TMX IP

Consolidated Depth of Book (CDB)

Functional Specifications
Version 1.2

November 13, 2013

NOTICE & DISCLAIMER

The information contained in this document and any other information or documentation related to the CDB specification including all past, present and future versions thereof and associated message formats (collectively referred to as the "CDB Specification") are provided “as is”, and neither TSX nor any of its affiliates makes any representation or warranty, express or implied, as to the contents of the CDB Specification (or the results obtained by using it) and each such person specifically disclaims any representation, warranty or condition of accuracy, completeness, merchantability, fitness for a particular purpose or that it is error-free. By using the CDB Specification, you are agreeing to assume the entire risk associated with its use.

TSX and its affiliates shall have no liability for damages of any kind arising in any manner out of or in connection with your use of, or your inability to use, the CDB Specification, whether direct, indirect, incidental, special or consequential (including, without limitation, loss of data, loss of use, third party claims, lost profits, lost revenues and strict liability), arising under contract or otherwise, whether or not TSX has been advised of, or otherwise might have anticipated the possibility of, such damages.

The information contained in the CDB Specification is proprietary and confidential information belonging to TSX and its licensors or service providers, as applicable. Copyright and trade-mark rights and any other intellectual property rights in the CDB Specification belong to TSX (or its licensors or service providers, as applicable). Your permitted use of the CDB Specification, in whole or in part, is limited to the non-exclusive, non-transferable, revocable, non-assignable, personal right for you only to build a connection between your systems and TSX’s market information system. Any uses not specifically set out herein are prohibited. TSX may revoke this right at any time, on notice to you, and you agree to promptly comply with such notice. Do not reproduce, disclose, distribute, publish, sell, modify**,** commercialize or make any part of the CDB Specification available, in whole or in part, to any person including to any of your affiliates, service providers or any other related party without TSX’s prior written consent. If you do not agree with, or do not wish to comply with, any of the above, please notify TSX and, in accordance with TSX's instructions, either return or destroy any copies or version of the CDB Specification in your possession.

This Notice, Disclaimer and Rights Granted sets out the terms and conditions on which you are permitted to use the CDB Specification. This Notice, Disclaimer and Rights Granted may be changed from time to time by TSX in its discretion. It is your obligation to check and comply at all times with the terms applicable to the most current version of the CDB Specification.

TABLE OF CONTENTS

1.0 Overview

This document describes the Consolidated Depth of Book (CDB) data feed provided by the TMX IP. The CDB is a multi-market single consolidated view of the orderbook. The CDB orderbook will distribute, by symbol, the aggregate orders and volume at each price for each marketplace.

The CDB business content messages are formatted using the STAMP protocol syntax. STAMP, the Securities Trading Access Message Protocol, is the messaging protocol developed by TSX for order entry. More details about the STAMP protocol are given in the *STAMP Specification Version 5.3* (Reference [1]).

1.1. Intended Audience

The intended audience of this specification are business analysts and programmer analysts.

All readers should familiarize themselves with *Section 1.2* – Rule Notation Conventions, paying close attention to how the notation conventions are defined, as this notation is used throughout the specification.

Business analysts should focus primarily on Section 4, Business Content Messages and Section 6, Field Definitions. These two sections define how the trading information is defined in the CDB. In addition to these sections, the business analysts should be familiar with the trading rules and trading scenarios that these messages represent.

Programmer analysts should be familiar with the entire specification, although their focus should be on message structure and parsing.

1.2. Rule Notation Conventions

This section describes the notation convention for the elements of STAMP syntax used in the business content. Although the rules presented below are somewhat formal in nature, for casual reading of the specification all that is required is to keep in mind the following points:

- Text presented in a `typewriter typeface` font means that it is a rule that is defined in the Field Definitions Section 6.
- Any rule that is enclosed in square brackets, "`[`" and "`]`" means that the rule is optional.
- The spaces between the rules mean that the rules are joined together.

