FASTMATCH

BINARY PROTOCOL SPECIFICATION DOCUMENT

STREAMING

REVISION HISTORY

Version	Last Updated	Updates
1.0	October 30, 2015	Initial Version
1.1	November 3, 2015	Changes to the Subscription Request Message
1.2	November 19, 2015	Increased size of ExecID to 64 in Trade and DoNotKnow messages Extended the list of DKReason codes
1.3	December 1, 2015	Replaced 'Status' field in Order Canceled message with a 'CcyPair' field
1.4.1	December 7, 2015	Changed message type for majority of business massages to lowercase. Replaced OrderID with ClOrderID in DoNotKnow messages
1.4.2	December 23, 2015	Fixed errors on diagrams page 13 and 15

CONTENTS

1 INTRODUCTION

This document is provided as a guide for FASTMATCH clients, as to how the FASTMATCH's Binary Streaming Protocol (FMBinaryStream) may be used for price making spot FX and Spot Metals on the FASTMATCH platform using the binary protocol.

FMBinaryStream is a simplified protocol that allows FASTMATCH participants to publish market data to FASTMATCH and receive new orders and cancel orders and provide executions.

All counterparties will need to certify their trading system with FASTMATCH in the User Acceptance Testing ("UAT") environment before being called production ready.

2 OVERVIEW

2.1 CONNECTIONS

FASTMATCH requires two TCP connections, one for market data and a second for trading. FASTMATCH will be the initiator, and Market Maker (MM) will be the acceptor to these connections. MM should have servers available during agreed upon market hours for FASTMATCH connection. FASTMATCH will connect and disconnect at agreed upon times. FASTMATCH will make two connections to MM:

Quote connection – FASTMATCH logs on and connects to MM. FASTMATCH will send market data subscription requests. MM will either begin streaming market data or reject the subscription for symbols or pairs they are not providing quotes for. FASTMATCH will send individual market data subscription messages for each symbol. This connection will be referred to as "STREAM" throughout this document.

Trading connection – FASTMATCH logs on and connects to MM. When an FASTMATCH client attempts to deal on a MM's quote, FASTMATCH will submit limit orders to the MM at the specified price. MM will respond with execution reports detailing a fill or a rejection of the requested order. This connection will be referred to as "TRADE" throughout this document.

3 CONNECTIVITY

3.1 FASTMATCH MATCHING ENGINE LOCATIONS

FASTMATCH matching engine is located in Equinix NY4, LD4 and TY3 Data Centers:

- NY4, 755 Secaucus Road, Secaucus, NJ 07094
- LD4, 2 Buckingham Avenue, Slough, Berkshire, SL1 4NB
- TY3, 1-9-20 Edagawa Koto-Ku Tokyo 135-0051

3.2 CONNECTIVITY OPTIONS

- Clients have a choice of establishing cross-connect and internet connectivity to FASTMATCH NY4, LD4 and TY3 locations.
- Local cross-connect to FASTMATCH ECN cages in NY4, LD4 and TY3 data centers could be used for both Production and UAT access.
- Metro connections from other data centers are accepted.
- Order Entry traffic will be TCP based with unique target (IP:port) for each TRADE session.
- Market Data traffic will be TCP based with unique target (IP:port) for each STREAM session.
- No multicast traffic will travel via client connectivity.
- To start UAT certification process, client can establish the Internet connectivity to FASTMATCH.

3.3 CROSS-CONNECTIVITY

- All client cross-connect connectivity is 1Gpbs Multimode or Single-mode fiber.
- FASTMATCH will issue a client a LOA to connect to FASTMATCH ECN with up to two fiber cross-connects.
- To avoid delays, we ask a client to confirm the correct firm or third-party agent name to be used in LOA.
- If two cross-connects are ordered, then they will be connected the different access switches for redundancy.
- BGP is preferred choice even on a single cross-connect connection for support purposes. And static routing is accepted if client hardware cannot support BGP.
- FASTMATCH will advertise registered IP address space from a registered BGP ASN.
- Cross-connect will be addressed using RFC 1918 address space (preferred). Registered IP space could be used to avoid the IP address conflict.
- FASTMATCH will accept client's registered IP Address and BGP ASN. If required, FASTMATCH will assign client's server farm IP addresses and BGP ASN.

3.4 INTERNET CONNECTIVITY

Internet connectivity is available at NY4, LD4 and TY3 locations. There are two internet providers at every location. They may be utilized as main and failover connections. The main connection can be selected based on roundtrip statistics.

3.5 SYSTEM/SERVICE AVAILABILITY

Market hours for trading are the same as for all FASTAMTCH sessions: the market opens on Sunday at 5:30 PM NY time and closes on Friday at 5:00 PM NY time. The FIX trade service is off-line from 17:00:00 EST/EDT until 17:30:00 EST/EDT, Daily Monday through Thursday. During this time FASTMATCH resets the inbound and outbound sequence numbers on all OUTCH trade sessions.

To successfully connect after 17:30:00 EST/EDT, trading sessions. Typically, when initially logging into a server the client will set the Requested Sequence Number field to 1 and leave the Requested Session field blank in the Login Request Packet

3.6 CERTIFICATION

Certification with FASTMATCH is aimed to be as straight forward as possible. UAT testing must be completed in advance of production trading. The FASTMATCH IT team will assist you in connecting and testing.

3.7 TIME

All times are in GMT per FIX protocol standard. We expect that all customers sync their clocks with an external time source.

3.8 DISCONNECTIONS

Connections can be lost due to network issues. If your session disconnects, all outstanding quotes will be cancelled from the market. Session information is not retained after a reset. FASTMATCH will send new market data subscription requests to restart price streams after a disconnection.

3.9 PROTOCOL STACK

Fastmatch Binary STREAM specification uses TCP protocol for order submission and execution delivery. Additionally, Fastmatch utilizes SoupBinTCP protocol on top of TCP for session control, reliable delivery, and recovery of execution stream.

