CONFIDENTIAL

CBOE®

# Application Programming Interface

# Version 9.0.2

# CBOE API Volume 3 CMi Programmer's Guide to Messages and Data Types

Programmer's Guide to Messages and Data Types for the CBOE Market Interface (CMi)

***CBOE PROPRIETARY INFORMATION***

15 July 2011

Document #[API-03]

# Front Matter

## Disclaimer

# Table of Contents

# Change Notices

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

| Date | Version | Description of Change |
|---|---|---|
| 15 July 2011 | V9.0.2 | New constants for Wash Trade Prevention |
| 29 Apr 2011 | V9.0.1 | No changes |
| 14 Jan 2011 | V9.0 | New interfaces for CMi V9 |
| 23 Mar 2010 | V7.0 | Updated cmiUtil |
| 08 Jan 2010 | V7.0 | New interfaces for CMi V7 |
| 07 Aug 2009 | V6.1 | New contingency types for BID_PEG_CROSS and OFFER_PEG_CROSS |
| 22 May 2009 | V6.0 | New constants for Directed AIM |
| | | New interfaces for floor trades |
| 21 Nov 2008 | V5.3 | New constant for Intermarket_Sweep_Book |
| 03 Oct 2008 | V5.2 | New constants for CBSX enhance flash and BATS |
| 18 Jul 2008 | V5.1 | New constants for internal routing of order - OHS |
| 29 Feb 2008 | V5.0 | New interfaces for CMi V5.0 |
| 18 Jan 2008 | V4.2.4 | New BillingTypeIndicators |
| 02 Nov 2007 | V4.2.3 | New values for cmiConstants.idl |
| 01 June 2007 | V4.2.2 | New values for CBSX billing type indicators in cmiConstants.idl |
| 23 Feb 2007 | V4.2.1 | New values for: |
| | | const cmiMarketData::MarketIndicator SlOW_ON_BID_SIDE = 33 |
| | | const cmiMarketData::MarketIndicator SlOW_ON_ASK_SIDE = 34 |
| | | const cmiMarketData::MarketIndicator SlOW_ON_BOTH_SIDE = 35 |
| 15 Dec 2006 | V4.2 | New constant for User Input Monitor cancel reason |
| 20 Sept 2006 | V4.1 | Updated the contingency order type constants for Stock |
| 08 Sept 2006 | V4.1 | Add new contingency order type constants for Stock |
| 25 May 2006 | V4.0 | Updates for Market Data Express (MDX) |

| Date | Version | Description of Change |
|---|---|---|
| 06 Jan 2006 | V3.2b | Added new aution type AUCTION_SAL |
| 29 Jul 2005 | V3.2 | Added new auction type AUCTION_HAL |
| 08 Apr 2005 | V3.1a | Documentation errata |
| 17 Dec 2004 | V3.1 | Updates for internalization and auctions |
| 18 Jun 2004 | V3.0 | Hybrid and Stock updates. |
| 28 Apr 2004 | V2.52 | Constants and error code updates for the V2.5 production release. |
| 06 Feb 2004 | V2.63 | Method description changes for this release. |
| 10 Oct 2003 | V2.62 | Method description changes. |
| 29 Aug 2003 | V2.61 | Method description changes. |
| 31 Jul 2003 | V2.6 | Changes to support Market Linkage and Stock |
| 08 Jul 2003 | V2.51 | Revisions since the last release. |
| 14 Mar 2003 | V2.5 | Support for Hybrid |
| 24 Jan 2003 | V2.1 | Production Release update to support Linkage P orders. |
| 20 May 2002 | V2.0.1 | Corrected the number of subaccount characters to 10 instead of 6. |
| 22 Apr 2002 | V2.0 | Production Release |
| 27 Feb 2002 | V2.0b | Software Development Kit Beta 2 |
| 23 Jan 2002 | V2.0a | Software Development Kit Beta 1 |
| 14 Dec 2001 | V2.0 | Updated documentation only for Version 2.0. |
| 27 Apr 2001 | V1.0b | Added Market Data role information. |
| 16 Mar 2001 | V1.0a | Error corrections and updated to reflect that Strategies will not be included in Version 1.0. |
| 15 Jan 2001 | V1.0 | Production Version. |
| 15 Sep 2000 | V0.9 | Network testable version. |
| 28 Apr 2000 | V0.8 | Includes revisions to the CMi API since the last release. Refer to the Release Notes for full details. |
| 11 Feb 2000 | V0.7 | Includes revisions to the API since the last update. |
| | | Created additional diagrams to improve readability; updated/corrected definitions. |
| 30 Sep 1999 | V0.5 | Includes revisions to API since last update. |
| 3 July 1999 | V0.4 | API is substantially finished. |
| | | Added diagrams that show the various interfaces |

| Date | Version | Description of Change |
|---|---|---|
| | | and the relationship to the message structures used in the CMi. |
| 8 May 1999 | V0.3-1 | Properly updated API reference sections |
| | | Continued expand description of the API |
| 30 Apr 1999 | V0.3 | Support for Strategies. |
| | | Support for crossing notification and requests. |
| | | Introduced an administrator interface to support those who administrate and operate trading rooms. |
| | | Version reporting of the API now supported. |
| | | Mandatory heartbeat between client and CAS. |
| | | Trader Interface renamed to OrderEntry. |
| | | Product Definition Interface added to permit definition of strategies and future definition of FLEX style and OTC instruments. |
| | | Simplified market query interface. |
| | | User is now able to change password through the SessionManager interface. |

# About This Document

## Purpose

This document provides details on the messages and data types used in the CBOE Market Interface (CMi).

## Intended Audience

Software developers using the CMi to develop applications that use CBOE Exchange Services.

## Prerequisites

This document assumes that the reader has a working knowledge of CORBA and one of the programming languages supported by CORBA, such as C++ or Java. See the reference section of a list of books and web sites that can provide you with fundamentals on CORBA. Specifically, you should be familiar with CORBA modules, interfaces, structs, operations, IORs, exceptions, and callbacks.

You should have already read Volume 1: Overview and Concepts and Volume 2: CMi Programmer's Guide to Interfaces and Operations.

## Related Documents

| Document Number | Document Description |
|---|---|
| roadmap.doc | CBOE API and CAS Document Road Map |
| API-01 | CBOE API Volume 1: Overview and Concepts |
| API-02 | CBOE API Volume 2: CMi Programmer's Guide to Interfaces and Operations |
| API-04 | CBOE API Volume 4 CMI Dictionary of Attributes and Operations |
| API-05 | CBOE API Volume 5: Using CMi with Specific Object Request Brokers |
| API-06 | CBOE API Volume 6: Connecting to the CBOE Network |
| API-07 | CBOE API Volume 7: CBOEdirect Certification and Testing Procedures |
| API-08 | CBOE API Volume 8: CMi Programmer's Guide to the Market Data Express (MDX) Data Feed |
| FIX-01 | CBOE API FIX Protocol Support Volume 1: Overview |
| CAS-01 | CBOE Application Server Volume 1: Overview and Concepts |
| CAS-02 | CBOE Application Server Volume 2: CBOE Application |

| **Document Number** | **Document Description** |
| --- | --- |
| | Server Simulator for Stand Alone Testing |

## Diagram Notation

The diagrams in this document are based upon the Unified Modeling Language standard. The diagrams were created using Rational Rose from Rational Software Version 4.5. The documents were generated using Rational SoDA for Word Version 2.52. Extensive use of stereotypes is used to elaborate IDL features.

## Programming Examples—Notation and Location

All programming examples are specified using Java package notation. For instance, the first example program is referenced in the documentation as:

com.cboe.examples.example1.example1.java

This example program can be found in the following directory for Java:

*${INSTALL_DIR}\Examples\Java\com\cboe\examples\example1\example1.java*

Where:

*${INSTALL_DIR}* is the install directory you selected during the installation of the CBOE Software Development Kit. The default location is *C:\CBOE\CMi2.0.*

The C++ version of the example program can be found in:

${INSTALL_DIR}\Examples\cpp

Note that all C++ code is kept in the same directory.

Scripts and IDL compilation for specific ORBS is provided in the Platforms directory, which is located in:

${INSTALL_DIR}\Examples\orb

## Support and Questions Regarding This Document

Questions regarding this document can be directed to The Chicago Board Options Exchange at 312.786.7300 or via e-mail: api@cboe.com.

The latest version of this document can be found at the CBOE web site http://systems.cboe.com.

# Introduction

The CBOE is adding new interfaces to provide access to exchange services. These interfaces are designed to support both electronic and open outcry trading at the Chicago Board Options Exchange.

The first of these interfaces is an Application Programming Interface (API) that provides access to all exchange trading services and is targeted at firms making markets at CBOE. This API is known as *the CBOE Market* Interface (CMi). CMi is a distributed object interface based upon the CORBA (Common Object Request Broker Architecture) standard from the Object Management Group (OMG). The interface is defined using the Interface Definition Language (IDL), which is an OMG and ISO standard. Messages are transported using the Internet Inter-Orb Protocol (IIOP), which operates over standard Internet protocols (TCP/IP).

The second interface is a message-based protocol based upon the Financial Information Exchange (FIX) protocol. CBOE's implementation of the FIX protocol should be of particular interest to retail and institutional firms that require order routing to CBOE markets. The FIX protocol is implemented over TCP/IP. FIX is an important messaging protocol in the financial industry. CBOE continues to work with the FIX protocol organization and to participate in FIX working groups to help evolve the FIX protocol for wider use in exchange based derivatives markets.

## An Introduction to CMi Datatypes

Every attempt has been made to apply a consistent usage of data types across the CMi. The CMi uses key values, which are unique numeric numbers to represent entities within the system. The CMi also features extensive use of CORBA IDL typedefs.

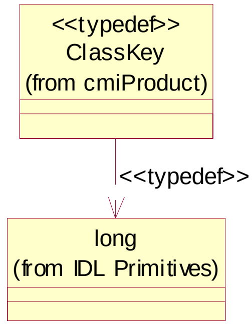
### The Use of Keys

The CMi uses key values to represent entities within the system. For instance each class within the system is identified by its classKey, which is a CORBA long. In the CMi you will consistently find these keys defined as a typedef to minimize changes, should the data type need to be changed in the future.

```
typedef long classKey;
```

The classKey is shown in the following UML Class Diagram. The typedef is shown as a class that has a dependency on an IDL datatype, in this case the long integer datatype.

***Class Diagram: TypeDefExample***




The typedef, in this case classKey, is then used throughout the CMi interfaces and structs. For instance, the following struct from the cmiProduct module, ProductKeysStruct, contains the keys for a product:

```
struct ProductKeysStruct {
    cmiProduct::ProductKey productKey;
    cmiProduct::ClassKey classKey;
    cmiProduct::ProductType productType;
    cmiProduct::ReportingClassKey reportingClass;
  };
```

The most important key types in CMi are the ProductKey—which defines each unique product at CBOE (option series and underlying products). Each product is assigned to a class. Each class is identified by a unique ClassKey. It is important to note that these keys are unique numbers, not necessarily serial numbers—you should assume that the keys are unordered.

## The use of CORBA Structs

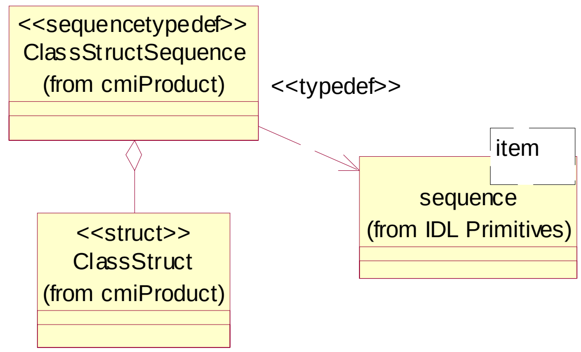
The CMi relies heavily on the use of CORBA structs that are passed between the CAS and your client application. This was done primarily to minimize the performance impacts inherent in distributed objects. These structs map to final classes in Java, structs in C++.

## Sequences of Structs

The CMi interfaces will often return a sequence of structs. The CMi uses typedefs to define sequences of structs. The following UML class diagram shows how this mapping is represented in this documentation. In the following example the ClassStructSequence is shown. It is shown as having a dependency on the CORBA Sequence and is composed of multiple (0 or more) ClassStructs.

```
typedef sequence <ClassStruct> ClassStructSequence;]
```

***Class Diagram: TypeDefSequenceExample***




## Mapping of CORBA IDL Datatypes to Programming Languages

The CORBA Standards specify how the IDL datatypes map to a particular programming language. For instance, Java does not have the typedef feature, so CORBA maps typedefs back to their underlying datatypes. For instance, in CMi the classKey is defined in IDL as:

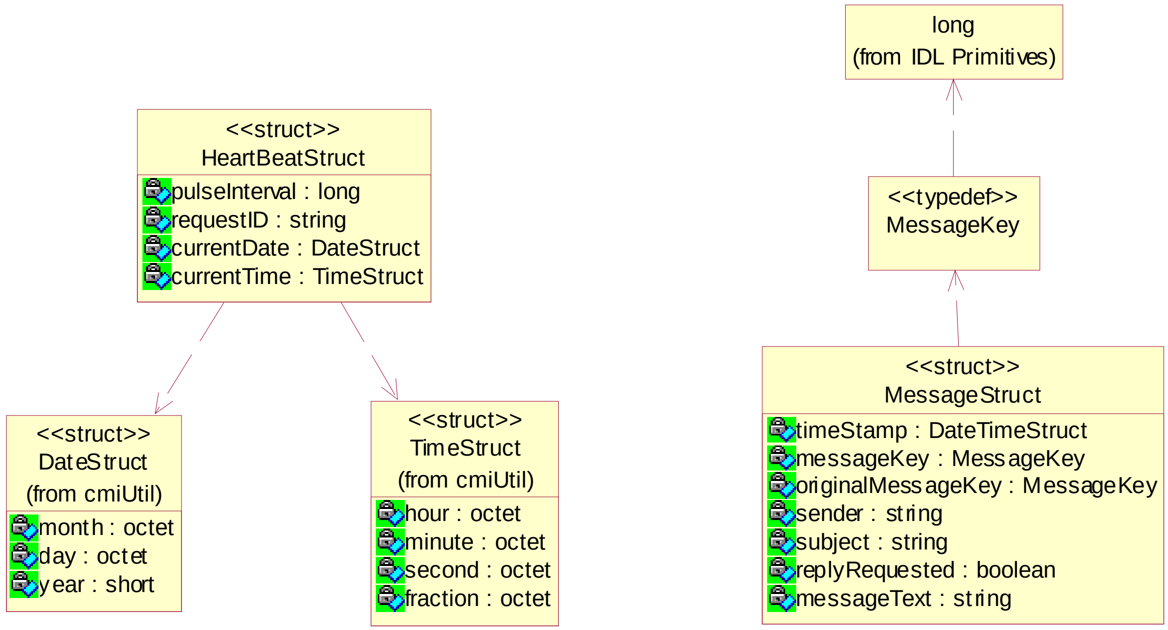
```
typedef long classKey;
```

Since Java does not support typedefs, all references to the classKey type are converted to the native IDL datatype, then mapped to the programming language. Since the classKey was defined as a long and an IDL long maps to a Java int, each occurrence of classKey will be changed to int.

More information can be obtained from the CORBA 2.2 Specification from http://www.omg.org.

## *Module cmiAdmin*

Types and messages for use in administration of the CAS and Client interface.




**Typedefs:**

| | |
|---|---|
| MessageKey *is a* long | Unique Identifer of a message sent between client application and system application. |

**Typedefs for Sequences:**

**Structs:**

### HeartBeatStruct

The HeartBeatStruct is passed to the client application over the CMIUserSessionAdmin::acceptHeartBeat() operation. The CMIUserSessionAdmin callback must be implemented in the client application.

It is important that the client not alter the RequestID. At this time, any changes to the pulseInterval are ignored by the CAS.

An example implementation of the CMIUserSessionAdmin callback interface is provided in com.cboe.examples.common.UserSessionAdminCallback.

| Attribute | Type | Documentation |
| --- | --- | --- |
| currentDate | DateStruct | Current Date reported by CAS |
| currentTime | TimeStruct | Current Time reported by the CAS. |
| pulseInterval | long | Interval between heartbeats in seconds |
| requestID | string | Contains the request ID that generated the heartbeat |

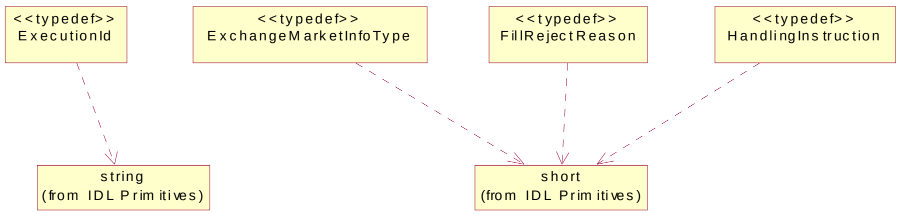
### MessageStruct

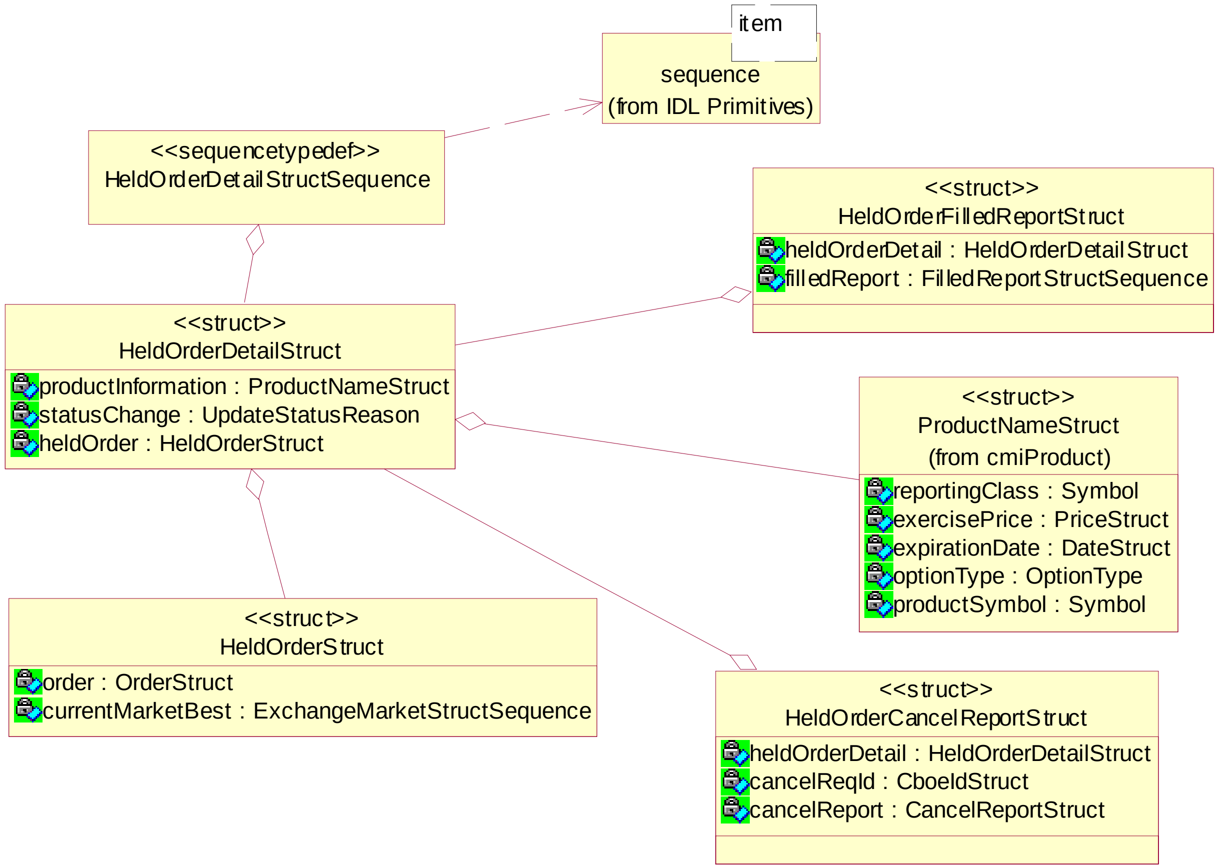
Messages passed from system operator to user of the trading system.

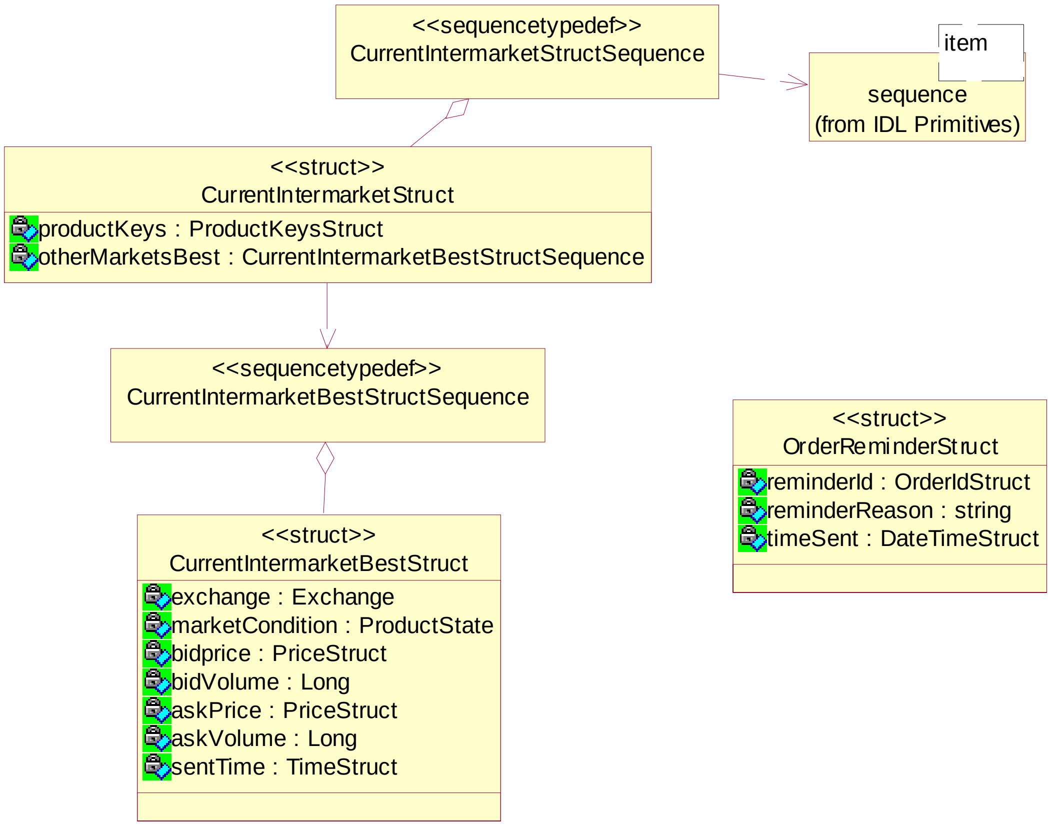
| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| messageKey | MessageKey | Unique identifier for the message |
| messageText | string | Text of the message |
| originalMessageKey | MessageKey | Original message key for which this message is a response |
| replyRequested | boolean | Indicates if a reply is requested - true if a response is requested |
| sender | string | Userid of the sender of the message |
| subject | string | Subject of the message |
| timeStamp | DateTimeStruct | Timestamp when message was sent |

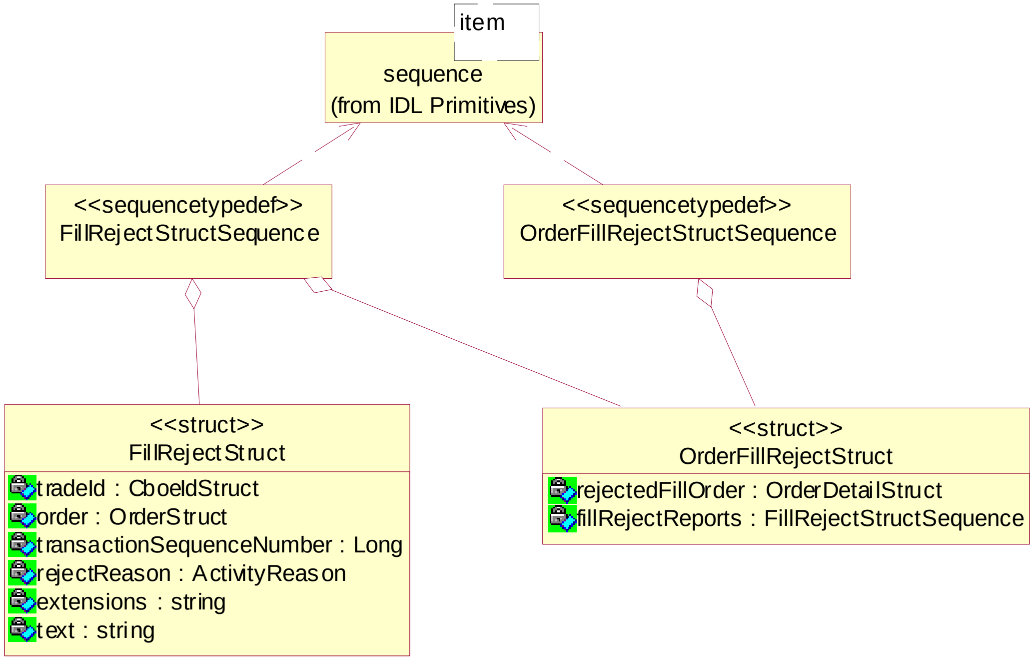
## *Module cmiIntermarketMessages*

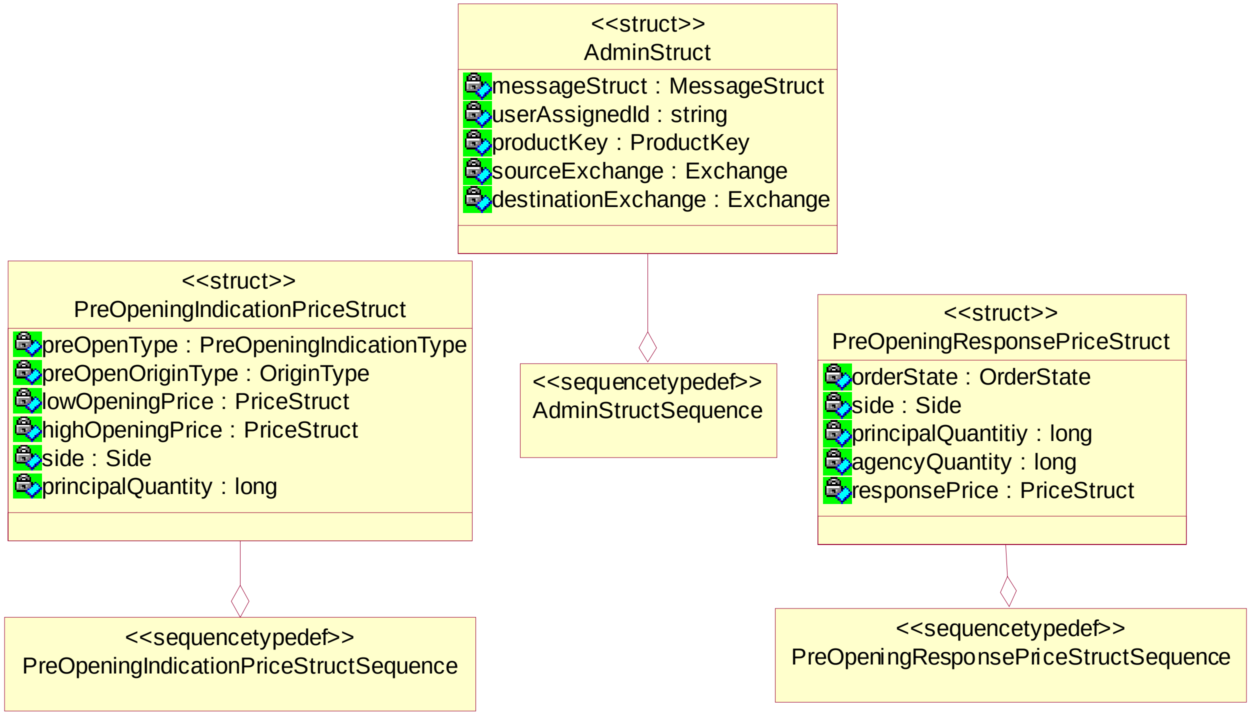
Intermarket messages used in the CBOE Market Interface.

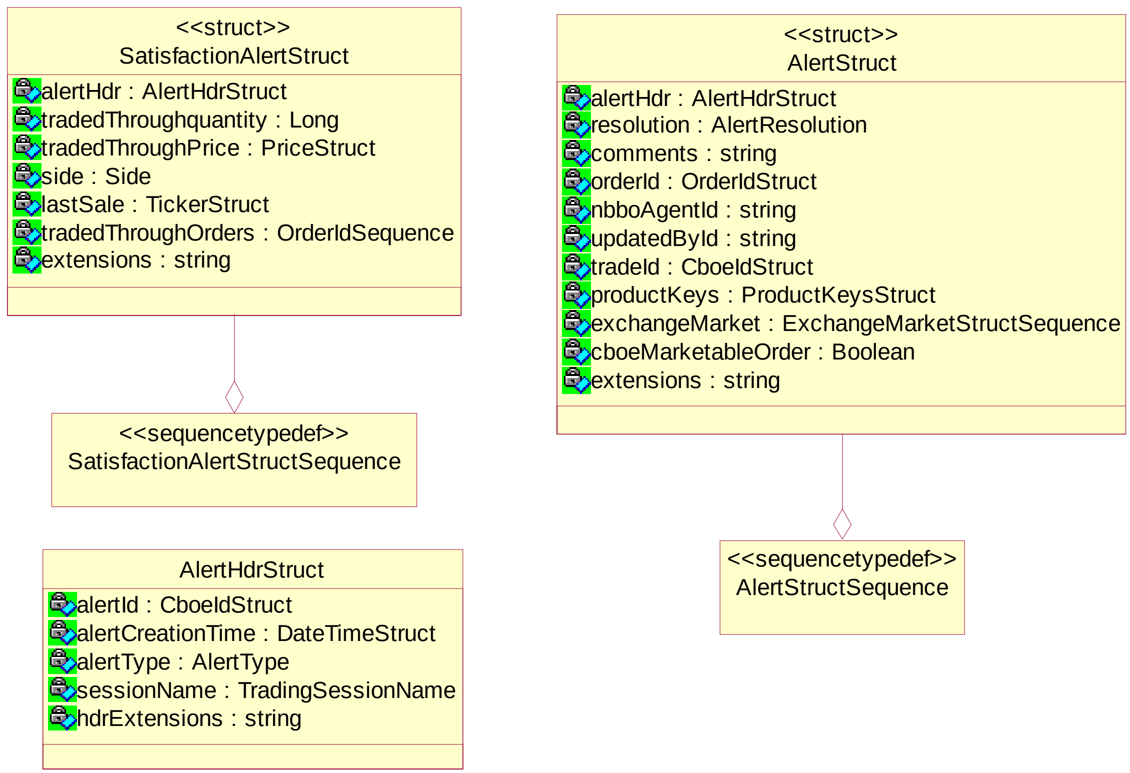


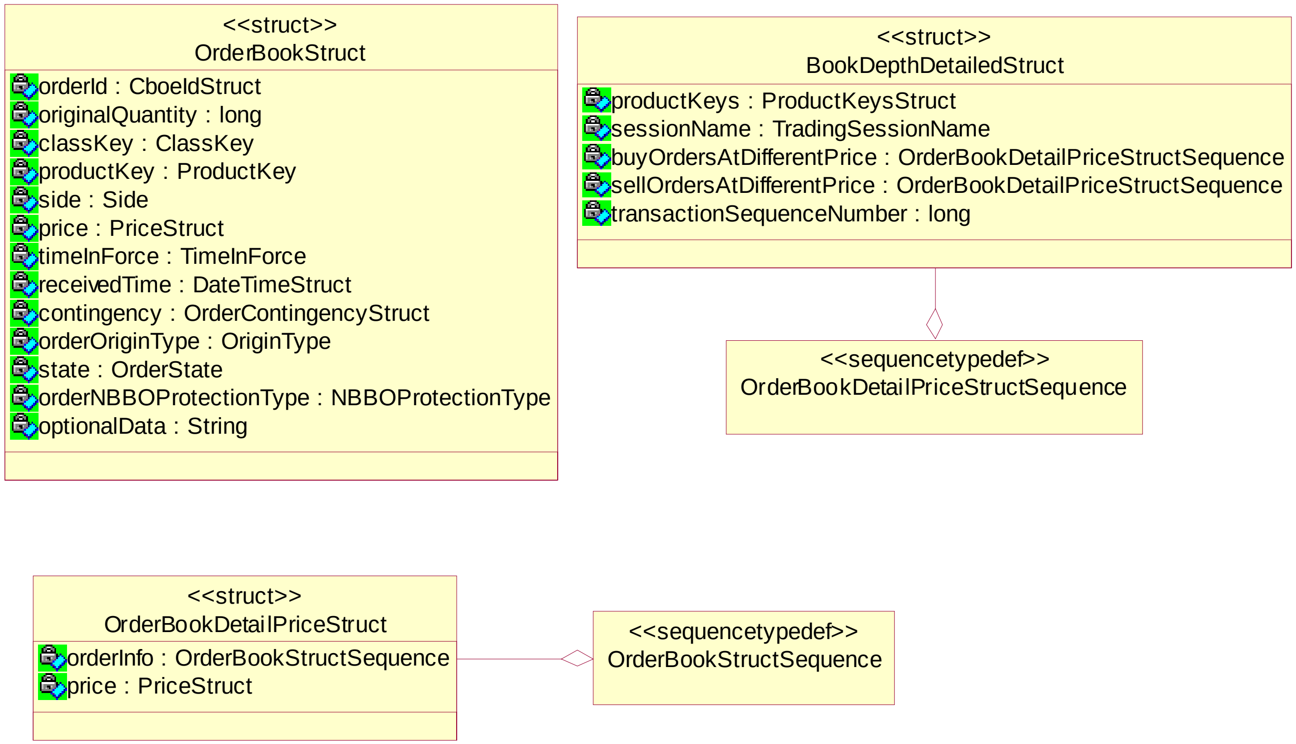

item
sequence
(from IDL Primitives)
<<sequencetypedef>>
HeldOrderDetailStructSequence
<<struct>>
HeldOrderFilledReportStruct
heldOrderDetail : HeldOrderDetailStruct
filledReport : FilledReportStructSequence
<<struct>>
HeldOrderDetailStruct
productInformation : ProductNameStruct
statusChange : UpdateStatusReason
heldOrder : HeldOrderStruct
<<struct>>
ProductNameStruct
(from cmiProduct)
reportingClass : Symbol
exercisePrice : PriceStruct
expirationDate : DateStruct
optionType : OptionType
productSymbol : Symbol
<<struct>>
HeldOrderStruct
order : OrderStruct
currentMarketBest : ExchangeMarketStructSequence
<<struct>>
HeldOrderCancelReportStruct
heldOrderDetail : HeldOrderDetailStruct
cancelReqId : CboeIdStruct
cancelReport : CancelReportStruct


<<sequencetypedef>>
CurrentIntermarketStructSequence
item
sequence
(from IDL Primitives)
<<struct>>
CurrentIntermarketStruct
productKeys : ProductKeysStruct
otherMarketsBest : CurrentIntermarketBestStructSequence
<<sequencetypedef>>
CurrentIntermarketBestStructSequence
<<struct>>
CurrentIntermarketBestStruct
exchange : Exchange
marketCondition : ProductState
bidprice : PriceStruct
bidVolume : Long
askPrice : PriceStruct
askVolume : Long
sentTime : TimeStruct
<<struct>>
OrderReminderStruct
reminderId : OrderIdStruct
reminderReason : string
timeSent : DateTimeStruct


item
sequence
(from IDL Primitives)
<<sequencetypedef>>
FillRejectStructSequence
<<sequencetypedef>>
OrderFillRejectStructSequence
<<struct>>
FillRejectStruct
tradeId : CboeIdStruct
order : OrderStruct
transactionSequenceNumber : Long
rejectReason : ActivityReason
extensions : string
text : string
<<struct>>
OrderFillRejectStruct
rejectedFillOrder : OrderDetailStruct
fillRejectReports : FillRejectStructSequence


<<struct>>
AdminStruct
messageStruct : MessageStruct
userAssignedId : string
productKey : ProductKey
sourceExchange : Exchange
destinationExchange : Exchange
<<sequencetypedef>>
AdminStructSequence
<<struct>>
PreOpeningIndicationPriceStruct
preOpenType : PreOpeningIndicationType
preOpenOriginType : OriginType
lowOpeningPrice : PriceStruct
highOpeningPrice : PriceStruct
side : Side
principalQuantity : long
<<sequencetypedef>>
PreOpeningIndicationPriceStructSequence
<<struct>>
PreOpeningResponsePriceStruct
orderState : OrderState
side : Side
principalQuantitiy : long
agencyQuantity : long
responsePrice : PriceStruct
<<sequencetypedef>>
PreOpeningResponsePriceStructSequence


<<struct>>
SatisfactionAlertStruct
alertHdr : AlertHdrStruct
tradedThroughquantity : Long
tradedThroughPrice : PriceStruct
side : Side
lastSale : TickerStruct
tradedThroughOrders : OrderIdSequence
extensions : string
<<sequencetypedef>>
SatisfactionAlertStructSequence
AlertHdrStruct
alertId : CboeIdStruct
alertCreationTime : DateTimeStruct
alertType : AlertType
sessionName : TradingSessionName
hdrExtensions : string
<<struct>>
AlertStruct
alertHdr : AlertHdrStruct
resolution : AlertResolution
comments : string
orderId : OrderIdStruct
nbboAgentId : string
updatedById : string
tradeId : CboeIdStruct
productKeys : ProductKeysStruct
exchangeMarket : ExchangeMarketStructSequence
cboeMarketableOrder : Boolean
extensions : string
<<sequencetypedef>>
AlertStructSequence


<<struct>>
OrderBookStruct
orderId : CboeIdStruct
originalQuantity : long
classKey : ClassKey
productKey : ProductKey
side : Side
price : PriceStruct
timeInForce : TimeInForce
receivedTime : DateTimeStruct
contingency : OrderContingencyStruct
orderOriginType : OriginType
state : OrderState
orderNBBOProtectionType : NBBOProtectionType
optionalData : String
<<struct>>
BookDepthDetailedStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
buyOrdersAtDifferentPrice : OrderBookDetailPriceStructSequence
sellOrdersAtDifferentPrice : OrderBookDetailPriceStructSequence
transactionSequenceNumber : long
<<sequencetypedef>>
OrderBookDetailPriceStructSequence
<<struct>>
OrderBookDetailPriceStruct
orderInfo : OrderBookStructSequence
price : PriceStruct
<<sequencetypedef>>
OrderBookStructSequence

---

**Typedefs:**

| | |
|---|---|
| | |
| | |
| ExchangeMarketInfoType *is a* short | |
| ExecutionId *is a* string | |
| FillRejectReason *is a* short | |
| HandlingInstruction *is a* short | |
| | |
| | |

**Typedefs for Sequences:**

AdminStructSequence *is a typedef of a sequence of* AdminStruct

AlertStructSequence *is a typedef of a sequence of* AlertStruct

CurrentIntermarketBestStructSequence *is a typedef of a sequence of* CurrentIntermarketStruct
CurrentIntermarketBestStructSequence *is a typedef of a sequence of* CurrentIntermarketBestStruct

ExchangeMarketStructV2Sequence *is a typedef of a sequence of* ExchangeMarketStructV2

CurrentIntermarketStructSequence *is a typedef of a sequence of* CurrentIntermarketStruct

AlertStructV2Sequence *is a typedef of a sequence of* AlertStructV2

FillRejectStructSequence *is a typedef of a sequence of* OrderFillRejectStruct

FillRejectStructSequence *is a typedef of a sequence of* FillRejectStruct

HeldOrderDetailStructSequence *is a typedef of a sequence of* HeldOrderDetailStruct

HeldOrderStructSequence *is a typedef of a sequence of* HeldOrderStruct

OrderBookDetailPriceStructSequence *is a typedef of a sequence of* BookDepthDetailedStruct

OrderBookStructSequence *is a typedef of a sequence of* OrderBookDetailPriceStruct

OrderFillRejectStructSequence *is a typedef of a sequence of* FillRejectStruct

OrderFillRejectStructSequence *is a typedef of a sequence of* OrderFillRejectStruct

PreOpeningResponsePriceStructSequence *is a typedef of a sequence of* PreOpeningResponsePriceStruct

SatisfactionAlertStructSequence *is a typedef of a sequence of* SatisfactionAlertStruct

---

**Structs:**

**AdminStruct**

Administration messages sent of Stock ITS participants.

| Attribute | Type | Documentation |
|---|---|---|
| destinationExchange | Exchange | Identifies the destination exchange. |
| messageStruct | MessageStruct | Information contained in the message.  See cmiAdmin::MessageStruct |
| productKey | ProductKey | Product identifier. |
| sourceExchange | Exchange | Identifies the exchange source. |
| userAssignedId | string | User identifier. |

**AlertStruct**

Alert sent to the Trading Floor Liaison (TFL) related to the NBBO agent.

| Attribute | Type | Documentation |
|---|---|---|
| alertHdr | AlertHdrStruct | Alert header |
| cboeMarketableOrder | Boolean | Order is marketable against CBOE BBBO. |
| comments | string | Comments pertaining to the alert. |
| exchangeMarket | ExchangeMarketStructSequence | Information on the exchange.  See cmiIntermarketMessages::ExchangeMarketStructSequence |
| extensions | string | |
| nbboAgentId | string | NBBO agent user ID. |
| orderId | OrderIdStruct | The order Id pertaining to the alert. |
| productKeys | ProductKeysStruct | Product identifier. |
| resolution | AlertResolution | Resolution for the alert. |
| tradeId | CboeIdStruct | Identifies the trade.  See cmiUtil::CboeIdStruct. |

| Attribute | Type | Documentation |
|---|---|---|
| updatedById | string | Tracks the last user who updated the field. |

