ARCABOOK COMMON CLIENT SPECIFICATION

Version
3.3

Date
May 17, 2025

DOCUMENT HISTORY

The following table provides a description of recent changes to this document.

Version	Date	Change Description
2.0	01/08/2015	Re-branded for ICE Structure and explanatory text revised for clarity and simplicity NYSE Security Status msg's Transaction ID field changed to Reserved Source Time Reference msg's SymbolIndex field changed to System ID IBF 2.0
2.0a	02/16/2015	Noted shorter length of Symbol Index Mapping (msg type 3) and Security Status (msg type 34) for Arca Integrated Feed only Minor tweaks to language
2.0b	02/18/2015	Corrections in section 3.6.2 regarding Order ID's Corrected Packet Header DeliveryFlags for Seq Num Reset messages (4.3 and 9.2)
2.0c	03/10/2015	Corrected Security Status msg length to 46 Added detail about multicast priming in Section 9 Added detail about refresh formats in 5.1.3 Added detail about IP Table filtering in 8.4
2.0d	04/07/2015	Clarified descriptions and availability of last 8 fields in the Security Status msg
2.0e	05/12/2015	Modified Security Status msg, Security Status field to reflect Arca adoption of NYSE SSR values Clarified the description of Time field in Security status message and System ID field in Order Acknowledgment message
2.0f	06/30/2015	Updated the Exchange Code field values in the Symbol Index Mapping Message to include Global OTC primary symbols Updated SSR Triggering Exchange ID field values in Security Status Message Updated Listed Market values in the Symbol Index Mapping File format to include the Listed Market for Global OTC primary symbols
2.0g	07/10/2015	Updated legal disclaimer for Global OTC on title page
2.0h	10/13/2015	Updated the location of Symbol Index Mapping File
2.0i	Mar 18, 2016	Corrected fields in the symbol index ftp file
2.0j	Mar 28, 2016	Clarified/corrected production hours in section 9.4
2.0k	June 16, 2016	Restored guidance on handling the Session Change msg for users of Arca Integrated and ArcaBook feeds. This was incorrectly removed from version 2.0.
2.0L	Aug 18, 2016	Clarified trailing pipe characters in Symbol Index Mapping File (section 10)

Version	Date	Change Description
2.0m	Oct 4, 2016	Updated Security Status message to include V as the exchange ID for IEX. Updated section 2.2 to reflect change of heartbeat frequency from 60 secs to 1 sec.
2.0n	Aug 28, 2020	Regulatory initiative - Updated CTS Halt Condition Codes on msgtype 34 - effective Q4 2020.
2.0o	Jan 21, 2022	Msgtype 34 - Added Publication of security status = B (Begin accepting orders via the Pillar Gateways) Section 3.5 - Update 'Price' field in all message types to signed binary integer.
2.0p	Mar 4, 2022	Section 9.6 - Updated symbol mapping file reference to Pillar Specifications.
2.0q	Mar 21, 2022	Updated NYSE branding and document formats only - no content change.
3.0	May 15, 2023	Added Section 3.5.1 Maximum Price Added Section 5.1.3.5 Differences between Legacy RCF and new Pillar Request Server Added Section 8.1 Pillar Request Server Sections 1, 3.7, 4.4, 5.1.2, 5.1.3 - Added clarifications Updated links for REFERENCE MATERIAL, CONTACT INFORMATION, and FURTHER INFORMATION sections
3.1	Feb 22, 2024	Updated definitions for 'SourceID' fields. Added 6 (suspend) as a valid value in security status and halt condition fields of MsgType 34 ~~Updated Section 9.4 ArcaBook Production Hours to account for new feed start time of 2:00am EST~~ Clarified in section 8.1.3 Request Quotas that quotas are maintained at a Client ID level, not a channel level
3.2	Nov 15, 2024	Updated Section 9.4 ArcaBook Production Hours to account for new feed start time of 2:00am ET Updated support contact information
3.3	May 17, 2025	Updated Symbol Index Mapping Message (Msg Type 3) to support "Exchange Code" of "M", representing NYSE Texas

REFERENCE MATERIAL

The following lists the associated documents, which either should be read in conjunction with this document or which provide other relevant information for the user:

- ICE Global Network
- NYSE Symbology Specification
- IP Addresses

CONTACT INFORMATION

- Support: support@nyse.com | +1 212 896-2830
- Data Sales: datasales@nyse.com

FURTHER INFORMATION

- For additional information, please visit the NYSE Real-Time Market Data pages
- For updated capacity information, visit our capacity pages

NYSE
AN ICE EXCHANGE

TABLE OF CONTENTS

1. INTRODUCTION

1.1 RECEIVING REAL TIME MARKET DATA

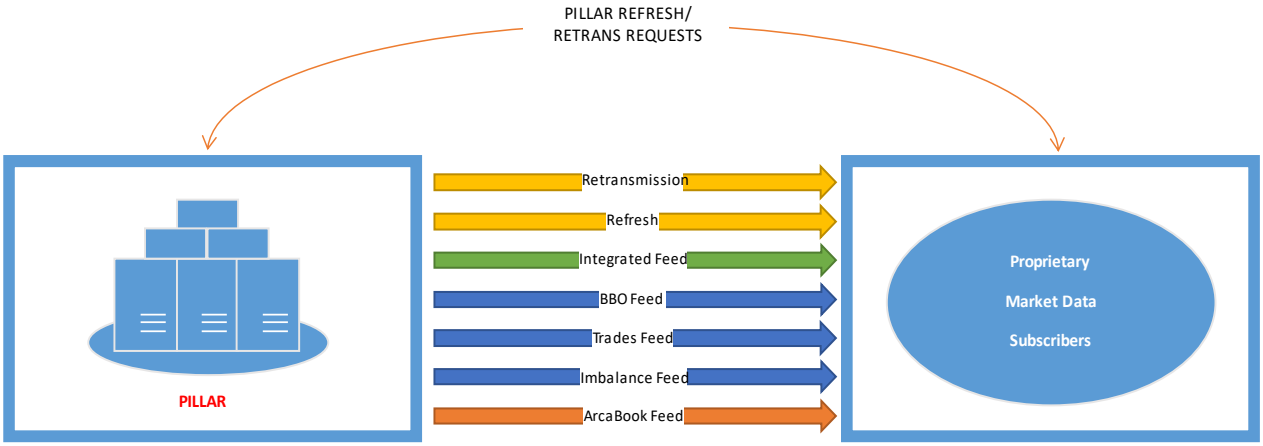
Pillar trading engine publishes real-time proprietary data in the form of messages with fixed length fields. All fields are binary except a very small number that are in ASCII format. For efficient use of the network, the messages are bundled into application packets, and the packets are published via the multicast protocol.

For capacity reasons, packets are routed over a number of predefined data sets called channels. Each channel is duplicated and published to two distinct multicast groups for redundancy. The two redundant multicast groups per channel (called lines) are referred to as line A and line B. The union of the data in all channels that make up a product is called a feed.