SoupBinTCP is a lightweight point-to-point protocol, built on top of TCP/IP sockets that allow delivery of a set of sequenced messages from a server to a client in real-time. SoupBinTCP guarantees that the client receives each message generated by the server in sequence, even across underlying TCP/IP socket connection failures.

SoupBinTCP is ideal for systems where a server needs to deliver a logical stream of sequenced messages to a client in real-time but does not require the same level of guarantees for client generated messages either because the data stream is unidirectional or because the server application generates higher-level sequenced acknowledgments for any important client-generated messages.

SoupBinTCP is designed to be used in conjunction with higher lever protocols that specify the contents of the messages that SoupBinTCP messages deliver. The SoupBinTCP protocol layer is opaque to the higher-level messages. Note that unlike the ASCII version, messages may include any possible byte.

SoupBinTCP also includes a simple scheme that allows the server to authenticate the client on login.

Fastmatch employs SoupBinTCP in accordance with SoupBinTCP, Version 3.0 specification with the **two exceptions:**

By default, all integers are sent and received in little-endian format for performance reason.

There is an extra field in Login Request Packet message indicating version number. It is equal 1 for the current version.

This protocol deviation helps to circumvent redundant conversion from little-endian to network order (big-endian) and back, assuming prevalence of Linux/Windows x64-processor based systems. Yet, upon a client's request SoupBinTCP/ FMBinaryStream may be configured to exchange data in network byte order (big-endian) endianness.

4 SUPPORTED MESSAGE SET

In the message descriptions that follow, the following convention is used to indicate direction:

- In – a message sent by a MM to FASTMATCH
- Out – a message sent by FASTMATCH to a MM.
- In/Out – a message that can be sent to or from a MM or FASTMATCH

Available fields, requirements, values and their associated meanings are documented in the Message Details section. The supported message set is as follows:

4.1 SESSION LEVEL MESSAGES

Session-level messages are the same for TCP and are identical to session-level messages used in OUCH spec.

Direction	Message	Type	Description
Out	**Login Request Packet**	L	FASTMATCH will send a Login Request Packet immediately upon establishing a connection to the MM. FASTMATCH and MM must mutually agree upon the username and password fields. They provide simple authentication to prevent a client from inadvertently connecting to the wrong server/session. Both Username and Password are case-insensitive and should be padded on the right

			with spaces. The server can terminate an incoming TCP/IP socket if it does not receive a Login Request Packet within a reasonable period of time (typically 30 seconds).
In	**Login Accept Packet**	A	MM server sends a Login Accepted Packet in response to receiving a valid Login Request from the client. This packet will always be the first sent by the server after a successful login request.
In	**Login Reject Packet**	J	MM server sends this packet in response to an invalid Login Request Packet from the FASTMATCH. The Login Rejected Packet should be the only packet sent by the MM in the case of an unsuccessful login attempt.
In	**Sequenced Data Packet**	S	The Sequenced Data Packets act as an envelope to carry the actual sequenced data messages that are transferred from the MM to FASTMATCH. Each Sequenced Data Packet carries one message from the higher-lever protocol. The sequence number of each message is implied; the initial sequence number of the first Sequenced Data Packet for a given TCP/IP connection is specified in the Login Accepted Packet and the sequence number increments by 1 for each Sequenced Data Packet transmitted. Since logical packets are carried via TCP/IP sockets, the only way logical packets can be lost is in the event of a TCP/IP socket connection failure. In this case, the client can reconnect to the server and request the next expect sequence number and pick up where it left off. **Fastmatch will not request retransmission Market Data messages**
In	**Server Heartbeat**	H	The MM server should send a FASTMATCH Heartbeat Packet anytime more than 1 second passes where no data has been sent to the client. The client can then assume that the link is lost if it does not receive anything for an extended period of time.
In	**End of Session**	Z	The MM server will send an End of Session Packet to denote that the current session is finished. The connection will be closed shortly after this packet, and the user will no longer be able to reconnect to the current session.
Out	**Unsequenced Data Packets**	U	The Unsequenced Data Packets act as an envelope to carry the actual data messages that are transferred from the FASTMATCH to the MM server. These messages are not sequenced and may be lost in the event of a socket failure. The higher-level protocol must be able to handle these lost messages in the case of a TCP/IP socket connection failure.

Out	**Client Heartbeat**	R	FASTMATCH should send a MM Heartbeat Packet anytime more than 1 second passes where no data has been sent to the server. The server can then assume that the link is lost if it does not receive anything for an extended period of time.
Out	**Client Logout Request**	O	FASTMATCH may send a Logout Request Packet to request the connection be terminated. Upon receiving a Logout Request Packet, the server will immediately terminate the connection and close the associated TCP/IP

4.2 BUSINESS MESSAGES

Direction	Message	Type	Purpose
Out	**Subscription Request**	s	Request to subscribe to a symbol. Application message sent by FASTMATCH after a session has been established telling the MM the currency pairs for which FASTMATCH wants to receive quotes. STREAM message type.
In	**Subscription Response**	r	Response indicating whether subscription accepted or rejected
In	**Book Update**	b	Book Update, followed by PriceAdd and PriceCancel updates.
In	**Price Add**	p	New Price Update
In	**Price Cancel**	x	Cancelling existing price on a book
Out	**New Order**	d	To submit orders to FASTMATCH
In	**New Order ACK**	a	To provide notification on order acceptance
In	**Order Canceled**	c	To provide notification on order cancellation
In	**Trade Execution**	t	To provide notification on order fills & partial fills
In	**Reject**	J	Reject Message that can be sent in response any message in case server considers message invalid. See Appendix for supported Reject Error codes.

