FINANCIAL INFORMATION EXCHANGE PROTOCOL

Version 2.7(a)

January 17, 1995

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).

PREFACE

In mid-1992 Fidelity Investments and Salomon Brothers began work on a project to link the two firms' trading systems for the purpose of automating the exchange equity-related indication and execution messages. Fidelity determined that it could help its traders become more effective by automating portions of their broker communication and thus reduce the time spent on the phone placing routine orders and recording fills. Salomon's motivation was the desire to leverage its investment in technology into increased service for its clients. The initial system link was brought up in early 1993 and has been in production each trading day since.

At the onset of the effort it was recognized that the success of the linkage would be determined by the two firms' ability to develop a workable set of messages and protocols to be used for the information exchange. The firms decided that the effort would be most successful if other buy- and sell-side representatives were involved in the specification discussions. Throughout 1993 a group of large financial firms met regularly to refine the specification and to share their experiences in its implementation. The group felt it necessary to limit participation in the effort until the concept had been proven in production and the rate of requests for modification to the protocol had slowed to the point which indicated it had stabilized. By early 1994 the majority of the committee members had implemented the protocol and felt that the stabilization requirement for general release had been satisfied.

The Financial Information Exchange Protocol is the result of these efforts. It is offered to the financial community as a starting point towards the establishment of a standard for broker/institution communication.

Feedback on this protocol, including comments on existing content and suggestions for additional message types, is requested. Feedback should directed to me at Salomon. My phone number in New Jersey is (201) 896-7789.

Christopher Morstatt

Vice President

Salomon Brothers Inc

July, 1994

Contents

FINANCIAL INFORMATION EXCHANGE PROTOCOL

INTRODUCTION

The Financial Information Exchange (FIX) Protocol is a message standard developed to facilitate the electronic exchange of information related to Equity and Fixed Income transactions. It is intended for use between brokers and institutions wishing to automate communications.

The message protocol, as defined, will support the following electronic conversations:

- Equity order submissions, cancellations and replacements
- Equity execution reporting
- Equity order statusing
- Equity trade allocation
- Indication of interest communication
- Completed trade advertisements
- Directed email and news messaging

The FIX protocol is defined at two levels; session and application. The session level is concerned with the delivery of data while the application level defines business related data content. This protocol is independent of the telecommunications protocol (X.25, asynch, internet, etc.) and medium chosen for electronic data delivery.

SESSION PROTOCOL

Each connection between an institution and broker will be defined as a session. The Financial Information Exchange session consists of three parts:

- Connection, authentication and initialization
- Message exchange
- Logout

The following section describes each of these parts.

1. **Connection, authentication and initialization -**

   All connections between a broker and institution are initiated from the institution's site. After establishing message level communication the sequence to connect to the broker is:

   - Authenticate as a valid user by submitting the *Logon* message which contains the login name and password as agreed upon between the parties.
   - After validation, the broker's system will start the initialization process which will:
     - forward all queued outgoing messages awaiting transmission,
     - issue a test request message, and,
     - if the response to the test request is other than 1 (one), issue a resend request for any untransmitted incoming messages.

2. Message exchange -

After completion of the initialization process, normal message exchange begins. The formats for all valid messages are detailed in the sections 'Administrative Messages' and 'Application Messages'.

General Message Format and Content:

The general format of a message is a standard message header followed by message body data and a standard trailer.

The Financial Information Exchange session protocol is based on an optimistic model; normal delivery of data is assumed (i.e. no acknowledgment of individual messages) with errors in delivery identified by message sequence number gaps. Each message (except certain of the administrative messages) is identified by a unique sequence number. It is the receiving application's responsibility to monitor incoming sequence numbers to identify gaps for response with resend request messages.

Each message is defined in a <tag>=<value> format. Except where noted, fields within a message can be defined in any sequence (i.e. relative position of a field within a record is inconsequential except where explicitly defined otherwise).

Fields within a record are to be separated by a delimiter character; the non-printing, ASCII "SOH" (#001), is used for this purpose. Examples cited in this document will use the character "|" to represent the delimiter.

There shall be no imbedded delimiter characters within fields except for type *data*.

Data Types:

Data types are mapped to ASCII strings as follows:

- int: Sequence of digits without commas or decimals and optional sign character (ASCII characters "-" and "0" - "9"). The sign character utilizes one byte (i.e. positive int is "**99999**" while negative int is "**-99999**").

  Examples: 723 in field 21 would be mapped int as |**21=723**|.

  -723 in field 12 would be mapped int as |**12=-723**|

- float: Sequence of digits with optional decimal point and sign character (ASCII characters "-", "0" - "9" and "."); the absence of the decimal point within the string will be interpreted as the float representation of an integer value.

- char: Alpha-numeric free format strings, can include any character or punctuation except the delimiter. All char fields are case sensitive (i.e. **morstatt ≠ Morstatt**).

- data: Raw data with no format or content restrictions. Data fields are always preceeded by a length field. *Caution: may contain the delimiter (SOH) character.*

Sequence Numbers:

All messages are identified by unique sequence number. Sequence numbers are re-initialized each day starting at 1 (one). Each connection will establish an independent incoming and outgoing sequence series; participants will maintain a sequence series to assign to outgoing messages and a series to monitor for sequence gaps on incoming messages.

Heartbeats:

When message traffic is light the sending application will generate *heartbeat* messages at regular time intervals. The heartbeat will contain the next sequence number to be transmitted. The heartbeat is useful for identifying when the last of a string of messages are lost and for monitoring the status of the communication link.

Possible Duplicates:

In certain circumstances the sending application may be unsure if a message was successfully transmitted to its destination. In such cases a *poss dupe* message will be generated. The poss dupe will be a resend (with the same sequence number) of the data in question with the PossDupFlag included and set to "Y" in the header. It is the receiving application's responsibility to handle the poss dupe message (i.e. treat as a new message or discard as appropriate). All messages created as the result of a resend request will contain the PossDupFlag field, messages lacking the PossDupFlag field are original transmissions. The PossDup flag is used only to identify the resend of messages with a previously recorded sequence number.

Possible Resends:

Application level messages which appear to the end-user to be in an ambiguous state may be resent with the PossResend flag set. An example of when this is useful include an order which remains unacknowledged for an inordinate length of time and which the end-user is may suspect has never been sent. The receiving application must recognize this flag and interrogate internal fields (order number, etc.) to determine if this order has been previously received.