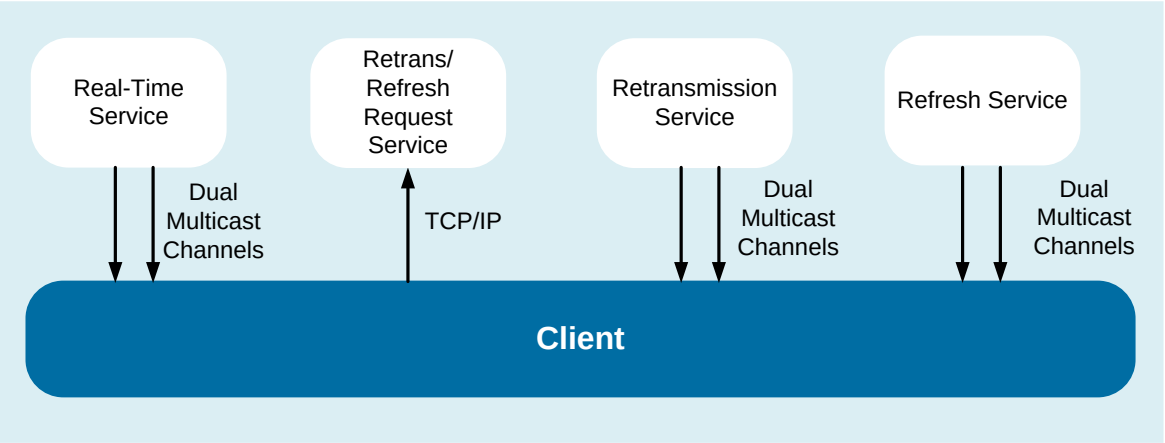
The IP addresses and port numbers of the production and test channels for each proprietary data feed can be found at https://www.nyse.com/publicdocs/nyse/data/IP_Addresses.xlsx.

A client application receives a product by subscribing to some or all of the channels that make up the feed.

In response to requests for retransmission and refresh, market data is published by the exchange over dedicated multicast channels which correspond one-to-one with the real-time channels.



1.2 RECOVERING FROM ERRORS



- In case of dropped multicast packets, the client can connect to a Request Server via TCP/IP to request retransmissions of missed messages.
- In case of client late start or intraday failure, the client can connect to the Request Server and request snapshot refreshes of the state of the market.
- At system startup, each channel publishes referential data about all symbols published on the channel. If a client process misses this initial spin of symbol data, he can connect to the Request Server and request a refresh of some or all of the missed data.

In response to these requests, retransmission and refresh data is published by the exchange over dedicated multicast channels which correspond one-to-one with the real-time channels.

See Error Handling and the Request Server for complete information.

2. Packets and Heartbeats

2.1 PACKET HEADER

All packets sent on any proprietary datafeed have a Packet Header followed by one or more messages (with the exception of Heartbeat packets which do not contain any messages).

The maximum length of a packet is 1400 bytes, so no message can be longer than 1400 – 16 bytes (max packet size - the length of the Packet Header).

Packet Header Structure

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
PktSize	0	2	Binary	The size of the packet in bytes, including this 16 -byte packet header
DeliveryFlag	2	1	Binary	A flag that indicates whether this is an original, retransmitted, or 'replayed' message. Valid values include: • 1 – Heartbeat • 10 – XDP Failover (see XDP Publisher Failover) • 11 – Original Message • 12 – Sequence Number Reset Message • 13 – Only one packet in retransmission sequence • 15 – Part of a retransmission sequence • 17 – Only one packet in Refresh sequence • 18 – Start of Refresh sequence • 19 – Part of a Refresh sequence • 20 – End of Refresh sequence • 21 – Message Unavailable
NumberMsgs	3	1	Binary	The number of messages in this packet
SeqNum	4	4	Binary	The message sequence number of the first message in this packet
SendTime	8	4	Binary	The time when this packet was published to the multicast channel, in seconds since Jan 1, 1970 00:00:00 UTC.
SendTimeNS	12	4	Binary	The nanosecond offset from the Send Time

2.2 HEARTBEATS

To assist the client in confirming connection health, application heartbeats are sent once a minute by the Request Server, and once a second by the real-time publishing servers (data, refresh and retransmissions channels).

A heartbeat consists of a packet containing a Packet Header and no messages. The Packet Header's Delivery Flag is set to 1 and Number Msgs is 0. Since a heartbeat packet contains no messages, a heartbeat does not increment the next expected sequence number. See Sequence Numbers.

Heartbeats sent by the Request Server must be acknowledged by the client. See Request Server.

3. Message Field Content

Messages are contiguous data structures consisting of fixed-length fields. No names or 'tags' appear in the message.

- Message fields align on 1 byte boundaries, so there are no filler fields for alignment purposes
- Binary fields are published in Little-Endian ordering
- All ASCII string fields are left aligned and null padded
- Segmentation of messages across packets is not supported, so a message will never straddle a packet boundary.
- The length of a message as actually published may differ from the length of the message structure defined in the client specifications. See Msg Size Field below for details.

3.1 MESSAGE HEADER

The format of each message varies according to type, but each type starts with a standard 4-byte message header:

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	The size of this message in bytes
MsgType	2	2	Binary	The type of this message

3.1.1 Msg Size Field

In order to handle future releases of proprietary data feeds smoothly, clients should never hard code msg sizes in feed handlers. Instead, the feed handler should use the Msg Size field to determine where the next message in a packet begins.

This allows

- Support of data feed format variations among markets
- Client flexibility when revised message structures go live in production

In example 1 below, a message type is defined in the specification to have different lengths in different markets. The trailing field is not published in the Arca market. An Arca-coded client can process NYSE data correctly (but of course cannot use the trailing Volume field without field-specific coding).

Example 1: Message type with format variations across markets

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary	Size of the message. **NYSE – 24 bytes** **NYSE American – 24 bytes** **NYSE Arca - 20 bytes**

Look at the Msg Size field to know where the next message starts.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Type	2	2	Binary	The type of this message: 998 – Example 1 msg type
SourceTimeNS	4	4	Binary	
SymbolIndex	8	4	Binary	
OrderID	12	4	Binary	
Price	16	4	Binary	
Volume	28	4	Binary	Not published in Arca market

Market-specific content

The variable message size can also insulate client code from future field additions that you may not need.

In example 2, an existing message type is 16 bytes long.

Example 2: Release N

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary	Size of the message: 16 bytes
Msg Type	2	2	Binary	The type of this message: 999 – Price message example
SourceTimeNS	4	4	Binary	
SymbolIndex	8	4	Binary	
Price	12	4	Binary	

Look at the Msg Size field to know where the next message starts.

In a future release, a four-byte volume field will be added, increasing the Msg Size to 20 bytes.