Out	**Order Timeout**	o	FASTMATCH defined application message. If a reply is not received from a MM to a New Order application message within a defined period of time, FASTMATCH will send the MM an Order Timeout message. The client who submitted the order will be informed of a "nothing done". The MM should assume that FASTMATCH will DK any fills for that order
Out	**Don't Know Trade**	q	Application rejection message sent to a MM in reply to any Trade message sent after a FASTMATCH Order Timeout message has been issued. Once an Order Timeout has occurred, Trade message for that order in question will be rejected whether for a fill or a reject of the original order. No response to DK is required from MM

5 IMPLEMENTATION

5.1 MESSAGE ENCRYPTION

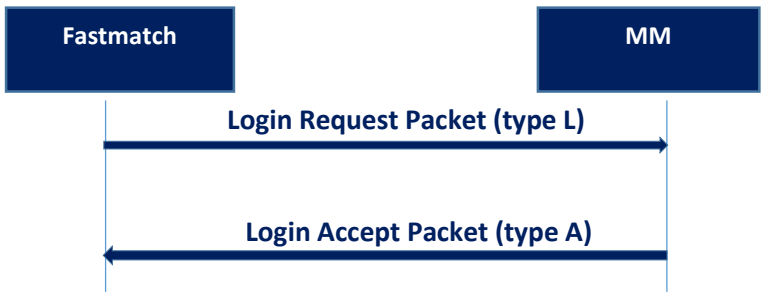
Encryption of binary messages themselves is not supported.

5.2 ESTABLISH CONNECTION / DISCONNECTION FOR TCP

5.2.1 CONNECT

Connection to the system is initiated by the client issuing a Login Request Packet.

If the connection can be accepted, then FASTMATCH will reply with Login Accept Packet, otherwise Login Reject Packet will be sent.



5.2.2 DISCONNECT

Closing of a connection is initiated by sending a Logout packet to the opposite party. At the end of trading day server sends End of Session message, followed by Logout. Additionally, client and server exchange heartbeats. If server misses at least 2 heartbeats from the client, the client is disconnected. Client later will have to re-subscribe to resume market data updates.

Client is expected to send unsubscribe messages if he no longer wishes to receive market data in specified symbol/type. If client sends Logout Request all subscription (that belong to specified SessionID) are cancelled.

5.3 FIELD TYPES

5.3.1 BYTE FIELD

8bit Signed Integer number (character interpreted as integer as per ASCII table)

5.3.2 SHORT FIELD

16bit Signed Integer number

5.3.3 INTEGER FIELD

32bit Signed Integer number. All integers are signed.

5.3.4 LONG FIELD

64bit Signed Integer Number

5.3.5 QUANTITY FIELD

Amounts are using 64bit Long numbers in little-endian format (unless client requests all integers to be in network order). Long value is scaled 2 decimal places to the right (multiplied by 100). For example, amount: 10000.12 should be sent as long number: 1000012. If quantity has no fractional part, it still has to be multiplied by 100. For example: 10000 should be sent as 1000000.

5.3.6 RATE FIELD

Rates are using 32bit Integer number. Integer value is scaled 5 decimal places to the right (multiplied by 100000 (100K)). For example 1.2345 should be sent as long number: 123450. 1200.01 should be sent as 120001000.

5.3.7 TIMESTAMPS

In FMBinaryStream header, timestamp is a 64bit Integer number, containing number of microseconds from Jan 1, 1970 GMT. In FMBinaryStream messages, timestamp field is a 64bit long containing microseconds from Jan 1, 1970 GMT. Example of such field is TransactTime.

5.3.8 DATE FIELD

Date is sent as 32bit Integer value containing seconds from Jan 1, 1970 GMT. Examples of such fields are TradeDate and SettlementDate.

5.3.9 ALPHANUMERIC FIELD

Contains alphanumeric characters.

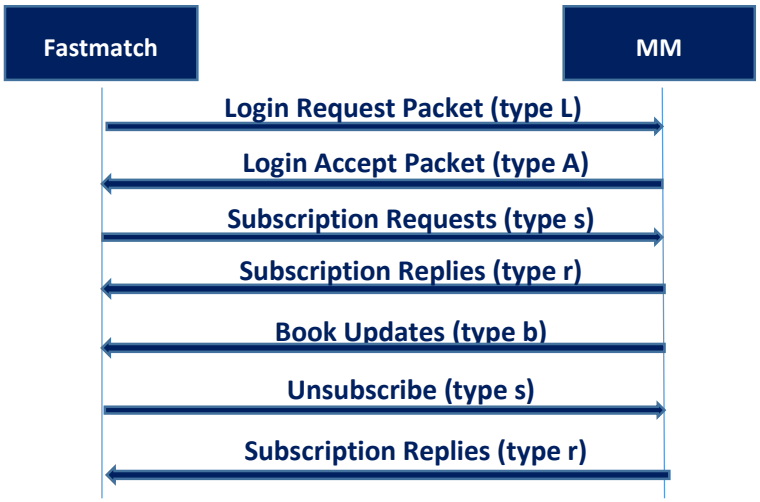
Currency pair field in all business messages is 8 byte long, from which the first seven are used.

Currency pairs are formed by concatenating the ISO currency codes (ISO 4217) of the base currency and the counter currency, separating them with a slash character in a CC1/CC2 format. The eighth byte can have any value e.g.

E	U	R	/	U	S	D	anything

5.4 MARKET DATA

The following diagram describes steps required to receive market data.



5.1 STREAMING RATE REQUEST

Upon a quote session being established, FASTMATCH sends market data subscription messages to the MM. Each Market Data Request message contains one symbol. MM should not start streaming any quotes until FASTMATCH sends market data subscriptions.

Upon accepting a market data subscription, a MM should immediately begin sending its streaming market data snapshots. If a market data subscription cannot be fulfilled, the MM should return a Market Data Request Reject message.

