex - Swap" to specify the order quantity for the future portion of a F/X swap.
→	640	Price2	N	Can be used with OrdType = "Forex - Swap" to specify the price for the future portion of a F/X swap which is also a limit order. For F/X orders, should be the "all-in" rate (spot rate adjusted for forward points).

→	77	PositionEffect		N	
→	203	CoveredOrUncovered		N	
→	210	MaxShow		N	
→	component block <PegInstructions>			N	Insert here the set of "PegInstruction" fields defined in "Common Components of Application Messages"
→	component block <DiscretionInstructions>			N	Insert here the set of "DiscretionInstruction" fields defined in "Common Components of Application Messages"
→	847	TargetStrategy		N	The target strategy of the order
→	*Start of Component block, expanded in line < StrategyParametersGrp >*				
→	957	NoStrategyParameters		N	Indicates number of strategy parameters
→	→	958	StrategyParameterName	N	Name of parameter
→	→	959	StrategyParameterType	N	Datatype of the parameter.
→	→	960	StrategyParameterValue	N	Value of the parameter
→	*End of Component block, expanded in line < StrategyParametersGrp >*				
→	848	TargetStrategyParameters		N	For further specification of the TargetStrategy
→	849	ParticipationRate		N	Mandatory for a TargetStrategy=Participate order and specifies the target particpation rate. For other order types optionally specifies a volume limit (i.e. do not be more than this percent of the market volume)
→	494	Designation		N	Supplementary registration information for this Order within the List
End of Component block, expanded in line < ListOrdGrp >					
StandardTrailer				Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details

Refer to FIXML element NewOrdList

List Strike Price

The strike price message is used to exchange strike price information for principal trades. It can also be used to exchange reference prices for agency trades.

The format for the List Strike Price message is as follows:

List Strike Price

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = m (lowercase)
66	ListID	Y	
422	TotNoStrikes	Y	Used to support fragmentation. Sum of NoStrikes across all messages with the same ListID.
893	LastFragment	N	Indicates whether this is the last fragment in a sequence of message fragments. Only required where message has been fragmented.
Start of Component block, expanded in line < InstrmtStrkPxGrp >			
428	NoStrikes	Y	Number of strike price entries
→	component block <Instrument>	Y	Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages" Required if NoStrikes > 0. Must be first field in repeating group.
→	*Start of Component block, expanded in line < UndInstrmtGrp >*		
→	711 NoUnderlyings	N	Number of underlyings
→	→ component block <UnderlyingInstrument>	N	Must be provided if Number of underlyings > 0
→	*End of Component block, expanded in line < UndInstrmtGrp >*		
→	140 PrevClosePx	N	Useful for verifying security identification
→	11 ClOrdID	N	Can use client order identifier or the symbol and side to uniquely identify the stock in the list.
→	526 SecondaryClOrdID	N	
→	54 Side	N	
→	44 Price	N	
→	15 Currency	N	
→	58 Text	N	
→	354 EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
→	355 EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.

End of Component block, expanded in line < InstrmtStrkPxGrp >		
StandardTrailer	Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element ListStrkPx

List Status

The list status message is issued as the response to a List Status Request message sent in an unsolicited fashion by the sell-side. It indicates the current state of the orders within the list as they exist at the broker's site. This message may also be used to respond to the List Cancel Request.

Orders within the list are statused at the summary level. Individual executions are not reported, rather, the current state of the order is reported.

The message contains repeating fields for each. The relative position of the repeating fields is important in this message, i.e. each instance of ClOrdID, CumQty, LeavesQty, CxlQty and AvgPx must be in the order shown below.

Description of ListOrderStatus field values:

- "InBiddingProcess": indicates that a list has been received and is being evalutated for pricing. It is envisaged that this status will only be used with the "Disclosed" List Order Trading model.
- "ReceivedForExecution": indicates that a list has been received and the sell side is awaiting the instruction to start working the trade. It is envisaged that this status will be used under both models.
- "Executing": indicates that a list has been received and the sell side is working it.
- "Canceling": indicates that a List Cancel Message has been received and the sell side is in the process of pulling any orders that were being worked. The status of individual order can be found out from the detail repeating group.
- "AllDone": indicates that a list has been executed as far as possible for the day. This would also apply if a list has been previously cancelled. The status of individual order can be determined from the detail repeating group.
- "Alert": used whenever any of the individual orders have a status that requires something to be done. For instance, an alert would be used when a buy-side firm has submitted a list that has individual stock reject that have not been addressed.
- "Rejected" used when a response cannot be generated. For example when the ListID is not recognised. The text field should include an explanation of why the Request s been rejected.

The list status message format is as follows:

List Status

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = N
66	ListID	Y	
429	ListStatusType	Y	
82	NoRpts	Y	Total number of messages required to status complete list.
431	ListOrderStatus	Y	
1385	ContingencyType	N	
1386	ListRejectReason	N	
83	RptSeq	Y	Sequence number of this report message.