If the client wishes to delay upgrading his feed handler for the new content, no coding is needed at the time of the release. Proper coding of the MsgSize field up front allows the client to handle the unforeseen 20-byte format. On his own schedule, the client can upgrade his feed handler to process the new field.

Example 2: Release N+1: a new field is added

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary	Size of the message: 20 bytes
Msg Type	2	2	Binary	The type of this message: 999 – Price message example
SourceTimeNS	4	4	Binary	
SymbolIndex	8	4	Binary	

Look at the Msg Size field to know where the next message starts.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Price	12	4	Binary	
Volume	16	4	Binary	New field

Unmodified clients can handle longer message structure (but can't benefit from new content)

3.2 DATE AND TIME CONVENTIONS

Dates and times are in UTC (Universal Time, Coordinated), and are expressed in nanoseconds since the Unix Epoch (Jan 1, 1970 00:00:00). A complete timestamp consists of two 4-byte fields: seconds since the Unix Epoch, and nanoseconds within the current second, as in a Unix timespec structure.

The Packet Header contains SendTime and SendTimeNS fields to show the time that the packet was published to the wire by the Publisher.

ArcaBook messages additionally contain a timestamp called Source Time to shows the time of the Matching Engine event that caused the publication of this message.

Source Time Reference messages are published per symbol.

Publishing per Matching Engine partition results in a much lower volume of Time Reference messages, and this approach will become standard across all feeds in future releases.

3.3 SEQUENCE NUMBERS

Each message in a given channel is assigned a unique sequence number. Sequence numbers increase monotonically per channel, and can be used to detect publication gaps.

To optimize publication efficiency, the sequence number is not explicitly published in each message. Instead, the Packet Header contains the sequence number of the first message in the packet, along with the number of messages in the packet. Using these fields, the client can easily associate the correct sequence number with each message.

The sequence number combined with the channel ID form a message ID which is unique across the feed.

3.4 SYMBOL SEQUENCE NUMBERS

In addition to the sequence number, many message types explicitly include a field called Symbol Sequence Number, which identifies the message's position in the sequence of all messages published by the feed for a given symbol.

Clients who are tracking only a small number of symbols may opt to ignore sequence numbers and track only Symbol Sequence Numbers for each symbol of interest. If such a client ever experiences a Symbol Sequence Number gap, he can request a refresh for that symbol.

3.5 PRICES

All price fields are published as signed binary integers. Arca Pillar Equities will not publish a negative price.

To interpret a price correctly, the client must use the published price value as a numerator along with the Price Scale Code in the symbol's Symbol Index Mapping Message (Msg Type 3) as follows:

$$Price = \frac{Numerator}{10^{PriceScaleCode}}$$

3.5.1 Maximum Price

It is recommended that at the start of each trading day, firms refer to the Price Scale Code published in the Symbol Index Mapping message. Order and Cancel/Replace messages entered with values larger than the max will be rejected.

- Symbols with price scale code 6, maximum price is $2,147.48
- Symbols with price scale code 4, maximum poce is $214,748.364; except for orders routed to NYSE Floor Broker Systems which have a maximum of $9,999.99
- Symbols with price scale code 3, maximum price is $999,999.999; except for orders routed to NYSE Floor Broker Systems which have a maximum of $999,999.99

The maximum value is determined on a per symbol basis, adjusted nightly based on closing last sale.

Price Scale	Closing Last Sale Threshold	Max Price
6	< $500.00	$2,147.480000
4	>= $500.00	$214,748.3640
3	>= $100,000.00	$999,999.999

3.6 ORDER ID'S AND TRADE ID'S

The Order ID and the Trade ID in order-based feeds such as NYSE Integrated Feed are binary integers that uniquely identify an order or an execution. Order IDs are valid for the trading day only. .

3.6.1 Matching Trade ID's to order entry messages

At present, Trade IDs do not correspond to any ID in the order entry messaging formats.

After future matching engine releases, Trade IDs will correspond to the exchange-generated ID returned to the client in the Execution Report. This will make it easy for the client to find his own trades in the market data feeds.

3.6.2 Matching Order ID's to order entry messages

At present in the Arca market, Order ID's are 4 bytes long. They can be combined with other fields present in the feeds to match them up with the exchange-generated ID that is returned to the client in the order acknowledgement message on order entry.

After future matching engine releases, Order IDs for all markets

- Will be 8 bytes long
- Will correspond to the 8-byte exchange Order ID returned to the client in the order entry APIs
- Will be unique per market across all symbols

Matching Arca Pillar Order ID with the Order Acknowledgement Message

The Arca matching engine generates a unique 64-bit order ID that consists of four concatenated data values - the Order ID, Market ID, System ID, and GTCIndicator. In all Pillar feeds, the Market ID, and System ID are provided in the Symbol Index Mapping, since these values are static for the trading day. The Order ID and GTCIndicator are contained in the order-related data messages.

The table below shows the data structure required to combine the Order ID, Market ID, System ID, and GTC Indicator to obtain the full UTP matching engine Order ID. The table assumes the client byte ordering is Little Endian. If the client byte ordering is Big Endian, the byte order is reversed.

Matching Arca Pillar Order ID with Order Acknowledgment Message

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
GTCIndicator	0	1	Binary	This field specifies if Trade Order ID is a GTC order: • 0 – Day Order • 1- GTC Order
System ID	1	1	Binary	Unique ID for a single ME instance (server) ID of the originating matching engine instance (server). This value is found in the Symbol Index Mapping message's ID field
Market ID	2	2	Binary	ID of the Originating market in the Symbol Index Mapping

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
OrderID	4	4	Binary	Order ID

3.7 SYMBOL INDEXES

In most proprietary data feeds, symbol-specific referential data is published in a Symbol Index Mapping Message (Msg Type 3) at system startup. Symbol Index Mapping messages appear in each channel only for the symbols that appear in that channel.

The Symbol Index Mapping message includes the ASCII symbol in NYSE format along with a unique ID called a Symbol Index.

Symbol Indexes are the same for each symbol every day and the same across all NYSE equity markets.

If more than one Symbol Index Mapping message is received for the same symbol within a trading day, the correspondence between the Symbol and the Symbol Index will not change, but other field values might. In this case, the latest field values override any earlier values, but do not apply retroactively.

Any client who misses this initial spin can request a refresh of Symbol Indexes by sending a Symbol Index Mapping Request Message (Msg Type 13) to the Request Server. The requested Symbol Index Mapping messages will be re-published over the Refresh channels.

4. Messages Sent by the Publisher

4.1 SYMBOL INDEX MAPPING MESSAGE (MSG TYPE 3)

This message is published over the real-time data channels at system startup or in the context of a refresh sequence after a Matching Engine or Pillar Publisher failover. It provides referential data for a single specified symbol.