5.2 BASE CURRENCY QUOTING CONVENTION

In all application messages, the Symbol field defines the currency pair, for example EUR/USD.

FASTMATCH follows the International Organization for Standardization (ISO) currency pair symbol convention of BASE/TERM or CCY1/CCY2. Rates are expressed as one unit of a BASE currency in units of the TERM currency. For example, a EUR/USD rate = 1.3675 means 1.3675 units of USD per one unit of EUR.

MM's must send quotes expressed in this convention. If a quote cannot be provided in the specified convention, the MM should reject the Market Data Request.

5.3 SEQUENCE NUMBERS

SoupBinTCP packets use implicit sequence numbers. Both parties have to increment their relevant sequence numbers. TCP protocol handles packet loss/recovery and sequence number gap should never occur on the client side. In case of unrecoverable gap, session should be closed and reconnected.

Sequence number is included in each BookUpdate message. If client detects sequence gap, client can clear the corresponding book and start building it again, applying new price updates and ignoring price cancel messages for non-existent update IDs. Note that alternatively, client can clear all books as soon as gap is detected in datagram header sequence number. Client can choose the behavior that better fits their needs.

5.4 SUPPORTED MARKET DATA ENTRY UPDATE TYPES

The following entry update types are supported:

- ‘1’ = Book

5.5 SUPPORTED SUBSCRIPTION ACTION TYPES

The following subscription types are supported:

- ‘1’ = Subscribe
- ‘2’ = Unsubscribe

5.6 SUPPORTED BOOK DEPTH

The following subscription types are supported:

- 0 = Full Book (will be limited to the maximum number of levels configured for the client)
- 1 = Top of the book
- N = Number of levels (will be limited to the maximum number of levels configured for the client)

5.7 SUBSCRIPTION REQUEST ID

Each subscription must provide request ID unique at least per instrument symbol. For example client can send 2 subscription requests: one for Book updates in EUR/USD, the other for Trade updates in EUR/USD. These 2 should have different subscription IDs, so that client can correlate subscription responses with original subscription requests using Request ID and Instrument Symbol. In order to correlate only by RequestID, client should make sure each subscription has a unique request ID.

5.8 INSTRUMENT SYMBOL AND INSTRUMENT ID

All updates are sent using 2 bytes instrument ID instead of instrument symbol string. MM provides Instrument IDs in Subscription Responses. Although InstrumentID is of type Short, spot currencies (and metals) are guaranteed to have InstrumentID. Clients can take advantage of that and utilize direct access table to quickly look up their relevant books/instruments by InstrumentID. Non-Spot instruments may have higher InstrumentID numbers.

Also note that Instrument IDs are the same throughout the entire trading day. Instrument IDs are not guaranteed to be the same on the next trading day.

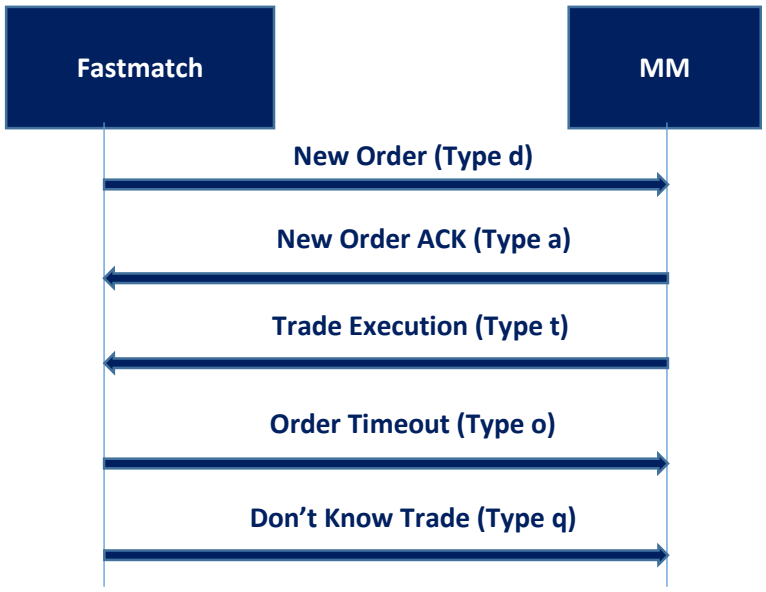
5.9 ORDERS

To place an order the client should submit a New Order message. The following is an example of a trade sequence.

5.1 ORDER INPUT AND EXECUTION

When a client attempts to deal on a MM quote, a New Order message is sent by FASTMATCH to the MM. Only OrdType “D” for a previously quoted foreign exchange order, are supported. All MMs must be able to support Limit FOK orders. This is the FASTMATCH default. For MMs that can support Limit IOC orders, this order type can be enabled. A MM should send a New Order ACK message immediately in response to a New Order message. MM sends a Trade (‘T’) message to report the fill of a New Order message or partial fill.

Trade Sequence Diagram



5.2 DELAYED EXECUTION REPORT

A MM must respond immediately with an Execution Report to New Order – Single messages. If an Execution Report is not received within a fixed number of milliseconds (set per Market Maker), the order will be considered "OUT", and the MM will be sent an Order Timeout message. FASTMATCH will respond to any Execution Report received after the Order Timeout has been sent with a "Don't Know Trade" message, indicating the Execution Report cannot be processed.

FASTMATCH business rules state that Order Timeouts are always reported as "rejected" or "not done" to the client. MM should establish procedures to close their own exposure should an Order Timeout occur for a trade they would have reported as filled. Neither FASTMATCH nor its clients will be responsible for executions received after the Order Timeout message is sent.

5.3 SUPPORTED ORDER TYPES

Only the 'Limit' order type (OrderType) is supported:

- 2 = Limit - At order to buy or sell at a specific price or better.