444	ListStatusText		N	
445	EncodedListStatusTextLen		N	Must be set if EncodedListStatusText field is specified and must immediately precede it.
446	EncodedListStatusText		N	Encoded (non-ASCII characters) representation of the ListStatusText field in the encoded format specified via the MessageEncoding field.
60	TransactTime		N	
68	TotNoOrders		Y	Used to support fragmentation. Sum of NoOrders across all messages with the same ListID.
893	LastFragment		N	Indicates whether this is the last fragment in a sequence of message fragments. Only required where message has been fragmented.
Start of Component block, expanded in line < OrdListStatGrp >				
73	NoOrders		Y	Number of orders statused in this message, i.e. number of repeating groups to follow.
→	11	ClOrdID	N	Required when referring to orders that where electronically submitted over FIX or otherwise assigned a ClOrdID.
→	37	OrderID	N	
→	526	SecondaryClOrdID	N	
→	14	CumQty	Y	
→	39	OrdStatus	Y	
→	636	WorkingIndicator	N	For optional use with OrdStatus = 0 (New)
→	151	LeavesQty	Y	Quantity open for further execution. LeavesQty = OrderQty - CumQty.
→	84	CxlQty	Y	
→	6	AvgPx	Y	
→	103	OrdRejReason	N	Used if the order is rejected
→	58	Text	N	
→	354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
→	355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
End of Component block, expanded in line < OrdListStatGrp >				
StandardTrailer			Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details

Refer to ListStat

List Execute

The List Execute message type is used by institutions to instruct the broker to begin execution of a previously submitted list. This message may or may not be used, as it may be mirroring a phone conversation.

The format for the list execute message is as follows:

List Execute

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = L
66	ListID	Y	Must be unique, by customer, for the day
391	ClientBidID	N	Used with BidType=Disclosed to provide the sell side the ability to determine the direction of the trade to execute.
390	BidID	N	
60	TransactTime	Y	Time this order request was initiated/released by the trader or trading system.
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element ListExct

List Cancel Request

The List Cancel Request message type is used by institutions wishing to cancel previously submitted lists either before or during execution.

After the list has been staged with the broker, it can be canceled via the submission of the List Cancel message. If the list has not yet been submitted for execution, the List Cancel message will instruct the broker not to execute it, if the list is being executed, the List Cancel message should trigger the broker's system to generate cancel requests for the remaining quantities of each order within the list. Individual orders within the list can be canceled via the Order Cancel Request message.

The List Status message type is used by the recipient of the List Cancel Request to communicate the status of the List Cancel Request.

The format for the list - cancel request message is as follows:

List Cancel Request

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = K
66	ListID	Y	
component block <Parties>		N	Insert here the set of "Parties" (firm identification) fields defined in "common components of application messages"
60	TransactTime	Y	Time this order request was initiated/released by the trader or trading system.
229	TradeOriginationDate	N	
75	TradeDate	N	
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
<!ListCxlReq

List Status Request

The list status request message type is used by institutions to instruct the broker to generate status messages for a list.

The format for the list - status request message is as follows:

List Status Request

Tag	*FieldName*	*Req'd*	*Comments*
StandardHeader		Y	MsgType = M
66	ListID	Y	
58	Text	N	
354	EncodedTextLen	N	Must be set if EncodedText field is specified and must immediately precede it.
355	EncodedText	N	Encoded (non-ASCII characters) representation of the Text field in the encoded format specified via the MessageEncoding field.
StandardTrailer		Y	

FIXML Definition for this message – see http://www.fixprotocol.org for details
Refer to FIXML element ListStatReq

Fragmentation for List Order Messages

The messages used in program trading support fragmentation for the same reason and in the same way as some other FIX messages (e.g. Mass Quote). If there are too many entries within a repeating group to fit into one physical message, then the entries can be continued in another message by repeating all of the top level information and then specifying the number of entries in the continued message. A "Total Entries" field is provided to specify the total number of entries in a repeating group which is split over multiple messages. This permits, but does not require, a receiving application to react in a stateful manner where it can determine if it has received all entries in a repeating group before carrying out some action. However, the overall approach to fragmentation is to permit each message to be processed in a stateless manner as it is received. Each message should contain enough information to have the entries applied to a system without requiring the next message if fragmentation has occurred. Also, a continued message should not require any information from the previous message. The messages that support fragmentation and the repeating groups supporting it are listed in the table below.

Message	**"Total Entries" field**	**Repeating group that may be fragmented**
New Order - List	TotNoOrders	Orders repeating group following the NoOrders field in the message definition table
List Strike Price	TotNoStrikes	Strike price repeating group following the NoStrikes field in the message definition table
List Status	TotNoOrders	Status per order repeating group following the NoOrders field in the message definition table

Maximum message size for fragmentation purposes can be determined by using the optional MaxMessageSize field in the Logon message or by mutual agreement between counterparties.

