TRADE MATCH

RTC

TCP/IP Logic Manual

Version 2.0.3

Created Date: 10/23/2001
Change Date: 04/12/2011
Printed Date: 08/23/2011
File: (c:\documentation\tcp_ip\rtc tcp_ip logic manual.doc)

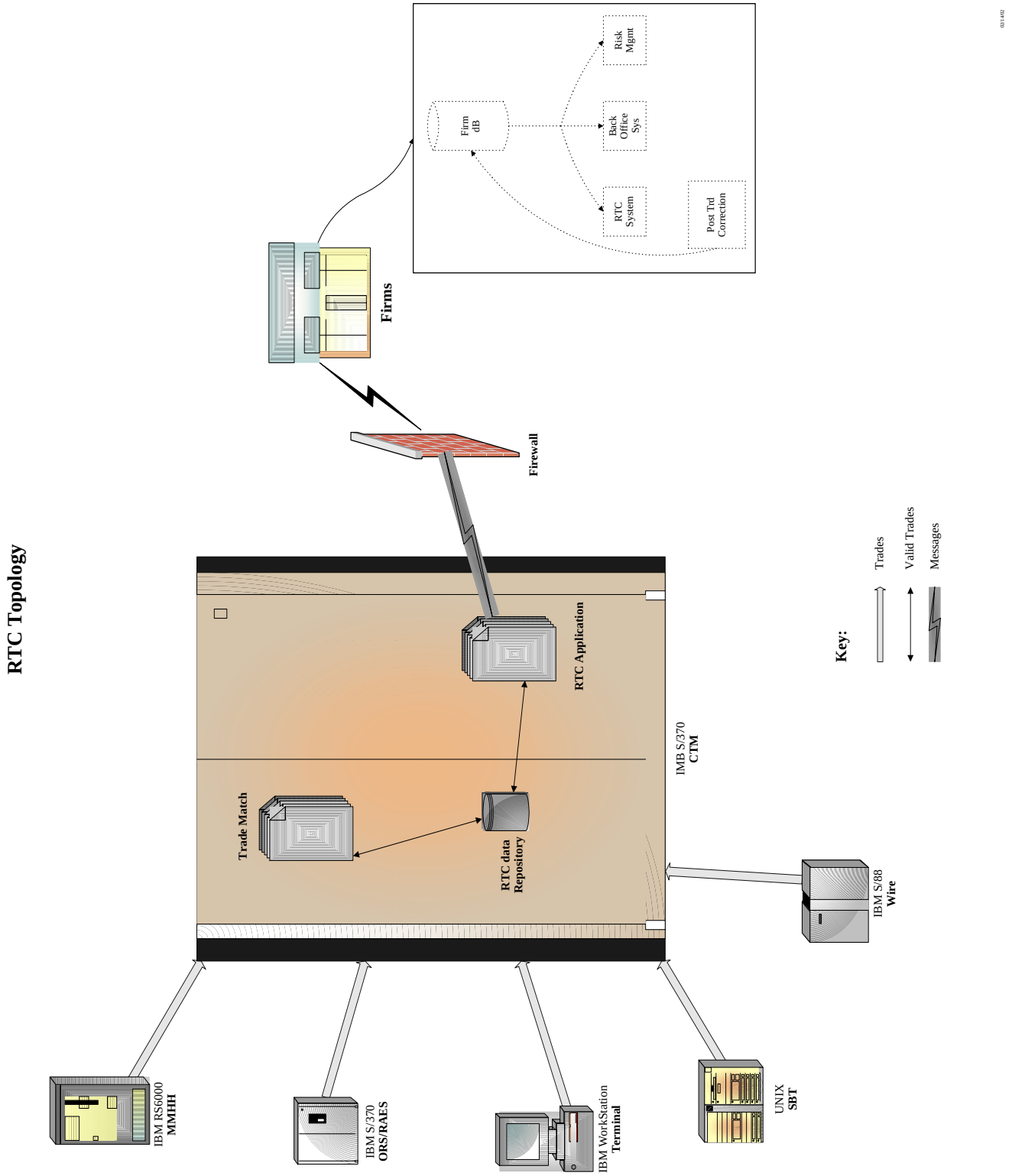
Table of Contents

Change Notices

The following change notices are provided to assist users of the CBOE RTC Services in determining the impact of changes to their applications.

Date	Version	Description of Change
12 Apr 2011	2.0.3	Changes made to Trade Data Copybook for CFLEX: *New copy book Appendix B* • Expiration price format change • Premium price format change • Transaction source value add
20 Apr 2009	2.0.2	Changes made to Trade Data Copybook in compliance with OSI: • Expiration price format change • Expiration date DD value.
20 Nov 2008	2.0.1	Changes made to Trade Data Copybook: New fields add to position 187 to 198.
23 Oct 2001	1.0	New document

1.00 RTC Topology



2.00 Overview of the Real-Time Communication process

2.01 Real-Time Communication

Real-Time Communication (RTC) is an interface which protects* and forwards or sends messages to or from an external source. These messages are instantly sent or received as soon as RTC takes receipt of it. RTC allows an external source to select from a variety of services, depending on the desired processing. The input service was designed to process input trade data, as well as to accept acknowledgements. The output service can provide a real-time source of traded activity, which can be used by any external system. This provides an up-to-the-minute view of traded activity and a means of updating back office application and risk management systems.

* *Protect: Assumes that the Requestor requested an application confirm for all messages exchanged.*

2.02 Continuous Trade Match

Continuous Trade Match (CTM) is the main data repository, referred to from this point as 'central data repository', from which RTC will receive and send to Requestors their trade data. It is also the repository that will receive and process trade data, sent by the Requestors to the RTC interface.

2.03 Messages

RTC messages consist of a header and a data section, where:

- The header section of the message contains the control information, which is an integral part of **every** message exchanged. The header defines the parameters associated with the solicited service. It also assists in distinguishing between the messages being sent back and forth.
- The data section represents the trade data, a predefined layout of this trade record as defined in a copybook provided by CTM. The copybook has been expanded and many fields have been made larger to accommodate for future scheduled enhancements.

2.04 Trades and Valid Trades

Messages destined to be sent to Requestors, which represent trades given to CTM via other applications:

- Trades are validated and stored on the central data repository.
- Once the central data repository receives the trades, RTC application programs will extract the data and verify whether the trades are candidates to be sent to a Requestor.
- When it is determined that the trades are candidates for sending to the Requestor, the trades are copied to the data repository in the format defined for that application, which will ship the data to the Requestor.
- The trades are then built into the RTC TCP/IP message and sent to the Requestor using the output service.
- Trades built into the message are either protected or deleted depending on the Requestors' decision to acknowledge the message with an application confirm.