Data Integrity:

Integrity of message data content can be verified in two ways; verification of record length and a simple checksum of characters. The record length is indicated in the BodyLength field and can be verified by counting the number of characters in the message following the BodyLength field up to, and including, the delimiter immediately preceeding the CheckSum tag (“10=”). An additional integrity check is to sum the ASCII value of each character up to, but not including, the CheckSum tag field and comparing the least significant eight bits of the calculated value to the CheckSum value (see Appendix C for a complete description).

Required Fields:

Each message within the protocol is comprised *required*, *non-required* and *conditionally required (fields which are required only when other fields are present)* fields. Systems should be written to operate when only the required and conditionally required fields are present.

Encryption:

The exchange of sensitive data across public carrier networks may make it advisable to employ data encryption techniques to mask the application messages.

The choice of encryption method will be determined by mutual agreement between the broker and institution.

Except where explicitly stated any field within a message can be encrpyted and included in the SecureData field.

Imbedded in the protocol are fields which enable the implementation of a public key signature and encryption methodology, straight DES encryption and clear text. The choice of encryption methodology will be defined in the LOGON message.

3. Logout -

Normal termination of the message exchange session will be initiated by the customer via the submission of the LOGOUT message. Termination by other means will be considered an abnormal condition.

Message header

Each message, administrative or application, is preceded by a standard header. The header is used to identify the message type, length, destination, sequence number, origination point and time.

The message header format is as follows:

Message Header

Tag	*Field Name*	*Req'd*	*Comments*
8	BeginString	Y	FIX.2.7 (*Always unencrypted*)
9	BodyLength	Y	(*Always unencrypted*)
35	MsgType	Y	(*Always unencrypted*)
49	SenderCompID	Y	(*Always unencrypted*)
50	SenderSubID	N	(*Always unencrypted*)
56	TargetCompID	Y	(*Always unencrypted*)
57	TargetSubID	N	"ADMIN" reserved for administrative messages not intended for a specific user. (*Always unencrypted*)
90	SecureDataLen	N	Required to identify length of encrypted section of message. (*Always unencrypted*)
91	SecureData	N	Required when message body is encrypted. Always immediately follows SecureDataLen field.
34	MsgSeqNum	Y	*(Can be embedded within encrypted data section.)*
43	PossDupFlag	N	Always required for retransmissions, whether prompted by the sending system or as the result of a resend request. *(Can be embedded within encrypted data section.)*
97	PossResendFlag	N	Required when message may be duplicate of another message sent under a different sequence number. *(Can be embedded within encrypted data section.)*
52	SendingTime	Y	*(Can be embedded within encrypted data section.)*
51	SendingDate	Y	*(Can be embedded within encrypted data section.)*

Message trailer

Each message, administrative or application, is terminated by a standard trailer. The trailer is used to segregate messages and contains the three digit character representation of the Checksum value.

The message trailer format is as follows:

Message Trailer

Tag	*Field Name*	*Req'd*	*Comments*
93	SignatureLength	N	Required when trailer contains signature.
89	Signature	N	
10	CheckSum	Y	(*Always unencrypted*)

ADMINISTRATIVE MESSAGES

The administrative class of messages are intended to address the utility needs of the protocol. The following section describes the use of each message and provides the message layout.

Administrative messages will be generated from both the broker and institution side of the connection.

Heartbeat -

When message traffic is light, the sending application will generate heartbeat messages at regular time intervals. The heartbeat will consist of the standard message header and trailer and will contain the next sequence number to be transmitted. The heartbeat is useful for identifying when the last of a string of messages are lost and for monitoring the status of the communication link.

The heartbeat is generated only when no regular message has been transmitted within the heartbeat interval. The heartbeat interval timer will be reset each time a regular message is transmitted. The interval value will be individually determined for each connection.

The standard message header is utilized for heartbeats with following provisions:

- The MsgType field is set to **"0"**
- The sequence number field contains the next message sequence number to be transmitted
- The sequence number is **NOT** incremented for heartbeats
- The SenderSubID, TargerSubID, SecureDataLen, SecureData, PossResendFlag and PossDup fields are not required.

The heartbeat format is as follows:

Heartbeat

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = 0 MsgSeqNo contains expected next sequence number; i.e. last sequence number + 1
	Standard Trailer	Y	

Logon Message -

The logon message is utilized authenticate a user attempting to establish a connection to a remote system. The logon message must be the first message sent by the connecting application.

The logon format is as follows:

Logon

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = A
98	EncryptMethod	Y	(*Always unencrypted*)
	Standard Trailer	Y	

Test Request -

The test request message is utilized to force a heartbeat from the opposing application. The test request message is useful for checking sequence number or verifying communication line status. The receiving application will respond to the test request message with a heartbeat. Sequence numbers are not incremented for test request messages.

The test request format is as follows:

Test Request

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = 1
	Standard Trailer	Y	

Resend Request -

The resend request is introduced by the receiving application to initiate the retransmission of messages. This function would be utilized if a sequence number gap is detected, if the receiving application lost a message, or as a function of the initialization process. Sequence numbers are not incremented for resend request messages.

The resend request can be used to request a single message, a range of messages or all messages subsequent to a particular message.

- To request a single message: BeginSeqNo = EndSeqNo
- To request a range of messages: BeginSeqNo = first message of range, EndSeqNo = last message of range
- To request all messages subsequent to a particular message: BeginSeqNo = first message of range, EndSeqNo = 99999

The resend request format is as follows:

Resend Request

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = 2
7	BeginSeqNo	Y	
16	EndSeqNo	Y	
	Standard Trailer	Y	

Reject -

The reject message will be issued when a message is received which cannot be parsed or contains invalid information (i.e. fails one of the validation tests). Where possible the cause of the failure will be described in the Text field.

The reject format is as follows:

Reject

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = 3
45	RefSeqNo	N	May not be available if message is unparsable
58	Text	N	Where possible, message to explain reason for rejection
	Standard Trailer	Y	

Sequence Reset -

The sequence reset message is used to reestablish the incoming sequence number on the opposing side. In the event of an application failure it may be necessary to resync sequence numbers on the sending and receiving sides.

The sending application will initiate the sequence reset. The message is used to reset the value of the last sequence number sent (which on the receiving side will reset the value of the next expected sequence number).

It is assumed that the purpose of the sequence reset message is to recover from an out-of- sequence condition, therefore, the MsgSeqNum in the header should be ignored (i.e. the receipt of a sequence reset message with an out of sequence MsgSeqNum should not generate resend requests).

The sequence reset can only increase the sequence number; if a sequence reset is received attempting to decrease the next expected sequence number the message should be rejected.

The sequence reset format is as follows:

Sequence Reset

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = 4
36	NewSeqNum	Y	
	Standard Trailer	Y	

Logout -

The logout message is to used to terminate the session. Disconnection without the sending of a logout message should be interpreted as an abnormal condition.

