

Market Data Platform FIX/FAST

Core Functionality

Version: 1.6
10/10/08

Introduction	1
Architecture	2
System Architecture Overview	2
Market Data Group	2
Incremental UDP Feed A and B	2
Market Recovery (UDP).....	3
Instrument Definition (UDP).....	3
Replay (TCP - Historical)	3
Services - Template Dissemination and Market Data Configuration	3
FTP Site Information - Template Dissemination and Market Data	
Configuration.....	4
System Startup	5
Pre-Opening Startup.....	5
Late Joiner Startup	5
FAST Implementation	7
Introduction	7
Stop Bit Encoding	7
FAST 7-Bit Binary Representation.....	7
Implicit Tagging.....	7
Field Encoding Operators	8
Dictionary Context	8
Field Operators	8
Data Types	9
FAST Template.....	9
Presence Map and Stop Bit	9
Presence Map Rules	10
Message Structure.....	10
Template ID Usage.....	11
Optional vs. Mandatory Fields	11
XML Template Example	12
FAST Decoding.....	13
Transfer Decoding Overview	14
Field Decoding Overview	15
When to Reset Decoder State	15
Receiving Data over MDP	15
Decoding a Packet.....	15
Decoding Messages in a Packet	15
Error Handling in a Broadcast Environment	16
Decoding Sequence.....	16
Transfer Decoding	16
Decoding Process.....	18
Build the FIX Message.....	19
Result - Decoded Values	21
Sample Template.....	22
Template Overview	25
XML Template Example.....	25
Template Implementation Considerations	26

Template Distribution	29
Template Modification	29
Incremental Book Management.....	31
Overview	31
Incremental Book Management Applicable FIX Message Structures.....	31
Common Book Update Tags	32
Central Limit Order Book	32
Book Management Mechanics - Multiple-Depth Book	33
Overview	33
Basic Book Update Data Block.....	34
Examples	35
Quantity on Buy Side Modified	35
Entire Order Canceled and New Order Entered	36
New Order Entered at Same Price	38
New Best Price Entered.....	39
Book Management Mechanics - Implied Book.....	40
Basic Book Update Data Block.....	41
Examples	41
Quantity on Buy Side Modified	42
Entire Order Canceled and New Order Entered	43
New Order Entered at Same Price	45
Consolidating Implied and Multiple-Depth Books into a Single Book	46
Book Management Mechanics - Top-of-Book.....	47
Overview	47
Basic Book Update Data Block.....	47
Indexing the Entry	48
Examples	48
Quantity on Buy Side Modified	48
Entire Order Canceled	49
New Order Entered	50
New Order Entered at Same Price	51
Book Management Mechanics - Indicative Prices	52
Overview	52
Basic Book Update Data Block.....	52
Indexing the Entry	53
Examples	53
Quantity on Buy Side Modified	53
Entire Indicative Price Canceled	54
New Indicative Price Entered.....	55
New Indicative Price Entered at Same Price	56
Real Time Statistics (Market Behavior Events).....	57
Pre-Opening Statistics	57
Last Best Price.....	59
Trade.....	60
Example 1 - Two Outright Orders Trading	60
Example 2 - Two Spread Orders Trading (Not Implied)	61
Example 3 - Spread Order Trading Against Two Outright Orders (Implied)	64

Example 4 - Butterfly Order Trading Against a Calendar Order and Two Outright Orders (Implied)	67
Session High and Low Trade Price.....	72
Best High Bid and Best Low Ask	73
Closing and Settlement Price.....	75
CME Globex Pricing.....	76
Tick Convention	76
Calculating Tick Size for a VTT Eligible Instrument Using the Security Definition (tag 35-MsgType=d) Message	76
Variable Tick Table	78
Calculating Tick Size for a Non-VTT Eligible Instrument Using the Security Definition (tag 35-MsgType=d) Message	79
Display	79
Recovery.....	80
Recovery Feeds.....	81
TCP Replay Overview	81
Implementation Considerations	82
Process.....	83
Market Recovery Overview.....	85
Instrument Replay Overview.....	86
Using the Incremental Market Data Feed to Determine State	86
Instrument Level Sequencing	87
Natural Refresh.....	88
Recovering Data - Process	92
Large Scale Outage Using Market Recovery - Queuing.....	93
Large Scale Outage Using Market Recovery - Concurrent Processing.....	93
Small Scale Data Recovery Using TCP Replay - Queuing.....	94
Small Scale Data Recovery Using TCP Replay - Concurrent Processing.....	95

1. Introduction

This document contains information on core features and functionality for Market Data Platform FIX/FAST.

Refer to the following sections for detailed information:

- “Architecture” on Page 2
- “FAST Implementation” on Page 7
- “Template Overview” on Page 25
- “Incremental Book Management” on Page 31
- “Real Time Statistics (Market Behavior Events)” on Page 57
- “CME Globex Pricing” on Page 76
- “Recovery” on Page 80

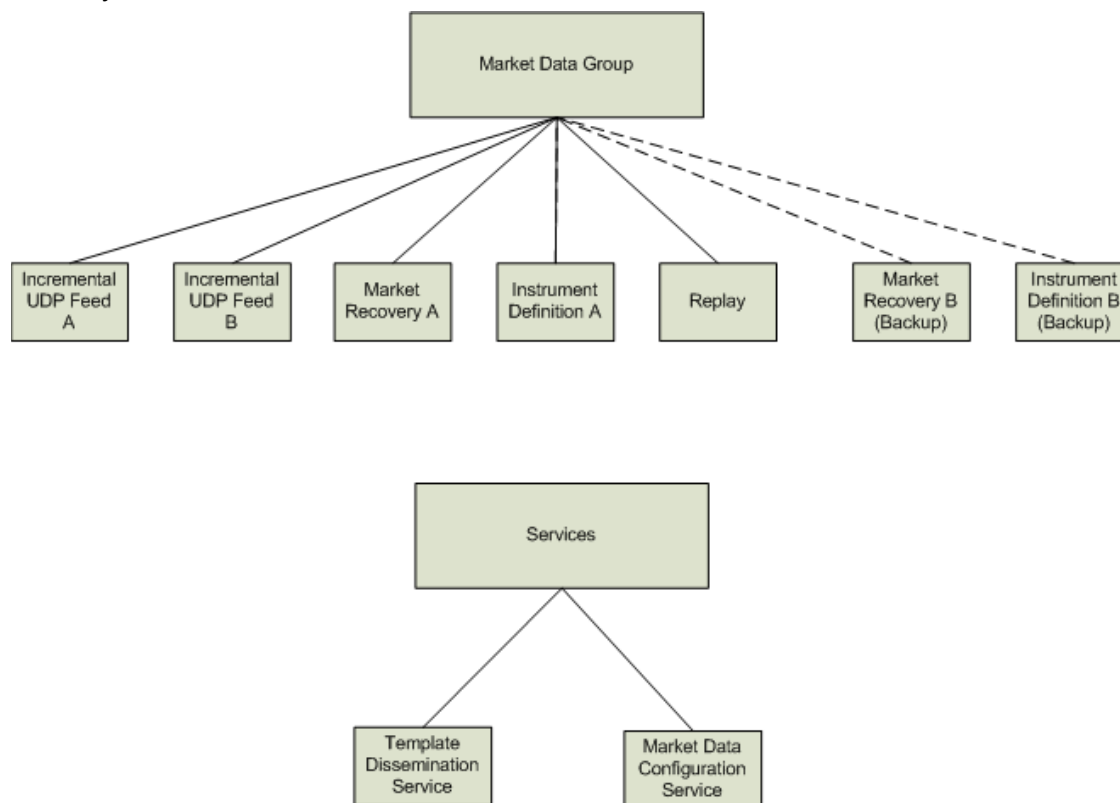
2. Architecture

This section contains an architecture overview for Market Data Platform FIX/FAST.

2.1 System Architecture Overview

This section contains a high level look at the production environment.

Figure 1. System Architecture



2.1.1 Market Data Group

Market Data Platform FIX/FAST architecture is explained in this section in terms of a market data group. A market data group, is a set of UDP channels used to produce market data messages for a set of instruments and / or a set of instrument groups.

2.1.1.1 Incremental UDP Feed A and B

UDP Feed A and UDP Feed B are used to disseminate CME Group incremental market data using bandwidth efficient FAST encrypted FIX messages. All FIX message types are sent through UDP Feed A and UDP Feed B applicable Market Data Groups (book update, statistics, quotes, instrument definitions, trades, instrument / instrument group, CME Globex status). Please note that a single FIX/FAST message can contain multiple updates for multiple instruments.

2.1.1.2 Market Recovery (UDP)

Market Recovery (UDP) Feed A is used to disseminate CME Group market data snapshots for all books with any activity since the beginning of the week. Market Recovery (UDP) Feed B functions as a backup in the event that Feed A becomes inoperable. Feed A and Feed B should not be used for arbitration. Book updates (limit and implied) and statistics are sent through the applicable market data channel. A single FIX/FAST message contains the market state for a given instrument. Snapshots are replayed at a constant flow of configurable TPS. Expired instruments are included on the Market Recovery feed.

Note: CME strongly recommends that the Market Recovery feeds be used for recovery purposes only. Once client systems have retrieved recovery data, client systems should stop listening to the Market Recovery feeds.

2.1.1.3 Instrument Definition (UDP)

Instrument Definition (UDP) Feed A is used to disseminate CME Group instrument definitions (equivalent of the MO / MU in RLC). A single FIX/FAST message contains the definition of a given instrument. Instrument definitions are replayed at a constant flow of configurable TPS. Expired instruments are not included on the Instrument Definition channel. Instrument Definition (UDP) Feed B functions as a backup in the event that Feed A becomes inoperable. Feed A and Feed B should not be used for arbitration.

2.1.1.4 Replay (TCP - Historical)

The TCP historical replay component allows you to request a replay of a set of messages already published on the UDP Incremental Market Data Channel or a snapshot message. The request specifies messages to be replayed. The request uses the FIX Market Data Request message (35=V) and responses can be the FIX Market Data Request Reject message (35=Y) or the list of messages to be replayed.

This type of request is sent through a new TCP connection established by the customer. The responses are sent by CME Group through this same connection and the connection is then closed by CME Group once the resend is complete. All responses are FIX/FAST encoded (including the reject response). Replay is limited in the number of messages that can be requested. This is not the preferred method for recovery (use market recovery).

2.1.2 Services - Template Dissemination and Market Data Configuration

There are two services available: Template Dissemination Service and Market Data Configuration Service.

Service	Description
Template Dissemination	FIX/FAST is a template based protocol. As a result, messages can only be interpreted using a template. Each message contains a unique Template ID that references the template to use to interpret the message. The template dissemination service provides a method for client systems to receive all of the CME active templates, or the templates associated with either a Template ID or a Market Data Group. For additional information on this, refer to "Template Distribution" on Page 29.
Market Data Configuration	The Market Data Configuration Service allows client systems to receive the list of all Market Data Channel configurations (multicast IP, product group, and Security ID). An FTP site is used for this service.

Note: FXMarketSpace client systems use a configuration file that is separate from the main configuration file.

2.1.2.1 FTP Site Information - Template Dissemination and Market Data Configuration

An FTP site is used to disseminate templates and market data configuration information. This FTP site contains both the template files and configuration files for all environments. The FTP site is a secure site that requires a user name and password for access. Template and market data configuration details for the production environment are only available to customers after the certification process is complete.

Information applies as follows in the table:

- Environment - specific environment (i.e. Certification, Production).

Note: The AutoCert+ tool will indicate which environment you need to connect to. For additional information on AutoCert+ access, refer to: [AutoCert+ Access Guide](#)

- Service - the Template or Configuration service.
- FTP Site - address of FTP site.
- User Name - identifies the username.
- Password - identifies the password.
- Directory Location - identifies the directory.
- Client System Update Schedule - Client systems should download updates according to the schedule specified.

Environment	Service	FTP Site	User Name	Password	Directory Location	Client System Update Schedule
Certification	Template	ftp.cmegroup.com	fixuser	r3@d0n1y	/Cert/Templates	Sunday prior to market open
Certification	Configuration	ftp.cmegroup.com	fixuser	r3@d0n1y	/Cert/Configuration	daily
Certification AutoCert+	Template	ftp.cmegroup.com	fixuser	r3@d0n1y	/CertAutoCertPlus/Templates	Sunday prior to market open
Certification AutoCert+	Configuration	ftp.cmegroup.com	fixuser	r3@d0n1y	/CertAutoCertPlus/Configuration	daily
New Release Certification	Template	ftp.cmegroup.com	fixuser	r3@d0n1y	/NRCert/Templates	Sunday prior to market open
New Release Certification	Configuration	ftp.cmegroup.com	fixuser	r3@d0n1y	/NRCert/Configuration	daily
New Release Certification AutoCert+	Template	ftp.cmegroup.com	fixuser	r3@d0n1y	/NRAutoCertPlus/Templates	Sunday prior to market open
New Release Certification AutoCert+	Configuration	ftp.cmegroup.com	fixuser	r3@d0n1y	/NRAutoCertPlus/Configuration	daily

Environment	Service	FTP Site	User Name	Password	Directory Location	Client System Update Schedule
Production	Template	ftp.cmegroup.com	fixuserprod	fxu38rpr0d	/Production/Templates	Sunday prior to market open
Production	Configuration	ftp.cmegroup.com	fixuserprod	fxu38rpr0d	/Production/Configuration	daily

Note: There is also an ftp site on the CME Group network that can be accessed as an alternative to the public ftp.cmegroup.com site. Please contact your account manager for additional information.

2.2 System Startup

This section contains a high level overview of the startup process for the production environment.

2.2.1 Pre-Opening Startup

For a startup prior to the weekly market open, all market data (book updates, statistics, quotes, instrument definitions, trades, instrument/instrument group, status) will be disseminated through the Incremental UDP Feed A and Feed B. Follow the process below to ensure that all necessary market data is received:

1. Listen to the Incremental feed for incremental market data and start normal processing.

2.2.2 Late Joiner Startup

For a late joiner startup, follow the process below to ensure that all necessary market data is received:

1. Download the configuration files and template files from the ftp site. Refer to “Services - Template Dissemination and Market Data Configuration” on Page 3 for more information.
2. Listen to the Instrument Definition feed.
3. Listen to the Incremental feed for incremental market data. Begin the natural refresh process and begin queuing messages.

Note: The incremental market data may complete a natural refresh (liquid instruments only) that would construct the current, correct state of a book. Refer to “Natural Refresh” on Page 88 for more information.

4. Listen to the Market Recovery feed for the latest snapshots.

Use the latest snapshot to verify that the book was correctly created via natural refresh and to retrieve the latest statistics. When the latest snapshots are received, the value for tag 369-LastMsgSeqNumProcessed in the snapshot matches tag 34-MsgSeqNum in the incremental feed message. Also, the value for tag 83-RptSeq in the snapshot matches tag 83-RptSeq in the incremental feed.

If the book for an instrument was not completely constructed using natural refresh, then apply the snapshot. Start discarding queued messages from the incremental feed until tag 34-MsgSeqNum in the message has the same value as tag 369-LastMsgSeqNumProcessed in the snapshot. The discarded messages contain information that was already included in the snapshot message.

Note: Information for instruments included for the first time in the latest incremental feed message (with 34-MsgSeqNum equal to tag 369-LastMsgSeqNumProcessed in the snapshot) may not be in the latest snapshot. Apply the latest incremental feed message to obtain this information.

5. Stop listening to the Market Recovery and Instrument Definition feeds.
6. Start normal processing.

3. FAST Implementation

This section describes how to implement FIX Adapted for STraming (FAST) protocol.

3.1 Introduction

The FIX Adapted for STraming (FAST) Protocol has been developed as part of the FIX Market Data Optimization Working Group. FAST is designed to optimize electronic exchange of financial data, particularly for high volume, low latency data dissemination. This document describes implementation of FAST in receiving and processing CME Group FIX/FAST-encoded electronic market data feed.

For more information see the FIX FAST (version 1.x.1) specification at:

<http://www.fixprotocol.org/documents/3066/FAST%20Specification%201%20x%201.pdf>

FAST is a data compression algorithm that significantly reduces bandwidth requirements and latency between sender and receiver. FAST works especially well at improving performance during periods of peak message rates. FAST extends the base FIX specification and assumes the use of FIX message formats and data structures. FAST is a standalone specification that uses templates to inform the receiver which operations to use in decoding. Templates allow FAST to achieve high levels of data compression with low processing overhead and latency compared to other compression utilities such as Zlib.

Note: This document describes concepts applicable to CME Group-specific FAST implementation; this document is supplementary to the FAST specification referenced above.

3.1.1 Stop Bit Encoding

Stop Bit encoding is a process incorporated in FAST that eliminates redundancy at the data field level by using a stop bit instead of the traditional separator byte. In FAST, a stop bit is used instead of FIX's traditional <SOH> separator byte. Thus 7 bits of each byte are used to transmit data and the eighth bit is used to indicate the end of a field.

3.1.1.1 FAST 7-Bit Binary Representation

FAST renders numbers into binary across the 7 data bits in each byte. Thus a number equal to or less than 2^7-1 , (127) occupies one byte, a number between 2^7 and $2^7*2 - 1$ (16,383), occupies two bytes, etc.

