Cboe Application Programming Interface

Version 1.3.3

CBOE Streaming Market Index (CSMI) – Index

API for CBOE Streaming Market Data Feed for Indices

Cboe PROPRIETARY INFORMATION

February 06, 2018

Front Matter

Disclaimer

This document is made available to Cboe members and member firms to enable them to develop software applications using the Cboe Streaming Current Market application programming interface (API) and is subject to the terms and conditions of a Software License Agreement that governs its use. This document is provided "AS IS" with all faults and without warranty of any kind, either express or implied.

Change Notices

The following change notices are provided to assist users of the Cboe Streaming Current Market Index features in determining the impact of changes to their applications.

Date	Version	Description of Change
02/06/2018	1.3.3	Updated date for parallel testing over 5 channels.
10/17/2017	1.3.2	Cboe branding/logo changes.
9/1/2017	1.3.1	Updated primary and secondary multicast group IPs for new CSMI platform.
8/30/2017	1.3	Updated Appendix A – Multicast Group and Port Information. Include FTSE, Cryptocurrency, and INAV.
6/7/2016	1.2	Updated the Index Options market hours start time from 5:00 a.m. to 2:00 a.m.
12/10/2015	1.2	Updated MD-EntryType table reference.
12/7/2015	1.1	Updated the cover page to reference indices
9/17/2015	1.0	New document

Support and Questions Regarding This Document

Questions regarding this document can be directed to the Cboe Operations Support Center ("OSC) at 312.786.7635 or via e-mail: indexsupport@cboe.com. The latest version of this document can be found at the Index microsite at http://batsintegration.cboe.com/index.

Table of Contents

Reference Tables ... 4

1 Introduction ... 6

1.1 System Overview ... 6

1.2 Hours of Operation ... 6

2 Data Feed and Message Overview ... 7

2.1 Data Feed Overview ... 7

2.2 Message Overview ... 7

3 Message Templates, Field Data Types and Data Encoding ... 8

3.1 Message Templates ... 8

3.2 Template IDs ... 9

3.3 Field Data Types and Data Encoding ... 9

4 Packet and Message Header Format ... 12

4.1 Packet Header ... 12

4.2 Message Header ... 13

5 Messages ... 15

5.1 Index Value – Template ID 22 ... 15

5.2 Heartbeat Message (Line Integrity Message) – Template ID 16 ... 16

6 Appendix A – Multicast Group and Port Information ... 17

7 Appendix B – Examples ... 20

7.1 Understanding the Hex Data Diagrams ... 20

7.2 Packet Header Example ... 20

7.3 Index Example ... 20

7.4 Heartbeat Message ... 22

Reference Tables

1 - Example of an XML based template 8
2 - Templates and their IDs 9
3 - Packet Format 12
4 - Packet Header 12
5 - Message Format 13
6 - Message Header 13
7 –Message Types 13
8 – Index Value Message Structure 15
9 – Index Value Template 16
10 – MD Entry Type 16
11 - Heartbeat Message Structure 16
12 - Heartbeat Template 16

Examples of Data Transmissions

1 - Packet Header Hex Dump 20
2 - Packet Header Decoded 20
3 - Heartbeat Hex Dump 22
4 - Heartbeat Decoded 22

1 Introduction

The Cboe Streaming Market Index feed publishes index data using the message format defined in this document. Data is transmitted using the IP Multicast network protocol. To connect to the CFN network, refer to the Cboe Futures Exchange Connectivity Manual.

1.1 System Overview

Cboe Streaming Market Index distributes index values. A **feed** is a set of one or more data channels. A **channel** consists of two Multicast groups in a primary/secondary architecture where the data is duplicated on the two Multicast groups for redundancy.

Communication is one way only with no mechanism for retransmission. Messages are encoded using a mix of ASCII characters and binary data in the format defined in this document. Cboe also distributes templates that describe the message structure of Cboe Streaming Market feeds via the API website at https://systems.cboe.com/Auth/CFN.aspx. These templates may be used for decoding messages. The templates and message structures defined in this document will be static, and will not change over the course of the trading day, nor even in most software releases. Clients can expect sufficient advance notice about any changes to these templates or message structures.

1.2 Hours of Operation

Messages will be published during these hours.

Normal market hours are as follows:

Open (CT)	Close (CT)
2:00 AM	5:30 PM

2 Data Feed and Message Overview

2.1 Data Feed Overview

The feed consists of one **data** channel. General characteristics of the feed include the following:

- The channel is duplicated and sent to 2 different multicast groups and ports over 2 networks in a primary / secondary configuration. Data sent to the primary and secondary multicast groups for each channel is identical.