The logout format is as follows:

Logout

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = 5
58	Text	N	
	Standard Trailer	Y	

APPLICATION MESSAGES

The exchange of business infomation is accomplished through the passing of application messages. The application message is composed of the standard header followed by the message body and trailer.

Descriptions and formats of the specific messages follow.

Advertisements -

Advertisement messages are used to announce completed transactions. The advertisement message can be transmitted in various transaction types; NEW, CANCEL and REPLACE. All message types other than NEW modify the state of the message AdvRefID.

The advertisement record format is as follows:

Advertisement

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = 7
2	AdvID	Y	
5	AdvTransType	Y	
3	AdvRefID	N	Required for Cancel and Replace AdvTransType messages
55	Symbol	Y	
65	SymbolSfx	N	
48	SecurityID	N	
22	IDSource	N	
4	AdvSide	Y	
53	Shares	Y	
44	Price	N	
15	Currency	N	Indication without currency field is interpreted as US dollars.
60	TransactTime	N	
58	Text	N	
	Standard Trailer	Y	

Indications of Interest -

Indication of interest messages are used to market merchandise which the broker is buying or selling in either a proprietary or agency capacity. The indications can be time bound with a specific expiration value. Indications are distributed with the understanding that other firms may react to the message first and that the merchandise may no longer be available due to prior trade.

Indication messages can be transmitted in various transaction types; NEW, CANCEL, and REPLACE. All message types other than NEW modify the state of the message identified in IOIRefID.

The indication of interest message format is as follows:

Indication of Interest

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = 6
23	IOIid	Y	
28	IOITransType	Y	
26	IOIRefID	N	Required for Cancel and Replace IOITransType messages
55	Symbol	Y	
65	SymbolSfx	N	
48	SecurityID	N	
22	IDSource	N	
54	Side	Y	Side of Indication Valid values: 1 = Buy 2 = Sell
27	IOIShares	Y	
44	Price	N	
15	Currency	N	Indication without currency field is interpreted as US dollars.
62	ValidUntilTime	N	
25	IOIQltyInd	N	
24	IOIOthSvc	N	Applicable only if advertised on public IOI service.
58	Text	N	
	Standard Trailer	Y	

News -

The news message is intended for use as a general free format message between the broker and institution. The message contains flags to identify the news item's urgency and to allow sorting by subject company (symbol). The News record can be originated at either the broker or institution side.

The news message format is as follows:

News

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = B
42	OrigTime	N	
61	Urgency	N	
46	***RelatdSym***	***N***	***Can be repeated multiple times if message is related to multiple symbols.***
33	LinesOfText	Y	
58	Text	Y	Repeating field, number of instances defined in LinesOfText
95	RawDataLength	N	
96	RawData	N	
	Standard Trailer	Y	

Email -

Format and purpose similar to News message, however, intended for private use between two parties.

The email message format is as follows:

Email

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = C
94	EmailType	Y	
42	OrigTime	N	
46	***RelatdSym***	***N***	***Can be repeated multiple times if message is related to multiple symbols.***.
37	OrdID	N	
11	ClOrdID	N	
33	LinesOfText	Y	
58	Text	Y	Repeating field, number of instances defined in LinesOfText
95	RawDataLength	N	
96	RawData	N	
	Standard Trailer	Y	

New Order - Single -

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

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 HandInst field). Execution instructions refer to how the order should be handled at the exchange (see ExecInst field).

The format for the new order message is as follows:

New Order - Single

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = D
11	ClOrdID	Y	
1	Account	N	
63	SettlmntTyp	N	Absence of this field is interpreted as Regular.
64	FutSettDate	N	Required when SettlmntTyp = 6 (Future) or SettlmntTyp = 8 (Sellers Option)
21	HandlInst	Y	
18	ExecInst	N	Can be defined multiple times for complex orders.
100	ExDestination	N	
81	ProcessCode	N	Used to identify soft trades at order entry.
55	Symbol	Y	
65	SymbolSfx	N	
48	SecurityID	N	
22	IDSource	N	
54	Side	Y	
38	OrderQty	Y	
40	OrdType	Y	
44	Price	N	Required for limit OrdTypes
99	StopPx	N	Required for stop OrdTypes
15	Currency	N	Message without currency field is interpreted as US dollars
59	TimeInForce	N	Absence of this field indicates Day order
12	Commission	N	
13	CommType	N	
47	Rule80A	N	
58	Text	N	

	Standard Trailer	Y	

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 as orders are worked

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.

Each execution message will contain information that will describe the current state of the order and execution status as understood by the broker.

Execution report messages can be transmitted as transaction types (ExecTranType) NEW, CANCEL, CORRECT or STATUS. Transaction types CANCEL and CORRECT modify the state of the message identified in field ExecRefID. Transaction type STATUS indicates that the execution message contains no new information, only summary information regarding order status.

- The NEW transaction type indicates that this message represents a new order, a change in status of the order, or a new fill against an existing order. The combination of the ExecTransType and OrdStatus fields will indicate how the message is to be applied to an order.
- The CANCEL transaction type applies at the execution level. The Cancel transaction will be used to cancel an execution which has been reported in error. The incorrect execution will be identified in the ExecRefID field.
- The CORRECT transaction type applies at the execution level and is used to modify an incorrectly reported fill. The incorrect execution will be identified in the ExecRefID field. *Note: Data reported in the CumQty 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.*

The OrdStatus field is used to identify the status of the current order. The order statuses are as follows:

New	Outstanding order with no executions
Partially Filled	Outstanding order with executions and remaining quantity
Filled	Order completely filled, no remaining quantity
Done	Order not, or partially, filled; no further executions forthcoming
Canceled	Canceled order with or without executions
Replaced	Replaced order with or without executions
Pending Cancel/ Replace	Order with cancel request pending, used to confirm receipt of cancel or replace request. DOES NOT INDICATE THAT THE ORDER HAS BEEN CANCELED OR REPLACED.
Stopped	Order has been stopped at the exchange
Rejected	Order has been rejected by broker. NOTE: An order can be rejected subsequent to order acknowledgement, i.e. an order can pass from New to Rejected status.

NOTE: The canceled and replaced order status are set in response to accepted cancel and replace requests. These requests are only acted upon when there is an outstanding order quantity. Requests to replace OrderQty to a level less than the LeavesQty will be rejected (OrderQty = CumQty + LeavesQty). Requests to change price on a filled order will be rejected (see Order Cancel Reject message type).

The field ClOrdID is provided for institutions to affix an identification number to an order to coincide with internal systems. The OrdId field is populated with a broker generated order number.