**BookDepthDetailedStruct**

Book depth order details

| Attribute | Type | Documentation |
|---|---|---|
| buyOrdersAtDifferentPrice | OrderBookDetailPriceStructSequence | Buy order prices. |
| productKeys | ProductKeysStruct | Products identifier. |
| sellOrdersAtDifferentPrice | OrderBookDetailPriceStructSequence | Sell order prices. |
| sessionName | TradingSessionName | Trading session name. |
| transactionSequenceNumber | long | Transaction identifier. |

**CurrentIntermarketBestStruct**

| Attribute | Type | Documentation |
|---|---|---|
| askPrice | PriceStruct | Current best intermarket ask price. |
| askVolume | Long | The intermarket ask volume. |
| bidprice | PriceStruct | Current intermarket bid price. |
| bidVolume | Long | The intermarket bid volume. |
| exchange | Exchange | Exchange identifier. |

| Attribute | Type | Documentation |
|---|---|---|
| marketCondition | ProductState | The type of market condition reported. |
| sentTime | TimeStruct | Time the intermarket information was sent from the trading system. |

### CurrentIntermarketStruct

| Attribute | Type | Documentation |
|---|---|---|
| otherMarketsBest | CurrentIntermarketBestStructSequence | A sequence of CurrentIntermarketBestStructs. |
| productKeys | ProductKeysStruct | Identifier of the intermarket product. |

### ExchangeMarketStruct

| Attribute | Type | Documentation |
|---|---|---|
| askExchangevolumes | ExchangeVolumeStructSequence | A series of exchange names and volumes. |
| bestAskPrice | PriceStruct | Exchange(s)  best ask price. |
| bestBidPrice | PriceStruct | Exchange(s) best bid price. |
| bidExchangeVolumes | ExchangeVolumeStructSequence | A series of exchange names and volume. |
| marketInfoType | ExchangeMarketInfoType | Exchange  market information. |

### ExchangeMarketStructV2

| Attribute | Type | Documentation |
|---|---|---|
| marketInfoType | ExchangeMarketInfoType | Exchange market information. |
| bestAskPrice | PriceStruct | Exchange(s) best ask price. |
| bestBidPrice | PriceStruct | Exchange(s) best bid price. |
| bidExchangeVolumes | ExchangeVolumeStructSequence | A series of exchange names and volume. |
| marketInfoType | ExchangeMarketInfoType | Exchange market information. |
| askExchangevolumes | ExchangeVolumeStructSequence | A series of exchange names and volumes. |
| time | long | |

**FillRejectRequestStruct**

| Attribute | Type | Documentation |
|---|---|---|
| fillReportExtensions | String | Extensions to the order message. |
| orderId | OrderIdStruct | Identifier of order to reject fill. |
| rejectReason | ActivityReason | Reject reason. |
| tradedQuantity | long | Quantity traded |
| tradePrice | PriceStruct | |

**FillRejectStruct**

| Attribute | Type | Documentation |
|---|---|---|
| extensions | string | Extensions to the order message. |

| Attribute | Type | Documentation |
|---|---|---|
| order | OrderStruct | Order information. |
| rejectReason | ActivityReason | Reject reason. |
| text | string | Fill reject text. |
| tradeId | CboeIdStruct | Unique identifier that identifies the trade. |
| transactionSequenceNumber | Long | Transaction sequence number for the fill reject. |

**HeldOrderCancelReportStruct**

| Attribute | Type | Documentation |
|---|---|---|
| cancelReport | CancelReportStruct | Held order cancel report. |
| cancelReqId | CboeIdStruct | Identifier of the cancel request. |
| heldOrderDetail | HeldOrderDetailStruct | Specifications of the held order. |

**HeldOrderCancelRequestStruct**

| Attribute | Type | Documentation |
|---|---|---|
| cancelReqId | CboeIdStruct | Identifier of the cancel request. |
| cancelRequest | CancelRequestStruct | Cancel request of the held order. |

**HeldOrderDetailStruct**

| Attribute | Type | Documentation |
|---|---|---|
| heldOrder | HeldOrderStruct | Order being held for price improvement. |
| productInformation | ProductNameStruct | Intermarket products information. |
| statusChange | UpdateStatusReason | Held order status update. |

**HeldOrderFilledReportStruct**

| Attribute | Type | Documentation |
|---|---|---|
| filledReport | FilledReportStructSequence | The fill report of the held order. |
| heldOrderDetail | HeldOrderDetailStruct | Details of the held order that was filled. |

**HeldOrderStruct**

| Attribute | Type | Documentation |
|---|---|---|
| currentMarketBest | ExchangeMarketStructSequence | Current intermarket best. |
| order | OrderStruct | Held order information. |

**OrderBookDetailPriceStruct**

Order book price information.

| Attribute | Type | Documentation |
|---|---|---|
| orderInfo | OrderBookStructSequence | Sequence of order information. |
| price | PriceStruct | Price information. |

**OrderBookStruct**

Order book information.

| Attribute | Type | Documentation |
|---|---|---|
| classKey | ClassKey | Product class identifier. |
| contingency | OrderContingencyStruct | Type of order contingency. |

| Attribute | Type | Documentation |
|---|---|---|
| optionalData | String | Optional text field.<br>Firms that want to give one DPM priority in participating in a trade use this field using P:EXCH.FIRM; or P:FIRM; EXCH is the exchange acronym. FIRM is the Firm acronym. EXCH is optional.<br>For Sweep and AIM: Enter A:AIS in the primary order to instruct CBOE to sweep all better priced protected quotes at away exchanges at the same time as the commencement of the AIM auction.<br>For AIM Auctions: Enter A:AIM as the first characters. If a Firm does not wish to cancel the primary order when the auction expires, the Firm must enter A:AIR, instead of A:AIM, in this field. This will designate the primary order to be returned to the system and trade or book as a regular order.<br>Options:<br>For Customer orders, the first four characters of this field get reported to the last four characters of the CBOE Trade Match Optional Data field. These four characters get reported to the OCC. |
| orderId | CboeIdStruct | Identifies the order. |
| orderNBBOProtectionType | NBBOProtectionType | Type of NBBO protection on the order. |
| orderOriginType | OriginType | Type of order (customer, MM, etc.) |
| originalQuantity | long | Beginning quantity. |
| price | PriceStruct | Order price. |
| productKey | ProductKey | Product identifier. |
| receivedTime | DateTimeStruct | Time the order was received. |
| side | Side | Buy or Sell side of the order. |
| state | OrderState | The state of the order. |
| timeInForce | TimeInForce | Time order is in force (Day, Good until cancelled, Good until date) |

### OrderFillRejectStruct

| Attribute | Type | Documentation |
|---|---|---|
| fillRejectReports | FillRejectStructSequence | A series of fill reject reports. |
| rejectedFillOrder | OrderDetailStruct | Rejected fill order detail. |

### OrderReminderStruct

| Attribute | Type | Documentation |
|---|---|---|
| reminderId | OrderIdStruct | Unique reminder id. |
| reminderReason | string | Reason for the reminder. |
| timeSent | DateTimeStruct | Time the reminder was sent. |

### PreOpeningIndicationPriceStruct

Pre-opening price information during the opening of a security.  Used by the Stock Intermarket Trading System (ITS)

| Attribute | Type | Documentation |
|---|---|---|
| highOpeningPrice | PriceStruct | High opening price in the price range used in the Stock ITS. |
| lowOpeningPrice | PriceStruct | The low opening price in the price range used in the Stock ITS. |
| preOpenOriginType | OriginType | Origin type used in pre-open by the Stock ITS. |
| preOpenType | PreOpeningIndicationType | A pre-open message type used by the Stock  ITS. |

| Attribute | Type | Documentation |
|---|---|---|
| principalQuantity | long | Principal order quantity in ITS. |
| side | Side | Buy or sell side of the order in ITS |

### PreOpeningResponsePriceStruct

Used by the Stock Intermarket Trading System (ITS) for participation in the opening.

| Attribute | Type | Documentation |
|---|---|---|
| agencyQuantity | long | Quantitiy of agency participant at the opening. |
| orderState | OrderState | State of the order. Optional, only valid state-Cancel. |
| principalQuantitiy | long | Quantitiy of Principal participant at the opening. |
| responsePrice | PriceStruct | The price the participant is responding with for the opening. See cmiUtil::PriceStruct |
| side | Side | Side of the order. |

### SatisfactionAlertStruct

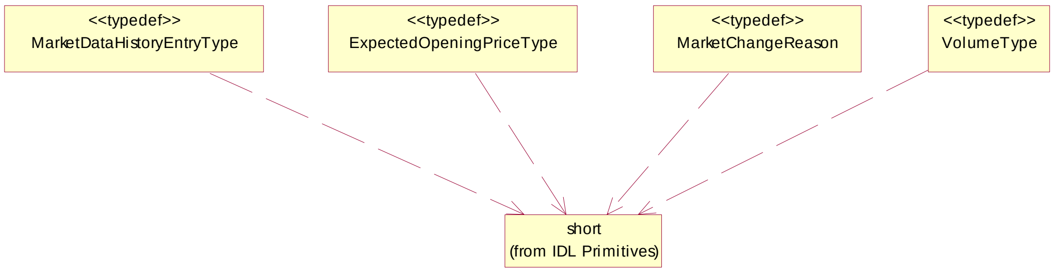
Alert that the CBOE Book was traded through by a remote exchange.

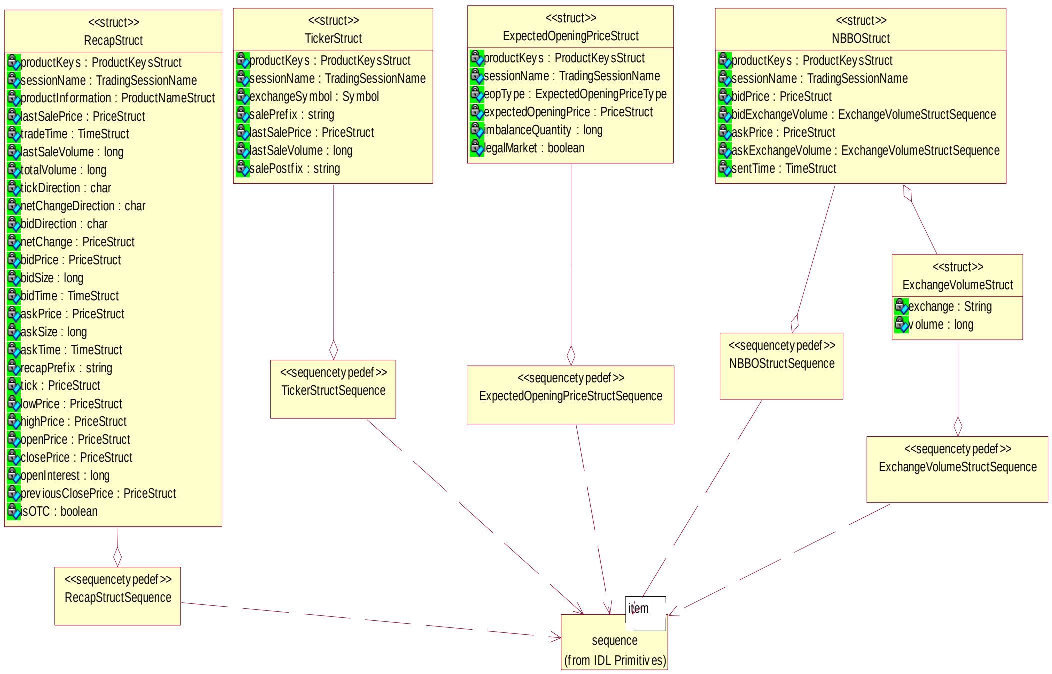
| Attribute | Type | Documentation |
|---|---|---|
| alertHdr | AlertHdrStruct | |
| extensions | string | |
| lastSale | TickerStruct | Last sale information. See cmiMarketData::TickerStruct |
| side | Side | The side (buy or sell) of the S order that was traded. |

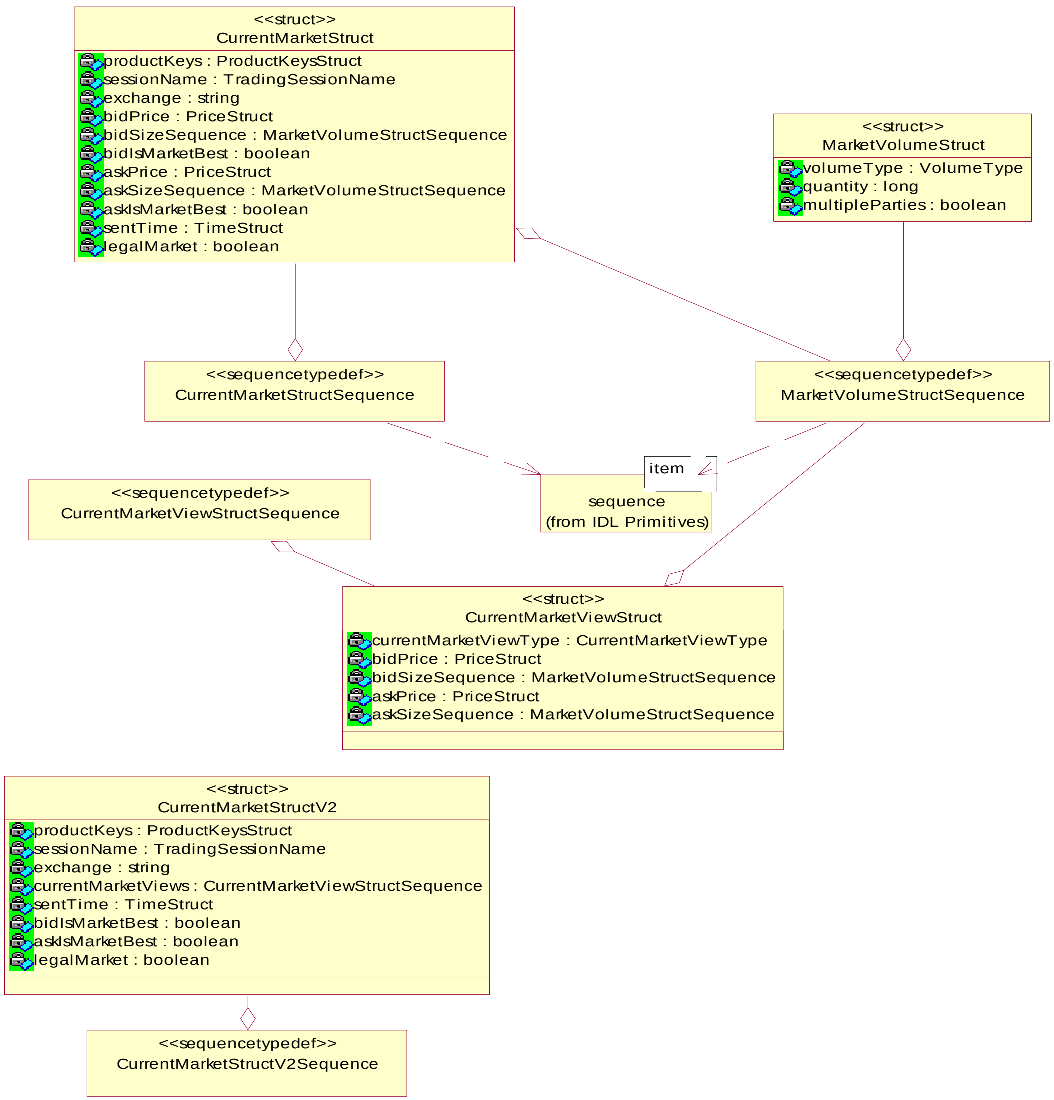
| Attribute | Type | Documentation |
|---|---|---|
| tradedThroughOrders | OrderIdSequence | Orders that were traded through. |
| tradedThroughPrice | PriceStruct | The trade through price. |
| tradedThroughquantity | Long | The trade through quantity. |

## *Module cmiMarketData*

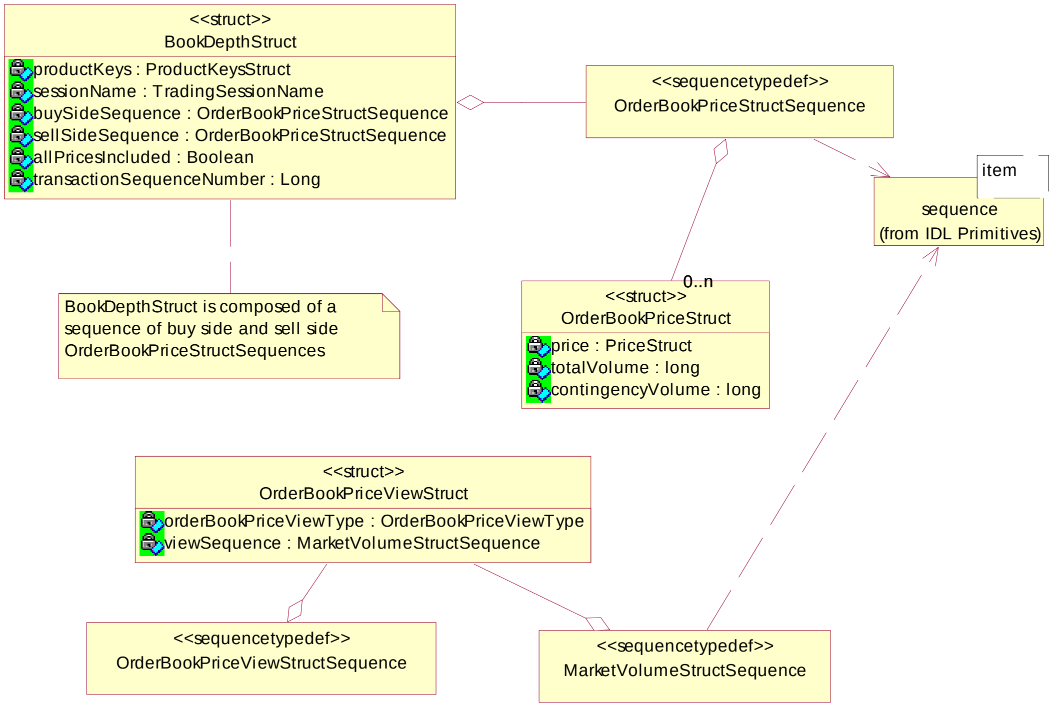
Order book depth information provides the total volume and contingency volume for the top N(5) best bid and ask prices. Order book depth information is available on a request not on a subscription basis.  Access to this call will be limited on a per minute basis and by user type.

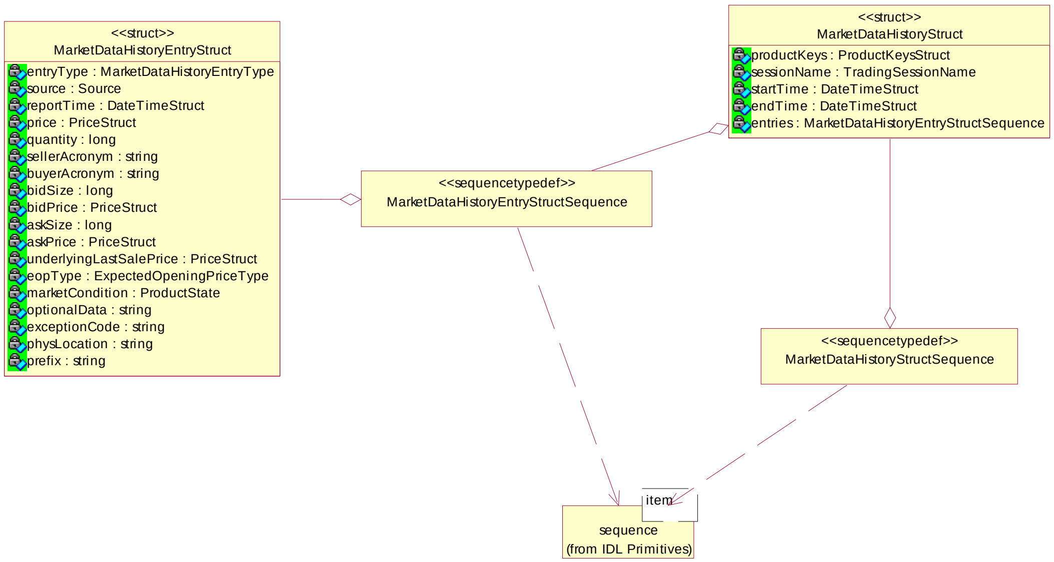



<<struct>>
RecapStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
productInformation : ProductNameStruct
lastSalePrice : PriceStruct
tradeTime : TimeStruct
lastSaleVolume : long
totalVolume : long
tickDirection : char
netChangeDirection : char
bidDirection : char
netChange : PriceStruct
bidPrice : PriceStruct
bidSize : long
bidTime : TimeStruct
askPrice : PriceStruct
askSize : long
askTime : TimeStruct
recapPrefix : string
tick : PriceStruct
lowPrice : PriceStruct
highPrice : PriceStruct
openPrice : PriceStruct
closePrice : PriceStruct
openInterest : long
previousClosePrice : PriceStruct
isOTC : boolean
<<struct>>
TickerStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
exchangeSymbol : Symbol
salePrefix : string
lastSalePrice : PriceStruct
lastSaleVolume : long
salePostfix : string
<<struct>>
ExpectedOpeningPriceStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
eopType : ExpectedOpeningPriceType
expectedOpeningPrice : PriceStruct
imbalanceQuantity : long
legalMarket : boolean
<<struct>>
NBBOStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
bidPrice : PriceStruct
bidExchangeVolume : ExchangeVolumeStructSequence
askPrice : PriceStruct
askExchangeVolume : ExchangeVolumeStructSequence
sentTime : TimeStruct
<<struct>>
ExchangeVolumeStruct
exchange : String
volume : long
<<sequencetypedef>>
TickerStructSequence
<<sequencetypedef>>
ExpectedOpeningPriceStructSequence
<<sequencetypedef>>
NBBOStructSequence
<<sequencetypedef>>
ExchangeVolumeStructSequence
<<sequencetypedef>>
RecapStructSequence
item
sequence
(from IDL Primitives)

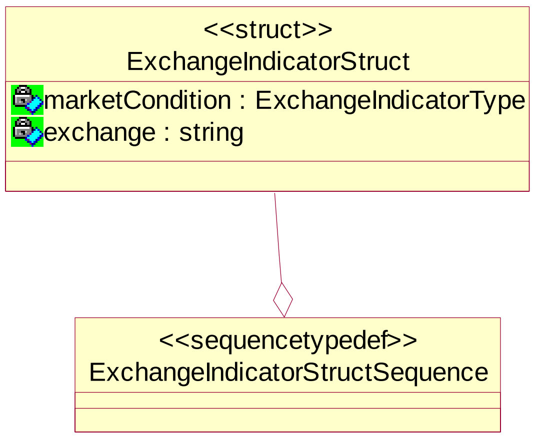


<<struct>>
CurrentMarketStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
exchange : string
bidPrice : PriceStruct
bidSizeSequence : MarketVolumeStructSequence
bidIsMarketBest : boolean
askPrice : PriceStruct
askSizeSequence : MarketVolumeStructSequence
askIsMarketBest : boolean
sentTime : TimeStruct
legalMarket : boolean
<<struct>>
MarketVolumeStruct
volumeType : VolumeType
quantity : long
multipleParties : boolean
<<sequencetypedef>>
CurrentMarketStructSequence
<<sequencetypedef>>
MarketVolumeStructSequence
<<sequencetypedef>>
CurrentMarketViewStructSequence
item
sequence
(from IDL Primitives)
<<struct>>
CurrentMarketViewStruct
currentMarketViewType : CurrentMarketViewType
bidPrice : PriceStruct
bidSizeSequence : MarketVolumeStructSequence
askPrice : PriceStruct
askSizeSequence : MarketVolumeStructSequence
<<struct>>
CurrentMarketStructV2
productKeys : ProductKeysStruct
sessionName : TradingSessionName
exchange : string
currentMarketViews : CurrentMarketViewStructSequence
sentTime : TimeStruct
bidIsMarketBest : boolean
askIsMarketBest : boolean
legalMarket : boolean
<<sequencetypedef>>
CurrentMarketStructV2Sequence


<<struct>>
BookDepthStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
buySideSequence : OrderBookPriceStructSequence
sellSideSequence : OrderBookPriceStructSequence
allPricesIncluded : Boolean
transactionSequenceNumber : Long
<<sequencetypedef>>
OrderBookPriceStructSequence
item
sequence
(from IDL Primitives)
BookDepthStruct is composed of a sequence of buy side and sell side OrderBookPriceStructSequences
0..n
<<struct>>
OrderBookPriceStruct
price : PriceStruct
totalVolume : long
contingencyVolume : long
<<struct>>
OrderBookPriceViewStruct
orderBookPriceViewType : OrderBookPriceViewType
viewSequence : MarketVolumeStructSequence
<<sequencetypedef>>
OrderBookPriceViewStructSequence
<<sequencetypedef>>
MarketVolumeStructSequence


<<struct>>
MarketDataHistoryEntryStruct
entryType : MarketDataHistoryEntryType
source : Source
reportTime : DateTimeStruct
price : PriceStruct
quantity : long
sellerAcronym : string
buyerAcronym : string
bidSize : long
bidPrice : PriceStruct
askSize : long
askPrice : PriceStruct
underlyingLastSalePrice : PriceStruct
eopType : ExpectedOpeningPriceType
marketCondition : ProductState
optionalData : string
exceptionCode : string
physLocation : string
prefix : string
<<struct>>
MarketDataHistoryStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
startTime : DateTimeStruct
endTime : DateTimeStruct
entries : MarketDataHistoryEntryStructSequence
<<sequencetypedef>>
MarketDataHistoryEntryStructSequence
<<sequencetypedef>>
MarketDataHistoryStructSequence
item
sequence
(from IDL Primitives)


<<struct>>
CurrentMarketStructV2
productKeys : ProductKeysStruct
sessionName : TradingSessionName
exchange : string
currentMarketViews : CurrentMarketViewStructSequence
sentTime : TimeStruct
bidIsMarketBest : boolean
askIsMarketBest : boolean
legalMarket : boolean
<<sequencetypedef>>
CurrentMarketStructV2Sequence
<<sequencetypedef>>
CurrentMarketViewStructSequence
<<struct>>
OrderBookPriceStructV2
price : PriceStruct
views : OrderBookPriceViewStructSequence
<<struct>>
OrderBookPriceViewStruct
orderBookPriceViewType : OrderBookPriceViewType
viewSequence : MarketVolumeStructSequence
<<sequencetypedef>>
OrderBookPriceStructV2Sequence
<<sequencetypedef>>
MarketVolumeStructSequence


<<struct>>
ExchangeIndicatorStruct
marketCondition : ExchangeIndicatorType
exchange : string
<<sequencetypedef>>
ExchangeIndicatorStructSequence


<<struct>>
RecapStructV4
classKey : ClassKey
exchange : string
highPrice : IntegerPrice
lowPrice : IntegerPrice
openPrice : IntegerPrice
previousClosePrice : IntegerPrice
priceScale : octet
productKey : ProductKey
productType : ProductType
sentTime : IntegerTime
statusCodes : string
<<sequencetypedef>>
RecapStructV4Sequence
<<struct>>
CurrentMarketStructV4
askPrice : IntegerPrice
askSizeSequence : MarketVolumeStructSequence
bidPrice : IntegerPrice
bidSizeSequence : MarketVolumeStructSequence
classKey : ClassKey
currentMarketType : CurrentMarketType
exchange : String
marketIndicator : MarketIndicator
priceScale : octet
productKey : ProductKey
productState : ProductState
productType : ProductType
sentTime : IntegerTime
bidTickDirection : TickDirectionType
<<sequencetypedef>>
CurrentMarketStructV4Sequence
<<struct>>
TickerStructV4
classKey : ClassKey
exchange : String
priceScale : octet
productKey : ProductKey
productType : ProductType
salePostfix : string
salePrefix : String
tradePrice : IntegerPrice
tradeTime : IntegerTime
tradeVolume : Long
sentTime : IntegerTime
<<sequencetypedef>>
TickerStructV4Sequence
<<struct>>
LastSaleStructV4
classKey : ClassKey
tickDirection : char
exchange : string
lastSalePrice : IntegerPrice
lastSaleTime : IntegerTime
lastSaleVolume : Long
netPriceChange : IntegerPrice
priceScale : octet
productKey : ProductKey
productType : ProductType
totalVolume : Long
sentTime : IntegerTime
<<sequencetypedef>>
LastSaleStructV4Sequence
<<struct>>
MarketVolumeStructV4
volumeType : VolumeType
quantity : Long
multipleParties : MultiplePartiesIndicator
<<sequencetypedef>>
MarketVolumeStructV4Sequence


<<struct>>
MarketDataDetailStruct
overrideIndicator : OverrideIndicatorType
nbboAskPrice : PriceStruct
nbboAskExchanges : ExchangeVolumeStructSequence
nbboBidPrice : PriceStruct
nbboBidExchanges : ExchangeVolumeStructSequence
tradeThroughIndicator : boolean
exchangeIndicators : ExchangeIndicatorStructSequence
bestPublishedBidPrice : PriceStruct
bestPublishedBidVolume : Long
bestPublishedAskPrice : PriceStruct
bestPublishedAskVolume : long
brokers : ExchangeAcronymStructSequence
contras : ExchangeAcronymStructSequence
extensions : KeyValueStructSequence
<<sequencetypedef>>
MarketDataHistoryDetailStructSequence
<<struct>>
MarketDataHistoryDetailStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
startTime : DateTimeStruct
endTime : DateTimeStruct
isOutOfSequence : boolean
entries : MarketDataHistoryDetailEntryStructSequence
<<struct>>
MarketDataHistoryDetailEntryStruct
historyEntry : MarketDataHistoryEntryStruct
detailData : MarketDataDetailStruct
<<sequencetypedef>>
MarketDataHistoryDetailEntryStructSequence
<<struct>>
MarketDataHistoryEntryStruct
entryType : MarketDataHistoryEntryType
source : Source
reportTime : DateTimeStruct
price : PriceStruct
quantity : long
sellerAcronym : string
buyerAcronym : string
bidSize : long
bidPrice : PriceStruct
askSize : long
askPrice : PriceStruct
underlyingLastSalePrice : PriceStruct
eopType : ExpectedOpeningPriceType
marketCondition : ProductState
optionalData : string
exceptionCode : string
physLocation : string
prefix : string

**Typedefs:**

| | |
|---|---|
| BookDepthUpdateType | Type of book depth update - indicates if updates is to insert, update, or delete a book depth entry. |
| ExpectedOpeningPriceType *is a* short | Indicates if there will be an opening price for a product or if there is a market imbalance. If there is a market imbalance then the expected opening price will indicate if there are more buyers or more sellers.<br><br>See cmiConstants::ExpectedOpeningPriceTypes. |
| IntegerPrice | Indicates the type of price.<br><br>See cmiConstants::PriceConstants |
| MarketChangeReason *is a* short | Reason market data was generated (See cmiConstants::MarketChangeReasons). |
| MarketDataHistoryEntryType *is a* short | Indicates the type of market data history, such as quote, price report, expected opening price, market condition.<br><br>See cmiConstants::MarketDataHistoryEntryTypes. |
| MarketIndicator | Indicates the type of market indicator, such as regular, auto_execution, bid_is_book, etc..<br><br>See cmiConstants::MarketIndicators. |
| VolumeType *is a* short | Type of volume being reported (by contingency type) (See cmiConstants::VolumeTypes). |

**Typedefs for Sequences:**

CurrentMarketStructSequence *is a typedef of a sequence of* CurrentMarketStruct

CurrentMarketStructSequence *is a typedef of a sequence of* CMICurrentMarketConsumer

CurrentMarketStructV2Sequence *is a typedef of a sequence of* CurrentMarketStructV2

CurrentMarketStructV4Sequence *is a typedef of a sequence of* CurrentMarketStructV4

CurrentMarketStructV4Sequence *is a typedef of a sequence of* CMICurrentMarketConsumer

CurrentMarketViewStructSequence *is a typedef of a sequence of* CurrentMarketViewStruct

CurrentMarketViewStructSequence *is a typedef of a sequence of* CurrentMarketStructV2

ExchangeIndicatorStructSequence *is a typedef of a sequence of* ExchangeIndicatorStruct

ExchangeVolumeStructSequence *is a typedef of a sequence of* ExchangeVolumeStruct

ExpectedOpeningPriceStructSequence *is a typedef of a sequence of* ExpectedOpeningPriceStruct

LastSaleStructV4Sequence *is a typedef of a sequence of* LastSaleStructV4

LastSaleStructV4Sequence *is a typedef of a sequence of* CMIRecapConsumer

MarketDataHistoryDetailEntryStructSequence *is a typedef of a sequence of* MarketDataHistoryDetailStruct

MarketDataHistoryEntryStructSequence *is a typedef of a sequence of* MarketDataHistoryEntryStruct

MarketDataHistoryStructSequence *is a typedef of a sequence of* MarketDataHistoryStruct

MarketVolumeStructSequence *is a typedef of a sequence of* OrderBookPriceViewStruct

MarketVolumeStructSequence *is a typedef of a sequence of* CurrentMarketViewStruct

MarketVolumeStructSequence *is a typedef of a sequence of* MarketVolumeStruct

MarketVolumeStructV4Sequence *is a typedef of a sequence of* MarketVolumeStructV4

NBBOStructSequence *is a typedef of a sequence of* NBBOStruct

OrderBookPriceStructSequence *is a typedef of a sequence of* OrderBookPriceStruct

OrderBookPriceStructV2Sequence *is a typedef of a sequence of* OrderBookPriceStructV2

OrderBookPriceViewStructSequence *is a typedef of a sequence of* OrderBookPriceViewStruct

RecapStructSequence *is a typedef of a sequence of* RecapStruct

RecapStructV4Sequence *is a typedef of a sequence of* RecapStruct
RecapStructV4Sequence *is a typedef of a sequence of* CMIRecapConsumer
RecapStructV4Sequence *is a typedef of a sequence of* RecapStructV4

TickerStructSequence *is a typedef of a sequence of* TickerStruct

TickerStructV4Sequence *is a typedef of a sequence of* TickerStruct
TickerStructV4Sequence *is a typedef of a sequence of* CMITickerConsumer
TickerStructV4Sequence *is a typedef of a sequence of* TickerStructV4

---

**Structs:**

**BookDepthStruct**

Provides the prices and volumes for the orders and quotes for a product traded on the CBOE electronic trading system. The volume is broken down by market participant (market maker, customer, broker- dealer) and type of contingency placed upon the orders.

The BookDepthStruct is made up of a buy and sell OrderBookPriceStructSequence. A class diagram for the structs used to report BookDepth is provided at the beginning of this section describing the cmiMarketData module.

| Attribute | Type | Documentation |
|---|---|---|
| allPricesIncluded | Boolean | Indicates if all prices from the order book have been set to true. If false the book depth is truncated due to message size restrictions. |
| buySideSequence | OrderBookPriceStructSequence | Sequence of prices on the buy side of the market. |
| productKeys | ProductKeysStruct | Product Key of the product whose order book update is being reported. |
| sellSideSequence | OrderBookPriceStructSequence | A sequence of prices on the ask side of the market. |
| sessionName | TradingSessionName | Name of the trading session from which this market data originated. |
| transactionSequenceNumber | Long | Sequence number of the book depth update. Used by the client application for message sequencing. |

**BookDepthStructV2**

Provides the prices and volumes for the orders and quotes for a product traded on the CBOE electronic trading system. The volume is broken down by market participant (market maker, customer, broker- dealer) and type of contingency placed upon the orders.

| Attribute | Type | Documentation |
|---|---|---|
| allPricesIncluded | boolean | Indicates if all prices from the order book have been set to true. If false the book depth is truncated due to message size restrictions. |
| buySideSequence | OrderBookPriceStructV2Sequence | Sequence of prices on the buy side of the market. |
| productKeys | ProductKeysStruct | Product Key of the product whose order book update is being reported. |

| Attribute | Type | Documentation |
|---|---|---|
| sellSideSequence | OrderBookPriceStructV2Sequence | A sequence of prices on the ask side of the market. |
| sessionName | TradingSessionName | Name of the trading session from which this market data originated. |
| transactionSequenceNumber | long | Sequence number of the book depth update. Used by the client application for message sequencing. |

**BookDepthUpdatePriceStruct**

Contains a price update for a book entry.

| Attribute | Type | Documentation |
|---|---|---|
| contingencyVolume | Long | Portion of the total volume that represents contingency orders. |
| price | PriceStruct | Price to be updated in the book. |
| totalVolume | Long | Total volume at this price. |
| updateType | BookDepthUpdateType | Type of update. |

**BookDepthUpdateStruct**

Book Depth Update Struct.

| Attribute | Type | Documentation |
|---|---|---|
| buySideChanges | BookDepthUpdatePriceStructSequence | Changes in quantity and/or price in the book on the Buy side. |
| exchange | String | Exchange identifier where the product trades. |
| productKeys | ProductKeysStruct | Identifier of the product for which this book depth is issued. |
| sellSideChanges | BookDepthUpdateStructSequence | Changes in quantity and/or price in the book on the Sell side. |

| Attribute | Type | Documentation |
|---|---|---|
| sequenceNumber | Long | Unique serial number that identifies this book depth update. |
| sessionName | TradingSessionName | Name of the trading session for which the book depth resides. |

**CurrentMarketStruct**

The CurrentMarketStruct is used to return the current market (or top of the book) for a product. The CurrentMarket is generated as a result of a quote or order being received by the system.

| Attribute | Type | Documentation |
|---|---|---|
| askIsMarketBest | boolean | Indicates if the ask price is the market best. |
| askPrice | PriceStruct | Current best Ask Price |
| askSizeSequence | MarketVolumeStructSequence | Sequence of MarketVolumeStructs that contain the order volume broken down by order contingency type at the current market. |
| bidIsMarketBest | boolean | Indicates if current market price is the market best. |
| bidPrice | PriceStruct | Current best Bid Price |
| bidSizeSequence | MarketVolumeStructSequence | Sequence of MarketVolumeStructs that contain the order volume broken down by order contingency type at the current market. |
| exchange | string | Exchange from which the bid originated. |
| legalMarket | boolean | Indicates if the current market bid-ask spread is a legal market per CBOE regulation |
| productKeys | ProductKeysStruct | Product Keys for the product whose current market is being reported |
| sentTime | TimeStruct | Time market information was sent from trading system. |
| sessionName | TradingSessionName | Name of trading session from which the current market message originated |

**CurrentMarketStructV2**

Returns the current market for a product by product key.

| Attribute | Type | Documentation |
|---|---|---|
| askIsMarketBest | boolean | Ask price is the market best. Set to "true" indicates that the ask price is part of the NBBO. Set to "false" indicates that the ask price may not be part of the NBBO. |
| bidIsMarketBest | boolean | The bid price is the market best. Set to "true" indicates that the bid price is part of the NBBO. Set to "false" indicates that the bid price may not be part of the NBBO. |
| currentMarketViews | CurrentMarketViewStructSequence | Sequence for current market views. |
| exchange | string | Name of the exchange. |
| legalMarket | boolean | In a legal market. |
| productKeys | ProductKeysStruct | Product Keys for the product whose current market is being reported. |
| sentTime | TimeStruct | Time the information was sent. |
| sessionName | TradingSessionName | Session name for the product that is being reported. |

**CurrentMarketStructV4**

The MDX current market structure.

| Attribute | Type | Documentation |
|---|---|---|
| askPrice | IntegerPrice | The ask price. |
| askSizeSequence | MarketVolumeStructSequence | Sequence of MarketVolumeStructs that contain the order volume broken down by order contingency type at the current market. |

| Attribute | Type | Documentation |
| --- | --- | --- |
| bidPrice | IntegerPrice | The bid price |
| bidSizeSequence | MarketVolumeStructSequence | Sequence of MarketVolumeStructs that contain the order volume broken down by order contingency type at the current market. |
| bidTickDirection | TickDirectionType | The direction of change for the bid<br>PLUS_TICK '+'<br>Means the current last sale price is higher than the previous last sale price. Also known as an "uptick".<br>MINUS_TICK '-'<br>Means the current last sale price is lower than the previous last sale price. Also known as a "downtick".<br>ZERO_MINUS_TICK '_'<br>Means the current last sale price is the same as the previous last sale price, but is lower than the most recent different last sale price.<br>ZERO_PLUS_TICK '*'<br>Means the current last sale price is the same as the previous last sale price, but is higher than the most recent different last sale price.<br>UNKNOWN_TICK ' 'Means the tick direction field is not used or not known. |
| classKey | ClassKey | Class key for the product whose current market is being reported. |
| currentMarketType | CurrentMarketType | The market type, either best market or best public market. |
| exchange | String | Name of the exchange. |
| marketIndicator | MarketIndicator | The market indicator for this product. |
| priceScale | octet | The number of decimal places to use in conjunction with the integer prices. |
| productKey | ProductKey | Product key for the product whose current market is being reported. |
| productState | ProductState | The product state for this product. |

| Attribute | Type | Documentation |
|---|---|---|
| productType | ProductType | The product type (equity, option, etc.) |
| sentTime | IntegerTime | Time that the information was sent. |

**CurrentMarketViewStruct**

The current market view being reported.

| Attribute | Type | Documentation |
|---|---|---|
| askPrice | PriceStruct | The ask price. |
| askSizeSequence | MarketVolumeStructSequence | A sequence of the ask size. |
| bidPrice | PriceStruct | The bid price. |
| bidSizeSequence | MarketVolumeStructSequence | A sequence of the bid size. |
| currentMarketViewType | CurrentMarketViewType | The type of current market view being reported. |