When appropriate, this specification uses an augmented Backus-Naur Form (BNF) notation, similar to that presented in *RFC 822 – Standard For The Format of ARPA Internet Text Messages* (Reference [8]). The differences from standard BNF involve naming rules and indicating repetition and "local" alternatives. Comments about a rule, such as the hexadecimal representation of a character, are introduced by a semicolon (";") in-line after the rule definition. All text after a semicolon until the end of a line forms the comment.

Rules are used throughout the text of the specification when appropriate to formally define a concept. All of the rules are gathered in the Field Definitions Section for convenience.

1.2.1. Rule Naming

Angle brackets ("<", ">") are used below in the syntax definition of rules to identify rule components; these brackets are not used, in general, in the rule names. The name of a rule is simply the name

itself rather than "<name>". Capitalized letters are used in names to highlight the meaning of the name.

1.2.2. Literal Text

Quotation marks enclose literal text (which is case sensitive). Literal text appears as is in the message content.

1.2.3. Alternatives: Rule1 | Rule2

Elements separated by vertical line ("|") are alternatives. Therefore, "[abc | def]" will accept abc or def.

1.2.4. Local Alternatives: (Rule1 | Rule2)

Elements enclosed in parentheses are treated as a single element. Thus, "(elem (abc | def) elem)" allows the token sequences "elem abc elem" and "elem def elem".

1.2.5. Repetition: *Rule

The character "*" preceding an element indicates repetition. The full form is:

```
<l>*<m>element
```

indicating at least <l> and at most <m> occurrences of element, with default values of 0 and infinity respectively.

So that "*(element)" allows any number, *including zero*; "1*element" requires at least one; and "1*2element" allows one or two.

If the repeated element is a FieldIdentifier, the repeated element will be represented in the datastream using the FieldIdentifierIndex notation as described in *Section 2 of the STAMP Specification Version 5.3.*

1.2.6. Optional: [Rule]

Square brackets enclose optional elements; eg., "[abc def]" is equivalent to "1*1(abc def)". The square bracket notation is used in the message description.

1.2.7. Specific Repetition: Nrule

"<n>(element)" is equivalent to "<n>*<n>(element)"; that is, exactly <n> occurrences of (element). Thus 2Digit is a 2-digit number, and 3AlphaNumeric is a string of three alphabetic characters. If the repeated element is a STAMP FieldIdentifier, the repeated element will be represented in the datastream using the FieldIdentifierIndex notation as described in *Section 2 of the STAMP Specification Version 5.3.*

1.2.8. Client/Server Notation Convention

For the purpose of this specification, "Client" (initial capital letter) refers to the computer application that "listens" for output messages from the CDB service.

2. Service Architecture

The CDB service adheres to TSX service architecture for market data dissemination defined in reference [3].

2.1. Framing

CDB market data message uses the following basic structure:

STX Transport Header Message ETX

where, STX is the Start of Text (Hexadecimal 0x02), and ETX is the End of Text (Hexadecimal 0x03), “Message” is the business content that is described in Section 3 and 4.

2.2. Transport Header

The “Transport Header” is a 22-byte section coded in ASCII and structured as follows:

Field	Length	Contents / Values
Length	4	Total length of header and message business content (excludes STX and ETX), padded with zeros to the left.
Sequence Number	9	Sequence number assigned at service broadcast, padded with zeros to the left. Blank on Heartbeat messages.
ServiceID	3	“BK1” Code identifying the service CDB
Retransmission Identifier	1	0 – Normal transmission 1 – Message being sent out of order from their generation by the trading system. This can be due to unusual processing causing delay or recovery from a problem or link failure.
Continuation Indicator	1	0 – This is stand alone packet (the message fits in one packet) 1 – This packet continues in the next packet (the message spans at least 2 packets). 2 – This packet is the continuation of the previous packet. 3 – This packet is both the continuation of the previous packet and continues in the next packet.
Message Type	2	“V “ for Heartbeat message (padded with a blank to the right). Left blank for all other message types.
Exchange Identifier	2	Code assigned to the originating exchange (padded with a blank to the right) as follows: “B” for TMX IP Consolidated Depth of Book

Every message packet is assigned a sequence number from 000000001 to 999999999 (decimal ASCII), with wrap-around. The sequence is reset to 1 each day and it is incremented by 1 for each packet sent.