Note: The TotNoOrders field has been added to the List Status message to support fragmentation in same way as other FIX messages. The NoRpts and RptSeq fields are preserved for backwards compatibility with previous versions of FIX which supported a stateful form of fragmentation.

Program/Basket/List Trading

Overview

A set of messages allow for the automation of program trading. While it is hoped that the message set is comprehensive enough, to automate the complete cycle, it is expected that not all messages will be used in all transactions. Although the message set may appear to be quite complex at first glance, most of the complexity arises from developing one message set that can be used to support two different business models for list trading. The two models, the "Disclosed" and "Non Disclosed" models, are described in the next two sections. The "Disclosed" model is commonly used in Japan while the "Non Disclosed" model is commonly used in Europe and America.

"Non Disclosed" model (e.g. US/European)

The buy-side details to the sell-side information about the sector, country and potential market impact of the stocks to be bought or sold. Using this information the sell-side firms bid for the trade. If successful the buy-side firm gives the sell-side firm a detailed list of the stocks to be traded and the sell-side firm executes the trades.

The important point in the "Non Disclosed" model is that the stocks in the list are not disclosed until a particular sell-side firm has won the portfolio.

"Disclosed" model (e.g. Japanese)

The buy-side details the exact stocks and sizes to be traded and the sell-side firm offers the buy-side firm a two-way price, to buy or to sell the indicated stocks. The buy-side firm then tells the successful sell-side firm to buy or sell on its behalf.

The important point in the "Disclosed" model is that all sell-side firms see all of the stocks and quantities in the portfolio during the bidding phase regardless of whether or not they win the business.

The New Order - List message can be used, with the side omitted as part of the bidding process, as is the practice in "Disclosed" model or once the bidding has been completed to exchange the list of stocks that make up the program to be traded. Pre-trade allocation is handled via a repeating group within the repeating order block.

Order modification and cancelation of a portfolio is a major change to the agreement between the buy-side and sell-side firms and as such this change should be conducted by telephone. If an automated route for dealing with amendment/cancelation is required then the existing messages can be used – List Cancel, Order Cancel/Replace Request.

The New Order - List message is based on the single order message and message flows for canceling a single stock line within a program trade should be those used to support order cancelation in the single order model (e.g. Order Cancel/Replace Request, etc). The ListID in those systems should be used to assist in identifying the order as part of a list trade. Similarly, the Order Status Request message can be used to request the status of a single order in a portfolio trade.

The List Strike Price Message details the prices that a principal trade is being executed at. In some transactions this appears to be generated by the sell side and checked by the buy-side, in others the reverse is true, and in other cases this information is not passed until the final execution reports

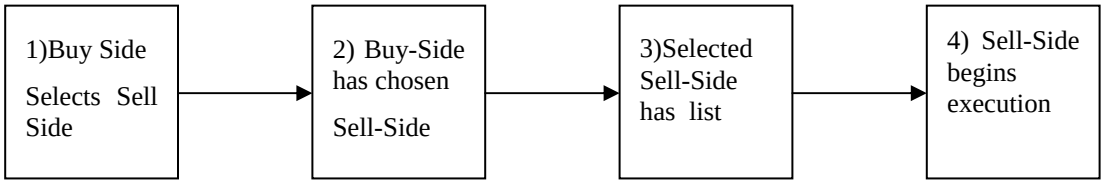
Pre-trade allocation is much more common place in the program trading community than it is in block trading. For the purposes of pre-trade allocation, a repeating group to specify AllocAccount and AllocQty has been added to the order message. It is assumed that participants will use either the FIX allocation messages and message flows for post trade allocation or their existing allocation systems. (e.g. in the event of a pre-allocation basket trade).

At any stage in the processing of a list message the buy-side may request the status of the list from the sell-side using the List Status Request message. The sell-side responds with a List Status Response message. The sell-side can also send the List Status Response message in an unsolicited fashion according to the requirements passed in the bidding phase or in the List message. The List Status Response message provides summary detail about each of the orders in the List. The sell-side should acknowledge any list request from the buy-side with a List Status Response message providing the current state.

Once the portfolio has been executed by the sell-side and a List Status Response message has been sent to the buy-side indicating "DONE" for each of the orders in the List, the list can be allocated. If pre-allocation information was provided with the original orders and the orders were fully executed then the allocation information is already known to the sell-side. If the pre-trade allocations are no longer appropriate post trade allocation may be preformed either using FIX Allocation messages or existing allocation systems.

Message Flow Diagrams

Overview of logical stages



The diagram above shows the logical stages involved in the execution of a program trade.