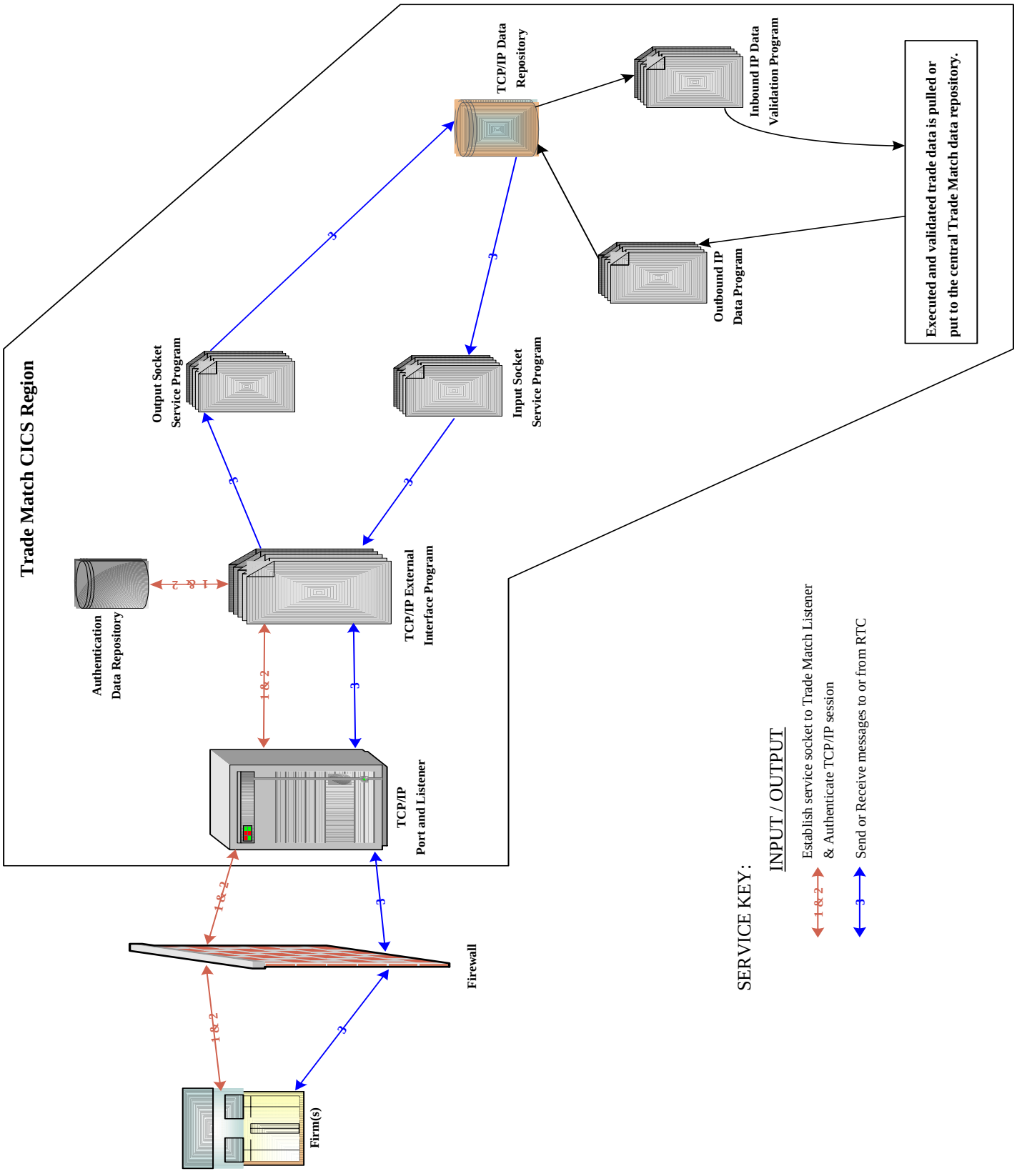
Messages sent to RTC, which represent trades destined for Trade Match:

- The trade data potion of the message is saved on the TCP/IP data repository.
- This trade data is validated for hard-edits. *See definition of hard-edit in glossary.*
- An acknowledgement is sent back to the Requestor if the Requestor elected to receive that acknowledgement.
- Trades flagged with hard-edits are either sent back to the Requestor or saved on the CTM message database for firms to edit via the ITP screens. This decision is based on the Requestor's election to receive application confirm messages.

Apart from trade data messages exchanged, this application will:

- Process echo messages for monitoring purposes.
- Resend messages that were protected, after being sent to the Requestor once before. These are messages that are resent to Requestors because RTC did not receive any application confirm from the Requestor for which the Requestor had elected to respond.
- Send gate control messages to Requestor instructing them to shut down the connection to RTC, due to some issue that CTM needs to resolve.

3.00 RTC TCP/IP Synopsis



4.00 RTC TCP/IP Details

4.01 Types of messages

RTC can process the following messages, which would be information contained in the data section of the message.

a. Inbound message

- An inbound message is essentially a firm trade intended for input into the central data repository.
- This message could contain, in the data section, one or more trade data records that the Requestor wants entered into central data repository for matching and clearing.

b. Outbound message (also known as Drop-Copy)

- A drop-copy message consists of a header plus trade data. The trade data is the trade that was executed on the CBOE trading floor of which the Requestor does not have an original trade copy.
- This trade message would be a copy of the validated trade that originated from an application that feed data to CTM for matching and clearing.
- This message could contain, in the data section, one or more trade data records that RTC will send to the Requestor.

c. Re-send message

- It is essentially an outbound message that the Requestor did not acknowledge with an application confirm message. This message in the data section will contain one or more trades that RTC will send to the Requestor. The trades in this message may not all be re-send trades, but a mixture of trades that are re-sends plus trades that were not previously sent. All re-send trades in a message will be marked in a field in the trade record as re-send. CTM *strongly advises* Requestors to check the re-send field for an indicator to properly process that message on the Requestor side.
- The way this process works is that when a message gets to the TCP/IP data repository and if the output service is established that was opened in an application confirm status (see section 6.01 'data confirm'), that message will be sent to the Requestor and marked as waiting for an application confirm. If the application confirm was never received, that message will be read again and aged for the time, using the time that the trade was sent to the Requestor and adding a predefined timer interval. If that value exceeds the time when the message was read again from the TCP/IP data repository and was still marked as awaiting application confirm, then that trade will be sent again to the Requestor **before** the trades that have not yet been sent to the Requestor. This cycle of re-sending will continue until an application confirms has been received for the trade in question.
- These messages are sent in sequential order. For example, if a change to the order quantity was received by RTC before the change to the broker field, then that is the same sequence the Requestor will receive those messages in the re-send process.

d. Delivery confirmation message (also known as Application Confirms)

- When the Requestor initiates this message, it will instruct RTC to delete the copy of the trade data record(s) from the TCP/IP data repository. It would also mean that those particular trade data records could not be sent again nor recoverable if the Requestor did not store them correctly.
- When this message is initiated by RTC, it will inform the Requestor that the message sent to RTC was successfully received. But this does not mean that the trade data records contained in that message where validated at the time that this message was sent by RTC to the Requestor.
- This message entails the guaranteed delivery of all messages exchanged between the Requestor and RTC.