2.3. Heartbeat Message

The Heartbeat message is sent every 60 seconds and is unsequenced. The Heartbeat message provides three information sections regarding real time message delivery, delimited by brackets:

- ◊ HEARTBEAT section, including date and time and decimals seconds since 1970 up to the microsecond,
- ◊ LAST SENT section, including sequence number of last message sent, time sent, and decimal seconds up to the microsecond,
- ◊ LAST HB section, including the "last sent" information passed in the last heartbeat message sent.

The information provided in the Heartbeat message allows clients to track real time delivery latencies.

The Heartbeat message is a fixed field length message with the following format:

Field	Length	Value /Definition	Description / Format
	1	"["	Separator
	10	"HEARTBEAT"	Section identifier
Date	10		Date in format YYYY-MM-DD
	1	blank	Separator string
TimeOfDay	8		Time of day in format HH:MM:SS
	1	"-"	Separator string
SecondsSince1970	19	6 decimals with embedded decimal point	Formatted with "%012d.%06d" in C language
	2	"][	Separator
	10	"LAST SENT "	Section identifier
SeqNbrOfLastMsgSent	9		Last sequence number sent, padded with 0s to the left
	1	"-"	Separator
TimeLastMsgSent	8		Time last message sent in format HH:MM:SS
	1	"-"	Separator
SecondsSince1970LastMsg	19	6 decimals with embedded decimal point	Formatted with "%012d.%06d" in C language
	2	"]["	Separator
	10	"LAST HB "	Section identifier – Last Heartbeat data, right-padded with blanks.
SeqNbrOfLastMsgSent	9		This number lets the client know if they missed a heartbeat
	1	"-"	Separator
TimeLastMsgSent	8		Time last message sent in format HH:MM:SS in last heartbeat
	1	"-"	Separator
SecondsSince1970LastMsg	19	6 decimals with embedded decimal point	Formatted with "%012d.%06d" in C language in last heartbeat
	1	"]"	Separator
OCSAsubject	20		TSX diagnostics
OCSAinstance	2		TSX diagnostics
Hostname	8		ID of the originating host.
Version	4		Version of the service being delivered

The following is an example of a heartbeat message:

\020208 BK100V T [HEARTBEAT 2001-10-16 01:30:24-001003210224.494190][LAST SENT 000507240-17:00:01-001003179601.128993][LAST HB 000507240-01:29:24-001003210164.494014]OCSA-BK1-1 Atibprd0p0.0 \03

The \02 and \03 strings represent respectively the STX and ETX characters framing the message.

2.4. Message Retransmission

CDB will provide support for automated retransmissions as defined in reference [3].

3. Message Structure

Business content in CDB messages is coded in STAMP format. This portion of the message is formally described as follows:

```
MessageContent          =  ControlHeader BusinessContent [ControlTrailer]
ControlHeader           =  ControlHeaderChar ControlHeaderContent
ControlHeaderContent    =  1*ControlHeaderField
ControlHeaderChar       =  <US-ASCII SOH> ; 0x01  Start of Heading
BusinessContent         =  BusinessContentChar 1*BusinessContentField
BusinessContentChar     =  <US-ASCII FS> ; 0x1c  File Separator
ControlTrailer          =  ControlTrailerChar
ControlTrailerChar      =  <US-ASCII GS> ; 0x1d   Group Separator
```

3.1. Control Header Content

```
ControlHeaderContent    =   DestAddress SequenceNumber [TimeStamp]
                            CdfRcvTimeStamp CdfPubTimeStamp
                            [SourceAddress] [LasTSXquenceReceived]
                            [Retrans] [RetransId] [CdfId]
                            [ConsInboundTimeStamp]
                            [ConsOutboundTimeStamp]
```

The CDB service includes the STAMP control layer header and trailer. The STAMP control header is described in detail in reference [1]. [The only STAMP header field that provides useful information in the context of service is `TimeStamp.`]

The `ControlHeaderChar` (0x01), `BusinessContentChar` (0x1c), and `ControlTrailerChar` (0x1d), separators are not explicitly mentioned in Section 4, Business Content Messages.