Transition	Description
1->2	This transition can occur in the following ways : • Buy-side has preferred list which means the business will be directed to a specific Sell-Side • Buy-Side provides details of the program to a number of sell-sides. This can be achieved using a mixture of telephone, fax, modem links, and FIX Bid messages.
2->3	Details of the program are transmitted to the chosen Sell-side using telephone, fax, modem links, or FIX New Order - List message.
3->4	This transition can occur in the following ways : • Buy-Side and Sell-Side communicate by telephone to confirm content of the program and the Buy-Side instructs the Sell-Side to begin execution. • Buy-Side sends a List Execute FIX message to instruct Sell-Side to begin execution
4	Once the list is being executed the FIX List status messages may be used to notify/request status of the list

Order Details sent via FIX (Bidding Outside of FIX)

Buy Side Institution				Sell Side Broker Dealer
		New Order - List Message (1 broker)	→	
	←	2) List Order Status (ACK)		

"Disclosed" Bid and Program Trade

Buy Side Institution				Sell Side Broker Dealer
		1) New Order - List Message (N brokers)	→	
	←	2) List Order Status (ACK) (1 per broker)		
		3) Bid Request Message (N brokers)	→	
	←	4) Bid Response (1 per broker)		
		5) Cancel bid sent to (N – 1) brokers	→	

"Non Disclosed" Bid and Program Trade

Buy Side Institution				Sell Side Broker Dealer
		1) Bid Request Message (N brokers)	→	
	←	2) Bid Response (1 per broker)		
	Buy-side selects one Sell-side and sends order detail for execution			
		3a) Bid Request Cancel (N – 1 brokers) 3b) New Order - List Message (1 sell-side)	→	
	←	4) List Status Response (ACK)		

Message Flows once a buy-side has chosen a single sell-side and transmitted New Order - List messages

The following message flows can occur in any order relative to each other and some may occur many times.

Optional notification to begin execution (may occur zero or one times)

Buy Side Institution		1) List Execute message (1 sell-side)	→	**Sell Side Broker Dealer**
	←	2) List Order Status (Ack) Response		

Optional transfer of Principal Portfolio Trade prices from buy-side to sell-side (may occur zero or one times)

Buy Side Institution		1) List Strike Price message	→	**Sell Side Broker Dealer**
	←	2) List Status message		

Optional transfer of Principal Portfolio Trade prices from sell-side to buy-side (may occur zero or one times)

Buy Side Institution	←	1) List Strike Price message		**Sell Side Broker Dealer**
		2) List Status message	→	

Optional Execution Report status update (may occur zero or N times)

Buy Side Institution	←	1) Execution Reports (if requested)	**Sell Side Broker Dealer**

Optional List Status Request (may occur zero or N times)

Buy Side Institution		1) List Status Request message	→	**Sell Side Broker Dealer**
	←	2) List Status message		

Optional Sell-Side unsolicited Status update (may occur zero or N times)

Buy Side Institution	←	2) List Status message	**Sell Side Broker Dealer**

Scenario 1 Bidding performed by Telephone and List provided via FIX

Message	Description	Purpose
	New Order - List Message from B/S to S/S	Details the list of stocks that an institution wishes to trade. Normally side is omitted and an indicator is set to show that this message is part of a bid
	List Status Response (Acknowledgement)	List status response indicates that the sell-side has received the New Order - List message. The status of each order in the list

		should indicate a status of *bid* or *rejected*. The former if the stock is recognised and the latter if the stock is not recognised.
may be omitted	List Execute Message	Details the specific bid that has been accepted. The specific bid indicates the direction of the list to be executed.
Required if previous provided	List Status Response	Details the status of each order in the list. The status should be *executing* for each order.
	Status updates may optionally follow	

Scenario 2 Fully Disclosed Program Trade – with bidding stage through FIX

Message	Description	Purpose
	New Order - List Message from B/S to S/S	Details the list of stocks that an institution wishes to trade. Normally side is omitted and an indicator is set to show that this message is part of a bid
	List Status Response (Acknowledgement)	List status response indicates that the sell-side has received the New Order - List message. The status of each order in the list should indicate a status of *bid* or *rejected*. The former if the stock is recognised and the latter if the stock is not recognised.
	Bid Request Message from B/S to S/S	Details the types of bids required, eg Side, Execution Type etc
may be omitted	Bid Response Message	Details the bid response for a program
may be omitted	List Execute Message	Details the specific bid that has been accepted. The specific bid indicates the direction of the list to be executed.
Required if previous provided	List Status Response	Details the status of each order in the list. The status should be *executing* for each order.
	Status updates may optionally follow	

Scenario 3 Non-Disclosed Program Trade – with bidding stage through FIX

Message	Description	Purpose
may be omitted done by phone	Bid Request from B/S to S/S	Details the liquidity information about the stocks that an institution wishes to trade. It does not identify the stocks in the program.
may be omitted	Bid Response Message from S/S to B/S	Details the bid response for a program

done by phone		
	List Message Detail Message from B/S to S/S	Details the list of stocks that an institution wishes to trade including the stock, quantity, and direction for each order.
Required if previous provided	List Status Response	Details the status of each order in the list. The status should be *awaiting execution*, *executing* or *rejected* for each order.
may be omitted done by phone	List Execute Message from B/S to S/S	Details the bid for a program
required if previous provided	List Status Response	Details the status of each order in the list. The status should be *executing* for each order.