**ExchangeIndicatorStruct**

Product states of exchanges, other than CBOE.

| Attribute | Type | Documentation |
|---|---|---|
| exchange | string | The name of the exchange. |
| marketCondition | ExchangeIndicatorType | The condition of the market. |

**ExchangeVolumeStruct**

Contains the exchange name and a volume - used in reporting NBBO

| Attribute | Type | Documentation |
|---|---|---|
| exchange | String | Exchange identifier |
| volume | long | Volume in contracts for options or futures, shares for stock |

**ExpectedOpeningPriceStruct**

The Expected Opening Price Message is transmitted when the Broker Service determines an expected opening price.

| Attribute | Type | Documentation |
|---|---|---|
| eopType | ExpectedOpeningPriceType | Expected opening price type. Value will be set to Opening Price, More Buyers, or More Sellers. See the cmiConstants::ExpectedOpeningPriceTypes for more information. |
| expectedOpeningPrice | PriceStruct | Expected Opening Price |
| imbalanceQuantity | long | Quantity of any order imbalance |
| legalMarket | boolean | Used to indicate if expected opening prices are within exchange prescribed bid-ask spread |
| productKeys | ProductKeysStruct | Product whose expected opening price is being reported |
| sessionName | TradingSessionName | Name of the trading session for which the expected opening price of a product was calculated |

**LastSaleStructV4**

The MDX last sale structure.

| Attribute | Type | Documentation |
|---|---|---|

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| classKey | ClassKey | Class key for the product whose current market is being reported. |
| exchange | string | Name of the exchange. |
| lastSalePrice | IntegerPrice | The last sale price integer form. |
| lastSaleTime | IntegerTime | The time of the last sale (milliseconds since midnight). |
| lastSaleVolume | Long | The volume of the last sale. |
| netPriceChange | IntegerPrice | Changes from the previous close price. For CBOE option, it is the change from the first last sale of the day.<br><br>I.E. currentLastSale - yesterdayClosePrice. This is a signed value as well and can therefore be a positive, zero, or negative IntegerPrice value.<br><br>Example: CurrentLastSale: 10.00 YesterdayClose: 10.05 ==> NetChange == - 0.05 (represented as integer -5 with a scale of 2). |
| priceScale | octet | The number of decimal places to use in conjunction with the integer prices. |
| productKey | ProductKey | Product key for the product whose current market is being reported. |
| productType | ProductType | The product type (equity, option, etc.) |
| sentTime | IntegerTime | The time the information was sent |

| Attribute | Type | Documentation |
|---|---|---|
| tickDirection | char | The direction of change in current last sale price relative to the previous last sale.<br>PLUS_TICK '+'<br>Means the current last sale price is higher than the previous last sale price. Also known as an "uptick".<br>MINUS_TICK '-'<br>Means the current last sale price is lower than the previous last sale price. Also known as a "downtick".<br>ZERO_MINUS_TICK '_'<br>Means the current last sale price is the same as the previous last sale price, but is lower than the most recent different last sale price.<br>ZERO_PLUS_TICK '*'<br>Means the current last sale price is the same as the previous last sale price, but is higher than the most recent different last sale price<br>UNKNOWN_TICK ' '<br>Means the tick direction field is not used or not known. |
| totalVolume | Long | The total volume for this product. |

**MarketDataDetailStruct**

Contains market data information and exchange indicator.

| Attribute | Type | Documentation |
|---|---|---|
| bestPublishedAskPrice | PriceStruct | The published market best ask price. |
| bestPublishedAskVolume | long | The market best ask volume that is published.. |
| bestPublishedBidPrice | PriceStruct | The published market best bid price. |

| Attribute | Type | Documentation |
|---|---|---|
| bestPublishedBidVolume | Long | The published market best bid volume. |
| brokers | ExchangeAcronymStructSequence | List of broker acronyms |
| contras | ExchangeAcronymStructSequence | A list of contra acronyms. |
| exchangeIndicators | ExchangeIndicatorStructSequence | List of the product states of other exchanges. |
| extensions | KeyValueStructSequence | Generic keys/values. |
| nbboAskExchanges | ExchangeVolumeStructSequence | Exchanges with the NBBO ask price. |
| nbboAskPrice | PriceStruct | The NBBO ask price. |
| nbboBidExchanges | ExchangeVolumeStructSequence | Exchanges with the NBBO bid. |
| nbboBidPrice | PriceStruct | The NBBO bid price. |
| overrideIndicator | OverrideIndicatorType | A type of market data override used by TPF. |
| tradeThroughIndicator | boolean | Indicates if the last sale violated the NBBO |

### MarketDataHistoryDetailEntryStruct

Contains market data history details.

| Attribute | Type | Documentation |
|---|---|---|
| detailData | MarketDataDetailStruct | Contains market data details. |
| historyEntry | MarketDataHistoryEntryStruct | Market data entry details. |

### MarketDataHistoryDetailStruct

Used for querying market data information.

| Attribute | Type | Documentation |
|---|---|---|
| endTime | DateTimeStruct | The end time for the market data query. |
| entries | MarketDataHistoryDetailEntryStruct Sequence | List of market data entries. |
| isOutOfSequence | boolean | |
| productKeys | ProductKeysStruct | Product keys for the product data that is being reported. |
| sessionName | TradingSessionName | Name of the trading session from which the market data originated. |
| startTime | DateTimeStruct | The start time for the market data query. |

**MarketDataHistoryEntryStruct**

The MarketDataHistoryStruct can contain several different types of market data events, including quotes (sized and unsized), price reports, expected opening prices, and market conditions as indicated in the entryType field.

| Attribute | Type | Documentation |
|---|---|---|
| askPrice | PriceStruct | Quote ask price |
| askSize | long | Quote with size - ask size |
| bidPrice | PriceStruct | Quote bid price |
| bidSize | long | Quote with size -  bid size |
| buyerAcronym | string | Buyer in the case of a price report |
| entryType | MarketDataHistoryEntryType | Type market data history entry. Refer to cmiConstants::MarketDataHistoryEntryTypes |
| eopType | ExpectedOpeningPriceType | If an expected opening price - the type of expected opening price. See cmiConstants::ExpectedOpeningPriceTypes. |
| exceptionCode | string | Exception Code |

| Attribute | Type | Documentation |
|---|---|---|
| marketCondition | ProductState | For market condition entries, this is the type of market condition that was reported. see cmiConstants::ProductStates. |
| optionalData | string | Additional text message.<br>Firms that want to give one DPM priority in participating in a trade use this field using P:EXCH.FIRM; or P:FIRM; EXCH is the exchange acronym. FIRM is the Firm acronym. EXCH is optional.<br>For Sweep and AIM: Enter A:AIS in the primary order to instruct CBOE to sweep all better priced protected quotes at away exchanges at the same time as the commencement of the AIM auction.<br>For AIM Auctions: Enter A:AIM as the first characters. If a Firm does not wish to cancel the primary order when the auction expires, the Firm must enter A:AIR, instead of A:AIM, in this field. This will designate the primary order to be returned to the system and trade or book as a regular order.<br>Options:<br>For Customer orders, the first four characters of this field get reported to the last four characters of the CBOE Trade Match Optional Data field. These four characters get reported to the OCC. |
| physLocation | string | Physical location |
| prefix | string | Information prepended on the market data history message |
| price | PriceStruct | Last sales price - if EntryType is PriceReport. |
| quantity | long | Quantity - if this is a quote entry with size or a price report. |
| reportTime | DateTimeStruct | Time market data event was originally reported |
| sellerAcronym | string | Seller in the case of a price report |
| source | Source | Market source see cmiConstants::Sources |

| Attribute | Type | Documentation |
|---|---|---|
| underlyingLastSalePrice | PriceStruct | In the case of an expected opening price - the last sales price for the underlying |

**MarketDataHistoryStruct**

Used to bundle and return of sequence of market data history entries (MarketDataHistoryEntryStruct) for a product for a requested time interval. The time interval is specified as [startTime,endTime].

This data is obtain by invoking the cmi::MarketQuery.getMarketDataHistoryByTime().

| Attribute | Type | Documentation |
|---|---|---|
| endTime | DateTimeStruct | End time of the last MarketDataHistoryEntryStruct being returned. |
| entries | MarketDataHistoryEntryStructSequence | Sequence of MarketDataHistoryEntryStructs for the product for the specified time interval being returned. |
| productKeys | ProductKeysStruct | Product keys for the product whose market data history is being returned. |
| sessionName | TradingSessionName | Name of the trading session from which the set of market data history records was generated |
| startTime | DateTimeStruct | Start time of the first MarketDataHistoryEntryStruct returned. |

**MarketVolumeStruct**

The MarketVolumeStruct is used to store a volume for a specific order contingency type

| Attribute | Type | Documentation |
|---|---|---|
| multipleParties | boolean | Indicates if orders from multiple parties make up this volume |

| Attribute | Type | Documentation |
|---|---|---|
| quantity | long | Quantity for this type of contingent order. For options the quantity is expressed as contracts. For stocks the quantity is expressed as shares. |
| volumeType | VolumeType | Type of volume being reported (by contingency type). See cmiConstants::VolumeTypes. |

### MarketVolumeStructV4

The MarketVolumeStructV4 is used to store a volume for a specific order contingency type

| Attribute | Type | Documentation |
|---|---|---|
| multipleParties | MultiplePartiesIndicator | Indicates if orders from multiple parties make up this volume |
| quantity | Long | Quantity for this type of contingent order. For options the quantity is expressed as contracts. For stocks the quantity is expressed as shares. |
| volumeType | VolumeType | Type of volume being reported (by contingency type). See cmiConstants::VolumeTypes. |

### NBBOStruct

National Best Bid and Offer

| Attribute | Type | Documentation |
|---|---|---|
| askExchangeVolume | ExchangeVolumeStructSequence | A series of exchange names and volumes |
| askPrice | PriceStruct | National best ask price |
| bidExchangeVolume | ExchangeVolumeStructSequence | A series of exchange names and volumes |

| Attribute | Type | Documentation |
|---|---|---|
| bidPrice | PriceStruct | National best bid price |
| productKeys | ProductKeysStruct | Products being reported |
| sentTime | TimeStruct | Time being reported |
| sessionName | TradingSessionName | Name of trading session from which the NBBO message originated |

**OrderBookPriceStruct**

A price within the order book and the volume at the price broken down by market participant and order contingency type. A component of the OrderBookSideStruct - which in turn is a component of the BookDepthStruct.

| Attribute | Type | Documentation |
|---|---|---|
| contingencyVolume | long | Contingent order volume at this price for only AON and FOK volume contingencies. |
| price | PriceStruct | Price in the order book. |
| totalVolume | long | The total volume at this price. |

**OrderBookPriceStructV2**

A price within the order book.

| Attribute | Type | Documentation |
|---|---|---|
| price | PriceStruct | Order book price. |
| views | OrderBookPriceViewStructSequence | A sequence of order book price views. |

### OrderBookPriceViewStruct

A price within the order book.

| Attribute | Type | Documentation |
|---|---|---|
| orderBookPriceViewType | OrderBookPriceViewType | A type of order book price view. |
| viewSequence | MarketVolumeStructSequence | A sequence of views. |

### RecapStruct

Recap provides detailed information on price and volume for a product Information is reported with ProductNameStruct and ProductKeys.

Usually the recap information subscribed to is for underlyings of products being traded. This is useful for automated quoting systems.

| Attribute | Type | Documentation |
|---|---|---|
| askPrice | PriceStruct | Ask Price |
| askSize | long | Ask Size |
| askTime | TimeStruct | Time of the last Ask Price |
| bidDirection | char | Indicates if the bid is greater or less than the previous bid '+' indicates bid is greater than previous, '-' indicates bid is lower than previous. |
| bidPrice | PriceStruct | Bid Price |
| bidSize | long | Bid Size |
| bidTime | TimeStruct | Time of the last bid |
| closePrice | PriceStruct | Closing Price |
| highPrice | PriceStruct | Daily High Price |
| isOTC | boolean | True if it's an Over The Counter |

| Attribute | Type | Documentation |
|---|---|---|
| lastSalePrice | PriceStruct | Last Sales Price |
| lastSaleVolume | long | Last Sales Volume |
| lowPrice | PriceStruct | Daily Low Price |
| netChange | PriceStruct | Change from the previous sale. |
| netChangeDirection | char | The net change is used to indicate how the price has changed with respect to the previous closing price. '+' indicates the price has increased since previous close and '-' indicates the price has decreased since previous close. |
| openInterest | long | Open Interest is not supported by CBOEdirect or OneChicago and can be obtained from the Options Clearing Corporation (OCC) or CBOE Financial Network (CFN). |
| openPrice | PriceStruct | Opening Price |
| previousClosePrice | PriceStruct | Previous Closing Price |
| productInformation | ProductNameStruct | Product name for the product whose recap is being reported |
| productKeys | ProductKeysStruct | Product keys for the product whose recap is being reported |
| recapPrefix | string | Prefix information provided from external market data sources |
| sessionName | TradingSessionName | Name of trading session from which the market recap originated |
| tick | PriceStruct | Tick size for product |
| tickDirection | char | Direction from the previous sale. '+' indicates if this price has increased, '-' indicates the price has decreased. |
| totalVolume | long | Daily Total Volume |
| tradeTime | TimeStruct | Time of the last trade |

**RecapStructV4**

The MDX recap structure.

| Attribute | Type | Documentation |
|---|---|---|
| classKey | ClassKey | Class key for the product whose current market is being reported. |
| exchange | string | Name of the exchange. |
| highPrice | IntegerPrice | Today's high price. |
| lowPrice | IntegerPrice | Today's low price. |
| openPrice | IntegerPrice | Today's opening price. |
| previousClosePrice | IntegerPrice | Previous closing price. |
| priceScale | octet | The number of decimal places to use in conjunction with the integer prices. |
| productKey | ProductKey | Product key for the product whose current market is being reported. |
| productType | ProductType | The product type (equity, option, etc.) |
| sentTime | IntegerTime | The time the information was sent. |
| statusCodes | string | Currently only used for equities, not options. Indicates whether there is Dow Jones News, Reuters News, whether the stock is Ex-Dividend, and whether it's opened or closed. |

**TickerStruct**

Last Sales information reported with ProductKeys for the product. Ticker data is provided for all products traded on the exchange and the underlyings. The ticker must be subscribed for by product.

| Attribute | Type | Documentation |
|---|---|---|
| exchangeSymbol | Symbol | Product Symbol as reported by the exchange that is the source of the ticker data |
| lastSalePrice | PriceStruct | Last Sales Price |
| lastSaleVolume | long | Last Sales Volume |

| Attribute | Type | Documentation |
|---|---|---|
| productKeys | ProductKeysStruct | Product keys for the product whose ticker data is being reported |
| salePostfix | string | Suffix information that is appended to the end of the ticker information by the reporting market |
| salePrefix | string | Prefix information provided by the market reporting the sale.<br><br>OPNT (Opening Trade) is used if the first trade is part of the opening rotation. FTAO (First Trade After Opening) is used if the first trade occurs after the opening. CNCO (Cancel Opening) is a legitimate code for OPRA and is issued for a bust that affects the opening trade. |
| sessionName | TradingSessionName | Name of the trading session from which this last sale data originated |

### TickerStructV4

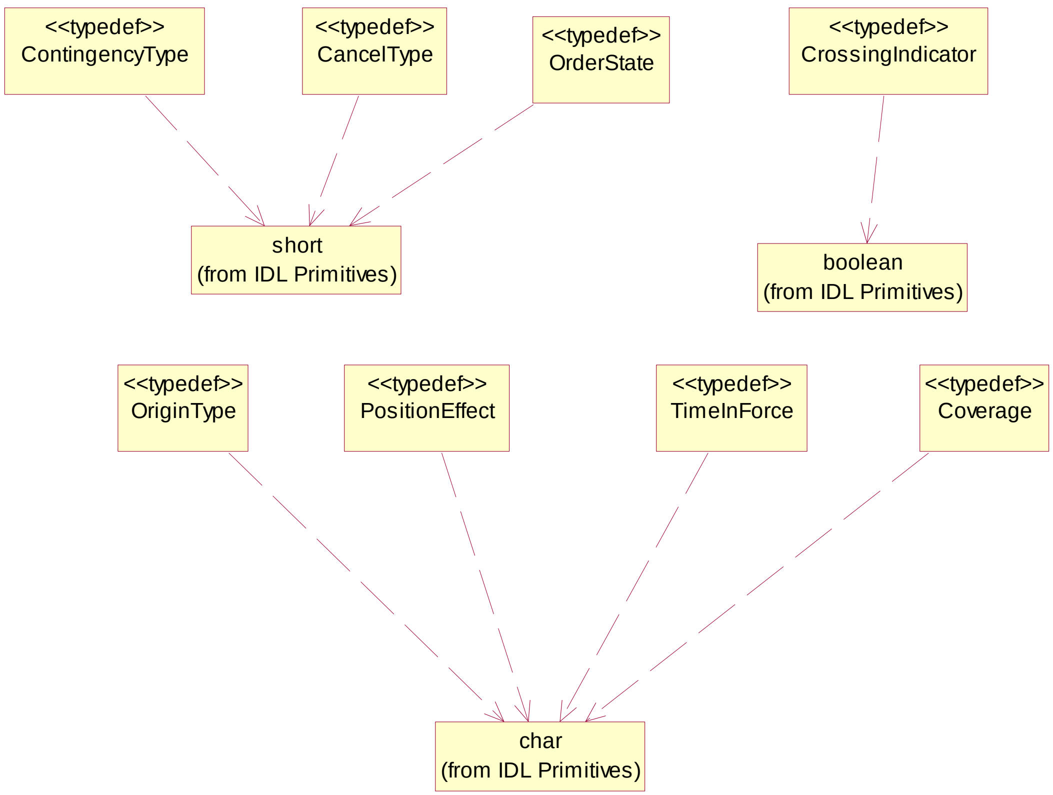
The MDX ticker structure.

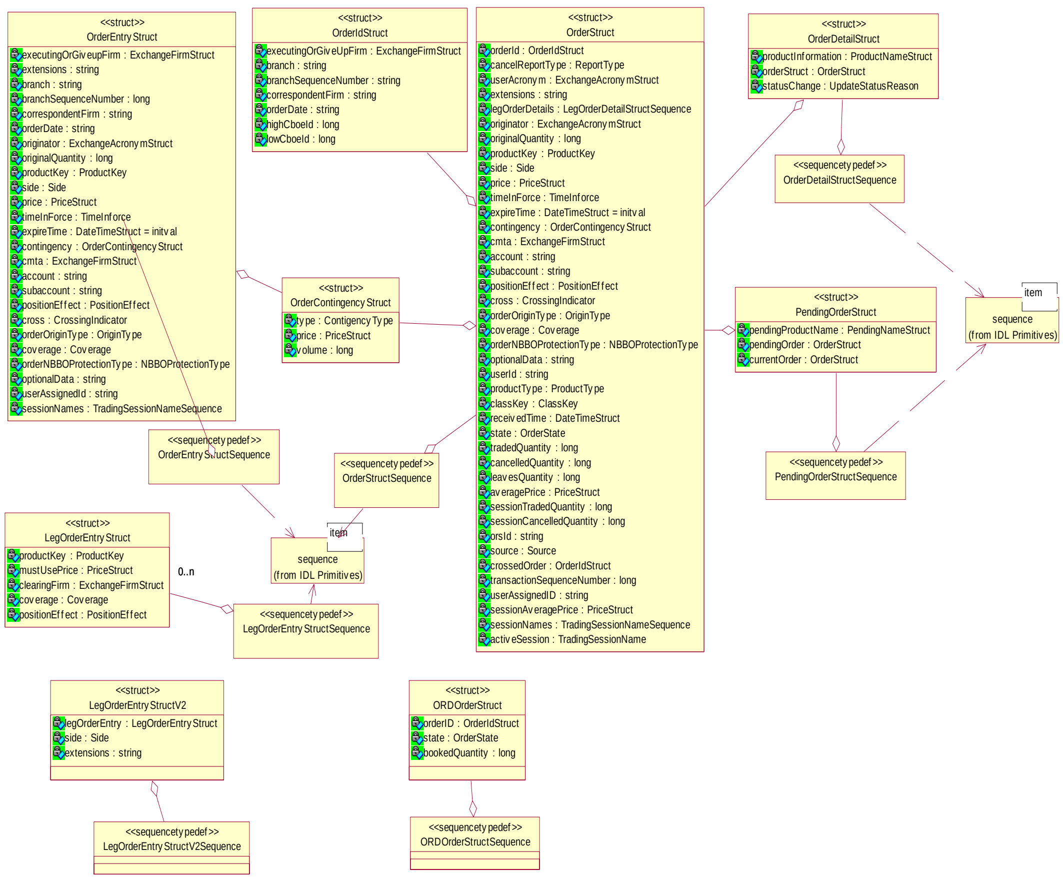
| Attribute | Type | Documentation |
|---|---|---|
| classKey | ClassKey | Class key for the product whose current market is being reported. |
| exchange | String | Name of the exchange. |
| priceScale | octet | The number of decimal places to use in conjunction with the integer prices. |
| productKey | ProductKey | Product key for the product whose current market is being reported. |
| productType | ProductType | The product type (equity, option, etc.) |
| salePostfix | string | Suffix information that is appended to the end of the ticker information by the reporting market. See the api-08, the Market Data Express document, for suffix definitions. |

| Attribute | Type | Documentation |
|---|---|---|
| salePrefix | String | Prefix information provided by the market reporting the sale. A dot '.' will be used in front of the prefixes-- .OPNT (Opening Trade) is used if the first trade is part of the opening rotation. .FTAO (First Trade After Opening) is used if the first trade occurs after the opening. .CNCO (Cancel Opening) is a legitimate code for OPRA and is issued for a bust that affectes the opening trade. |
| sentTime | IntegerTime | The time the information was sent. |
| tradePrice | IntegerPrice | The trade price. |
| tradeTime | IntegerTime | The time of the trade (milliseconds since midnight). |
| tradeVolume | Long | The volume for this trade. |

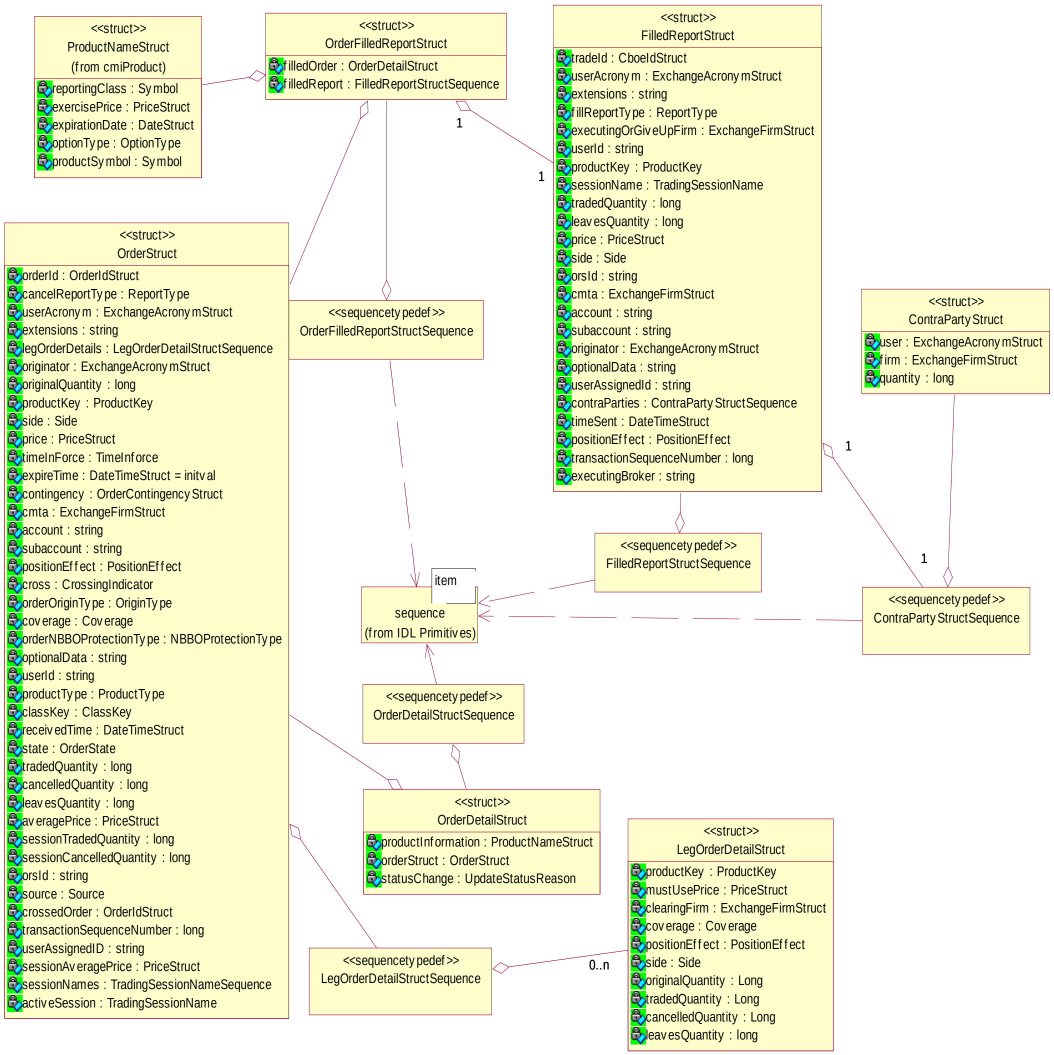
## *Module cmiOrder*

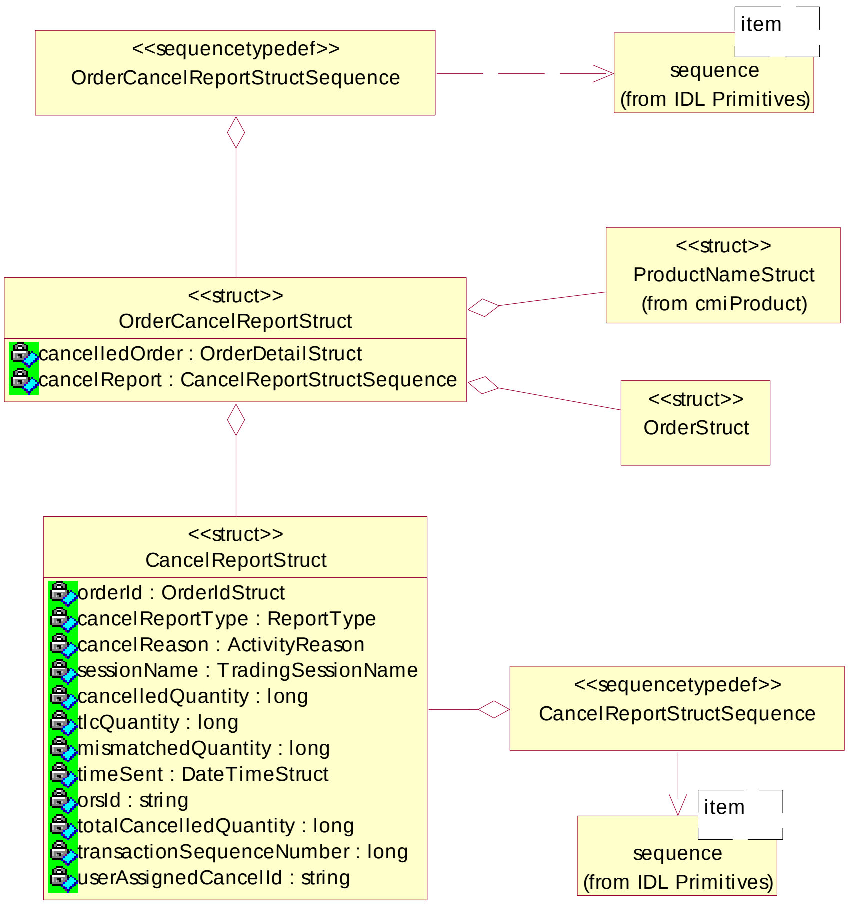
The cmiOrder module contains messages for order routing. Message are available for submission of orders, cancel requests, request for quotes, and reports on order activity.

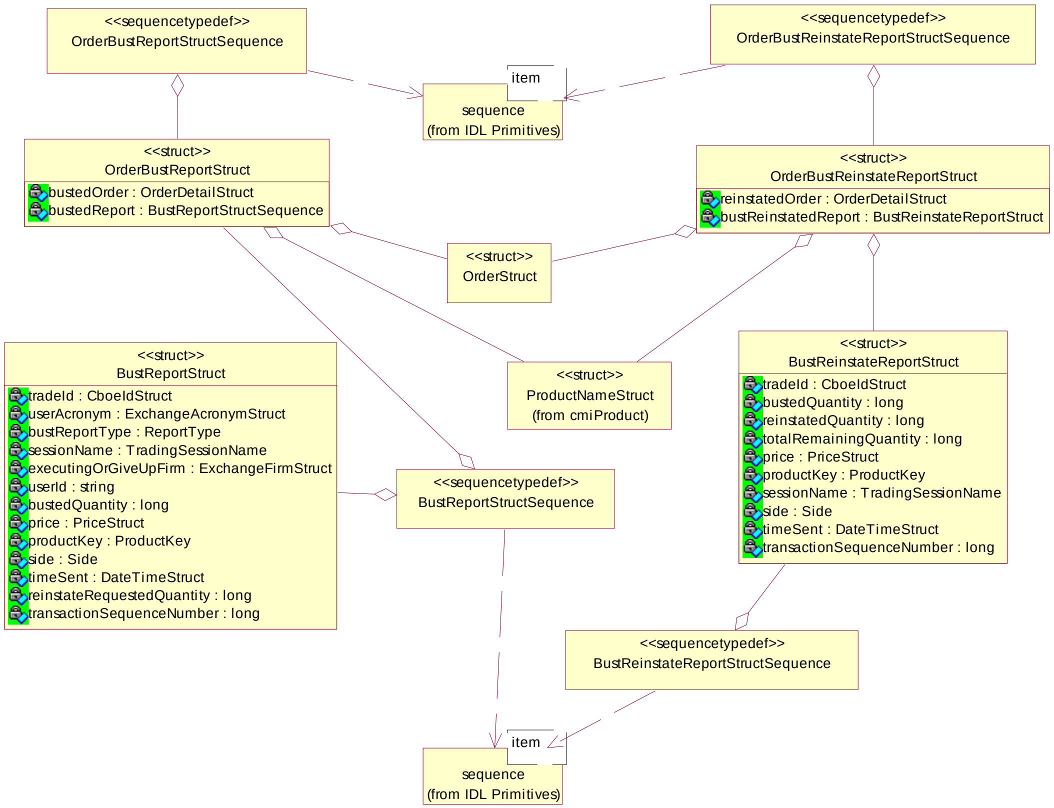


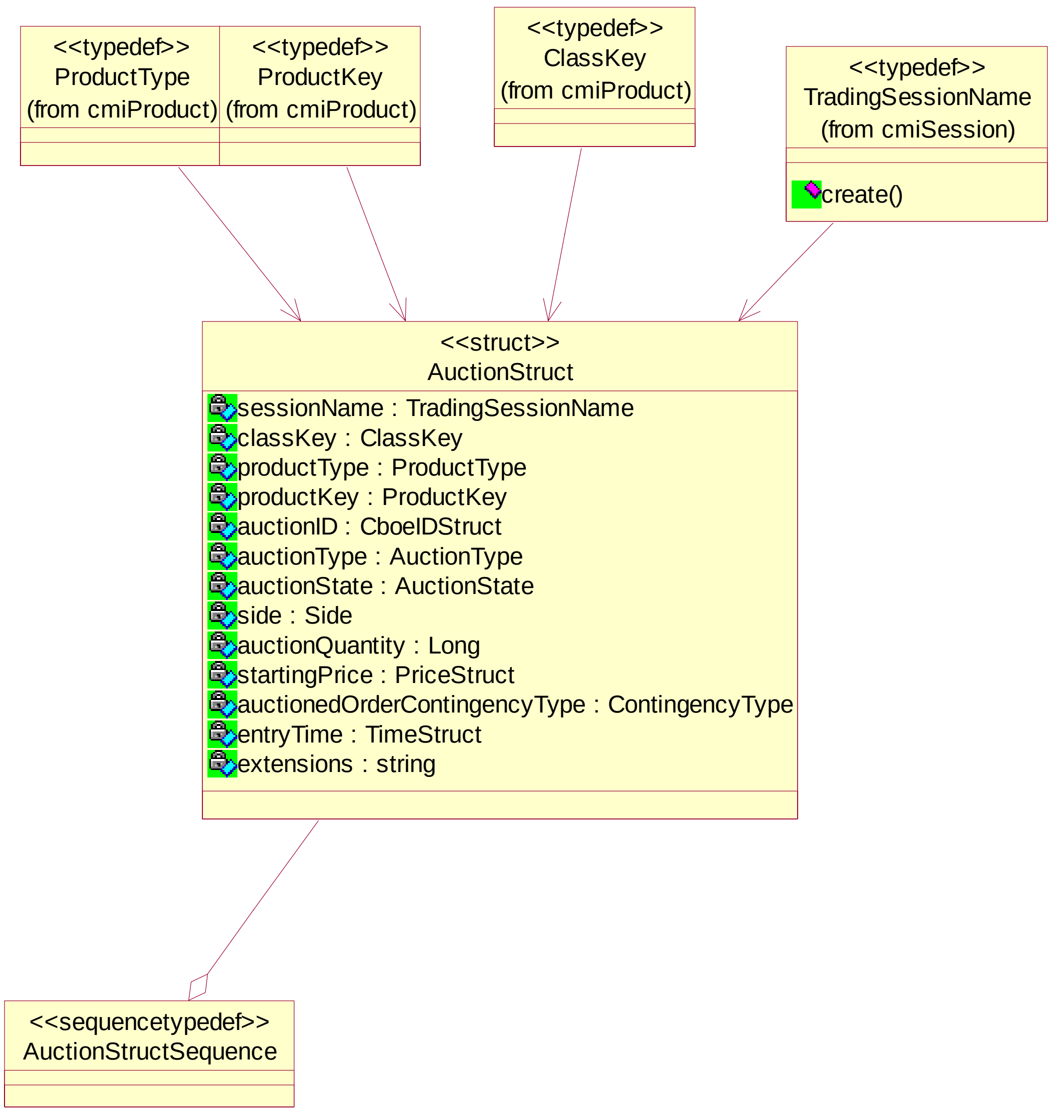


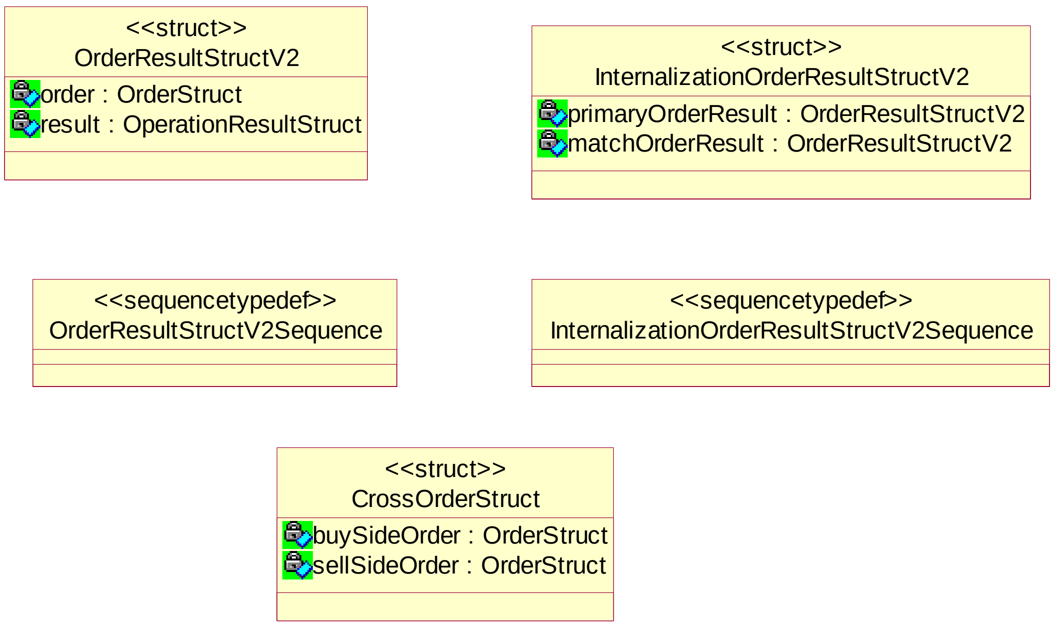
<<struct>>
OrderEntryStruct
executingOrGiveupFirm : ExchangeFirmStruct
extensions : string
branch : string
branchSequenceNumber : long
correspondentFirm : string
orderDate : string
originator : ExchangeAcronymStruct
originalQuantity : long
productKey : ProductKey
side : Side
price : PriceStruct
timeInForce : TimeInforce
expireTime : DateTimeStruct = initval
contingency : OrderContingencyStruct
cmta : ExchangeFirmStruct
account : string
subaccount : string
positionEffect : PositionEffect
cross : CrossingIndicator
orderOriginType : OriginType
coverage : Coverage
orderNBBOProtectionType : NBBOProtectionType
optionalData : string
userAssignedId : string
sessionNames : TradingSessionNameSequence
<<struct>>
OrderIdStruct
executingOrGiveUpFirm : ExchangeFirmStruct
branch : string
branchSequenceNumber : string
correspondentFirm : string
orderDate : string
highCboeId : long
lowCboeId : long
<<struct>>
OrderStruct
orderId : OrderIdStruct
cancelReportType : ReportType
userAcronym : ExchangeAcronymStruct
extensions : string
legOrderDetails : LegOrderDetailStructSequence
originator : ExchangeAcronymStruct
originalQuantity : long
productKey : ProductKey
side : Side
price : PriceStruct
timeInForce : TimeInforce
expireTime : DateTimeStruct = initval
contingency : OrderContingencyStruct
cmta : ExchangeFirmStruct
account : string
subaccount : string
positionEffect : PositionEffect
cross : CrossingIndicator
orderOriginType : OriginType
coverage : Coverage
orderNBBOProtectionType : NBBOProtectionType
optionalData : string
userId : string
productType : ProductType
classKey : ClassKey
receivedTime : DateTimeStruct
state : OrderState
tradedQuantity : long
cancelledQuantity : long
leavesQuantity : long
averagePrice : PriceStruct
sessionTradedQuantity : long
sessionCancelledQuantity : long
orsId : string
source : Source
crossedOrder : OrderIdStruct
transactionSequenceNumber : long
userAssignedID : string
sessionAveragePrice : PriceStruct
sessionNames : TradingSessionNameSequence
activeSession : TradingSessionName
<<struct>>
OrderDetailStruct
productInformation : ProductNameStruct
orderStruct : OrderStruct
statusChange : UpdateStatusReason
<<sequencetypedef>>
OrderDetailStructSequence
<<struct>>
OrderContingencyStruct
type : ContigencyType
price : PriceStruct
volume : long
<<struct>>
PendingOrderStruct
pendingProductName : PendingNameStruct
pendingOrder : OrderStruct
currentOrder : OrderStruct
item
sequence
(from IDL Primitives)
<<sequencetypedef>>
OrderEntryStructSequence
<<sequencetypedef>>
OrderStructSequence
<<sequencetypedef>>
PendingOrderStructSequence
<<struct>>
LegOrderEntryStruct
productKey : ProductKey
mustUsePrice : PriceStruct
clearingFirm : ExchangeFirmStruct
coverage : Coverage
positionEffect : PositionEffect
0..n
item
sequence
(from IDL Primitives)
<<sequencetypedef>>
LegOrderEntryStructSequence
<<struct>>
LegOrderEntryStructV2
legOrderEntry : LegOrderEntryStruct
side : Side
extensions : string
<<struct>>
ORDOrderStruct
orderID : OrderIdStruct
state : OrderState
bookedQuantity : long
<<sequencetypedef>>
LegOrderEntryStructV2Sequence
<<sequencetypedef>>
ORDOrderStructSequence


<<struct>>
ProductNameStruct
(from cmiProduct)
reportingClass : Symbol
exercisePrice : PriceStruct
expirationDate : DateStruct
optionType : OptionType
productSymbol : Symbol
<<struct>>
OrderFilledReportStruct
filledOrder : OrderDetailStruct
filledReport : FilledReportStructSequence
<<struct>>
FilledReportStruct
tradeId : CboeIdStruct
userAcronym : ExchangeAcronymStruct
extensions : string
fillReportType : ReportType
executingOrGiveUpFirm : ExchangeFirmStruct
userId : string
productKey : ProductKey
sessionName : TradingSessionName
tradedQuantity : long
leavesQuantity : long
price : PriceStruct
side : Side
orsId : string
cmta : ExchangeFirmStruct
account : string
subaccount : string
originator : ExchangeAcronymStruct
optionalData : string
userAssignedId : string
contraParties : ContraPartyStructSequence
timeSent : DateTimeStruct
positionEffect : PositionEffect
transactionSequenceNumber : long
executingBroker : string
<<struct>>
OrderStruct
orderId : OrderIdStruct
cancelReportType : ReportType
userAcronym : ExchangeAcronymStruct
extensions : string
legOrderDetails : LegOrderDetailStructSequence
originator : ExchangeAcronymStruct
originalQuantity : long
productKey : ProductKey
side : Side
price : PriceStruct
timeInForce : TimeInforce
expireTime : DateTimeStruct = initval
contingency : OrderContingencyStruct
cmta : ExchangeFirmStruct
account : string
subaccount : string
positionEffect : PositionEffect
cross : CrossingIndicator
orderOriginType : OriginType
coverage : Coverage
orderNBBOProtectionType : NBBOProtectionType
optionalData : string
userId : string
productType : ProductType
classKey : ClassKey
receivedTime : DateTimeStruct
state : OrderState
tradedQuantity : long
cancelledQuantity : long
leavesQuantity : long
averagePrice : PriceStruct
sessionTradedQuantity : long
sessionCancelledQuantity : long
orsId : string
source : Source
crossedOrder : OrderIdStruct
transactionSequenceNumber : long
userAssignedID : string
sessionAveragePrice : PriceStruct
sessionNames : TradingSessionNameSequence
activeSession : TradingSessionName
<<sequencetypedef>>
OrderFilledReportStructSequence
<<struct>>
ContraPartyStruct
user : ExchangeAcronymStruct
firm : ExchangeFirmStruct
quantity : long
<<sequencetypedef>>
FilledReportStructSequence
item
sequence
(from IDL Primitives)
<<sequencetypedef>>
ContraPartyStructSequence
<<sequencetypedef>>
OrderDetailStructSequence
<<struct>>
OrderDetailStruct
productInformation : ProductNameStruct
orderStruct : OrderStruct
statusChange : UpdateStatusReason
<<struct>>
LegOrderDetailStruct
productKey : ProductKey
mustUsePrice : PriceStruct
clearingFirm : ExchangeFirmStruct
coverage : Coverage
positionEffect : PositionEffect
side : Side
originalQuantity : Long
tradedQuantity : Long
cancelledQuantity : Long
leavesQuantity : long
<<sequencetypedef>>
LegOrderDetailStructSequence
1
1
1
1
0..n