The execution report message format is as follows:

Execution Report

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = 8
37	OrderID	Y	
11	ClOrdID	N	Required for executions against electronically submitted orders which were assigned an ID by the institution. Not required for orders manually entered by the broker.
66	ListID	N	Required for executions against orders which were submitted as part of a list.
17	ExecID	Y	
20	ExecTransType	Y	
19	ExecRefID	N	Required for Cancel and Correct ExecTransType messages
39	OrdStatus	Y	
103	OrdRejReason	N	For optional use with OrdStatus = 8 (Rejected)

1	Account	N	Required for executions against electronically submitted orders which were assigned an account by the institution
63	SettlmntTyp	N	Absence of this field is interpreted as Regular.
64	FutSettDate	N	Required when SettlmntTyp = 6 (Future) or SettlmntTyp = 8 (Sellers Option)
55	Symbol	Y	
65	SymbolSfx	N	
48	SecurityID	N	
22	IDSource	N	
54	Side	Y	
38	OrderQty	Y	
44	Price	N	
99	StopPx	N	Required for OrdType = 4 (Stop Limit).
15	Currency	N	Message without currency field is interpreted as US dollars
59	TimeInForce	N	Absence of this field indicates Day order
18	ExecInst	N	May be defined repeated times in a single message to indicate multiple instructions
47	Rule80A	N	
32	LastShares	Y	Not required for OrdStatus = 0 (New) or ExecTransType = 3 (Status)
31	LastPx	Y	Not required for OrdStatus = 0 (New) or ExecTransType = 3 (Status)
30	LastMkt	N	
29	LastCapacity	N	
14	CumQty	Y	Not required for OrdStatus = 0 (New)
6	AvgPx	Y	Not required for OrdStatus = 0 (New)
60	TransactTime	N	
58	Text	N	
	Standard Trailer	Y	

Order Cancel/Replace 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 Cancel Request for this purpose.

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

Only a limited number of fields can be changed via the cancel/replace request message. These fields are:

- ExecInst
- OrderQty
- OrdPx
- TimeInForce

The format of the cancel request message is:

Cancel/Replace Request

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = G
37	OrderID	N	Unique identifier of *original* order as assigned by broker
41	OrigClOrdID	Y	Unique identifier of *original* order as assigned by institution.
11	ClOrdID	Y	Unique identifier of *replacement* order as assigned by institution
66	ListID	N	Required for List Orders
1	Account	N	
63	SettlmntTyp	N	Absence of this field is interpreted as Regular.
64	FutSettDate	N	Required when SettlmntTyp = 6 (Future) or SettlmntTyp = 8 (Sellers Option)
21	HandlInst	Y	Must match original order
18	ExecInst	N	Can be defined multiple times for complex orders. Replacement order will be created with new parameters (i.e. original order values will not be brought forward to replacement order unless redefined within this message).
100	ExDestination	N	
55	Symbol	Y	Must match original order
65	SymbolSfx	N	
48	SecurityID	N	Must match original order
22	IDSource	N	Must match original order
54	Side	Y	Must match original side, however, Buy and Buy Minus can be interchanged as well as Sell and Sell Plus

38	OrderQty	Y	
40	OrdType	Y	
44	Price	N	Required for limit OrdTypes
99	StopPx	N	Required for stop OrdTypes
15	Currency	N	Message without currency field is interpreted as US dollars. Must match original order.
59	TimeInForce	N	Absence of this field indicates Day order
12	Commission	N	
13	CommType	N	
47	Rule80A	N	Must match original order
58	Text	N	
	Standard Trailer	Y	

Order Cancel Request -

The order cancel request message is used to request the cancellation of all remaining quantity of an existing order.

Do not use this message to reduce the quantity of (i.e. partially cancel) an outstanding order, use the Cancel/Replace Request for this purpose.

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

The format of the cancel request message is:

Order Cancel Request

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = F
101	CxlOrdReqId	Y	Unique ID of Cancel Request
37	OrderID	N	Broker ID of original order
11	ClOrdID	Y	Client ID of original order
66	ListID	N	Required for List Orders
1	Account	N	
63	SettlmntTyp	N	Absence of this field is interpreted as Regular.
64	FutSettDate	N	Required when SettlmntTyp = 6 (Future) or SettlmntTyp = 8 (Sellers Option)
21	HandlInst	Y	
55	Symbol	Y	
65	SymbolSfx	N	
48	SecurityID	N	
22	IDSource	N	
54	Side	Y	
38	OrderQty	Y	
40	OrdType	Y	
44	Price	N	Required for limit and stop OrdTypes
15	Currency	N	Message without currency field is interpreted as US dollars
59	TimeInForce	N	Absence of this field indicates Day order
12	Commission	N	
13	CommType	N	
47	Rule80A	N	
58	Text	N	
	Standard Trailer	Y	

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; orders which are filled cannot be changed.

The execution message will be used to respond to accepted cancel request and cancel/replace request messages.

The order cancel reject message format is as follows:

Order Cancel Reject

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = 9
37	OrderID	Y	
101	CxlOrdReqId	Y	ID of Cancel Request.
66	ListID	N	Required for rejects against orders which were submitted as part of a list.
102	CxlRejReason	N	
58	Text	N	
	Standard Trailer	Y	

Order Status Request -

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

The format of the order status request message is:

Order Status Request

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = H
37	OrderID	N	
11	ClOrdID	Y	
55	Symbol	Y	
65	SymbolSfx	N	
54	Side	Y	
	Standard Trailer	Y	

Allocation -

The allocation record is used by the institution to instruct the broker on how to allocate executed shares to sub-accounts.

The allocation record contains repeating fields for each sub-account; the repeating fields are shown below in typeface ***Bold-Italic.*** The relative position of the repeating fields is important in this record, i.e. each instance of allocation must be in the order shown below.

- The total shares allocated must equal the Shares value which must equal the total executed quantity of the original order.
- The number of instances of allocations is indicated in NoAllocs.
- Multiple orders can be combined for allocation by identifying the number of orders in the NoOrders field and each individual order in the OrderID fields. Combined orders must have the same ticker, trade date, settlement date and side.
- Single orders cannot be combined with list orders for allocation.