3.2. Fields

Both the Control and Business Content sections are further divided into fields. Each field consists of a field identifier and a field value. The identifiers and values are variable in length and content; consult Section 6, “Field Definitions” for appropriate qualifying rules.

A field is divided into two sections: a field identifier and an optional field value. The FieldIdentifier is introduced by a FieldIdentifierChar. The optional FieldValue is introduced by the US-ASCII equals sign, =.

It is possible to have a FieldIdentifier without a FieldValue, in which case the FieldValue assumes a default value. For more information, see Section 6, “Field Definitions”.
The formal notation for a field is:

```
Field                       =      FieldChar FieldIdentifier "=" [FieldValue]
FieldChar                   =      <US-ASCII RS; Record Separator> ; 0x1e
```

3.2.1. Field ordering

The order of the fields within a section of a STAMP message is position independent. The fields must be of the correct type (for example, the fields within the ControlHeader must be of the type ControlHeader-Field) and may be in any order within the section.

3.2.2. Field identifier

In Section 6, “Field Definitions”, the field identifier is a number that is used as an index to identify the syntactic meaning of the field value. As an example, if the field identifier of a field is 55, the field value is a stock symbol.

For repeating groups of field identifiers, a “dot” notation is used:

- If a message contains multiple occurrences of a field identifier, each occurrence is represented by an additional field identifier index.
- If there are linked groups of fields, the index is used to link the elements syntactically
- Field indexes must start at zero and be contiguous.
- A field identifier without an explicit index is equivalent to an index of zero.

Fields at the same index level are conceptually “records”. The contiguous nature of the index refers to the conceptual record, not individual field identifiers. For example, a STAMP message with these tags, “`11.0=ABC”,”11.1=DEF”,”15.1=5`”, is valid and represents a situation where tag 15 is optional and not present for the “`0`” record. However, there would be at least one field at each index level.

3.2.3. Field value

The field value contains the value of the field. To use a previous example, if the identifier is `55` and the value is `BCE`, then the stock symbol for this message is `BCE`.

The formal notation for a field value is:

```
FieldValue          =     1*PrintableChar
```

4. Business Content Messages

The messages described in this section are the consolidated orderbook messages that are broadcast from the TMX IP to the Client.

4.1. CDBSymbol Message

At the start of day, the CDBSymbol message will be sent out for each of the valid CDB symbols for the current trading day.

CDBSymbolMessage	=	ControlHeader BusinessContent
ControlHeaderContent	=	CdfPubTimeStamp CdfRcvTimeStamp DestAddress SequenceNumber SourceAddress TimeStamp [CdfId] [ConsInboundTimeStamp] [ConsOutboundTimeStamp] [LasTSXquenceReceived] [Retrans] [RetransId]
BusinessContent	=	BusinessClass [BoardLot] [Currency] [CUSIP] [FaceValue] [ProductType] Symbol [SymbolFullName]
BusinessClass	=	"CDBSymbol"

NOTE:

CDBSymbol messages exceeding 1,400 characters in length will be delivered on multiple packets using the header Continuation Indicator described in Section 2.2. Please see also reference [3]. All packets corresponding to a single logical CDBSymbol message must be reassembled by the receiver before attempting to parse the STAMP tags inside the message.

4.2. CDBOrderbook Message

CDBOrderbook message provides the symbol, price, aggregated volume and number of orders by marketplace for all open orders.

The CDBOrderbook message sent at the start of day is the information as of the end of the previous business day. Its purpose is to enable the initialization of the Client's consolidated orderbook for the current trading session.

- The initial CDBOrderbook message is sent at 05:00 ET each trading day.

Under certain circumstances, such as a recovery of a dropped message, the CDB will push out a CDBOrderbook message intraday to provide an updated CDB Orderbook for a specific security. Clients should replace their existing CDB orderbook with the new CDBOrderbook.