Illustration of liquidity indicator fields usage

Normally details, by country and by sector, as number at <5%, no in 5-10%, no in 10-30% and number at > 30% eg 1@ 70%, 1 @ 600% For example

Country	<5%	5 – 10%	10 - 30%	> 30%
DEM	1 Sec $1000000	4 Sec $2000000	7 Sec $1500000	1 Sec @60%, $3000000 1 Sec @300% $8000000
ESP	4 Sec $3000000	5 Sec $3000000	3 Sec $3500000	
UK	3 Sec $4500000	6 Sec $3600000	2 Sec $5000000	1 Sec @450% $9000000

Sector	<5%	5 – 10%	10 - 30%	> 30%
Industrials	2 Sec $1500000	5	4	1 Sec @300% $8000000
Pharmaceuticals	4 Sec	3	3	1 Sec @450% $9000000
Hotels	2	7	5	1 Sec @60%, $3000000

Illustration of liquidity indicator fields usage

The liquidity indicator fields are used to describe the shape of a basket trade in terms of the liquidity and classification of the stocks contained within the list. Thus a list that may be described by the following to tables.

List liquidity information by country.

% columns refer to percentage of average daily volume.

Country	<5%	5 – 10%	10 - 30%	> 30%
DEM	1 Security Value $1000000	4 Security Value $2000000	7 Security Value $1500000	1 Security Value $3M @ 60% 1 Security Value $8M @300%
ESP	4 Security Value $3000000	5 Security Value $3000000	3 Security Value $3500000	
UK	3 Security Value $4500000	6 Security Value $3600000	2 Security Value $5000000	1 Security Value $9M @ 450%

List liquidity information by Security Sector.

% columns refer to percentage of average daily volume.

Sector	<5%	5 – 10%	10 - 30%	> 30%
Industrials	2 Security Value $1500000	5 Security Value $2600000	4 Security Value $3000000	1 Security Value $8M @300%
Pharmaceutical	4 Security Value $3000000	3 Security Value $3000000	3 Security Value $1500000	1 Security Value $9M @450%
Hotels	2 Security Value $4000000	7 Security Value $3000000	5 Security Value $2500000	1 Security Value $3M @60%

Would be represented by the following BidRequest Message.

BidRequest Message (Non Disclosed bid, basket of securites, not an exchange for physical trade)

Client Bid ID	Bid Request Trans Type	Total Num Securities	Bid Type	Side Value 1	Repeating fields						
					Bid Descr-iptor Type	Bid Descr-Iptor	Side Value Ind	Liquidity Value	Liquidity Num Secu-rities	Liquidity Pct Low	Liquidity Pct High
1001	N	38	1	37100000	2	DEM	1	1000000	1	0.00	0.05
					2	DEM	1	2000000	4	0.05	0.10
					2	DEM	1	1500000	7	0.10	0.30
					2	DEM	1	3000000	1	0.60	*- NP
					2	DEM	1	8000000	1	3.00	*- NP
					2	ESP	1	3000000	4	0.00	0.05
					2	ESP	1	3000000	5	0.05	0.10
					2	ESP	1	3500000	3	0.10	0.30
					2	UK	1	4500000	3	0.00	0.05
					2	UK	1	3600000	6	0.05	0.10
					2	UK	1	2000000	2	0.10	0.30
					2	UK	1	9000000	1	4.50	*- NP
					1	Ind	1	1500000	2	0.00	0.05
					1	Ind	1	2600000	5	0.05	0.10
					1	Ind	1	3000000	4	0.10	0.30
					1	Ind	1	8000000	1	3.00	*- NP
					1	Pharm	1	3000000	4	0.00	0.05
					1	Pharm	1	3000000	3	0.05	0.10
					1	Pharm	1	1500000	3	0.10	0.30
					1	Parm	1	9000000	1	4.50	*- NP
					1	Hotels	1	4000000	2	0.00	0.05
					1	Hotels	1	3000000	7	0.05	0.10
					1	Hotels	1	2500000	5	0.10	0.30
					1	Hotels	1	3000000	1	0.60	*- NP

Notes

- *- NP field not present in repeating group as entry is describing a single stock at a specific liquidity
- Where the BidDescriptorType set to 1 the entry in the BidDescriptor field is free text, the sector names Pharmaceuticals and Industrials have been shorten to make every thing fit.

Contingent Orders

Overview

Contingency orders, such as a One-Cancels-Other (OCO), consist of two or more orders whose behavior is contingent on each other. Contingent orders are also known as “alternative”, "either-or" or “linked” orders. A contingent order is an order whose execution is dependent upon the execution of another order as part of a stipulated execution condition.

It should be noted that the orders in a contingent order list are treated as individual orders and executed as such. The contingent order as a whole is not an "order" that can be executed in and of itself.