Allocation

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = J
70	AllocID	Y	
71	AllocTransTyp	Y	
72	RefAllocID	N	Required for AllocTransType = R (Replace) or C (Cancel)
73	NoOrders	N	Indicates number of orders to be combined for allocation. Absense of this field indicates allocation is for one order.
11	***ClOrdID***	***Y***	***Can be repeated to identify multiple orders to be combined for average pricing.***
37	***OrderID***	***N***	***Can be repeated to identify multiple orders to be combined for average pricing.***
66	***ListID***	***N***	***Required for List Orders. Can be repeated to identify multiple orders to be combined for average pricing.***
54	Side	Y	
55	Symbol	Y	
65	SymbolSfx	N	
48	SecurityID	N	
22	IDSource	N	
53	Shares	Y	Total number of shares to be allocated, (must match total executed quantity of order)
6	AvgPx	Y	
74	AvgPrxPrecision	N	Absense of this field indicates that default precision arranged by the broker/institution is to be used.

75	TradeDate	N	Absense of this field indicates current day
63	SettlmntTyp	N	Absence of this field is interpreted as Regular.
77	OpenClose	N	
58	Text	N	
78	NoAllocs	Y	
79	***AllocAccount***	***Y***	
80	***AllocShares***	***Y***	
81	***ProcessCode***	***N***	
76	***ExecBroker***	***N***	***Required for step-in and step-out trades***
12	***Commission***	***N***	
13	***CommType***	***N***	
85	***NoDlvyInst***	***N***	
92	***BrokerOfCredit***	***N***	
86	***DlvyInst***	***N***	
	Standard Trailer	Y	

Allocation ACK -

The allocation ACK record is used by the broker to acknowledge the receipt and status of an allocation record received from the institution.

Allocation ACK

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = P
70	AllocID	Y	
87	AllocStatus	Y	
88	AllocRejCode	N	Required for AllocStatus = 1 (rejected)
58	Text	N	Can include explanation for AllocRejCode = 7 (other)
	Standard Trailer	Y	

New Order List -

The new order list message type is used by institutions wishing to electronically submit lists of related orders to a broker for execution.

The New Order List is intended for use in *staging* lists to be executed by the broker. If the institution wishes to work a list using the broker's execution services the orders should be submitted as individual New Order - Single's.

After staging, the list can be operated on in the following ways:

Execute: The broker can be instructed to release the list for execution by sending the List-Execute message.

Cancel: After the list has been staged with the broker, it can be cancelled 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 cancelled via the Order Cancel Request message.

Status: A status of the list can requested via the submission of the List-Status Request message. The broker will respond with one or more List-Status messages which will report executed quantity, cancelled quantity and average price for each order in the list.

Replace: Individual orders within the list can be replaced via Order Cancel/Replace Request messages.

Executions against orders within the list will *not* normally be reported as they occur. (If this feature is desired the institution and broker should arrange for this reporting as a custom feature using the Execution message.) Executions against the list will be reported within the List-Status message.

The format for the new order list message is as follows:

New Order - List

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = E
66	ListID	Y	Must be unique, by customer, for the day
67	ListSeqNo	Y	
68	ListNoOrds	Y	
69	ListExecInst	N	Include only in ListSeqNo = 1 message
11	ClOrdID	Y	
1	Account	N	
63	SettlmntTyp	N	Absence of this field is interpreted as Regular.
64	FutSettDate	N	Required when SettlmntTyp = 6 (Future) or SettlmntTyp = 8 (Sellers Option)
21	HandlInst	Y	
18	ExecInst	N	Can be defined multiple times for complex orders.

100	ExDestination	N	
55	Symbol	Y	
65	SymbolSfx	N	
48	SecurityID	N	
22	IDSource	N	
54	Side	Y	
38	OrderQty	Y	
40	OrdType	Y	
44	Price	N	Required for limit OrdTypes
99	StopPx	N	Required for stop OrdTypes
15	Currency	N	Message without currency field is interpreted as US dollars
59	TimeInForce	N	Absence of this field indicates Day order
12	Commission	N	
13	CommType	N	
47	Rule80A	N	
58	Text	N	
	Standard Trailer	Y	

List Status -

The list status message is issued as the response to a List Status Request message and indicates the current state of the orders within the list as they exists at the broker's site.

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 order; the repeating fields are shown below in typeface ***Bold-Italic.*** The relative position of the repeating fields is important in this record, i.e. each instance of ClOrdID, CumQty, CxlQty and AvgPx must be in the order shown below.

Each list status message will report on only a maximum of 50 orders; if the list contains more than 50 orders multiple status messages will be required.

The list status message format is as follows:

List Status

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = N
66	ListID	Y	
82	NoRpts	Y	Total number of messages required to status complete list.
83	RptSeq	Y	Sequence number of this report message.
73	NoOrders	Y	Number of orders statused in this message, i.e. number of repeating groups to follow.
11	***ClOrdID***	***Y***	
14	***CumQty***	***Y***	
84	***CxlQty***	***Y***	
6	***AvgPx***	***Y***	
	Standard Trailer	Y	

List Execute -

The list execute message type is used by institutions to instruct the broker to begin execution of a previously submitted list.

The format for the list execute message is as follows:

List Execute

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = L
66	ListID	Y	Must be unique, by customer, for the day
58	Text	N	
	Standard Trailer	Y	

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 cancelled 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 cancelled via the Order Cancel Request message.

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

List Cancel Request

Tag	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = K
66	ListID	Y	
58	Text	N	
	Standard Trailer	Y	

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	*Field Name*	*Req'd*	*Comments*
	Standard Header	Y	MsgType = M
66	ListID	Y	
58	Text	N	
	Standard Trailer	Y	

Field Definitions

The following is a catalog of fields used to define the application and session protocol messages.

Field ID (TAG)	Field Name	Format	Description
1	Account	char	Account mnemonic
2	AdvId	int	Unique identifier of advertisement message
3	AdvRefID	int	Reference identifier used with CANCEL and REPLACE transaction types.
4	AdvSide	char	Broker's side of advertised trade Valid values: B = Buy S = Sell X = Cross T = Trade
5	AdvTransType	char	Identifies advertisement message transaction type Valid values: N = New C = Cancel R = Replace
6	AvgPx	float	Calculated average price of all fills on this order. Field may be blank for NEW, CANCEL and REPLACE ExecTransTypes. Valid values: (0 - 99999.99999)
7	BeginSeqNo	int	Message sequence number of first record in range to be resent
8	BeginString	char	Identifies beginning of new message and protocol version. ALWAYS FIRST FIELD IN MESSAGE. (*Always unencrypted*) Valid value: FIX.2.0