<<sequencetypedef>>
OrderCancelReportStructSequence
item
sequence
(from IDL Primitives)
<<struct>>
ProductNameStruct
(from cmiProduct)
<<struct>>
OrderCancelReportStruct
cancelledOrder : OrderDetailStruct
cancelReport : CancelReportStructSequence
<<struct>>
OrderStruct
<<struct>>
CancelReportStruct
orderId : OrderIdStruct
cancelReportType : ReportType
cancelReason : ActivityReason
sessionName : TradingSessionName
cancelledQuantity : long
tlcQuantity : long
mismatchedQuantity : long
timeSent : DateTimeStruct
orsId : string
totalCancelledQuantity : long
transactionSequenceNumber : long
userAssignedCancelId : string
<<sequencetypedef>>
CancelReportStructSequence
item
sequence
(from IDL Primitives)


<<sequencetypedef>>
OrderBustReportStructSequence
item
sequence
(from IDL Primitives)
<<sequencetypedef>>
OrderBustReinstateReportStructSequence
<<struct>>
OrderBustReportStruct
bustedOrder : OrderDetailStruct
bustedReport : BustReportStructSequence
<<struct>>
OrderBustReinstateReportStruct
reinstatedOrder : OrderDetailStruct
bustReinstatedReport : BustReinstateReportStruct
<<struct>>
OrderStruct
<<struct>>
BustReportStruct
tradeId : CboeIdStruct
userAcronym : ExchangeAcronymStruct
bustReportType : ReportType
sessionName : TradingSessionName
executingOrGiveUpFirm : ExchangeFirmStruct
userId : string
bustedQuantity : long
price : PriceStruct
productKey : ProductKey
side : Side
timeSent : DateTimeStruct
reinstateRequestedQuantity : long
transactionSequenceNumber : long
<<struct>>
ProductNameStruct
(from cmiProduct)
<<struct>>
BustReinstateReportStruct
tradeId : CboeIdStruct
bustedQuantity : long
reinstatedQuantity : long
totalRemainingQuantity : long
price : PriceStruct
productKey : ProductKey
sessionName : TradingSessionName
side : Side
timeSent : DateTimeStruct
transactionSequenceNumber : long
<<sequencetypedef>>
BustReportStructSequence
<<sequencetypedef>>
BustReinstateReportStructSequence
item
sequence
(from IDL Primitives)


<<typedef>>
ProductType
(from cmiProduct)
<<typedef>>
ProductKey
(from cmiProduct)
<<typedef>>
ClassKey
(from cmiProduct)
<<typedef>>
TradingSessionName
(from cmiSession)
create()
<<struct>>
AuctionStruct
sessionName : TradingSessionName
classKey : ClassKey
productType : ProductType
productKey : ProductKey
auctionID : CboeIDStruct
auctionType : AuctionType
auctionState : AuctionState
side : Side
auctionQuantity : Long
startingPrice : PriceStruct
auctionedOrderContingencyType : ContingencyType
entryTime : TimeStruct
extensions : string
<<sequencetypedef>>
AuctionStructSequence




**Typedefs:**

| | |
|---|---|
| AuctionState *is* a typedef | The state of the auction (i.e. started). See cmiConstants::AuctionState |

| | |
|---|---|
| AuctionType *is* a typedef | The auction type. See cmiConstants::AuctionType |
| CancelType *is a* short | Type of cancellation action requested. Refers to how to treat the cancel quantity on a cancel request (See cmiConstants::CancelType). |
| ContingencyType *is a* short | Order type contingency (Fill or kill, all or none, market on close, etc.) (See cmiConstants::ContingencyTypes) |
| Coverage *is a* char | Specified type of coverage for an order. Can either be covered, uncovered, unspecified. See cmiConstants::CoverageTypes. |
| CrossingIndicator *is a* boolean | Indicates if order is a part of a crossing order |
| MatchType *is* a typedef | Used to enter the price for internalization orders. The price can be auto matched, fixed limit price match or guaranteed starting price. Currently, limit price and auto match are the only two MatchTypes supported. |
| NBBOProtectionType *is* a typedef | This field determines if the order receives NBBO protection. The order has to be a customer order for this field to matter. |
| OrderState *is a* short | State of the order (Booked, canceled, etc.) (See cmiConstants::OrderStates |
| OriginType *is a* char | Type of originator of the order - 'C'-Customer, 'F'-Firm, 'M'-Market Maker, 'B'-Broker (See cmiConstants::OrderOrigins). |
| PositionEffect *is a* char | Position Effect of the order (Opening or Closing) (See cmiConstants::PositionEffects) |
| TimeInForce *is a* char | Time order is in force (Day, Good until cancelled, Good until date) (See cmiConstants::TimesInForce) |

**Typedefs for Sequences:**

AuctionStructSequence *is a typedef of a sequence of* AuctionStruct

BustReinstateReportStructSequence *is a typedef of a sequence of* BustReinstateReportStruct

BustReportStructSequence *is a typedef of a sequence of* BustReportStruct

BustReportStructSequence *is a typedef of a sequence of* OrderBustReportStruct

CancelReportStructSequence *is a typedef of a sequence of* CancelReportStruct

ContraPartyStructSequence *is a typedef of a sequence of* ContraPartyStruct

FilledReportStructSequence *is a typedef of a sequence of* FilledReportStruct

LegOrderDetailStructSequence *is a typedef of a sequence of* LegOrderDetailStruct

LegOrderEntryStructSequence *is a typedef of a sequence of* LegOrderEntryStruct

OrderBustReinstateReportStructSequence *is a typedef of a sequence of* OrderBustReinstateReportStruct

OrderBustReportStructSequence *is a typedef of a sequence of* OrderBustReportStruct

OrderCancelReportStructSequence *is a typedef of a sequence of* OrderCancelReportStruct

OrderDetailStructSequence *is a typedef of a sequence of* OrderDetailStruct

OrderEntryStructSequence *is a typedef of a sequence of* OrderEntryStruct

OrderFilledReportStructSequence *is a typedef of a sequence of* OrderFilledReportStruct

OrderStructSequence *is a typedef of a sequence of* OrderStruct

ORDOrderStructSequence *is a typedef of a sequence of* ORDOrderStruct

PendingOrderStructSequence *is a typedef of a sequence of* PendingOrderStruct

---

**Structs:**

### AuctionStruct

Contains auction information.

| Attribute | Type | Documentation |
|---|---|---|
| auctionedOrderContingencyType | ContingencyType | The contingency type of the auctioned order. |
| auctionID | CboeIDStruct | Unique identifier of the auction. |
| auctionQuantity | Long | The order quantity in the auction. |
| auctionState | AuctionState | The state (i.e. Started) of the auction. |
| auctionType | AuctionType | Type of auction supported by CBOEdirect. |
| classKey | ClassKey | A unique number that is used to identify a product class. |
| entryTime | TimeStruct | The time the order entered the auction. |
| extensions | string | Used to persist arbitrary data along with the auction. |
| productKey | ProductKey | Unique identifier for a product in the system. Applies to all products, including underlying products, not just products traded on the system |
| productType | ProductType | Type of product (debt, equity, option, etc.) (See cmiConstants::ProductTypes). |
| sessionName | TradingSessionName | Name of the trading session where the user permits the order to be traded. |
| side | Side | Either Bid or Ask (see cmiConstants::Sides). |
| startingPrice | PriceStruct | The auction start price. |

### AuctionSubscribeResultStruct

Contains the auction type and subscription result.

| Attribute | Type | Documentation |
|---|---|---|
| auctionType | AuctionType | The type of auction subscription. |

| Attribute | Type | Documentation |
|---|---|---|
| subscriptionResult | OperationResultStruct | The result of the subscription. See cmiUtil. |

### BustReinstateReportStruct

Details of a bust trade reinstatement - which represents the event when a trade was busted by trading operations and the quantity of the trade was able to be reinstated into the order book.

| Attribute | Type | Documentation |
|---|---|---|
| bustedQuantity | long | The busted trade quantity |
| price | PriceStruct | Trade price at which the bust trade was restated |
| productKey | ProductKey | Unique identifier of the product for which a busted trade was reinstated. |
| reinstatedQuantity | long | Quantity that was reinstated |
| sessionName | TradingSessionName | Name of the trading session in which the busted trade was reinstated. |
| side | Side | Side of the trade for this user |
| timeSent | DateTimeStruct | Timestamp when the bust reinstate report was issued |
| totalRemainingQuantity | long | The total remaining quantity after the bust trade was reinstated |
| tradeId | CboeIdStruct | Trade Id that was reinstated |
| transactionSequenceNumber | long | Transaction sequence number for the bust trade reinstatement report |

### BustReportStruct

Report for a trade that was busted by trading operations.

| Attribute | Type | Documentation |
|---|---|---|
| bustedQuantity | long | Quantity of the trade that was busted |
| bustReportType | ReportType | Type of trade bust report. |
| executingOrGiveUpFirm | ExchangeFirmStruct | Executing or Give Up Firm associated with the trade |
| price | PriceStruct | Price of the busted trade |
| productKey | ProductKey | Product Key of the product of the trade that was busted |
| reinstateRequestedQuantity | long | Quantity of the busted trade that the user wants to reinstate |
| sessionName | TradingSessionName | Name of the trading session in which the trade was busted. |
| side | Side | Side of the trade that was busted |
| timeSent | DateTimeStruct | Time the bust report was sent |
| tradeId | CboeIdStruct | Trade identifier |
| transactionSequenceNumber | long | Unique transaction number for this report |
| userAcronym | ExchangeAcronymStruct | The user acronym for the trade bust. |
| userId | string | User associated with the trade |

### CancelReportStruct

Order Cancel Report sent in response to a Cancel or CancelReplace request.

| Attribute | Type | Documentation |
|---|---|---|
| cancelledQuantity | long | Cancelled quantity |
| cancelReason | ActivityReason | Sets a reason for a cancel report. |
| cancelReportType | ReportType | Identifies the type of cancel report. |

| Attribute | Type | Documentation |
|---|---|---|
| mismatchedQuantity | long | Quantity difference between what the client thought the remaining quantity was at the time of the cancel request and the actual remaining quantity when the cancel request was received by the trading system |
| orderId | OrderIdStruct | Identifier for the cancelled order |
| orsId | string | ORS System ID for the order. |
| sessionName | TradingSessionName | Name of the trading session in which an order was fully or partially canceled. |
| timeSent | DateTimeStruct | Time cancel report was sent by system to user. |
| tlcQuantity | long | Quantity that was too late to cancel - most likely due to a fill that took place prior to the cancel request being received by the system |
| totalCancelledQuantity | long | Total quantity cancelled on order |
| transactionSequenceNumber | long | Transaction number assigned to the message to guarantee uniqueness and ordering between CAS and Exchange Services. Can be ignored by client application, as the CAS guarantees ordering. |
| userAssignedCancelId | string | The user assigned id that was entered by the user on the cancel request. |

### CancelRequestStruct

Used to specify type quantity cancelled for an existing order

| Attribute | Type | Documentation |
|---|---|---|
| cancelReason | ActivityReason | Reason for the cancel request |
| cancelType | CancelType | The type of cancellation requested. Specify to cancel all, cancel a specific quantity or specify the remaining quantity desired after the cancel. |

| Attribute | Type | Documentation |
|---|---|---|
| orderId | OrderIdStruct | Identifier of order to be cancelled. |
| quantity | long | Quanity to be cancelled or remaining quantity based upon the cancel type (See cmiConstants::OrderCanceTypes). |
| sessionName | TradingSessionName | Name of the trading session in which the order is to be canceled. |
| userAssignedCancelId | String | User assigned identifier for this cancel request. Recommend that this id follow branch + branch sequence number semantics. |

### ContraPartyStruct

Contraparty of a trade - as reported on order or quote fill report.

| Attribute | Type | Documentation |
|---|---|---|
| firm | ExchangeFirmStruct | Member Firm of contraparty (Executing or Giveup Firm) |
| quantity | long | Quantity traded with this contraparty |
| user | ExchangeAcronymStruct | User acronym of contra party. |

### CrossOrderStruct

Contains the information for the buy side and sell side of a cross order.

| Attribute | Type | Documentation |
|---|---|---|
| buySideOrder | OrderStruct | Contains the buy side information of a cross order. |
| sellSideOrder | OrderStruct | Contains the sell side information of the cross order. |

**FilledReportStruct**

Fill report from exchange - this struct is used for both order fills and quote fill reports.

| Attribute | Type | Documentation |
|---|---|---|
| account | string | The account field has multiple uses based upon type of market participant. For customers and firms it is used for customer account information. For market makers it contains the clearing subaccount information. |
| cmta | ExchangeFirmStruct | Clearing Member Trade Agreement - used if a different clearing member firm is assigned to the order. |
| contraParties | ContraPartyStructSequence | Contra parties that were traded against (includes the volume for each contra party) |
| executingBroker | string | Executing broker acrynom for trade. |
| executingOrGiveUpFirm | ExchangeFirmStruct | Member Firm Acronym who is the executing or give up firm. |
| extensions | string | Internal CBOE field for extensions to the order message. |
| fillReportType | ReportType | Type of fill report requested. |
| leavesQuantity | long | Quantity remaining in order that has not been traded |

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| optionalData | string | Optional data entered by the Originator when the order was entered. Optional text field.<br>Firms that want to give one DPM priority in participating in a trade use this field using P:EXCH.FIRM; or P:FIRM; EXCH is the exchange acronym. FIRM is the Firm acronym. EXCH is optional.<br>For Sweep and AIM: Enter A:AIS in the primary order to instruct CBOE to sweep all better priced protected quotes at away exchanges at the same time as the commencement of the AIM auction.<br>For AIM Auctions: Enter A:AIM as the first characters. If a Firm does not wish to cancel the primary order when the auction expires, the Firm must enter A:AIR, instead of A:AIM, in this field. This will designate the primary order to be returned to the system and trade or book as a regular order.<br>Options:<br>For Customer orders, the first four characters of this field get reported to the last four characters of the CBOE Trade Match Optional Data field. These four characters get reported to the OCC.<br>Preferred DPM:<br>Firms that want to give one DPM priority in participating in a trade use this field using P:EXCH.FIRM; or P:FIRM; EXCH is the exchange acronym. FIRM is the Firm acronym. EXCH is optional |
| originator | ExchangeAcronymStruct | The Member Firm or Organization that originated the Order. |
| orsId | string | Order Routing System Identifier |
| positionEffect | PositionEffect | Position effect (opening or closing or none) |
| price | PriceStruct | Price at which this portion of the order was traded |
| productKey | ProductKey | Product key of the product that was traded. |
| sessionName | TradingSessionName | Name of the trading session in which the trade occurred. |

| Attribute | Type | Documentation |
|---|---|---|
| side | Side | Side of the order (Bid or Ask) |
| subaccount | string | For market makers used to populate the clearing optional data. |
| timeSent | DateTimeStruct | Time the trade was executed by the trading system |
| tradedQuantity | long | Quantity traded |
| tradeId | CboeIdStruct | Identifier of the trade. Systems should track trades using this identifier, as it is referenced on trade busts and trade bust reinstates. Firms should also keep track of the OrderId and transactionSequenceNumber to ensure that the fill report message is unique. |
| transactionSequenceNumber | long | Transaction Sequence Number assigned to the report to help the client application order the fill report. |
| userAcronym | ExchangeAcronymStruct | The acronym for a user. |
| userAssignedId | string | User assigned identifier for quotes. Provided for FIX compatibility. |
| userId | string | User id of the user that submitted the order. |

**InternalizationOrderResultStruct**

Contains two OrderResultStructs for internalization orders; one for the primary order and one for the match order.

| Attribute | Type | Documentation |
|---|---|---|
| matchOrderResult | OrderResultStruct | Contains the information from the Order Result Struct for the internalized match order. |
| primaryOrderResult | OrderResultStruct | Contains the information from the Order Result Struct for the internalized primary order. |

**InternalizationOrderResultStructV2**

Contains two OrderResultStructs for internalization orders; one for the primary order and one for the match order.

| Attribute | Type | Documentation |
|---|---|---|
| matchOrderResult | OrderResultStructV2 | Contains the information from the Order Result Struct V2 for the internalized match order. |
| primaryOrderResult | OrderResultStructV2 | Contains the information from the Order Result Struct V2 for the internalized primary order. |

**LegOrderDetailStruct**

The Leg Order Detail Struct contains the product information and full information for the leg.

| Attribute | Type | Documentation |
|---|---|---|
| cancelledQuantity | Long | The cancelled leg quantity. |
| clearingFirm | ExchangeFirmStruct | The number or acronym to be associated with this leg if it is a non-option product. |
| coverage | Coverage | Whether the leg is covered or uncovered. |
| leavesQuantity | long | Current remaining open quantity for the order. |
| mustUsePrice | PriceStruct | The optional price field when a fixed leg price is normally set to no value. |
| originalQuantity | Long | Original leg quantity. |
| positionEffect | PositionEffect | Indicates that the resulting position in the leg product is in opening or close. |
| productKey | ProductKey | Details the product of the leg. |
| side | Side | The side of the leg. |
| tradedQuantity | Long | The traded leg quantity. |

### LegOrderEntryStruct

Message used to enter a strategy order into the system.

| Attribute | Type | Documentation |
|---|---|---|
| clearingFirm | ExchangeFirmStruct | Number or acronym to be associated with this leg if it is a non-option product. |
| coverage | Coverage | Whether the leg is covered or uncovered. |
| mustUsePrice | PriceStruct | Optional price field when a fixed leg price is normally set to no value. |
| positionEffect | PositionEffect | Indicates that the resulting position in the leg product is in opening or close. |
| productKey | ProductKey | The product of the leg. |

### LegOrderEntryStructV2

Message used to enter a strategy order into the system for the V2 interfaces.

| Attribute | Type | Documentation |
|---|---|---|
| extensions | string | Used to persist arbitrary data along with the order. |
| legOrderEntry | LegOrderEntryStruct | Used to enter a complex order.  References the LegOrderEntryStruct. |
| side | Side | Used to indicate the short sale position of a particular leg of the complex order. |

**OrderBustReinstateReportStruct**

Details of a bust trade reinstatement

| Attribute | Type | Documentation |
|---|---|---|
| bustReinstatedReport | BustReinstateReportStruct | Details of the bust trade reinstatement |
| reinstatedOrder | OrderDetailStruct | The order struct at the time of this event. |

**OrderBustReportStruct**

The message that is sent to a user when a trade is busted - contains the product information, order details, and the details of a trade bust.

| Attribute | Type | Documentation |
|---|---|---|
| bustedOrder | OrderDetailStruct | The order struct at the time of this event. |
| bustedReport | BustReportStructSequence | Bust trade report |

**OrderCancelReportStruct**

Order cancel report

| Attribute | Type | Documentation |
|---|---|---|
| cancelledOrder | OrderDetailStruct | The order struct at the time of this event. |
| cancelReport | CancelReportStructSequence | Cancel report |

### OrderContingencyStruct

The OrderContingencyStruct is used to define a contingency on an order.

| Attribute | Type | Documentation |
|---|---|---|
| price | PriceStruct | Price for contingency order |
| type | ContigencyType | Order contingency type (all or none, fill or kill, etc.) (See cmiConstants::ContingencyTypes |
| volume | long | Quantity for this order |

### OrderDetailStruct

The Order Detail Struct contains the product information and full information for the order.

| Attribute | Type | Documentation |
|---|---|---|
| orderStruct | OrderStruct | Order record from the order handling service |
| productInformation | ProductNameStruct | Product name for the product for which the order was submitted |
| statusChange | UpdateStatusReason | Status change that caused issuance of OrderDetailStruct from the CAS. |

### OrderEntryStruct

Message used to enter an order into the system.

| Attribute | Type | Documentation |
|---|---|---|

| Attribute | Type | Documentation |
|---|---|---|
| account | string | Accounts must be approved by the Membership Department. Account information can be obtained by calling the getValidUser method. Users can only use one account value per product class. The exact size is 3 and the data type is alpha only.<br><br>Options and Futures: Account field is required for market-maker order entry but optional for broker-dealer order entry. Accounts must be approved by the CBOE Membership Department and must be pre-configured by your API testing representative. Exact size is 3 and data type is alpha only.<br><br>Linkage: For entering a Principal (P) Linkage order to an away market, this field must contain the market-maker's clearing account. This is normally the Q-Account or market-maker acronym.<br><br>Clearing Information:<br><br>For market-makers, this typically would be either the joint account (often called q-account) or the market-maker three-letter badge acronym. Passed through to OCC. Required for Market-Maker and DPM roles in all sessions. For Market-Maker and DPM roles, CBOE validates the value of this field on inbound orders against the CBOE Membership system. For Market-Maker and DPMs, user cannot use more than one account per class. Optional for Broker-Dealer and Firm roles. CBOE performs no validation checks on the value of this field for Broker-Dealer and Firm roles. Exact size is 3 and data type is alpha only. For futures, if desired, this field can be the same value as subaccount. |
| branch | string | Used by the Originator to identify the Branch. Required for all roles. For W_MAIN orders, this must be a 3 character alpha only field that must be all uppercase. Orders in purely electronic sessions that do not interact with the trading floor may use between 1-3 alpha only, all uppercase characters. |

| Attribute | Type | Documentation |
|---|---|---|
| branchSequenceNumber | long | Sequence number that can be used by the Originator to uniquely identify the order. Maximum size is 4 digits and data type is numeric only. Branch + branchSequenceNumber must be unique for each day for each executingOrGiveUpFirm (give up firm or clearing firm) + correspondentFirm (if a correspondentFirm is used at all). |
| cmta | ExchangeFirmStruct | Clearing Member Trade Agreement Firm. Used to designate a clearing firm, if different from executingOrGiveUpFirm. This field is not connected to the user ID, product class, account, or subaccount at all and is not configured by the CBOE. It can be submitted on the fly. Optional for all roles. The maximum size is 5 and data type is alphanumeric.<br><br>Clearing information:<br><br>The CMTA (Clearing Member Trade Agreement) field is used to designate an OCC clearing firm if it is different from the executingOrGiveUpFirm. CBOE performs no validation checks on the CMTA field against the CBOE Membership system. This field is optional for all roles in all sessions.CMTA is comprised of two components: an exchangeFirmStruct which contains the exchange code and CMTA firmNumber. If you use CMTA, then you must use submit both of these two components.The exchange string is the exchange on which your order will trade. The exchange portion of the CMTA field is alpha only.The firmNumber is the OCC clearing firm where the order will clear. The firmNumber portion of the CMTA field is numeric only. Even though the maximum size for the firmNumber component is 5, CBOE will read the first three numbers of this field to use as the OCC clearing firm. In other words, if the desired CMTA firm at the OCC is "123", do not send "00123", send "123". |

| Attribute | Type | Documentation |
|---|---|---|
| contingency | OrderContingencyStruct | Contingency assigned to the order. CBOE currently only supports NONE, AON, FOK and IOC. The field is strongly encouraged for all orders. The maximum size is 2 with data type numberic only.<br><br>For Auctions:<br><br>- The OrderEntryStruct must contain an OrderContingencyStruct with type=ContingencyTypes.AUCTION_RESPONSE. The side of the auction response order must be tradable against the side indicated in the auction event. |
| correspondentFirm | string | Optional field that is available for use by the originating firm to identify a correspondent firm. This field is not connected to the user ID, product class, account, or subaccount at all and is not configured by the CBOE. It can be submitted on the fly. The maximum size is 4 characters and the data type is uppercase alphabetic only. This field should be left blank for Linkage orders<br><br>Clearing information:<br><br>The correspondent firm field is used by the executing give up firm to differentiate the firm or system sending the order. The 1st three characters of this field are mapped to the optional data field on the CBOE Trade Match (CTM) record. This field has no impact on the clearing of the trade. This field is optional. CBOE performs no validation checks on the correspondentFirm field against the CBOE Membership system. Maximum size is 4 characters and data type is uppercase alpha only. |
| coverage | Coverage | Covered, uncoverd, or unspecified. Optional for all roles. The maximum size is 1 and the data value is alphabetic. |
| cross | CrossingIndicator | 'C' if the order is a crossing order. Not supported in the current version of the CMi and is for futuer use in a subsequent version. CBOE currently ignores this value. |

| Attribute | Type | Documentation |
|---|---|---|
| executingOrGiveupFirm | ExchangeFirmStruct | The Member Firm for which the order will execute. Required for all roles. Broker (Broker-Dealer) role may choose from list of approved firms and the Market-Maker and DPM roles must use default firm only. The exact size is 3 and the data type is numeric only.<br><br>Clearing information:<br><br>This is the CBOE clearing firm that is representing the order in live trading (post trade processing firm). If no CMTA firm is present in the order, then the executingOrGiveUpFirm represents the OCC clearing firm where the order will clear. This field is required for all orders sent to CBOE for all roles in all sessions regardless of whether a CMTA firm is given or not. CBOE performs validation checks of executingOrGiveUpFirm against the CBOE Membership system on options orders routed to all sessions. Broker-Dealer and Firm roles must choose from a list of pre-approved and pre-configured executingOrGiveUpFirms and the Market-Maker and DPM roles must use default executingOrGiveUpFirm only.This field is comprised of two components: an ExchangeFirmStruct which contains the Exchange code and firmNumber.The Exchange string is the exchange on which your order will trade. The Exchange portion of the executingOrGiveUpFirm field is alpha only.If there is no CMTA given in the order, then the executingOrGiveUpFirm firmNumber will be the OCC clearing firm where the order will clear. If there is a CMTA given in the order, then the firmNumber is the CBOE clearing firm that is representing the order in live trading (post trade processing firm). The firmNumber portion of the CMTA field is numeric only. Even though the maximum size for the firmNumber component is 5, CBOE will only read the first three numbers of this field to use as the executingOrGiveUpFirm. In other words, if the desired executingOrGiveUpFirm firm is "123", do not send "00123", send "123". |

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| expireTime | DateTimeStruct | Date Time when order is to expire. CBOE currently ignores this value. |
| extensions | string | Used to persist arbitrary data along with the order.<br><br>For Auctions:<br><br>The OrderEntryStruct extensions field must contain a new field for auction response orders corresponding to the auction ID of the auction. The extensions field should contain the substring "auctionId=123:456" where 123 is the high CBOE Id and 456 is the low CBOE Id of the auction ID specified in the auction event. The standard field separator ("\u0001") must be used in between subfields of the extensions field.<br><br>Valid values for multiple routing choices on Linkage orders sent through CBOE are: REDI, OES, BRASS, LIME, ASSENT |

| Attribute | Type | Documentation |
|---|---|---|
| optionalData | string | Optional text field.<br><br>For Qualified Contingent Trade (QCT orders: Enter A:AIQ as the first characters in the primary order to instruct CBOE to treat the paired orders as QCT orders.<br><br>Firms that want to give one DPM priority in participating in a trade use this field using P:EXCH.FIRM; or P:FIRM; EXCH is the exchange acronym. FIRM is the Firm acronym. EXCH is optional.<br><br>For Sweep and AIM: Enter A:AIS in the primary order to instruct CBOE to sweep all better priced protected quotes at away exchanges at the same time as the commencement of the AIM auction.<br><br>For AIM Auctions: Enter A:AIM as the first characters. If a Firm does not wish to cancel the primary order when the auction expires, the Firm must enter A:AIR, instead of A:AIM, in this field. This will designate the primary order to be returned to the system and trade or book as a regular order.<br><br>Directed AIM: CBOE high and low order values prefixed by DAIM or Directed AIM match order.<br><br>Orders of origin Customer ("C"):<br><br>The first four characters are reported to the last four characters of CBOE Trade Match Optional Data field. These four characters are reported to OCC. Do not put "C:" in this field.<br><br>Orders of origin In-Crowd Market-maker ("I")<br><br>This field is not required. Do not put "I:" in this field.<br><br>Preferred DPM<br><br>Firms that give one DPM priority in participating in a trade use this field.Firm is specified as P:firm; and can coexist with other data that may be present in this field. "Firm" is the CBOE firm acronym as listed in the Order Test Plan. Please note that the colon : and semi-colon ; are both mandatory. |
| | | Linkage: This field is not used for Linkage.<br><br>Strategy order: For clearing purposes, this field must be populated not the leg struct. |

| Attribute | Type | Documentation |
|---|---|---|
| orderDate | string | An optional order date. If not set then CBOE will default to the current date. For new orders this field should be left blank or set to the current date. Format is YYYYMMDD. Do not use hyphens "-" or slashes "/" to separate the year, month and day. The exact size is 8 and the data type is numeric only. |
| orderNBBOProtectionType | NBBOProtectionType | This field determines if the order receives NBBO protection. The order has to be a customer order for this field to matter. |
| orderOriginType | OriginType | Origin of the order. See cmiConstants::OrderOrigins. Required for all roles. The maximum size is 1 character and the data type is alphabetic only. Broker-Dealer role can submit any allowable value, but Market-Maker can only enter value "M". |
| originalQuantity | long | The original quantity of the order. Required field with maximum size 9,999 for RTH and no maximum quantity for ETH. The data type is numeric only. |

| Attribute | Type | Documentation |
|---|---|---|
| originator | ExchangeAcronymStruct | Optional field. The member firm or organization that originated the order. If this field is populated, it should be populated with the User ID of the user entering the order. The maximum size is 3 and the data type is alphanumeric. Options and Futures: If a Broker_Dealer enters an order on behalf of a market-maker in the W_MAIN, CFE_MAIN, COF_MAIN or ONE_MAIN sessions, the market-maker's acronym must be entered in this field. The default exchange is CBOE, but an exchange may be specified. This is also called "Originator". Examples: "CBOE:ABC", or you may enter "ABC".<br><br>Clearing information:<br><br>This field would only be used for orders of origin "M", "I", and "N". This field is comprised of two components: an Exchange string which contains the Exchange code and acronym.The Exchange string is the exchange on which the market-maker will clear the trade. The Exchange portion of the originator field is alpha only.The acronym is the three-letter acronym ("badge") of the market-maker who originates the order. This field will typically be three characters (occasionally two).<br><br>Orders of origin "M"<br><br>Orders of origin "M" entered by a broker-dealer role must supply the three-letter acronym ("badge") of the market-maker. If a broker-dealer role submits an options order of origin "M" on behalf of a CBOE market-maker, then the three-letter (all alpha, all caps) MM acronym must go into either the originator field or the originator portion (positions 13-15) of the optional data field. The market-maker role does not have to enter the originator field when entering orders of origin "M". However, if a market-maker role wishes to enter the originator field when entering orders of origin "M", then it may enter the originator acronym into either the originator field or the originator portion (positions 13-15) of the optional data field.<br><br>Orders of origin "I"<br><br>Orders of origin "I" entered by a broker-dealer role must supply the three-letter acronym ("badge") of the market-maker. If a broker-dealer role submits an options order of origin "I" on behalf of a CBOE market-maker, then the three-letter (all alpha, all caps) MM acronym must go into the originator field (tag 9465 in FIX). If a broker-dealer role submits an options order of origin "I" on behalf of a CBOE market-maker, then it may also if it wishes put the originator acronym in the originator portion (positions 13-15) of the |

| Attribute | Type | Documentation |
|---|---|---|
| positionEffect | PositionEffect | The effect that the order has on the user's overall position (Opening, Closing, or Not Applicable). Required for Customer ("C") options orders. Ignored for Firm ("F") orders. This field is required for all W_MAIN orders (both single leg and strategy). However, this value must be set to a value of "N" for all orders of origin "M" or "I" in the W_MAIN session. This field is optional for orders in purely electronic sessions (ONE_MAIN, CFE_MAIN, W_STOCK) that don't interact with the trading floor. The maximum size is 1 and the data type is alpha only. This field must be set to "O" (open) for Linkage orders. |
| price | PriceStruct | Required for all roles. Limit price of the order. Must be in minimum increments of nickels (0.05) below 3.00 and dimes (0.10) above 3.00. The whole portion of the price is maximum size of 3 in numberic format. The decimal portion is numeric only and should be given multiplied by 1 billion. |
| productKey | ProductKey | ProductKey for the product for which the order is being submitted. Required for all roles with the maximum size of 20 with data type numeric only. |
| sessionNames | TradingSessionNameSequence | Sequence of trading session names where the order is eligible to trade. At this time only one trading session name can be specified. If no trading session name is specified - the order will default to the main trading session for a product. |
| side | Side | Side of the order: either Buy or Sell. Required for all roles. The exact size is 1 and the data type is alphabetic only.<br><br>This field is also used for short sale indication on simple orders. Valid values are in cmiConstants:Sides |

| Attribute | Type | Documentation |
|---|---|---|
| subaccount | string | The maximum size is 10 and the data type is alphanumeric. CBOE performs no validation on this field.<br><br>Options and Futures: Required for all futures orders (ONE and CFE). Should not be used for orders of origin "M", "N", or "I". For entering orders of any origin other than "M", "I" or "N", (including "C" = customer), this field is optional and can contain any value. Specifies the market-maker account where the trade will clear at the OCC. This field could be the Q-account, joint account, or the market-maker acronym of the market-maker.<br><br>Clearing information:<br><br>CBOE performs no validation checks on subaccount against the CBOE Membership system. Maximum size is 6 and data type is alphanumeric. For Broker and Firm roles, if subaccount is used then the account field is not required. If subaccount is specified, then the trade will clear into the subaccount instead of the account field.<br><br>Options: This field is optional for all roles in the W_MAIN session.<br><br>Futures: Subaccount is required for CFE and OneChicago futures. It specifies the account into which the trade will clear. |
| timeInForce | TimeInforce | Time the order is in force (Day, Good until Cancelled, Good until Date) (See cmiConstants::TimesInForce). Required for all roles. |
| userAssignedId | string | Similar to Optional Data except the information never leaves CBOEdirect. Optional for all roles. The maximum size is 20 and the data type is aphanumeric. |

**LightOrderEntryStruct**

Message used to enter a light order into the system.

| Attribute | Type | Documentation |
|---|---|---|

| Attribute | Type | Documentation |
|---|---|---|
| branch | string | Used by the Originator to identify the Branch. Required for all roles. For W_MAIN orders, this must be a 3 character alpha only field that must be all uppercase. Orders in purely electronic sessions that do not interact with the trading floor may use between 1-3 alpha only, all uppercase characters. |
| branchSequenceNumber | long | Sequence number that can be used by the Originator to uniquely identify the order. Maximum size is 4 digits and data type is numeric only. Branch + branchSequenceNumber must be unique for each day for each executingOrGiveUpFirm (give up firm or clearing firm) + correspondentFirm (if a correspondentFirm is used at all). |
| originalQuantity | long | The order's original quantity |
| Price | double | Bid or offer – depending on the side. |
| productKey | ProductKey | ProductKey for the product for which the order is being submitted. Required for all roles with the maximum size of 20 with data type numeric only. |
| side | Side | Side of the order: either Buy or Sell. Required for all roles. The exact size is 1 and the data type is alphabetic only. |
| positionEffect | PositionEffect | Position Effect (Opening or closing or none) |
| coverage | Coverage | Covered, uncovered, or unspecified |
| isNBBOProtected | boolean | Order is NBBO protected |
| isIOC | boolean | Order is an IOC order |
| orderOriginType | OriginType | Origin of the order. 'C ' - customer, 'F'- firm, 'B' - Broker/Dealer, 'X' - Customer Broker/Dealer, 'M' - Market Maker. |
| cmtaExchange | Exchange | The exchange for the CMTA |
| cmtaFirmNumber | string | The Firm number of the CMTA |
| pdpm | string | Preferred Designated Primary Market Maker |

| Attribute | Type | Documentation |
|---|---|---|
| userAssignedId | string | User Assigned order identifier - added to track user assigned quote identifiers, can also be used when entering orders |
| activeSession | TradingSessionName | Indicates the session that is currently active. |

### OrderFilledReportStruct

Order Filled Report

| Attribute | Type | Documentation |
|---|---|---|
| filledOrder | OrderDetailStruct | The order struct at the time of this event. |
| filledReport | FilledReportStructSequence | Fill report details |

### OrderIdStruct

The OrderIdStruct contains two keys. The primary key is the combination of highCboeId and lowCboeId. The alternate key is the combination of the other five fields: executingOrGiveUpFirm, branch, branchSequenceNumber, correspondentFirm, and orderDate. The system will look at the high and low fields first. If one of the two fields is non-zero, the system will use these fields, and ignore the other five fields. If both the high and low fields are zero (zero is not a legal ID, and represents NO_VALUE), the system will use the other five fields as the order ID.

| Attribute | Type | Documentation |
|---|---|---|
| branch | string | Used by the Originator to Identify the Branch. This is a 3 character alphabetic only field. |
| branchSequenceNumber | string | Unique sequence number for the order that is assigned by the Originator.This is a 4 digit numeric only field. |

| Attribute | Type | Documentation |
|---|---|---|
| correspondentFirm | string | Optional field that is available for use by the originating firm to identify a correspondent firm. |
| executingOrGiveUpFirm | ExchangeFirmStruct | Member Firm Acronym of the firm for which the order will execute. This is often the same as the Originator. If the Originator is a member organization - the executing Firm will be set to the member firm that the order will be given up to. Note that CBOE Trading operations must pre-approve and configure all give up relationships. |
| highCboeId | long | Major part of CBOE order identifier |
| lowCboeId | long | Minor or least significant part of CBOE order key |
| orderDate | string | Date assigned to the order by the Order Service. It is in YYYYMMDD format. |

### OrderResultStruct

Contains a valid order ID struct and uses the OperationResultStruct in cmiUtil.

| Attribute | Type | Documentation |
|---|---|---|
| orderID | OrderIdStruct | Order ID for the associated order. |
| result | OperationResultStruct | Returns the OperationResultStruct used to define generic operation when an exception is not thrown. See cmiUtil. |

### OrderResultStructV2

Contains a valid order struct and uses the OperationResultStruct in cmiUtil.

| Attribute | Type | Documentation |
|---|---|---|

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| order | OrderStruct | Returns the order information from the OrderStruct. |
| result | OperationResultStruct | Returns the OperationResultStruct used to define generic operation when an exception is not thrown. See cmiUtil. |

### LightOrderResultStruct

Message sent to Firm on successful Light Order entry.