See Symbol Indexes for more information.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
Msg Size	0	2	Binary	Size of the message: 44 bytes
Msg Type	2	2	Binary	The type of this message: • 3 – Symbol Index Mapping Message
SymbolIndex	4	4	Binary	The unique ID of this symbol for all products within this market. This ID cannot be used to cross reference a security between markets.
Symbol	8	11	ASCII	Null-terminated ASCII symbol in NYSE Symbology. For more information, see the NYSE Symbology Spec.
Reserved	19	1	Binary	This field is reserved for future use
Market ID	20	2	Binary	ID of the Originating Market: 3 – NYSE Arca Equities
System ID	22	1	Binary	ID of the Originating matching engine server.
Exchange Code	23	1	ASCII	Exchange where the symbol is listed: • 'A' – NYSE American • 'M' - NYSE Texas • 'N' – NYSE • 'L' - LTSE • 'P' – NYSE Arca • 'Q' – NASDAQ • 'Z' – BATS
PriceScaleCode	24	1	Binary	Specifies placement of the decimal point in price fields for this security. See Prices .
Security Type	25	1	ASCII	Type of Security used by all Pillar markets: • 'A' – ADR • 'C' - COMMON STOCK • 'D' – DEBENTURES • 'E' – ETF

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
				• 'F' – FOREIGN • 'H' – AMERICAN DEPOSITARY SHARES • 'I' – UNITS • 'L' – INDEX LINKED NOTES • 'M' - MISC/LIQUID TRUST • 'O' – ORDINARY SHARES • 'P' - PREFERRED STOCK • 'R' – RIGHTS • 'S' - SHARES OF BENEFICIARY INTEREST • 'T' – TEST • 'U' – UNITS • 'W' – WARRANT
Lot Size	26	2	Binary	Round lot size in shares.
PrevClosePrice	28	4	Binary	The previous day's closing price for this security.
PrevCloseVolume	32	4	Binary	The previous day's closing volume for the security.
Price Resolution	36	1	Binary	• 0 - All Penny • 1 - Penny/Nickel • 5 - Nickel/Dime
Round Lot	37	1	ASCII	Round Lots Accepted: • 'Y' – Yes • 'N' – No
MPV	38	2	Binary	Note: This field is available on NYSE and NYSE MKT only, and is left as a future enhancement on Arca. Clients will be notified upon availability.
Unit of Trade	40	2	Binary	This field specifies the security Unit of Trade in shares. Valid values are 1, 10, 50 and 100 Note: This field is available on NYSE and NYSE MKT only, and is left as a future enhancement on Arca. Clients will be notified upon availability.
Reserved	42	2	Binary	Reserved for future use. Disregard any content.

4.2 SECURITY STATUS MESSAGE (MSG TYPE 34)

This message informs clients of changes in the status of a specific security, such as Trading Halts, Short Sale Restriction state changes, etc. Security Status of "B" is published once the Pillar Gateways begin accepting orders on a given market, e.g. NYSE Arca Pillar publisher data feeds will publish this security status at 2:30am ET.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 22 Bytes
MsgType	2	2	Binary	The type of this message: • 34 – Security Status Message
SourceTime	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
SourceTimeNS	8	4	Binary	The nanosecond offset from the SourceTime
SymbolIndex	12	4	Binary	The unique ID of the symbol in the Symbol Index msg
SymbolSeqNum	16	4	Binary	The unique ID of this message in the sequence of messages published for this specific symbol.
Security Status	20	1	ASCII	The new status that this security is transitioning to. The following are Halt Status Codes: • '4' - Trading Halt • '5' - Resume • '6' - Suspend The following are Short Sale Restriction Codes (published for all symbols traded on this exchange): • 'A' – Short Sale Restriction Activated (Day 1) • 'C' – Short Sale Restriction Continued (Day 2) • 'D' - Short Sale Restriction Deactivated Market Session values : • 'P' – Pre-opening • 'B' - Begin accepting orders • 'E' – Early session • 'O' – Core session • 'L' – Late session • 'X' – Closed If this security is not halted at the time of a session change, the Halt Condition field = ~. If this security is halted on a session change, Halt Condition is

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
				non-~, and the security remains halted into the new session. The following values are the Price Indication values: • 'I' – Halt Resume Price Indication • 'G' – Pre-Opening Price Indication
Halt Condition	21	1	ASCII	• ' ' - Security not delayed/halted • 'D' - News released • 'I' - Order imbalance • 'P' - News pending • 'M' – LULD pause • 'X' - Equipment changeover • 'A' - Additional Information Requested • 'C' - Regulatory Concern • 'E' - Merger Effective • 'F' - ETF Component Prices Not Available • 'N' - Corporate Action • 'O' - New Security Offering • 'V' - Intraday Indicative Value Not Available • '6' - Suspend Market Wide Circuit Breakers: • 1 - Market Wide Circuit Breaker Halt Level 1 • 2 - Market Wide Circuit Breaker Halt Level 2 • 3 - Market Wide Circuit Breaker Halt Level 3

4.3 SEQUENCE NUMBER RESET MESSAGE (MSG TYPE 1)

This message is sent to reset the Message Sequence Number at start of day, or in response to failures.

This message always appears in its own dedicated packet with a Sequence Number of 1 (the new, reset number). The packet Delivery Flag is normally 12, as on system startup, but during failover events it is set to 10.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTIO
MsgSize	0	2	Binary	Size of the message: 14 Bytes
MsgType	2	2	Binary	The type of this message: • 1 – Sequence Number Reset message
SourceTime	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
SourceTimeNS	8	4	Binary	The nanosecond offset from the SourceTime
ProductID	12	1	Binary	The unique ID for this NYSE feed listed in the feed's client specification.
ChannelID	13	1	Binary	The ID of the multicast channel over which the packet was sent.

4.4 SOURCE TIME REFERENCE MESSAGE (MSG TYPE 2)

For high-volume feeds such as Integrated, this message is sent at the start of every second during periods of active data publication. Unlike some control messages, Source Time Reference messages can come in packets containing market data messages.

The client can concatenate the SourceTime field with the SourceTimeNS field in subsequent market data messages to get full 8-byte Matching Engine event timestamps. The contents of the ID field can be linked via the Symbol Index Mapping Message (Msg Type 3) to the applicable data messages.

See Date and Time Conventions for more information.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 16 bytes
MsgType	2	2	Binary	The type of message: 2 – Source Time Reference Message
ID	4	4	Binary	Symbol Index of the symbol to which this message applies.
SymbolSeqNum	8	4	Binary	The unique ID of this message in the sequence of messages published for this specific symbol.
SourceTime	12	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.