5.4 ORDER TIME IN FORCE

The 'TimeInForce' is used to set the expiration of the order. The following values are supported:

- 2 - IOC, Immediate Or Cancel
- 3 - FOK, Fill Or Kill

5.5 ORDER EXECUTION NOTIFICATIONS

FASTMATCH will send a series of Execution Report messages to the client to provide notification regarding order execution.

Trade Report messages are used to indicate:

- Order accepted
- Order partially filled
- Order filled
- Order cancelled
- Order rejected

5.6 CANCEL ON DISCONNECT (COD)

Cancel on Disconnect (COD) monitors the ECN system for involuntary lost connections between users and the ECN platform. If a lost connection is detected, COD cancels all resting Orders for the disconnected session for the registered ECN user. It is the user's responsibility to reenter all orders that have been cancelled by COD.

6 SESSION MESSAGE DEFINITIONS

Each message starts with a SoupBinTCP Session Level Header.

6.1 LOGICAL PACKET STRUCTURE

The SoupBinTCP client and server communicate by exchanging a series of logical packets. Each SoupBinTCP logical packet has:

A. a two byte little-endian length that indicates the length of rest of the packet (meaning the length of the payload plus the length of the packet type – which is 1)
B. a single byte header which indicates the packet type
C. a variable length payload

SoupBinTCP Logical Packet Structure

SoupBinTCP Header		Body
Two Byte Packet Length	Packet Type	Variable-length payload

For all cases when field length is one byte a corresponding ASCII numeric value of the character can be used. E.g. a printable value 'L' can be represented as one byte decimal 76

6.2 LOGIN REQUEST PACKET

PacketType = 'L'

Name	Offset	Length	Value	Description
PacketLength	0	2	Integer	Packet Length - Number of bytes after this field until the next packet
PacketType	2	1	'L'	Log on
Version	3	2	Integer	= 1 for current version

Username	5	6	Alpha	Username
Password	11	10	Alpha	Password
Session	21	10	Alphanumeric	Specifies session that client wants to log into or all blanks to log into current session
NextSeqNum	31	20	Numeric	Next sequence number in ASCII that client wishes to receive or 0 to start receiving most recent message.

6.3 LOGIN ACCEPT PACKET

Fastmatch will send out Login Accepted Packet in response to Login Request from the client.

PacketType = 'A'

Name	Offset	Length	Value	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	'A'	Login Accepted
Session	3	10	Alphanumeric	SessionID
SequenceNum	13	20	Numeric	Next sequence number to be sent in ASCII

6.4 LOGIN REJECT PACKET

PacketType = 'J'

Name	Offset	Length	Value	Description
PacketLength	0	2	Integer	Number of bytes after this field until the next packet
PacketType	2	1	'J'	Login Rejected
Reject Reason	3	1	Byte	Reject codes: 'A' - Not authorized 'S' - Session not available 'V' – Invalid Version

6.5 SEQUENCED DATA PACKET

PacketType = 'S'

Name	Offset	Length	Value	Description

PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	'S'	Sequenced data packet
Message	3	Variable	Any	Ouch Outgoing Packet

6.1 SERVER HEARTBEAT

PacketType = 'H'

Name	Offset	Length	Value	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	'H'	Server Heartbeat

6.2 CLIENT HEARTBEAT

PacketType = 'R'

Name	Offset	Length	Value	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	'R'	Client Heartbeat

6.3 END OF SESSION

PacketType = 'Z'

Name	Offset	Length	Value	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	'Z'	End of Session

6.4 UNSEQUENCED DATA PACKETS (INCOMING OUCH MESSAGES)

PacketType = 'U'

Name	Offset	Length	Value	Description
PacketLength	0	2	Integer	Packet Length - Number of bytes after this field until the next packet

PacketType	2	1	“U”	End of Session
Message	3	Variable	Alphanumeric	

6.5 CLIENT LOGOUT

PacketType = ‘O’

Name	Offset	Length	Value	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	‘O’	Client Logout

7 BUSINESS MARKET DATA MESSAGE DEFINITIONS

Market Data Business message has the following logical structure:

Session Level Header	FM ITCH Header	Message Blocks

7.1 FM MD HEADER

Payload of each message will always start with MD header (10 bytes), containing timestamp, StreamID to which all subsequent updates belong, as well as MsgBlockCount with number of message blocks following this header. Each message block has a business message type.

Field	Offset	Length	Type	Description
Timestamp	3	8	Long	Time in microseconds from Epoch (Jan 1, 1970)
StreamID	11	1	Byte	The ID of logical stream to which messages in a block belong (for multi-stream trading only)
MsgBlockCount	12	1	Byte	Number of message blocks following this header

7.2 BOOK UPDATE

MessageType: ‘b’

All add price updates and cancel price updates will be preceded with BookUpdate. BookUpdate contains instrument ID, number of price updates in specified instrument, as well as sequence number of this book update. BookUpdate should be followed by ‘UpdateMsgCount‘ of PriceAdd and PriceCancel messages.

Field	Offset	Length	Type	Description

PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	"S"	Sequenced Data
Timestamp	3	8	Long	Time in microseconds from Epoch (Jan 1, 1970)
StreamID	11	1	Byte	The ID of logical stream to which messages in a block belong (for multi-stream trading only)
MsgBlockCount	12	1	Byte	Number of message blocks following this header
Type	13	1	Byte	Type 'b'
InstrumentID	14	2	Short	Instrument ID of this book update
UpdateMsgCount	16	1	Byte	Number of updates in specified instrumentID, following this update header
SequenceNumber	17	4	Integer	Sequence number of update for this book update

7.3 PRICE ADD

MessageType: 'p'

PriceAdd must always be sent as part of BookUpdate.