e. Echo message (also known as heartbeats)

- A heartbeat message is simply a message sent to the Requestor to ensure they are still connected.
- In response to this message, the Requestor will echo the heartbeat message back to RTC.
- All heartbeat messages are initiated by RTC and will be exchange only through the RTCOUTPT socket. Refer to section 6.04.
- CBOE's network operation monitor will register all the echo messages. If at any time these cycles of messages are interrupted, the network monitor will alert the operation staff who will take the necessary step to investigate the problem and determine if the Requestor needs to be contacted.
- A heartbeat message does not contain any trade data.

f. Gate Control Message *(not used any more)*

- ~~This is a message sent to the Requestor by RTC informing them to close their socket(s). This could be because CTM had a problem and needed to re-cycle their application.~~
- ~~RTC could also send this message at end of day to instruct the Requestor to close their service for the day.~~

4.02 Trade data

- A message sent to or received from a Requestor can contain up to a maximum of 20 trade data records.
- Requestors can expect to receive ~~Market Maker Hand Held (MMHH)~~, ~~Retail Automatic Execution System (RAES)~~, ~~WIRE (TP25)~~, ~~Screen Based Trading (SBT)~~ Hybrid Trades, RTC Inbound maintenance trades, CFLEX trades and Terminal entered trades based on their election.
- The trade data that is sent to Requestors are unmatched trades.
- Trade data sent to RTC by the Requestor would be the opposite side of a trade executed on CBOE's trading floor.
- Trade data will be validated for errors, both for trades executed on the CBOE floor and for trades sent to CTM via the RTC service.
- Errors found on the trade data will be marked in the field corresponding to its error field name in the trade data layout (see Appendix A).

4.03 Segregation of trade data

RTC can segregate trade data to Requestors by the following categories:

- Origin - Trade data from any one or all of the points of entry mentioned on section 4.02.
- Format - Trade data that can be sent can consist of only 'adds' or only 'changes' or only 'deletes' or all these formats.
- Executing Firm - Many larger firms have other clearing firm numbers that they manage, and RTC can send them all or any one of the executing firm's trade data to a specified destination.
- Affiliation - Many executing firms clear trades for other smaller firms. For the smaller firms, RTC can directly send them their trade data to a specified destination, with the authorization of the executing firm.

4.04 Queuing

- Queuing is a term RTC uses to define the messages that are waiting for an application confirm from the Requestor in order to purge the trade from the data repository.
- Once a Requestor establishes an output service, messages will queue for that trading day and will be purged at the end of that business day.

4.05 Service

- The input service will consist of a Requestor establishing a connection and socket to RTC. In this service the Requestor can send trade data and/or application confirm messages to RTC.
- The output service will consist of a Requestor establishing a connection and socket to RTC. In this service the Requestor can expect to receive trade data and/or application confirm messages from RTC.
- IP sockets will be authenticated for service type, executing firm number plus its associated destination and password. If any part of this string is invalid, the socket is closed and a message related to its closure is relayed to the Requestor.
- IP sockets opened for an executing firm number plus destination combination will only be able to send and receive messages for that firm and it associated firms.
- Messages sent for firms other than those that the socket was established for, will be closed after two attempts. For example, if firm 0123 with destination ABCD sent trade data for firm 0111 (a firm not affiliated to 0123) to RTC then that trade will be rejected. On the second such encounter, the input socket for firm 0123ABCD will be closed by RTC.
- The trade window at CBOE will handle initial service requests.
- The Requestors via the ITP screens can maintain their message service passwords. This is explained in the RTC TCP/IP user manual.

4.06 Issues

- The method to send or receive messages is NOT available via the Internet. This service is provided only using the Virtual Private Network (VPN) configuration.
- Development of application level code is not restricted to any platform.
- Requestors will need to connect to an IP port residing on the Java platform application at CBOE. The IP port number for the test application is 30102.
- The TCP/IP address for the server on which the test port resides is 170.137.230.44.
- This application does not support FTP.
- The TCP/IP application will functional all day during trading business days.
- Disconnection of the service at end-of-day is at the discretion of the Requestors. Knowing that if the service is disconnected and trading is still in process, then those trades will be queued and sent out upon re-connection. Requestor acknowledges that the latency of these trades will be greater than the normal latency of trade dropped while the connection is active.

4.07 Security

RTC TCP/IP is not available via the Internet. Requestors are required to establish a private dedicated circuit. RTC has in place a second level of security, which will authenticate a password set-up and maintained by the Requestor.

5.00 Application to Application Protocol Layout.

5.01 Overview

The TCP/IP buffer size is 4096, and is sectioned as shown below.

Field	Description	Size	Format	Group level description
Message Length	TCP/IP total message length	4 bytes	ASCII	Total msg size: 4 bytes
Message Type	Response code	2 bytes	ASCII	Application Header: 26 bytes
Origin Name	Origin name of message	8 bytes	ASCII	
Password	Enhanced security	8 bytes	ASCII	
Service Name	Service requested for message	8 bytes	ASCII	
Version Type	Message header version type	2 bytes	ASCII	Future enhancement
Data Rows	Number of rows in blocked data	2 bytes	ASCII	Number of records in blk
Data	Maximum size of data area (block data)	4062 bytes	ASCII	Trade Record: 200 bytes Total in block: 20 records

5.02 Message Length

This is the value of the total size of the data being sent to or received from the Requestor. For better understanding see Section 5.08.

5.03 Message Type

Type	Description
1	Connect Application Session Primary
2	Connect Application Session Secondary
3	Connect Accept
4	Connect Reject
5	Disconnect Application Session Primary
6	Disconnect Application Session Secondary
7	Disconnect Accept
8	Data
9	Data Reject
10	Data with Confirm Request
11	Confirm Response
12	Echo Request
13	Echo Response
14	Disconnect Reject

The appropriate message type code should be inserted in this field. For example, a Requestor should place a '1' when they are requesting a socket to open a service to RTC. Similarly, RTC will return a '3' in this field if the connection was accepted.

5.04 Origin Name

Name	Description
0123ABCD	Where '0123' is the executing firm number and 'ABCD' is the executing firm destination.

The executing firm number is a number assigned to firm for clearing. Those firms that do not have a clearing firm number but clear trades through a firm with a clearing number, can still send and receive trade data to and from RTC. The destination is a four-letter acronym assigned by RTC for the purpose of identifying a firm and also monitoring it via the network configuration at the CBOE operation center.

5.05 Password

Name	Description
Password	To be provided by the Requestor.

This field is used to provide added security for both RTC and the Requestor to ensure that the trade either being sent or being received is really of the Requestor who subscribed to that service. The password can be maintained by the Requestor via the ITP screens and is explained in section 6.00 of this document. The format of this password can be alphanumeric, numeric, or both and must a maximum of eight bytes in length.

5.06 Service Name

Name	Description
RTCINPUT	Service to send trade data to RTC
RTCOUTPT	Service to send trade information to Requestors

These are the service types currently offered by RTC.