4.5 SYMBOL CLEAR MESSAGE (MSG TYPE 32)

In case of a failure and recovery of a Matching Engine or an Pillar Publisher, the publisher may send a full state refresh for every symbol affected. This kind of unrequested refresh is preceded by a Symbol Clear message. The client should react to receipt of a Symbol Clear message by clearing all state information for the specified symbol in anticipation of receiving a full state refresh.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 20 Bytes
MsgType	2	2	Binary	The type of this message: • 32 – Symbol Clear
SourceTime	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
SourceTimeNS	8	4	Binary	The nanosecond offset from the SourceTime
SymbolIndex	12	4	Binary	The unique ID of the symbol in the Symbol Index msg
NextSourceSeqNum	16	4	Binary	The sequence number in the next message for this symbol

4.6 TRADING SESSION CHANGE MESSAGE (MSG TYPE 33)

This message announces the start of a new trading session for a specified symbol. It is only used in the Arca market.

When processing the Arca Integrated and Arcabook feeds, on receipt of a Trading Session Change message, all orders that are not eligible for the current or future trading sessions should be deleted from the book. No explicit deletes will be sent.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 21 Bytes
MsgType	2	2	Binary	The type of this message: • 33 – Trading Session Change
SourceTime	4	4	Binary	The time when this msg was generated in the order book, in seconds since Jan 1, 1970 00:00:00 UTC.
SourceTimeNS	8	4	Binary	The nanosecond offset from the SourceTime
SymbolIndex	12	4	Binary	The unique ID of the symbol in the Symbol Index msg
SymbolSeqNum	16	4	Binary	The unique ID of this message in the sequence of messages published for this specific symbol.
Trading Session	20	1	Binary	Valid values: • 0x01 - Morning hours • 0x02 - National hours (core)

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
				• 0x04 - Late hours

5. Messages Sent by Refresh and Retrans Servers Only

5.1 REFRESH HEADER (MSG TYPE 35)

The first message in each packet of refresh messages published over the Refresh multicast channels is of this type. Valid values for the DeliveryFlag in the PacketHeader are:

- 17 – Only one packet in Refresh sequence
- 18 – Start of Refresh sequence
- 19 – Part of a Refresh sequence
- 20 – End of Refresh sequence

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 16 Bytes
MsgType	2	2	Binary	The type of this message: • 35 – Refresh Header Message
CurrentRefreshPkt	4	2	Binary	The current refresh packet in the update
TotalRefreshPkts	6	2	Binary	The total number of refresh packets you should expect in the update
LastSeqNum	8	4	Binary	The last sequence number sent on the channel for any symbol. The refresh is the state of the order book as of this sequence number.
LastSymbolSeqNum	12	4	Binary	The last symbol sequence number sent for this symbol. The refresh is the symbol state of this symbol as of this symbol sequence number.

5.1.1 Shortened Refresh Header

The first message in the first packet for a given symbol is a full 16-byte Refresh Header message.

Every other packet for the same symbol contains an 8-byte Refresh Header. The LastSeqNum and the LastSymbolSeqNum fields are removed so as not to send duplicate information in every packet.

5.1.2 Refresh Example

Assuming this refresh of a single symbol requires three packets:

The first, second and third Packet structures look as follows:

PACKET HDR delivery = first	**FULL REFRESH HDR**	**MESSAGE 1**	**MESSAGE 2**	**...**	**MESSAGE N**

PACKET HDR delivery = part	**SHORT REFRESH HDR**	**MESSAGE 1**	**MESSAGE 2**	**...**	**MESSAGE N**

PACKET HDR delivery = last	SHORT REFRESH HDR	MESSAGE 1	MESSAGE 2	...	MESSAGE N

For a depth of book feed such as Integrated or ArcaBook, the sequence of refresh messages per symbol consists of the following message types:

1. Symbol Index Mapping Message (Msg Type 3)
2. Imbalance Message (Msg Type 105), if there is a current imbalance
3. Security Status Message (Msg Type 34)
4. Trade Session Change (Msg Type 33), indicates the current trading session (applies to Arca market only)
5. Add Order Refresh (Msg Type 106/108), repeated as needed to specify the book state for this symbol

5.1.3 Header Fields in the Refresh Channels

5.1.3.1 Refresh response to a request for all Symbol Index Mapping messages

There are no Refresh Header messages

- First packet — Delivery Flag =18 (START of refresh)
- Intermediate packets — Delivery Flag = 19 (PART of refresh)
- Last packet — Delivery Flag = 20 (END of refresh)

** If total number of symbols on the channel is less than 32, only END of refresh is published.

5.1.3.2 Refresh response to a request for a single Symbol Index Mapping message

There is no Refresh Header message.

- One packet is sent — Delivery Flag = 17 (ONE packet in the refresh)

5.1.3.3 Refresh response to a request for a full refresh of all symbols

Each packet contains messages for a single symbol only.

- All packets for the first symbol — Delivery Flag =18 (START of refresh)
- All packets for intermediate symbols — Delivery Flag = 19 (PART of refresh)
- All packets for the last symbol — Delivery Flag = 20 (END of refresh)

The first message in each packet is a Refresh Header.

For each symbol:

- The currentRefreshPkt and totalRefreshPkts fields in the Refresh Header apply to this symbol only.
- The first packet contains a full Refresh Header (16 bytes). The LastSequenceNumber field contains the sequence number of the last message processed in this channel for any symbol. The LastSymbolSeqNum field contains the last Symbol Sequence Number processed for this symbol.
- All subsequent packets contain a short Refresh Header (8 bytes).

** If total number of symbols on the channel is less than 2, only END of refresh is published.

5.1.3.4 Refresh response to a request for a full refresh of a single symbol

- If there are multiple packets in the response Delivery Flags = 19 (PART of refresh)
- If there is only one packet in the response Delivery Flag = 17 (ONE packet in the refresh sequence)

All packets begin with a Refresh Header message.

- The first packet contains a full Refresh Header (16 bytes).
- The first packet for a symbol contains a full Refresh Header (16 bytes). The LastSequenceNumber field contains the sequence number of the last message processed in this channel for any symbol. The LastSymbolSeqNum field contains the last Symbol Sequence Number processed for this symbol.
- All subsequent packets contain a short Refresh Header (8 bytes).

** If there are no symbols on the respective channel, the refresh request will be acknowledged, but no response will be published on the multicast response channels.