CDBOrderBookMessage	=	ControlHeader BusinessContent
ControlHeaderContent	=	CdfPubTimeStamp CdfRcvTimeStamp DestAddress SequenceNumber SourceAddress TimeStamp [CdfId] [ConsInboundTimeStamp] [ConsOutboundTimeStamp] [LasTSXquenceReceived] [Retrans] [RetransId]
BusinessContent	=	BusinessClass ExchangeId MarketSide Price Symbol Volume
BusinessClass	=	"CDBOrderbook"

NOTE:

CDBOrderbook messages exceeding 1,400 characters in length will be delivered on multiple packets using the header Continuation Indicator described in Section 2.2. Please see also reference [3]. All packets corresponding to a single logical CDBOrderbook message must be reassembled by the receiver before attempting to parse the STAMP tags inside the message.

Values for `BusinessClass` and `Symbol` will only be present in the first packet and will not be repeated in the remaining packets of an CDFOrderbook message exceeding 1,400 characters in length and delivered in multiple packets.

For repeating groups of field identifiers, a "dot" notation is used:

- If a message contains multiple occurrences of a field identifier, each occurrence is represented by an additional field identifier index.
- If there are linked groups of fields, the index is used to link the elements syntactically
- Field indexes must start at zero and be contiguous.
- A field identifier without an explicit index is equivalent to an index of zero.

Fields at the same index level are conceptually "records". The contiguous nature of the index refers to the conceptual record, not individual field identifiers. For example, a STAMP message with these tags, "`11.0=ABC","11.1=DEF","15.1=5`", is valid and represents a situation where tag 15 is optional and not present for the "`0`" record. However, there would be at least one field at each index level.

4.3. CDBUpdate Message

The `CDBUpdate` message is sent when an update occurs to the consolidated orderbook as a result of an action (new order, order cancellation, order execution/trade) sent to the CDB from a contributing marketplace. The message includes the aggregated volume, price, symbol, and marketplace identifier.

CDBUpdate message	=	ControlHeader BusinessContent
ControlHeaderContent	=	CdfPubTimeStamp CdfRcvTimeStamp DestAddress SequenceNumber SourceAddress TimeStamp [Retrans] [RetransId] [CdfId] [ConsInboundTimeStamp] [ConsOutboundTimeStamp]
BusinessContent	=	BusinessClass ExchangeId MarketSide Price Symbol Volume
BusinessClass	=	"CDBUpdate"

NOTE:

When a CDBUpdate message is received the client should update the orderbook with new entry or re-place the volume of an existing entry (ExchangeId, Marketside, Price and Volume exist in the orderbook) with the volume in the CDBUpdate Message. If volume in the CDBUpdateMessage is "0" the entry should be removed from the book.

4.4. StockStatus Notification

A stock status notification is sent by the marketplace in response to a change in stock status on the marketplace.

StockStatusMessage	=	ControlHeader BusinessContent
ControlHeaderContent	=	CdfPubTimeStamp CdfRcvTimeStamp DestAddress SequenceNumber SourceAddress TimeStamp [CdfId] [ConsInboundTimeStamp] [ConsOutboundTimeStamp] [LasTSXquenceReceived] [Retrans] [RetransId]
BusinessContent	=	BusinessClass ExchangeId StockState Symbol TradingSysTimeStamp
BusinessClass	=	“StockStatus”

5. Operating Sequence

5.1. Transmission Times

(1) Clients can listen on the CDB port at any time during the day. The unsequenced Heartbeat message is transmitted every 60 seconds.

(2) Re-transmission requests can be sent from 5:00AM to 22:00PM.

(3) Transmission times for CDB are Eastern Standard/Daylight Savings Time.