5.07 Version Type

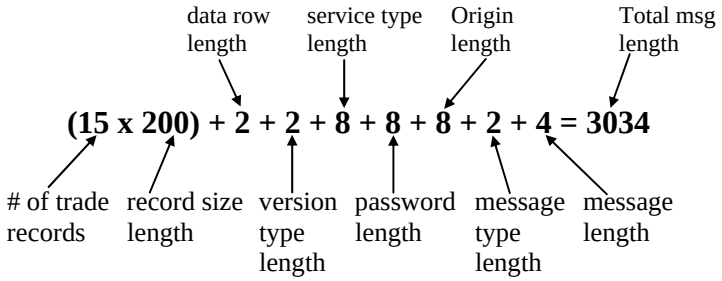
Type	Description
1	RTC TCP/IP application

The version type is a title used in this context for a layout of the buffer area. For this application, RTC has chosen to use version that meets its requirements

5.08 Data Rows

Field	Description
Data Rows	Value determined by the service program that is building the block of data.

The value in this field will be calculated in the service program that is building the block of data to send to either the Requestor or to RTC. One trade data record would be one data row. For example, if the message contains '15' trade records either being sent to or received from RTC the value in the data row field will be '15.' Hand in hand with this field is the calculation of the 'message length' defined in section 3.01. Using this as an example the value of 'message length' will be:



5.09 Data

Name	Description
Data1	200 byte trade data
Data2	200 byte trade data
.	.
.	.
Data19	200 byte trade data
Data20	200 byte trade data

Data area is where the trade data will reside and 20 or fewer of these records could either be sent or received using this format. See Appendix A.

6.00 Application to Application Protocol for Requestors requesting RTC OUTPUT service.

The RTC Output service socket will provide the Requestor with a copy of all its executed trades and also provide a copy of the validated trades that the Requestor may have sent to RTC via the input socket. At the TCP/IP code level the application will always send out a "protocol confirm" by the application receiving a message. For examples of the following procedures in frame format, see Appendix D.

6.01 Initial Process

Requestor needs to 'connect' to CBOE IP address and port and upon connection send the R500 (section 6.02).

6.02 Service Initialization and Authentication

The procedure described here is to request an RTC service initialization and authentication in one message.

- This is the required format.

Field	Description	Size	Format	Field Contain
Transaction Code	Transaction code to start listener	4 bytes	ASCII	R500
Comma	Field required by listener	1 byte	ASCII	,
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD (see section 5.04)
Filler	Filler	1 byte	ASCII	Blank
Password	Firm selected password	8 bytes	ASCII	12345678 (see section 5.05)
Filler	Filler	1 byte	ASCII	Blank
Service Name	Service name for message	8 bytes	ASCII	RTCOUTPT (see section 5.06)
Filler	Filler	1 byte	ASCII	Blank
ASCII/EBCDIC	Format being used to exchange data	1 bytes	ASCII	A or E
Filler	Filler	1 byte	ASCII	Blank
Data confirm	Application confirm of data	1 bytes	ASCII	Y or N
Filler	Filler	5 bytes	ASCII	Blanks

- RTC receives the request, performs authorization and save the relevant information. If accepted, RTC return's a "connection accept" otherwise it return's a "connect reject."

Field	Description	Size	Format	Field Contain
Message Length	TCP/IP total message length	4 bytes	ASCII	0030 *
Message Type	Response code	2 byte	ASCII	03
Service Name	Service name for message	8 byte	ASCII	RTCOUTPT
Password	Firm selected password	8 bytes	ASCII	12345678
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD

** This value represents the calculated value of the length of the message being sent.*

6.03 Data Transfer

- Requestor receives connection accept.
- Requestor sends 'confirm' message to RTC.

Field	Description	Size	Format	Field Contain
Message Length	TCP/IP total message length	4 bytes	ASCII	Calculated value of this msg
Message Type	Response code	2 bytes	ASCII	11
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD
Password	Firm selected password	8 bytes	ASCII	12345678
Service Name	Service name for message	8 bytes	ASCII	RTCOUTPT

- It is important to note here that if there are no trades for RTC to send to the firm than after 60 seconds of non-data transfer activity, RTC will send out a 'heartbeat' (see section 6.04). If there is data to send, it will be immediately blocked and sent.

- RTC builds block of data and sends 'data with confirm' and the data.

Field	Description	Size	Format	Field Contain
Message Length	TCP/IP total message length	4 bytes	ASCII	4034*
Message Type	Response code	2 bytes	ASCII	10
Service Name	Service name for message	8 bytes	ASCII	RTCOUTPT
Password	Firm selected password	8 bytes	ASCII	12345678
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD
Version Type	Message header version type	2 bytes	ASCII	01
Data Rows	Number of rows in blocked data	2 bytes	ASCII	20**
Data	Maximum size of data area	4000 bytes	ASCII	Trade records

* Twenty trade data records where sent, therefore the message length reflects the calculated value.
** Number of trade records blocked in the data field. Note: each trade record is 200 bytes in length.

RTC would have stored these messages as 'awaiting application confirm' or as 'completed' based on the data confirm field. In the case of 'awaiting application confirm,' the requestor would need to send the application confirm message via the input service for RTC to mark those messages as 'complete'.

- Requestor receives 'data with confirm.'
- Requestor sends 'confirm' message to RTC.

Field	Description	Size	Format	Field Contain
Message Length	TCP/IP total message length	4 bytes	ASCII	Calculated value of this msg
Message Type	Response code	2 bytes	ASCII	11
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD
Password	Firm selected password	8 bytes	ASCII	12345678
Service Name	Service name for message	8 bytes	ASCII	RTCOUTPT

Requestor *must* send a message type 11 back to RTC for the message type 10 that RTC sends to the firm, *before* the firm can send an application confirm via the RTCINPUT socket to RTC. This is the requirement for the output socket opened by the requestor as "data confirm" set to "Y" (see 6.01, first bullet point).

6.04 Echo Exchange

- RTC, upon determining that there are no trade data to send, will initiate an echo message to the Requestor.
- The echo message interval is set at preset timer interval.

Field	Description	Size	Format	Field Contain
Message Length	TCP/IP total message length	4 bytes	ASCII	Calculated value of this msg
Message Type	Response code	2 bytes	ASCII	12
Service Name	Service name for message	8 bytes	ASCII	RTCOUTPT
Password	Firm selected password	8 bytes	ASCII	12345678
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD

- Requestor sends ‘echo response’ message to RTC.

Field	Description	Size	Format	Field Contain
Message Length	TCP/IP total message length	4 bytes	ASCII	Calculated value of this msg
Message Type	Response code	2 bytes	ASCII	13
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD
Password	Firm selected password	8 bytes	ASCII	12345678
Service Name	Service name for message	8 bytes	ASCII	RTCOUTPT

This exchange of data and echo messages will continue until the firm closes the service socket.