5.1.3.5 Differences between Legacy RCF and new Pillar Request Server

The key behavioral differences are the Legacy RCF service and the new Pillar Request Server are as follows:

1. Pillar Request server will accept Client connection to Module A and Module B but will accept connection request only on last/recently connected session. E.g. if a client connects to B while it was already having a connection with A module, the Pillar Request Server will disconnect A connection. In essence, new connection will override the old connection.
2. Retransmission requests will be allowed from Sequence Number 1. The Legacy RCF supports only the last 1 Million Sequence Numbers.
3. Once a client establishes a TCP/IP connection, the Request Server will send a heartbeat to the client approximately every 60 seconds. Clients must respond to with a Heartbeat Response message within 5 seconds, otherwise the Request Server will assume the client or the network has failed and close the connection.
4. Client wild card requests to the Pillar Request Server:
    a. Max Number of Full Refresh Wild Card Requests = 500 per UserId
    b. Max Number of Symbol Index Mapping Wild Card Requests = 500 per UserId
5. For out of bound sequence requests, Pillar Request Server will not send Invalid Request Response with Status Code "2 – Rejected due to invalid sequence range". Examples:
    a. If last processed Seq is 1000 and Request comes for 900-1100, We will send MsgUnavailable for 1001-1100
    b. If last processed Seq is 1000 and Request comes for 1100-1200, We will send MsgUnavailable for 1100-1200.
6. In the Legacy RCF service, only the first trailing character of the SourceID had to be NULL. In the new Pillar Request Server, there are stricter validations on any invalid characters. Example:
    a. ABCDEFG<NULL><JUNK><JUNK> will be accepted as a valid sourceID in the Legacy RCF but this sourceID format will be rejected in the new Pillar Request Server. SourceID format should be ABCDEFG<NULL><NULL><NULL>.
7. With the Pillar implementation, clients should use the LastSeqNum in the RefreshHeader, keeping in mind, that this is a symbol-based recovery. Clients can apply the buffered/new messages higher than the LastSeqNum on top of the Refresh State, per Symbol. Example:

a. When requesting a refresh for all symbols on a Pillar Channel, the LastSeqNum will not be the same per symbol. In the Legacy RCF implementation, the value for LastSeqNum was set to the same number across all symbols.

5.2 MESSAGE UNAVAILABLE MESSAGE (MSG TYPE 31)

This message will be sent over the Retransmission multicast channels to inform clients of unavailability of a range of messages (or part of a range) for which they may have requested a retransmission.

The Message Unavailable message will be sent as the only message in a packet, and the Packet Header Delivery Flag will be set to 21.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 14 Bytes
MsgType	2	2	Binary	The type of this message: • 31 – Message Unavailable
BeginSeqNum	4	4	Binary	The beginning sequence number of the unavailable range of messages.
EndSeqNum	8	4	Binary	The ending sequence number of the unavailable range of messages.
ProductID	12	1	Binary	The unique ID of the feed for which the retransmission was requested (listed in the feed's client specification).
ChannelID	13	1	Binary	The ID of the multicast channel for which the retransmission was requested.

6. Pillar Request Server - Client Request Message

6.1 RETRANSMISSION REQUEST MESSAGE (MSG TYPE 10)

Clients who have experienced a sequence number gap and need a retransmission of the missed messages should send a Retransmission Request message via TCP to the Request Controller. A Request Response message will be sent over the TCP connection back to the client, and if the request was valid, the requested message(s) will be re-published over the relevant Retransmission multicast channel.

The retransmitted message(s) will have the same message format and content as the original messages that were missed.

Retransmission Requests should be sent in a packet whose Packet Header Delivery Flag is set to 11, and which contains a valid sequence number.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 24 Bytes
MsgType	2	2	Binary	The type of this message: 10 – Retransmission Request message
BeginSeqNum	4	4	Binary	The beginning sequence number of the range of messages to be retransmitted.
EndSeqNum	8	4	Binary	The end sequence number of the range of messages to be retransmitted.
SourceID	12	10	ASCII	The ID of the client requesting this retransmission . This field is up to 10 characters, null terminated.
ProductID	22	1	Binary	The unique ID of the feed for which a retransmission is requested (listed in the feed's client specification).
ChannelID	23	1	Binary	The ID of the multicast channel on which the gap occurred.

6.2 REFRESH REQUEST MESSAGE (MSG TYPE 15)

Clients who have experienced a failure and need a refresh of the state of one or all symbols in a specific channel should send a Retransmission Request message via TCP to the Request Controller. A Request Response message will be sent over the TCP connection back to the client, and if the request was valid, the requested message(s) will be published over the relevant Refresh multicast channel.

Retransmission Requests should be sent in a packet whose Packet Header Delivery Flag is set to 11, and which contains a valid sequence number.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 20 Bytes
MsgType	2	2	Binary	The type of this message: 15 – Refresh Request Message
SymbolIndex	4	4	Binary	The ID (from the Symbol Index msg) of the symbol for which a refresh is requested. To request a refresh for all symbols in the channel, set this field to 0.
SourceID	8	10	ASCII	The ID of the client requesting this refresh. This field is up to 10 characters, null terminated.
ProductID	18	1	Binary	The unique ID of the feed for which the refresh is requested (listed in the feed's client specification).
ChannelID	19	1	Binary	The ID of the multicast channel for which the refresh is requested.

6.3 SYMBOL INDEX MAPPING REQUEST MESSAGE (MSG TYPE 13)

This message is sent by clients via TCP/IP requesting the Symbol Index Mapping messages for one or all symbols in a specified channel.

The Symbol Index Mapping Request messages should be sent in a packet whose Packet Header Delivery Flag is set to 11, and which contains a valid sequence number.

FIELD NAME	OFFSET	SIZE (BYTES)	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 21 Bytes
MsgType	2	2	Binary	The type of this message: 13 – Symbol Index Mapping Request Message
SymbolIndex	4	4	Binary	The ID (from the Symbol Index msg) of the symbol for which a refresh is requested. To request a refresh for all symbols in the specified channel, set this field to 0.
SourceID	8	10	ASCII	The ID of the client requesting this symbol refresh. This field is up to 10 characters, null terminated.
ProductID	18	1	Binary	The unique ID of the feed for which the refresh is requested (listed in the feed's client specification).
ChannelID	19	1	Binary	The ID of the multicast channel for which the refresh is requested.
RetransmitMethod	20	1	Binary	The delivery method for the requested symbol index mapping information. Valid values: • 0 – deliver via UDP

6.4 HEARTBEAT RESPONSE MESSAGE (MSG TYPE 12)

Clients who remain connected to the Retransmission Server intraday must respond to a Heartbeat with a Heartbeat Response message within 5 seconds. If no timely client response is received, the connection will be closed.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 14 Bytes
MsgType	2	2	Binary	The type of this message: • 12 – Heartbeat Response message
SourceID	4	10	ASCII	The ID of the connected client . This field is up to 10 characters, null terminated.

7. Messages Sent by Request Server

7.1 REQUEST RESPONSE MESSAGE (MSG TYPE 11)

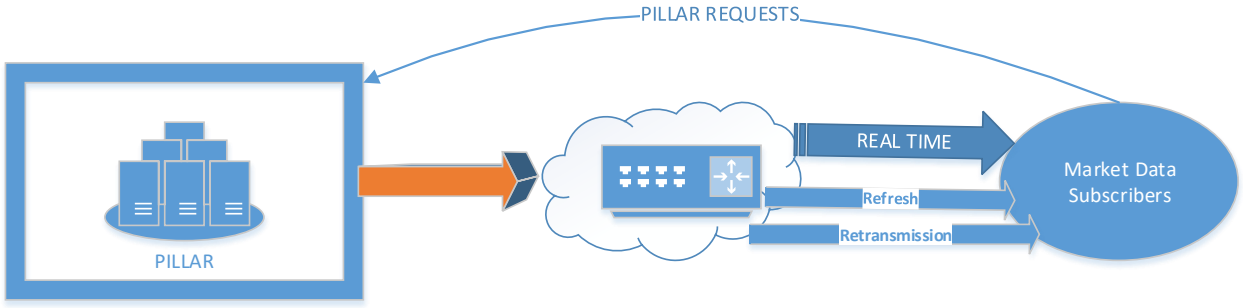
This message will be sent immediately via TCP/IP in response to the client's request for retransmission, refresh or Symbol Mapping messages.