Field	Offset	Length	Type	Description
Type	21	1	Byte	Type 'p'
PriceUpdateID	22	4	Integer	Price update unique per session per instrument
Quantity	26	8	Long	Quantity x100
MinQuantity	34	8	Long	Minimum Quantity x100
Rate	42	4	Integer	Rate x100000
Side	46	1	Byte	Side '1' – Bid, '2' - Offer

7.4 PRICE CANCEL

Message Type 'x'

PriceCancel must always be sent as part of BookUpdate.

Field	Offset	Length	Type	Description
Type	21	1	Byte	Type 'x'
PriceUpdateID	22	4	Integer	Price update ID of a previously quoted and active price

7.5 SUBSCRIPTION REQUEST

MessageType: ‘s’

Field	Offset	Length	Type	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	‘S’	Sequenced Data
Timestamp	3	8	Long	Time in microseconds from Epoch (Jan 1, 1970)
StreamID	11	1	Byte	The ID of logical stream to which messages in a block belong (for multi-stream trading only)
MsgBlockCount	12	1	Byte	Number of message blocks following this header
Type	13	1	Byte	Type ‘s’
InstrumentSymbol	14	12	Alphanumeric	String symbol for instrument (e.g. EUR/USD). Should be right padded with zeros.
RequestID	26	4	Integer	Subscription Request ID
SessionID	30	10	Alphanumeric	Session ID assigned at Login
Action Type	40	1	Byte	’1’ – Subscribe, ‘2’ – Unsubscribe
SubscriptionDepth	41	1	Byte	Subscription Depth (‘0’ – full book, ‘1’ – top level, ‘N’ – level count) *

*During unsubscribe request, subscription type and depth are ignored.

7.6 SUBSCRIPTION RESPONSE

MessageType: ‘r’

Field	Offset	Length	Type	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	‘S’	Sequenced Data
Timestamp	3	8	Long	Time in microseconds from Epoch (Jan 1, 1970)
StreamID	11	1	Byte	The ID of logical stream to which messages in a block belong (for multi-stream trading only)
MsgBlockCount	12	1	Byte	Number of message blocks following this header
Type	13	1	Byte	Type ‘r’
InstrumentSymbol	14	12	Alpha	String symbol for instrument (e.g. EUR/USD). Should be right padded with zeros.
InstrumentID	26	2	Short	Instrument ID (immutable for entire trading day)

RequestID	28	4	Integer	Subscription Request ID
SessionID	32	10	Alphanumeric	Session ID assigned at Login
Status	42	1	Byte	'1' – Accepted, '2' – Rejected
ErrorCode	43	1	Byte	See list of reject codes in Appendix

7.7 REJECT

MessageType: 'J'

Field	Offset	Length	Type	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	Byte	Sequenced 'S' or Unsequenced'U'
Timestamp	3	8	Long	Time in microseconds from Epoch (Jan 1, 1970)
StreamID	11	1	Byte	The ID of logical stream to which messages in a block belong (for multi-stream trading only)
MsgBlockCount	12	1	Byte	Number of message blocks following this header
Type	13	1	Byte	Type 'J'
MessageType	14	1	Byte	Type of rejected message
RejectCode	15	2	Short	Reject Code
RejectMessage	17	20	Alpha	Reject String

7.8 MARKET DATA ERROR CODES

Error	Code	Description
NoError	'0'	No Error
InvalidMessageType	'1'	Invalid message type
InvalidMessageLength	'2'	Invalid message length
InvalidInstrument	'3'	Invalid Instrument
UnauthorizedDestination	'4'	Destination not authorized (logged in)
OtherError	'Z'	Any other error

7.9 SUBSCRIPTION REJECT REASON CODES

Error	Code	Description
NotAuthorized	A	Not authorized to subscribe
UnsupportedInstrument	S	Instrument not supported
UnsupportedVersion	V	Unsupported version number
DuplicateRequestID	D	Duplicate request ID
SubscriptionAlreadyActive	P	Subscription is already in progress
ExchangeNotOpen	E	Exchange not accepting subscriptions temporarily
InvalidActionType	F	Invalid subscription action
InvalidSubscriptionType	G	Invalid subscription type
InvalidDepth	H	Unsupported subscription depth
UnauthorizedSession	J	Subscription is not authorized

8 BUSINESS TRADE MASSAGES DEFINITIONS

Market Data Business message has the following logical structure:

Session Level Header	FM TRADE Header	Message Blocks

All TRADE messages include Session Level Header (SoupBinTCP) and the following FM TRADE header.

8.1 FM TRADE HEADER

Field	Offset	Length	Type	Description
MessageType	3	1	Byte	Example: ‘D’
Timestamp	4	4	Integer	Time in milliseconds from 17:00pm Eastern Time – value date rollover time
StreamID	8	1	Byte	The ID for the stream (for multi-stream trading only)

8.2 NEW ORDER

MessageType: ‘d’ Unsequenced

Field	Offset	Length	Type	Description

PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	'U'	Unsequenced Data
MessageType	3	1	Byte	Example: 'd'
Timestamp	4	4	Integer	Time in milliseconds from 17:00pm Eastern Time – value date rollover time
StreamID	8	1	Byte	The ID for the stream (for multi-stream trading only)
ClOrderId	9	4	Integer	Client Order ID
CcyPair	13	8	Alpha	Example: CC1/CC2
OrderType	21	1	Byte	'2' – Limit
Side	22	1	Byte	'1' – Buy, '2' – Sell
Quantity	23	8	Long	Order Size (x100)
MinQty	31	8	Long	Minimum Order Size (x100)
Rate	39	4	Integer	Order Price (x100000)
TimeInForce	43	1	Byte	'2' – IOC, '3' – FOK
QuoteID	44	4	Integer	This may contain a PriceUpdateID from a specific Price Add to be traded against
ContraCliID	48	2	Short	This may contain a ID of an order initiator, 0 otherwise
ContraBroker	50	4	Alpha	FM Contra Broker Codes (See Fastmatch contra broker codes)