The support of contingent orders is currently limited to "vanilla" types of contingent orders. More complex uses, for example an "one-trigger-other" triggering an "one-cancel-other" consisting of a “stop loss” order and a “take profit” order, are outside the scope of the standard.

General usage notes:

- Contingent orders are sent using the New Order List message. List Orders are identified with the ListID, this field is also echoed back in Execution Reports and allows the user to keep the set of contingent orders together.
- Although a submitted New Order List cannot be updated, the individual orders of a contingency can be updated by using the Order Cancel Replace Request message, referencing the ClOrdID of the order to be updated.
- Certain properties (e.g. pre-allocations, time in force) of a contingent order are often the same for the set of orders. The recommendation is that the Initiator (sender of the contingent order) defines such properties for the first order of the list and the receiver applies them to the rest of the orders. Bilateral agreement determines what fields are applicable for this treatment.
- The Initiator can cancel the contingent orders through a List Cancel Request identifying the contingent order by specifying the ListID.
- The Initiator can cancel individual orders within the contingent order through the normal Order Cancel Request - subject to bilateral agreement - referencing the ClOrdID of the order.
- Contingencies can have various restrictions subject to bilateral agreement, for example:
  - Limited number of orders allowed in a contingent order
  - Disallow complex conditions as for example pegging, triggers, reserve size and quantity conditions
  - Allowed for continuous trading sessions only (e.g. not allowed in auctions)
  - Orders cannot be individually added or removed (you have to cancel the entire contingent order and enter a new one)
- Multi-leg orders cannot be included as part of a contingency.

Note that the Respondent (shown as "Receiver" in the workflow diagrams below) may be either a broker-dealer or a marketplace. Broker-dealers would generally work or manage each of the individual orders within the contingent order in their trade order management systems. In a marketplace (e.g. exchange), the individual orders will be entered into the order book and remain until they are filled (partial or full), cancelled or expired.

Types of Contigent Orders

One-Cancels-Other (OCO)

An OCO order is an order whose execution results in the immediate cancellation of another order linked to it. Cancellation of the Contingent Order happens on a best efforts basis. In an OCO case, both orders are live in the marketplace at the same time. The execution of either order triggers an attempt to cancel the unexecuted order.

Partial executions will also trigger an attempt to cancel the other order. An OCO contingency is entered as a List Order specifying ContingencyType (1385) = 1 - One Cancels the Other.

Vanilla workflow:

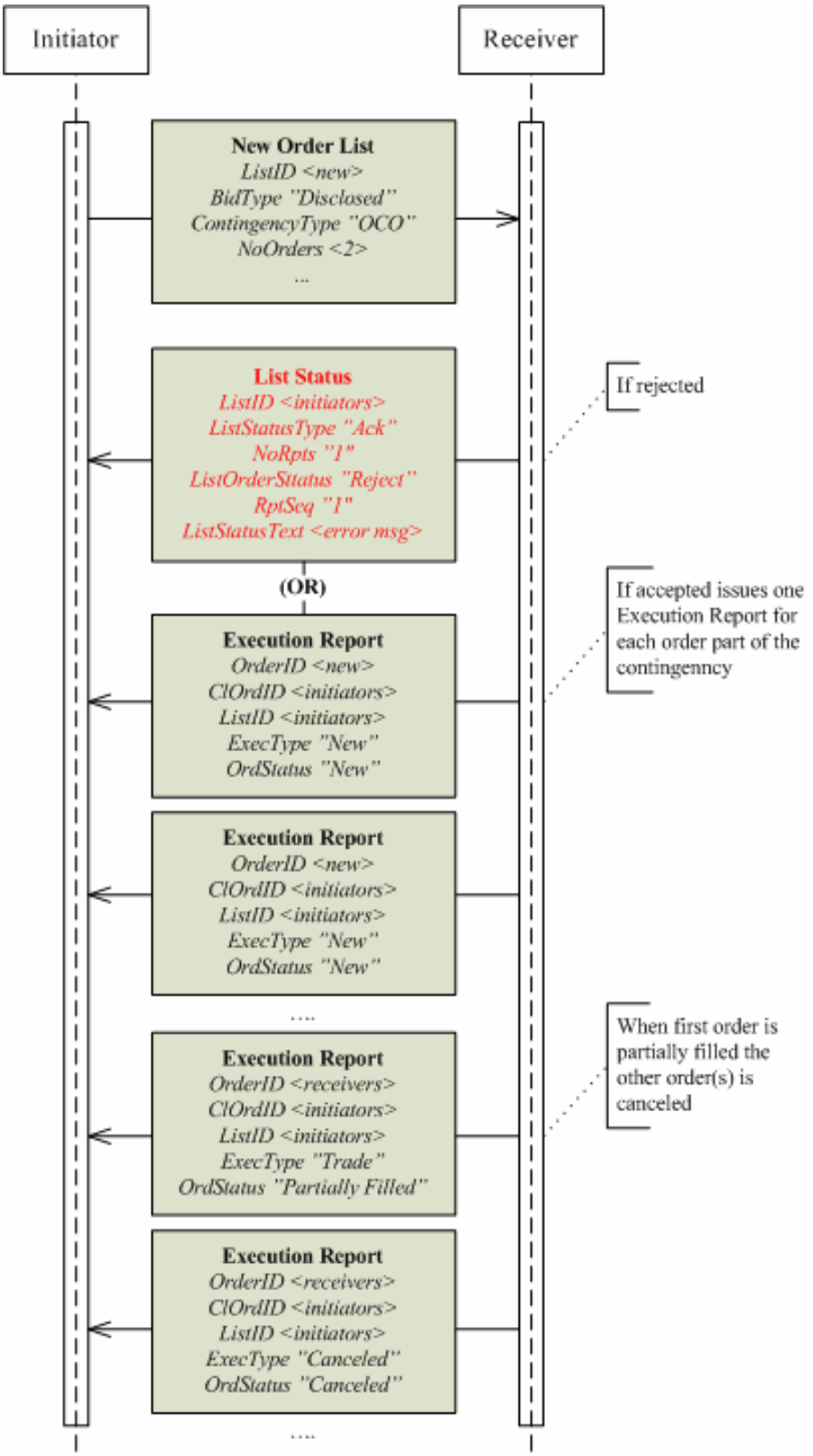
1. The Initiator enters two (or more) orders in a single transaction and states that there should be an OCO contingency between them.
2. If the Respondent immediately (partially) filled one of the orders, the other order(s) are canceled.
3. If none of the orders are immediately (partially) filled, the orders are "live" until one is (partially) filled, the contingent order is cancelled or expires.
4. If one of the orders is (partially) filled, the other(s) are canceled.