7.00 Application to Application Protocol for Requestors requesting RTC INPUT service.

The RTC Input service accepts messages from the Requestor. These are messages that the Requestor needs to post the CTM's central data repository for matching and clearing. As mentioned previously, these messages are validated for critical fields of the trade data. At the TCP/IP code level the application will always send out a "protocol confirm" by the application receiving a message. For examples of the following procedures in frame format, see Appendix C.

7.01 Initial Process

Requestor needs to 'connect' to CBOE IP address and port and upon connection send the R500 (section 7.02).

7.02 Service Initialization and Authentication

The procedure described here is to request an RTC service initialization and authentication in one message.

- This is the required format.

Field	Description	Size	Format	Field Contain
Transaction Code	Transaction code to start listener	4 bytes	ASCII	R500
Comma	Field required by listener	1 byte	ASCII	,
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD (see section 5.04)
Filler	Filler	1 byte	ASCII	Blank
Password	Firm selected password	8 bytes	ASCII	12345678 (see section 5.05)
Filler	Filler	1 byte	ASCII	Blank
Service Name	Service name for message	8 bytes	ASCII	RTCINPUT (see section 5.06)
Filler	Filler	1 byte	ASCII	Blank
ASCII/EBCDIC	Format being used to exchange data	1 bytes	ASCII	A or E
Filler	Filler	1 byte	ASCII	Blank
Data confirm	Application confirm of data	1 bytes	ASCII	Y or N
Filler	Filler	5 bytes	ASCII	Blanks

- RTC receives the request, performs authorization and saves the relevant information. If accepted, RTC return's a "connect accept" otherwise it return's a "connect reject."

Field	Description	Size	Format	Field Contain
Message Length	TCP/IP total message length	4 bytes	ASCII	0030 *
Message Type	Response code	2 byte	ASCII	03
Service Name	Service name for message	8 byte	ASCII	RTCINPUT
Password	Firm selected password	8 bytes	ASCII	12345678
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD

* *This value represents the calculated value of the length of the message being sent.*

At this point the RTCINPUT socket is open and active. The requestor will have to initiate any activity on this socket with a "message type" of 10. See 7.03.

7.03 Data Transfer

- Requestor sends request of 'data with confirm.'

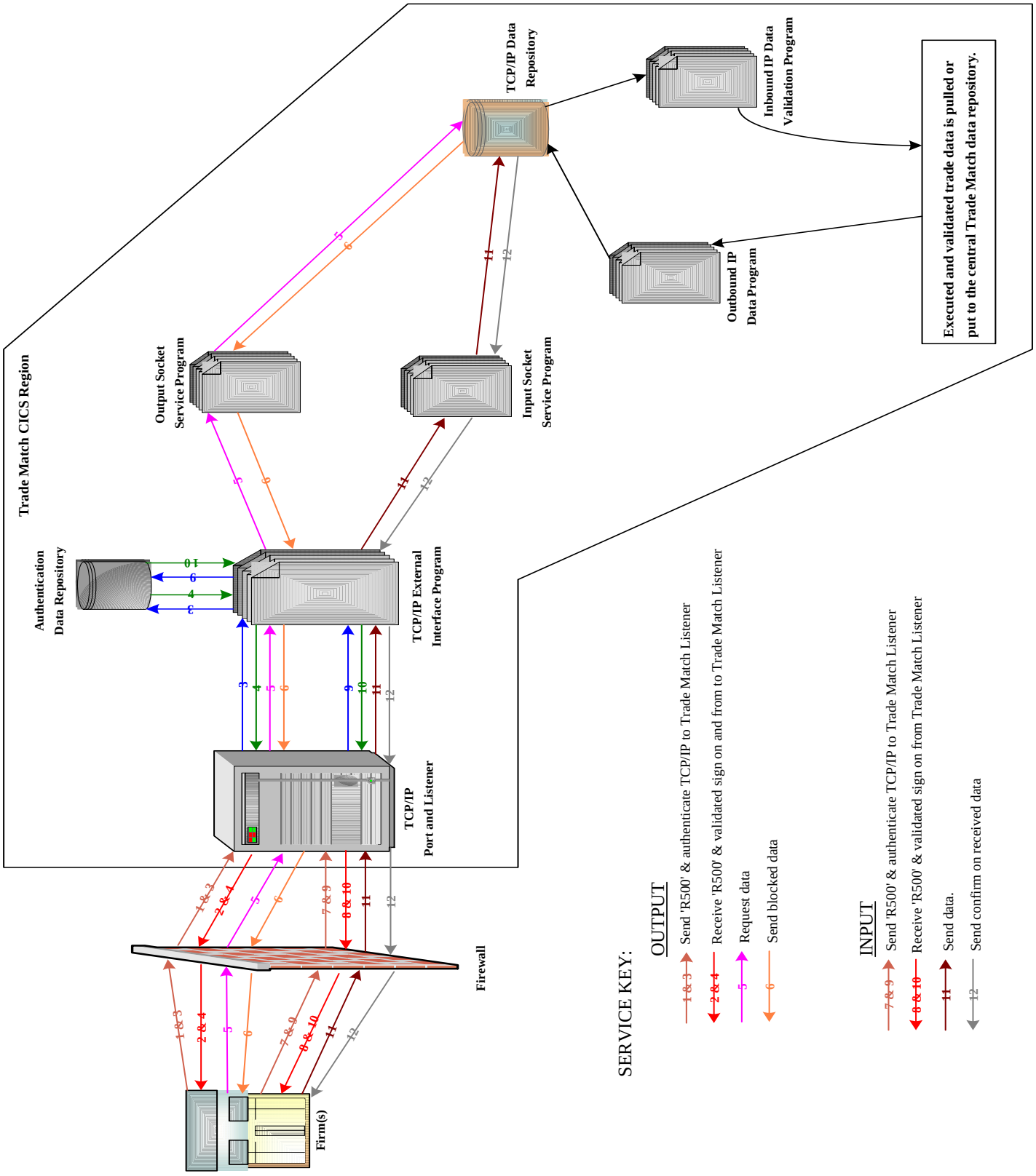
Field	Description	Size	Format	Field Contain
Message Length	TCP/IP total message length	4 bytes	ASCII	4034*
Message Type	Response code	2 byte	ASCII	10
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD
Password	Firm selected password	8 bytes	ASCII	12345678
Service Name	Service name for message	8 byte	ASCII	RTCINPUT
Version Type	Message header version type	2 bytes	ASCII	01
Data Rows	Number of rows in blocked data	2 bytes	ASCII	20**
Data	Maximum size of data area	4000 bytes	ASCII	Trade records

* Twenty trade data records were sent; therefore the message length reflects the calculated value.
** Number of trade records blocked in the data field. Note: each trade record is 200 bytes in length.

- RTC receives the request with data.
- RTC will add the trades to the IPA dB and send 'protocol confirm'.