8.3 NEW ORDER ACK

MessageType: 'a' Sequenced

Field	Offset	Length	Type	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	'S'	Sequenced Data
MessageType	3	1	Byte	Example: 'a'
Timestamp	4	4	Integer	Time in milliseconds from 17:00pm Eastern Time – value date rollover time
StreamID	8	1	Byte	The ID for the stream (for multi-stream trading only)
ClOrderId	9	4	Integer	Client Order ID
CcyPair	13	8	Alpha	Example: CC1/CC2
OrderID	21	8	Long	123456
AckStatus	29	1	Byte	'1' – Accept, '2' – Reject

ErrorCode	30	1	Byte	See Full List of Codes

8.4 ORDER CANCELED

MessageType: ‘c’ Sequenced

Field	Offset	Length	Type	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	‘U’	Unsequenced Data
MessageType	3	1	Byte	Example: ‘c’
Timestamp	4	4	Integer	Time in milliseconds from 17:00pm Eastern Time – value date rollover time
StreamID	8	1	Byte	The ID for the stream (for multi-stream trading only)
ClOrderID	9	4	Integer	Client Order ID
OrderID	13	8	Long	OrderID assigned by ECN
CcyPair	21	8	Alpha	Example: CC1/CC2

8.5 TRADE

MessageType: ‘t’ Sequenced

Field	Offset	Length	Type	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	‘S’	Sequenced Data
MessageType	3	1	Byte	Example: ‘t’
Timestamp	4	4	Integer	Time in milliseconds from 17:00pm Eastern Time – value date rollover time
StreamID	8	1	Byte	The ID for the stream (for multi-stream trading only)
ClOrderID	9	4	Integer	Client Order ID
CcyPair	13	8	Alpha	Example: CC1/CC2
FillQty	21	8	Long	1234567 (x100)
FillRate	29	4	Integer	Fill Price (x100000)
Side	33	1	Byte	‘1’ – Buy, ‘2’ - Sell
ExecID	34	64	Alphanumeric	FX:123456:122337234:B
LeavesQty	98	8	Long	Remaining (Unfilled) Qty

Account	106	8	Alpha	Trade Account
TransactTime	114	8	Timestamp	Milliseconds in GMT from Jan 1, 1970
SettlDate	122	4	Date	Seconds in GMT from Jan 1, 1970
TradeDate	126	4	Date	Seconds in GMT from Jan 1, 1970

8.6 ORDER TIME OUT

MessageType: 'o' Unsequenced

If no Trade Report is received within "x" milliseconds (configured per MM), an Order Time Out message is sent to the price provider. At this time FASTMATCH will consider this trade as "Nothing Done" and will release the order on the client side. Should the price provider have executed the trade, they will need to take action on their end to close the position as neither FASTMATCH nor the client will be responsible. "x" is presently defaulted to 50 milliseconds, but this is a configurable value per market maker.

Field	Offset	Length	Type	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	'U'	Unsequenced Data
MessageType	3	1	Byte	Example: 'o'
Timestamp	4	4	Integer	Time in milliseconds from 17:00pm Eastern Time – value date rollover time
StreamID	8	1	Byte	The ID for the stream (for multi-stream trading only)
ClOrdID	9	4	Integer	Unique order identifier assigned by FASTMATCH.

8.7 DON'T KNOW TRADE (DK)

MessageType: 'q' Unsequenced

If a price provider sends an Execution Report for a trade in an Order Time Out state, FASTMATCH will respond with a DK message. This is informational.

Field	Offset	Length	Type	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	'U'	Unsequenced Data
MessageType	3	1	Byte	Example: 'q'
Timestamp	4	4	Integer	Time in milliseconds from 17:00pm Eastern Time – value date rollover time

StreamID	8	1	Byte	The ID for the stream (for multi-stream trading only)
ClOrderID	9	4	Integer	Client (FM) Order ID on missing execution
ExecID	13	64	Integer	Execution ID of missing execution
DKReason	77	1	Byte	'A' - Unknown Symbol 'B' - Invalid Side 'C' - Quantity Exceeds Order 'E' - Price Exceeds Limit 'F' - Settlement Date Mismatch 'G' - Min Qty Violation 'O' - No Matching Order (also for Timed Out Orders) 'Z' - Other
CcyPair	78	8	Alpha	Example: EUR/USD
Side	86	1	Byte	'1' – Buy, '2' - Sell

8.8 REJECT

MessageType: 'J' Sequenced

Field	Offset	Length	Type	Description
PacketLength	0	2	Integer	Packet Length- Number of bytes after this field until the next packet
PacketType	2	1	Byte	Sequenced or Unequenced Data
MessageType	3	1	Byte	Example: 'J'
Timestamp	4	4	Integer	Time in milliseconds from 17:00pm Eastern Time – value date rollover time
StreamID	8	1	Byte	The ID for the stream (for multi-stream trading only)
MessageType	9	1	Byte	Type of rejected message
RejectCode	10	2	Short	Reject Code
RejectMessage	12	20	Alpha	Reject String

8.9 TRADE ERROR CODES

Error	Code	Description
NoError	0	

DuplicateOrderId	1	Duplicate ClOrdId from client
ExchangeNotOpen	2	Exchange not accepting orders temporarily
InvalidQty	3	Invalid Quantity
InvalidRate	4	Invalid Rate
UnknownStreamId	5	StreamId cannot be mapped to internal clientId
InvalidAccount	6	Invalid account field
UnknownSymbol	7	Symbol not supported
InvalidSide	8	Invalid Side
InvalidMinQty	9	Invalid MinQty (wither < 0 or > OrderQty)
UnsupportedTIF	A	Invalid time in force
InvalidMaxShow	B	Invalid max show (< 0 or > OrderQty)
InvalidType	E	Invalid order type
OrderNotFound	F	Order not found by ClOrdId
TooLateToCancel	G	Order already filled or cancelled
SymbolDisabled	H	Symbol temporarily disabled for trading
TooManyOpenOrders	I	Client has too many open orders
TradingSuspended	J	Client session is suspended
OtherError	Z	Any other error