5.2. Trading Hours for the participating Marketplaces

Exchange	Regular Trading Session (EST)			Extended Trading Session (EST)		
	Pre-Open	Open	Close	Pre-Open	Open	Close
Alpha Group	07 :00	09 :30	16 :00	N/A	16 :15	17 :00
Chi-X Canada	07 :00	08 :30	17 :00	N/A	N/A	N/A
CX2	07 :00	08 :30	17 :00	N/A	N/A	N/A
CNSX	07 :00	09 :30	16 :00	N/A	N/A	N/A
Lynx ATS	N /A	08 :30	17 :00	N/A	N/A	N/A
Omega ATS	N /A	08 :30	17 :00	N/A	N/A	N/A
Pure Trading	07 :00	08 :00	17 :00	N/A	N/A	N/A
TMX Select	N/A	8 :00	17 :00	N/A	N/A	N/A
TSX	07 :00	09 :30	16 :00	N/A	16 :15	17 :00
TSX Venture Exchange	07 :00	09 :30	16 :00	N/A	16 :15	17 :00

6. Field Definitions

A

AlphaNumeric – alphabetic and numeric characters.

```
AlphaNumeric = all US-ASCII character, 0x00 to 0x7f
```

B

BoardLot – boardlot volume.

```
FieldIdentifier = 115
BoardLot        = Volume  ;  no default
```

BusinessClass – the message class for a Business Content Layer message.

```
FieldIdentifier = 6  ; no default
BusinessClass   = "CDBSymbol"
                  "CDBOrderBook"
                  "CDBUpdate"
                  "StockStatus"
```

Maximum 35 Characters

BusinessContent – the business fields for a STAMP message.

```
BusinessContent = BusinessContentChar 1*BusinessContentField
```

BusinessContentChar – the character that introduces BusinessContent.

```
BusinessContentChar = <US-ASCII FS; File Separator>  ; 0x1c
```

BusinessContentField – a field found in the Business Content section of a message.

C

CdfId – Unique internal identifier which includes an internal sequence number assigned by the system to each CDB message for tracking and audit

```
FieldIdentifier = 513         ; no default
CdfId           = 1*31 AlphaNumeric
```

ConsInboundTimeStamp – Unique internal inbound CDB consolidation timestamp assigned by the system to each CDF message for tracking and audit

```
FieldIdentifier      = 515     ; no default
ConsInboundTimeStamp = 17Digit ; YYYYMMDDHHMMSSmmm (year, month, day,
                               ; hour, minute, second, millisecond)
```

ConsOutboundTimeStamp – Unique internal outbound CD**B** consolidation timestamp assigned by the system to each CDB message for tracking and audit

```
FieldIdentifier       = 514      ; no default
ConsOutboundTimeStamp = 17Digit  ; YYYYMMDDHHMMSSmmm (year, month, day,
                                 ; hour, minute, second, millisecond)
```

CdfPubTimeStamp – the time at which the CDB message was sent.

```
FieldIdentifier = 501         ; no default
CDFPubTimeStamp = 17Digit     ; YYYYMMDDHHMMSSmmm (year, month, day,
                              ; hour, minute, second, millisecond)
```

CdfRcvTimeStamp – the time at which the CDB message was received.

```
FieldIdentifier = 502          ; no default
CdfRcvTimeStamp = 17Digit      ; YYYYMMDDHHMMSSmmm (year, month, day,
                               ; hour, minute, second, millisecond)
```

ControlHeader – the portion of the STAMP message that contains administrative information.

```
ControlHeader = ControlHeaderChar 1*ControlHeaderField
```

ControlHeaderChar – the character that introduces ControlHeader.

```
ControlHeaderChar = <US-ASCII SOH; Start of Heading> ; 0x01
```

ControlHeaderField – a field found in the ControlHeader section of a message.

Currency – the currency of the price.

```
FieldIdentifier = 58
Currency        = "$CAD"    | ; default
                  "$USD"
```

CUSIP - clearing and settlement registration number.

```
FieldIdentifier = 171
CUSIP           = 9*12AlphaNumeric  ;  no default
```

D

Date – the date format.

```
Date = 8Digit  ; in YYYYMMDD format
```

DestAddress – the destination STAMP address.

```
FieldIdentifier = 17
DestAddress     = DirectedAddress | BroadcastAddress  ; no default
```

Note that only servers are allowed to use `BroadcastAddress`.