- There are no retransmissions. Index values are computed and transmitted at regular intervals.

- A sequence number is sent for each message. This can be used to identify missed messages.

- Messages are placed into blocks (packets) for delivery which allows for multiple messages per block. The maximum block size is 1000 bytes.

- The message structures, field names and field values are based as much as possible on the FIX 5.0 SP2 standard. However, messages are encoded using a proprietary ASCII + binary format, and FIX tags are not transmitted in the data stream. The FIX format was used for the convenience of those familiar with the FIX standard, so messages are defined in terms of FIX field names and FIX tags. Some user-defined fields were necessary for those fields not in the FIX specification, and some modifications to standard FIX fields are implemented for efficiency reasons.

2.2 Message Overview

The following types of messages are transmitted over the feed:

2.2.1 Index Value

Index value messages contain the values associated with a calculated index. For some indices, a bid and ask value may also be calculated which is like the index value, but is calculated from bid and ask prices instead of last sale prices. Index values are benchmark values upon which tradable products may be based, but an index itself is not tradable.

Index values are not sent using classKey and security IDs like other CSM feeds. Index values do not have a security definition associated with them, instead the index symbol is sent in every index value message.

2.2.2 Heartbeats

Heartbeat / line integrity messages are transmitted every five seconds. These messages may be used to determine that a channel is working during times when market data is not transmitted on the feed (such as pre-market or post-market times).

3 Message Templates, Field Data Types and Data Encoding

3.1 Message Templates

Messages for the Cboe Streaming Market Index feed are described in this document in tabular text format and as an XML template. Templates define the content and characteristics of the messages to be encoded or decoded.

XML templates that describe the structure of messages are available to recipients on the Cboe web site at https://systems.cboe.com/Auth/CFN.aspx. Firms are encouraged to write software capable of using the XML templates to decode data from a Cboe Streaming Market feed.

XML templates are used to specify the structure, data types, field names, and FIX tags of a message:

1 - Example of an XML based template

`<template name="MDIncRefresh" id = “0”>`	Start of new template
`  <string     name="MessageType"     id="35" byteLength="1" value="X" />`	Defines a String Data Type Field, The id which represents the fix Tag is not transferred on the wire.
`  <uInt32 name="MsgSeqNum" id=”34” />`	
`  <sequence name="MDEntries">`	Defines the start of a repeating group
`    <length name="NoMDEntries” id =”268"/>`	Length of repeating group
`    <string name="Symbol" id=”55”/>`	
`    <uInt32 name="Quantity” id=”53 />`	
`  </sequence>`	End repeating group
`</template>`	End template

3.2 Template IDs

Each message structure is defined with a unique template ID. A template ID is a binary integer value stored as the first byte of every message that identifies the structure of the message.

Template IDs are assigned from a common pool for the Cboe Streaming Market and the CSM Level 2 Book Depth feed, so there may be gaps in the numbering when either specification is updated. Numbers are assigned sequentially as they are needed for new message structures and old template ids are retired when new versions of the feed are launched.

The template ids for Cboe Streaming Market Index feeds are as follows:

2 - Templates and their IDs

Template Name	Template ID	Assigned in Version
Heartbeat	16	1.0
Index Value	22	1.0

3.3 Field Data Types and Data Encoding

Fields defined in messages for Cboe Streaming Market Index feed will have one of the following data types and methods of encoding:

3.3.1 STRING Field

Strings are ASCII character arrays or single-byte characters. There are two types of string fields which are encoded differently:

Single Byte String

If the *byteLength* attribute of a string field is defined as "1", for example:

```
<string name="MessageType" id="35" byteLength="1"/>
```

it is a single byte string, which is encoded with a single ASCII character.

Character Array String

If the *byteLength* attribute is not defined for the field, for example:

```
<string name="Symbol" id="55"/>
```

The string is variable length and is encoded with an unsigned binary byte indicating the length of string, followed by the string's characters:

Length of String (1 byte)	String Characters

For example, the string "IBM" would be encoded as:
Binary value 3, then the characters "IBM".

3.3.2 INTEGER and LENGTH Fields

Integer and Length fields are big endian binary encodings of numeric values. The *byteLength* attribute in a template field definition can act as a modifier to restrict the number of bytes used to encode the integer value. Unsigned integers are encoded as zero or positive-only values. The top-most bit is part of the magnitude of the value. Signed integers are encoded as two's-complement binary values with the top-most bit as the sign bit. Length fields are unsigned integer values used to indicate the length of a Sequence field.