Field	Description	Size	Format	Field Contain
Message Length	TCP/IP total message length	4 bytes	ASCII	0030
Message Type	Response code	2 byte	ASCII	11
Service Name	Service name for message	8 byte	ASCII	RTCINPUT
Password	Firm selected password	8 bytes	ASCII	12345678
Origin Name	Origin name of message	8 bytes	ASCII	0123ABCD

8.00 RTC TCP/IP Functionality



9.00 Frequently asked questions.

9.01 Why was TCP/IP chosen?

The TCP/IP communication method was chosen because it is now widely used and allows implementation over multiple platforms to a mainframe application (i.e. mainframe, mid-size, PC, UNIX).

9.02 What is socket?

A *socket* is a special type of *file handle*, which is used by a process to request network services from the operating system. Sockets are a mechanism that allows programs to communicate across a network.

9.03 What is port?

It is a gateway to connect sockets.

9.04 Will CBOE network control know if a Requestor socket is down?

Yes, the CBOE side of the interface has a constantly updated network monitor function. This monitor will send out an "are you there?" echo pulse to all remote connections at regular intervals. The network control operators also have the ability to selectively echo certain Requestor(s) or all Requestors. In addition the monitor will highlight any connection which has not "talked to us" for a period of time.

9.05 What happens to my messages if the output socket drops for any length of time?

If the output socket terminates, the external interface program will not initiate the respective service program to read the IP data repository. Since a read is not initiated, data message blocks will not be built and the messages will remain on the data repository until the Requestor re-establishes the output socket.

9.06 What other software do I need besides the RTC interface package to connect?

Requestor requirements will vary by platform. However, some overall conceptual needs are:

1) A transaction processor suitable for real-time multi-tasking communications.
2) A host operating system that supports the necessary file or database access.
3) TCP/IP or other communication software supporting the network protocol.

9.07 Will the Requestor be able to send trades if the input socket goes down?

No, the application on the Requestor side will not be able to send any trade data. As such RTC will not receive any trade data until the input socket is re-established.

9.08 What if the output socket never comes back up that day?

RTC will not be able to send that data across via the IP link, but the Requestor can request a batch extract of their data and have it RJE transmitted. But the data that is on the TCP/IP database will be gone the next morning when the databases are refreshed.

9.09 What language is the interface written in?

The interface code is written in COBOL. Requestors can use any language they prefer as long as the IP protocols are used.

9.10 How can the Requestor verify that CBOE received everything the Requestor sent?

The Requestor will receive real-time confirmation of each individual message sent, if an application confirmation was requested. However, the Requestor can request a batch extract of their data and have it RJE transmitted, or just a count of all the trades taken. It could then run a compare of their real-time database against the downloaded CBOE image database.

9.11 What should the Requestor do with the confirmation messages CBOE sends?

The Requestor should process them against their trades' database, marking that activity as having been completed or in-error. The confirm message carries back a response code reflecting the quality of the information sent. Under most circumstances, CBOE will capture and commit your trade to the CTM database (soft edits). However, a Requestor's confirm processing program should investigate the response code for possible "hard" edit errors and realize that a hard edit means that your request did not reach the CTM database. In any case, the Requestor trades database should be updated to reflect the specific response received so that the Requestor system will reflect the most current status of the trade. Capturing this confirm message data also enables the Requestor's system to make decisions about any subsequent activity (e.g. corrections) against that trade.

9.12 Should Requestors keep an outbound copy of each message sent?

CBOE assumes that the Requestor's system has each trade "committed" to their own database before it is sent to CBOE. Therefore, any message that may not have reached CBOE can be re-constructed from the Requestor's own database.

9.13 Will CBOE process Requestor trade in sequence?

The real time interface is NOT a sequential system. In a sequential system, transactions are processed in a "single thread", with each record being dependant on the completion of the previous record. For performance reasons, this single threading is highly undesirable. In the RTC scenario, CBOE services Requestor requests as we get them. This means that changes or deletes will be applied, as they were send by the Requestor.

9.14 What if a Requestor continuously sends add transactions for different trades?

RTC will multi-task to process each 'add' and send back the confirmations as the trades are committed to our database. There are no inherent data integrity problems because each trade is different and each is an 'add.' There is no reason at all for RTC to "single thread", (sequentially process), these messages as no sequential dependence exists. Add trades represent over 99% of the messages the Requestor sends to CBOE, this alone is sufficient reason to multi-task transactions.

9.15 What if a Requestor continuously sends Change or Delete transactions for the same trade?

Under this scenario there may be a data-integrity problem in that subsequent updates against the same trades are dependent upon all the previous updates. Take the following transactions against trade number XYZ12345 as an example:

TRADE	ACTION	DATA SENT
XYZ12345	Add	A valid trade, quantity = 1000
XYZ12345	Change	Changing quantity to 100
XYZ12345	Change	Changing quantity to 10
XYZ12345	Change	Changing broker acronym
XYZ12345	Delete	Either contains the rest of the trade data or spaces

Messages are added to TCP/IP data repository based on the time the application received the message. Similarly, these messages are processed and validated in sequence the message was added to the repository. Therefore, for the above example the 'add' trade will be validated and added to the central data repository if all the critical fields are valid. Next the first 'change' will be processed to change the quantity to 100 from 1000. The 'change' to quantity of 10 will be next and the broker acronym 'change' will be third. The 'delete' will follow the last 'change' and this will delete the trade from the central data repository. As such, if the Requestor sent the same sequence of message again, the exact same procedure will follow.

9.16 Will CBOE provide a test environment for Requestors to test the RTC interface?

Yes. Test time on the CBOE host can be scheduled. Please see contact list in appendix.

9.17 Is there any assurance that messages traversing the line will not be corrupted by noise?

There is no assurance about the noise. These will be some noise in the TCP/IP line, and that is the reason the total number bytes sent to the Requestor is indicated at the beginning of every message sent by RTC.

9.18 What is the difference between the old LU trade data and the new IP trade data?

The following 6 fields have been expanded:

Field name	*Old size*	*New size*
Executing firm	3	5
Executing broker	3	4
Class	5	6
Premium price decimal	2	7
Contract quantity	5	7
Executing time	4	6

9.19 What is the purpose of the 'Heartbeat' message?

The 'heartbeat' message is only to verify the socket is still open for RTC to send or receive trade data. It is important to note this message will only be sent when there is no trade data to send and a pre-determined amount of time has elapsed. At that point RTC will send out a 'heartbeat' message and expect a response.

9.20 How fast is that traffic on this application?

RTC is able to send data to the requestor within a second of RTC getting the data from our internal applications, if the socket to the requestor is in a ready state.

9.21 What is meant by 'if the socket is ready state'?

When a RTCOUTPT socket is open by the requestor as outlined in section 6.03, the socket is in a ready state. It is ready to send data to the firm as soon as the data is received by the RTC application. If the socket is not in a ready state and data is received by RTC that data will not be able to be sent until the socket is returned to a ready state.

9.22 Can the number of trades sent on the RTCOUTPT socket be changed intra day?

Yes. For example if the socket is sending 5 trades at a time and the requestor would like to either increase or decrease the value, they would send in a requestor via email to CTM and we will change it. It is important to note here that when number of trade value is change, the requestor needs to close that particular socket and re-connect to get the new configuration take effect.