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
MsgSize	0	2	Binary	Size of the message: 29 Bytes
MsgType	2	2	Binary	The type of this message: • 11 – Request Response Message
RequestSeqNum	4	4	Binary	The sequence number of the request message sent by the client. This can be used by the client to couple this response with the original request message.
BeginSeqNum	8	4	Binary	For Retrans Request responses, the beginning sequence number of the requested retransmission range. For responses to Refresh or Symbol Mapping Requests, 0.
EndSeqNum	12	4	Binary	For Retrans Request responses, the ending sequence number of the requested retransmission range. For responses to Refresh or Symbol Mapping Requests, 0.
SourceID	16	10	ASCII	The ID of the client making the request. This field is up to 10 characters, null terminated.
ProductID	26	1	Binary	The unique ID of the feed for which the request was made (listed in the feed's client specification).
ChannelID	27	1	Binary	The ID of the multicast channel for which the request was made.
Status	28	1	ASCII	The reason why the request was rejected. Valid values: • '0' – Message was accepted • '1' – Rejected due to an Invalid Source ID • '2' – Rejected due to invalid sequence range • '3' – Rejected due to maximum sequence range (see threshold limits) • '4' – Rejected due to maximum request in a day

FIELD NAME	OFFSET	SIZE	FORMAT	DESCRIPTION
				• '5' – Rejected due to maximum number of refresh requests in a day • '6' – Rejected. Request message SeqNum TTL (Time to live) is too old. Use refresh to recover current state if necessary. • '7' – Rejected due to an Invalid Channel ID • '8' – Rejected due to an Invalid Product ID • '9' – Rejected due to: 1) Invalid MsgType, or 2) Mismatch between MsgType and MsgSize

8. Error Handling via the Pillar Request Server

8.1 PILLAR REQUEST SERVER



Similar to the NYSE Pillar Gateway, the new Pillar Request Server will facilitate market data client requests for Refresh/ Retransmission with the following features:

- In case of dropped multicast packets, the client can connect to the Pillar Request Server via TCP/IP to request retransmissions of missed messages.
- In case of client late start or intraday failure, the client can connect to the Pillar Request Server and request snapshot refreshes of the state of the market.
- At system startup, each channel publishes referential data about all symbols published on the channel. If a client process misses this initial spin of symbol data, clients can connect to the Pillar Request Server and request a refresh of some or all of the missed data.
- This service is subject to Pillar's IP Table filtering in order to safeguard against events similar to denial-of-service attacks. The filtering prevents any client from making further connections to the Pillar Request Server after the client has connected a truly excessive number of times.

Customers are required to certify their market data sessions for refresh/retransmission in the NYSE Pillar CERT environment before activation in production.

8.1.1 Request Processing

Clients may send several requests at the same time with the same Source ID. There is no need to wait for one request to be fulfilled before requesting another one. Responses to all requests are published in the order in which they are received, although overlapping requests may be de-duplicated for efficiency.

While it is possible to connect to the Request Server only as needed, and disconnecting after each request, it is recommended that Customers remain connected to the Request Server for the entire trading day.

8.1.2 Handling Sequence Number Gaps

Since multicast is an unreliable protocol, messages can be dropped. For this reason, clients are advised to process both lines in a channel. If a gap occurs on one line, the gap can be filled immediately from the other.

If a gap occurs on both lines simultaneously, the client can send a Retransmission Request Message (Msg Type 10) via TCP to the Request Server. The Retransmission Request contains a unique client ID called a Source ID, along with the Product and Channel IDs and the sequence number range of the missing messages.

On receipt of a Retransmission Request message, the Request Server will send back a Request Response Message (Msg Type 11). If any of the fields of the the Retransmission Request contained malformed or meaningless information, the request is rejected. If the request is accepted, the Retransmission Server will re-send the requested messages via multicast over the Retransmission channels.

If the request is rejected for exceeding a predefined system limit, the client will be prevented from making any further requests. See Request Quotas. If further requests are required, please contact NYSE.

8.1.3 Request Quotas

The table below summarizes the retransmission/refresh request limitations that are enforced by the Pillar Request Server. The numbers represent thresholds per Client ID.

Any client who has been blocked by exceeding these quotas can connect to the Module B Pillar Request Server.

FEATURE	DESCRIPTION
Max number of packets per Retransmission Request	Retransmission requests for more than 1,000 messages will not be honored.
Max number of Retransmission Requests in a day	Retransmission requests from a client who has already made 10,000 retransmission requests today will not be honored, and the client will be blocked from making retransmission requests for the remainder of the day.
Max number of Refresh Requests in a day	Refresh requests from a client who has already made 5,000 refresh requests today will not be honored.
Max number of Wildcard Refresh Requests in a day	Wildcard Refresh requests from a client who has already made 500 refresh requests today will not be honored.

8.1.4 Retransmission Format

Retransmitted messages have the same message format and content as the originally published messages (including the Sequence Numbers, but they may be packetized differently for best efficiency.

Packets of retransmitted messages have special Delivery Flag values in the Packet Header:

- 13 – Only one packet in retransmission sequence
- 15 – Part of a retransmission sequence

8.1.5 Recovering from Client Late Starts or Intraday Failures

If a client process experiences a late start or an intraday failure, the client will usually want to receive snapshots of the current market state for each symbol before resuming processing of real-time data. To accomplish this, the client can request a refresh from the Pillar Request Server.

1. Subscribe to the Publisher multicast channels. Any messages received should be cached but not processed until all refresh information is processed.
2. Connect to the Pillar Request Server. This connection should be maintained all day.
3. Subscribe to the Refresh multicast channels.
4. Send a Refresh Request Message (Msg Type 15) to the Request Server.

The Refresh Request contains:

1. A unique client ID called a Source ID
2. Product and Channel IDs
3. A Symbol Index, specifying a particular symbol to be refreshed or else if 0, specifying all symbols.

On receipt of a Refresh Request message, the Request Server will send back a Request Response Message (Msg Type 11). If any of the fields of the the Refresh Request contained malformed or meaningless information, the request is rejected. If the request is rejected for exceeding a predefined system limit, the client will be prevented from making any further requests. See Request Quotas. Session related Quota can

be replenished based on User/Client Id Level flag "replenish". If further requests are required, please contact NYSE.