3.1.2 Implicit Tagging

In traditional FIX messages each field takes the form "Tag=Value<SOH>" where the tag is a number representing which field is being transmitted and the value is the actual data content. The ascii <SOH> character is used as a byte delimiter to terminate the field. For example:

35=x|268=3 (message header)

279=0|269=2|270=9462.50|271=5|48=800123|22=8 (trade)

279=0|269=0|270=9462.00|271=175|1023=1|48=800123|22=8|346=15 (new bid 1)

279=0|269=0|270=9461.50|271=133|1023=2|48=800123|22=8|346=12 (new bid 2)

FAST eliminates redundancy with a template that describes the message structure. This technique is known as implicit tagging as the FIX tags become implicit in the data. A FAST template replaces the tag=value syntax with "implicit tagging" as follows:

- tag numbers are not present in the message but specified in the template

- fields in a message occur in the same sequence as tags in the template
- the template specifies an ordered set of fields with operators

3.1.3 Field Encoding Operators

FAST functions as a state machine and must know which field values to keep in memory. FAST compares the current value of a field to the prior value of that field and determines if the new value should be constant, default, copy, delta (integer or string), increment, or tail.

3.1.3.1 Dictionary Context

CME Group uses a dictionary context on a per-packet basis. A dictionary is a cache in which previous values are maintained. **All dictionary entries are reset to the initial values specified after each UDP packet. Currently, CME Group sends one message per UDP packet.**

3.1.3.2 Field Operators

Note: The following are general descriptions and may have exceptions depending upon the scenario.

A field within a FAST template will generally have one of the Field Operators described below indicating the required decoding action. Please note that in some cases it is possible for a field to have no Field Operator.

- **Constant** – indicates that the field will always contain a predetermined value as specified by the *value* attribute. If the value is optional this field will contain a Pmap bit; if mandatory this field will not contain a Pmap bit.
- **Default** – indicates that the default value defined in the template should be used as the decoded value when a data value is not present. If a data value is present, use that value.
- **Copy** – indicates that the data value in the prior occurrence of this field should be used if a data value is not present for this occurrence.
- **Delta for integers** – when used with an integer, indicates that the data value represents the arithmetic difference between the current and prior values.
- **Delta for strings** – when used with a string, indicates that the data value is either pre-pended or appended to the prior value of this field after removing the specified number of bytes from the beginning or end of the string.
 - A negative subtraction length means the operation takes place on the front of the string.
 - A zero or positive subtraction length means the operation takes place on the end of the string.
- **Increment** – indicates that the data value for the prior occurrence of this field should be incremented by 1 if a data value is not present for the current occurrence. This operator works with integers only.
- **Tail** – this operator works with strings and byte vectors and specifies the number of characters to remove and append to the base value. The length of this value must be constant.

3.1.4 Data Types

A field within a FAST template will have one of the following Data Types indicating the required decoding action:

- **String** – used to represent ASCII or Unicode values using the FAST 7-bit binary encoding.
- **Signed Integer** – used to represent a signed (+/-) integer using the FAST 7-bit binary encoding. A two's complement integer representation is used, with the most significant data bit being the sign bit. Note that contiguous leading bits of the same value must be dropped (so, for instance, only 1 byte is required to encode -1, 0xFF). In some cases, a 0x00 byte will be the most significant byte sent to preserve sign, so 64 is represented 0x00 0xC0.
- **Unsigned Integer** – used to represent unsigned integers using the FAST 7-bit binary encoding.
- **Decimal** – used to represent a floating point number as exponent and mantissa. The exponent is a signed integer used to express precision and the mantissa is a signed integer used to express the value. The numerical value is obtained by multiplying the mantissa with the base-10 power of the exponent expressed as:

$$\text{number} = \text{mantissa} * 10^{\text{exp}}$$

The exponent and mantissa can be encoded as a single, composite field or as individual conjoined fields.

3.1.5 FAST Template

A FAST template corresponds to a FIX message type and uniquely identifies an ordered collection of fields. The template also includes syntax indicating the type of field and transfer decoding to apply. A template is communicated between CME Group and client systems in XML syntax using the FAST v1.1 Template Definition Schema maintained by FIX. The XML format is human- and machine-readable and can be used for authoring and storing FAST templates. Session Control Protocol (SCP) will not be used.

3.1.6 Presence Map and Stop Bit

The Presence Map (Pmap) indicates which fields in the template have data present and which fields have data implied. A Pmap is a sequence of encoded bits with each bit representing a template field according to sequence. Fields with data present have the Pmap bit set to '1'. Fields with data implied have the Pmap bit set to '0' (Exception: fields with a Constant operator in which the bit is set to '1' for an implied state).

A Pmap occurs at the beginning of each FAST message and at the beginning of any sequence/group of fields as long as those fields are defined such that a Pmap slot is required.

3.1.6.1 Presence Map Rules

Pmap rules determine when a FAST template field requires a corresponding Pmap bit. A field does not require a bit in the pmap when it meets any of the following criteria:

1. The field is defined as mandatory without a field operator – a value will always be present.
2. The field is defined as mandatory with a constant operator – a value should always be instantiated in the decoded message based on the value in the template.
3. The field is defined as mandatory with a delta operator – a delta value is always present.
4. The field is defined as optional without a field operator – either a value or NULL will always be present.
5. The field is defined as optional with a delta field operator – a delta value or NULL will always be present.

See the FAST v1.1 specification for the complete information on presence map rules.

Rules for Determining if a Presence Map Bit is Required

Operation	Mandatory	Optional
None	No	No
Constant	No	Yes
Copy	Yes	Yes
Default	Yes	Yes
Delta	No	No
Increment	Yes	Yes
Tail	Yes	Yes

3.1.6.2 Message Structure

Note: The Market Data Incremental Refresh (tag 35-MessageType = X) message is used for example purposes throughout this document; not all FAST messages follow this format.

The FIX Market Data Incremental Refresh (tag 35-MessageType = X) message is made up of three components: a Header, a Body and a set of one or more Market Data Entries as shown in the diagram below. The **Header** carries transmission details. The **Body** carries information pertinent to all entries in the message such as TradeDate. The **Market Data Entry** carries specific instructions for updating the book or recording trades. The templates provided by CME Group conform to this general structure and use only fields that are part of the Market Data Incremental Refresh (tag 35-MessageType = X) message.

Optional fields are useful when a generic template is used to provide multiple market data types such as book updates and trades. In this situation, there may be fields which are present in one occurrence of the repeating group but not in another within a given message.

For example, the MDEntries repeating group that is present in the MDIncRefresh template shown above can be used to express a trade, bid, ask, high, low, etc. within a single message. A trade entry will not use tag 346, NumberOfOrders, which will then be defined as optional. However, FAST requires that this field be accounted for in the encoded message if specified in the template. This is done through the use of a reserved value of NULL to indicate that the field is not present in the decoded data. In the serialization layer, FAST reserves binary zeros to indicate a NULL value which tells the decoder that no data is present for this Template Distribution.

Note: CME Group requires that client systems use an API layer to load templates rather than implement hard coded templates. Since templates are subject to change, this will facilitate template modification in production environments.

Table 3.1. Rules for Determining if a Field is Nullable

Operation	Mandatory	Optional
None	No	Yes
<Constant/>	No	No
<Copy/>	No	Yes
<Default/>	No	Yes
<Delta/>	No	Yes
<Increment/>	No	Yes
<Tail/>	No	Yes

3.1.6.5 XML Template Example

A template consists of Field Instructions that define the fields contained in the message. Field Instructions specify the field name, tag number, data type, field operator, and presence attribute that indicates if a field is optional or mandatory.

A sample market data template is shown below. The syntax is standard XML and can be parsed using a variety of open source tools. Valid template syntax is determined by the FAST Template Schema which is available in the FAST v1.1 specification.

- The real-time feed templates are based on the Market Data Incremental Refresh message type.

The information contained in a template is passed to the client FAST decoder at run-time such that the decoder recognizes how each field is encoded in terms of data type representation (Transfer) and data redundancy removal (Field).

The template is constructed of several sections including Template Identification, Header, Body and Sequence. Template Identification provides the template name and identifier. The Header includes FIX header fields such as ApplVerID (tag 1128), MsgType (tag 35), and SendingTime (tag 52). The Body provides information common across all repeating groups. Sequence represents a repeating group with a corresponding length field and a set of repeating group fields which carry the detailed entry information.


```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <template name="MDIncRefresh" id="30" xmlns="http://www.cme.com/mdp/fast-templates" >
3    <typeRef name="MDIncRefresh"/>
4    <string name="ApplVerID" id="1128"> <constant value="7"/> </string>
5    <string name="MessageType" id="35"> <constant value="X"/> </string>
6    <string name="SenderCompID" id="49"> <constant value="CME"/> </string>
7    <uint32 name="MsgSeqNum" id="34"> </uint32>
8    <uint64 name="SendingTime" id="52"> <delta/> </uint64>
9    <uint32 name="TradeDate" id="75"><copy/> </uint32>
10   <sequence name="MDEntries"><length name="NoMDEntries" id="268"></length>
11     <decimal name="MDEntryPx" id="270">
12       <exponent><copy value="-2"/></exponent>
13       <mantissa><delta/></mantissa> </decimal>
14     <int32 name="MDEntrySize" id="271"> <delta/> </int32>
15     <uint32 name="MDEntryTime" id="273"> <delta/> </uint32>
16     <uint32 name="MDPriceLevel" id="1023" presence="optional"> <increment/> </uint32>
17     <uint32 name="MDUpdateAction" id="279"> <copy value="1"/> </uint32>
18     <string name="MDEntryType" id="269"> <copy value="0"/> </string>
19     <uint32 name="SecurityID" id="48"> <copy/> </uint32>
20     <uint32 name="SecurityIDSource" id="22"> <constant value="8"/> </uint32>
21     <int32 name="NumberOfOrders" id="346" presence="optional"> <delta/> </int32>
22     <string name="QuoteCondition" id="276" presence="optional"> <copy value="K"/> </string>
23     <string name="TickDirection" id="274" presence="optional"> <default/> </string>
24     <int32 name="NetChgPrevDay" id="451" presence="optional"> <default/> </int32>
25     <string name="TradeCondition" id="277" presence="optional"> <default/> </string>
26     <int32 name="TradeVolume" id="1020" presence="optional"> <default/> </int32>
27     <string name="TradingSessID" id="336"><default value="1"/> </string>
28   </sequence>
29 </template>
30

```

Please refer to the Appendix for a decomposition of the template above with an explanation for each instruction.

Templates are available from the ftp site at <ftp.cmegroup.com>.

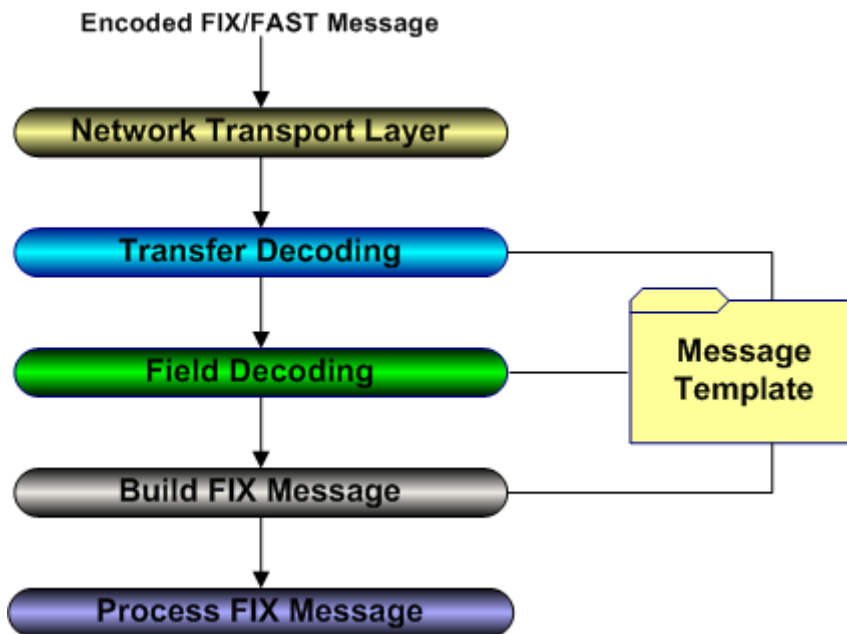
3.2 FAST Decoding

This section presents one possible approach to decoding CME Group FAST messages to demonstrate the fundamental concepts involved in the process. This approach is provided for example purposes and is not definitive.

FAST processing decodes a FIX/FAST message by means of FAST templates provided by CME Group. The FAST template contains the instructions to decode and reconstruct compressed message data into the FIX format and also supports repeating groups (sequences) that allow a single message to convey multiple instructions (i.e. book update, trade, high/low, etc.).

Note: CME Group FAST encoding is bit-specific to the FAST message specification. CME Group FAST encoding always uses the global dictionary.

A decoding sequence can follow the steps diagramed below; this document describes Transfer Decoding, Field Decoding, and Building the FIX message only.



Step 1. Transport - Client system receives encoded FAST message.

Step 2. Transfer Decoding

- Identify template
- extract binary encoded bits
- map bits to fields per template field

Step 3. Field Decoding - Apply operators to determine values per template field.

Step 4. Build FIX message.

Step 5. Process FIX message.

3.3 Transfer Decoding Overview

Transfer Decoding is the initial step that converts data from the FAST 7-bit binary format.

Note: The decoding process can take place on a field-by-field basis, de-serializing and rebuilding the decoded value for each field. Alternately, the decoding process may de-serialize the entire message and then make a second iteration over the message to rebuild the decoded values.

The FAST transfer format is a binary representation enhanced with the following attributes:

- **Stop Bit** - Each data byte contains a stop bit to indicate whether this is the last byte of a field. The stop bit is the seventh bit of the most significant byte. A stop bit set to 1 indicates the last byte of a field. A stop bit set to zero indicates this is not last byte of the field.

- **Presence Map** - The Presence Map occurs at the start of every message or repeating group and indicates the presence or absence of individual fields. The template specifies the type of transfer decoding to employ for each field in the message (i.e. string, integer, decimal, etc.).



WARNING

FAST does not support timestamps. CME Group will convert the timestamp to an integer by removing punctuation from the UTC timestamp prior to encoding. The decoding application should convert the integer to the FIX UTC format after decoding. In addition, the timestamp header on the message packet should not be used for latency comparisons. The message needs to be decoded to determine the timestamp for updates.

3.3.1 Field Decoding Overview

Field decoding is the second part of the decompression process that reconstructs data values according to template-specified operations. Field decoding operations are assigned per field within the template; decoding reinstates data as indicated by the template.

3.3.1.1 When to Reset Decoder State

The state of the decoder will need to be reset for each received UDP packet and the dictionary applied. Resetting state means that all fields are set to their initial pre-processing state. This is because UDP transport is not reliable and data in a packet cannot be dependent on data in a previous packet since it is possible for packets to be lost or arrive out of sequence.

3.3.2 Receiving Data over MDP

Currently, CME Group will send only a single FIX message per packet; however, in future implementations CME Group may send multiple messages per packet. The basic operation requires the decoding of data within a discreet packet as well as the ability to determine the end of one message and beginning of another within that packet.

3.3.2.1 Decoding a Packet

Decoding of the datagram should be conducted as a unit of work in which only the data in a given packet participates in the scope of the decoding process. To determine when the process has fully decoded a discreet message, it is important to use the template to step through a message field-by-field.

3.3.2.2 Decoding Messages in a Packet

When the decoder determines that it has processed all repeating groups in the message, and that there are no further fields specified by the template, then it can conclude that the end of the message has been reached. In this way, it is necessary to use the presence maps, NoMDEntries, and the template to determine when end of message has been reached. To recap, the steps involved are:

- Decode fields in the non-repeating body of the message using the high-level presence map.
- If the last field in the template has been reached then end of message has been encountered.

- If the NoMDEntries field is present, then process each repeating group using the repeating group presence map and template.
- When number of repeating groups processed equals the count specified in NoMDEntries, then the end of the repeating groups has been encountered.

3.3.2.3 Error Handling in a Broadcast Environment

If FAST encounters an error during the process of decoding the contents of a UDP packet, the process should stop, throw an exception, and decode the next packet. In all likelihood, the error is due to an inconsistency between the encoded data and the template being used to decode the data. At this point, it is advised that parties synchronize templates in order to ensure that the same data definitions are being used.

3.4 Decoding Sequence

This section provides a detailed example of the process for decoding a FAST message. You should thoroughly test your application to determine that you have properly decoded a message.

Note: This decoding example is a generic description of the basic decoding process; it does not reflect the structure of the reference code.

FAST message decoding consists of three major steps:

Step 1. Transfer Decoding

- Identify Template ID and Pmap bit value for each given field. Pmap values are 0, 1.
- Decode FAST 7-bit binary values to identify data present in the message.

Step 2. Field Decoding

- Apply field operators to extracted binary values.
- Determine state of field to be decoded.

Step 3. Build FIX Message

- Apply FIX or internal structure to decoded message

The decoding sequence in the example used throughout this section uses a FAST message containing a header, trade, and two new bids.