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| branch | string | Used by the Originator to Identify the Branch. This is a 3 character alphabetic only field. |
| branchSequenceNumber | long | Unique sequence number for the order that is assigned by the Originator.This is a 4 digit numeric only field. |
| orderHighId | long | Major part of CBOE order identifier |
| orderLowId | long | Minor or least significant part of CBOE order key |
| side | Side | The side of the light order |
| leavesQuantity | long | The remaining light order quantity |
| tradedQuantity | long | The quantity that was traded on the order |
| cancelledQuantity | long | The order quantity that was cancelled |
| reason | ActivityReason | The reason for the result struct |
| time | DateTimeStruct | Date and time of the result stuct |

### LightOrderReplaceResultStruct

Message sent to Firm on successful Light Order replace.

| **Attribute** | **Type** | **Documentation** |
|---|---|---|

| Attribute | Type | Documentation |
|---|---|---|
| originalOrder | LightOrderResultStruct | Returns the LightOrderResultStruct of the original order |
| newOrder | LightOrderResultStruct | Returns the LightOrderResultStruct of the replacement order. |

**OrderStruct**

The OrderStruct is the record of the order as maintained by the trading system.

| Attribute | Type | Documentation |
|---|---|---|
| account | string | The order struct at the time of this event. |
| activeSession | TradingSessionName | Indicates the session that's currently active. |
| averagePrice | PriceStruct | Average price of all executions against this order for the entire life of the order. |
| cancelledQuantity | long | Quantity cancelled |
| cancelReportType | ReportType | Type of report - overall strategy, leg of strategy. |
| classKey | ClassKey | Class Key for the product being traded |
| cmta | ExchangeFirmStruct | Clearing Member Trade Agreement Firm. Used to designate a clearing firm, if different from ExecutingOrGiveUpFirm. |
| contingency | OrderContingencyStruct | Contingency assigned to this order. |
| coverage | Coverage | Covered, uncovered, or unspecified |
| cross | CrossingIndicator | 'C' if the order is a crossing order. |
| crossedOrder | OrderIdStruct | The order id of the crossed order - if the order is a cross. Not supported in the current version of the CMi and is for future use in a subsequent version. |
| expireTime | DateTimeStruct | Date Time when order is to expire. |
| extensions | string | Additional information persisted with the order. |

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| leavesQuantity | long | Remaining open quantity. |
| legOrderDetails | LegOrderDetailStructSequence | Leg detail of a crossing order. |
| optionalData | string | Optional data.<br>For Qualified Contingent Trade (QCT) orders: Enter A:AIQ as the first characters in the primary order to instruct CBOE to treat the paired orders as QCT orders<br>Firms that want to give one DPM priority in participating in a trade use this field using P:EXCH.FIRM; or P:FIRM;  EXCH is the exchange acronym.  FIRM is the Firm acronym.  EXCH is optional.<br>For Sweep and AIM: Enter A:AIS in the primary order to instruct CBOE to sweep all better priced protected quotes at away exchanges at the same time as the commencement of the AIM auction.<br>For AIM Auctions:  Enter A:AIM as the first characters.  If a Firm does not wish to cancel the primary order when the auction expires, the Firm must enter A:AIR, instead of A:AIM, in this field.  This will designate the primary order to be returned to the system and trade or book as a regular order.<br>Options:<br>For Customer orders, the first four characters of this field get reported to the last four characters of the CBOE Trade Match Optional Data field. These four characters get reported to the OCC. |
| orderId | OrderIdStruct | Order identification - contains both the user order identification and the system assigned OrderKey |
| orderNBBOProtectionType | NBBOProtectionType | This field determines if the order receives NBBO protection.  The order has to be a customer order for this field to matter. |
| orderOriginType | OriginType | Origin of the order. 'C ' - customer, 'F'- firm, 'B'  - Broker/Dealer, 'X' -  Customer Broker/Dealer, 'M' - Market Maker. |
| originalQuantity | long | Original quantity of the order. |

| Attribute | Type | Documentation |
|---|---|---|
| originator | ExchangeAcronymStruct | The Member Firm or Member Organization the originated the order. |
| orsId | string | Order Routing System Identifier |
| positionEffect | PositionEffect | Position Effect (Opening or closing or none) |
| price | PriceStruct | Bid or offer price - depending on side |
| productKey | ProductKey | ProductKey for the product for which the order is being submitted. |
| productType | ProductType | CBOE product type |
| receivedTime | DateTimeStruct | Time the order was received by the system |
| sessionAveragePrice | PriceStruct | Average price of all executions against this order for this trading session. |
| sessionCancelledQuantity | long | Quanity cancelled for this order during the current trading session. |
| sessionNames | TradingSessionNameSequence | Names of the trading sessions where the user permits the order to be traded. |
| sessionTradedQuantity | long | Quantity of the order that was executed during this trading session. |
| side | Side | Side of the order either Buy or Sell. |
| source | Source | Source of the order - either SBT or TPF (See cmiConstants::Sources) |
| state | OrderState | Order State (booked, active, cancelled, etc.) (See cmiConstants::OrderStates) |
| subaccount | string | Sub account |
| timeInForce | TimeInforce | Time the order is in force (Day, Good until Cancelled, Good until Date) (See cmiConstants::TimesInForce). |
| tradedQuantity | long | Quantity traded |

| Attribute | Type | Documentation |
|---|---|---|
| transactionSequenceNumber | long | Transaction Sequence Number to identify the ordering of order reports. OrderDetailStructs can be received via different callbacks - ordering is not guaranteed |
| userAcronym | ExchangeAcronymStruct | User assigned acronym. |
| userAssignedID | string | User Assigned order identifier - added to track user assigned quote identifiers, can also be used when entering orders |
| userId | string | Trading system userid of the user that submitted the order |

### ORDOrderStruct

| Attribute | Type | Documentation |
|---|---|---|
| bookedQuantity | long | The quantity that went to the order book. |
| orderID | OrderIdStruct | The order identification |
| state | OrderState | The state of the order. |

### PendingOrderStruct

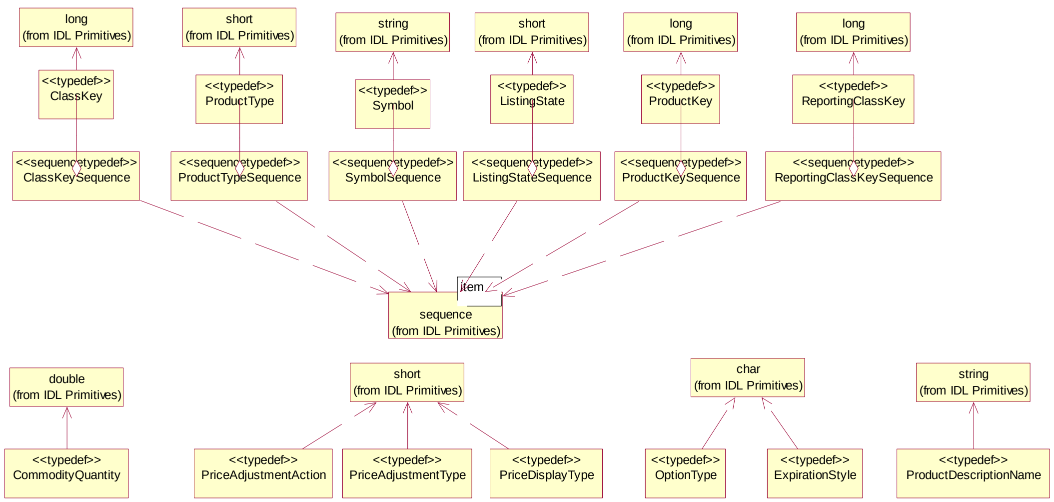
The Pending Order Struct is used when a product adjustment is pending. When an adjustment occurs, due to a corporate action then existing orders will likewise have to be adjusted. The Pending Order Struct has the current order and the adjusted order.

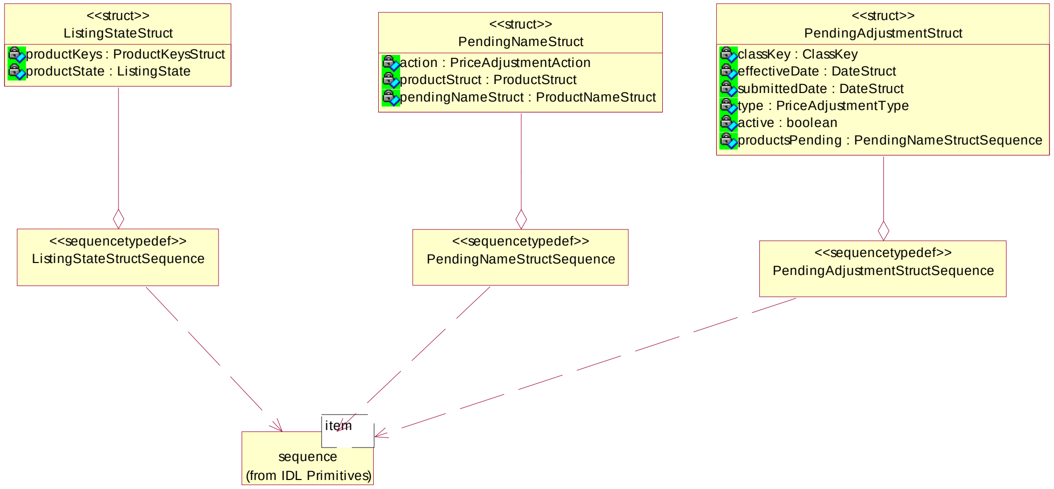
| Attribute | Type | Documentation |
|---|---|---|
| currentOrder | OrderStruct | The current order whose product has a pending adjustment. |

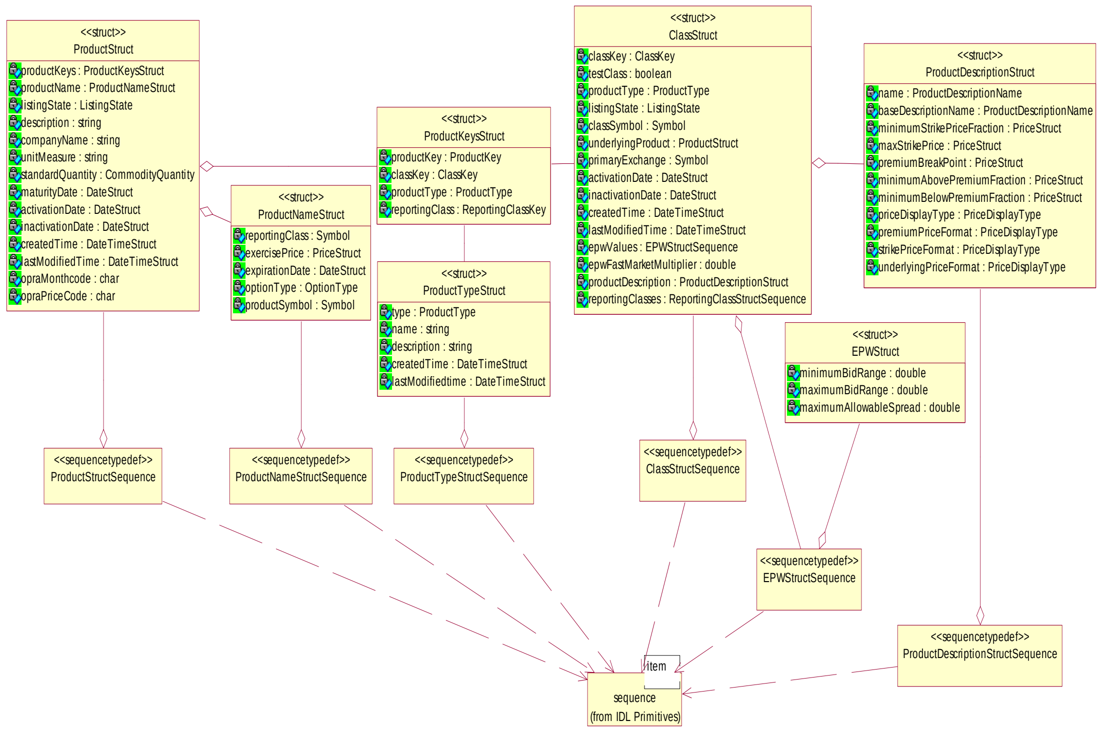
| Attribute | Type | Documentation |
|---|---|---|
| pendingOrder | OrderStruct | The order that will replace the current order - when the pending adjustment takes effect. |
| pendingProductName | PendingNameStruct | Provides the name of the current product and the adjusted product. |

## *Module cmiProduct*

Messages describing products within the system.




<<struct>>
ListingStateStruct
productKeys : ProductKeysStruct
productState : ListingState
<<struct>>
PendingNameStruct
action : PriceAdjustmentAction
productStruct : ProductStruct
pendingNameStruct : ProductNameStruct
<<struct>>
PendingAdjustmentStruct
classKey : ClassKey
effectiveDate : DateStruct
submittedDate : DateStruct
type : PriceAdjustmentType
active : boolean
productsPending : PendingNameStructSequence
<<sequencetypedef>>
ListingStateStructSequence
<<sequencetypedef>>
PendingNameStructSequence
<<sequencetypedef>>
PendingAdjustmentStructSequence
item
sequence
(from IDL Primitives)


<<struct>>
ProductStruct
productKeys : ProductKeysStruct
productName : ProductNameStruct
listingState : ListingState
description : string
companyName : string
unitMeasure : string
standardQuantity : CommodityQuantity
maturityDate : DateStruct
activationDate : DateStruct
inactivationDate : DateStruct
createdTime : DateTimeStruct
lastModifiedTime : DateTimeStruct
opraMonthcode : char
opraPriceCode : char
<<struct>>
ProductKeysStruct
productKey : ProductKey
classKey : ClassKey
productType : ProductType
reportingClass : ReportingClassKey
<<struct>>
ProductNameStruct
reportingClass : Symbol
exercisePrice : PriceStruct
expirationDate : DateStruct
optionType : OptionType
productSymbol : Symbol
<<struct>>
ProductTypeStruct
type : ProductType
name : string
description : string
createdTime : DateTimeStruct
lastModifiedtime : DateTimeStruct
<<struct>>
ClassStruct
classKey : ClassKey
testClass : boolean
productType : ProductType
listingState : ListingState
classSymbol : Symbol
underlyingProduct : ProductStruct
primaryExchange : Symbol
activationDate : DateStruct
inactivationDate : DateStruct
createdTime : DateTimeStruct
lastModifiedTime : DateTimeStruct
epwValues : EPWStructSequence
epwFastMarketMultiplier : double
productDescription : ProductDescriptionStruct
reportingClasses : ReportingClassStructSequence
<<struct>>
ProductDescriptionStruct
name : ProductDescriptionName
baseDescriptionName : ProductDescriptionName
minimumStrikePriceFraction : PriceStruct
maxStrikePrice : PriceStruct
premiumBreakPoint : PriceStruct
minimumAbovePremiumFraction : PriceStruct
minimumBelowPremiumFraction : PriceStruct
priceDisplayType : PriceDisplayType
premiumPriceFormat : PriceDisplayType
strikePriceFormat : PriceDisplayType
underlyingPriceFormat : PriceDisplayType
<<struct>>
EPWStruct
minimumBidRange : double
maximumBidRange : double
maximumAllowableSpread : double
<<sequencetypedef>>
ProductStructSequence
<<sequencetypedef>>
ProductNameStructSequence
<<sequencetypedef>>
ProductTypeStructSequence
<<sequencetypedef>>
ClassStructSequence
<<sequencetypedef>>
EPWStructSequence
<<sequencetypedef>>
ProductDescriptionStructSequence
item
sequence
(from IDL Primitives)

## Typedefs:

| | |
|---|---|
| ClassKey *is a* Quote<br>ClassKey *is a* FloorTradeMaintenanceService<br>ClassKey *is a* AuctionStruct<br>ClassKey *is a* MarketQuery<br>ClassKey *is a* long | A unique number that is used to identify a product class. |
| CommodityQuantity *is a* double | Quantity of the underlying commodity that the product is based on - for commodity derivatives only not option products. |
| ExpirationStyle *is a* char | Style of Expiration (American or European) (See cmiConstants::ExpirationStyles). |
| ListingState *is a* short | Listing State of class (Active, Inactive, Obsolete, Unlisted) (refer to cmiConstants::ListingStates) |
| OptionType *is a* char | Type of option - either 'C' - Call or 'P' - Put (See cmiConstants::OptionTypes). |
| PriceAdjustmentAction *is a* short | Action to perform when applying a price adjustment to a product (see cmiConstants::PriceAdjustmentActions). |
| PriceAdjustmentType *is a* short | Type of price adjustment (corporate action) (see cmiConstants::PriceAdjustmentTypes) |
| PriceDisplayType *is a* short<br>PriceDisplayType *is a* short | Indicates if the price is to be displayed as a Fraction or a Decimal format. |

| | |
|---|---|
| ProductDescriptionName *is a* string | Name that identifies a particular ProductDescription. ProductDescriptions (combinations of attributes on prices and formats for products) are stored and referenced by name for ease in creation of new products. |
| ProductKey *is a* MarketQuery<br>ProductKey *is a* AuctionStruct<br>ProductKey *is a* long | Unique identifier for a product in the system. Applies to all products, including underlying products, not just products traded on the system. |
| ProductType *is a* short<br>ProductType *is a* AuctionStruct | Type of product (debt, equity, option, etc.) (See cmiConstants::ProductTypes). |
| ReportingClassKey *is a* long | Unique identifier of a reporting class. |
| Symbol *is a* string | CBOE Ticker symbol for a product |

**Typedefs for Sequences:**

ClassKeySequence *is a typedef of a sequence of* ClassKey

ClassKeySequence *is a typedef of a sequence of* ProductQueryV6

ClassStructSequence *is a typedef of a sequence of* ClassStruct

EPWStructSequence *is a typedef of a sequence of* EPWStruct

ListingStateSequence *is a typedef of a sequence of* ListingState

ListingStateStructSequence *is a typedef of a sequence of* ListingStateStruct

PendingAdjustmentStructSequence *is a typedef of a sequence of* PendingAdjustmentStruct

PendingNameStructSequence *is a typedef of a sequence of* PendingNameStruct

ProductDescriptionStructSequence *is a typedef of a sequence of* ProductDescriptionStruct

ProductKeySequence *is a typedef of a sequence of* ProductKey

ProductNameStructSequence *is a typedef of a sequence of* ProductNameStruct

ProductStructSequence *is a typedef of a sequence of* ProductStruct

ProductTypeSequence *is a typedef of a sequence of* ProductType

ProductTypeStructSequence *is a typedef of a sequence of* ProductTypeStruct

ReportingClassKeySequence *is a typedef of a sequence of* ReportingClassKey

SymbolSequence *is a typedef of a sequence of* Symbol

---

**Structs:**

### ClassStruct

Defines a class of products traded within the system. Definition includes underlying product, product type, price multiplier.

Each product traded in the system is assigned to a class. This applies to non-derivative products, as well.

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| activationDate | DateStruct | Activation date. |
| classKey | ClassKey | Class Key (Unique Identifier). |
| classSymbol | Symbol | Class Symbol. |
| createdTime | DateTimeStruct | Time class was created in the system. |
| epwFastMarketMultiplier | double | Fast market multiplier that is to be used for quote bid-ask spreads under fast market conditions |
| epwValues | EPWStructSequence | A table of bid-ask spread widths based upon price ranges in the bid prices |
| inactivationDate | DateStruct | Inactivation date. |

| Attribute | Type | Documentation |
|---|---|---|
| lastModifiedTime | DateTimeStruct | Last time class information was modified. |
| listingState | ListingState | Listing state of the product, such as active, inactive, obsolete, unlisted (see cmiConstants::ListingStates) |
| primaryExchange | Symbol | Primary exchange class is traded on. |
| productDescription | ProductDescriptionStruct | Product details |
| productType | ProductType | Type of product (equity, future, option, etc.) (Refer to cmiConstants::ProductTypes). |
| reportingClasses | ReportingClassStructSequence | List of reporting classes. |
| testClass | boolean | Test classes in the system that are not real products. |
| underlyingProduct | ProductStruct | Underlying Product of products belonging to this class. |

### EPWStruct

Describes the exchange prescribed price width for a range of prices as defined by [minimumBidRange,MaximumBidRange]

| Attribute | Type | Documentation |
|---|---|---|
| maximumAllowableSpread | double | Bid-ask spread that is in effect when the bid price is within the [MinimumBidRange, MaximumBidRange] |
| maximumBidRange | double | Maximum bid for which this bid-ask spread applies |
| minimumBidRange | double | Minimum bid for which this bid-ask spread applies |

### ListingStateStruct

Status message that reports a change in the listing state of a product.

| Attribute | Type | Documentation |
|---|---|---|
| productKeys | ProductKeysStruct | ProductKeys for the product for which the listing state has changed. |
| productState | ListingState | New listing state of the product (see cmiConstants::ListingStates). |

### PendingAdjustmentStruct

Indicates an adjustment that will occur for an option class, usually as a result of a corporate action (stock split, merger, etc.) and the list of products affected by the adjustment.

| Attribute | Type | Documentation |
|---|---|---|
| active | boolean | Indicates if this adjustment is currently active |
| classKey | ClassKey | Unique identifier of the class for which the pending adjustment is being reported |
| effectiveDate | DateStruct | Effective date of adjustment. |
| productsPending | PendingNameStructSequence | List of products belonging to this series that will be modified (added, deleted, updated) on the effective date. For options - these are a list of series that will be introduced when the effective date of the adjustment occurs. |
| submittedDate | DateStruct | Date adjustment was submitted |
| type | PriceAdjustmentType | Type of adjustment (see cmiConstants::PriceAdjustmentTypes) |

### PendingNameStruct

An update action for a product that is to be changed as result of a corporate action (added, deleted, updated, moved).

| Attribute | Type | Documentation |
|---|---|---|
| action | PriceAdjustmentAction | Action to be performed on product (see cmiConstants::PriceAdjustmentActions). |
| pendingNameStruct | ProductNameStruct | New naming information for product (can include exercise price, symbol, etc.). |
| productStruct | ProductStruct | The product to be updated (add, changed, deleted) on the effective date. |

**ProductDescriptionStruct**

Product description contains information on price ranges and price formats. The format for the information (attribute names and definitions) is taken from the TPF trading engine product service. Most of the information in this struct can be safely ignored in client applications.

| Attribute | Type | Documentation |
|---|---|---|
| baseDescriptionName | ProductDescriptionName | The ProductDescription that this particular product description is based upon |
| maxStrikePrice | PriceStruct | Maximum strike price for this product. |
| minimumAbovePremiumFraction | PriceStruct | Fraction (tick size) to be used if the premium price is above the PremiumBreakPoint |
| minimumBelowPremiumFraction | PriceStruct | Fraction (tick size) to be used if the premium price is below the PremiumBreakPoint |
| minimumStrikePriceFraction | PriceStruct | Minimum strike price fraction for the product. This is used by the trading system when adjusting products due to corporate actions (splits). |
| name | ProductDescriptionName | Name of this Product Description |

| Attribute | Type | Documentation |
|---|---|---|
| premiumBreakPoint | PriceStruct | Premium (price where product is trading) break point for changes in tick size. If premium is below breakpoint the tick size is the minimumBelowPremiumFraction. If premium is above the breakpoint, then use minimumAbovePremiumFraction. |
| premiumPriceFormat | PriceDisplayType | Price format for premium - either fraction or decimal |
| priceDisplayType | PriceDisplayType | Not used in SBT |
| strikePriceFormat | PriceDisplayType | Price format (fraction or decimal) of the strike price |
| underlyingPriceFormat | PriceDisplayType | Price format of the underlying product |

**ProductKeysStruct**

Struct that provides the set of keys that defines the product. These include the unique identifer for a product, the ProductKey and the ClassKey of the product, the type of product, and the Reporting class symbol.

| Attribute | Type | Documentation |
|---|---|---|
| classKey | ClassKey | Unique identifier of the class to which the product belongs |
| productKey | ProductKey | Unique identifier for the product |
| productType | ProductType | Product type (see cmiConstants::ProductTypes) |
| reportingClass | ReportingClassKey | Reporting class of the product |

**ProductNameStruct**

Defines the name of a product - includes the reporting class symbol, expiration date, exercise price, and type of option.

| Attribute | Type | Documentation |
|---|---|---|

| Attribute | Type | Documentation |
|---|---|---|
| exercisePrice | PriceStruct | Exercise Price |
| expirationDate | DateStruct | Expiration date is the date that the option or future will expire. This is usually the day after the third Friday of the respective month. |
| optionType | OptionType | Option Type |
| productSymbol | Symbol | Symbol for product - this is the symbol of the class that is associated to the product - usually the underlying symbol. |
| reportingClass | Symbol | Reporting class symbol. This is the symbol that is displayed for trading. |

**ProductStruct**

Defines a product in the system, including name, product keys, etc.

| Attribute | Type | Documentation |
|---|---|---|
| activationDate | DateStruct | Activation date |
| companyName | string | Company name |
| createdTime | DateTimeStruct | Time product was created |
| description | string | Description of the product |
| inactivationDate | DateStruct | Inactivation date |
| lastModifiedTime | DateTimeStruct | Last modification time |
| listingState | ListingState | Listing state of product (see cmiConstants:: ListingStates) |
| maturityDate | DateStruct | Maturity date is used for bonds only.  It will be empty for options, futures, or any other non-bond product. |
| opraMonthcode | char | OPRA month code |

| Attribute | Type | Documentation |
|---|---|---|
| opraPriceCode | char | OPRA Price Code |
| productKeys | ProductKeysStruct | Product keys for the product |
| productName | ProductNameStruct | Product name |
| standardQuantity | CommodityQuantity | Standard quantity |
| unitMeasure | string | Unit of measure |

### ProductTypeStruct

| Attribute | Type | Documentation |
|---|---|---|
| createdTime | DateTimeStruct | Date and time when product was created. |
| description | string | Product Description. |
| lastModifiedtime | DateTimeStruct | Last date time the product was modified. |
| name | string | Product Name |
| type | ProductType | Product Type. |

### ReportingClassStruct

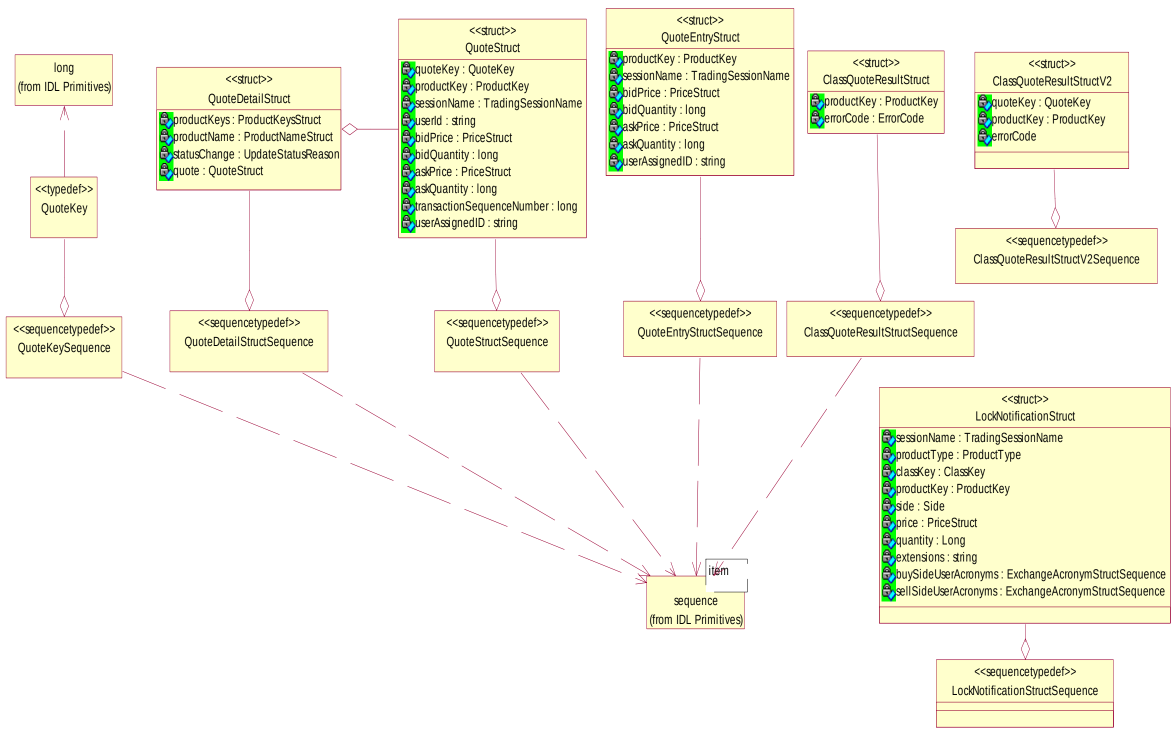
Defines a reporting class in the system, including class key, product type, etc.

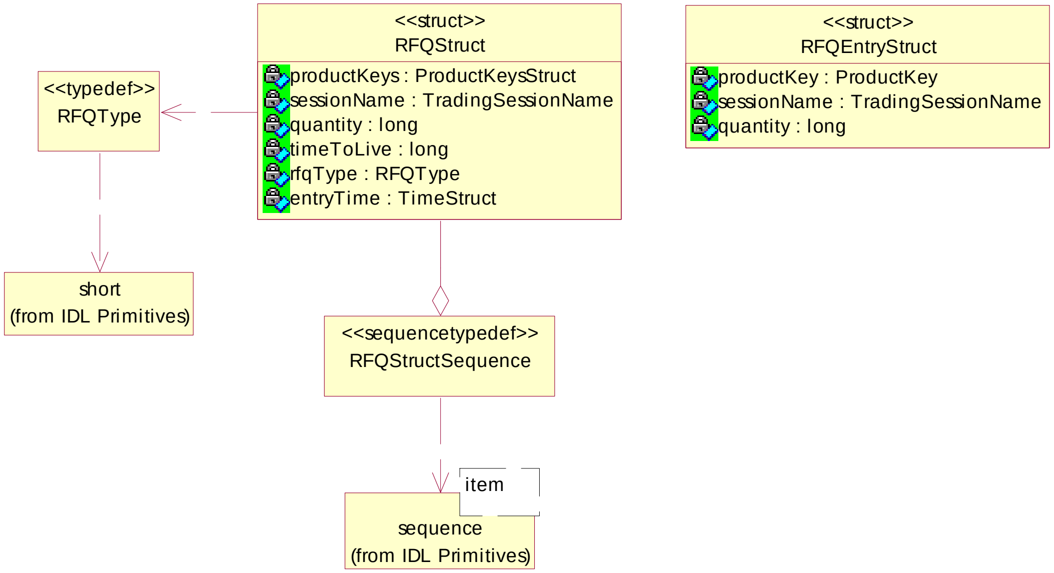
| Attribute | Type | Documentation |
|---|---|---|
| activationDate | DateStruct | The date the reporting class was activated. |
| classKey | ReportingClassKey | Unique identifier for the reporting class. |

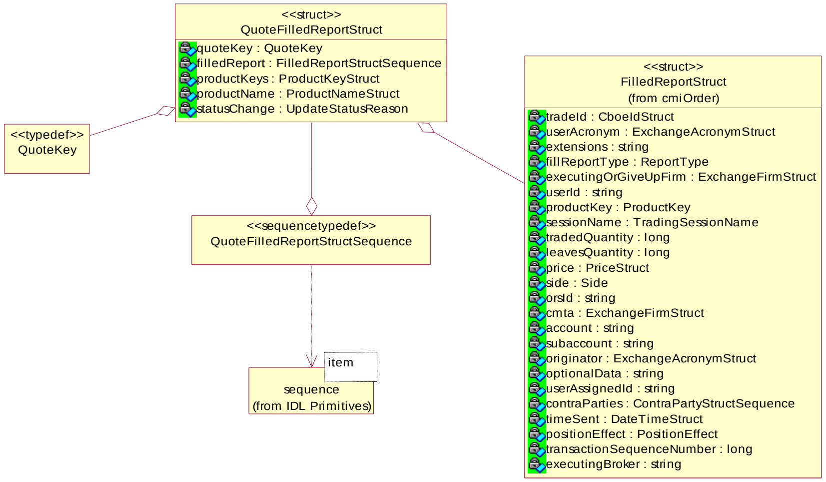
| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| contractSize | long | The size of the contract traded for the reporting class. |
| createdTime | DateTimeStruct | The time the reporting class was created. |
| inactivationDate | DateStruct | The date the reporting class was inactivated. |
| lastModifiedTime | DateTimeStruct | The last time the reporting class was modified. |
| listingState | ListingState | The list state for the reporting class. |
| productClassKey | ClassKey | Identifier for the product class. |
| productClassSymbol | Symbol | Ticker symbol for the product class. |
| productType | ProductType | Identifies the product type for the associated reporting class. |
| reportingClassSymbol | Symbol | Ticker symbol for the reporting class. |
| transactionFeeCode | string | Identifes the associated fee for the trade. |

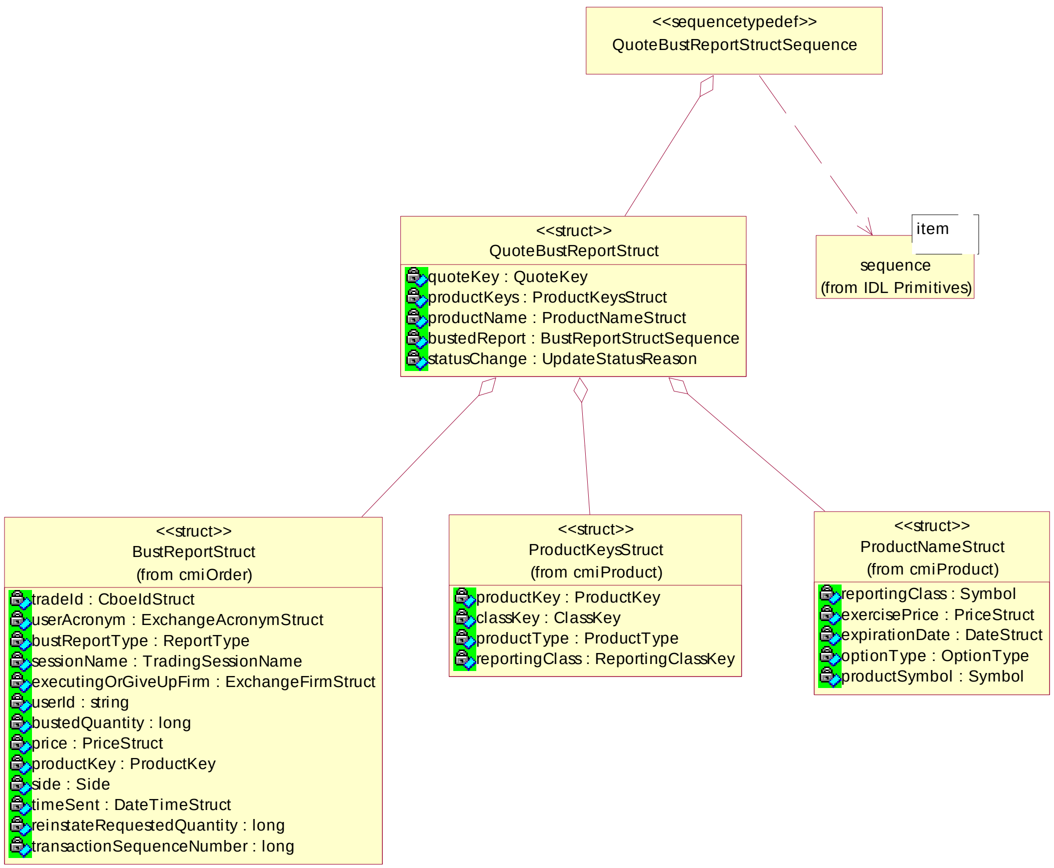
## *Module cmiQuote*

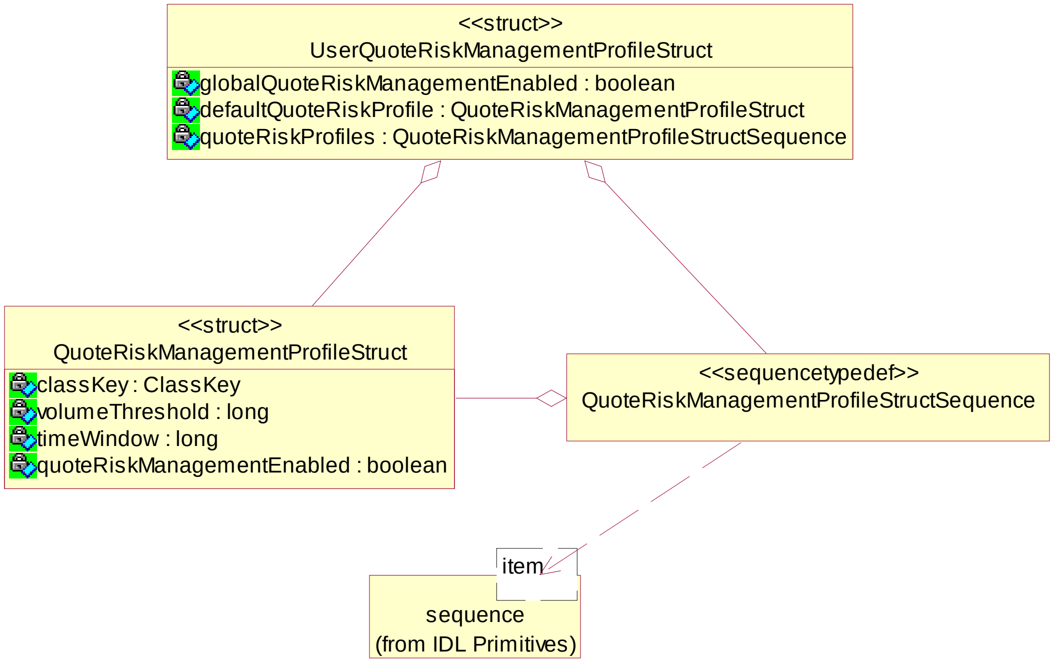
The cmiQuote module contains structs and datatypes for Quotes and Request for Quotes. Messages are provided for quote status and quote filled reports.