9.23 Can only one socket connection be used to open socket connection for RTCOUTPT and RTCINPUT?

No. The firm has to open two separate socket connections. One to establish a RTCOUTPT connection and another to establish a RTCINPUT connection.

Appendix

A. Trade Data Copybook.

NOTE - This copy book is being deprecated
For the purpose of this table, InB will stand for inbound messages and OtB will stand for outbound messages. Where these symbols are not used, the field content will apply to both inbound and outbound messages. The record format (fmt) indicator of 'A' means the field is alphanumeric and the indicator of 'N' means the field is numeric.

Field Name	Record Position	Size	Fmt	Description
Transaction code	1	1	A	InB: A =Add; C=Change; D=Delete. OtB: X=Add; Y=Change; Z=Delete.
Response code	2	1	A	InB: Blank. OtB: O=OK; S=Soft edits; H=Hard Edits.
Trade date	3-10	8	A	CCYYMMDD format.
Executing firm #	11-15	5	N	Zeroes in the first two bytes and Numeric executing firm value in the next three bytes.
Filler	16	1	A	Space.
Executing broker	17-20	4	A	Broker acronym in the first three bytes.
Filler	21	1	A	Space.
Transaction number	22-28	7	A	Transaction number assigned to the trade.
Transaction source	29	1	A	InB: L=Real-Time Comm OtB: O=ORS; R=RAES; C=Cobra; W=Wire; T=Terminal; B=Batch; H=Hand-held; S=SBT; L=RTC; Q=Hybrid
Transaction status	30	1	A	InB: Space OtB: M=Matched; U=Unmatched; B=Busted.
Buy Sell code	31	1	A	B or 1 = Buy; S or 2 = Sell.
Class code	32-37	6	A	Trading class.
Put/Call indicator	38	1	A	P=Put; C=Call.
Expiration date	39-46	8	A	CCYYMMDD format; E.g. 20020410. DD will contain true day value.
Exercise price	47-54	8	N	First 5 bytes are the integer potion, right justified and zero filled. The last 3 bytes are the decimal potion left justified and zero filled. E.g. $31.55 = 00031550 or $1.75 = 00001750.
Premium price	55-65	11	N	First 4 bytes are the integer potion, right justified and zero filled. The last 7 bytes are the decimal potion left justified and zero filled. E.g. $65.75 = 00657500000.
Filler	66-67	2	A	Space.
Market Maker broker code	68-70	3	A	Market maker broker acronym.
Filler	71	1	A	Space.
CMTA firm number	72-74	3	A	Firm number for the CMTA firm.
Filler	75	1	A	Space.
Origin indicator	76	1	A	Origin indicator.
Open/close indicator	77	1	A	Open or Close trade indicator.
Contra Firm	78-80	3	A	Contra (opposite) firm number.
Filler	81	1	A	Space.
Contra broker	82-84	3	A	Contra (opposite) broker acronym.
Filler	85	1	A	Space.
Contract quantity	86-92	7	N	Quantity right justified, zero filled. E.g. 0000125.
Executed trade time	93-98	6	N	HHMMSS format.
Firm optional data	99-114	16	A	Any data the firm wants to enter.
Halt match indicator	115	1	A	Halt matching. Value space or 'H'.

Field Name	Record Position	Size	Fmt	Description
Operator name	116-119	4	A	InB: Space or Nulls OtB: Operator who changed the trade.
Terminal name	120-123	4	A	InB: Space or Nulls. OtB: Terminal id from which the change was made.
Entry Firm code	124-126	3	N	Firm number to whom this trade belongs.
Filler	127	1	A	Space.
Premium price ind.	128	1	A	D= Decimal; F=Fractional. For TCP/IP this field must always have 'D."
Filler	129-130	2	A	Space.
Original Premium (decimal value)	131-132	2	A	Value of the premium numerator.
Hard edit flag	133	1	A	InB: Space. OtB: 1=Trade date; 2=Executing firm; 3=Executing broker; 4=Class code; 5=Duplicate trade; 6=Record not found; 7=Transaction code; 8=Transaction number; 9=Trade bust not allowed; A=Matching; B=Clearing; W=Unauthorized firm; Z=Undetermined dB error.
Soft edit executing broker	134	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit contra firm	135	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit contra broker	136	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit market maker broker	137	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit buy-sell Code	138	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit contra quantity	139	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit series	140	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit premium price	141	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit open close	142	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit origin indicator	143	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Message confirm	144	1	A	Value 'C'.
Message resend flag	145	1	A	Value 'R' or Space.
Message format	146	1	A	F=Full message or P=Partial message. Default 'F'. In later release CTM will develop variable length trade records.
Message ABS time	147-161	15	N	InB: Space. OtB: system time when trade was received by RTC.
Firm Affiliation	162-166	5	N	InB: Space. OtB: Trading firm affiliation number.
Branch Sequence	167-186	20	A	Consist of order date (8), branch (3), branch sequence number (4), correspondent firm (4), and blank (1).
Customer Id †	187-196	10	A	Customer Identification number.
Trade Type	197-197	1	A	Special Trades (see appendix C)
Bill Type	198-198	1	A	Maker/Taker indicator (see appendix C)
Filler	199-200	2	A	Space.

† For records sent inbound to RTC as a change the following will apply for this field:
Field has value – Field in trade record will be update with the value sent.
Field has blank – Field in trade record will not be update and original value retained.
Field has '*' - Field in trade record will be blank out.

B. CFLEX Trade Data Copybook.

NOTE - The Copy book in Appendix A above is being replaced with this copy book.
For the purpose of this table, InB will stand for inbound messages and OtB will stand for outbound messages. Where these symbols are not used, the field content will apply to both inbound and outbound messages. The record format (fmt) indicator of 'A' means the field is alphanumeric and the indicator of 'N' means the field is numeric.