3.4.1 Transfer Decoding

Upon receipt of the FAST message, the client decoder loads the template according to the Template ID. From this template the client decoder identifies how to decode each field, whether a field will have a Pmap bit, and if a field can carry a null value. In this example the MDIncRefresh template contains a header and repeating group structure (*italics*).

Note: The following template is an EXAMPLE TEMPLATE.

```
<template name="MDIncRefresh" id="30">
  <typeRef name="MDIncRefresh"/>
  <string name="MessageType" id="35"> <constant value="X"/> </string>
  <sequence name="MDEntries"><length name="NoMDEntries" id="268"></length>
    <uint32 name="MDUpdateAction" id="279"> <copy value="0"/> </uint32>
    <uint32 name="MDEntryType" id="269"> <copy value="0"/> </uint32>
    <decimal name="MDEntryPx" id="270">
      <exponent><copy value="-2"/></exponent>
      <mantissa><delta/></mantissa> </decimal>
    <int32 name="MDEntrySize" id="271"> <delta/> </int32>
    <uint32 name="MDPriceLevel" id="1023" presence="optional"> <increment/>
    <uint32 name="SecurityID" id="48"> <copy/> </uint32>
    <uint name="SecurityIDSource" ID="22"><constant value="8"/></uint32>
    <int32 name="NumberOfOrders" id="346" presence="optional"> <delta/> </int32>
  </sequence>
</template>
```

3.4.2 Decoding Process

The client system reads the Pmap to identify required field-level decoding.

Pmap 1 11000000	Header Fields	Pmap 2 10101100	Trade Fields	Pmap 3 10101000	New Bid 1 Fields	Pmap 4 10000000	New Bid 2 Fields
--------------------	------------------	--------------------	-----------------	--------------------	---------------------	--------------------	---------------------

In this example light grey values indicate an unused bit.

Step 1. **Extract Pmap1 from data.** Pmap1 is the top-level Pmap that contains the Pmap bits for all fields requiring a Pmap bit in the message.

- The first bit in Pmap1 after the stop bit (in bold) is assigned to the Template ID, which always has a bit. The bit is turned on indicating that the Template ID is present in the encoded message and that the corresponding template must be loaded.
- The first field in the template is the MessageType, defined as a mandatory field using a Constant operator with a value of "X" retrieved from the template and therefore does not have a bit.
- The next field in the template is NoMDEntries. The length specified in the sequence determines the number of repeating groups in the message. This field does not have a bit since the field is mandatory and does not have an operator.
- **The remaining bits in Pmap1 are defaulted to 0 but are not referenced by the decoding process; at this point, the Header fields have been decoded.**

Step 2. **Extract Pmap2 from data.** Pmap2 represents the first repeating group and contains trade-related fields.

- The next field in the template is MDUpdateAction. By definition, this field requires a bit in the Pmap since it uses the *Copy* field operator. In this example, the bit is set to 0 indicating that no data is present and that the value must be derived from the initial value.
- The next field in the template is MDEntryType, which is a *Copy* field. The Pmap set to '1' indicates an encoded value present in the field.
- The decoding process continues using the template and Pmap2 to traverse the data.

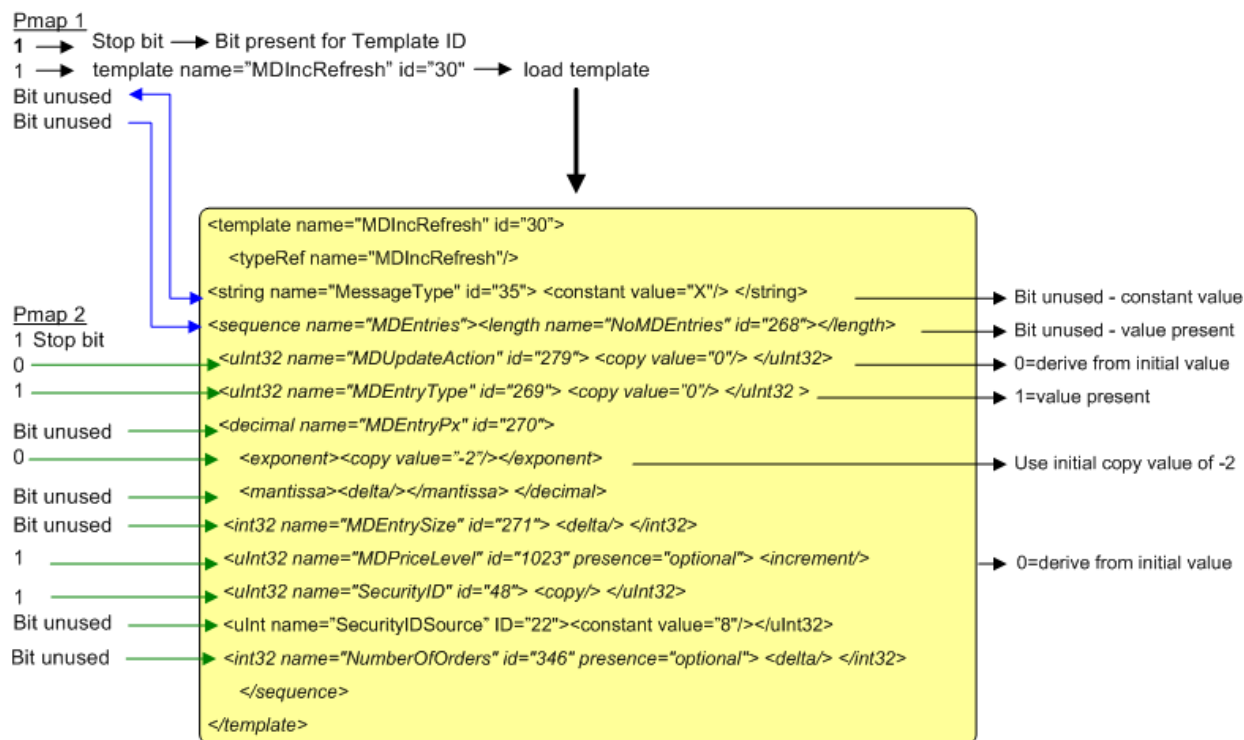
Step 3. **Extract Pmap3 from data.** Pmap3 represents the presence map for repeating group2 followed by the New Bid1 fields.

- By the time the decoding process reaches Pmap3, prior values have been established for several fields based on the content of the prior repeating group.
- The decoding process uses the prior values in combination with the encoded data to generate the proper decoded values.

Step 4. **Extract Pmap4 from data.** Pmap4 is the final presence map and the decoder recognizes that all bits are set to off indicating that either prior values should be used or that fields do not use a bit.

The following diagram illustrates the decoding process flow:

DECODER



Bit unused = no Pmap bit required for delta value

3.4.3 Build the FIX Message

The table in this example shows the step-by-step conversion for each field of the FAST message into the FIX format.

The following steps describe the actions taken to obtain the data shown in the Message Decoding Process table.

- Step 1. Decoding begins with the identification of the Pmap bit for each field.
- Step 2. The encoded FAST 7-bit binary values are obtained as shown in the "Encoded FAST 7-Bit Binary Value" column.
- Step 3. The encoded FAST 7-bit binary values from step 2 are deserialized based on the data type specified in the template.
- Step 4. The decoder maintains the state of prior values for each field throughout decoding and applies them for fields having operators of Delta, Copy, or Increment.
- Step 5. Obtain fully decoded values.

Message Decoding Process Table									
				Step 1	Step 2	Step 3	Step 4		Step 5
#	Field Name	Presence*	Data Type/ Field Operator	Pmap Bit	Encoded FAST Hex/ Binary Value	Deserialized Encoded Value	Prior Value	Initial Value	Decoded Value
BIT STREAM = 11000000 10011110 100000011									
1	Template ID	M	uint32/ copy	1	0x9e – 10011110	30			TID = 30
2	MsgType	M	string/ constant	No bit	None	None		X	35=X
3	NoMDent	M	uint32/no operator	No bit	0x83 - 10000011	3			268=3
Repeating Group 1 - Trade									
BIT STREAM = 10101100 10000010 00111001 01100000 11001010 10000101 10000000 00110000 01101010 11111011 10000000									
4	UpdateAction	M	uint32/ copy	0	None	None		0	279=0
5	EntryType	M	uint/copy	1	0x82 - 10000010	2		0	269=2
6	EntryPrice	M	Exp: int32/copy Mant: int64/ delta	Exp - 0 Mant – No bit	Exp: None Mant: 0x39 0x60 0xca – 00111001 01100000 11001010	Exp = None Mantissa = 946250		-2	270=9462.50
7	EntrySize	M	int32/ delta	No bit	0x85 - 10000101	5			271=5
8	PriceLevel	O	uint32/ increment	1	0x80 – 10000000	NULL		1	1023=No Value
9	SecurityID	M	uint32/ copy	1	0x30 0x6a 0xfb - 00110000 01101010 11111011	800123			48=800123
10	Security- IDSource	M	uint32/ constant	No bit	None	None		8	22=8
11	NoOrders	O	int32/ delta	No bit	0x80 – 10000000	NULL		0	346=No Value
Repeating Group 2 – New Bid 1									
BIT STREAM = 10101000 10000000 11001110 00000001 10101010 10000010 10010000									
12	UpdateAction	M	uint32/ copy	0	None	None	0		279=0
13	EntryType	M	uint/copy	1	0x80 - 10000000	0	2		269=0
14	EntryPrice	M	Exp: int32/copy Mant: int64/ delta	0 No bit	Exp: None Mant: 0xce - 11001110	Exp = None Mantissa = -50	-2 946250		270=9462.00
15	EntrySize	M	int32/ delta	No bit	0x01 0xaa – 00000001 10101010	170	5		271=175
16	PriceLevel	O	uint32/ increment	1	0x02 - 10000010	2*	NULL		1023=1

17	SecurityID	M	uint32/ copy	0	None	None	800123		48=800123
18	Security- IDSource	M	uint32/ constant	No bit	None	None	8		22=8
19	NoOrders	O	int32/ delta	No bit	0x90 - 10010000	16*	0**		346=15
Repeating Group 3 – New Bid 2									
BIT STREAM = 10000000 11001110 11010110 11111101									
20	UpdateAc- tion	M	uint32/ copy	0	None	None	0		279=0
21	EntryType	M	uint/copy	0	None	None	0		269=0
22	EntryPrice	M	Exp: int32/copy Mant: int64/ delta	0 No bit	Exp: None Mant: 0xce - 11001110	Exp = None Mantissa = -50	-2 946200		270=9461.50
23	EntrySize	M	int32/ delta	No bit	0xd6 – 11010110	-42	175		271=133
24	PriceLevel	O	uint32/ increment	0	None	None	1		1023=2
25	SecurityID	M	uint32/ copy	0	None	None	800123		48=800123
26	Security- IDSource	M	uint32/ constant	No bit	None	None	8		22=8
27	NoOrders	O	int32/ delta	No bit	0xfd - 11111101	-3	15		346=12

*M=mandatory O=optional

**Subtract 1 from non-negative optional integer fields

***For delta fields, if a null is received, the prior value in the dictionary is not changed. In this case, the initial value was not specified in the template, so the initial value is set to '0'. The first encoded field received has a value of null, so the prior value is '0' when decoding the second instance of this field.

3.4.3.1 Result - Decoded Values

35=x|268=3 (message header)

279=0|269=2|270=9462.50|271=5|48=800123|22=8 (trade)

279=0|269=0|270=9462.00|271=175|1023=1|48=800123|22=8|346=15 (new bid 1)

279=0|269=0|270=9461.50|271=133|1023=2|48=800123|22=8|346=12 (new bid 2)

3.5 Sample Template

Line #	Template Syntax	Use and Description
2	<code><template name="MDIncRefresh" id="35"></code>	Provides the template name (MDIncRefresh) and template identifier (35).
3	<code><typeRef name="MDIncRefresh" /></code>	Specifies the type reference name that allows the template to be referenced in other templates.
4	<code><string name="ApplVerID" id="1128"> <constant value="FIX.5.0" /> </string></code>	Field instruction for ApplVerID defined as a string with an identifier of 1128 corresponding to the FIX tag number. ApplVerID has a constant field operator with a value of FIX.5.0 indicating the FIX version.
5	<code><string name="MessageType" id="35"> <constant value="X" /> </string></code>	Field instruction for MessageType defined as a string with identifier = 35 corresponding to the FIX tag number. MessageType has a constant field operator with a value of X which indicates the FIX message type—in this case Market Data Incremental Refresh.
6	<code><string name="SenderCompID" id="49"> <constant value="CME" /> </string></code>	Field instruction for SenderCompID defined as a string with identifier = 49 corresponding to the FIX tag number. SenderCompID has a constant field operator with a value of 'CME' which indicates the originator of the data.
7	<code><uint32 name="MsgSeqNum" id="34"> <increment /> </uint32></code>	Field instruction for MsgSeqNum defined as an unsigned integer with identifier = 34 corresponding to the FIX tag number. MsgSeqNum has an increment field operator.
8	<code><uint64 name="SendingTime" id="52"> <delta /> </uint64></code>	Field instruction for SendingTime defined as an unsigned integer and with identifier = 52 corresponding to the FIX tag number. SendingTime has a delta field operator.
9	<code><sequence name="MDEntries"> <length name="NoMDEntries" id="268"> <copy /> </length></code>	Sequence instruction demarks the beginning of the MDEntries repeating group. The sequence includes a length field called 'NoMDEntries' that specifies the number of repeating groups present in the message. NoMDEntries has a copy field operator
10	<code><int32 name="MDEntryPx" id="270"> <delta /> </int32></code>	Field instruction for MDEntryPx (first field instruction in repeating group) defined as a signed integer with identifier = 270 corresponding to the FIX tag number. MDEntryPx has a delta field operator.
11	<code><int32 name="MDEntrySize" id="271"> <delta /> </int32></code>	Field instruction for MDEntrySize defined as a signed integer with identifier = 271 corresponding to the FIX tag number. MDEntrySize has a delta field operator
12	<code><uint32 name="MDEntryTime" id="273"> <delta /> </uint32></code>	Field instruction for MDEntryTime which is defined as an unsigned integer with identifier = 273 corresponding to the FIX tag number. MDEntryTime has a delta field operator.

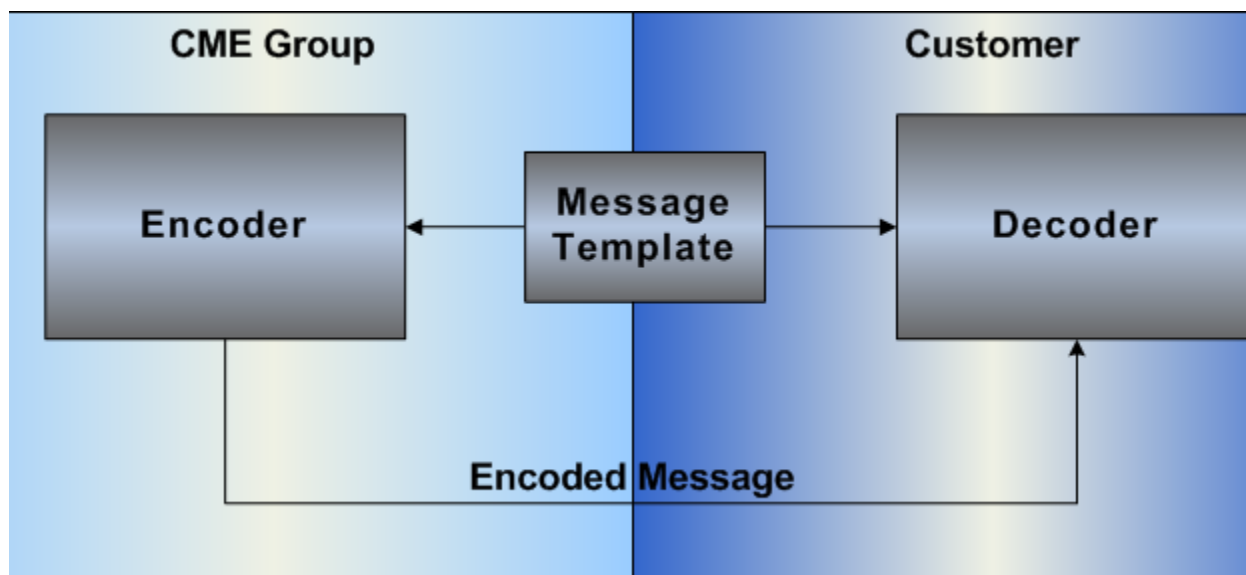
Line #	Template Syntax	Use and Description
13	<code><uint32 name="MDPriceLevel" id="1023" presence="optional"> <increment /> </uint32></code>	Field instruction for MDPriceLevel defined as an unsigned integer with identifier = 1023 corresponding to the FIX tag number. MDPriceLevel has a delta field operator.
14	<code><uint32 name="MDUpdateAction" id="279"> <copy value="0" /> </uint32></code>	Field instruction for MDUpdateAction defined as an unsigned integer and identifier = 279 corresponding to the FIX tag number. MDUpdateAction has a copy field operator with a value of 0.
15	<code><string name="MDEntryType" id="269"> <default value="1" /> </string></code>	Field instruction for MDEntryType which is defined as a string and has an identifier of 269 which corresponds to the FIX tag number. MDEntryType has a default field operator with a value of 1.
16	<code><int32 name="SecurityID" id="48"><delta/></int32></code>	Field instruction for SecurityID defined as an unsigned integer and has an identifier = 48 corresponding to the FIX tag number. SecurityID has a delta field operator.
16	<code><uint32 name="SecurityID" idSource id="22"><constant value="8"></uint32></code>	Field instruction for SecurityIDSource defined as an unsigned integer and has an identifier = 22 corresponding to the FIX tag number. SecurityDesc has a constant field operator.
17	<code><int32 name="NumberOfOrders" id="346" presence="optional"> <delta /> </int32></code>	Field instruction for NumberOfOrders defined as a signed integer and with identifier = 346 corresponding to the FIX tag number. NumberOfOrders has a delta field operator.
18	<code><string name="QuoteCondition" id="276" presence="optional"> <copy /> </string></code>	Field instruction for QuoteCondition defined as a string with identifier = 276 corresponding to the FIX tag number. QuoteCondition has a copy field operator.
19	<code><string name="TickDirection" id="274" presence="optional"> <default /> </string></code>	Field instruction for TickDirection is defined as a string with identifier = 274 corresponding to the FIX tag number. TickDirection has a default field operator.
20	<code><uint32 name="NetChgPrevDay" id="451" presence="optional"> <default /> </uint32></code>	Field instruction for NetChgPrevDay defined as an unsigned integer with identifier = 451 corresponding to the FIX tag number. NetChgPrevDay has a default field operator.
21	<code><string name="TradeCondition" id="277" presence="optional"> <default /> </string></code>	Field instruction for TradeCondition defined as a string with identifier = 277 corresponding to the FIX tag number. TradeCondition has a default field operator.
22	<code><int32 name="TradeVolume" id="1020" presence="optional"> <default /> </int32></code>	Field instruction for TradeVolume defined as a signed integer with identifier = 1020 corresponding to the FIX tag number. TradeVolume has a default field operator.