If the request is accepted, the Pilar Request Server will send the snapshot message(s) over the specific Refresh channel. All these messages should be used to rebuild the current state of the order book. Once all refresh messages are processed, messages from the Publisher can now be processed. Note that any messages received whose sequence numbers are lower than the LastSequenceNumber indicated in the refresh sequence should be discarded.

8.1.6 Refresh Message Format

Each refresh packet begins with a Packet Header, followed by a Refresh Header (Msg Type 35).

The Packet Header for a refresh packet has special Delivery Flag values:

- 17 – Only one packet in Refresh sequence
- 18 – Start of Refresh sequence
- 19 – Part of a Refresh sequence
- 20 – End of Refresh sequence

The Refresh Header identifies the position of the current packet is in this sequence of Refresh packets, along with the total number packets in this sequence. By use of the Delivery Flag and the packet sequence information in the Refresh Header, the client can know when the last packet of the refresh sequence has been received.

No dedicated retransmission service is available for the Refresh Server; if message loss is detected in a refresh channel, clients should submit another refresh request.

8.1.7 Refreshing Symbol Information

At system startup, each channel publishes a Symbol Index Mapping Message (Msg Type 3) for each symbol published on this channel.

If a client process misses the initial spin of symbol information, client may wish to receive a refresh of some or all Symbol Index Mapping messages before resuming processing of real-time data. To do this, the client should follow the procedure described in Recovering from Client Late Starts or Intraday Failures, but request a Symbol Index Mapping Request Message (Msg Type 13) to the Request Server instead of a Refresh Request Message.

8.1.8 Symbol Index Mapping Refresh Format

Requested Symbol Index Mapping messages are published by the Refresh Server with the same Packet Header Delivery Flags used for Refresh publications. Refresh Headers are not used in Symbol Index Mapping refreshes.

8.1.9 Pillar Request Server Implementation

1. Pillar Request server will accept Client connection to Module A and Module B but will accept connection request only on last/recently connected session. E.g. if a client connects to B while it was already having a connection with A module, the Pillar Request Server will disconnect A connection. In essence, new connection will override the old connection.
2. Retransmission requests will be allowed from Sequence Number 1.
3. Once a client establishes a TCP/IP connection, the Request Server will send a heartbeat to the client approximately every 60 seconds. Clients must respond to with a Heartbeat Response

message within 5 seconds, otherwise the Request Server will assume the client or the network has failed and close the connection.

4. Client wild card requests to the Pillar Request Server:
    a. Max Number of Full Refresh Wild Card Requests = 500 per Client ID
    b. Max Number of Symbol Index Mapping Wild Card Requests = 500 per Client ID
5. For out of bound sequence requests, Pillar Request Server will not send Invalid Request Response with Status Code "2 – Rejected due to invalid sequence range". Examples:
    a. If last processed Seq is 1000 and Request comes for 900-1100, Pillar will send MsgUnavailable for 1001-1100
    b. If last processed Seq is 1000 and Request comes for 1100-1200, Pillar will send MsgUnavailable for 1100-1200.
6. In the Pillar Request Server, there are strict validations on any invalid characters. Example:
    a. ABCDEFG<NULL><JUNK><JUNK> will be rejected in the Pillar Request Server. Client ID format should be ABCDEFG<NULL><NULL><NULL>.
7. With the Pillar implementation, clients should use the LastSeqNum in the RefreshHeader, keeping in mind, that this recovery is symbol-based. Clients can apply the buffered/new messages higher than the LastSeqNum on top of the Refresh State, per Symbol.
    a. e.g. When requesting a refresh for all symbols on a Pillar Channel, the LastSeqNum will not be the same per symbol.

8.1.10 Request Server Denial of Service

NYSE Pillar engine maintains a running counter of log in attempts (both successful and unsuccessful) and session level rejects per client ID over the course of a trading day. If either of the counters reaches 100, the Pillar Denial of Service (DOS) functionality will be triggered on the Pillar Request Server for that client ID.

On a subsequent connection event, Pillar Request Server will lockout the client id for 60 seconds.

If the firm's client id continues to exhibit DOS behaviors, NYSE operations may shutdown the offending retransmission port for the day. In this event, the firm should reach out to NYSE connectivity support, support@nyse.com

9. Operational Information

9.1 SYSTEM BEHAVIOR ON START AND RESTART

At system startup or at start of system recovery following a failure, Pillar feeds send the following messages over each channel:

- Multicast priming from the primary Publisher's source IPs: a series of Heartbeat packets for several seconds with the Delivery Flag set to 1 and sequence number set to 1, the next expected sequence number.
- Sequence Number Reset Message (Msg Type 1), the sequence number is 1 and the packet DeliveryFlag is 12.
- A full spin of Symbol Index Mapping Messages (Msg Type 3), for securities published on this channel.

9.2 PILLAR PUBLISHER FAILOVER

When failing over to the backup Pillar Publisher, the following refresh information is published.

Note: During the failover refresh, DeliveryFlag fields for all Packet Headers except Heartbeats are set to 10.

1. Multicast priming from the backup Publisher's source IPs: a series of Heartbeat packets for several seconds with the Delivery Flag set to 1 and sequence number set to 1, the next expected sequence number.
2. A Sequence Number Reset Message (Msg Type 1) is sent in its own packet
3. For each symbol, the following are published,
    - Symbol Index Mapping Messages (Msg Type 3)
    - Symbol Clear Message (Msg Type 32)
    - The last Security Status Message (Msg Type 34)
    - The last Trading Session Change Message (Msg Type 33) (applies to the Arca market only)
    - All required refresh messages
    - The last Source Time Reference Message (Msg Type 2)

Once all symbols have been refreshed, Packet Header DeliveryFlag fields return to the normal 11.

9.3 DISASTER RECOVERY SITE

All proprietary data feeds are published out of the NYSE Mahwah data center. In case of catastrophic failure in Mahwah, all affected systems including datafeeds will be coldstarted at the Cermak Disaster Recovery site in Chicago.

The Cermak configuration of channels and multicast groups is identical to Mahwah for all feeds, except that the source IPs are different. The initial publication sequence is as described in System Behavior on Start and Restart.

9.4 ARCABOOK PRODUCTION HOURS

EVENT	NYSE ARCATIME (ET)
Sequence Number Reset	2:03am*
Symbol Mapping	2:03am*
Early Open Auction	4:00am
Core Open Auction and Open	9:30am
Closing Auction and Close	4:00pm
End of Late Session	8:00pm

*Feed start time. No messages are sent before this, but clients might see earlier source times for messages that were generated prior to the feed start time

9.5 CERT TESTING HOURS

NYSE CERT testing hours are approximately 8:00 AM until 6:00 PM.

- Technology Member Services (TMS)
- TMS Hotline : 212-896-2830, or tms@nyse.com

9.6 SYMBOL INDEX MAPPING FILE

For customers that prefer to download the symbol index mapping from an FTP server, a file containing a subset of the index mapping information is available. Refer to the Pillar Common Client specifications.