There are several types of integer or length fields:

Integer Field Type	Byte Length	Encoding	Example
uInt32	1	8 bit unsigned integer	<uInt32 name="MDPriceLevel" id="1023" byteLength="1"/>
uInt32	4 or omitted	32 bit unsigned integer	<uInt32 name="MDEntrySize" id="271" byteLength="4"/>
uInt64	8 or omitted	64 bit unsigned integer	<uInt64 name="SendingTime" id="52"/>
int32	1	8 bit signed integer	<int32 name="MDPriceLevel" id="1023" byteLength="1"/>
int32	4 or omitted	32 bit signed integer	<int32 name="MDEntrySize" id="271"/>
int64	8 or omitted	64 bit signed integer	<int64 name="SendingTime" id="52" byteLength="8">
length	1	8 bit unsigned integer	<length name="NoMDEntries" id="268" byteLength="1"/>

The table below shows the min and max values for different integer data types.

Type	Min	Max
uInt32 with byteLength="1"	0	255
uInt32	0	4,294,967,295
uInt64	0	18,446,744,073,709,551,615
int32 with byteLength="1"	-128	127
int32	-2,147,483,648	2,147,483,647
int64	- 9,223,372,036,854,775,808	9,223,372,036,854,775,807
length with byteLength="1"	0	255

3.3.3 DECIMAL Field

A decimal field is used to represent a floating point number as exponent and mantissa. The exponent is a signed 8 bit integer used to express precision and the mantissa is a signed 32 bit 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: number = mantissa * 10^{exp}. The exponent and mantissa is decoded as a single, composite field.

Decimal fields are 5 bytes in length. The first byte is the exponent and the remaining 4 bytes are the mantissa. For example, the number 0.90 is encoded as FE0000005A.

FE (exponent) == -2, 0000005A (mantissa) == 90, value == 90 * 10^{-2} == 90 * 0.01 == 0.90

3.3.4 SEQUENCE Field

A sequence is a repeating group of fields. A length field encoded as an unsigned int immediately precedes the fields contained in the sequence. The length field is defined in a template with a special attribute of "<length", and it can be modified with *byteLength* attribute. If byteLength="1", the encoded length field is a single 8-bit unsigned byte. All sequences transmitted to Cboe Streaming Market feeds use a single 8-bit unsigned length and can be no longer than 255 entries.

Sequences are encoded as follows:

Length field	Group#1 Field #1	Group#1 Field#2	…	Group#1 Field#N	Group#2 Field#1	Group#2 Field#2	…	Group#2 Field#N	…

Here is an example sequence with 2 MDEntries elements.

Field	Length in Bytes	Value	Comments
NoMDEntries	1	2	Length of MDEntries Sequence
MDEntryType	1	'0'	Bid entry
MDEntryPx	5	FE0000000A	FE == -2 exponent, 0000000A == 10 mantissa, value == 0.10
MDEntryType	1	'1'	Ask entry
MDEntryPx	4	FE00000010	FE == -2 exponent, 00000010 == 16 mantissa, value == 0.16

4 Packet and Message Header Format

All messages are sent in Multicast packets. Each packet consists of a packet header and one or more messages.

3 - Packet Format

Packet (a.k.a. Block)					
Packet Header					Contents
Version	**Length**	**Sending Time**	**Number of messages**	**First Msg Seq #**	**Messages...**

4.1 Packet Header

Each packet has a packet header that appears once at the beginning of the packet. The packet header has the following structure:

4 - Packet Header

Field Name	Type	Length (Bytes)	Comments
Version	uInt32	1	The version associated with the contents and format of this header. Currently, this will be a constant value of 1.
Length	uInt32	2	Length of the packet including this length field and the version. Note that this is a 2 byte length.
Sending Time	uInt64	8	The time that this packet was sent. It applies to all messages in this packet.
Number of messages	uInt32	1	The number of messages in this packet.
First Msg Seq #	uInt32	4	The sequence number on the first message in this packet.

The version of a packet indicates the format of the packet. This may be incremented in future releases to indicate a change in the format of the packet. Initially, it is set to the number 1.

The Packet Length is encoded as a 2 byte (16 bit) unsigned integer that includes the length of the version, the 2 byte Packet Length itself, and the remainder of the packet.