Line #	Template Syntax	Use and Description
23	<code><string name="Trading-SessID" id="336"> <default value="1" /> </string></code>	Field instruction for TradingSessID defined as a string with identifier = 336 corresponding to the FIX tag number. TradingSessID has a default field operator. TradingSessID is the last field in the Sequence

4. Template Overview

CME Group provides a single xml file that contains a collection of templates and is shared across all market data channels between the encoder (CME side) and decoder (customer side). Each template has a unique template ID that describes the format of an encoded message. A template ID is carried in every encoded message to provide a reference to the correct template for decoding purposes.

A template selection process should be carried out on the customer side. The decoder should dynamically identify and retrieve the correct template indicated in the encoded message, then use that template to decode the message.



The template ID is an unsigned integer carried as the first data field following the first Pmap of every message allowing the decoding system to apply the correct template to the message upon receiving it.

Example: Template ID

```

2  <template name="MDIncRefresh" id="35" xmlns="http://www.fixprotocol.org/ns/fast/1.1">
3  <typeRef name="MDIncRefresh"/>

```

CME Group will not send the same template ID more than once in a UDP packet.

4.1 XML Template Example

A template consists of Field Instructions that define the fields contained in the message. Field Instructions specify the field name, tag number, data type, field operator, and presence attribute that indicates if a field is optional or mandatory.

A sample market data template is shown below. The syntax is standard XML and can be parsed using a variety of open source tools. Valid template syntax is determined by the FAST Template Schema which is available in the FAST v1.1 specification.

- The real-time feed templates are based on the Market Data Incremental Refresh message type.

The information contained in a template is passed to the client FAST decoder at run-time such that the decoder recognizes how each field is encoded in terms of data type representation (Transfer) and data redundancy removal (Field).

The template is constructed of several sections including Template Identification, Header, Body and Sequence. Template Identification provides the template name and identifier. The Header includes FIX header fields such as ApplVerID (tag 1128), MsgType (tag 35), and SendingTime (tag 52). The Body provides information common across all repeating groups. Sequence represents a repeating group with a corresponding length field and a set of repeating group fields which carry the detailed entry information.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <template name="MDIncRefresh" id="30" xmlns="http://www.cme.com/mdp/fast-templates" >
3    <typeRef name="MDIncRefresh"/>
4    <string name="ApplVerID" id="1128" <constant value="7"/> </string>
5    <string name="MessageType" id="35" <constant value="X"/> </string>
6    <string name="SenderCompID" id="49" <constant value="CME"/> </string>
7    <uint32 name="MsgSeqNum" id="34" </uint32>
8    <uint64 name="SendingTime" id="52" <delta/> </uint64>
9    <uint32 name="TradeDate" id="75" <copy/> </uint32>
10   <sequence name="MDEntries" <length name="NoMDEntries" id="268" </length>
11     <decimal name="MDEntryPx" id="270"
12       <exponent><copy value="-2"/></exponent>
13       <mantissa><delta/></mantissa> </decimal>
14     <int32 name="MDEntrySize" id="271" <delta/> </int32>
15     <uint32 name="MDEntryTime" id="273" <delta/> </uint32>
16     <uint32 name="MDPriceLevel" id="1023" presence="optional" <increment/> </uint32>
17     <uint32 name="MDUpdateAction" id="279" <copy value="1"/> </uint32>
18     <string name="MDEntryType" id="269" <copy value="0"/> </string>
19     <uint32 name="SecurityID" id="48" <copy/> </uint32>
20     <uint32 name="SecurityIDSource" id="22" <constant value="8"/> </uint32>
21     <int32 name="NumberOfOrders" id="346" presence="optional" <delta/> </int32>
22     <string name="QuoteCondition" id="276" presence="optional" <copy value="K"/> </string>
23     <string name="TickDirection" id="274" presence="optional" <default/> </string>
24     <int32 name="NetChgPrevDay" id="451" presence="optional" <default/> </int32>
25     <string name="TradeCondition" id="277" presence="optional" <default/> </string>
26     <int32 name="TradeVolume" id="1020" presence="optional" <default/> </int32>
27     <string name="TradingSessID" id="336" <default value="1"/> </string>
28   </sequence>
29 </template>
30

```

4.2 Template Implementation Considerations

The following items should be considered before implementing template functionality:

- Client systems should use the defined sizes and type for each tag in the [FIX Message Specifications](#) as a guide for storing data. **Do not use the template to define this.**
- Any change to the template will result in an update to the template ID.
- The following template changes should be handled by the client without any changes to their decoder.

Note: Customers will be notified prior to a template change via a [Globex Advisory Notice](#) and a [Market Data Advisory Notice](#).

Template Change	Decoder Impact	Template Release Plan	Additional Information
New Tags (CME Group originated)	none	A template that contains a new tag (CME Group originated) will be released in the New Release Certification environment approximately 4 weeks prior to Production environment (and Certification environment) roll-out.	A new tag may be defined as a result of a change in the business logic (which client systems can choose to implement or not). This change should not require a modification to the client system decoder, however, there may be changes required to the clients back-end systems if they choose to implement this change.
New Tags (FIX originated)	none	A template that contains a new tag (FIX originated) will be released in the New Release environment approximately 4 weeks prior to Production environment (and Certification environment) roll-out.	A new tag may be defined as a result of a change in the business logic (which client systems can choose to implement or not). This change should not require a modification to the client system decoder, however, there may be changes required to the clients back-end systems if they choose to implement this change.
Removed Tags	none	A template in which a tag is no longer available will be released in the New Release environment approximately 4 weeks prior to Production environment (and Certification environment) roll-out.	A tag may be removed, for example, if it is no longer needed.

Template Change	Decoder Impact	Template Release Plan	Additional Information
Modified Tags	none	A template that contains a modified tag will be released in the New Release environment approximately 4 weeks prior to Production environment (and Certification environment) roll-out.	A modification to an existing tag may be defined as a result of a change in the business logic (which client systems can choose to implement or not). This tag modification should not require a modification to the client system decoder, however, there may be changes required to the clients back-end systems if they choose to implement this change.
Rename or Reorder Tags	none	A template that contains a renamed or reordered tag will be released in the New Release environment approximately 2 weeks prior to Production environment (and Certification environment) roll-out.	A renamed or reordered tag may be defined to increased efficiency. This tag modification should not require a modification to the client system decoder, however, there may be changes required to the clients back-end systems if they choose to implement this change.
New Message Types	none	A template that contains a new message type will be released in the New Release environment approximately 4 weeks prior to Production environment roll-out.	A new message type may be defined as a result of enhanced Globex functionality, or a change in business logic (which client systems can choose to implement or not). This change should not require a modification to the client system decoder, however, there may be changes required to the clients back-end systems if they choose to implement this change.
Modified Operators	none	A template that contains a modified operator will be released in the New Release environment approximately 2 weeks prior to Production environment roll-out.	

Template Change	Decoder Impact	Template Release Plan	Additional Information
Modified Data Types	none	A template that contains a modified data type will be released in the New Release environment approximately 2 weeks prior to Production environment roll-out.	

4.3 Template Distribution

The current template for each environment is available for download from an ftp site. Refer to “Services - Template Dissemination and Market Data Configuration” on Page 3 for an outline of the process.

Note: CME strongly recommends that you download the current templates file every Sunday prior to market open.

Historical templates.xml files will be maintained on the ftp site and stored in a directory indicating the dates they were effective. The historical templates will be moved to the “Archive” directory for the corresponding environment.

4.4 Template Modification

A template archiving process is available. From the template ftp site (ftp.cmegroup.com), the ‘Templates’ directory for the corresponding environment (Production, New Release, and Certification) contains two sub-directories:

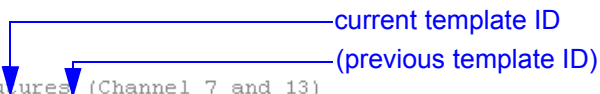
- **Active** - contains the current templates.xml file.
- **Archive** - contains the oldest version of the templates.xml file.

When modifying the templates.xml file, CME Group makes a copy of the active version, adds the next available template identifier, then makes the necessary changes in the templates.xml file and replaces it in the Active directory. Two versions of the template remain in the templates.xml file, with the previous template listed under the new template. When a template is changed for the third time, the oldest version of the template is moved to the templates.xml file in the Archive directory.

Example:

If template 30 previously replaced template 28 and now a new template 57 is being generated to replace template 30, then template 28 should be removed from the Active directory and added to the Archive directory. Templates 57 and 30 remain in the Active directory.

The template.xml file in the Active directory lists both the current and the previous template IDs, as displayed in the following example:



```

*Incremental refresh templates
*Equity Futures and Commodity Futures (Channel 7 and 13)
MDIncRefresh.(generic).....= 57    (30)
MDIncRefresh.....= 32    (-)
MDIncRefresh.....= 56    (33)

*Interest Rate Futures, Nymex Futures, CBOT Futures (Chan
MDIncRefresh.(generic).....= 59    (34)
MDIncRefresh.....= 35    (-)
MDIncRefresh.....= 36    (-)
MDIncRefresh.....= 60    (37)

*FX Futures (Channel 6, 11, 22)
MDIncRefresh.(generic).....= 55    (38)
MDIncRefresh.....= 39    (-)
MDIncRefresh.....= 58    (40)

*Equity Options, FX Options, Commodity Options and Nymex
MDIncRefresh.....= 41    (-)
MDIncRefresh.(generic).....= 61    (42)

*Interest Rate Options (Channel 10)
MDIncRefresh.....= 43    (-)
MDIncRefresh.....= 44    (-)
MDIncRefresh.(generic).....= 62    (45)

*Security Definition template
MDSecurityDefinition..... = 52    (46)

*Quote template
MDQuoteRequest..... = 54    (47)

*Market State Template
MDSecurityStatus..... = 48    (-)

*Snapshot template
MDSnapshotFullRefresh..... = 53    (51)

*Admin Templates
MDNewsMessage..... = 49    (-)
MDHeartbeat..... = 50    (-)
MDLogon..... = 1    (-)
MDLogout..... = 2    (-)

```

The following example displays the current template name “57” with the previous template name “30” listed below it. A note below the previous template indicates the current template ID.