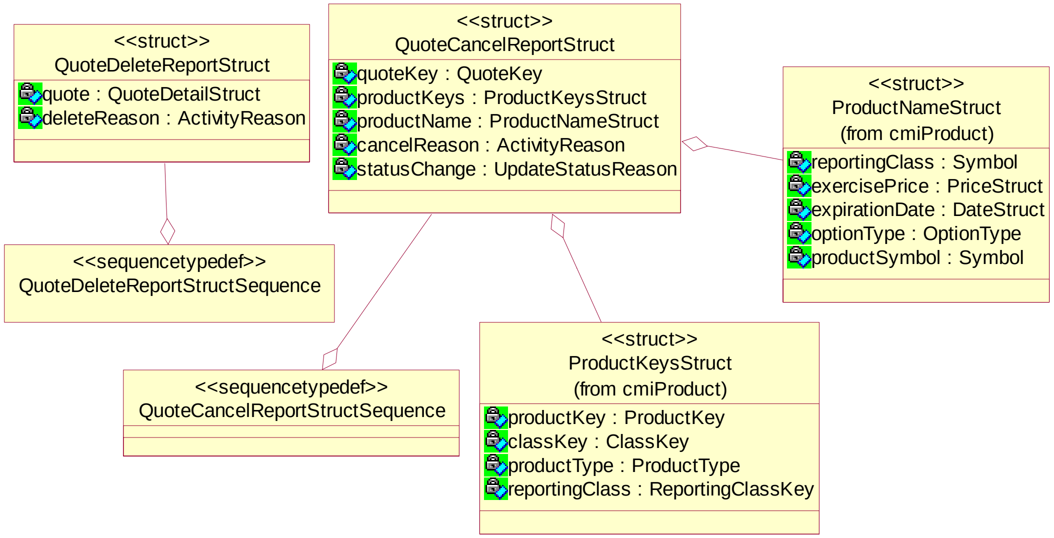


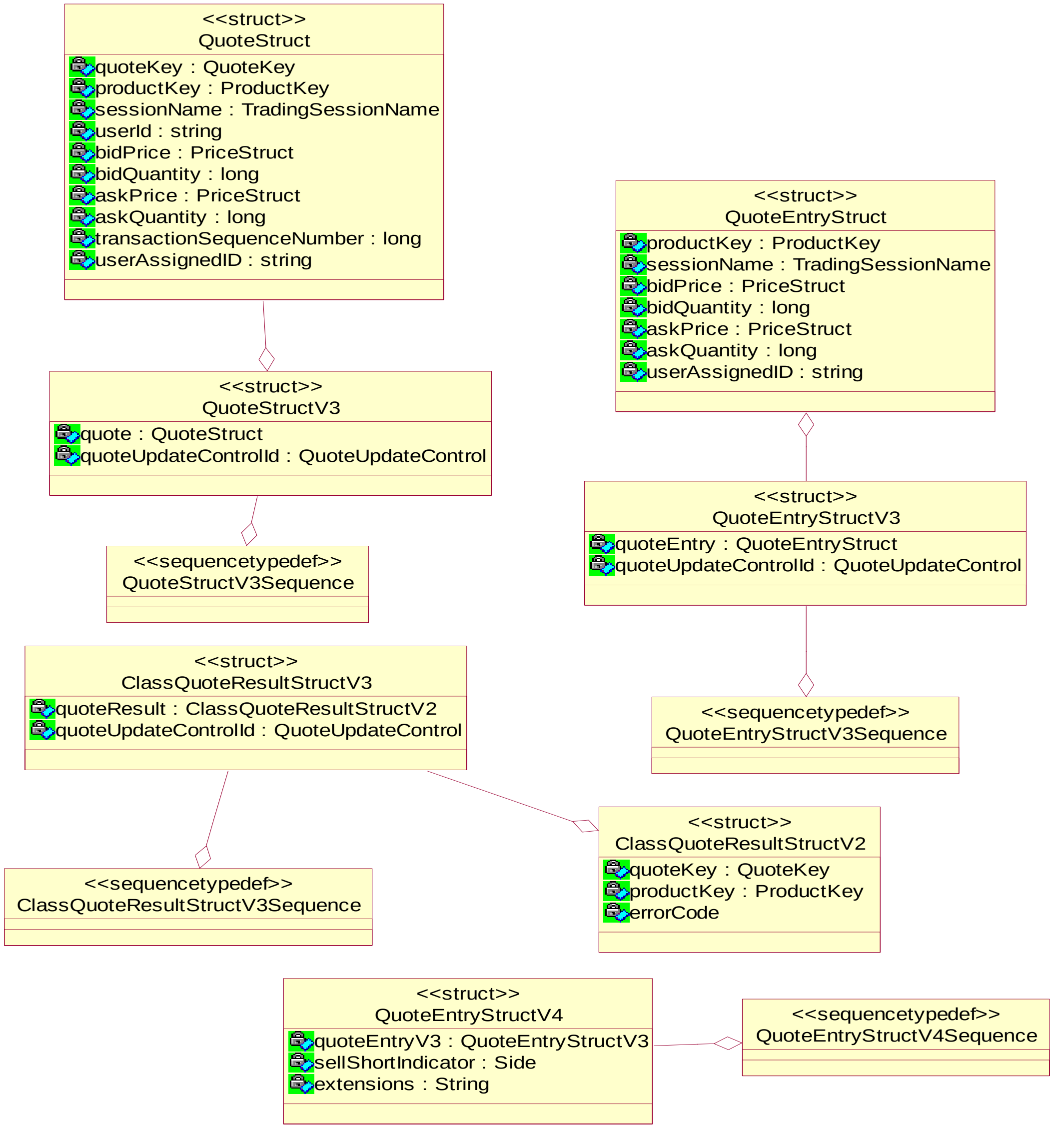

<<typedef>>
RFQType
<<struct>>
RFQStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
quantity : long
timeToLive : long
rfqType : RFQType
entryTime : TimeStruct
<<struct>>
RFQEntryStruct
productKey : ProductKey
sessionName : TradingSessionName
quantity : long
short
(from IDL Primitives)
<<sequencetypedef>>
RFQStructSequence
item
sequence
(from IDL Primitives)


<<struct>>
QuoteFilledReportStruct
quoteKey : QuoteKey
filledReport : FilledReportStructSequence
productKeys : ProductKeyStruct
productName : ProductNameStruct
statusChange : UpdateStatusReason
<<typedef>>
QuoteKey
<<sequencetypedef>>
QuoteFilledReportStructSequence
item
sequence
(from IDL Primitives)
<<struct>>
FilledReportStruct
(from cmiOrder)
tradeId : CboeIdStruct
userAcronym : ExchangeAcronymStruct
extensions : string
fillReportType : ReportType
executingOrGiveUpFirm : ExchangeFirmStruct
userId : string
productKey : ProductKey
sessionName : TradingSessionName
tradedQuantity : long
leavesQuantity : long
price : PriceStruct
side : Side
orsId : string
cmta : ExchangeFirmStruct
account : string
subaccount : string
originator : ExchangeAcronymStruct
optionalData : string
userAssignedId : string
contraParties : ContraPartyStructSequence
timeSent : DateTimeStruct
positionEffect : PositionEffect
transactionSequenceNumber : long
executingBroker : string


<<sequencetypedef>>
QuoteBustReportStructSequence
item
sequence
(from IDL Primitives)
<<struct>>
QuoteBustReportStruct
quoteKey : QuoteKey
productKeys : ProductKeysStruct
productName : ProductNameStruct
bustedReport : BustReportStructSequence
statusChange : UpdateStatusReason
<<struct>>
BustReportStruct
(from cmiOrder)
tradeId : CboeIdStruct
userAcronym : ExchangeAcronymStruct
bustReportType : ReportType
sessionName : TradingSessionName
executingOrGiveUpFirm : ExchangeFirmStruct
userId : string
bustedQuantity : long
price : PriceStruct
productKey : ProductKey
side : Side
timeSent : DateTimeStruct
reinstateRequestedQuantity : long
transactionSequenceNumber : long
<<struct>>
ProductKeysStruct
(from cmiProduct)
productKey : ProductKey
classKey : ClassKey
productType : ProductType
reportingClass : ReportingClassKey
<<struct>>
ProductNameStruct
(from cmiProduct)
reportingClass : Symbol
exercisePrice : PriceStruct
expirationDate : DateStruct
optionType : OptionType
productSymbol : Symbol


<<struct>>
UserQuoteRiskManagementProfileStruct
globalQuoteRiskManagementEnabled : boolean
defaultQuoteRiskProfile : QuoteRiskManagementProfileStruct
quoteRiskProfiles : QuoteRiskManagementProfileStructSequence
<<struct>>
QuoteRiskManagementProfileStruct
classKey : ClassKey
volumeThreshold : long
timeWindow : long
quoteRiskManagementEnabled : boolean
<<sequencetypedef>>
QuoteRiskManagementProfileStructSequence
item
sequence
(from IDL Primitives)


<<struct>>
QuoteDeleteReportStruct
quote : QuoteDetailStruct
deleteReason : ActivityReason
<<struct>>
QuoteCancelReportStruct
quoteKey : QuoteKey
productKeys : ProductKeysStruct
productName : ProductNameStruct
cancelReason : ActivityReason
statusChange : UpdateStatusReason
<<struct>>
ProductNameStruct
(from cmiProduct)
reportingClass : Symbol
exercisePrice : PriceStruct
expirationDate : DateStruct
optionType : OptionType
productSymbol : Symbol
<<sequencetypedef>>
QuoteDeleteReportStructSequence
<<sequencetypedef>>
QuoteCancelReportStructSequence
<<struct>>
ProductKeysStruct
(from cmiProduct)
productKey : ProductKey
classKey : ClassKey
productType : ProductType
reportingClass : ReportingClassKey


<<struct>>
QuoteStruct
quoteKey : QuoteKey
productKey : ProductKey
sessionName : TradingSessionName
userId : string
bidPrice : PriceStruct
bidQuantity : long
askPrice : PriceStruct
askQuantity : long
transactionSequenceNumber : long
userAssignedID : string
<<struct>>
QuoteEntryStruct
productKey : ProductKey
sessionName : TradingSessionName
bidPrice : PriceStruct
bidQuantity : long
askPrice : PriceStruct
askQuantity : long
userAssignedID : string
<<struct>>
QuoteStructV3
quote : QuoteStruct
quoteUpdateControlId : QuoteUpdateControl
<<struct>>
QuoteEntryStructV3
quoteEntry : QuoteEntryStruct
quoteUpdateControlId : QuoteUpdateControl
<<sequencetypedef>>
QuoteStructV3Sequence
<<struct>>
ClassQuoteResultStructV3
quoteResult : ClassQuoteResultStructV2
quoteUpdateControlId : QuoteUpdateControl
<<sequencetypedef>>
QuoteEntryStructV3Sequence
<<struct>>
ClassQuoteResultStructV2
quoteKey : QuoteKey
productKey : ProductKey
errorCode
<<sequencetypedef>>
ClassQuoteResultStructV3Sequence
<<struct>>
QuoteEntryStructV4
quoteEntryV3 : QuoteEntryStructV3
sellShortIndicator : Side
extensions : String
<<sequencetypedef>>
QuoteEntryStructV4Sequence

**Typedefs:**

| | |
|---|---|
| QuoteKey *is a* long | Unique Identifier for a quote that is assigned by the system. |
| | |
| RFQType *is a* short | Type of RFQ - Indicates if the RFQ was submitted manually or via an automated quotation system |

**Typedefs for Sequences:**

ClassQuoteResultStructSequence *is a typedef of a sequence of* ClassQuoteResultStruct

ClassQuoteResultStructV2Sequence *is a typedef of a sequence of* ClassQuoteResultStructV2

ClassQuoteResultStructV3Sequence *is a typedef of a sequence of* ClassQuoteResultStructV3

ClassQuoteResultStructV3Sequence *is a typedef of a sequence of* Quote

LockNotificationStructSequence *is a typedef of a sequence of* LockNotificationStruct

QuoteBustReportStructSequence *is a typedef of a sequence of* QuoteBustReportStruct

QuoteCancelReportStructSequence *is a typedef of a sequence of* QuoteCancelReportStruct

QuoteDeleteReportStructSequence *is a typedef of a sequence of* QuoteDeleteReportStruct

QuoteDetailStructSequence *is a typedef of a sequence of* QuoteDetailStruct

QuoteEntryStructSequence *is a typedef of a sequence of* QuoteEntryStruct

QuoteEntryStructV3Sequence *is a typedef of a sequence of* QuoteEntryStructV3

QuoteEntryStructV4Sequence *is a typedef of a sequence of* QuoteEntryStructV4

QuoteFilledReportStructSequence *is a typedef of a sequence of* QuoteFilledReportStruct

QuoteKeySequence *is a typedef of a sequence of* QuoteKey

QuoteRiskManagementProfileStructSequence *is a typedef of a sequence of* QuoteRiskManagementProfileStruct

QuoteStructSequence *is a typedef of a sequence of* QuoteStruct

QuoteStructV3Sequence *is a typedef of a sequence of* QuoteStructV3

RFQStructSequence *is a typedef of a sequence of* RFQStruct

---

**Structs:**

### ClassQuoteResultStruct

Quotes are submitted as a group of quotes for products belonging to a class - this struct is used to indicate an error in a specific quote from a sequence of quotes that was submitted.

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| errorCode | ErrorCode | Error code if error was encountered for a particular series in the class. Refer to cmiErrorCodes::DataValidationCodes and cmiErrorCodes::TransactionFailedCodes for a list of enumerations that can apply to quote rejection. |
| productKey | ProductKey | Product key of the product for which the quote error was encountered |

### ClassQuoteResultStructV2

Quotes are submitted as a group of quotes for products belonging to a class - this struct is used to indicate an error in a specific quote from a sequence of quotes that was submitted.

| Attribute | Type | Documentation |
|---|---|---|
| errorCode | | |
| productKey | ProductKey | Product key of the product for which the quote error was encountered. |
| quoteKey | QuoteKey | Quote key of the product for which the quote error was encountered. |

### ClassQuoteResultStructV3

Quotes are submitted as a group of quotes for products belonging to a class - this struct is used to indicate an error in a specific quote from a sequence of quotes that was submitted.

| Attribute | Type | Documentation |
|---|---|---|
| quoteResult | ClassQuoteResultStructV2 | Result of the quote. |
| quoteUpdateControlId | QuoteUpdateControl | Identification for quote update. |

### LockNotificationStruct

Notification of locked quote.

| Attribute | Type | Documentation |
|---|---|---|
| buySideUserAcronyms | ExchangeAcronymStructSequence | User's acronym on the Buy side |
| classKey | ClassKey | Class identifier. |
| extensions | string | |

| Attribute | Type | Documentation |
|---|---|---|
| price | PriceStruct | Quote price. |
| productKey | ProductKey | Product identifier. |
| productType | ProductType | Product type identifier. |
| quantity | Long | |
| sellSideUserAcronyms | ExchangeAcronymStructSequence | Sell side user's acronyms. |
| sessionName | TradingSessionName | Trading session identifier. |
| side | Side | Side of the quote. |

**QuoteBustReportStruct**

Report of a trade busted by trading operations.

| Attribute | Type | Documentation |
|---|---|---|
| bustedReport | BustReportStructSequence | Bust report |
| productKeys | ProductKeysStruct | Product keys of the product for which the trade is being busted |
| productName | ProductNameStruct | Product name of the product for which the trade is being busted |
| quoteKey | QuoteKey | Identifier of the quote for which the trade is being busted |
| statusChange | UpdateStatusReason | Quote status as a result of the trade bust (See cmiConstants::StatusUpdateReasons). |

**QuoteCancelReportStruct**

| Attribute | Type | Documentation |
|---|---|---|
| cancelReason | ActivityReason | The reason for the quote cancel |
| productKeys | ProductKeysStruct | Product keys of the product for which the trade is being cancelled. |
| productName | ProductNameStruct | Name of the product for which the trade is being cancelled. |
| quoteKey | QuoteKey | Identifier of the quote for which the trade is being cancelled. |
| statusChange | UpdateStatusReason | Quote status as a result of the cancel |

### QuoteDeleteReportStruct

Report of a quote delete.

| Attribute | Type | Documentation |
|---|---|---|
| deleteReason | ActivityReason | The reason the quote was deleted. |
| quote | QuoteDetailStruct | Details of the quote that was deleted. |

### QuoteDetailStruct

Details for quote in the system for a specific product. This is returned by the system when requested by the user.

| Attribute | Type | Documentation |
|---|---|---|
| productKeys | ProductKeysStruct | Product Keys for the quote being returned |
| productName | ProductNameStruct | Product name for the quote being returned |
| quote | QuoteStruct | Quote |
| statusChange | UpdateStatusReason | Reason that the quote detail message was sent. |

### QuoteEntryStruct

Struct used to enter a quote.

| Attribute | Type | Documentation |
|---|---|---|
| askPrice | PriceStruct | Ask price of the quote |
| askQuantity | long | Ask quantity |
| bidPrice | PriceStruct | Bid price of the quote |
| bidQuantity | long | Bid quantity |
| productKey | ProductKey | Product key for the product being quoted |
| sessionName | TradingSessionName | Name of the trading session for which the quote is to be entered. |
| userAssignedID | string | Optional user identifier for the quote. The identifier will be carried through system and returned on a fill report - if the quote results in a trade. |

### QuoteEntryStructV3

Struct used to enter a quote.

| Attribute | Type | Documentation |
|---|---|---|
| quoteEntry | QuoteEntryStruct | Data required for quote entry. |
| quoteUpdateControlId | QuoteUpdateControl | Identification for a quote update. |

### QuoteEntryStructV4

Used to enter a quote

| Attribute | Type | Documentation |
|---|---|---|
| extensions | String | Used to persist arbitrary data along with the quote. |
| quoteEntryV3 | QuoteEntryStructV3 | Data required for quote entry.  References the QuoteEntryStructV3. |
| sellShortIndicator | Side | Used to indicate the short sale position on the individual quote or on each of the quotes of the mass quotes. |

**QuoteFilledReportStruct**

Fill report for a quote

| Attribute | Type | Documentation |
|---|---|---|
| filledReport | FilledReportStructSequence | Fill report for the quote |
| productKeys | ProductKeyStruct | Product keys of the product being quoted |
| productName | ProductNameStruct | Name of the product being quoted |
| quoteKey | QuoteKey | Unique identifier for the quote that was filled |
| statusChange | UpdateStatusReason | Quote status as a result of this report. Ignore this value, using instead the statusChange value in the QuoteDetailStruct. |

**QuoteRiskManagementProfileStruct**

Risk management parameters for quotes issued for a class

| Attribute | Type | Documentation |
|---|---|---|
| classKey | ClassKey | Class that the quote risk management parameters are to be applied |
| quoteRiskManagementEnabled | boolean | Set to true, quote management is enabled for this class. |

| Attribute | Type | Documentation |
|---|---|---|
| timeWindow | long | The time window for the quote risk management threshold is expressed in milliseconds. |
| volumeThreshold | long | Volume threshold for the class |

### QuoteStruct

Quote struct created by the system after receiving a valid quote entry struct from a user.

| Attribute | Type | Documentation |
|---|---|---|
| askPrice | PriceStruct | Ask price of the quote |
| askQuantity | long | Ask quantity of the quote |
| bidPrice | PriceStruct | Bid price of the quote |
| bidQuantity | long | Bid quantity of the quote |
| productKey | ProductKey | Unique identifier for a quote in the system |
| quoteKey | QuoteKey | Unique identifier for a quote that is assigned by the system |
| sessionName | TradingSessionName | Name of the trading session for which the quote is applicable. |
| transactionSequenceNumber | long | Sequence number assigned by server to guarantee uniqueness of message and ordering between CAS and Exchange services. |
| userAssignedID | string | User supplied identifier. Provided for FIX compatibility. |
| userId | string | Identifies the user that submitted the quote |

### QuoteStructV3

Quote struct created by the system after receiving a valid quote entry struct from a user.

| Attribute | Type | Documentation |
|---|---|---|
| quote | QuoteStruct | Data required for quote entry. |
| quoteUpdateControlId | QuoteUpdateControl | Identification for quote update. |

### RFQEntryStruct

Struct used to enter a Request for Quote

| Attribute | Type | Documentation |
|---|---|---|
| productKey | ProductKey | Unique identifier of the product for which an RFQ is requested |
| quantity | long | Quantity of the order |
| sessionName | TradingSessionName | Name of trading session for which the quote is being requested. |

### RFQStruct

Request For Quote struct that is sent to subscribers for RFQs

| Attribute | Type | Documentation |
|---|---|---|
| entryTime | TimeStruct | Time the RFQ was entered |
| productKeys | ProductKeysStruct | Product keys for the product for which an RFQ is requested |
| quantity | long | Quantity for which the quote is requested |
| rfqType | RFQType | Indicates if the RFQ was generated manually or by an automated quotation system |
| sessionName | TradingSessionName | Name of the trading session for which the request for quote has been requested. |

| Attribute | Type | Documentation |
|---|---|---|
| timeToLive | long | Time the RFQ is valid. This is the response window assigned by the exchange. |

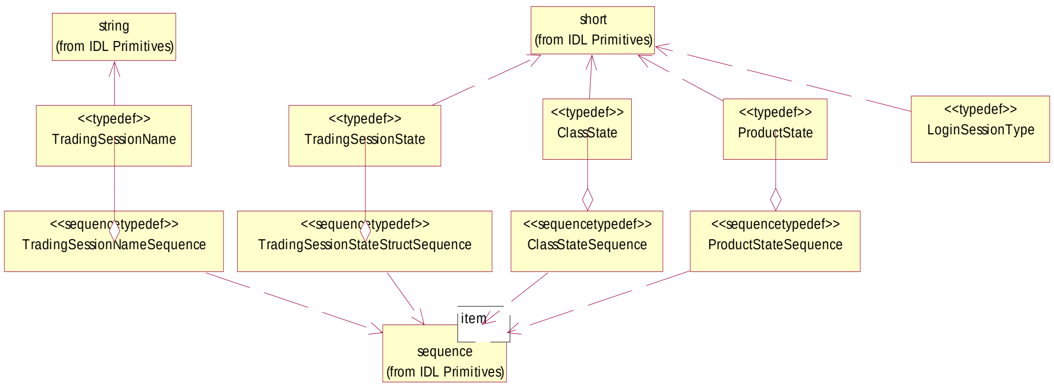
**UserQuoteRiskManagementProfileStruct**

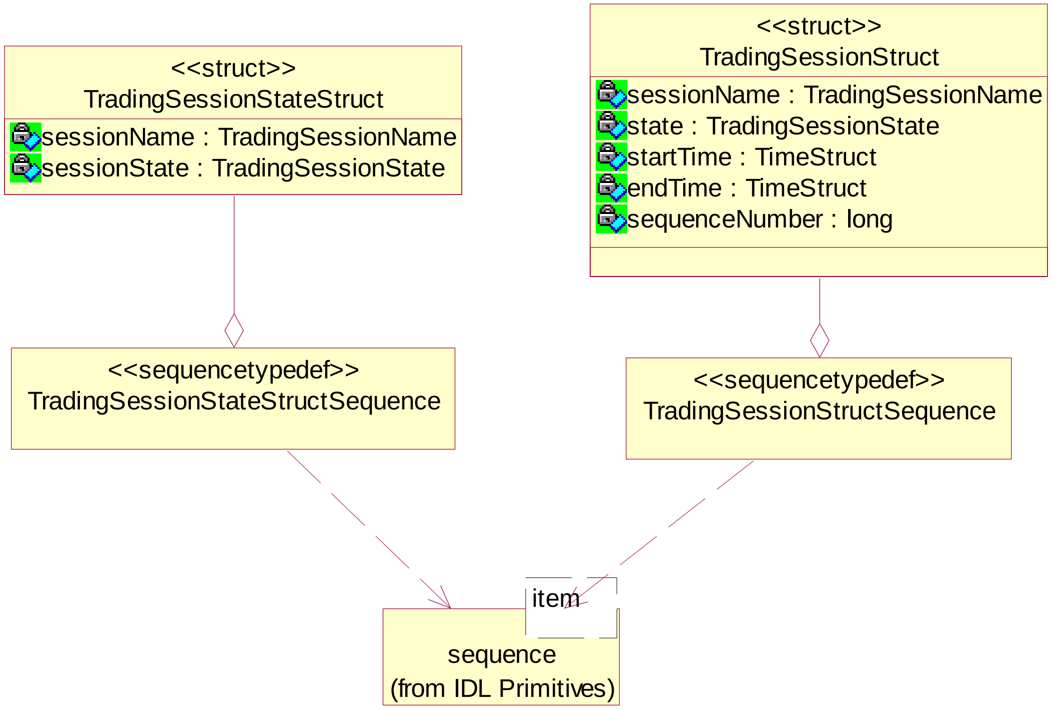
Risk quote management profile for a user that consists of default risk profile and a sequence of risk quote profiles for classes that the user may quote.

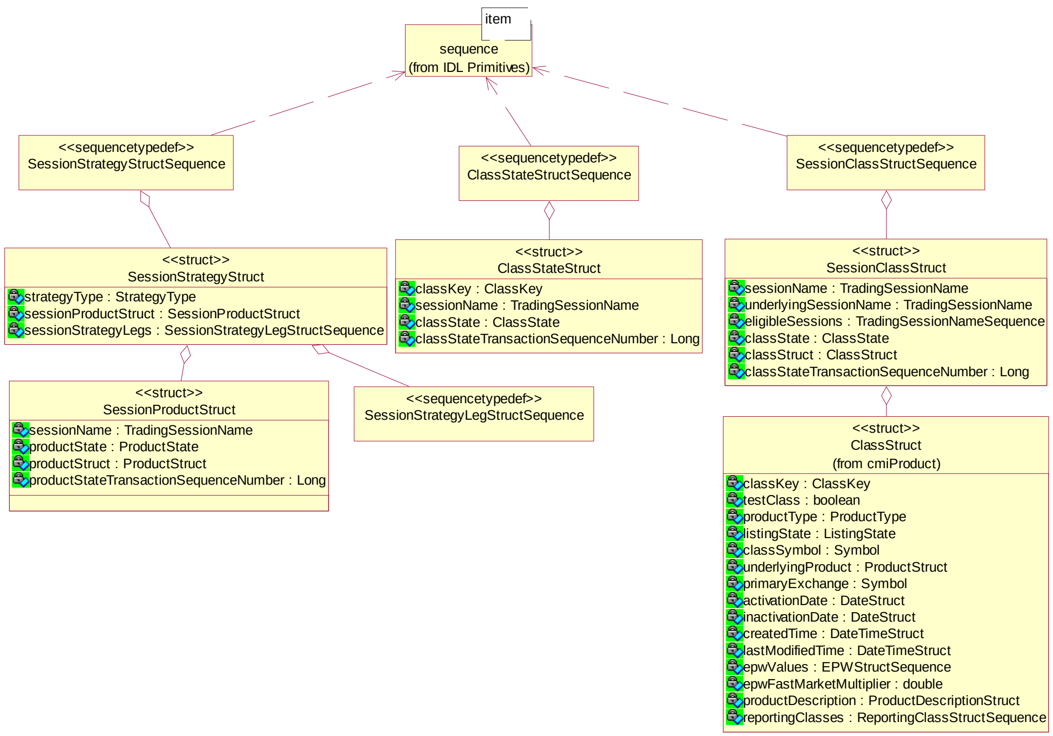
| Attribute | Type | Documentation |
|---|---|---|
| defaultQuoteRiskProfile | QuoteRiskManagementProfileStruct | Default risk quote management profile |
| globalQuoteRiskManagementEnabled | boolean | Indicates if use is using quote risk management |
| quoteRiskProfiles | QuoteRiskManagementProfileStruct Sequence | Sequence of quote risk management profiles for classes beign quoted by the user |

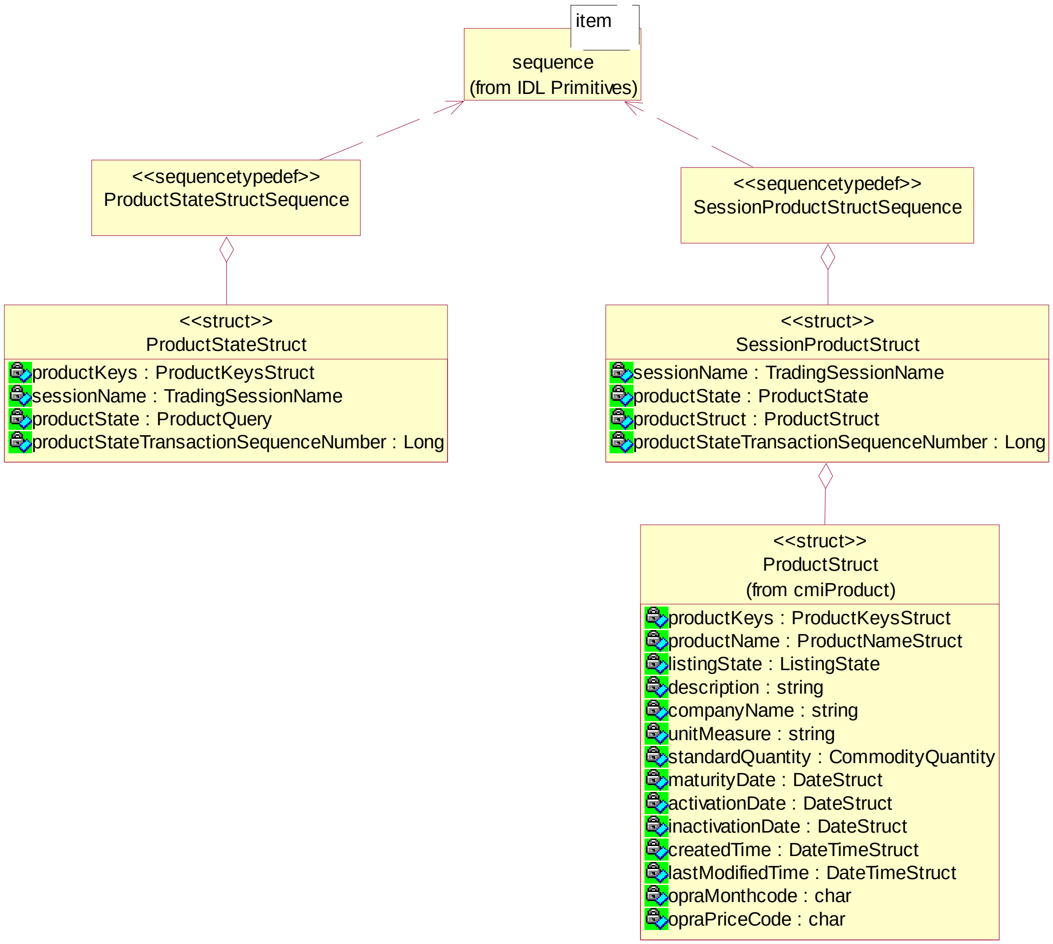
## *Module cmiSession*

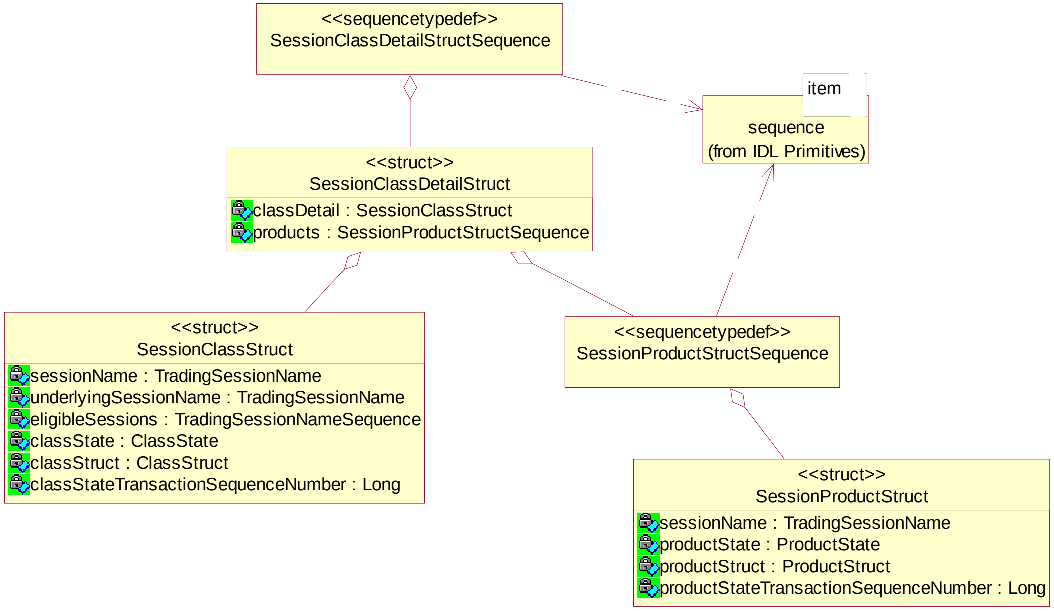
Messages from the TradingSession service. The TradingSession service is used to determine which products trade in a given session. The service is also used to obtain session and product state information.




<<struct>>
TradingSessionStateStruct
sessionName : TradingSessionName
sessionState : TradingSessionState
<<struct>>
TradingSessionStruct
sessionName : TradingSessionName
state : TradingSessionState
startTime : TimeStruct
endTime : TimeStruct
sequenceNumber : long
<<sequencetypedef>>
TradingSessionStateStructSequence
<<sequencetypedef>>
TradingSessionStructSequence
item
sequence
(from IDL Primitives)


item
sequence
(from IDL Primitives)
<<sequencetypedef>>
SessionStrategyStructSequence
<<sequencetypedef>>
ClassStateStructSequence
<<sequencetypedef>>
SessionClassStructSequence
<<struct>>
SessionStrategyStruct
strategyType : StrategyType
sessionProductStruct : SessionProductStruct
sessionStrategyLegs : SessionStrategyLegStructSequence
<<struct>>
ClassStateStruct
classKey : ClassKey
sessionName : TradingSessionName
classState : ClassState
classStateTransactionSequenceNumber : Long
<<struct>>
SessionClassStruct
sessionName : TradingSessionName
underlyingSessionName : TradingSessionName
eligibleSessions : TradingSessionNameSequence
classState : ClassState
classStruct : ClassStruct
classStateTransactionSequenceNumber : Long
<<struct>>
SessionProductStruct
sessionName : TradingSessionName
productState : ProductState
productStruct : ProductStruct
productStateTransactionSequenceNumber : Long
<<sequencetypedef>>
SessionStrategyLegStructSequence
<<struct>>
ClassStruct
(from cmiProduct)
classKey : ClassKey
testClass : boolean
productType : ProductType
listingState : ListingState
classSymbol : Symbol
underlyingProduct : ProductStruct
primaryExchange : Symbol
activationDate : DateStruct
inactivationDate : DateStruct
createdTime : DateTimeStruct
lastModifiedTime : DateTimeStruct
epwValues : EPWStructSequence
epwFastMarketMultiplier : double
productDescription : ProductDescriptionStruct
reportingClasses : ReportingClassStructSequence


item
sequence
(from IDL Primitives)
<<sequencetypedef>>
ProductStateStructSequence
<<sequencetypedef>>
SessionProductStructSequence
<<struct>>
ProductStateStruct
productKeys : ProductKeysStruct
sessionName : TradingSessionName
productState : ProductQuery
productStateTransactionSequenceNumber : Long
<<struct>>
SessionProductStruct
sessionName : TradingSessionName
productState : ProductState
productStruct : ProductStruct
productStateTransactionSequenceNumber : Long
<<struct>>
ProductStruct
(from cmiProduct)
productKeys : ProductKeysStruct
productName : ProductNameStruct
listingState : ListingState
description : string
companyName : string
unitMeasure : string
standardQuantity : CommodityQuantity
maturityDate : DateStruct
activationDate : DateStruct
inactivationDate : DateStruct
createdTime : DateTimeStruct
lastModifiedTime : DateTimeStruct
opraMonthcode : char
opraPriceCode : char




## Typedefs:

| | |
|---|---|
| ClassState *is a* short | State of class. State is currently maintained at the product (series) level. An interface to report state at the class level is provided for use with CBOE future CBOE markets. |

| | |
|---|---|
| LoginSessionType *is a* UserAccessV6<br>LoginSessionType *is a* UserAccessV3<br>LoginSessionType *is a* UserAccessV4<br>LoginSessionType *is a* short | Indicates type of login being requested - either primary or secondary. Supports login to multiple CASs for fail over. |
| ProductState *is a* short | State of a product within a trading session (such as, Closed, Open, etc.) (See cmiConstants::ProductStates). |
| TradingSessionName *is a* Quote<br>TradingSessionName *is a* FloorTradeMaintenanceService<br>TradingSessionName *is a* MarketQuery<br>TradingSessionName *is a* string<br>TradingSessionName *is a* AuctionStruct | Name of the trading session. The trading session names are the same for both the CMi and the FIX interfaces. There is no enumeration for trading session names in the cmiConstants module because trading sessions are subject to change as trading at CBOE evolves. The getCurrentTradingSessions() operation is provided in cmi::Sessions to return the list of available trading session. |
| TradingSessionState *is a* short | Trading Session State (Open or Closed) (See cmiConstants::TradingSessionStates). |

**Typedefs for Sequences:**

ClassStateSequence *is a typedef of a sequence of* ClassState

ClassStateStructSequence *is a typedef of a sequence of* ClassStateStruct

ProductStateSequence *is a typedef of a sequence of* ProductState

ProductStateStructSequence *is a typedef of a sequence of* ProductStateStruct

SessionClassDetailStructSequence *is a typedef of a sequence of* SessionClassDetailStruct

SessionClassStructSequence *is a typedef of a sequence of* SessionClassStruct

SessionProductStructSequence *is a typedef of a sequence of* SessionProductStruct

SessionStrategyStructSequence *is a typedef of a sequence of* SessionStrategyStruct

TradingSessionNameSequence *is a typedef of a sequence of* TradingSessionName

TradingSessionStateStructSequence *is a typedef of a sequence of* TradingSessionStateStruct

TradingSessionStateStructSequence *is a typedef of a sequence of* TradingSessionState

TradingSessionStructSequence *is a typedef of a sequence of* TradingSessionStruct

---

**Structs:**

### ClassStateStruct

Message used to report a change in the state of a class that is active on a trading session.

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| classKey | ClassKey | Unique identifier of the class |
| classState | ClassState | State of the class. NOT CURRENTLY IN USE. |
| classStateTransactionSequenceNumber | Long | Sequence number used to order class status information. |
| sessionName | TradingSessionName | Name of the trading session where the class is active. |

### ProductStateStruct

Message used to report product state change

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| productKeys | ProductKeysStruct | Product keys for the product whose state is being reported. |

| Attribute | Type | Documentation |
|---|---|---|
| productState | ProductQuery | State of the product (open, closed, halted, etc.) |
| productStateTransactionSequenceNumber | Long | Transaction sequence number used to sequence product state reports across product state change messages and session product messages. |
| sessionName | TradingSessionName | Name of the session for which the product state is applicable. |

### SessionClassDetailStruct

Message the returns all classes and all products for that class that trade on a specific trading session.

| Attribute | Type | Documentation |
|---|---|---|
| classDetail | SessionClassStruct | Session class struct for a class that trades on a specific trading session. |
| products | SessionProductStructSequence | The list of products for the class that trades on a specific trading session. It is important to note that the system is capable of having different products eligible for different trading sessions. |

### SessionClassStruct

Message used to report classes that trade during a specific trading session.

| Attribute | Type | Documentation |
|---|---|---|
| classState | ClassState | NOT IMPLEMENTED AT THIS TIME. |
| classStateTransactionSequenceNumber | Long | Transaction sequence number used for sequencing class state reports across SessionClassStruct and ClassStateStruct messages. |

| Attribute | Type | Documentation |
|---|---|---|
| classStruct | ClassStruct | Attributes of the class |
| eligibleSessions | TradingSessionNameSequence | List of all trading sessions for which this class is eligible for trading. |
| sessionName | TradingSessionName | Name of the trading session which the class trades upon. |
| underlyingSessionName | TradingSessionName | Name of the session where the underlying trades. |

**SessionManagerStructV2**

| Attribute | Type | Documentation |
|---|---|---|
| sessionManager | UserSessionManager | |
| sessionManagerV2 | UserSessionManagerV2 | |

**SessionProductStruct**

Message used to report products that trade during a specific trading session. Also contains the current product state.

| Attribute | Type | Documentation |
|---|---|---|
| productState | ProductState | State of the product (open, closed, halted, etc.) |
| productStateTransactionSequenceNumber | Long | Transaction sequence number used to sequence product state reports in a client application across SessionProductStruct and ProductStateStructs. |
| productStruct | ProductStruct | Attributes of the product |

| Attribute | Type | Documentation |
|---|---|---|
| sessionName | TradingSessionName | Name of the trading session upon which the product trades. This is the trading session that was queried and also the trading session for which the product state is applicable. |

**SessionStrategyLegStruct**

Message used to report strategy legs that trade during a specific trading session.

| Attribute | Type | Documentation |
|---|---|---|
| productKey | ProductKey | Identifier of the product. |
| ratioQuantity | Long | The quanity of this product that is required to make up the overall strategy. |
| sessionName | TradingSessionName | The trading session name where the leg is being traded. |
| side | Side | The side of the leg product. |

**SessionStrategyStruct**

Message that reports strategies that trades on a specific trading session.

| Attribute | Type | Documentation |
|---|---|---|
| sessionProductStruct | SessionProductStruct | Attributes of the product. |
| sessionStrategyLegs | SessionStrategyLegStructSequence | Individual products that make up the strategy. |
| strategyType | StrategyType | The type of strategy eligible for trading in a trading session. |

### TradingSessionStateStruct

Message sent to the subscribed user reporting a change in  trading session status.

| Attribute | Type | Documentation |
|---|---|---|
| sessionName | TradingSessionName | Name of the trading session for which the state change is being reported. |
| sessionState | TradingSessionState | New Trading Session State. |

### TradingSessionStruct

Detailed information (duration, start and end time, name) of a trading session and a list of the classes that will trade on that session.

| Attribute | Type | Documentation |
|---|---|---|
| endTime | TimeStruct | End of session time. |
| sequenceNumber | long | Sequence number used to guarantee uniqueness and order of message between CAS and Exchange Services. |
| sessionName | TradingSessionName | Name of the trading session. |
| startTime | TimeStruct | Starting time of trading session. |
| state | TradingSessionState | Trading Session State (See cmiConstants::TradingSessionStates). |

# *Module cmiTrade*

Contains structs and types required to support external trades.


<<sequencetypedef>>
ExternalBustTradeStructSequence
<<struct>>
ExternalTradeEntryStruct
quantity : long
price : PriceStruct
sessionName : TradingSessionName
productKey : ProductKey
theTradeSource : TradeSource
handlingInstruction : short
name
externalTradeType : TradeType
bustable : Boolean
businessDate : DateStruct
timeTraded : DateTimeStruct
parties : ExternalBustTradeStructSequence
settlementDate : DateStruct
transactionTime : DateTimeStruct
asOffFlag : Boolean
<<struct>>
FloorTradeEntryStruct
sessionName : TradingSessionName
productKey : ProductKey
quantity : long
price : PriceStruct
side : Side
account : String
subaccount : String
cmta : ExchangeFirmStruct
executingMarketMaker : ExchangeAcronymStruct
firm : ExchangeFirmStruct
positionEffect : char
contraBroker : ExchangeAcronymStruct
contraFirm : ExchangeFirmStruct
timeTraded : DateTimeStruct
optionalData : String
<<struct>>
ExternalAtomicTradeEntryStruct
entryTime : DateTimeStruct
entryType : TradeReportEntryType
quantity : Long
sessionName : String
buyerOriginType : char
buyerCmta : ExchangeFirmStruct
buyerPositionEffect : char
buyerAccount : string
buyerSubaccount : string
buyerBroker : ExchangeAcronymStruct
buyerOriginator : ExchangeAcronymStruct
buyerFirm : ExchangeFirmStruct
buyerOptionalData : string
sellerOriginType : char
sellerCmta : ExchangeFirmStruct
sellerPositionEffect : char
sellerAccount : string
sellerSubaccount : String
sellerBroker : ExchangeAcronymStruct
sellerOriginator : ExchangeAcronymStruct
sellerFirm : ExchangeFirmStruct
sellerOptionalData : string
<<struct>>
ExternalAtomicTradeResultStruct
atomicTradeId : CboeIdStruct
matchedSequenceNumber : Long
active : boolean
entryTime : DateTimeStruct
entryType : TradeReportEntryType
lastUpdateTime : DateTimeStruct
lastEntryType : TradeReportEntryType
quantity : Long
sessionName : String
buyerOriginType : char
buyerFirmBranch : string
buyerFirmBranchSequenceNumber : Long
buyerCmta : ExchangeFirmStruct
buyerCorrespondentId : string
buyerPositionEffect : char
buyerAccount : string
buyerSubaccount : string
buyerBroker : ExchangeAcronymStruct
buyerOriginator : ExchangeAcronymStruct
buyerFirm : ExchangeFirmStruct
buyerOptionalData : string
buyerOrderOrQuoteKey : CboeIdStruct
buyerOrderOrQuote : boolean
reinstatableForBuyer : boolean
sellerOriginType : char
sellerFirmBranch : String
sellerFirmBranchSequenceNumber
sellerCmta : ExchangeFirmStruct
sellerCorrespondentId : string
sellerPositionEffect : char
sellerAccount : string
sellerSubaccount : string
sellerBroker : ExchangeAcronymStruct
sellerOriginator : ExchangeAcronymStruct
sellerFirm : ExchangeFirmStruct
sellerOptionalData : string
sellerOrderOrQuoteKey : CboeIdStruct
sellerOrderOrQuote : boolean
reinstatableForSeller : boolean
<<struct>>
ExternalTradeReportStruct
quantity : Long
price : PriceStruct
sessionName : TradingSessionName
productKey : ProductKey
theTradeSource : TradeSource
tradeId : CboeIdStruct
externalTradeType : TradeType
bustable : Boolean
businessDate : DateStruct
timeTraded : DateTimeStruct
parties : ExternalAtomicTradeResultsStructSequence
settlementDate : DateStruct
transactionTime : DateTimeStruct
asOfFlag : Boolean
<<sequencetypedef>>
ExternalAtomicTradeResultStructSequence

**Typedefs:**

| | |
|---|---|
| | |

**Typedefs for Sequences:**

ExternalAtomicTradeResultStructSequence *is a typedef of a sequence of* ExternalTradeReportStruct

ExternalAtomicTradeResultStructSequence *is a typedef of a sequence of* ExternalAtomicTradeResultStruct

ExternalBustTradeStructSequence *is a typedef of a sequence of* ExternalTradeEntryStruct

**Structs:**

**ExternalAtomicTradeEntryStruct**

NOT CURRENTLY SUPPORTED

Contains external trade fields for EFPs and block trades.

| Attribute | Type | Documentation |
|---|---|---|
| buyerAccount | string | The buyer side account number. |
| buyerBroker | ExchangeAcronymStruct | Buyer side broker acronym. |
| buyerCmta | ExchangeFirmStruct | The CMTA for the buyer side of the trade. |
| buyerFirm | ExchangeFirmStruct | Buyer side Firm acronym. |
| buyerOptionalData | string | Buyer side optional data. This is an optional field. |
| buyerOriginator | ExchangeAcronymStruct | Buyer side originator acronym. |
| buyerOriginType | char | The buyer side origin. |
| buyerPositionEffect | char | The buyer side position. |
| buyerSubaccount | string | Buyer side subaccount |
| entryTime | DateTimeStruct | The time the external trade was entered. |
| entryType | TradeReportEntryType | Type of trade entered in the system. |
| quantity | Long | Trade quantity. |
| sellerAccount | string | The seller's account number. |
| sellerBroker | ExchangeAcronymStruct | The seller side broker acronym. |
| sellerCmta | ExchangeFirmStruct | The seller side CMTA. |
| sellerFirm | ExchangeFirmStruct | Seller side Firm acronym. |
| sellerOptionalData | string | Seller side optional data. This field is optional. |
| sellerOriginator | ExchangeAcronymStruct | Seller side originator acronym. |
| sellerOriginType | char | The seller side origin. |
| sellerPositionEffect | char | The seller side position. |
| sellerSubaccount | String | The seller side subaccount. |
| sessionName | String | The name of the trading session where the trade took place. |

**ExternalAtomicTradeResultStruct**

NOT CURRENTLY SUPPORTED

Contains external trade result fields for EFPs and block trades.

| Attribute | Type | Documentation |
|---|---|---|
| active | boolean | Shows if trade is active. |
| atomicTradeId | CboeIdStruct | Atomic trade identification. |
| buyerAccount | string | The buyer side account number. |
| buyerBroker | ExchangeAcronymStruct | The buyer side broker acronym. |
| buyerCmta | ExchangeFirmStruct | The CMTA for the buyer side of the trade. |
| buyerCorrespondentId | string | The buyer side correspondent ID. |
| buyerFirm | ExchangeFirmStruct | The buyer side Firm acronym. |
| buyerFirmBranch | string | The buyer side Firm branch number. Used for orders only. |
| buyerFirmBranchSequenceNumber | Long | The buyer side branch sequence number. Used for orders only. |
| buyerOptionalData | string | The buyer side optional data field. |
| buyerOrderOrQuote | boolean | The buyer side order or quote. |
| buyerOrderOrQuoteKey | CboeIdStruct | The buyer side order or quote identifier. |
| buyerOriginator | ExchangeAcronymStruct | The buyer side originator acronym. |
| buyerOriginType | char | The buyer side origin. |
| buyerPositionEffect | char | The buyer side position. |
| buyerSubaccount | string | The buyer side subaccount. |
| entryTime | DateTimeStruct | Trade entry time. |
| entryType | TradeReportEntryType | Type of trade entered in the system. |

| Attribute | Type | Documentation |
|---|---|---|
| lastEntryType | TradeReportEntryType | Collection of last entry types. |
| lastUpdateTime | DateTimeStruct | Last time the trade was updated. |
| matchedSequenceNumber | Long | The sequence number for the match trade. |
| quantity | Long | The trade quantity. |
| reinstatableForBuyer | boolean | Sets buyer side reinstateable trade. |
| reinstatableForSeller | boolean | Sets seller side reinstatable trade. |
| sellerAccount | string | The seller side account. |
| sellerBroker | ExchangeAcronymStruct | The seller side broker acronym. |
| sellerCmta | ExchangeFirmStruct | Seller side CMTA. |
| sellerCorrespondentId | string | Seller side correspondent identification. |
| sellerFirm | ExchangeFirmStruct | The seller side Firm acronym. |
| sellerFirmBranch | String | The seller side Firm branch number. Used for orders only. |
| sellerFirmBranchSequenceNumber | | Seller side Firm branch sequence number. Used for orders only. |
| sellerOptionalData | string | The seller side optional data. |
| sellerOrderOrQuote | boolean | Seller side order or quote. |
| sellerOrderOrQuoteKey | CboeIdStruct | Seller side order or quote identifier. |
| sellerOriginator | ExchangeAcronymStruct | The seller side originator acronym. |
| sellerOriginType | char | Seller side origin. |
| sellerPositionEffect | char | Seller side position. |
| sellerSubaccount | string | The seller side subaccount. |
| sessionName | String | The name of the trading session where the trade took place. |

**ExternalBustTradeStruct**

NOT CURRENTLY SUPPORT

Bust trade fields for EFPs and Block trades.

| Attribute | Type | Documentation |
|---|---|---|
| atomicTradeId | CboeIdStruct | Identifier of the atomic trade. |
| bustedQuantity | long | The bust trade quantity. |
| buyerReinstateRequested | Boolean | Sets buyer trade reinstate. |
| sellerReinstateRequested | Boolean | Sets buyer trade reinstate. |

**ExternalTradeEntryStruct**

NOT CURRENTLY SUPPORTED.