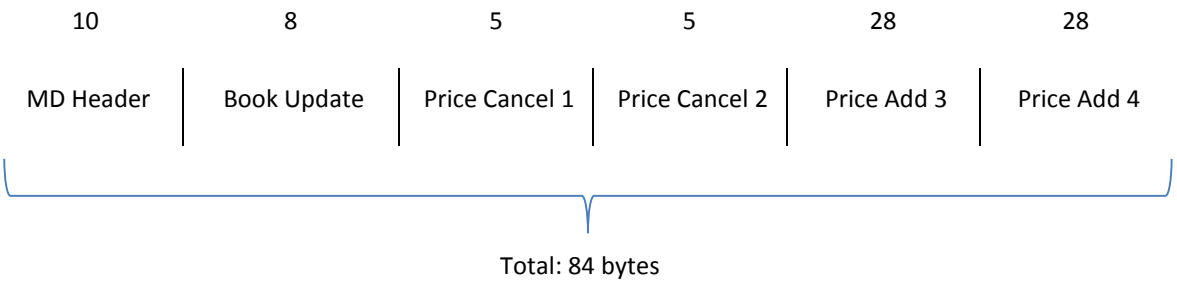
8.10 REJECT REASON CODES

Error	Code	Description
NotAuthorized	A	Access is not authorized for used credentials
SessionNotAvailable	S	Session is not available
InvalidVersion	V	Invalid version number

9 MESSAGE EXAMPLES

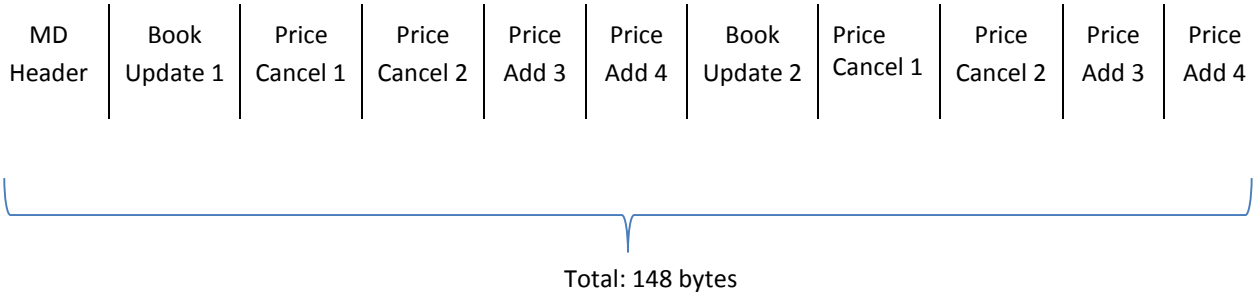
9.1 BOOK UPDATE MESSAGE EXAMPLE

The following is an example of a sample book update with two quotes cancelled and two new quotes added:



The above update will take 84 bytes (excluding SoupBinTCP header).

Example of similar book updates for two different currencies:



The above update will take 148 bytes (excluding SoupBinTCP header).

10 APPENDIX

10.1 FASTMATCH CONTRA BROKER CODES

Company Name	Settlement Location	SWIFT	Broker Code
ABN Amro Clearing Bank NV	Amsterdam	FBGC NL 21	AACB
ABN Amro Bank NV	Amsterdam	ABNA NL 2A	ABNA
Australia and New Zealand Banking Group Limited	Melbourne	ANZB AU3M	ANZM
Bank of America NA	San Francisco	BOFA US 6S	BASF
Bank of New York Mellon	New York	IRVT US 3N	BNYN
Bank of New York Mellon	London	MELN GB 2C	BNYL
Bank of New York Mellon	Tokyo	IRVT JP JX	BNYQ
Bank of New York Mellon	Singapore	IRVT SG SX	BNYS
Bank of New York Mellon	Seoul	IRVT KR SX	BNYK
Barclays Bank plc	London	BARC GB 2L	BARC
BNP Paribas	Paris	BNPA FR PP	BNPP
Citibank NA	London	CITI GB 2L	CITL
Commerzbank AG	Frankfurt	COBA DE FF	CBKF
Credit Suisse AG	Zurich	CRES CH ZZ	CSSZ
Credit Suisse International	London	CSFP GB 2L	CSIL
Deutsche Bank AG	London	DEUT GB 2L	DEUL
Goldman Sachs & Co.	New York	GOLD US 33	GSFX
J.Aron & Company	New York	ARON US 33	ARON
HSBC Bank plc	London	MIDL GB 2L	HSBL
HSBC Bank USA NA	New York	MRMD US 33	HSBN
The Hongkong & Shanghai Banking Corporation Ltd	Hong Kong	HSBC HK HH	HSBK
ING Bank NV	Amsterdam	INGB NL 2A	INGA
Jefferies Bache Financial Services Inc.	New York	PBFS US 33	JBFS
JPMorgan Chase Bank NA	New York	CHAS US 33	JPMN
Morgan Stanley & Co. llc	New York	MSNY US 33	MSFX
Nomura Bank International plc	London	NBIL GB 2L	NOML
Royal Bank of Canada	London	ROYC GB2L	RBCL
Royal Bank of Canada	Toronto	ROYC CAT2	RBCT
The Royal Bank of Scotland plc	London	RBOS GB 2L	RBSL
Société Générale	Paris	SOGE FR PP	SOGP
UBS AG	Zurich	UBSW CH ZH	UBSZ
Westpac Banking Corporation	Sydney	WPAC AU 2F	WBCA