```

-->
+ <template name="MDIncRefresh_57" id="57" dictionary="57" xmlns="http://www.fixprotocol.org/ns/fast/td/1.1">
- <template name="MDIncRefresh_30" id="30" dictionary="30" xmlns="http://www.fixprotocol.org/ns/fast/td/1.1">
  <!-- desc="DEPRECATED. USE VERSION 57" -->
- <string name="ApplVerID" id="1128">
  <constant value="8" />
</string>

```

5. Incremental Book Management

5.1 Overview

CME Group uses an electronic market data format based on the FIX standard which provides a sound model for reducing the overall content of data transmitted through an incremental book management approach. FIX provides a flexible protocol for market data messaging which is well suited for transmitting electronic books from the CME Globex trading platform to customers. Incremental book management provides significant efficiency and flexibility across the entire market data infrastructure. The use of incremental FIX market data messaging in combination with FAST compression produces a highly optimized feed which results in bandwidth savings as well as latency reductions.

CME Group represents markets in executable orders and quotes using a central limit order book. This order book is constantly changing as market events cause orders to be added, modified, and cancelled. Updates are then sent out over a market data stream so client systems can then construct a copy of the book in order to track prices in the market and submit appropriately priced orders.

Note: Multiple data blocks may be sent in the same message. To determine the number of data blocks you will receive in the message, refer to tag 268-NoMDEntries. Within the message, data blocks may be for different instruments or entry types (book update, statistics, or trades).

5.2 Incremental Book Management Applicable FIX Message Structures

The Market Data Incremental Refresh (tag 35-MsgType = X) message is used to apply instructions to a book. These instructions are incremental and update applicable parts of the book as necessary, as opposed to refreshing the entire book each time there is an update. This message is used to maintain the aggregate order book for CME Group products. This message, on a real-time basis, is also used to send statistics.

The FIX Market Data message has a FIX header followed by a number of data blocks. Each data block represents a single instruction such as a book update or trade. This section refers to the items as data blocks when describing how the book is maintained. A single FIX Market Data message can contain many data blocks, such as New bid/ask, Change bid/ask, Delete bid/ask, and Overlay bid/ask, across multiple instruments.

5.2.1 Common Book Update Tags

Common Tags

Tag	Field Name	Description
279	MDUpdateAction	Type of Market Data update action.
269	MDEntryType	Type of Market Data entry.
83	RptSeq	Sequence number per Instrument update.
276	QuoteCondition	Space-delimited list of conditions describing a quote.
48	SecurityID	Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1023	MDPriceLevel	Position in the book.
273	MDEntryTime	Time of Market Data Entry.
270	MDEntryPx	Price of the Market Data Entry.
271	MDEntrySize	Quantity or volume represented by the Market Data Entry.
346	NumberOfOrders	Number of orders in the market.
336	TradingSessionID	Identifier for Trading Session.
1070	MDQuoteType	Identifies the type of quote.

5.3 Central Limit Order Book

The central limit order book is available over a market data feed that provides an aggregate book with a pre-determined number of price levels. The term “aggregate book” is used to refer to the case where orders are summarized at each price level. The quantity associated with that price level is the aggregation of all individual order quantities for that price level. CME Group offers the following:

- multiple-depth book
- top-of-book
- 2-deep book for implied prices futures

Note: Client systems must determine the book-depth for an instrument from tag 264-MarketDepth in the Security Definition (tag 35-MsgType=d) message.

Note: Client systems must be able to process all valid values in tag 279-MDUpdateAction.

5.4 Book Management Mechanics - Multiple-Depth Book

This section applies to CME Group products that have a multiple-depth book. Client systems must determine the book-depth for an instrument from tag 264-MarketDepth in the Security Definition (tag 35-MessageType=d) message.

The examples in this section illustrate the mechanics of a 5-deep book. All books that are **not top-of-book** will use the same mechanics. For example, a 3-deep book will use the same mechanics as described in this section. For top-of-book only, refer to “Book Management Mechanics - Top-of-Book” on Page 47.

5.4.1 Overview

CME Group provides a multiple-depth book for several products. Client systems must determine the book-depth for an instrument from tag 264-MarketDepth in the Security Definition (tag 35-MessageType=d) Message. The aggregate book reports summarized order quantities and order counts at a given price level. The depth represents the number of price levels that are supported in the feed. The view can be visualized as a number of rows in a table for each of the bid and ask sides. On each side, there are a number of rows showing the quantity available at a number of price levels. An aggregate depth book is sequenced by price, descending for bid and ascending for ask.

CME Group provides the best bid and ask in the market for each contract. Trade details and instrument status are provided in separate data blocks. The following table illustrates the types of information that can be displayed:

Top of Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	100	9427.50	9428.00	40	2
19	500	9427.00	9428.50	600	35
34	750	9426.50	9429.00	850	55
25	400	9426.00	9429.50	350	21
14	300	9425.50	9430.00	150	12

CME Group maintains the Aggregate Depth view with the following data blocks:

- **Add** - to create/insert a new price at a specified price level (tag 279 MDUpdateAction=0)
- **Change** - change quantity for a price at a specified price level (tag 279 MDUpdateAction=1)
- **Delete** - remove a price at a specified price level (tag 279 MDUpdateAction=2)

An Aggregate book is built from a series of data blocks which indicate whether an entry is to be inserted (Add), changed (Change), or removed (Delete). All data blocks are issued for a specified entry type (tag 269), price (tag 270), and price level (tag 1023). The incremental instruction approach assumes the use of the Market Data Incremental Refresh (tag 35-MessageType=X) message. The Bid and Ask sides are updated independently with separate data blocks. The practice of sending separate data blocks provides efficiencies by allowing only the bid or ask to be sent, based on which side has changed, rather than both sides.

CME Globex sends an add data block if there is a new price level. Client systems should then shift price levels down, and delete any price levels past the defined depth of the book as indicated in tag 264-Market-Depth in the Security Definition (tag 35-MessageType=d) Message.

CME Globex sends a delete data block to remove a price level in the book. Client systems should shift prices below the data block up to the price level vacated by the deleted price level. If available, an add data block will be sent to fill in the last price level.

The change data block is sent to update characteristics of a price level without changing the price itself, or impacting any other prices on the book. The change data block is sent to update the order count and / or quantity for a price level. The change data block is not sent when the price changes at a given price level.

In general, if a trade occurs, CME Group will send a delete or change data block to update the book. The trade data block itself is not used to update the order book.

Note: The last best price data block will appear similar to a book update data block. To identify that the data block is a last best price data block (and NOT a book update), verify that tag 276 - QuoteCondition contains a value of C (Exchange Best).

5.4.1.1 Basic Book Update Data Block

A FIX message sent to update the top of book will update one side only. The tags normally sent for a book update data block are:

- tag 279-MDUpdateAction
- tag 269-MDEntryType
- tag 83-RptSeq
- tag 1023-MDPriceLevel
- tag 273-MDEntryTime
- tag 271-MDEntrySize
- tag 270-MDEntryPx
- tag 346-NumberOfOrders
- tag 48-SecurityID
- tag 22-SecurityIDSource
- tag 336-TradingSessionID

5.4.2 Examples

This example shows how a book is built and updated. The book generally consists of a set of bid prices and ask prices, in which bids are descending and asks are ascending. The quantity and order count is provided at each price level.

The following table illustrates an initial market book.

5-Deep Book

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	100	9427.50	9428.00	40	2
19	500	9427.00	9428.50	600	35
34	750	9426.50	9429.00	850	55
25	400	9426.00	9429.50	350	21
14	300	9425.50	9430.00	150	12

5.4.2.1 Quantity on Buy Side Modified

The quantity of an order can be modified. The book will show an update to the size displayed:

Buy 90 @ 9427.50

Book Update Instruction

Book Update - Data Block

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	1	1 = change. Type of Market Data update action.
269	MDEntryType	0	0 = bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
1023	MDPriceLevel	1	Price level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	90	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9427.50	Price of the Market Data Entry.
346	NumberOfOrders		Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	8 = CME. Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Updated Book

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	90	9427.50	9428.00	40	2
19	500	9427.00	9428.50	600	35
34	750	9426.50	9429.00	850	55
25	400	9426.00	9429.50	350	21
14	300	9425.50	9430.00	150	12

5.4.2.2 Entire Order Canceled and New Order Entered

An entire order can be canceled, which removes the top of book, and a new order data block will be sent to fill the open price level 5. The book will show the removal of price level 1, followed by an addition to price level 5. A delete data block will be sent for trades.

Note: Client systems should shift prices below the data block up to the price level vacated by the deleted price level. If all levels in the book are full, an add data block will be sent to fill in the last price level.

Cancel 90 @ 9427.50

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	2	2 = delete. Type of Market Data update action.
269	MDEntryType	0	0 = bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
1023	MDPriceLevel	1	Price level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	0	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	0	Price of the Market Data Entry.
346	NumberOfOrders		Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
336	TradingSessionID		Identifier for Trading Session.

Add 400@ 9425.00

Book Update Instruction

Book Update - Data Block 2

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = add. Type of Market Data update action.
269	MDEntryType	0	0 = bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
1023	MDPriceLevel	5	Price level 5. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	400	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9425.00	Price of the Market Data Entry.
346	NumberOfOrders	1	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Updated Book

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
19	500	9427.00	9428.00	40	2
34	750	9426.50	9428.50	600	35
25	400	9426.00	9429.00	850	55
14	300	9425.50	9429.50	350	21
1	400	9425.00	9430.00	150	12

5.4.2.3 New Order Entered at Same Price

A new order can be entered to a book at the same price level. The book will show an update:

New Order 3 @ 9427.00

Book Update Instruction

Book Update - Data Block

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	1	1 = change. Type of Market Data update action.
269	MDEntryType	0	0 = bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
1023	MDPriceLevel	1	Price level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	503	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9427.00	Price of the Market Data Entry.
346	NumberOfOrders	20	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Updated Book

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
20	503	9427.00	9428.00	40	2
34	750	9426.50	9428.50	600	35
25	400	9426.00	9429.00	850	55
14	300	9425.50	9429.50	350	21
1	400	9425.00	9430.00	150	12

5.4.2.4 New Best Price Entered

A new order can be entered to an existing book as a new best price. The data block indicates that a new order should be inserted at price level 1 (new best price). As a result, all orders on the book should be shifted down accordingly.

New Order 200 @ 9427.50

Book Update Instruction

Book Update - Data Block

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	0	0 = bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
1023	MDPriceLevel	1	Price level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	200	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9427.50	Price of the Market Data Entry.
346	NumberOfOrders	1	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Updated Book

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	200	9427.50	9428.00	40	2
20	503	9427.00	9428.50	600	35
34	750	9426.50	9429.00	850	55
25	400	9426.00	9429.50	350	21
14	300	9425.50	9430.00	150	12

5.5 Book Management Mechanics - Implied Book

CME Group provides a 2-deep best bid and ask in the market for each implied prices futures contract. **Implied book updates are denoted by the presence of tag 276-QuoteCondition = K (Implied).** Trade details and instrument status are provided in separate data blocks. The following table illustrates the types of information that can be displayed:

2-Deep Book

Bid		Ask	
Quantity	Price	Price	Quantity
100	9427.50	9428.00	40
200	9427.00	9428.50	100

CME Group maintains the Aggregate Depth view with the following data blocks:

- **Add** - to create/insert a new price at a specified price level (tag 279 MDUpdateAction=0)
- **Change** - change quantity of a price at a specified price level (tag 279 MDUpdateAction=1)
- **Delete** - remove a price at a specified price level (tag 279 MDUpdateAction=2)

An Aggregate book is built from a series of data blocks which indicate whether an entry is to be inserted (Add), changed (Change), or removed (Delete). All data blocks are issued for a specified entry type (tag 269), price (tag 270), and price level (tag 1023). The incremental instruction approach assumes the use of the Market Data Incremental Refresh message (tag 35-MsgType=X). The Bid and Ask sides are updated independently with separate data blocks. The practice of sending separate data blocks provides efficiencies by allowing only the bid or ask to be sent, based on which side has changed, rather than both sides.

CME Globex sends an add data block if there is a new price level. Client systems should then shift price levels down, and delete any price levels past 2-deep.

CME Globex sends a delete data block to remove a price level in the book. Client systems should shift prices below the data block up to the price level vacated by the deleted price level. If all levels in the book are full, an add data block will be sent to fill in the last price level.

The change data block is sent to update characteristics of a price level without changing the price itself, or impacting any other prices on the book. The change data block is sent to update the order quantity for a price level. The change data block is not sent when the price changes at a given price level.

If a trade occurs, CME Group will send a delete or change data block to update the book. The trade data block itself is not used to update the order book.

5.5.1 Basic Book Update Data Block

A FIX message sent to update the top of book will update one side only. The tags normally sent for a book update data block are:

Note: tag 346-NumberOfOrders is not included in this data block.

- tag 279-MDUpdateAction
- tag 269-MDEntryType
- tag 83-RptSeq
- tag 276-QuoteCondition
- tag 1023-MDPriceLevel
- tag 273-MDEntryTime
- tag 271-MDEntrySize
- tag 270-MDEntryPx
- tag 48-SecurityID
- tag 22-SecurityIDSource
- tag 336-TradingSessionID

5.5.2 Examples

This example shows how a book is built and updated. The book generally consists of a set of bid prices and ask prices, in which bids are descending and asks are ascending. The quantity is provided at each price level. For implied prices futures, a two-deep book is available.

The following table illustrates an initial market book.

2-Deep Book

Bid		Ask	
Quantity	Price	Price	Quantity
100	9427.50	9428.00	40
200	9427.00	9428.50	100

5.5.2.1 Quantity on Buy Side Modified

The quantity of an order can be modified. The book will show an update to the size displayed:

Buy 90 @ 9427.50

Book Update Instruction

Book Update - Data Block

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	1	1 = Change. Type of Market Data update action.
269	MDEntryType	0	0 = Bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
276	QuoteCondition		Space-delimited list of conditions describing a quote.
1023	MDPriceLevel	1	Price Level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	90	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9427.50	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Updated Book

Bid		Ask	
Quantity	Price	Price	Quantity
90	9427.50	9428.00	40
200	9427.00	9428.50	100

5.5.2.2 Entire Order Canceled and New Order Entered

An entire order can be canceled, which removes the top of book and a new order data block will be sent to fill the open price level 2. The book will show the removal of the first price level, followed by an addition to the second price level.

Note: Client systems should shift prices below the data block up to the price level vacated by the deleted price level. If all levels in the book are full, an add data block will be sent to fill in the last price level.

Cancel 90 @ 9427.50

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	2	2 = Delete. Type of Market Data update action.
269	MDEntryType	0	0 = Bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
276	QuoteCondition		Space-delimited list of conditions describing a quote.
1023	MDPriceLevel	1	Price Level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	90	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9427.50	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Add 80 @ 9426.50

Book Update Instruction

Book Update - Data Block 2

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = Add. Type of Market Data update action.
269	MDEntryType	0	0 = Bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
276	QuoteCondition		Space-delimited list of conditions describing a quote.
1023	MDPriceLevel	2	Price Level 2. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	80	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9426.50	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Updated Book

Bid		Ask	
Quantity	Price	Price	Quantity
200	9427.00	9428.00	40
80	9426.50	9428.50	100

5.5.2.3 New Order Entered at Same Price

A new order can be entered to a book at the same price level. The book will show an update:

New Order 3 @ 9427.00

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	1	1 = Change. Type of Market Data update action.
269	MDEntryType	0	0 = Bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
276	QuoteCondition		Space-delimited list of conditions describing a quote.
1023	MDPriceLevel	1	Price Level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	203	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9427.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Updated Book

Bid		Ask	
Quantity	Price	Price	Quantity
203	9427.00	9428.00	40
80	9426.50	9428.50	100

5.6 Consolidating Implied and Multiple-Depth Books into a Single Book

The data block for a multiple-depth book reports modifications to prices, quantities, and order count, and the data block for an implied prices book reports modifications to prices and quantities. The multiple-depth book should be used in conjunction with implied prices book to create an accurate book for all contracts with implied functionality. To create a consolidated book, the multiple-depth book and the implied book must be built and managed separately, then consolidated to reflect the current state of the market.

Multiple-Depth Book

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
19	500	9427.00	9428.00	40	2
34	750	9426.50	9428.50	600	35
25	400	9426.00	9429.00	850	55
14	300	9425.50	9429.50	350	21
10	200	9425.00	9430.50	150	12

Implied Book

Bid		Ask	
Quantity	Price	Price	Quantity
100	9427.50	9428.00	40
200	9427.00	9430.00	100

After building the multiple-depth book and implied books, the actual book can be built by merging the multiple-depth book and implied book tables.

Note: Order count is not displayed in the combined book.

Consolidated Book

Bid		Ask	
Quantity	Price	Price	Quantity
100	9427.50	9428.00	80
700	9427.00	9428.50	600
750	9426.50	9429.00	850
400	9426.00	9429.50	350
300	9425.50	9430.00	100

5.7 Book Management Mechanics - Top-of-Book

This section applies to CME Group products that are top-of-book. Client systems must determine the book-depth for an instrument from tag 264-MarketDepth in the Security Definition (tag 35-MessageType=d) message.

5.7.1 Overview

CME Group provides a multiple-depth book for several products. Client systems must determine the book-depth for an instrument from tag 264-MarketDepth in the Security Definition (tag 35-MessageType=d) Message.

CME Group provides the current best bid and ask (top-of-book) in the market for certain instruments. Client systems must determine the book-depth for an instrument from tag 264-MarketDepth in the Security Definition (tag 35-MessageType=d) Message. Trade details and instrument status are provided in separate data blocks. The following table illustrates the types of information that can be displayed:

1-deep Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	100	9427.50	9428.00	40	2

CME Group maintains a top-of-book view with the following data block:

- **Overlay** - add, modify, or delete an entry in the book. (tag 279 MDUpdateAction=5)

Note: When the best price changes, CME sends a single overlay instruction for price level 1.

The incremental instruction approach assumes the use of the Market Data Incremental Refresh message (tag 35-MessageType=X). The Bid and Ask sides are updated independently with separate data blocks. The practice of sending separate data blocks provides efficiencies by allowing only the bid or ask to be sent, based on which side has changed, rather than both sides.

5.7.1.1 Basic Book Update Data Block

A FIX message sent to update a top-of-book updates one side only. The tags normally sent for a top-of-book data block are:

- tag 279-MDUpdateAction
- tag 269-MDEntryType
- tag 83-RptSeq
- tag 1023-MDPriceLevel
- tag 273-MDEntryTime
- tag 271-MDEntrySize
- tag 270-MDEntryPx
- tag 346-NumberOfOrders

- tag48-SecurityID
- tag22-SecurityIDSource
- tag 336-TradingSessionID

5.7.1.2 Indexing the Entry

Top-of-book entries are referenced using a composite index which consists of the Security Description, Tag 269-MDEntryType (bid/ask) and Tag 270-MDEntryPx. This set of fields acts as a composite key that allows the entry (bid or ask) to be accessed and subsequently, updated or deleted.

5.7.2 Examples

This example shows how a book is built and updated. The book generally consists of a set of bid prices and ask prices, in which bids are descending and asks are ascending. The quantity and order count are provided.

The following table illustrates an initial market book.

Top-of-Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	100	9427.50	9428.50	40	2

5.7.2.1 Quantity on Buy Side Modified

The quantity of an order can be modified. The book will show an update to the size displayed:

Buy 90 @ 9427.50