Usage notes:

- All orders have the same Time in Force. If not, bilateral agreement must define the behavior of differing Time in Force usage. For example:
  - When one order expires, the others are canceled, or
  - When all but one order has expired, the entire contingent order is expired
- More than two orders can be part of a contingency, where the first order (partial) fill leads to the cancellation of all others.

Figure 6 below illustrates a general OCO workflow.

Figure 6: Vanilla OCO workflow



One-Triggers-Other (OTO)

A one-trigger-other contingent order involves two (or more) orders - a primary order and a secondary order. The primary order is a live marketplace order. The secondary order, usually held in a separately, is not a market order. If the primary order executes in full, the secondary order is released to the marketplace and becomes live. An OTO order can be made up of stock orders, option orders, or a combination of both. An OTO contingency is entered as a List Order specifying ContingencyType (1385) = 2 - One Triggers the Other.

Vanilla workflow:

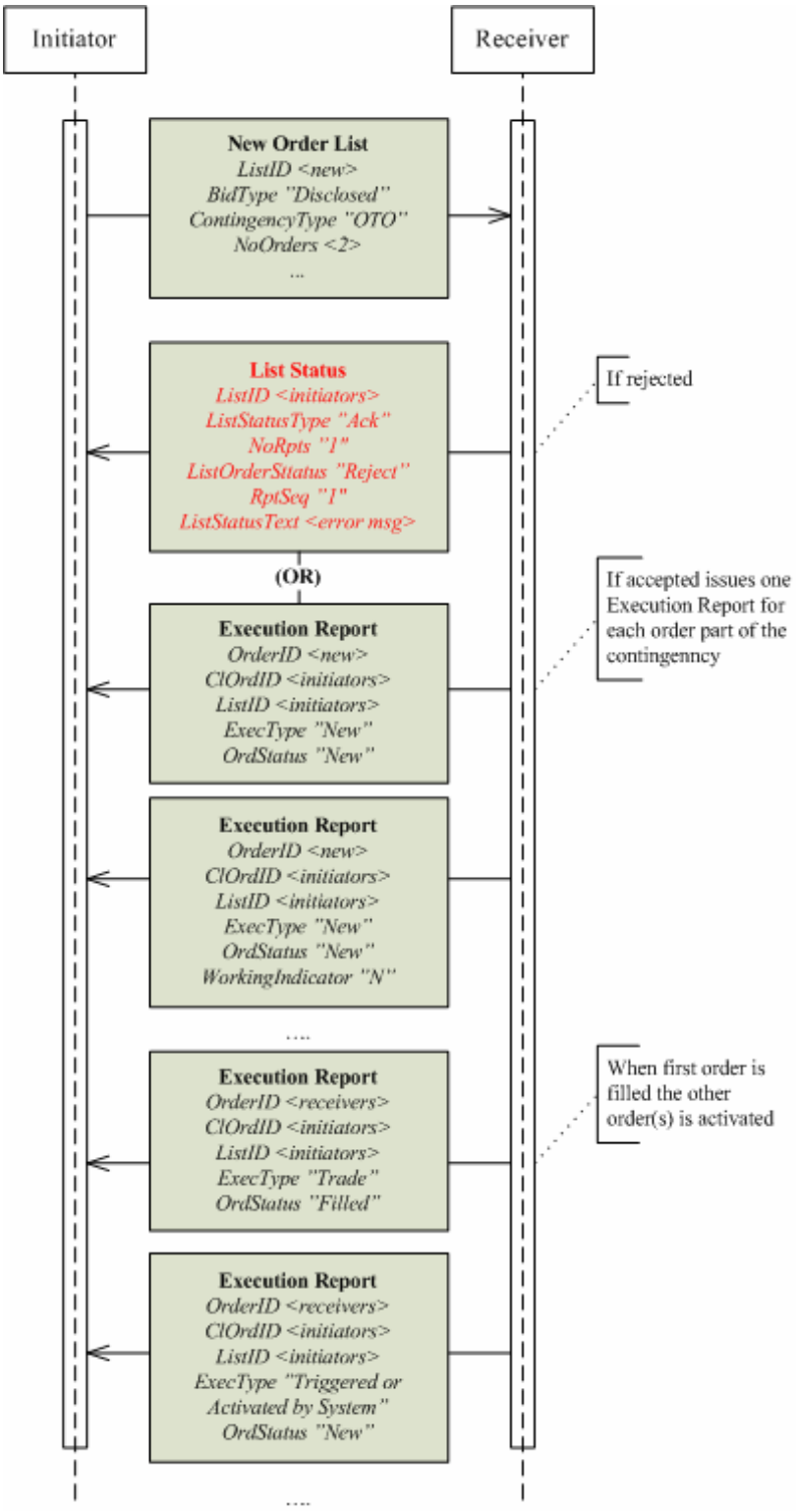
1. The Initiator enters two orders in a single transaction and states that there should be an OTO contingency between them. One of the orders is marked as the primary order and the other as a secondary order.
2. If the Respondent immediately (fully) filled the primary order, the secondary order is activated for execution.
3. If the primary order is not immediately (fully) filled, it remains "live" until it is (fully) filled or expires
4. If the primary order is (fully) filled, the secondary order is activated for execution.