The Sending Time is the time that the Cboe Streaming Market Index application published the packet on the feed. The sending time is the millisecond timestamp from midnight, January 1, 1970 UTC.

The "First Msg Seq #" is the sequence number of the first message of this packet, and the "Number of Messages" indicates the total number of messages contained in the packet.

For verification of data at the channel level, one could compute the expected "first msg seq #" of the next packet by adding the number of messages to the current packet's "first msg seq #".

4.2 Message Header

A packet contains multiple messages. Each message is preceded by a message header common to all messages.

5 - Message Format

Message						
Message Header				Contents		
		Template defined fields				
Length	**Template ID**	**Msg Type**	**Msg Seq #**	**Field #1**	**…**	**Field #N**

6 - Message Header

Field ID	Field Name	Type	Length (Bytes)	Comments
	Message Length	uInt32	2	The length of this message including the 2 bytes for this length field.
	Template ID	uInt32	1	The Template ID is for decoding the message. See table: *2 - Templates and their IDs*
35	MessageType	String	1	See table: *7 –Message Types*
34	MsgSeqNum	uInt32	4	Sequence Number

The Message Length is encoded as a 2 byte (16 bit) unsigned integer that includes itself and the remainder of the message including all Message Header fields.

The Template ID defines the specific Structure of the message.

The Message Type defines the market data message type compliant to the FIX standard.

The message sequence number is a consecutively increasing number from the previous message. The first message in a packet will start with 1 number greater than the last message in the previous packet

7 –Message Types

X	Index Value
0	Heartbeat

4.2.1 Message Sequence Numbers

Every packet has a "first" sequence number in the header. This is the number associated with the first message in the packet. Each subsequent message in the packet has a sequence number that is one greater than the previous message. The next packet will have a starting sequence number that is one more than the last message in the previous packet except for the start of the trading day's session and in the event of a Cboe system failure. When the session is started the sequence number will be reset to 1. During a Cboe system failure the sequence number can reset to a lower value than was previously seen prior to the failure. In either case, the sequence number will again increase by one for each message.

Each channel of a feed has its own sequence number associated with it starting with sequence number 1. Verification of message sequence numbering must be done for each individual channel.

Firms must ensure that the sequence numbers maintain continuity. Any deviation from an expected sequence number must be considered as an error condition. Firms are required to take appropriate recovery action any time that an unexpected sequence number is detected.

4.2.2 Recovery from Unexpected Message Sequence Numbers

Each message sent on a channel causes its *MsgSeqNumber* to increment by one. To detect missing data at the channel-level, compare each incoming *MsgSeqNumber* with the last received *MsgSeqNumber* + 1. If the incoming *MsgSeqNumber* is not equal to the (last received *MsgSeqNumber* + 1), data is missing from the channel.

Regardless of whether missing data is detected or not, the *MsgSeqNumber* of the incoming message should be stored associated with the channel so subsequent missing data can be detected.

When missing data is detected at the channel-level, all market data for products that were received over that channel should be treated as "suspect", meaning their market data may be incorrect. At the time missing data is detected on a channel, there is no way to know which index's data is missing, therefore, all indexes received from that channel must be treated as though the market data for those indexes may be incorrect.

Indexes marked as suspect or possibly-incorrect should remain in that state until an index value message for that index is received. Use it to update the suspect index's market data and mark the index's market data as no longer suspect.

5 Messages

5.1 Index Value – Template ID 22

Index value messages contain the values associated with a calculated index. They are transmitted when an index is calculated.

Market data associated with the index are contained in MDEntries with an MDEntryType indicating the kind of market data. The Index Value message may include any of the following MDEntryTypes (See table 10 – MD Entry Type).

- Index Value
- Bid
- Ask

At least one MDEntry with MDEntryType == Index Value will always be present in the Index Value message. For indices where a Bid and Ask index value is calculated, MDEntryTypes Bid and Ask will be present. For indices where Bid and Ask is not calculated, MDEntryTypes for Bid and Ask are omitted.