Trade entry struct for external trade.

| Attribute | Type | Documentation |
|---|---|---|
| asOffFlag | Boolean | |
| businessDate | DateStruct | The business day of the external trade. |
| bustable | Boolean | Defines external trade as bustable. |
| externalTradeType | TradeType | Type of external trade. |
| handlingInstruction | short | Specific trade instructions. |
| name | | |

| Attribute | Type | Documentation |
|---|---|---|
| parties | ExternalBustTradeStructSequence | Trade participants. |
| price | PriceStruct | The price of the external trade. |
| productKey | ProductKey | Product identifier. |
| quantity | long | The quantity amount of the trade. |
| sessionName | TradingSessionName | The trading session for the external trade entry. |
| settlementDate | DateStruct | The trade settlement date. |
| theTradeSource | TradeSource | External trade source. |
| timeTraded | DateTimeStruct | The time of the trade. |
| transactionTime | DateTimeStruct | The time of the trade transaction. |

**ExternalTradeReportStruct**

Contains external trade report fields.

| Attribute | Type | Documentation |
|---|---|---|
| asOfFlag | Boolean | |
| businessDate | DateStruct | The date the trade occurred. |
| bustable | Boolean | Sets the trade to be bustable or unbustable. |
| externalTradeType | TradeType | The type of external trade. |
| parties | ExternalAtomicTradeResultsStructSequence | Trade participants. |
| price | PriceStruct | The external trade price. |
| productKey | ProductKey | Product identifier. |
| quantity | Long | The trade quantity for the external trade report. |

| Attribute | Type | Documentation |
|---|---|---|
| sessionName | TradingSessionName | The trading session where the trade occurred. |
| settlementDate | DateStruct | The trade settlement date. |
| theTradeSource | TradeSource | External trade source. |
| timeTraded | DateTimeStruct | The time the trade occurred. |
| tradeId | CboeIdStruct | Trade identification. |
| transactionTime | DateTimeStruct | The time the trade transaction occurred. |

### FloorTradeEntryStruct

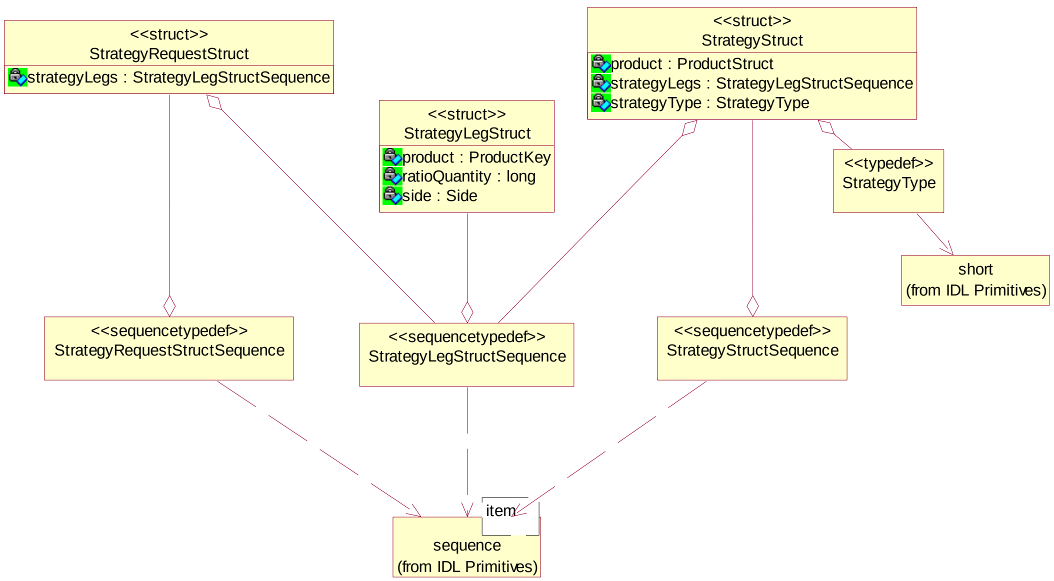
Trade entry struct for trading floor trades (i.e. MMHH trades).

| Attribute | Type | Documentation |
|---|---|---|
| account | String | Firm account number. |
| cmta | ExchangeFirmStruct | The Firm's CMTA. |
| contraBroker | ExchangeAcronymStruct | The acronym of the Contra Broker. |
| contraFirm | ExchangeFirmStruct | The Firm acronym of the Contra Broker. |
| executingMarketMaker | ExchangeAcronymStruct | The acronym of the Market Maker executing the trade. |
| firm | ExchangeFirmStruct | Firm acronym. |
| optionalData | String | Optional data field |
| positionEffect | char | |
| price | PriceStruct | The trade price. |
| productKey | ProductKey | Product identifier. |
| quantity | long | Trade quantity. |
| sessionName | TradingSessionName | The trading session where the floor trade occurred. |

| Attribute | Type | Documentation |
|---|---|---|
| side | Side | The buy or sell side of the trade . |
| subaccount | String | Firm subaccount. |
| timeTraded | DateTimeStruct | The time the floor trade occurred. |

## *Module cmiStrategy*

Contains structs and types required to support trading derivative strategies.



**Typedefs:**

| | |
|---|---|
| MonthIncrement is a typedef | This field is the month increment for the strategy product. |
| StrategyType *is a* short | |

**Typedefs for Sequences:**

StrategyLegStructSequence *is a typedef of a sequence of* StrategyLegStruct

StrategyLegStructSequence *is a typedef of a sequence of* ProductDefinition

StrategyRequestStructSequence *is a typedef of a sequence of* StrategyRequestStruct

StrategyStructSequence *is a typedef of a sequence of* StrategyStruct

**Structs:**

**StrategyLegStruct**

The StrategyLegStruct is used to define component products that make up a Strategy product - such as a spread or combination.

Definition of one product that makes up a strategy product. Each leg of a strategy must also include the ratio of this product to the other products in the strategy is defined and whether the product is bought or sold as part of this strategy product.

| Attributes: | Type | Documentation |
|---|---|---|
| product | ProductKey | Product to be included as a leg of a strategy product |
| ratioQuantity | long | Ratio of this product to the other products in the strategy |
| side | Side | Is the product to be bought or sold as part of this strategy product |

**StrategyRequestStruct**

Contains a series of products with market side (Buy, Sell) and a ratio to the other products that are to be defined as a strategy product that is traded atomically as a new product.

| Attributes: | Type | Documentation |
|---|---|---|
| strategyLegs | StrategyLegStructSequence | Sequence of strategy legs - with one product per leg |

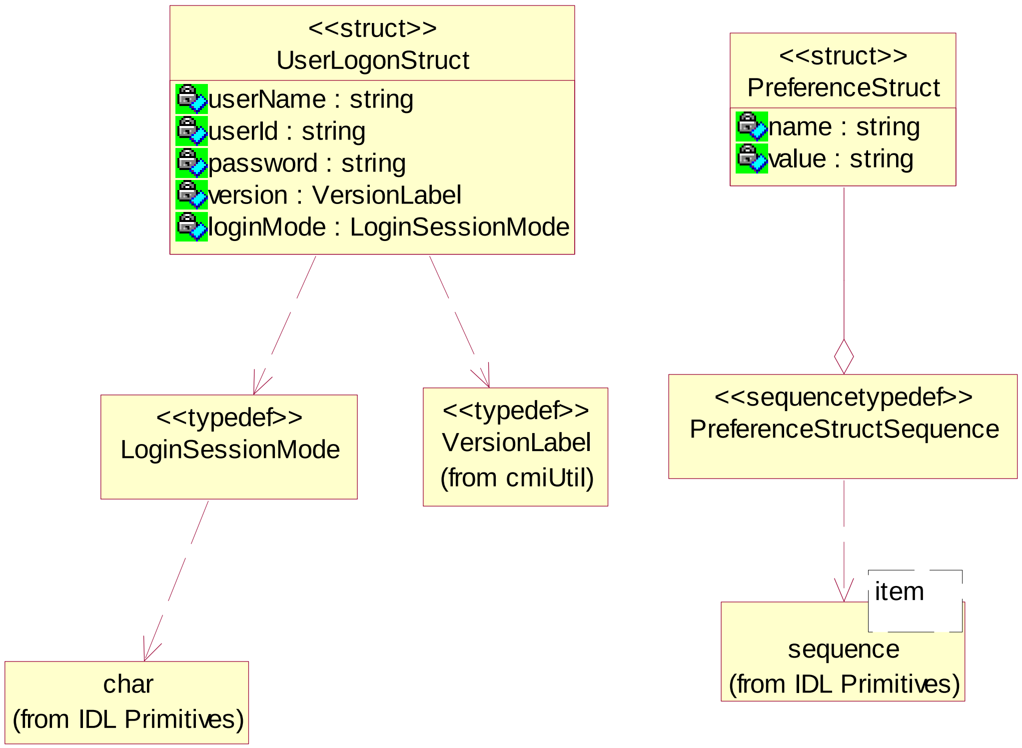
**StrategyStruct**

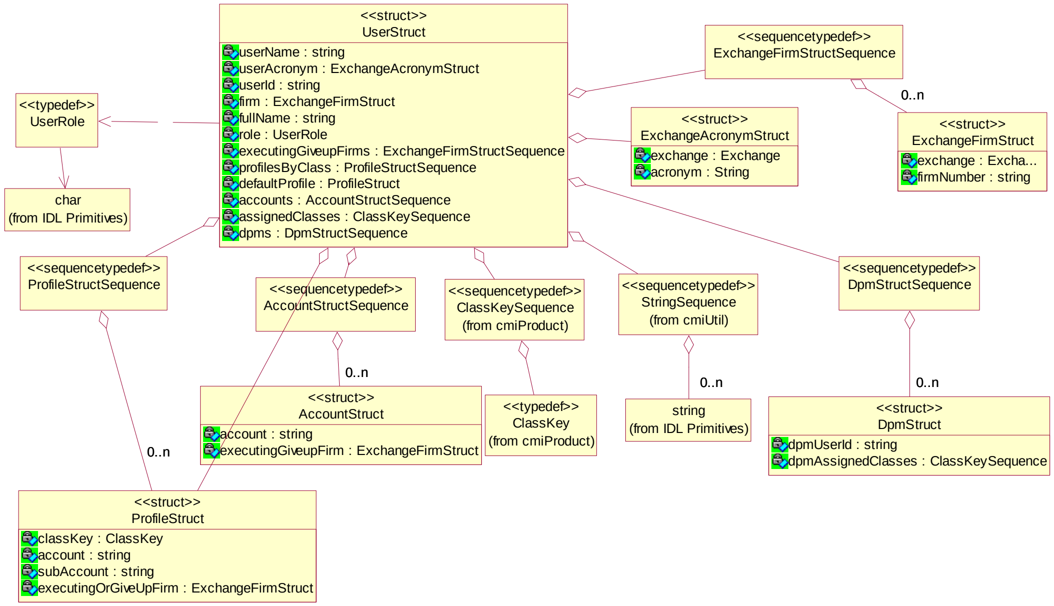
Strategy Product available for trading within the system. Contains the list of products that make up the strategy.

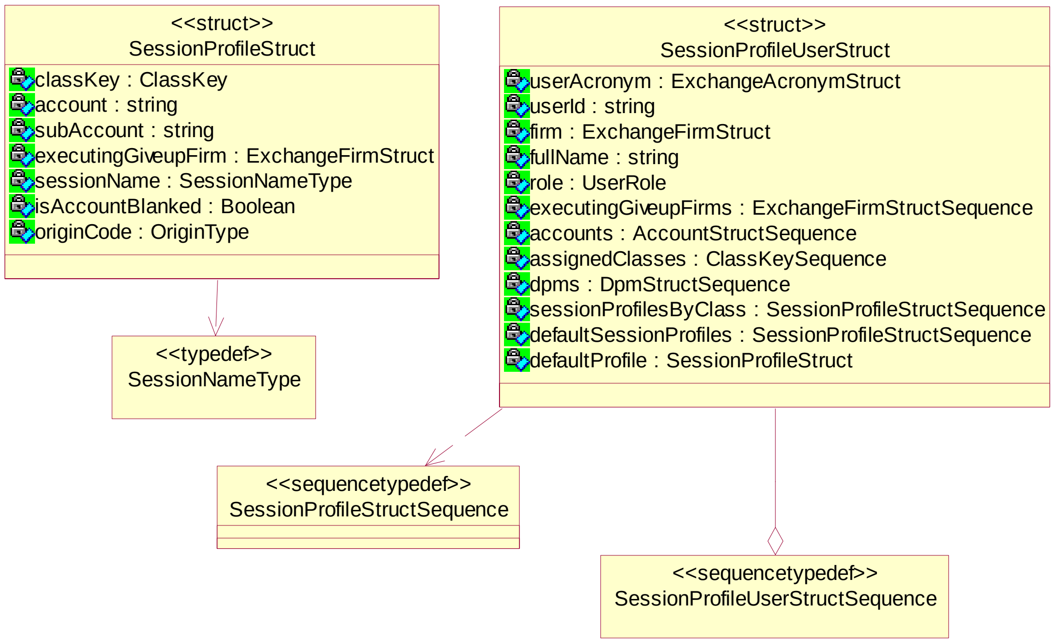
| Attributes: | Type | Documentation |
|---|---|---|
| product | ProductStruct | Product information for the Strategy product |
| strategyLegs | StrategyLegStructSequence | Sequence of strategy legs - one for each product that makes up the strategy product |
| strategyType | StrategyType | Defines the type of strategy (spread, combination, etc). |

# *Module cmiUser*

User preferences and privileges




<<struct>>
UserStruct
userName : string
userAcronym : ExchangeAcronymStruct
userId : string
firm : ExchangeFirmStruct
fullName : string
role : UserRole
executingGiveupFirms : ExchangeFirmStructSequence
profilesByClass : ProfileStructSequence
defaultProfile : ProfileStruct
accounts : AccountStructSequence
assignedClasses : ClassKeySequence
dpms : DpmStructSequence
<<sequencetypedef>>
ExchangeFirmStructSequence
0..n
<<struct>>
ExchangeFirmStruct
exchange : Excha...
firmNumber : string
<<typedef>>
UserRole
char
(from IDL Primitives)
<<struct>>
ExchangeAcronymStruct
exchange : Exchange
acronym : String
<<sequencetypedef>>
ProfileStructSequence
<<sequencetypedef>>
AccountStructSequence
<<sequencetypedef>>
ClassKeySequence
(from cmiProduct)
<<sequencetypedef>>
StringSequence
(from cmiUtil)
<<sequencetypedef>>
DpmStructSequence
0..n
<<struct>>
AccountStruct
account : string
executingGiveupFirm : ExchangeFirmStruct
<<typedef>>
ClassKey
(from cmiProduct)
0..n
string
(from IDL Primitives)
0..n
<<struct>>
DpmStruct
dpmUserId : string
dpmAssignedClasses : ClassKeySequence
0..n
<<struct>>
ProfileStruct
classKey : ClassKey
account : string
subAccount : string
executingOrGiveUpFirm : ExchangeFirmStruct




**Typedefs:**

| | |
|---|---|
| Exchange is a typedef | This field identifies the exchange for the current user. |
| LoginSessionMode *is a* char | Mode of login session - Simulator, Network-Testing, Production |

| | |
|---|---|
| SessionNameType is a typedev | Identifies the session type used by the user. |
| UserRole *is a* char | Each user is assigned to a role. The role defines what role the user plays in the market. The functions provided by the CAS can be restricted by role. At this time roles are defined for Market Makers, Brokers, Clearing Firms, and Help Desk. (See cmiConstants::UserRoles). |

---

**Typedefs for Sequences:**

AccountStructSequence *is a typedef of a sequence of* AccountStruct

DpmStructSequence *is a typedef of a sequence of* DpmStruct

ExchangeFirmStructSequence *is a typedef of a sequence of* ExchangeFirmStruct

PreferenceStructSequence *is a typedef of a sequence of* PreferenceStruct

ProfileStructSequence *is a typedef of a sequence of* ProfileStruct

SessionProfileUserStructSequence *is a typedef of a sequence of* SessionProfileUserStruct

---

**Structs:**

**AccountStruct**

Accounting information to be stored for a user

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| account | string | Account to be associated with the user |

| Attribute | Type | Documentation |
|---|---|---|
| executingGiveupFirm | ExchangeFirmStruct | Executing or give up firm to be used for this user |

### DpmStruct

Identifies a Designated Primary Market Maker and the classes in which they have been assigned.

| Attribute | Type | Documentation |
|---|---|---|
| dpmAssignedClasses | ClassKeySequence | List of classes that have been assigned to the Designated Primary Market Maker. |
| dpmUserId | string | User id of the Designated Primary Market Maker. |

### ExchangeAcronymStruct

Exchange acronym information.

| Attribute | Type | Documentation |
|---|---|---|
| acronym | String | Exchange acronym identifier. |
| exchange | Exchange | Exchange identifier. |

### ExchangeFirmStruct

Exchange firm information.

| Attribute | Type | Documentation |
|---|---|---|
| exchange | Exchange | Exchange identifier. |

| Attribute | Type | Documentation |
|---|---|---|
| firmNumber | string | Firm identifier. |

**PreferenceStruct**

System preferences are stored with a name that is qualified with a path that forms a hierrachy.

| Attribute | Type | Documentation |
|---|---|---|
| name | string | Name of a preference - that can include a hierarchy. |
| value | string | Value of the preference. |

**ProfileStruct**

Used to configure account information for classes that will be quoted by the user

| Attribute | Type | Documentation |
|---|---|---|
| account | string | Account to be used when reporting trades against quotes for this class |
| classKey | ClassKey | Class key for class for which the profile is applicable |
| executingOrGiveUpFirm | ExchangeFirmStruct | Executing or give up firm to be reported on trades for quotes on this class |
| subAccount | string | SubAccount to be used when reporting trades that occured based upon quotes for this class |

**SessionProfileStruct**

Identifies the user's session profile.

| Attribute | Type | Documentation |
| --- | --- | --- |
| account | string | The session account. |
| classKey | ClassKey | The class identifier for the session. |
| executingGiveupFirm | ExchangeFirmStruct | The executing give up firm for the session. |
| isAccountBlanked | Boolean | Account is left blank. |
| originCode | OriginType | |
| sessionName | SessionNameType | Profile session name. |
| subAccount | string | Subaccount for the session. |

**SessionProfileUserStruct**

| Attribute | Type | Documentation |
| --- | --- | --- |
| accounts | AccountStructSequence | List of accounts assigned to this user for the session. |
| assignedClasses | ClassKeySequence | List of assigned classes for this user for the session. |
| defaultProfile | SessionProfileStruct | Generic profile for all classes and all sessions. |
| defaultSessionProfiles | SessionProfileStructSequence | Lists session default profiles. |
| dpms | DpmStructSequence | List of DPMs assigned to this user for the session. |
| executingGiveupFirms | ExchangeFirmStructSequence | List of executing or give up firms for this user for the session. |
| firm | ExchangeFirmStruct | The user's firm for the session. |
| fullName | string | The user's full name. |
| role | UserRole | The user's role for the session. |

| Attribute | Type | Documentation |
|---|---|---|
| sessionProfilesByClass | SessionProfileStructSequence | List of session profiles by class, excluding all profiles with default class keys. |
| userAcronym | ExchangeAcronymStruct | The user acronym for the session. |
| userId | string | The user ID for the session--unique per user. |

**UserLogonStruct**

Struct used to logon to the CAS. This struct is passed to the cmi::UserAccess::logon() method.

| Attribute | Type | Documentation |
|---|---|---|
| loginMode | LoginSessionMode | The loginMode is of type LoginSessionMode. There are three modes which you can login to the CAS. The LoginSessionModes interface provides the different types of login modes as constants. The first is Stand-Alone testing - which refers to a client application program communicating with the CAS Simulator. Network Testing mode refers to talking to a CBOE test system over the CBOE network. Production refers to communicating to the CBOE production trading system. |
| password | string | Alphanumeric string with a maximum length of 12 characters. The password is initially specified by CBOE Trading Operations.<br><br>An interface is provided on the UserSessionManager object to change the password. |
| userId | string | User identifier assigned by the exchange to represent a user of the trading system. The userId is unique per user. |
| userName | string | Name of the user. |
| version | VersionLabel | The version is obtained by implementing the Version Interface in your application. |

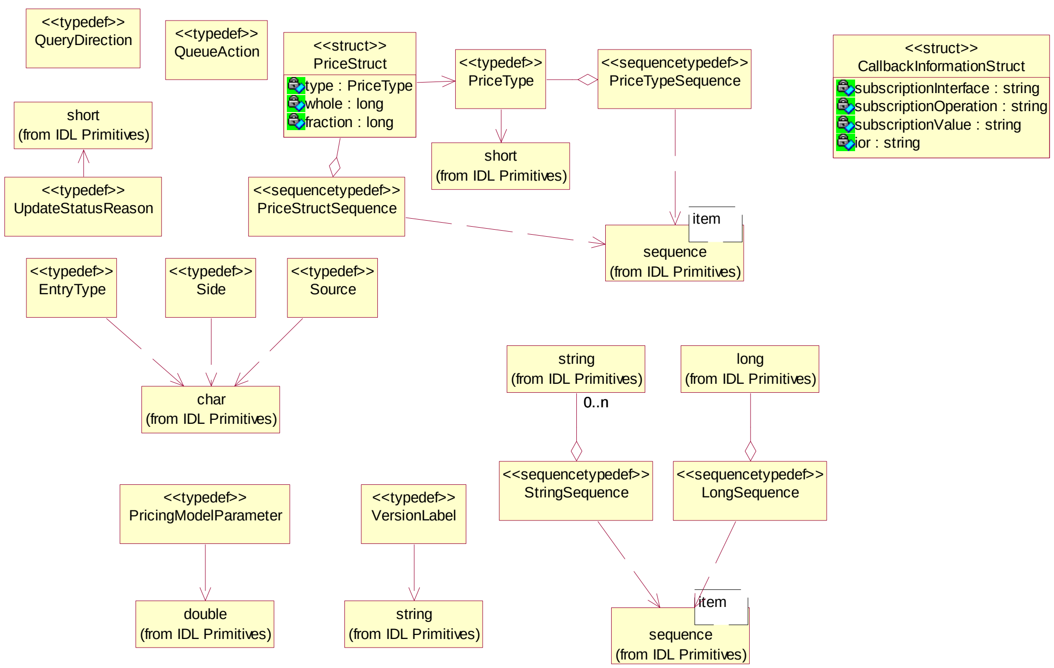
**UserStruct**

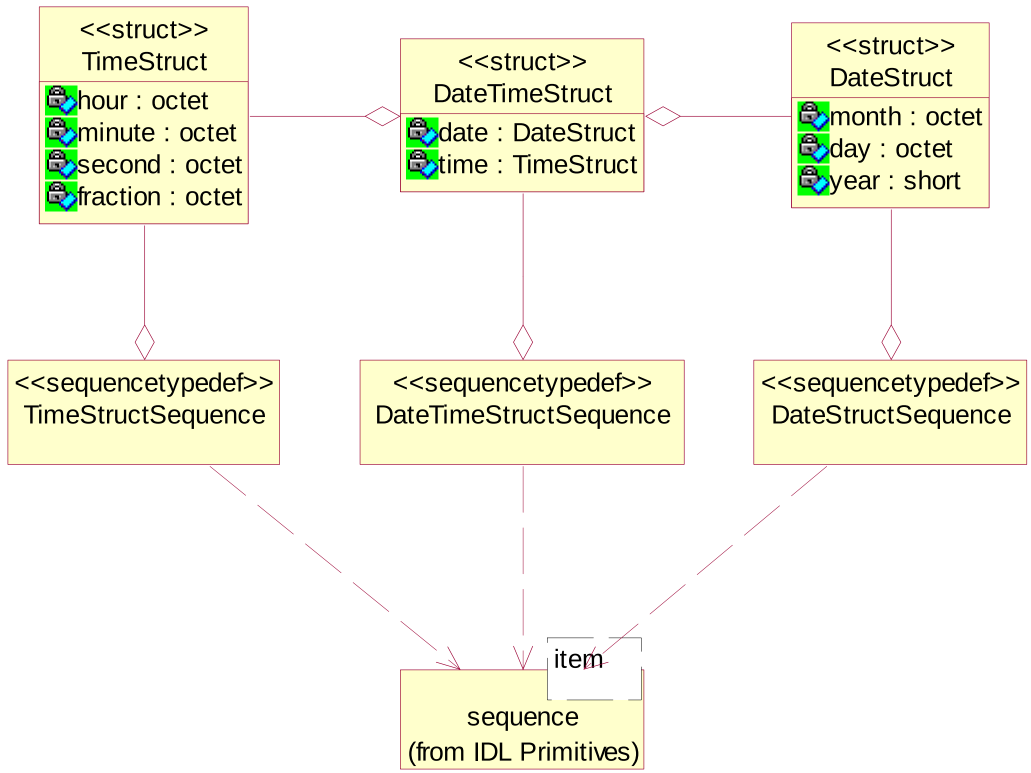
Represents a user within the trading system.

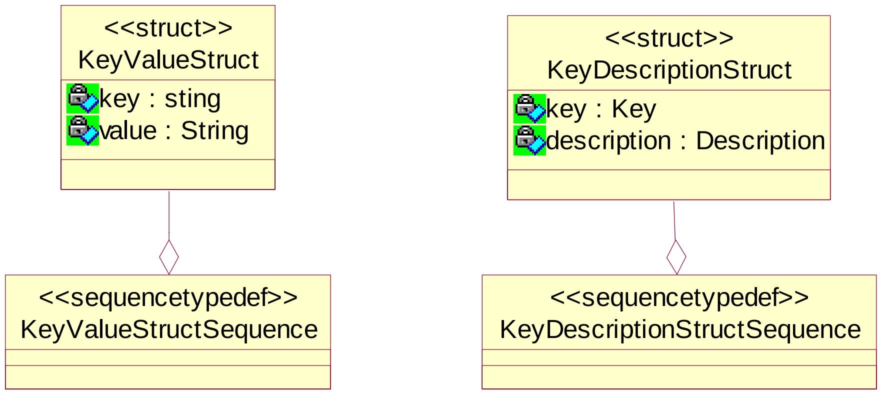
| Attribute | Type | Documentation |
| --- | --- | --- |
| accounts | AccountStructSequence | List of accounts in use by this user |
| assignedClasses | ClassKeySequence | Sequence of classes assigned to this user |
| defaultProfile | ProfileStruct | Default profile for any classes not explicitly listed in profilesByClass |
| dpms | DpmStructSequence | |
| executingGiveupFirms | ExchangeFirmStructSequence | List of executing or give up firms for this user |
| firm | ExchangeFirmStruct | The member firm or organization to which the user belongs. |
| fullName | string | Full name of the user. |
| profilesByClass | ProfileStructSequence | Accounting profiles for classes that are quoted by this user |
| role | UserRole | The role of the user (See cmiConstants::UserRoles). |
| userAcronym | ExchangeAcronymStruct | The acronym of a user within the system. |
| userId | string | User identifier assigned by the exchange to represent a user of the trading system. The userId is unique per user. |
| userName | string | The alphanumeric identifier assigned to the user. |

# *Module cmiUtil*

Common utility type used within the CBOE Market Interface




<<struct>>
TimeStruct
hour : octet
minute : octet
second : octet
fraction : octet
<<struct>>
DateTimeStruct
date : DateStruct
time : TimeStruct
<<struct>>
DateStruct
month : octet
day : octet
year : short
<<sequencetypedef>>
TimeStructSequence
<<sequencetypedef>>
DateTimeStructSequence
<<sequencetypedef>>
DateStructSequence
item
sequence
(from IDL Primitives)




**Typedefs:**

| | |
|---|---|
| ActivityReason is a typedef | Identifies the source of an activity record. Activity records can be generated as a result of system or user actions. |
| EntryType *is a* char | Type of activity history. |
| LinkageMechanism is a typedef | Specifies a Linkage function. See (cmiConstants::ExtensionFields) |
| OrderFlowDirection | Specifies whether the order flow is inbound, outbound or both. (see cmiConstants::OrderFlowDirectionType) |
| PriceType *is a* short | Type of price stored in PriceStruct (Market, Valued, No Price) (See cmiConstants::PriceTypes). |
| PricingModelParameter *is a* double | Parameter for pricing model |

| | |
|---|---|
| QueryDirection *is a* MarketQuery<br>QueryDirection *is a* short | Specifies the direction for retrieval of history information (see cmiConstants::QueryDirections). |
| QueueAction *is a* CMIRecapConsumer<br>QueueAction *is a* CMITickerConsumer<br>QueueAction *is a* MarketQuery<br>QueueAction *is a* CMICurrentMarketConsumer | Specifies actions on a queue. |
| ReportType is a typedef | Type of report (fill, cancel, etc) |
| Side *is a* char | Either Bid or Ask (see cmiConstants::Sides). |
| Source *is a* char | Specifies if the source is from the open outcry systems (TPF) or the electronic trading system (SBT) (See cmiConstants::Sources). |
| TradingClassStatusIndicator is a typedef short | Indicates the status of a class as it pertains to outages. |
| UpdateStatusReason *is a* short | Reason for update. Refer to cmiConstants::StatusUpdateReasons for a list of possible values. |
| VersionLabel *is a* string | Constant that specifies the CMI IDL Version. |

**Typedefs for Sequences:**

DateStructSequence *is a typedef of a sequence of* DateStruct

DateTimeStructSequence *is a typedef of a sequence of* DateTimeStruct

KeyDescriptionStructSequence *is a typedef of a sequence of* KeyDescriptionStruct

KeyValueStructSequence *is a typedef of a sequence of* KeyValueStruct

LongSequence *is a typedef of a sequence of* long

PriceStructSequence *is a typedef of a sequence of* PriceStruct

PriceTypeSequence *is a typedef of a sequence of* PriceType

StringSequence *is a typedef of a sequence of* string

TimeStructSequence *is a typedef of a sequence of* TimeStruct

---

**Structs:**

### CallbackInformationStruct

Describes a callback that was registered with the CAS.

This struct is used to report a callback context back to the client application. It is currently used to report a callback that was deregistered by the CAS due to an error condition - normally a non-response from a client side callback.

| **Attribute** | **Type** | **Documentation** |
|---|---|---|
| ior | string | The stringified Interoperable Object Reference for the callback object described in the CallbackInformationStruct. |
| subscriptionInterface | string | The name of the interface of the callback object that is described in the CallbackInformationStruct. |
| subscriptionOperation | string | Name of the operation on the callback object that is being described in the CallbackInformationStruct. This is usually the operation that did not provide a response to the CAS - prompting the callback object to be deregistered. |
| subscriptionValue | string | This is a string representation of the data for which the callback object was subcribed. |

### CboeIdStruct

Structure that stores a CBOE generated key. The key length is 64 bits, which is divided into to 32 bit longs.

| Attribute | Type | Documentation |
|---|---|---|
| highCboeId | long | Major part of the key (most significant bits). |
| lowCboeId | long | Minor part of key (least significant bits). |

### DateStruct

The DateStruct is used when only the day, month, and year are required. The month of this struct will be the normal values 1 through 12 that one would expect and not the 0-11 values used in the Java Programming Language.

| Attribute | Type | Documentation |
|---|---|---|
| day | octet | value between 1 and 31 |
| month | octet | value between 1 and 12 |
| year | short | value of year in YYYY format |

### DateTimeStruct

The DateTimeStruct is a combination of the DateStruct and the TimeStruct. The DateTimeStruct is used anywhere a full timestamp is required.

| Attribute | Type | Documentation |
|---|---|---|
| date | DateStruct | refer to DateStruct |
| time | TimeStruct | refer to TimeStruct |

**KeyDescriptionStruct**

Descripton of the type of key that was used.

| Attribute | Type | Documentation |
|---|---|---|
| description | Description | Describes the key |
| key | Key | Generic value |

**KeyValueStruct**

The KeyValueStruct contains the key and value information.

| Attribute | Type | Documentation |
|---|---|---|
| key | sting | |
| value | String | |

**OperationResultStruct**

This struct and corresponding sequence is used to define generic operation results when CBOE does not throw an exception. It contains and error code and error message.

| Attribute | Type | Documentation |
|---|---|---|
| errorCode | ErrorCode | Value is zero. |
| errorMessage | string | Value is an empty string. |

### PriceStruct

The PriceStruct is used to store a price. The implementation uses a fraction component that is divided by 1 billion (10E9). This format was selected to provide significant digits sufficient to meet current securities decimalization requirements as well as future multi-currency scenarios.

| Attribute | Type | Documentation |
|---|---|---|
| fraction | long | Decimal portion to be divided by the cmiConstants::PriceScale.DEFAULT_SCALE |
| type | PriceType | Type of price stored in PriceStruct (See cmiConstants::PriceTypes) |
| whole | long | Whole number portion of the price |

### TimeStruct

The TimeStruct is used when only the time in hours, minutes, seconds, and fraction of seconds are required. * The fraction portion of the TimeStruct will be in hundredths. A more generic name was used to allow for changing this to thousandths with a minimum of effort.

| Attribute | Type | Documentation |
|---|---|---|
| fraction | octet | Value between 0 and the denominator-1 |

| Attribute | Type | Documentation |
| --- | --- | --- |
| hour | octet | value between 0 and 24 |
| minute | octet | value between 0 and 60 |
| second | octet | value between 0 and 60 |

## *cmiConstants*

### ActivityFieldTypes

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ACCOUNT | ActivityFieldType | 2 | |
| ASK_OTY | ActivityFieldType | 4 | |
| ASK_PRICE | ActivityFieldType | 3 | |
| BID_OTY | ActivityFieldType | 6 | |
| BID_PRICE | ActivityFieldType | 5 | |
| BOOKED_QUANTITY | ActivityFieldType | 32 | |
| BUSTED_QUANTITY | ActivityFieldType | 7 | |
| CANCEL_REASON | ActivityFieldType | 31 | |
| CANCELLED_QUANTITY | ActivityFieldType | 8 | |
| CMTA | ActivityFieldType | 9 | |
| CONTINGENCY_TYPE | ActivityFieldType | 10 | |
| EVENT_STATUS | ActivityFieldType | 11 | |
| EXEC_BROKER | ActivityFieldType | 35 | |
| LEAVES_QUANTITY | ActivityFieldType | 12 | |
| MISMATCHED_QUANTITY | ActivityFieldType | 13 | |
| OPTIONAL_DATA | ActivityFieldType | 14 | |
| ORDER_STATE | ActivityFieldType | 33 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ORDERID | ActivityFieldType | 1 | |
| ORIGINAL_QUANTITY | ActivityFieldType | 15 | |
| PRICE | ActivityFieldType | 16 | |
| PRODUCT | ActivityFieldType | 34 | |
| PRODUCT_STATE | ActivityFieldType | 17 | |
| QUANTITY | ActivityFieldType | 18 | |
| QUOTE_UPDATE_CONTROL_ID | ActivityFieldType | 36 | Identification for the quote update. |
| QUOTEKEY | ActivityFieldType | 19 | |
| REINSTATED_QUANTITY | ActivityFieldType | 20 | |
| REPLACE_ORDERID | ActivityFieldType | 21 | |
| RFQ_TYPE | ActivityFieldType | 22 | |
| SIDE | ActivityFieldType | 23 | |
| TIME_IN_FORCE | ActivityFieldType | 24 | |
| TIME_TO_LIVE | ActivityFieldType | 25 | |
| TLC_QUANTITY | ActivityFieldType | 26 | |
| TRADED_QUANTITY | ActivityFieldType | 27 | |
| TRADEID | ActivityFieldType | 28 | |
| TRANSACTION_SEQUENCE_NUMBER | ActivityFieldType | 29 | |
| USER_ASSIGNED_ID | ActivityFieldType | 30 | |