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	5	5 = overlay. Type of Market Data update action.
269	MDEntryType	0	0 = bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
1023	MDPriceLevel	1	Price level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	90	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9427.50	Price of the Market Data Entry.
346	NumberOfOrders	1	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Top-of-Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	90	9427.50	9428.50	40	2

5.7.2.2 Entire Order Canceled

An entire order can be canceled, which removes the top of book. The book will show an update:

Cancel 90 @ 9427.50

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	5	5 = overlay. Type of Market Data update action.
269	MDEntryType	0	0 =bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
1023	MDPriceLevel	1	Price level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	0	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx		Price of the Market Data Entry.
346	NumberOfOrders	0	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Top-of-Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
			9428.50	40	2

5.7.2.3 New Order Entered

A new order can be entered to a book. The book will show an update:

New Order 10 @ 9428.00

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	5	5 = overlay. Type of Market Data update action.
269	MDEntryType	0	0 =bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
1023	MDPriceLevel	1	Price level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	10	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9428.00	Price of the Market Data Entry.
346	NumberOfOrders	1	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Top-of-Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	10	9428.00	9428.50	40	2

5.7.2.4 New Order Entered at Same Price

A new order can be entered to a book at the same price level. The book will show an update:

New Order 3 @ 9428.00

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	5	5 = overlay. Type of Market Data update action.
269	MDEntryType	0	0 = bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
1023	MDPriceLevel	1	Price level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	13	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9428.00	Price of the Market Data Entry.
346	NumberOfOrders	2	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.

Top-of-Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
2	13	9428.00	9428.50	40	2

5.8 Book Management Mechanics - Indicative Prices

5.8.1 Overview

CME Group provides the current best resolved bid and ask theoretical book (top of book only) from a group of market makers for indicative prices. The following table illustrates the types of information that can be displayed:

1-deep Book - Indicative Prices

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	100	9427.50	9428.00	40	2

CME Group maintains the 1-deep book view with the following data block:

- **Overlay** - add, modify, or delete an entry in the book. (tag 279 MDUpdateAction=5)

Note: When the best price changes, CME Group sends a single overlay instruction for price level 1.

The incremental instruction approach assumes the use of the Market Data Incremental Refresh message (tag 35-MsgType=X). The Bid and Ask sides are updated independently with separate data blocks. The practice of sending separate data blocks provides efficiencies by allowing only the bid or ask to be sent, based on which side has changed, rather than both sides.

5.8.1.1 Basic Book Update Data Block

A FIX message sent to update the 1-deep book updates one side only. The tags normally sent for a 1-deep book data block are:

- tag 279-MDUpdateAction
- tag 269-MDEntryType
- tag 83-RptSeq
- tag 1070-MDQuoteType
- tag 1023-MDPriceLevel
- tag 273-MDEntryTime
- tag 271-MDEntrySize
- tag 270-MDEntryPx
- tag 346-NumberOfOrders
- tag48-SecurityID
- tag22-SecurityIDSource

5.8.1.2 Indexing the Entry

1-deep book entries are referenced using a composite index which consists of the Security Description, Tag 269-MDEntryType (bid/ask) and Tag 270-MDEntryPx. This set of fields acts as a composite key that allows the entry (bid or ask) to be accessed and subsequently, updated or deleted.

5.8.2 Examples

This example shows how a book is built and updated. The book generally consists of a set of bid prices and ask prices, in which bids are descending and asks are ascending. The quantity and order count are provided at each price level. For indicative prices, only top-of-book is available.

The following table illustrates an initial market book.

1-deep Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	100	9427.50	9428.50	40	2

5.8.2.1 Quantity on Buy Side Modified

The quantity of an indicative price can be modified. The book will show an update to the size displayed:

Buy 90 @ 9427.50

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	5	5 = overlay. Type of Market Data update action.
269	MDEntryType	0	0 = bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
1070	MDQuoteType	0	0 = Indicative. Identifies the type of quote.
1023	MDPriceLevel	1	Price level 1. Position in the book.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	90	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9427.50	Price of the Market Data Entry.
346	NumberOfOrders	1	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

1-deep Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	90	9427.50	9428.50	40	2

5.8.2.2 Entire Indicative Price Canceled

An entire indicative price can be canceled, which removes the top of book. The book will show an update:

Cancel 90 @ 9427.50

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	5	5 = overlay.
269	MDEntryType	0	0 =bid.
83	RptSeq		Sequence number per Instrument update.
1070	MDQuoteType	0	0 = Indicative. Identifies the type of quote.
1023	MDPriceLevel	1	Price level 1.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	0	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx		Price of the Market Data Entry.
346	NumberOfOrders	0	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

1-deep Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
			9428.50	40	2

5.8.2.3 New Indicative Price Entered

A new indicative price can be entered to a book. The book will show an update:

New Order 10 @ 9428.00

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	5	5 = overlay.
269	MDEntryType	0	0 =bid.
83	RptSeq		Sequence number per Instrument update.
1070	MDQuoteType	0	0 = Indicative. Identifies the type of quote.
1023	MDPriceLevel	1	Price level 1.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	10	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9428.00	Price of the Market Data Entry.
346	NumberOfOrders	1	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

1-deep Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
1	10	9428.00	9428.50	40	2

5.8.2.4 New Indicative Price Entered at Same Price

A new indicative price can be entered to a book at the same price level. The book will show an update:

New Order 3 @ 9428.00

Book Update Instruction

Book Update - Data Block 1

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	5	5 = overlay.
269	MDEntryType	0	0 = bid.
83	RptSeq		Sequence number per Instrument update.
1070	MDQuoteType	0	0 = Indicative. Identifies the type of quote.
1023	MDPriceLevel	1	Price level 1.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	13	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9428.00	Price of the Market Data Entry.
346	NumberOfOrders	2	Number of orders in the market.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource		Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

1-deep Book - Best Bid/Ask

Bid			Ask		
Order Count	Quantity	Price	Price	Quantity	Order Count
2	13	9428.00	9428.50	40	2

6. Real Time Statistics (Market Behavior Events)

There are a number of statistics (market data events) which are related to changes in a book but are not used to update the book. The following artifacts fit this category: last best price, trade, high/low trade price, best high bid, and best low ask, price, and pre-opening statistics. These events describe the behavior of the market and allow a user to know when the market is moving in a certain direction and provide historical information on how the market has performed.

Note: Multiple data blocks may be sent in the same message. To determine the number of data blocks you will receive in the message, refer to tag 268-NoMDEntries. Within the message, data blocks may be for different instruments or entry types (book update, statistics, or trades).

6.1 Pre-Opening Statistics

Pre-opening statistics are used to indicate simulated buy or sell, stop spike, and pre-opening prices. Prior statistics are also available.

FIX Syntax for Prior - Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	M	M = prior. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
336	TradingSessionID		Identifier for Trading Session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.

FIX Syntax for Opening - Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	4	4 = opening price. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

FIX Syntax for Simulated Sell - Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	E	E = simulated sell. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
271	MDEntrySize		Quantity or volume represented by the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

FIX Syntax for Simulated Buy - Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	F	F = simulated buy. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
271	MDEntrySize		Quantity or volume represented by the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

6.2 Last Best Price

The last best price data block indicates that a new best bid or new best ask price has occurred, and explicitly conveys events which affect the top of the book. **Last Best Price data blocks are denoted by the presence of tag 276-QuoteCondition = C (Exchange Best).**

In certain cases, such as an order submission immediately followed by an order cancellation, the top of the book can be bettered without resulting in a new top of book data block being sent. This is due to the transitory nature of the event. However, last best price is used to capture this information and convey it to the market.

FIX Syntax for Last Best Price (Bid or Ask) - Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	0 or 1	0 = bid and 1 = ask. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

FIX Syntax for Last Best Price (Bid or Ask) - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
276	QuoteCondition	C	C = exchange best. Space-delimited list of conditions describing a quote.

Note: Tag 277-TradeCondition is not sent for this data block.

6.3 Trade

The trade data block is sent when a trade occurs to provide volume and trade statistics. The following sections detail various trade types and their respective data blocks that demonstrate the use of Tag 277-TradeCondition.

It should be noted that Tag 277-TradeCondition is present and set to 1 when the reported trade price is an assigned price by CME Globex. This occurs for a leg price in a spread transaction.

Tag 277-TradeCondition is not present when the reported trade price is not assigned by CME Globex.

When Tag 277-TradeCondition is not present or Tag 277-TradeCondition=E (opening trade), then the data represents the last trade.

Tag 269-MDEntryType=2 (trade) is not used to update the view of the book.

The trade data block may indicate whether the aggressor of the trade is on the buy side or the sell side. A trade data block may also be flagged as either the beginning or the end of a Globex event.

6.3.1 Example 1 - Two Outright Orders Trading

The following example illustrates the trade data block that is sent when a buy and sell match on GEZ2.

FIX Syntax for GEZ2 Outright Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.

FIX Syntax for GEZ2 Outright Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.
5799	MatchEventIndicator	1 or 2	1 = Beginning and 2 = End. Indicates the beginning or the end of a match event. If there is no value present, then the message is not at the beginning or the end of a Globex event. Note: For this release, only 1 is a valid value. 2 will become a valid value in a future release.

Note: Tag 277-TradeCondition is not sent for this data block.

6.3.2 Example 2 - Two Spread Orders Trading (Not Implied)

Several trade data blocks are sent for a spread. A trade may be executed in an off-market manner, and in this case the trade does not update any of the trade statistics.

The following example illustrates the data blocks that are sent for a match event involving one order on ESM8-ESU8 against one order on ESM8-ESU8.

The resulting trade data blocks that will occur are:

- One trade data block for ESM8-ESU8 (tag 277-TradeCondition is not sent)
- One trade data block for ESM8 (tag 277-TradeCondition = 1)
- One trade data block for ESU8 (tag 277-TradeCondition = 1)

FIX Syntax for ESM8-ESU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.

FIX Syntax for ESM8-ESU8 Trade - Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	1360.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.
5799	MatchEventIndicator	1 or 2	1 = Beginning and 2 = End. Indicates the beginning or the end of a match event. If there is no value present, then the message is not at the beginning or the end of a Globex event. Note: For this release, only 1 is a valid value. 2 will become a valid value in a future release

Note: Tag 277-TradeCondition is not sent for this data block.

FIX Syntax ESM8 Trade - Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.

FIX Syntax ESM8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	1360.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
277	TradeCondition	1	Space-delimited list of conditions describing a trade. <ul style="list-style-type: none"> 1 = Price calculated by Globex. Note: Tag 277- TradeCondition is not sent for a last best price data block.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.

FIX Syntax ESU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	1360.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.

FIX Syntax ESU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
277	TradeCondition	1	Space-delimited list of conditions describing a trade. • 1 = Price calculated by Globex. Note: Tag 277 - TradeCondition is not sent for a last best price data block.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.

6.3.3 Example 3 - Spread Order Trading Against Two Outright Orders (Implied)

The following example illustrates the data blocks that are sent for a match event involving one order on GEM8-GEU8 against one order on GEM8 and one order on GEU8.

The resulting trade data blocks that will occur are:

- One trade data block for GEM8-GEU8 (tag 277-TradeCondition is not sent)
- One trade data block for GEM8 (tag 277-TradeCondition is not sent)
- One trade data block for GEU8 (tag 277-TradeCondition is not sent)

FIX Syntax for GEM8-GEU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.

FIX Syntax for GEM8-GEU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.
5799	MatchEventIndicator	1 or 2	1 = Beginning and 2 = End. Indicates the beginning or the end of a match event. If there is no value present, then the message is not at the beginning or the end of a Globex event. Note: For this release, only 1 is a valid value. 2 will become a valid value in a future release

Note: Tag 277-TradeCondition is not sent for this data block.

FIX Syntax GEM8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.

FIX Syntax GEM8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.

Note: Tag 277-TradeCondition is not sent for this data block.

FIX Syntax GEU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.

FIX Syntax GEU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.

Note: Tag 277-TradeCondition is not sent for this data block.

6.3.4 Example 4 - Butterfly Order Trading Against a Calendar Order and Two Outright Orders (Implied)

The following example illustrates the data blocks that are sent for a match event involving one order on GE:BF M8-U8-Z8 against one order on GEM8-GEU8, one order on GEU8, and one order on GEZ8.

The resulting trade data blocks that will occur are:

- One trade data block for GE:BF M8-U8-Z8 (tag 277-TradeCondition is not sent)
- One trade data block for GEM8-GEU8 (tag 277-TradeCondition is not sent)
- One trade data block for GEM8 (tag 277-TradeCondition = 1)
- One trade data block for GEU8 (tag 277-TradeCondition = 1)
- One trade data block for GEU8 (tag 277-TradeCondition is not sent)
- One trade data block for GEZ8 (tag 277-TradeCondition is not sent)

FIX Syntax GE:BF M8-U8-Z8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.

FIX Syntax GE:BF M8-U8-Z8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.
5799	MatchEventIndicator	1 or 2	1 = Beginning and 2 = End. Indicates the beginning or the end of a match event. If there is no value present, then the message is not at the beginning or the end of a Globex event. Note: For this release, only 1 is a valid value. 2 will become a valid value in a future release

Note: Tag 277-TradeCondition is not sent for this data block.

FIX Syntax GEM8-GEU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.

FIX Syntax GEM8-GEU8 Trade - Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.

Note: Tag 277-TradeCondition is not sent for this data block.

FIX Syntax GEM8 Trade - Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.

FIX Syntax GEM8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
277	TradeCondition	1	Space-delimited list of conditions describing a trade. • 1 = Price calculated by Globex. Note: Tag 277 - TradeCondition is not sent for a last best price data block.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.

FIX Syntax for GEU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
277	TradeCondition	1	Space-delimited list of conditions describing a trade. • 1 = Price calculated by Globex. Note: Tag 277 - TradeCondition is not sent for a last best price data block.

FIX Syntax for GEU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.

FIX Syntax GEU8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.

Note: Tag 277-TradeCondition is not sent for this data block.

FIX Syntax GEZ8 Trade - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0 or 2	0 = new and 2 = delete. Type of Market Data update action.
269	MDEntryType	2	2 = trade. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize	5	Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.
1020	TradeVolume		Total traded volume since the beginning of the session.
274	TickDirection		Direction of the tick. If there is no value present, then there is no change.
451	NetChgPrevDay		Net change from previous day's closing price versus last traded price.
5797	AggressorSide	1 or 2	1 = Buy and 2 = Sell. Indicates the side that is the aggressor of the trade. If there is no value present, then there is no aggressor.

Note: Tag 277-TradeCondition is not sent for this data block.

6.4 Session High and Low Trade Price

The high trade price data block is sent for a trade event that has produced the highest trade price for the current session. Likewise, the low trade price data block indicates that a trade event has produced the lowest trade price for a given session. High and low trade prices are helpful in tracking market trends. They also provide historical information for the current session regarding market behavior.

CME Group will provide high and low prices only as a response to the event that produces them.

FIX Syntax for Session High Trade Price - Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	7	7 = session high. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

FIX Syntax for Session Low Trade Price- Market Data Incremental Refresh (tag 35-MsgType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	8	8 = session low. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

6.5 Best High Bid and Best Low Ask

The best high bid and best low ask prices are used to indicate the highest bid and lowest ask prices of the session. These prices are useful in tracking market behavior. A best high bid and best low ask may be different from the high trade price described above if an arriving order is given a price better than the speci-

fied limit order. An order to buy at 9751 may trade at 9750 if a resting sell order is already on the book at 9750. In this case, the best high bid price would be higher than the high trade price.

FIX Syntax for Best High Bid - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	N	N = session high bid. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize		Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

FIX Syntax for Best Low Ask - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	O	O = session low ask. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
271	MDEntrySize		Quantity or volume represented by the Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

6.6 Closing and Settlement Price

The price data block is sent to update previous day adjustment and settlement price. This data block is useful for obtaining the settlement price and the previous day's adjusted closing price and is sent after the close of the trading session.

FIX Syntax for Previous Day Adjustment - Market Data Incremental Refresh (tag 35-MessageType = X)

Tag Number	Tag Name	Value	Description
279	MDUpdateAction	0	0 = new. Type of Market Data update action.
269	MDEntryType	5	5 = closing price and 6 = settlement price. Type of Market Data entry.
83	RptSeq		Sequence number per Instrument update.
273	MDEntryTime		Time of Market Data Entry.
270	MDEntryPx	9550.00	Price of the Market Data Entry.
48	SecurityID		Unique instrument ID as qualified by the exchange per tag 22-SecurityIDSource.
22	SecurityIDSource	8	Identifies source of tag 48-SecurityID value. This value is always 8 for CME and is required if tag 48-SecurityID is specified.

7. CME Globex Pricing

This section describes the pricing and tick convention used for instruments available on CME Globex. The CME Globex price is the format used in all price tags in order entry iLink and market data MDP interfaces. One method for obtaining the CME Globex price is from tag 270-MDEntryPX of the Market Data Incremental Refresh tag 35-MsgType=X message.

7.1 Tick Convention

The tick size is the minimum fluctuation in price allowed for a futures or options contract during a trading session as specified by the contract terms of CME Group. There are two methods available for determining tick size. The first method is to locate the tick size from the Variable Tick Table (VTT) if the instrument is eligible. If the instrument is not eligible for the VTT, the second method is to obtain the tick size directly from the message.

7.1.1 Calculating Tick Size for a VTT Eligible Instrument Using the Security Definition (tag 35-MsgType=d) Message

In a repeating group, when tag 871-InstAttribType is equal to 23, it is VTT eligible and will be coupled with tag 872-InstAttribValue. The following process should be performed to calculate the tick size for a VTT eligible instrument.

1. Obtain the Security Definition (tag 35-MsgType=d) message.
2. If the message contains the tag 871-InstAttribType=23, the instrument is VTT eligible. If tag 871-InstAttribType is not present or not equal to 23, refer to "Calculating Tick Size for a Non-VTT Eligible Instrument Using the Security Definition (tag 35-MsgType=d) Message" on Page 79.
3. The tag 872-InstAttribValue will contain the VTT index code as shown in the "Variable Tick Table" on Page 78. Find the row that contains both the VTT Index Code and the current price. The corresponding tick size is listed in the last column of the row.

Example for VTT Eligible Instruments

Below is an example that demonstrates how to calculate the tick size step-by-step.

1. Obtain the Security Definition (tag 35-MsgType=d) message.

Message Type: Security Definition (tag 35-MsgType=d)