Usage notes:

- Both orders have the same Time in Force. If not, bilateral agreement must define the behavior of differing Time in Force usage. For example:
  - When the first order expires, the other is canceled, or
  - When the first order expires, the second order is activated, or
  - If the second order expires, the contingency is expired but the first order is retained (i.e. still active)
- More than two orders can be part of the contingency, where the first order (full) fill leads to the triggering of all others.

Figure 7 below illustrates a general OTO workflow.

Figure 7: One-triggers-other workflow



One-Updates-Other (OUO)

A one-updates-other order is a contingent order whose execution results in the immediate reduction of quantity in another order linked to it in the order list. The quantity reduction happens on a best effort basis. In an OUO order both orders are live at the same time. The execution of any of the orders in the order list triggers an attempt to reduce the remaining quantity of the other order(s), partial executions included. There are two variants for OUO orders:

- **Proportional Quantity Reduction**. Instead of canceling the other Contingent Order(s), their quantity is reduced in proportion to the filled quantity. An OUO with proportional quantity reductions is entered as a List Order specifying ContingencyType (1395) = 4 - One Updates the Other - Proportional Quantity Reduction. Example:
  - Order A is for 100; Order B is for 50.
  - When order B is partially filled for 25 (50 %), order A is restated to a leaves quantity of 50 (50 %).
- **Absolute Quantity Reduction**. Instead of canceling the other Contingent Order(s), their quantity is reduced with the same partially filled value. An OUO with absolute quantity reductions is entered as a List Order specifying ContingencyType (1385) = 3 - One Updates the Other - Absolute Quantity Reduction.
  - Order A is for 100; Order B is for 50.
  - When order B is partially filled for 25, order A is restated to a leaves quantity of 75.

Vanilla workflow:

1. The Initiator enters two or more orders and states that there should be an OUO contingency between them.
2. If the Respondent (partial or fully) fills one of the orders, the other order(s) are updated to reduce the quantity according to the instructions given.
3. If none of the orders are (partially or fully) filled, the orders remain active until they are (partially or fully) filled or expires
4. If one of the orders is (partially) filled, the other(s) are updated reducing the quantity according to the instructions given.

Usage notes:

- Special attention is needed regarding the order that gets its quantity reduced with the fill for the other order. This attention focuses the rules for the quantity fields of the Execution Report. As the second order is **not** (partially) filled, Last- (32), Cum- (14) and LeavesQty (151) fields cannot be updated. The option remaining is therefore to update the OrderQty (38) field. If the quantity is exhausted through the update, the order is as a consequence canceled.
- All orders have the same Time in Force. If not, bilateral agreement must define the behavior, e.g.:
  - When one order expires, the others are canceled, or
  - When all but one order has expired, the contingency is dropped

Figure 8 below illustrates a general OUO workflow.

Figure 8: One-Updates-Other workflow