9	BodyLength	int	Message length, in bytes, forward to the CheckSum field. ALWAYS SECOND FIELD IN MESSAGE. (*Always unencrypted*) Valid values: 0 - 9999
10	CheckSum	char	Three byte, simple checksum (see Appendix B for description). ALWAYS LAST FIELD IN RECORD, also serves as end of record delimiter. Always defined as three characters. Checksum is calculated on encrypted data stream as transmitted between parties. (*Always unencrypted*)
11	ClOrdID	char	Order identifier assigned by institution application
12	Commission	float	Commission Valid values: -9.999 - 9999.999
13	CommType	char	Commission type Valid values: 1 = per share 2 = percentage 3 = absolute
14	CumQty	int	Total number of shares filled. Field may be blank for NEW, CANCEL and REPLACE ExecTransTypes. Valid values: (0 - 1000000000)
15	Currency	char	Identifies currency used for price, Absense of this field in a message is interpreted as US dollars. See Appendix A for valid values.
16	EndSeqNo	int	Message sequence number of last record in range to be resent. If request is for a single record BeginSeqNo = EndSeqNo. If request is for all messages subsequent to a particular message, EndSeqNo = "99999"
17	ExecID	int	Unique identifier of execution message (Will be blank for OrdStat messages)

18	ExecInst	char	Instructions for order handling on exchange trading floor. If more than one instruction is applicable to an order, this field can be defined multiple times in a single message. Some of following values are only valid for manually handled orders (M), some are only valid for DOT orders (D), some are applicable to both (MD). Valid values: 1 = Not held (M) 2 = Work (M) 3 = Go along 4 = Over the day (M) 5 = Held (M) 6 = Participate don't initiate (M) 7 = Strict scale (M) 8 = Try to scale (M) 9 = Stay on bidside (M) 0 = Stay on offerside (M) A = No cross (M) B = OK to cross (M) C = Call first (M) D = Percent of volume (M) E = Do not increase - DNI (D) F = Do not reduce - DNR (D) G = All or none - AON (D)
19	ExecRefID	int	Reference identifier used with Cancel, Replace and Correct transaction types.
20	ExecTransType	char	Identifies transaction type Valid values: 0 = New 1 = Cancel 2 = Correct 3 = Status
21	HandlInst	char	Instructions for order handling on Broker trading floor Valid values: 1 = DOT order, private, no Broker intervention 2 = DOT order, public, Broker intervention OK 3 = Manual order, best execution
22	IDSource	char	Identifies class of alternative SecurityID Valid values: 1 = CUSIP 2 = SEDOL 3 = QUIK
23	IOIid	int	Unique identifier of IOI message.

24	IOIOthSvc	char	Indicates if, and on which other services, the indication has been advertised. Each character represents an additional service (e.g. if on Bridge and Autex, field = BA, if only on Autex, field = A) Valid values: A = Autex B = Bridge
25	IOIQltyInd	char	Relative quality of indication Valid values: L = Low M = Medium H = High
26	IOIRefID	int	Reference identifier used with CANCEL, REPLACE, PURGE and REMOVE transaction types.
27	IOIShares	char	Number of shares in numeric or relative size. Valid values: 0 - 1000000000 S = Small M = Medium L = Large
28	IOITransType	char	Identifies advertisement message transaction type Valid values: N = New C = Cancel R = Replace
29	LastCapacity	char	Broker capacity in order execution Valid values: 1 = Agent 2 = Cross as agent 3 = Cross as principal 4 = Principal
30	LastMkt	char	Market of execution for last fill Valid values: A = AMEX B = Boston D = Cincinnati M = Midwest N = NYSE O = OTC P = PCSE W = PBW

31	LastPX	float	Price of last fill. Field will be blank for NEW, CANCEL and REPLACE ExecTransTypes and may be blank when order is NEW, CXLD, DONE or RPLD status Valid value: (0 - 9999.999)
32	LastShares	int	Quantity of shares bought/sold on this fill. Field will be absent for NEW, CANCEL and REPLACE ExecTransTypes and may be absent when order is NEW, CXLD, DONE or RPLD status.
33	LinesOfText	int	Identifies number of lines of text body
34	MsgSeqNum	int	Integer message sequence number. Incremented for each message except heartbeat. Valid values: 0 - 99999
35	MsgType	char	Defines message type. ALWAYS THIRD FIELD IN MESSAGE. (*Always unencrypted*) *Note: A "U" as the first character in the MsgType field indicates that the message format is privately defined between the sender and receiver.* Valid values: 0 = Heartbeat 1 = Test Request 2 = Resend Request 3 = Reject 4 = Sequence Reset 5 = Logout 6 = Indication of Interest 7 = Advertisement 8 = Execution Report 9 = Order Cancel Reject A = Logon B = News C = Email D = Order - Single E = Order - List F = Order Cancel Request G= Order Cancel/Replace Request H= Order Status Request J = Allocation K = List Cancel Request L = List Execute M = List Status Request N = List Status P = Allocation ACK

36	NewSeqNo	int	New sequence number Valid values: 0 - 99999
37	OrderID	char	Unique identifier for Order as assigned by broker
38	OrderQty	int	Number of shares ordered Valid values: (0 - 1000000000)
39	OrdStatus	char	Identifies current status of order. Valid values: 0 = New 1 = Partially filled 2 = Filled 3 = Done for day 4 = Canceled 5 = Replaced 6 = Pending Cancel/Replace 7 = Stopped 8 = Rejected
40	OrdType	char	Order type. Valid values: 1 = Market 2 = Limit 3 = Stop 4 = Stop limit 5 = Market on close 6 = With or without 7 = Limit or better 8 = Limit with or without 9 = On basis A = On close B = Limit on close
41	OrigClOrdID	char	Original order id as assigned by the institution, used to identify original order in cancel/replace requests.
42	OrigTime	char	Time of message origination in HH:MM:SS format Valid values: HH: 00 - 23 MM: 00 - 59 SS: 00 - 59

43	PossDupFlag	char	Indicates possible retransmission of message with this sequence number Valid values: Y = Possible duplicate N = Original transmission
44	Price	float	Price per share Valid values: 0 - 9999.9999
45	RefSeqNum	int	Reference message sequence number Valid values: 0 - 99999
46	RelatdSym	char	Symbol of issue related to story. Can be repeated within message to identify multiple companies.
47	Rule80A	char	Indicates order type upon which exchange Rule 80A is applied. Valid values: A = Agency single order I = Individual Investor, single order D = Program Order, index arb, for Member firm/org C = Program Order, non-index arb, for Member firm/org J = Program Order, index arb, for individual customer K = Program Order, non-index arb, for individual customer U = Program Order, index arb, for other agency Y = Program Order, non-index arb, for other agency M = Program Order, index arb, for other member N = Program Order, non-index arb, for other member W = All other orders as agent for other member
48	SecurityID	char	CUSIP or other alternate security identifier
49	SenderCompID	char	Assigned value. (*Always unencrypted*)
50	SenderSubID	char	Assigned value. (*Always unencrypted*)
51	SendingDate	char	Date of message transmission in YYMMDD format Valid values: YY = 00-99 MM = 01-12 DD = 01-31