```
1128=8 35=d 49=CME 34=9647 52=20080608171603359 15=USD 22=8 48=300998 55=OS 107=SPM8
P1750 200=200806 202=175000 207=XCME 461=OPAFPS 462=5 562=1 711=1 311=[N/A] 305=8
309=2536 827=0 864=2 865=5 866=20060707 1145=202500000 865=7 866=20080619
1145=202500000 870=5 871=23 872=01 871=24 872=1 871=24 872=2 871=24 872=3 871=24 872=14
947=USD 1140=9999 1141=1 1022=GBX 264=1 1143=600 1146=0 1147=0 1148=5 1149=9999900
1150=36530 1151=SP 1180=8 9787=0.01 9850=5
```


2. Verify that the message contains the tag 871-InstAttribType=23 and is therefore a VTT eligible instrument. If the tag is not present, refer to “Calculating Tick Size for a Non-VTT Eligible Instrument Using the Security Definition (tag 35-MsgType=d) Message” on Page 79.

871=23

3. The tag 872-InstAttribValue will contain the VTT code as shown in the “Variable Tick Table” on Page 78.

872=01

4. Since tag 872-InstAttribValue is 01 for a market price of 510 ($P=510 > 500$), the tick size for the contract is 10. If the market moves and the market price changes to 480 ($P=480, -500 \leq P \leq 500$), the tick size of the contract would change to 5.

7.1.2 Variable Tick Table

Variable Tick Table Index Code	Current CME Globex Price (P)	CME Globex Tick Size
01	$P < -500$	10
01	$-500 \leq P \leq 500$	5
01	$P > 500$	10
02	$0 \leq P \leq 5$	0.5
02	$P > 5$	1
03	$0 \leq P \leq 10$	1
03	$P > 10$	2
04	$P < -500$	25
04	$-500 \leq P \leq 500$	5
04	$P > 500$	25
05	Any	1
06	Any	2
07	Any	1
09	The 09 Variable Tick Index Code indicates that the spread is eligible to trade in 0.5 (half-tick) increments when any of the leg settlement prices are equal to or less than 5.0 full ticks.	
10	$P < -300$	25
10	$-300 \leq P \leq 300$	5
10	$P > 300$	25
11	$P < -300$	10
11	$-300 \leq P \leq 300$	5
11	$P > 300$	10
12	$P < -5$	0.50
12	$-5 \leq P \leq 5$	0.25
12	$P > 5$	0.50

7.1.3 Calculating Tick Size for a Non-VTT Eligible Instrument Using the Security Definition (tag 35-MsgType=d) Message

If the instrument is non VTT-eligible, use the tick value found in tag 969-MinPriceIncrement.

1. Obtain the Security Definition (tag 35-MsgType=d) message.
2. If the Security Definition (tag 35-MsgType=d) message does not contain the tag 871-InstAttribType=23 in the repeating group, the instrument is not VTT eligible.
3. The tick size will be contained in tag 969-MinPriceIncrement.

Example for non-VTT Eligible Instruments

Below is an example that demonstrates how to calculate the tick size step-by-step.

1. Obtain the Security Definition (tag 35-MsgType=d) message:

Message type: Security Definition (tag 35-MsgType=d):