Digit – representation of numeric values.

```
Digit = “0” | “1” | “2” | “3” | “4” | “5” | “6” | “7” | “8” | “9”
```

E

Empty – nothing.

```
Empty = “”
```

ExchangeId –identifies the exchange from which the message originated.

```
FieldIdentifier = 247
 ExchangeId      = "AIS"- Alpha IntraSpread
                   "ALP"- Alpha
                   "CDX"- TSXVenture
                   "CHI"- Chi-Xcanada
                   "CHT"- CX2
                   "CNQ"- CNSX
                   "ICX"- InstinetCanadaCross
                   "LIQ"- LiquidnetCanada
                   "LYX" - Lynx ATS
                   "OMG"- OmegaATS
                   "PUR"- PURE
                   "SEL"- TMX Select
                   "TCM"- TriActMatchNow
                   "TSE"- TSX
```

F

FaceValue – the face value of a debenture.

```
FieldIdentifier = 119
FaceValue       = Price  ;  no default
```

Field – a unit within a section that includes a FieldIdentifier and an optional FieldValue.

```
Field = FieldChar FieldIdentifier "=" [FieldValue]
```

FieldChar – the character that introduces a field.

```
FieldChar = <US-ASCII RS; Record Separator>  ; 0x1e
```

FieldIdentifier – the value that identifies what the field means.

```
FieldIdentifier = 1*4Digit [FieldIdentifierIndex] ; 1 to 9999, no default
```

FieldIdentifierIndex – an instance of a specific field within a message.

```
FieldIdentifierIndex  = "." 1*4Digit  ; 0 to 9999, default is 0
```

FieldValue – the value of the field.

```
FieldValue = 1*PrintableChar
```

G

H

Hexadecimal – hexadecimal number representation.

```
Hexadecimal = Digit | "a" | "b" | "c" | "d" | "e" | "f"
```

I

J

L

LasTSXquenceReceived – the last sequence number received.

```
FieldIdentifier    = 15
LasTSXquenceNumber = SequenceNumber  ; no default
```

M

MarketSide – the buy or sell side of the market.

```
FieldIdentifier = 197
```
MarketSide = “Buy” | “Sell” ; no default

N

NumericPrice – a price in a currency.

```
NumericPrice = 1*6Digit [“.” 1*4Digit]
```

O

P

Price – the price for the consolidated orders.

```
FieldIdentifier = 41
Price           = NumericPrice
```

PrintableASCII – characters that have a glyph from the US-ASCII character set.

```
PrintableASCII = <any printable char from US-ASCII char set plus HT>
                 ; 0x09, 0x20 to 0x3c, 0x3e to 0x7e
```

PrintableChar – characters that have a glyph.

```
PrintableChar = PrintableASCII | PrintableLatin1
```

ProductType – the product type for a symbol.

```
FieldIdentifier = 105
ProductType     = “TSX VENTURE-Equity”     |  ; TSX VENTURE default
                  “TSX VENTURE-Debenture   |  ;
                  “Debenture”              |  ;
                  “Equity”                 |  ; TSX default
```

R

Retrans – a marker that indicates the message is a retransmitted message.

```
FieldIdentifier = 97
Retrans         = “Y”| “N”  ; default is “N”
```

RetransId – an identifier as to which retransmission request caused the retransmission.

```
FieldIdentifier = 147
RetransId       = 1*5AlphaNumeric  ; no default
```

S

SequenceNumber – the sequence number of the message.

```
FieldIdentifier = 50
SequenceNumber  = 1*9Digit  ; 0 to 999,999,999  ; no default
```

SourceAddress – the source STAMP address.

```
FieldIdentifier = 54
SourceAddress = DirectedAddress  ; no default
```

StockState – the possible states that a stock may be in that are broadcast.

```
FieldIdentifier    = 161
StockState         = "Authorized"
                     "AuthorizedDelayed"
                     "AuthorizedFrozen"
                     "AuthorizedHalted"
                     "Inhibited"
                     "InhibitedDelayed"
                     "InhibitedFrozen"
                     "InhibitedHalted"
                     "AuthorizedPriceMovementDelayed"
                     "InhibitedPriceMovementDelay"
                     "InhibitedPriceMovementFrozen"
```