8 – Index Value Message Structure

Field ID	Field Name	Type	Length (Bytes)	Comments
	Standard Header			See table: *7 –Message Types,* MessageType = "X"
55	Symbol	string		Index symbol
	MDEntries	Sequence		

The `MDEntries` sequence field has the following sub fields (For detailed sequence field format information, refer to the Fields Data Types section in this document):

Field ID	Field Name	Type	Length (Bytes)	Comments
268	NoMDEntries	length	1	Number of MDEntries in this message. Will not exceed 255
The Following Fields Repeat NoMDEntries times				
269	MDEntryType	single byte string	1	Entry Type. See table: 10 – MD Entry Type
270	MDEntryPx	decimal	5	Index value associated with MDEntryType

9 – Index Value Template

```
<template   name="IndexValue"                id="22">

    <string     name="MessageType"              id="35"     byteLength="1"
                                                            value="X"  />

    <uInt32     name="MsgSeqNum"                id="34"     byteLength="4"/>

    <string     name="Symbol"                   id="55"     />

    <sequence   name="MDEntries">

        <length     name="NoMDEntries"          id="268"    byteLength="1"/>

        <string     name="MDEntryType"          id="269"    byteLength="1"/>

        <decimal    name="MDEntryPx"            id="270"    byteLength="5"/>

    </sequence>

</template>
```

10 – MD Entry Type

0	Bid
1	Ask
3	Index Value (Used only in Index Value messages)

5.2 Heartbeat Message (Line Integrity Message) – Template ID 16

This message contains only a standard header. The heartbeat will repeat at a regular interval.

11 - Heartbeat Message Structure

Field ID	Field Name	Type	Length (Bytes)	Comments
	Standard Header			See table: *7 –Message Types*, MessageType = “0” (zero)

12 - Heartbeat Template

```
<template   name="Heartbeat"                id="16">

  <string       name="MessageType"              id="35"         byteLength="1"
                                                                value="0"  />

  <uInt32       name="MsgSeqNum"                id="34"         byteLength="4"/>

</template>
```

6 Appendix A – Multicast Group and Port Information

Please work with the Cboe NOC to make the necessary network updates to be able access the CSMI platform for the certification, production, and DR environments. *The current NY4 CSMI platform will be deprecated and will be replaced by a new NY5 CSMI platform effective January 22, 2018. It will remain available as a secondary source of CSMI data through February 2, 2018 at which point it will be deprecated.*

Cboe Index Primary Groups (Current NY4 Platform)

Group	Description	Target Location ID	A or B	Port	RP	Source Networks
233.103.126.82/32	Cboe CSM Index Prod A Groups		A			
233.103.126.82	Cboe Market Data	0	A	64876	170.137.128.253	170.137.144.0/26

Cboe Index Backup Groups (Current NY4 Platform)

Group	Description	Target Location ID	A or B	Port	RP	Source Networks
233.103.126.210/32	Cboe CSM Index Prod B Groups		B			
233.103.126.210	Cboe Market Data	0	B	64878	170.137.128.254	170.137.144.64/26

Cboe Index Groups (Effective 01/22/18)

On the effective date, the Cboe Index Groups will be migrated to a the new CSMI platform requiring network configuration updates as follows. *CSMI data will be also available for parallel production testing using this configuration effective December 11, 2017.*

Data Channel	Location	Type	RP	Source IP	IP	Port
CSMI	NY5 Prod DC	Primary	74.115.128.168	174.136.169.32/29	224.0.131.168	30201
		Secondary	74.115.128.169	174.136.169.40/29	233.130.124.168	30201
	DR DC	Primary	170.137.16.130	170.137.16.192/29	233.182.199.8	31201
	NY5 Cert DC	Primary	74.115.128.166	174.136.160.32/29	224.0.74.176	32201
MSCI	NY5 Prod DC	Primary	74.115.128.168	174.136.169.32/29	224.0.131.169	30201
		Secondary	74.115.128.169	174.136.169.40/29	233.130.124.169	30201
	DR DC	Primary	170.137.16.130	170.137.16.192/29	233.182.199.9	31201
	NY5 Cert DC	Primary	74.115.128.166	174.136.160.32/29	224.0.74.177	32201

Cboe Index Groups (Effective 02/26/18)

On the effective date, the Cboe Index Groups will be split out from 2 to 5 multicast groups as follows.