52	SendingTime	char	Time of message transmission in HH:MM:SS format Valid values: HH: 00 - 23 MM: 00 - 59 SS: 00 - 59
53	Shares	int	Number of shares Valid values: 0 - 1000000000
54	Side	char	Side of order Valid values: 1 = Buy 2 = Sell 3 = Buy minus 4 = Sell plus 5 = Sell short 6 = Sell short exempt 7 = Traded 8 = Crossed
55	Symbol	char	Ticker symbol
56	TargetCompID	char	Assigned value. (*Always unencrypted*)
57	TargetSubID	char	Assigned value. (*Always unencrypted*)
58	Text	char	Free format text string
59	TimeInForce	char	Specifies how long the order remains in effect. Absence of this field is interpreted as DAY. Valid values: 0 = Day 1 = Good Till Cancel (GTC) 2 = At the Opening (OPG) 3 = Immediate or Cancel (OC) 4 = Fill or Kill (FOK) 5 = Good Till Crossing (GTX)
60	TransactTime	char	Time of execution/order creation in HH:MM:SS format Valid values: HH: 00 - 23 MM: 00 - 59 SS: 00 - 59

61	Urgency	char	Urgency flag Valid values: 0 = Normal 1 = Flash 2 = Background
62	ValidUntilTime	char	Indicates expiration time of indication message in HH:MM:SS format Valid values: HH: 00 - 23 MM: 00 - 59 SS: 00 - 59
63	SettlmntTyp	char	Indicates order settlement period. Absence of this field is interpreted as Regular. Valid values: 0 = Regular 1 = Cash 2 = Next Day 3 = T+2 4 = T+3 5 = T+4 6 = Future 7 = When Issued 8 = Sellers Option
64	FutSettDate	char	Specific date of trade settlement in YYMMDD format. Required when SettlmntTyp = 6 (Future) or SettlmntTyp = 8 (Sellers Option). Valid values: YY = 00-99 MM = 01-12 DD = 01-31
65	SymbolSfx	char	Additional information about the security (e.g. preferred, warrants, etc.). Absence of this field indicates common. Valid values: As defined in the NYSE Stock and bond Symbol Directory and in the AMEX Fitch Directory
66	ListID	char	Customer assigned list identifier used to associate multiple individual orders.
67	ListSeqNo	int	Sequence of individual order within list (i.e. *ListSeqNo* of *ListNoOrds,* 2 of 25, 3 of 25, . . .)

68	ListNoOrds	int	Total number of orders within list (i.e. *ListSeqNo* of *ListNoOrds,* e.g. 2 of 25, 3 of 25, . . .)
69	ListExecInst	char	Free format text message containing list handling and execution instructions.
70	AllocID	int	Unique identifier for allocation record.
71	AllocTransType	char	Identifies allocation transaction type Valid values: 0 = New 1 = Replace 2 = Cancel
72	RefAllocID	int	Reference identifier to be used with Replace and Cancel AllocTransType records.
73	NoOrders	int	Indicates number of orders to be combined for average pricing and allocation.
74	AvgPrxPrecision	int	Indicates number of decimal places to be used for average pricing. Absense of this field indicates that default precision arranged by the broker/institution is to be used.
75	TradeDate	char	Indicates date of trade referenced in this record in YYMMDD format. Absense of this field indicates current day. Valid values: YY = 00-99 MM = 01-12 DD = 01-31
76	ExecBroker	char	Identifies executing / give-up broker. Standard NASD market-maker mneumonic is preferred.
77	OpenClose	char	For options only.
78	NoAllocs	int	Number of AllocAccount/AllocShares/ProcessCode instances included in allocation record.
79	AllocAccount	char	Sub-account mneumonic

80	AllocShares	int	Number of shares to be allocated to specific sub-account
81	ProcessCode	char	Processing code for sub-account. Absense of this field in AllocAccount / AllocShares / ProcessCode instance indicates regular trade. Valid values: 0 = regular 1 = soft dollar 2 = step-in 3 = step-out 4 = soft-dollar step-in 5 = soft-dollar step-out
82	NoRpts	int	Total number of reports within series.
83	RptSeq	int	Sequence number of message within report series.
84	CxlQty	int	Total number of shares cancelled for this order.
85	NoDlvyInst	int	Number of delivery instruction fields to follow
86	DlvyInst	char	Free format text field to indicate delivery instructions
87	AllocStatus	int	Identifies status of allocation. Valid values: 0 = accepted 1 = rejected 2 = partial accept
88	AllocRejCode	int	Identifies reason for rejection. Valid values: 0 = unknown account(s) 1 = incorrect quantity 2 = incorrect average price 3 = unknown executing broker mnuemonic 4 = commission difference 5 = unknown OrderID 6 = unknown ListID 7 = other
89	Signature	data	Electronic signature

90	SecureDataLen	int	Length of encrypted message
91	SecureData	data	Actual encrypted data stream
92	BrokerOfCredit	char	Broker to receive trade credit
93	SignatureLength	int	Number of bytes in signature field.
94	EmailType	char	Email message type. Valid values: 0 = New 1 = Reply 2 = Admin Reply
95	RawDataLength	int	Number of bytes in raw data field.
96	RawData	data	Unformatted raw data, can include bitmaps, word processor documents, etc.
97	PossResend	char	Indicates that message may contain information that has been sent under another sequence number.
98	EncryptMethod	int	Method of encryption. Valid values: 0 = None 1 = PKCS 2 = DES 3 = PKCS/DES
99	StopPx	float	Price per share Valid values: 0 - 9999.9999

100	ExDestination	int	Execution destination as defined by institution when order is entered. Valid values: 0 = None 1 = NYSE/AMEX 2 = Midwest 3 = Boston 4 = Posit 5 = Instinet 6 = Cincinnati 7 = PCSE 8 = PBW 9 = OTC
101	CxlOrdReqId	char	Unique ID of cancel request.
102	CxlRejReason	int	Code to identify reason for cancel rejection. Valid values: 0 = Too late to cancel 1 = Unknown order
103	OrdRejReason	int	Code to identify reason for order rejection. Valid values: 0 = Broker option 1 = Unknown symbol 2 = Exchange closed 3 = Order exceeds limit