Symbol – the security/issue symbol.

```
FieldIdentifier = 55
Symbol          = 1*17AlphaNumeric  ; no default
```

SymbolFullName – the security/issue symbol's complete company name.

```
FieldIdentifier = 177
SymbolFullName  = 1*80PrintableASCII  ;  no default
```

T

TimeStamp – the time at which the message was sent from the source marketplace

```
FieldIdentifier = 56          ; no default
TimeStamp       = 16Digit     ; YYYYMMDDHHMMSShh (year, month, day, hour
                              ; minute, second, hundredths of a second)
```

Note that for a retransmitted message, the value of TimeStamp is the time of the retransmission, not the transmission time of the original message.

TradingSysTimeStamp – the time at which the action occurred on the originating marketplace.

```
FieldIdentifier     = 57
TradingSysTimeStamp = TimeStamp  ;  no default
```

U

V

VersionNumber – the version number of the STAMP protocol specification used.

```
FieldIdentifier = 65
VersionNumber   = "Version 5.5"
```

Volume – the quantity of shares for an order.

```
FieldIdentifier = 64
Volume          = 1*9Digit  ; no default
```

W

7. Field Defintion by Numerical Order

6	BusinessClass
15	LasTSXquenceReceived
17	DestAddress
41	Price
50	SequenceNumber
54	SourceAddress
55	Symbol
56	TimeStamp
57	TradingSysTimeStamp
58	Currency
64	Volume
65	VersionNumber
97	Retrans
105	ProductType
119	FaceValue
147	RetransId
161	StockState
171	Cusip
177	SymboFullName
197	MarketSide
247	ExchangeId
501	CdfPubTimeStamp
502	CdfRcvTimeStamp
513	CdfId
514	ConsOutboundTimeStamp
515	ConsInboundTimeStamp

8. Field Defintion by Alphabetical Order

6	BusinessClass
513	CdfId
515	ConsInboundTimeStamp
514	ConsOutboundTimeStamp
501	CdfPubTimeStamp
502	CdfRcvTimeStamp
58	Currency
171	Cusip
17	DestAddress
247	ExchangeId
119	FaceValue
15	LasTSXquenceReceived
197	MarketSide
41	Price
105	ProductType
97	Retrans
147	RetransId
50	SequenceNumber
54	SourceAddress
161	StockState
55	Symbol
177	SymboFullName
56	TimeStamp
57	TradingSysTimeStamp
65	VersionNumber
64	Volume

9. References

[1] STAMP Specification, Version 5.5, TSX, April 2001

[2] Toronto Broadcast Feed Specification, Version 3.0, TSX, April 2001

[3] CDF, *Protocol Specifications and Service Access*, TSX, 2008

[4] RFC 768, *User Datagram Protocol*, J. Postel, September 1981, http://www.internet.nic

[5] RFC 791, *Internet Protocol*, J. Postel, September 1981, http://www.internet.nic

[6] RFC 792, *Internet Control Message Protocol*, J. Postel, September 1981, http://www.internet.nic

[7] RFC 793, *Transmission Control Protocol*, J. Postel, September 1981, http://www.internet.nic

[8] RFC 822, *Standard For The Format of ARPA Internet Text Messages*, D.H.Crocker, August 1982, , http://www.internet.nic .

REVISION HISTORY

Version	Date	Changes
1.2	November 13, 2013	Revise Tag 247 ExchangeId to include "CXT" for CX2; "CNQ" for CNSX, "LYX" for Lynx ATS. Revised Section 5.2 Trading Hours for the CDB to include CX2, CNSX, and Lynx ATS
1.1	March 31, 2011	Revise Tag 247 ExchangeId to include "SEL" for TMX Select Revised Section 5.2 Trading Hours for the CDB to include TMX Select
1.0	February 28, 2011	