Field Name	Record Position	Size	Fmt	Description
Transaction code	1	1	A	InB: A =Add; C=Change; D=Delete. OtB: X=Add; Y=Change; Z=Delete.
Response code	2	1	A	InB: Blank. OtB: O=OK; S=Soft edits; H=Hard Edits.
Trade date	3-10	8	A	CCYYMMDD format.
Executing firm #	11-15	5	N	Zeroes in the first two bytes and Numeric executing firm value in the next three bytes.
Filler	16	1	A	Space.
Executing broker	17-20	4	A	Broker acronym in the first three bytes.
Filler	21	1	A	Space.
Transaction number	22-28	7	A	Transaction number assigned to the trade.
Transaction source	29	1	A	InB: L=Real-Time Comm OtB: T=Terminal; B=Batch; S=SBT; L=RTC; Q=Hybrid; X=CFlex
Transaction status	30	1	A	InB: Space OtB: M=Matched; U=Unmatched; B=Busted.
Buy Sell code	31	1	A	B or 1 = Buy; S or 2 = Sell.
Class code	32-37	6	A	Trading class.
Put/Call indicator	38	1	A	P=Put; C=Call.
Expiration date	39-46	8	A	CCYYMMDD format; E.g. 20020410. DD will contain true day value.
Exercise price	47-55	9	N	First 5 bytes are the integer potion, right justified and zero filled. The last 4 bytes are the decimal potion left justified and zero filled. E.g. $31.5543 = 000315543 or $1.75 = 000017500.
Premium price	56-67	12	N	First 5 bytes are the integer potion, right justified and zero filled. The last 7 bytes are the decimal potion left justified and zero filled. E.g. $65.75345 = 000657534500.
Market Maker broker code	68-70	3	A	Market maker broker acronym.
Filler	71	1	A	Space.
CMTA firm number	72-74	3	A	Firm number for the CMTA firm.
Filler	75	1	A	Space.
Origin indicator	76	1	A	Origin indicator.
Open/close indicator	77	1	A	Open or Close trade indicator.
Contra Firm	78-80	3	A	Contra (opposite) firm number.
Filler	81	1	A	Space.
Contra broker	82-84	3	A	Contra (opposite) broker acronym.
Filler	85	1	A	Space.
Contract quantity	86-92	7	N	Quantity right justified, zero filled. E.g. 0000125.
Executed trade time	93-98	6	N	HHMMSS format.
Firm optional data	99-114	16	A	Any data the firm wants to enter.
Halt match indicator	115	1	A	Halt matching. Value space or 'H'.
Operator name	116-119	4	A	InB: Space or Nulls OtB: Operator who changed the trade.
Terminal name	120-123	4	A	InB: Space or Nulls. OtB: Terminal id from which the change was made.
Entry Firm code	124-126	3	N	Firm number to whom this trade belongs.

Field Name	Record Position	Size	Fmt	Description
Filler	127	1	A	Space.
Premium price ind.	128	1	A	D = Decimal
Original Premium (decimal value)	129-132	4	A	Value of the premium numerator.
Hard edit flag	133	1	A	InB: Space. OtB: 1=Trade date; 2=Executing firm; 3=Executing broker; 4=Class code; 5=Duplicate trade; 6=Record not found; 7=Transaction code; 8=Transaction number; 9=Trade bust not allowed; A=Matching; B=Clearing; W=Unauthorized firm; Z=Undetermined dB error.
Soft edit executing broker	134	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit contra firm	135	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit contra broker	136	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit market maker broker	137	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit buy-sell Code	138	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit contra quantity	139	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit series	140	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit premium price	141	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit open close	142	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Soft edit origin indicator	143	1	A	InB: Space. OtB: Value '*' Indicate field is in error.
Message confirm	144	1	A	Value 'C'.
Message resend flag	145	1	A	Value 'R' or Space.
Message format	146	1	A	F=Full message or P=Partial message. Default 'F'. In later release CTM will develop variable length trade records.
Message ABS time	147-161	15	N	InB: Space. OtB: system time when trade was received by RTC.
Firm Affiliation	162-166	5	N	InB: Space. OtB: Trading firm affiliation number.
Branch Sequence	167-186	20	A	Consist of order date (8), branch (3), branch sequence number (4), correspondent firm (4), and blank (1).
Customer Id †	187-196	10	A	Customer Identification number.
Trade Type	197-197	1	A	Special Trades (see appendix C)
Bill Type	198-198	1	A	Maker/Taker indicator (see appendix C)
Filler	199-200	2	A	Space.

† For records sent inbound to RTC as a change the following will apply for this field:
Field has value – Field in trade record will be update with the value sent.
Field has blank – Field in trade record will not be update and original value retained.
Field has '*' - Field in trade record will be blank out.

C. Field definitions

Trade Type

B - Block Trade
C - Cash Trade
E - Exchange For Physical
F - Intermarket Sweep
G - GWAP (Gamma Weighted Average Price) Trade
H - Handheld Trade
I - Cross Product AIM Cross Trade
L - No Print Linkage Trade
M - Manual Trade
N - Next Day trade
P - Par trade
Q - Par to Market Maker Trade
R - Regular Trade
S - Cross product Cross Trade
T - Two Day Trade
V - The Atomic trade (SPX) for VXS product
X - Cross Product Leg Trade
Z - Sold
1 - Regular Trade Reversal
2 - No Print Linkage Trade Reversal
3 - No Print Linkage Trade Manual

Bill Type

A - Maker
B - Odd Lot Response
C - Cross
E - Flash Response
F - Flash
L - Linked Away Response
N - Odd Lot Flash
O - Opening Trade
Q - Resting
R - Taker
S - Cross Price Imp
T - Flash Price Imp
U - Flash Response Price Imp
V - Maker Turner
W - Resting Turner
X - Linked Away

D. Contact List

Software/Technical support:

Indravadan Merai	(312) 786-8027	merai@cboe.com
Uday Aravapalli	(312) 786-7634	aravapal@cboe.com
Chris Hou	(312) 786-7216	hou@cboe.com

Communication:

Dave Gonzales	(312) 786-8060	gonzales@cboe.com

TCP/IP Network:

Requestor's are instructed to send an email to RTC@cboe.com outlining their intent.

Glossary

CTM	-	Continuous Trade Match or Trade Match; the CICS Trade Match system is known as ITP.
CFLEX	-	Acronym for CBOE Flex Trade. FLEX options are acronym for FLexible EXchange Traded option.
Destination	-	A four-letter acronym assigned to a Requestor by Trade Match that is integral part of the associating service type to that Requestor.
Echo	-	A message sent from CBOE to Requestor, which the Requestor receive and sends back to CBOE without altering it in any way; also called a 'heartbeat.'
Hard-Edit	-	Errors found on trade record that are considered fatal to continue processing the trade. Error flags marked on trade record. A hard-edit is a definition given to an error found in the trade data that the trade could not be processed and passed on to CTM.
MMHH	-	Market Maker Hand Held system.
OCC	-	Option Clearing Corporation.
RAES	-	Retail Automatic Execution System.
Requestor	-	Any entity requesting RTC TCP/IP service.
RJE	-	Remote Job Entry. Processes by which the Requestor can submit a JCL job at CBOE to execute a process.
RTC	-	Real Time Communication.
RTC Trades	-	Messages that the Requestor sends to CBOE, in other words inbound messages.
RTC Drops	-	Message that CBOE sends to the Requestor, in other words outbound messages.
SBT	-	Screen Based Trading also know as CBOEDirect.
Soft-Edit	-	Errors found on trade record that are considered minor but the trade is processed and error flags marked on trade record. A soft-edit is a definition given to an error found in the trade data that is not deemed fatal and the trade is processed by CTM.
TERMINAL	-	Trades entered into Trade Match via the ITP screens.
TCP/IP	-	Transmission Control Protocol (TCP) and Internal Protocol (IP)
TP25	-	Is the process of a member firm transmitting its side of a filled buy/sell trade directly to Trade Match.
WIRE	-	Better known as TP25.