Appendix A
Valid Currency Codes

AED	United Arab Emirates Dirham
AFA	Afghani
ALL	Albanian Lek
ANG	Netherlands Antilles Guilder
AOK	Angolan Kwanza
ARA	Argentinan Austral
ATS	Austrian Schilling
AUD	Austral $
BBD	Barbados $
BDT	Bangladesh Taka
BEC	Cnv Belgium Franc
BEF	Belgium Franc
BEL	Fin Belgium Franc
BGL	Bulgarian Lev
BHD	Bahrainian Dinar
BIF	Burndi Franc
BMD	Bermudian Dollar
BND	Brunei Dollar
BOP	Bolivian Peso
BRN	Brazilian New Cruzdo
BSD	Bahamain $
BUK	Burma Kyat
BWP	Botswanan Pula
BZD	Belize Dollar
CAD	Canadian Dollar
CHF	Swiss Franc
CLP	Chilian Peso
CNY	Chinese Yn Renminb
COP	Columbian Peso
CRC	Costa Rican Colon
CSK	Czechoslovakian Koruna
CUP	Cuban Peso
CVE	Cape Verde Escd
CYP	Cyprus Pound
DEM	Fed Rep Of Germany Deuts Mark
DJF	French Afars+Issas Djbt Franc
DKK	Danish Krone
DOP	Dominican Republic Peso
DZD	Alegerian Dinar
ECC	Ecu Compon
ECS	Ecuadoran Sucre
ECU	European Currency Unit
EGP	Egyptian Pound
ESA	B Spanish Pesata
ESB	A Spanish Pesata
ESP	Spanish Pesata
ETB	Ethiopian Birr
EUA	European Unit Acct
FIM	Finish Markka
FJD	Fiji Dollar
FKP	Falkland Is. (Mal) Pound
FRF	French Franc
GBP	Canton+Enderbury Pnd Sterling
GHC	Ghana Cedi
GIP	Gibralter Pound
GMD	Gambian Dalasi
GNS	Guinea-Bissau Syli
GQE	Equatorial Guinea Ekwele
GRD	Greek Drachma
GTQ	Guatemalan Quetzal
GWB	Sao Tome + Principe Gun Escudo
GWP	Guinea-Bissau Peso
GYD	Guyana Dollar
HKD	Honk Kong Dollar
HNL	Honduran Lempira
HTG	Haitian Gourde
HUF	Hungarian Forint
IDR	Indonesian Rupiah
IEP	Irish Pound
ILS	Israeli Shekl
INR	Indian Rupee
IQD	Iraqian Dinar
IRR	Iranian Rail
ISK	Iceland Krona
ITL	Italian Lira
JMD	Jamaican Dollar
JOD	Jordanian Dinar
JPY	Japanese Yen
KES	Kenyan Shilling
KHR	Kampuchea Dem. Riel
KMF	Comoros Franc
KPW	Dem Peo Rep Of Korea Won
KRW	Rep Of Korea Won
KWD	Kuwait Dinar
KYD	Cayman Island Dollar
LAK	Lao Peoples Dem Kip
LBP	Lebonese Pound
LKR	Sri Lanka Rupee
LRD	Liberian Dollar
LSM	Lesotho Maloti
LUF	Luxembourg Franc
LYD	Libian Arab Republic Dinar
MAD	Moroccan Dirham
MGF	Madagascan Franc
MLF	Mali Franc
MNT	Mongolian Tugrik
MOP	Macau Pataca
MRO	Mauritania Ouguiya
MTP	Malta Pound
MUR	Mauritius Rupee
MVR	Maldives Rupee
MWK	Malawi Kwacha
MXP	Mexican Peso
MYR	Malaysian Ringgt
MZM	Mozamique Metical
NGN	Nigerian Naira
NIC	Nicaraguan Cordoba
NLG	Netherlands Guilder
NOK	Norwegian Krone
NPR	Nepal Rupee
NZD	New Zealand Dollar
OMR	Oman Rial Omani
PAB	Panamanian Balboa
PES	Peruvian Sol
PGK	Papua New Guinea Kina
PHP	Philippine Peso
PKR	Pakistani Rupee
PLZ	Polish Zloty
PTE	Portugese Escud
PYG	Paraguan Guarani
QAR	Qutar Rial
ROL	Romanian Leu
RWF	Rwandan Franc
SAR	Saudi Arabian Riyal
SBD	Solomon Islands Dollar
SCR	Seychelles Rupee
SDP	Sudan Pound
SDR	Spc Drw Rt
SEK	Swedish Krone
SGD	Singapore Dollar
SHP	St. Helena Pound
SLL	Sierra Leone Leone
SOS	Somalian Shilling
SRG	Surinam Guilder
STD	Sao Tome + Principe Dobra
SUR	Ukrainian Ssr Rouble
SVC	El Salvador Colon
SYP	Syrian Arab Rep Pound
SZL	Swaziland Lilangeni
THB	Thai Baht
TND	Tunisian Dinar
TOP	Tonga Pa'anga
TPE	Tim Escudo
TRL	Turkish Lira
TTD	Trinidad & Tobago Dollar
TWD	Tai Dollar
TZS	Un Rep Tanzania Shilling
UGS	Ugandan Shilling
USD	Us Dollar
UYP	Urug Peso
VEB	Bolivar
VND	Vietnamese Dong
VUV	Vanuatuan Vatu
WST	Samoan Tala
XAF	Gabon Cfa Franc
XCD	St. Lucia Ecb Dollar
XOF	Togo Wafr Franc
XPF	New Caledonian Cfp Franc
YDD	Yemn Dinar
YER	Yemen Rial
YUD	Yugo Dinar
ZAR	South African Rand
ZMK	Kwacha
ZRZ	Zaire
ZWD	Zimbabwe Dollar

Appendix B

CheckSum Calculation

The checksum of a FIX message is calculated by summing every byte of the message up to but not including the checksum field itself. This checksum is then transformed into a modulo 256 number for transmission and comparison. The checksum is calculated after all encryption is completed, i.e. the message as transmitted between parties is processed.

For transmission, the checksum must be sent as printable characters, so the checksum is transformed into three ASCII digits.

For example, if the checksum has been calculated to be 274 then the modulo 256 value is 22. This value would be transmitted a |10=022| where "10="is the tag for the checksum field.

A sample code fragment to generate the checksum field is as follows:

```
char *cksum( char *buf, long bufLen )
GenerateCheckSum (int checksum)
{
    static char tmpBuf[ 4 ];
    long idx;
    unsigned int cks;

    for( idx = 0L, cks = 0; idx < bufLen; cks += (unsigned int)buf[ idx++ ] );
    sprintf( tmpBuf, "%03d", (unsigned int)( cks % 256 ) );
    return( tmpBuf );
}
```