```
1128=8 9=425 35=d 49=CME 34=13591 52=20080522221850254 15=USD 22=8 48=806800 55=GE
107=GEM8 200=200806 207=XCME 461=FFDXSX 462=14 562=1 827=2 864=2 865=5 866=19980617
1145=213000000 865=7 866=20080616 1145=100000000 870=5 871=24 872=1 871=24 872=2 871=24
872=3 871=24 872=4 871=24 872=5 947=USD 969=0.5 1140=30000 1141=1 1022=GBX 264=5 1142=A
1143=10.00 1144=3 1146=12.5000 1148=9559.50 1149=9959.50 1150=9759.50 1151=GE 1180=9
9787=0.01 10=216
```

2. If the Security Definition (tag 35-MsgType=d) message does not contain the tag 871-InstAttribType=23 repeating group, the instrument is not VTT eligible. If it contains tag 87-InstAttribType=23, refer to “Calculating Tick Size for a VTT Eligible Instrument Using the Security Definition (tag 35-MsgType=d) Message” on Page 76.
3. The tick size will be contained in tag 969-MinPriceIncrement.

969=0.5

In this example, the tick size is 0.5.

7.2 Display

Tag 9787-DisplayFactor is multiplied by both CME Globex Tick and Price Conventions to equal the displayed tick and price. The following table is a sample display of CME Globex ticks and prices.

Contract	CME Globex Tick (tag 969- MinPriceIncrement)	CME Globex Price (tag 270- MDEntryPx)	Display Factor (tag 9787- DisplayFactor)	Display Tick	Display Price
ESH6	25	113700	0.01	.25	1137.00
GEM6	.5	9886.5	0.01	.005	98.865

CME Globex Price x Display Factor = Display Price

CME Globex Tick x Display Factor = Display Tick

The resulting display of price and tick are suggested by CME Group but are not required.

8. Recovery

CME Group offers several options for recovering missed messages or synchronizing client systems to the latest state: TCP Replay, Market Recovery, and Instrument Replay. Instrument level sequencing and natural refresh can be utilized to supplement the recovery process.

This section contains an overview of the various recovery methods and supplemental methods.

- “Recovery Feeds” on Page 81 - this section describes the following: TCP Replay, Market Recovery, and Instrument Replay
- “Using the Incremental Market Data Feed to Determine State” on Page 86 - the methods in this section provide supplemental methods to the recovery process, however they are not guaranteed to completely recover missed messages or synchronize client systems to the latest state.

There is also a section that describes at a high level the process to recover in various situations based on the recovery and supplemental methods.

- “Recovering Data - Process” on Page 92

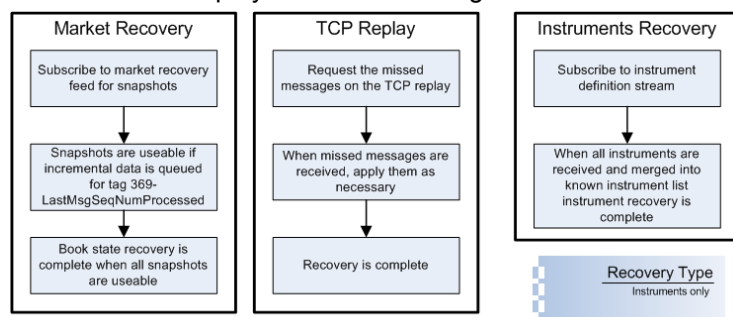
Note: CME Group strongly recommends that client systems process both the A and B incremental market data feeds.

8.1 Recovery Feeds

Message loss is detected using the decoded FIX message sequence numbers (tag 34-MsgSeqNum). The message sequence number is an incrementing number. If a gap is detected between messages in tag 34-MsgSeqNum, this indicates a message has been missed. It should be assumed at this point that all books maintained in the client system may no longer have the correct, latest state maintained by CME Group. Client systems must resynchronize all books to the latest state maintained by CME Group, and determine whether any new instrument definitions were published. During this synchronization process, all books are initially assumed to be in an incorrect state and are recovered during the synchronization process.

The following options are offered by CME Group to recover missed messages or synchronize client systems to the latest state:

- “TCP Replay Overview” on Page 81
- “Market Recovery Overview” on Page 85
- “Instrument Replay Overview” on Page 86



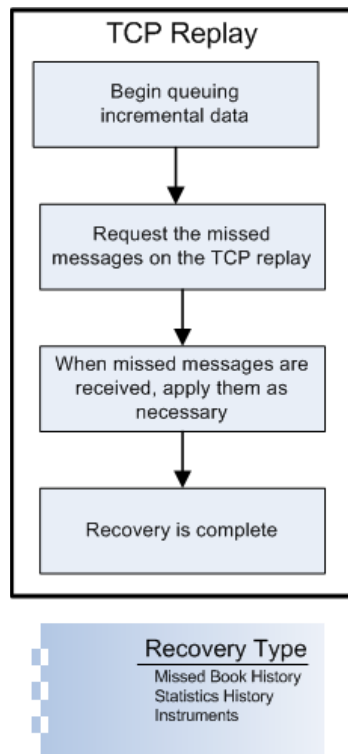
8.1.1 TCP Replay Overview

TCP replay should be used for small-scale data recovery (refer to “Implementation Considerations” on Page 82 for additional information). Client systems can recover specific messages that were missed using the sequence number and the TCP historical replay component. This method recovers all missed messages. **CME Group logs IP addresses, ports, and passwords of the originator and the Channel ID and range of the requested sequence number.**

The TCP historical replay component allows you to request a replay of a set of messages already published on the UDP Incremental Market Data Channel or a snapshot message. The request specifies messages to be replayed. The request uses the FIX Market Data Request message (35=V). Possible responses are the Logout (35=5) message and the list of messages to be replayed. If the Logout (35=5) message is sent, tag 58-Text will list the reason for the logout.

This type of request is sent through a new TCP connection established by the customer. When making the request, the channel and sequence number are specified. The responses are sent by CME Group through this same connection and the connection is then closed by CME Group once the resend is complete. All requests from the client system to CME Group are in FIX format. All responses from CME Group to the client system are FIX/FAST encoded (including the reject response). The TCP Replay contains a stop-bit delimited block size represented as an unsigned integer in FAST. This stop-bit is used instead of FIX’s current 2 byte separator. Replay is limited in the number of messages that can be requested.

Note: Client systems should queue real-time data until all missed data is recovered. The recovered data should then be applied prior to queued data.



8.1.1.1 Implementation Considerations

The following are limitations to consider prior to implementing TCP replay functionality:

- TCP requests and responses are via TCP.
- Resend request messages and other inbound messages (i.e. heartbeat) are in FIX format, outbound response messages are FAST encoded.
- There is a 24 hour time limit on the number of messages available via the TCP replay functionality.
- The maximum number of messages that can be requested in one resend request message is 2000.

Note: If you want to request more than 2000 messages, you will need to send multiple resend request messages.

8.1.1.2 Process

1. Customer establishes a TCP connection.
 - Refer to the configuration file for TCP ip and port information. For additional information on the configuration file, refer to “Services - Template Dissemination and Market Data Configuration” on Page 3.
2. Customer sends Logon (tag 35-MsgType=A) message to CME Group.

Note: Customer Username and Password are verified. If the Username and Password are incorrect, a Logout (tag 35-MsgType=5) will be sent.

3. CME Group sends Logon (tag 35-MsgType=A) message to the customer.

Note: CME Group will send a Logout (tag35-MsgType=5) message if a Market Data Request (tag35-MsgType=V) is not received in 5 seconds.

4. Customer sends Market Data Request (tag 35-MsgType=V) message to CME Group.

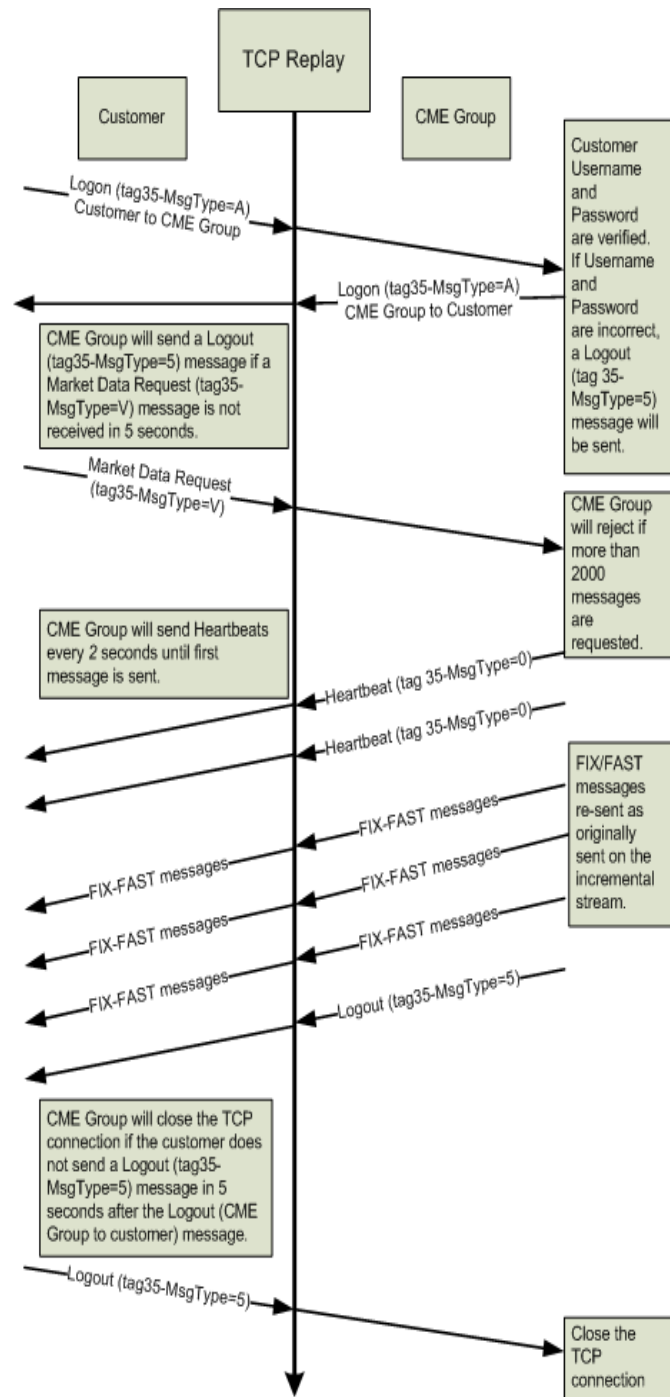
Note: Client systems must indicate the channel ID (tag 1180-ApplFeedID) and sequence numbers (tag 1182-ApplBeginSeqNo and tag 1183-ApplEndSeqNo) in the Market Data Request (tag 35-MsgType=V) message.

5. CME Group sends Heartbeat (tag35-MsgType=0) messages to customer every 2 seconds until the first recovery message is sent.
6. CME Group sends FIX/FAST recovery messages that were requested by the customer in the Market Data Request (tag35-MsgType=V) message.
7. CME Group sends a Logout (tag35-MsgType=5) message to the customer.

Note: CME Group will close the TCP connection if the customer does not send a Logout (tag 35-MsgType=5) message within 5 seconds from the time CME Group sends a Logout (tag35-MsgType=5) message.

8. Customer sends a Logout (tag 35-MsgType=5) message to CME Group.
9. CME Group closes the TCP connection.

Note: The username and password for the TCP Replay component for the New Release and Certification testing environments are both "CME" (do not include quotes). The username and password for the production environment are provided after certification is completed. For more information, contact your CME Globex account manager.

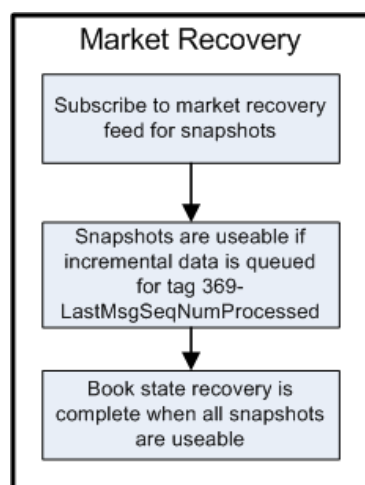


8.1.2 Market Recovery Overview

This recovery method should be used for large-scale data recovery (i.e. major outage or late joiners) to synchronize client systems to the latest state maintained by CME Group. Client systems can use the Market Recovery feed on each channel to determine the state of each book in every channel. Each Market Recovery feed constantly loops and sends the Market Data Snapshot Full Refresh (tag 35=MsgType = W) message. The Market Recovery feed is known to be valid as of a sequence number on the Incremental Market Data feed, which is found in tag 369-LastMsgSeqNumProcessed. This sequence number (tag 369-LastMsgSeqNumProcessed) is found on each Market Data Snapshot Full Refresh (tag 35=W) message. If the Market Recovery method is used, client systems also need to subscribe to the Instrument Definition feed to determine whether any new instruments were defined. Client systems will recover the most recent statistics on the Market Recovery feed. Any intermediary statistics will not be recovered.

Note: Client systems should queue real-time data until all snapshot data is retrieved. The queued data should then be applied as necessary.

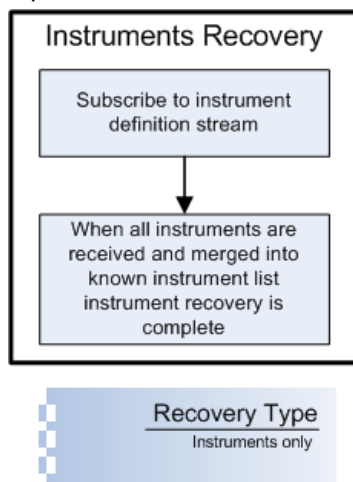
Note: CME strongly recommends that the Market Recovery feeds be used for recovery purposes only. Once client systems have retrieved recovery data, client systems should stop listening to the Market Recovery feeds.



Recovery Type
Book Snapshot
Statistics Snapshot

8.1.3 Instrument Replay Overview

Client systems can use the Instrument Replay feed on each channel to determine if any new instruments were defined through the Security Definition (tag 35-MsgType=d) message. The Instrument Replay feed constantly replays the current week's Security Definition (tag-MsgType=d) messages, and will reflect any additions, modifications, or deletes to CME Group contracts. Client systems should wait to begin the recovery process until tag 34-MsgSeqNum is 1 and tag 911-TotNumReports are at the last indicated number of reports is returned. CME Group does not guarantee the order of the messages being sent.



8.2 Using the Incremental Market Data Feed to Determine State

In addition to the recovery methods that are offered by CME Group, there are additional mechanisms that client systems can utilize to enhance the recovery process.

- “Instrument Level Sequencing” on Page 87
- “Natural Refresh” on Page 88

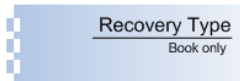
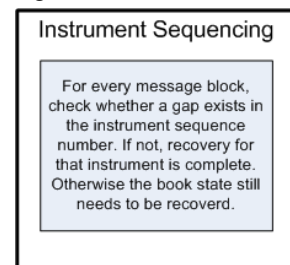
8.2.1 Instrument Level Sequencing

Market Data Incremental Refresh messages (tag 35=X) contain instrument sequence numbers (tag 83-RptSeq), in addition to message sequence numbers (tag 34-MsgSeqNum). Every repeating group instance of a market data entry contains an incrementing sequence number (tag 83-RptSeq) that is associated with the instrument for which data is present in the block. Instrument level sequencing can be used to identify which instruments you have not missed messages for, and apply during the TCP Replay method.

Client systems can keep track of the instrument sequence number (tag 83-RptSeq) for every instrument by inspecting incoming data and determining whether there is a gap in the instrument sequence number (tag 83-RptSeq).

- If there is a gap in the instrument sequence number (tag 83-RptSeq), it indicates that data was missed for the instrument when message loss occurred.
- If there is no gap, the data can be used immediately, and it also indicates that the book for this instrument still has a correct, current state.

It is likely that if only a small number of messages have been missed, there will be data in subsequent messages which are not affected by the missing data. If there are 10 instruments in a channel, for example, and the missed messages contain data for 2 of these instruments, any subsequent messages containing data about the other instruments are still valid.



8.2.2 Natural Refresh

The client system must track the state of the book at all times with the FIX Market Data Incremental Refresh message (tag 35-MessageType=X) book update data blocks. It is possible, though not guaranteed, that a set of these book update data blocks can be used to construct the current, correct state of a book without prior book state knowledge. While client systems wait for the recovery of missing data, they may opt to also use a natural refresh algorithm to recover book state prior to recovering the lost data. Natural refresh can also be used to re-instate the top-of-book. Natural refresh should be used for 5-deep or 2 deep book updates only in conjunction with market recovery or TCP replay. Prior to beginning a natural refresh, the entire book should be emptied. Natural refresh assumes no prior knowledge of book state.

The following example illustrates how natural refresh occurs. Note that in this example, using instructions 4 and 5, the top two levels of the offer book are known to be good. This portion of the book can then be displayed or used.

Note: Natural Refresh is not guaranteed and should not be considered a definitive substitute for recovering lost data.

Example:

In this example, the book should be emptied:

Top of Book - Best Bid/Ask

Bid		Ask	
Quantity	Price	Price	Quantity

Consider the following received book update data blocks for the offers for a given instrument with a 5-deep book.

1. Insert at book level 3, price 50, quantity 10:
 - tag 279-MDUpdateAction = 0 (new)
 - tag 1023-MDPriceLevel = 3
 - tag 271-MDEntrySize = 10
 - tag 270-MDEntryPx = 50

Top of Book - Best Bid/Ask

Bid		Ask	
Quantity	Price	Price	Quantity
		50	10

2. Insert at book level 3, price 40, quantity 10:

- tag 279-MDUpdateAction = 0 (new)
- tag 1023-MDPriceLevel = 3
- tag 271-MDEntrySize = 10
- tag 270-MDEntryPx = 400

Top of Book - Best Bid/Ask

Bid		Ask	
Quantity	Price	Price	Quantity
		40	10
		50	10

3. Insert at book level 3, price 30, quantity 10:

- tag 279-MDUpdateAction = 0 (new)
- tag 1023-MDPriceLevel = 3
- tag 271-MDEntrySize = 10
- tag 270-MDEntryPx = 30

Top of Book - Best Bid/Ask

Bid		Ask	
Quantity	Price	Price	Quantity
		30	10
		40	10
		50	10

4. Update book level 1, price 10, quantity 10:

- tag 279-MDUpdateAction = 1 (change)
- tag 1023-MDPriceLevel = 1
- tag 271-MDEntrySize = 10
- tag 270-MDEntryPx = 10

Top of Book - Best Bid/Ask

Bid		Ask	
Quantity	Price	Price	Quantity
		10	10
		30	10
		40	10
		50	10

5. Update book level 2, price 20, quantity 10:

- tag 279-MDUpdateAction = 1 (change)
- tag 1023-MDPriceLevel = 2
- tag 271-MDEntrySize = 10
- tag 270-MDEntryPx = 20

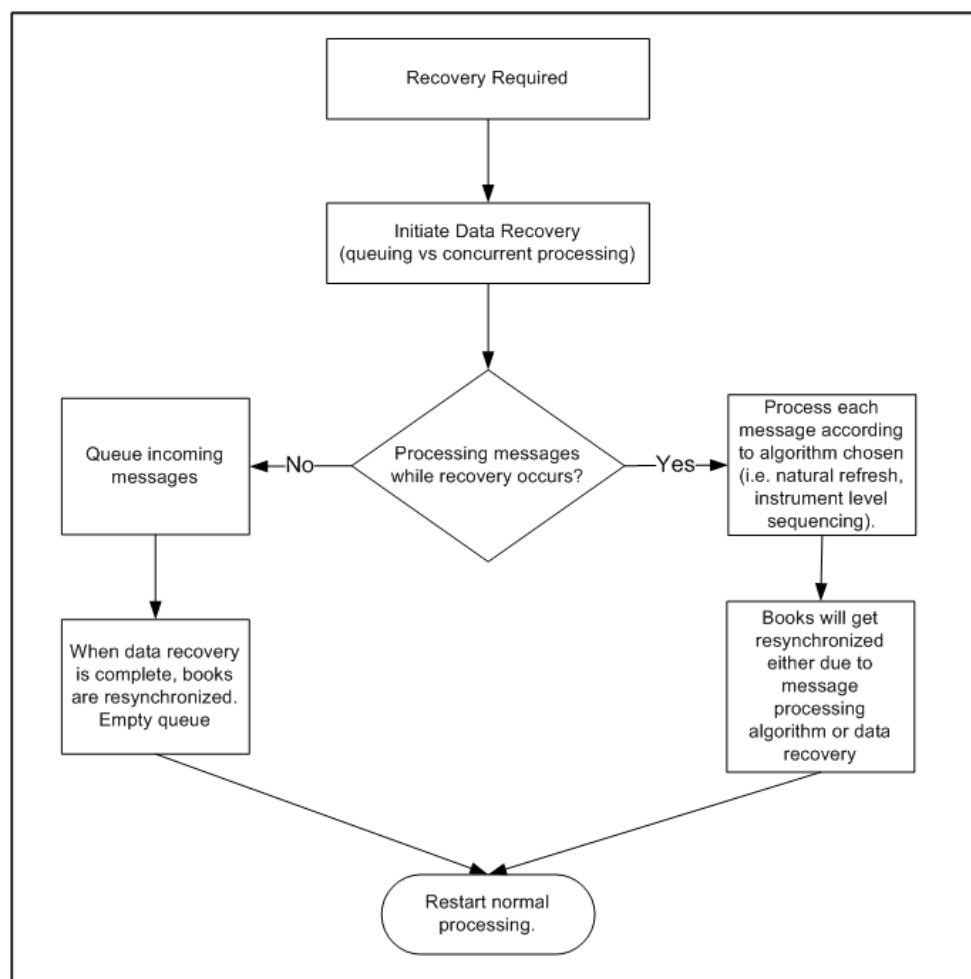
Top of Book - Best Bid/Ask

Bid		Ask	
Quantity	Price	Price	Quantity
		10	20
		20	10
		30	10
		40	10
		50	10

8.3 Recovering Data - Process

This section describes CME Group recommended recovery processes. Based on the recovery and supplemental methods, client systems can choose to implement those methods as applicable. In general, two paths can be followed: processing while recovering and queuing while recovering.

Note: The recovering data process applies to affected channels only. Unaffected channels can continue normal processing.



The following sections provide a high level process to recover in various situations:

- “Large Scale Outage Using Market Recovery - Queuing” on Page 93
- “Large Scale Outage Using Market Recovery - Concurrent Processing” on Page 93
- “Small Scale Data Recovery Using TCP Replay - Queuing” on Page 94
- “Small Scale Data Recovery Using TCP Replay - Concurrent Processing” on Page 95

Note: TCP replay should be used for small-scale data recovery. The Market Recovery feed should be used for large-scale data recovery (i.e. major outage or late joiners).

8.3.1 Large Scale Outage Using Market Recovery - Queuing

This section describes at a high level the process to follow for a large scale outage in which the client system is out of sync. This process uses Market Recovery - queuing of the Incremental Market Data feed and Market Recovery feed until the client system is synchronized to the latest state advertised by CME Group. In order to avoid an excessive number of queued messages, process snapshots and apply the applicable incremental feed as the snapshots arrive.

Note: If the Market Recovery method is used, client systems also need to subscribe to the Instrument Definition feed to determine whether any new instruments were defined. Client systems will not recover any missed statistics on the Market Recovery feed.

1. Identify channel(s) in which the client system is out of sync.
2. Listen to and queue the Incremental Market Data feed for the affected channel(s).
3. Listen to the Market Recovery feed for the affected channel(s).
4. Verify that all snapshots have been received for a given Market Recovery feed.
5. Apply **all** incremental data in sequence, **where tag 34-MsgSeqNum is greater than the lowest value for tag 369-LastMsgSeqNumProcessed**.
6. Using queued data, restart normal processing.

8.3.2 Large Scale Outage Using Market Recovery - Concurrent Processing

This section describes at a high level the process to follow for a large scale outage using Market Recovery while continuing to process the Incremental Market Data feed and obtaining snapshots from the Market Recovery feed. Once books are recovered, they can resume normal processing even if other books are still being recovered.

Note: If the Market Recovery method is used, client systems also need to subscribe to the Instrument Definition feed to determine whether any new instruments were defined. Client systems will not recover any missed statistics on the Market Recovery feed.

1. Identify channel(s) in which the client system is out of sync.
2. Listen to the Incremental Market Data feed for the affected channel(s) and optionally attempt a natural refresh.
3. Listen to the Market Recovery feed for the affected channel(s).
4. For each book, compare tag 369-LastMsgSeqNumProcessed on the Market Recovery feed to tag 34-MsgSeqNum on the Incremental Market Data feed and verify that the value for tag34-MsgSeqNum is not lower.
5. Restart normal processing.

8.3.3 Small Scale Data Recovery Using TCP Replay - Queuing

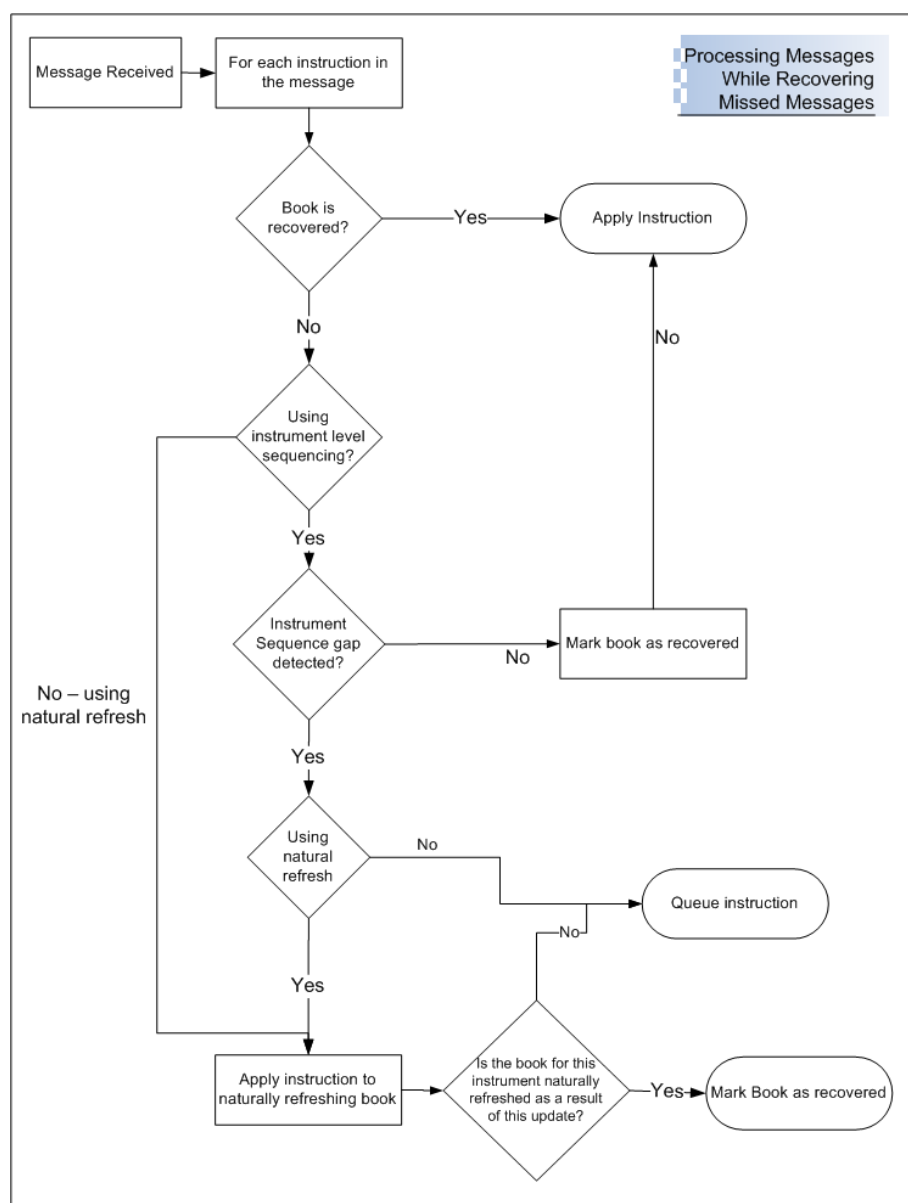
This section describes at a high level the process to follow for small scale data recovery using TCP Replay and queuing the Incremental Market Data feed until all missed messages are recovered. Client systems can recover specific messages that were missed using the sequence number and the TCP historical replay component.

1. Identify which messages have been missed (tag 34-MsgSeqNum).
2. Stop normal processing and initiate data recovery for affected channel(s).
3. Listen to and queue the Incremental Market Data feed for the affected channel(s).
4. Login to the TCP Replay component and request a replay of missed messages.
5. Apply resent messages in proper sequence.
6. Apply queued messages once replayed messages have been processed.
7. Stop queuing messages and restart normal processing.

8.3.4 Small Scale Data Recovery Using TCP Replay - Concurrent Processing

This section describes at a high level the process to follow for small scale data recovery using TCP Replay while continuing to process the Incremental Market Data feed. Client systems can recover specific messages that were missed using the sequence number and the TCP historical replay component.

1. Identify which messages have been missed (tag 34-MsgSeqNum).
2. Listen to and process the Incremental Market Data feed.
3. Login to the TCP Replay component and request a replay of missed messages.
4. If using Natural Refresh or Instrument Level Sequencing on the Incremental Market Data feed, refer to the following flowchart and apply to process.



5. For queued messages (see diagram):
 - Apply resent messages in proper sequence.
 - Apply queued messages once replayed messages have been processed.
6. Process missed messages from the TCP Replay component.
7. Books will get resynchronized either due to message processing algorithm or data recovery.
8. Restart normal processing.

Revision History

Initial Release	Version	Last Update	Author	Description
9/14/07	1.0	N/A	LM	Initial Release
9/14/07	1.1	10/31/07	LM	Clarified FTP Production Environment process. Updated "TCP Replay Overview" on Page 81. Updated "Market Recovery (UDP)" on Page 3 and "Instrument Definition (UDP)" on Page 3.
10/31/07	1.2	2/1/08	CR/DJ/LM	Rebrand CME Group. Added note regarding market recovery feeds to "Market Recovery (UDP)" on Page 3 and "Market Recovery Overview" on Page 85. Clarified "Services - Template Dissemination and Market Data Configuration" on Page 3. Added clarification to book depths to "Incremental Book Management" on Page 31. Added table to "Optional vs. Mandatory Fields" on Page 11. Updated "Real Time Statistics (Market Behavior Events)" on Page 57 to include more information on the use of tag 277-TradeCondition. Added applicable FIX/FAST 1.2 updates except recovery enhancements.

Initial Release	Version	Last Update	Author	Description
10/31/07	1.3	8/20/08	DT	<p>Updated "Figure 1. System Architecture" on Page 2</p> <p>Updated "Market Recovery (UDP)" on Page 3 and "Instrument Definition (UDP)" on Page 3 with Feed A and B information.</p> <p>Added "System Startup" on Page 5.</p> <p>Corrected (tag 25-MsgType=d) to (tag 35-MsgType=d) in "Book Management Mechanics - Multiple-Depth Book" on Page 33, "Book Management Mechanics - Implied Book" on Page 40, and "Book Management Mechanics - Top-of-Book" on Page 47.</p> <p>Added introduction of tags 5797-Aggressor-Side and 5799-MatchEventIndicator and added tags to examples throughout "Trade" on Page 60.</p> <p>Removed tag 271-MDEntrySize from table in "Last Best Price" on Page 59.</p> <p>Removed sentence about recovering missed statistics from "Recovery Feeds" on Page 81.</p> <p>Added note about username and password to "Process" on Page 83.</p> <p>Replaced ftp.cme.com with ftp.cmegroup.com throughout document.</p> <p>Updated environment names throughout document.</p>
10/31/07	1.4	9/3/2008	DT	<p>Removed note about accessing production environment details from "FTP Site Information - Template Dissemination and Market Data Configuration" on Page 4.</p> <p>Added Production environment user name and password to table under "FTP Site Information - Template Dissemination and Market Data Configuration" on Page 4.</p>

Initial Release	Version	Last Update	Author	Description
10/31/07	1.5	9/16/2008	DT	Added note about separate configuration file for FXMarketSpace to “Services - Template Dissemination and Market Data Configuration” on Page 3
10/31/07	1.6	10/10/2008	DT	Replaced mention of 35=Y with 35=5.