Data Channel	Location	Type	RP	Source IP	IP	Port
CSMI	NY5 Prod DC	Primary	74.115.128.168	174.136.169.32/29	224.0.131.168	30201
		Secondary	74.115.128.169	174.136.169.40/29	233.130.124.168	30201
	DR DC	Primary	170.137.16.130	170.137.16.192/29	233.182.199.8	31201
	NY5 Cert DC	Primary	74.115.128.166	174.136.160.32/29	224.0.74.176	32201
MSCI	NY5 Prod DC	Primary	74.115.128.168	174.136.169.32/29	224.0.131.169	30201
		Secondary	74.115.128.169	174.136.169.40/29	233.130.124.169	30201
	DR DC	Primary	170.137.16.130	170.137.16.192/29	233.182.199.9	31201
	NY5 Cert DC	Primary	74.115.128.166	174.136.160.32/29	224.0.74.177	32201
FTSE	NY5 Prod DC	Primary	74.115.128.168	174.136.169.32/29	224.0.131.170	30201
		Secondary	74.115.128.169	174.136.169.40/29	233.130.124.170	30201
	DR DC	Primary	170.137.16.130	170.137.16.192/29	233.182.199.10	31201
	NY5 Cert DC	Primary	74.115.128.166	174.136.160.32/29	224.0.74.178	32201
CCCY	NY5 Prod DC	Primary	74.115.128.168	174.136.169.32/29	224.0.131.171	30201
		Secondary	74.115.128.169	174.136.169.40/29	233.130.124.171	30201
	DR DC	Primary	170.137.16.130	170.137.16.192/29	233.182.199.11	31201
	NY5 Cert DC	Primary	74.115.128.166	174.136.160.32/29	224.0.74.179	32201
INAV	NY5 Prod DC	Primary	74.115.128.168	174.136.169.32/29	224.0.131.172	30201
		Secondary	74.115.128.169	174.136.169.40/29	233.130.124.172	30201
	DR DC	Primary	170.137.16.130	170.137.16.192/29	233.182.199.12	31201
	NY5 Cert DC	Primary	74.115.128.166	174.136.160.32/29	224.0.74.180	32201

7 Appendix B – Examples

7.1 Understanding the Hex Data Diagrams

Through this section you will see hexadecimal printouts of data that is from a channel. Each of the examples will be in a similar format.

```
Offset          Hexadecimal  data  4  bytes  per  group       ASCII  representation
00000000  CD580000 00030000 0134C8E5 A8992B6C  |.X.......4....+l|

00000010  031FFE00 00006600 00006401 00        |......f...d..   |
```

The first column of numbers is the zero-based byte offset of the first hex byte on that line, expressed as a hexadecimal offset. The next 4 columns of data are hexadecimal values for the bytes, with 4 bytes per group. Spaces are for formatting purposes only. Characters between pipes (“|”) are ascii representations of the hex data. Periods indicate non-printable values.

The first 16 bytes of each hex dump is the packet header.

7.2 Packet Header Example

1 - Packet Header Hex Dump

```
00000000  01038B00 000135A6 B387070B 00000000  |......5.........|
```

2 - Packet Header Decoded

```
SendTime(02/22 14:14:37.831)

BlkVersion(1) BlkSize(907) MsgsInBlock(11) FirstMsgSeq(0)
```

7.3 Index Example

3 – Index Hex Dump

```
00000000  01002F00 00015023 AC595A01 00000002  |../...P#.YZ.....|
00000010  001F1658 00000002 034F4558 0333FE00  |...X.....OEX.3..|
00000020  014B6030 FE00014B 4E31FE00 014B71    |.K`0...KN1...Kq |
```

4 - Index Decoded

```
Message#(1) MsgSize(31)
TemplateId(22) TemplateName(IndexValue)   AppMsg: IndexValue Type: X
  FieldType        FieldName                        Id      Value
  sbSTRING:        MessageType                      35      X
  UINT32:          MsgSeqNum                        34      2
  STRING:          Symbol                           55      OEX
 SEQUENCE:        NoMDEntries                      268     Len=3
    MSG_GROUP NumElems: 2
      sbSTRING:        MDEntryType                      269     3
      uDECIMAL:        MDEntryPx                        270     848.32
    MSG_GROUP NumElems: 2
      sbSTRING:        MDEntryType                      269     0
      uDECIMAL:        MDEntryPx                   270    848.14
    MSG_GROUP NumElems: 2
      sbSTRING:        MDEntryType                 269    1
      uDECIMAL:        MDEntryPx                   270    848.49
```

7.4 Heartbeat Message

3 - Heartbeat Hex Dump

```
00000000  01001800 000135A7 00C6C901 00000F95  |......5.........|
00000010  00081030 00000F95                    |...0....        |
```

4 - Heartbeat Decoded

```
Message#(1) MsgSize(8)
TemplateId(16)   AppMsg: Heartbeat Type: 0
  FieldType        FieldName                        Id     Value
  sbSTRING:        MessageType                      35     0
  UINT32:          MsgSeqNum                        34     3989
```