**ActivityReasons**

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AWAY_EXCHANGE_CANCEL | ActivityReason | 199 | Used for Linkage when cancel reason is not provided (could be user cancel or cancel remaining). |
| BROKER_OPTION | ActivityReason | 100 | Broker option (admission of non-compliance). |
| CANCEL_ON_FALLBACK | ActivityReason | 800 | Force cancel orders due to system fallback. |
| CANCEL_ON_RSS | ActivityReason | 24 | Cancel report reason used when a series becomes restricted. Any resting orders that would open a position will be canceled by the system. |
| CANCEL_PENDING | ActivityReason | 101 | Used for Linkage. |
| COMM_DELAYS | ActivityReason | 126 | Communication delays to OPRA. |
| CROSS_IN_PROGRESS | ActivityReason | 15 | The order was the same as the resting Cross order. The resting Cross order did not route as a pair and is waiting to trade. |
| CROWD_TRADE | ActivityReason | 102 | Used for Linkage. |
| DUPLICATE_ORDER | ActivityReason | 103 | Duplicate order (e.g. dupe of ClOrdID). |
| EXCHANGE_CLOSED | ActivityReason | 104 | Exchange closed (an executing participant's order handling system receives an order when the market is closed). |
| FAILOVER | ActivityReason | 11 | This code may be used to indicate that a quote has been canceled due to CBOEdirect server failover. |
| GATE_VIOLATION | ActivityReason | 105 | Order received too soon (does not meet gate requirements) |
| INSUFFICIENT_CROSS_ORDER_DOLLAR_AMOUNT | ActivityReason | 21 | The Cross order price was less than the defined minimum dollar amount. |
| INSUFFICIENT_CROSS_ORDER_SIZE | ActivityReason | 20 | The Cross order size was less than the defined minimum order size. |
| INSUFFICIENT_CUSTOMER_ORD | ActivityReason | 19 | The Cross order quantity was less than the customer order |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ER_QUANTITY | | | quanitity at CBOE's best market. |
| INSUFFICIENT_QUANTITY | ActivityReason | 5 | Cancel reason that applies to quotes only. |
| INSUFFICIENT_QUANTITY_BUY_SIDE | ActivityReason | 8 | |
| INSUFFICIENT_QUANTITY_SELL_SIDE | ActivityReason | 9 | |
| INVALID_ACCOUNT | ActivityReason | 106 | Clearing account missing. |
| INVALID_AUTOEX_VALUE | ActivityReason | 107 | Invalid Auto-Ex value. |
| INVALID_CMTA | ActivityReason | 108 | Unknown clearing firm. |
| INVALID_EXCHANGE | ActivityReason | 138 | Currently used for TPF Linkage. |
| INVALID_FIRM | ActivityReason | 109 | Exec Broker missing. |
| INVALID_NBBO | ActivityReason | 16 | Signifies that the NBBO was invalid for the order. |
| INVALID_ORIGIN_TYPE | ActivityReason | 110 | OrderCapacity missing/invalid. |
| INVALID_POSITION_EFFECT | ActivityReason | 111 | OpenClose missing/invalid |
| INVALID_PRICE | ActivityReason | 112 | Reference price is out of bounds. |
| INVALID_PRODUCT | ActivityReason | 113 | Unknown, invalid, or ineligible (for Linkage) symbol (series, expiration, strike). |
| INVALID_PRODUCT_TYPE | ActivityReason | 114 | Complex order. |
| INVALID_QUANTITY | ActivityReason | 115 | Order exceeds limit. |
| INVALID_SESSION_ID | ActivityReason | 13 | The session is not valid. |
| INVALID_SIDE | ActivityReason | 116 | Invalid side. |
| INVALID_SUBACCOUNT | ActivityReason | 117 | Sub-account ID missing. |
| INVALID_TIME_IN_FORCE | ActivityReason | 118 | TimeInForce missing/invalid |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| INVALID_USER | ActivityReason | 119 | Account missing. |
| LATE_PRINT | ActivityReason | 120 | Late print to OPRA Tape. |
| LINKAGE_CONDITIONAL_FIELD_MISSING | ActivityReason | 900 | BusinessRejectReason - Conditionally required field missing. |
| LINKAGE_EXCHANGE_UNAVAILABLE | ActivityReason | 901 | BusinessRejectReason - Application not available (the session between the Hub and the distination participant is unavailable or the destination exchange's order handling application is down.) |
| LINKAGE_INVALID_DESTINATION | ActivityReason | 903 | BusinessRejectReason - Unknown ID (invalid DeliverToCompID) |
| LINKAGE_INVALID_MESSAGE | ActivityReason | 902 | BusinessRejectReason - Bad message. |
| LINKAGE_INVALID_PRODUCT | ActivityReason | 904 | BusinessRejectReason - Unknown Security |
| LINKAGE_SESSION_REJECT | ActivityReason | 905 | BusinessRejectReason - Other (used if a message relayed by the Hub received a session-level Reject from the destination participant.) |
| WASH_TRADE_PREVENTION | ActivityReason | 906 | Cancel reason if order is about to trade against another order or quote from the same user. |
| MISMATCHED_QUANTITY | ActivityReason | 907 | Reject reason for light order cancel/replace. |
| LOST_CONNECTION | ActivityReason | 4 | Cancel reason that applies to quotes only. |
| MISSING_EXEC_INFO | ActivityReason | 122 | Execution information missing. |
| NO_MATCHING_ORDER | ActivityReason | 123 | Used for Linkage. |
| NO_USER_ACTIVITY | ActivityReason | 23 | No quoting activity has occurred in a certain period of time. |
| NON_BLOCK_TRADE | ActivityReason | 124 | Used for Linkage. |
| NOT_ACCEPTED | ActivityReason | 140 | Currently used for TPF Linkage. |
| NOT_FIRM | ActivityReason | 121 | Instrument state is invalid for Linkage (the receiver is in Non- |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | Firm mode). |
| NOT_NBBO | ActivityReason | 125 | Not at NBBO |
| NOT_WITHIN_NBBO | ActivityReason | 17 | The Cross order price was not within the NBBO. |
| NOTHING_DONE | ActivityReason | 1 | |
| ORDER_TIMEOUT | ActivityReason | 141 | Used for Linkage. |
| ORDER_TOO_LATE | ActivityReason | 134 | Too late to enter. |
| ORIGINAL_ORDER_REJECTED | ActivityReason | 127 | Used for Linkage. |
| OTHER | ActivityReason | 128 | Used for Linkage. |
| PROCESSING_PROBLEMS | ActivityReason | 129 | Used for Linkage. |
| PRODUCT_HALTED | ActivityReason | 130 | Instrument state is invalid for Linkage (the receiver has the instrument halted) |
| PRODUCT_IN_ROTATION | ActivityReason | 131 | Instrument state is invalid for Linkage (the receiver is in Rotation) |
| PRODUCT_SUSPENDED | ActivityReason | 172 | This code may be used to indicate that a quote has been canceled due to product state transition to ProductStates.SUSPENDED |
| QRM_REMOVED | ActivityReason | 7 | Cancel reason that applies to quotes only. |
| QUOTE_IN_TRIGGER | ActivityReason | 12 | Quote is in a trigger status. |
| QUOTE_UPDATE_CONTROL | ActivityReason | 10 | Ability to update a quote. |
| QUOTE_UPDATE_REQUESTED | ActivityReason | 25 | Requests that the user re-enter their quotes. |
| REJECTED_LINKAGE_TRADE | ActivityType | 170 | The Linkage trade was rejected. |
| SAL_IN_PROGRESS | ActivityReason | 14 | This activity reason is used when SAL auction is in progress. |
| SATISFACTION_ORD_REJ_OTHER | ActivityReason | 171 | Satisfaction order was rejected.  Used for Linkage. |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| SELL_SHORT_RULE_VIOLATION | ActivityReason | 22 | The order violated the Sell Short rule. |
| SPECIAL_ADJUSTMENT | ActivityReason | 6 | Orders automatically canceled due to a special cash adjustment (product adjustment) |
| STALE_EXECUTION | ActivityReason | 132 | Used for Linkage. |
| STALE_ORDER | ActivityReason | 133 | Stale order (Time-to-live of received order has expired) |
| SYSTEM | ActivityReason | 3 | A non-user initiated cancel, e.g. Replacement order canceled due to the system's inability to cancel the original order. |
| TRADE_BUSTED | ActivityReason | 135 | Trade busted/corrected. |
| TRADE_REJECTED | ActivityReason | 136 | Trade rejected. |
| TRADE_THROUGH_CBOE | ActivityReason | 18 | The order price traded through CBOE's market. |
| TRANSACTION_FAILED | ActivityReason | 139 | Currently used for TPF Linkage. |
| UNKNOWN_ORDER | ActivityReason | 137 | Unknown order. |
| USER | ActivityReason | 2 | A user initiated cancel. |

**ActivityTypes**

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AWAY_EXCHANGE_MARKET | ActivityType | 315 | |
| BOOK_ORDER | ActivityType | 8 | |
| BUST_ORDER_FILL | ActivityType | 4 | |
| BUST_QUOTE_FILL | ActivityType | 107 | |
| BUST_QUOTE_LEG_FILL | ActivityType | 157 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BUST_REINSTATE_ORDER | ActivityType | 5 | |
| BUST_REINSTATE_STRATEGY_LEG | ActivityType | 55 | |
| BUST_STRATEGY_LEG_FILL | ActivityType | 54 | |
| CANCEL_ALL_ORDERS | ActivityType | 11 | |
| CANCEL_ALL_QUOTES | ActivityType | 104 | |
| CANCEL_ORDER | ActivityType | 3 | |
| CANCEL_ORDER_REQUEST | ActivityType | 303 | Activity type for Linkage. |
| CANCEL_ORDER_REQUEST_REJECT | ActivityType | 304 | Activity type for Linkage. |
| CANCEL_ORDER_REQUEST_REJECT_REJECTED | ActivityType | 308 | Activity type for Linkage. |
| CANCEL_QUOTE | ActivityType | 103 | |
| CANCEL_REPLACE_ORDER | ActivityType | 6 | |
| CANCEL_REPLACE_ORDER_REQUEST | ActivityType | 13 | |
| CANCEL_REPORT_REJECT | ActivityType | 305 | Activity type for Linkage. |
| CANCEL_REPORT_REJECT_REJECTED | ActivityType | 309 | Activity type for Linkage. |
| CANCEL_STRATEGY_LEG | ActivityType | 53 | |
| FILL_ORDER | ActivityType | 2 | |
| FILL_QUOTE | ActivityType | 102 | |
| FILL_REJECT | ActivityType | 302 | Activity type for Linkage. |
| FILL_REJECT_REJECTED | ActivityType | 307 | Activity type for Linkage. |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| FILL_STRATEGY_LEG | ActivityType | 52 | |
| HELD_FOR_IPP_PROTECTION | ActivityType | 12 | |
| LINKAGE_DISQUALIFIED_EXCHANGE | ActivityType | 316 | |
| MANUAL_ORDER_FR | ActivityType | 720 | Used for Linkage Fish and Return functionality |
| MANUAL_ORDER_FR_TIMEOUT | ActivityType | 721 | Used for Linkage Fish and Return functionality |
| MANUAL_ORDER_FR_TIMEOUT_FAILURE | ActivityType | 722 | Used for Linkage Fish and Return functionality |
| MANUAL_ORDER_SR | ActivityType | 718 | |
| MANUAL_ORDER_SR_TIMEOUT | ActivityType | 719 | |
| MANUAL_ORDER_SR_TIMEOUT_FAILURE | ActivityType | 816 | |
| NEW_ORDER | ActivityType | 1 | |
| NEW_ORDER_REJECT | ActivityType | 301 | Activity type for Linkage. |
| NEW_ORDER_REJECT_REJECTED | ActivityType | 306 | Activity type for Linkage. |
| NEW_ORDER_STRATEGY_LEG | ActivityType | 51 | |
| NEW_QUOTE | ActivityType | 101 | |
| NEW_RFQ | ActivityType | 201 | |
| PRICE_ADJUST_ORDER | ActivityType | 10 | |
| PRICE_ADJUST_ORDER_LEG | ActivityType | 60 | |
| QUOTE_LEG_FILL | ActivityType | 152 | |
| ROUTE_TO_AWAY_EXCHANGE | ActivityType | 310 | |
| STATE_CHANGE_ORDER | ActivityType | 9 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| SYSTEM_CANCEL_QUOTE | ActivityType | 105 | |
| UPDATE_ORDER | ActivityType | 7 | |
| UPDATE_QUOTE | ActivityType | 106 | |
| UPDATE_STRATEGY_LEG | ActivityType | 57 | |

**AlertResolutions**

Constants for alert resolutions.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ADJUSTED | AlertResolution | "A" | |
| AWAY_MARKET_REFUSE_TO_TRADE_OR_FADE | AlertResolution | "NU" | |
| AWAY_MARKET_UNAVAIL_TO_TRADE | AlertResolution | "IN" | |
| BOOK_TAKEN_OUT_AFTER_NOTIFICATION | AlertResolution | "BA" | |
| CBOE_SYSTEM_PROBLEMS | AlertResolution | "SP" | |
| CONTRA_UNAVAILABLE | AlertResolution | "CU" | |
| DELAYED_REPORT | AlertResolution | "DR" | |
| ERRONEOUS_REPORT | AlertResolution | "ER" | |
| EXECUTED_UNDER_FIRM_INSTRUCTIONS | AlertResolution | "FI" | |
| FAST_MARKET_AWAY | AlertResolution | "FM" | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| FIRM_DISCRETION | AlertResolution | "FD" | |
| FLASH_QUOTE | AlertResolution | "FQ" | |
| NBBO_FADE | AlertResolution | "NF" | |
| NBBO_LOCKED_W_CBOE | AlertResolution | "LB" | |
| NOT_ADJUSTED | AlertResolution | "UA" | |
| NOT_RESOLVED | AlertResolution | "NR" | |
| OTHER | AlertResolution | "O" | |
| PARTIAL_PRICE_ADJUSTMENT | AlertResolution | "AP" | |
| PARTIAL_QUANTITY_ADJ | AlertResolution | "AQ" | |
| POST_TRADE_QUOTE | AlertResolution | "PQ" | |
| SHUT_OFF_ERROR | AlertResolution | "SE" | |
| SINGLE_LISTED_OPTION | AlertResolution | "SL" | |
| TRADE_BUSTED | AlertResolution | "TB" | |
| TRADE_ENTERED_ON_REFRESHED_QUOTE | AlertResolution | "TO" | |

**AlertTypes**

Constants for the different types of alerts.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CBOE_TRADE_THROUGH | AlertType | 1 | |
| NBBO_TRADE_THOUGH | AlertType | 2 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NO_NBBO_AGENT | AlertType | 4 | |
| NON_EXECUTION | AlertType | 3 | |
| SATISFACTION_ALERT | AlertType | 5 | |

**AuctionStates**

The current state of the auction.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ABORTED | AuctionStates | 4 | The auction has been aborted. - NOT currently supported. |
| ENDED | AuctionStates | 2 | The auction has ended.  NOT currently supported. |
| PREMATURELY_ENDED | AuctionState | 3 | The auction has ended prematurely. - NOT currently supported. |
| STARTED | AuctionState | 1 | The auction has started. |

**AuctionTypes**

This interface provides the types of auctions that are supported.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AUCTION_HAL | AuctionType | 4 | Hybrid Automated Liaison auction type. |
| AUCTION_INTERNALIZATION | AuctionType | 1 | Used for auctioning internalization orders. |
| AUCTION_REGULAR_SINGLE | AuctionType | 3 | Used for auctioning normal orders (single-leg orders that are not internalized) - NOT currently supported. |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AUCTION_SAL | AuctionType | 5 | Simple Auction Liaison (SAL) is a mechanism that provides a price-improvement auction for simple (non-complex) orders. - NOT CURRENTLY SUPPORTED. |
| AUCTION_STRATEGY | AuctionType | 2 | Used for strategy auctions. |
| AUCTION_UNSPECIFIED | AuctionTypes | 0 | NOT currently supported. |
| STOCK_NBBO_FLASH | AuctionTypes | 6 | This attribute is used in the W_STOCK session to indicate (Flash) the Stock NBBO. This is not an auction but it is being used as an AuctionType in order to share the auction event channel. If the user tries to respond with an auction, a reject message will be received. |
| STOCK_ODD_LOT | AuctionTypes | 7 | This new message type is used to indicate the side and size of the odd lot order, as well as the NBBO price to all interested parties. |

**BillingTypeIndicators**

Billing type indicators.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CROSS | BillingTypeIndicators | C | Refers to a trade whereby both buyer and seller are represented on a single transaction. Thus, neither is really a maker or taker per se, but rather virtualy meet one another. |
| CROSS_PRICE_IMP | BillingTypeIndicators | S | Related to trades specifically between the primary and matched orders of an AIM order. |
| FLASH | BillingTypeIndicators | F | Refers to an order that is being presented to the dealers for a short-term auction for step-up, before the order is routed to an away exchange for a fill. |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| FLASH_PRICE_IMP | BillingTypeIndicators | T | Refers to trades between an auctioned order and the incoming response after the autcion started. |
| FLASH_RESPONSE | BillingTypeIndicators | E | Refers to the dealter responding to a flash and effectively stepping up to improve the CBSX market to the revailing price and fulfilling the customer here. |
| FLASH_RESPONSE_PRICE_IMP | BillingTypeIndicators | U | Relates to trades between an auctioned order and the incoming response after the auction started. |
| LINKED_AWAY | BillingTypeIndicator | X | Refers to an order that was sent to another market for execution. |
| LINKED_AWAY_RESPONSE | BillingTypeIndicator | L | Refers to the response from the other exchange filling the CBSX order sent to them. |
| MAKER | BillingTypeIndicators | A | Refers to the person adding liquidity to the market, by basically having an order or quote resting in the book to be traded against. |
| MAKER_TURNER | BillingTypeIndicators | V | Relates to trades between a non-auctioned order and the resting quote or order. |
| ODD_LOT_FLASH | BillingTypeIndicators | N | CBSX odd lot order or the odd lot portion of a mixed lot order that is being flashed. |
| ODD_LOT_RESPONSE | BillingTypeIndicators | B | Response for odd lot orders that are being flashed. |
| OPENING | BillingTypeIndicators | O | Refers to all executions that take place as part of the opening rotation process itself. |
| RESTING | BillingTypeIndicators | Q | Relates to autioned orders and resting quote or orders. |
| RESTING_TURNER | BillingTypeIndicators | W | Relates to the auctioned order and the resting quote or orders. |
| TAKER | BillingTypeIndicators | R | Refers to the person taking liquidity from the market, by basically sending an order to trade against the book. (i.e. they are coming in and taking out the best price) |

### BookDepthTypes

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| DELETE_PRICE | BookDepthUpdateType | 'D' | |
| INSERT_PRICE | BookDepthUpdateType | 'I' | |
| UPDATE_PRICE | BookDepthupdateType | 'U' | |

### ClassStates

Status information is disseminated at the Class level.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CLOSED | ClassState | 10 | |
| ENDING_HOLD | ClassState | 9 | |
| FAST_MARKET | ClassState | 6 | |
| HALTED | ClassState | 5 | |
| NO_SESSION | ClassState | 7 | |
| NOT_IMPLEMENTED | ClassState | 1 | |
| ON_HOLD | ClassState | 8 | |
| OPEN | ClassState | 4 | |
| OPENING_ROTATION | ClassState | 3 | |
| PRE_OPEN | ClassState | 2 | |
| SUSPENDED | ClassState | 11 | Used in a class state change callback event to indicate that a |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | failover situation has occurred affecting the class for which the callback is subscribed and that the class is suspended from trading. |

**ContingencyTypes**

Constants that represent types of contingencies to be applied to an order.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AON | ContingencyType | 2 | All or None |
| AUCTION_RESPONSE | ContingencyType | 14 | Auction response for the order. |
| AUTOLINK_CROSS | ContingencyType | 20 | Used for Stock:<br><br>Auto Link Cross is an order that will be Autolinked if CBOE is not the NBBO and the order is tradable at other markets. If there is remaining quantity, then this order will trade against an AutoLink_Cross_Match order (see below). Autolink_cross orders and autolink_cross_match orders can route in a single paired message or as two separate orders. Both orders must be for the same price but do not necessarily need to be for the same size. The autolink_cross order will not execute unless an autolink_cross_match order is received.<br><br>The matching orders (if not arriving as a pair) have to arrive within a specified time otherwise the original cross order will be cancelled |
| AUTOLINK_CROSS_MATCH | ContingencyType | 21 | Used for Stock:<br><br>Users have to submit two orders for AUTOLINK crosses. |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | One is AUTOLINK_CROSS and the second one is AUTOLINK_CROSS_MATCH. If away markets are better, then AUTOLINK_CROSS will first sweep away markets and the CBOE book. If there is still quantity remaining, then it will trade against the AUTOLINK_CROSS_MATCH.<br><br>The matching orders (if not arriving as a pair) have to arrive within a specified time otherwise the original cross order will be cancelled. |
| BID_PEG_CROSS | ContingencyType | 29 | Supported in the W_STOCK session. Order price field will be set as "Market."<br><br>Used to specify on inbound paired orders whether the paried orders should cross at the NBBO bid price or 'x' better than the NBBO bid where 'x' can be designated as a value in pennies including up to 4 decimal places to indicate sub-penny values. The contingency information on the paired orders should have:<br><br>(1) indication that the paired order set should cross at the NBBO bid (BID_PEG_CROSS)<br>(2) a numeric value stored in the contingency price field representing how many pennies (including up to 4 decimal places to indicate sub-penny values) better than the NBBO bid at which this paired order set should trade. If the specified increment should cause the price to go outside the NBBO (worse than the NBBO), the system should only increment to the other side of the NBBO. |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | |
| CASH_CROSS | ContingencyType | 32 | Cash settlement cross order. Supported in the W_STOCK session. Order price field will be set as "Limit." |
| CLOSE | ContingencyType | 12 | On close |
| CROSS | ContingencyType | 18 | Used for Stock:<br><br>These two orders can arrive as a pair (together) or one after the other. In either case they will need the same contingency. If they are for the same price and size they will trade against each other immediately as long as the price is at or within CBOE's quote and the NBBO. If the trade price is equal to the CBOE's best market and the market includes public customer orders, the cross order must be: X shares or more; be for a dollar amount greater than or equal to $Y; and larger than any public customer interest at that price. Both orders must be for the same price and size and neither order will execute unless both orders are received.<br><br>The matching orders (if not arriving as a pair) have to arrive within a specified time otherwise the original cross order will be cancelled. |
| CROSS_WITHIN | ContingencyType | 22 | Used for Stock: CURRENTLY UNSUPPORTED<br><br>Similar to cross orders except that the CROSS_WITHIN order shall trade at or better than the NBBO and within CBOE's market. |
| DO_NOT_ROUTE | ContingencyType | 26 | Orders marked DO_NOT_ROUTE will trade immediately or book as long as they do not violate the NBBO. If a DO_NOT_ROUTE order cannot be booked or traded without |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | violating the NBBO, it will be cancelled. This contingency type will be available for any origin type, including "I" orders |
| FOK | ContingencyType | 3 | Fill or Kill |
| INTERMARKET_SWEEP | ContingencyType | 15 | Intermarket Sweep Orders are treated as IOC orders but will trade against the book without regard to the NBBO. Trades executed as a result of these orders are exempted from RegNMS trade-through rules. ISO orders may only be sent if the sender has simultaneously sent orders to any other markets with protected quotes priced better than the limit price on the ISO. |
| INTERMARKET_SWEEP_BOOK | ContingencyType | 28 | Unlike Intermarket_Sweep orders, which are treated like IOC orders, orders marked Intermarket_Sweep_Book will be booked by the system if they cannot be traded. |
| IOC | ContingencyType | 4 | Immediate or Cancel |
| MIDPOINT_CROSS | ContingencyType | 17 | Used for Stock:<br><br>These two orders can arrive as a pair (together) or one after the other. In either case they will need the same contingency. A mid-point-cross trades at the middle of current NBBO and will trade at ½ cent increments. Both orders must be for the same size..<br><br>The matching orders (if not arriving as a pair) have to arrive within a specified time otherwise the original cross order will be cancelled. |
| MIN | ContingencyTypes | 6 | Minimum - refer to Quantity in contingency field |
| MIT | ContingencyType | 9 | Market if touched |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NBBO_FLASH_RESPONSE | ContingencyType | 27 | When a marketable order is received that cannot trade against the CBSX quote due to NBBO restrictions, the order will be flashed at the NBBO. When the auction message appears, users responding to the flash will respond with the new flash response contingency. |
| NBBO_FLASH_THEN_CANCEL | ContingencyType | 25 | Orders marked NBBO_FLASH_THEN_CANCEL will be flashed through the flash process if they cannot be executed immediatley at the NBBO. Any part of the order that is not filled will be cancelled after the flash. |
| NEXT_DAY_CROSS | ContingencyTypes | 33 | Next day settlement cross order. Supported in the W_STOCK session. Order price field will be set as "Limit." |
| NONE | ContingencyType | 1 | No contingency |
| NOTHELD | ContingencyType | 7 | Not held |
| OFFER_PEG_CROSS | ContingencyType | 30 | Supported in the W_STOCK session. Order price field will be set as "Market."<br><br>Used to specify on inbound paired orders whether the paried orders should cross at the NBBO offer price or 'x' better than the NBBO offer price where 'x' can be designated as a value in pennies including up to 4 decimal places to indicate sub-penny values. The contingency information on the paired orders should have:<br><br>(1) indication that the paired order set should cross at the NBBO offer (OFFER_PEG_CROSS).<br><br>(2) a numeric value stored in the contingency price field representing how many pennies (including up to 4 decimal places to indicate sub-penny values) better than the NBBO offer at which this paired order set should trade. If the |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | specified increment should cause the price to go outside the NBBO (worse than the NBBO), the system should only increment to the other side of the NBBO. |
| OPG | ContingencyType | 5 | Opening |
| RESERVE | ContingencyType | 16 | A reserve order has two quantities associated with it, order quantity and display quantity. Only the display quantity is visible as CBOE's book. The remaining quantity (order quantity - display quantity) is available to trade but not visible. |
| STOCK_ODD_LOT_NBBO_ONLY | ContingencyType | 24 | Odd lot order for Stock that should be cancelled if it cannot trade at the NBBO. |
| STP | ContingencyType | 10 | Stop order. CBSX does not support Stop orders. |
| STP_LIMIT | ContingencyType | 13 | Stop limit. CBXS does not support Stop limit orders. |
| STP_LOSS | ContingencyType | 11 | Stop loss. CBSX does not support Stop loss orders. |
| TIED_CROSS | ContingencyType | 19 | CURRENTLY UNSUPPORTED<br><br>Used for Stock:<br>Tied Cross: Similar to Cross orders except that they can trade at or better than CBOE's current market and NBBO trade through is allowed.<br><br>The matching orders (if not arriving as a pair) have to arrive within a specified time otherwise the original cross order will be cancelled. |
| TIED_CROSS_SWEEP | ContingencyType | 31 | Supported in the W_STOCK session. |
| TIED_CROSS_WITHIN | ContingencyType | 23 | Used for Stock: CURRENTLY UNSUPPORTED |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | Similar to TIED_CROSS orders except that the TIED_CROSS_WITHIN order shall trade at or better than the NBBO and within CBOE's market. |
| TWO_DAY_CROSS | ContingencyType | 34 | Two day settlement cross order. Supported in the W_STOCK session. Order price field will be set as "Limit." |
| WTP | ContingencyType | 35 | Wash Trade Prevention: When marked with this contingency, the order will be cancelled if it is about to trade against another order or quote from the same user. Additionally, the resting quote or order will be cancelled. |
| WD | ContingencyType | 8 | With discretion |

### CoverageTypes

Type of order - is the position covered or uncovered

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| COVERED | Coverage | 'C' | |
| UNCOVERED | Coverage | 'U' | |
| UNSPECIFIED | Coverage | 'B' | |

### CurrentMarketTypes

The types of current market contained in the V4 current market message.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BEST_MARKETS | CurrentMarketType | 1 | |
| BEST_PUBLIC_MARKETS | CurrentMarketType | 2 | |

**CurrentMarketViewtypes**

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CONTINGENT | CurrentMarketViewType | 1 | |
| NON_CONTINGENT | CurrentMarketViewType | 2 | |
| NON_Q | CurrentMarketViewType | 3 | |

**EntryTypes**

Constants that specify the type of order action to be taken

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ADD | EntryType | 'A' | |
| BOOK | EntryType | 'B' | |
| BUST | EntryType | 'K' | |
| CANCEL | EntryType | 'C' | |
| CANCEL_REPLACE | EntryType | 'R' | |
| FILL | EntryType | 'F' | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| PRICE_ADJUST | EntryType | 'P' | |
| STATE | EntryTypes | 'S' | |
| UPDATE | EntryType | 'U' | |

**ExchangeIndicatorTypes**

Product states of external exchanges.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CLEAR | ExchangeIndicatorType | 21 | External exchange's products are available for trading. |
| FAST_MARKET | ExchangeIndicatorType | 23 | External exchanges products are in a fast market. |
| HALTED | ExchangeIndicatorType | 22 | External exchange's products are halted. |
| OPENING_ROTATION | ExchangeIndicatorType | 24 | External exchanges products are in opening rotation |

**ExchangeMarketInfoType**

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BBO_ORDER_EXECUTED | ExchangeMarketInfoType | 4 | |
| BBO_ORDER_RECEIVED | ExchangeMarketInfoType | 3 | |
| NBBO_ORDER_EXECUTED | ExchangeMarketInfoType | 2 | |
| NBBO_ORDER_RECEIVED | ExchangeMarketInfoType | 1 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| WORST_BBO_IN_TIME_WINDOW | ExchangeMarketInfoType | 6 | |
| WORST_NBBO_IN_TIME_WINDOW | ExchangeMarketInfoType | 5 | |

**ExchangeStrings**

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AMEX | Exchange | "AMEX" | American Stock Exchange |
| BATS | Exchange | "BATS" | Better Alternative Trading System |
| BSE | Exchange | "BSE" | Boston Stock Exchange |
| CBOE | Exchange | "CBOE" | Chicago Board Options Exchange |
| CBOE2 | ExchangeStrings | "CBOE2" | Chicago Board Options Exchange 2 |
| CBOT | Exchange | "CBOT" | Chicago Board of Trade |
| CFE | Exchange | "CFE" | CBOE Futures Exchange |
| CHX | Exchange | "CHX" | Chicago Stock Exchange |
| CME | Exchange | "CME" | Chicago Mercantile Exchange. |
| CSE | Exchange | "CSE" | Cincinnati Stock Exchange. |
| ISE | Exchange | "ISE" | International Stock Market |
| LIFFE | Exchange | "LIFFE" | International Financial Futures and Options Exchange |
| NASD | Exchange | "NASD" | National Association of Security Dealers |
| NASDAQ | Exchange | "NASDAQ" | National Association of Securities Dealers Automated |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NQLX | Exchange | "NQLX" | Nasdaq Liffe Markets |
| NSX | Exchange | "NSX" | National Stock Exchange |
| NYME | Exchange | "NYME" | New York Mercantile Exchange |
| NYSE | Exchange | "NYSE" | New York Stock Exchange |
| ONE | Exchange | "ONE" | OneChicago Exchange |
| PHLX | Exchange | "PHLX" | Philadelphia Stock Exchange |
| PSE | Exchange | "PSE" | Pacific Stock Exchange |

**ExpectedOpeningPriceTypes**

Defines types of Expected Opening Price.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| DPM_QUOTE_INVALID | ExpectedOpeningPriceType | 9 | Invalid DPM quote is entered. |
| MORE_BUYERS | ExpectedOpeningPriceType | 2 | This type has been deprecated. |
| MORE_SELLERS | ExpectedOpeningPriceType | 3 | This type has been deprecated. |
| MULTIPLE_OPENING_PRICES | ExpectedOpeningPriceType | 5 | |
| NEED_DPM_QUOTE_TO_OPEN | ExpectedOpeningPriceType | 8 | |
| NEED_MORE_BUYERS | ExpectedOpeningPriceTypes | 3 | Indicates that the current market is imbalanced and will need more buyers' contracts (n) at the opening price. The message will provide users with the potential opening price and the imbalanced quantity (n). |
| NEED_MORE_SELLERS | ExpectedOpeningPriceTypes | 2 | Indicates that the current market is imbalanced and will need more sellers' contracts (m) at the opening price. The message will provide users with the potential opening price and the |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | imbalanced quantity (m). |
| NEED_QUOTE_TO_OPEN | ExpectedOpeningPriceType | 6 | |
| NO_OPENING_TRADE | ExpectedOpeningPriceType | 4 | Message will only be sent if the previous message was something other than value 4. |
| OPENING_PRICE | ExpectedOpeningPriceType | 1 | |
| PRICE_NOT_IN_QUOTE_RANGE | ExpectedOpeningPriceType | 7 | |
| PRICE_NOTE_IN_BOTR_RANGE | ExpectedOpeningPriceType | 10 | Indicates that the potential opening price is not in the valid Best of the Rest (BOTR) range based on the current BOTR price, current potential opening quantity and the system configuration. The message will provide users with the potential opening price and opening quantity. |

**ExpirationStyles**

Used to specify new option style

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AMERICAN | ExpirationStyle | 'A' | |
| EUROPEAN | ExpirationStyle | 'E' | |

**ExtensionFields**

| Name | Type | Initial Value | Documentation |
|---|---|---|---|

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ADJUSTED_PRICE_INDICATOR | ExtensionField | "5205" | Used for Linkage. |
| ASSOCIATED_ORDER_ID | ExtensionFields | "assocOrderId" | Customer orders associated with Linkage orders. |
| AUCTION_ID | ExtensionField | "auctionId" | Used for auction response. |
| AUTO_EXECUTION_SIZE | ExtensionField | "5201" | Used for Linkage. |
| AWAY_CANCEL_REPORT_EXEC_ID | ExtensionField | "awayCancelReportExecId" | |
| AWAY_EXCHANGE_EXEC_ID | ExtensionField | "17" | Used for Linkage. |
| AWAY_EXCHANGE_ORDER_ID | ExternalField | "37" | Used for Linkage. |
| AWAY_EXCHANGE_TRANSACT_TIME | ExtensionField | "60" | Used for Linkage. |
| AWAY_EXCHANGE_USER_ACRONYM | ExtensionField | "1" | Used for Linkage. |
| BARTID | ExtensionField | "BARTID" | Used for routing an order to a BART terminal. |
| BILLING_TYPE | ExtensionFields | "billingType" | Used for CBSX billing |
| CBOE_EXEC_ID | ExtensionField | "cboeExecId" | Used for Linkage. |
| CMTA_FIRM | ExtensionFields | "cmta" | Used for COB auction messages. |
| CORRESPONDENT_FIRM | ExtensionFields | "corresfirm" | Used for COB auction messages. |
| DIRECTED_FIRM | ExtensionFields | "dfirm" | Used for Directed AIM auctions. |
| DPM | ExtensionFields | "DDPM" | Used for Directed AIM auctions. |
| EXCHANGE_DESTINATION | ExtensionField | "100" | Used for Linkage. |
| EXEC_BROKER | ExtensionField | "execBroker" | |
| EXECUTING_FIRM | ExtensionFields | "firm" | Used for COB auction messages. |
| EXECUTION_RECEIPT_TIME | ExtensionField | "5207" | Used for Linkage. |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| EXPIRATION_TIME | ExtensionField | "ExpirationTime" | |
| FADE_EXCHANGE | ExtensionFields | "FADE_EXCHANGE" | |
| GUICFI | ExtensionFields | "guicfi" | Used for COB auction messages. |
| HANDLING_INSTRUCTION | ExtensionField | "21" | Used for Linkage. |
| LINKAGE_MECHANISM | ExtensionFields | "LinkageMechanism" | Used for Linkage. |
| MEET_FIRM_NAME | ExtensionField | "9381" | Firm information for stock leg of a buy-write |
| MEET_LOCATION_IN | ExtensionField | "9380" | Used for stock leg of a buy-write |
| MEET_LOCATION_OUT | ExtensionField | "207" | Used for stock leg of a buy-write |
| NBBO_ASK_PRICE | ExtensionFields | "nbboask" | Used for COB auction messages |
| NBBO_BID_PRICE | ExtensionFields | "nbbobid" | Used for COB auction messages. |
| OLA_REJECT_REASON | ExtensionField | "5209" | Used for Linkage. |
| ORDER_CAPACITY | ExtensionField | "6528" | Used for Linkage. |
| ORDER_RESTRICTIONS | ExtensionField | "6529" | Used for Linkage. |
| ORIGINAL_ORDER_TIME | ExtensionField | "5208" | Used for Linkage. |
| ORIGINAL_QUANTITY | ExtensionFiedl | "originalQuantity" | |
| ORS_ID | ExtensionField | "orsId" | |
| PDPM | ExtensionFields | "DPMM" | Used for Directed AIM auctions. |
| SATISFACTION_ALERT_ID | ExtensionFields | "satAlertId" | Associates an outbound S order with the alert. |
| SATISFACTION_ORDER_DISPOSITION | ExtensionField | "5206" | Used for Linkage. |
| SIDE | ExtensionField | "side" | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| STOCK_FIRM | ExtensionField | "STOCK_FIRM" | Firm information for stock leg of a buy-write. |
| STOCK_FIRM_NAME | ExtensionField | "STOCK_FIRM_ NAME" | Firm information for stock leg of a buy-write. |
| TEXT | ExtensionField | "58" | Used for Linkage. |
| TRADE_THRU_PRICE | ExtensionField | "5204" | Used for Linkage. |
| TRADE_THRU_SIZE | ExtensionField | "5203" | Used for Linkage. |
| TRADE_THRU_TIME | ExtensionField | "5202" | Used for Linkage. |
| USER_ASSIGNED_CANCEL_ID | ExtensionField | "11" | Used for Linkage. |

**FillRejectReasons**

Fill report reject reasons (See cmiConstants::FillRejectReasons)

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| INVALID_PRICE | FillRejectReason | ActivityReasons:: INVALID_PRIC E | |
| INVALID_PRODUCT | FillRejectReason | ActivityReasons:: INVALID_PROD UCT | |
| INVALID_QUANTITY | FillRejectReason | ActivityReasons:: INVALID_QUA NTITY | |
| INVALID_SIDE | FillRejectReason | ActivityReasons:: INVALID_SIDE | |
| NO_MATCHING_ORDER | FillRejectReason | ActivityReasons:: | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | NO_MATCHING_ORDER | |
| OTHER | FillRejectReason | ActivityReasons::OTHER | |
| STALE_EXECUTION | FillRejectReason | ActivityReasons::STALE_EXECUTION | |

**HandlingInstructions**

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AUTOMATIC | HandlingInstruction | 1 | |
| MANUAL | HandlingInstruction | 2 | |
| OTHER | HandlingInstruction | 3 | |

**LinkageMechanisms**

Constants for the device that is used to fill an NBBO customer order other than CBOEdirect.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NON_CBOE_LINKAGE_MECHANISM | LinkageMechanism | 1 | |

### ListingStates

Contains constants that represent listing state of a product

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ACTIVE | ListingState | 1 | This signifies that the product is listed for trading. |
| INACTIVE | ListingState | 2 | Initial value=2. This signifies that the product is NOT listed for trading. CBOE will maintain inactive productes for a configured retention period (usually 5 days), and then move them into the obsolete state. |
| OBSOLETE | ListingState | 4 | Initial value=4. This is a temporary server state as a product goes from active/inactive to unlisted. It should not be seen by CMi users normally. Obsolete indicates that the product is ready to be removed from the CBOEdirect system on the next cleanup. |
| UNLISTED | ListingState | 3 | Initial value=3. This is a listing state for a product that will not be available to download and is not eligible for trading. API users will not see a product with this listing state. When CBOE creates a new product but it has not yet been listed for trading, CBOE assignes it to the unlisted state. A product can reside in the CBOEdirect system indefinitely with the unlisted state. In this sense, unlisted is the same as inactive. |

### LoginSessionModes

The client application program can be configured at logon time (using the UserAccess::logon() operation) to operate in different session modes. There are session modes for stand alone testing using a CAS simulator, network testing version using a real CAS, and production - also using a real CAS.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NETWORK_TEST | LoginSessionMode | 2 | This login session mode indicates that the client is expecting to communicate with a real CAS that is running in a test mode. An exception will be thrown if the CAS is not in a test mode. |
| PRODUCTION | LoginSessionMode | 3 | This login session mode is used to connect to a production CAS. An exception will be thrown if the CAS is not in production mode. |
| STAND_ALONE_TEST | LoginSessionMode | 1 | Stand alone testing mode. In this mode the client application communicates with a CAS simulator. An exception will be thrown if an attempt is made to connect to real CAS. |

**LoginSessionTypes**

Indicates type of login requested - either primary or secondary.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| PRIMARY | LoginSessionType | 1 | Login being requested will be the primary login for the client. |
| SECONDARY | LoginSessionType | 2 | Login requested is a secondary - for fail over. |

**MarketChangeReasons**

Reason for a market data transmission.  Indicates if it was an exchange market change or a NBBO or both.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| COMBINED | MarketChangeReason | 3 | |
| EXCHANGE | MarketChangeReason | 1 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NBBO | MarketChangeReason | 2 | |

**MarketDataHistoryEntryTypes**

Defines type of Market Data History message

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| EXPECTED_OPEN_PRICE | MarketDataHistoryEntryType | 3 | |
| MARKET_CONDITION_ENTRY | MarketDataHistoryEntryType | 4 | |
| PRICE_REPORT_ENTRY | MarketDataHistoryEntryType | 2 | |
| QUOTE_ENTRY | MarketDataHistoryEntryType | 1 | |
| UNSIZED_QUOTE_ENTRY | MarketDataHistoryEntryType | 5 | |

**MarketIndicators**

Contains the constants used in the MarketIndicator field of the CurrentMarketStructV4.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ADDITIONAL_INFO_REL_SEC | MarketIndicator | 27 | |
| ADDITIONAL_INFORMATION | MarketIndicator | 23 | |
| ASK_IS_BOOK | MarketIndicator | 4 | |
| AUTO_EXECUTION | MarketIndicator | 2 | |
| BID_ASK_IS_BOOK | MarketIndicator | 5 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BID_IS_BOOK | MarketIndicator | 3 | |
| CLOSED_MARKET_MAKER | MarketIndicator | 22 | |
| CLOSING | MarketIndicator | 14 | |
| DEPTH_BID_OFFER | MarketIndicator | 18 | |
| DEPTH_ON_BID | MarketIndicator | 13 | |
| DEPTH_ON_OFFER | MarketIndicator | 12 | |
| DISQUALIFIED | MarketIndicator | 10 | |
| EQUIPMENT_CHANGOVER | MarketIndicator | 31 | |
| FAST_MARKET | MarketIndicator | 8 | |
| HALT_IN_VIEW_COMMON | MarketIndicator | 30 | |
| HALT_REL_SEC_NEWS_DISS | MarketIndicator | 20 | |
| HALT_REL_SEC_NEWS_PEND | MarketIndicator | 21 | |
| INACTIVE | MarketIndicator | 6 | |
| NEWS_DISSEMINATION | MarketIndicator | 15 | |
| NEWS_PENDING | MarketIndicator | 26 | |
| NO_OPEN_RESUME | MarketIndicator | 32 | |
| NON_FIRM_QUOTE | MarketIndicator | 24 | |
| OPENING | MarketIndicator | 25 | |
| ORDER_IMBALANCE | MarketIndicator | 19 | |
| ORDER_INFLUX | MarketIndicator | 16 | |
| PRE_OPENING_INDICATION | MarketIndicator | 17 | |
| REGULAR | MarketIndicator | 1 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ROTATION | MarketIndicator | 7 | |
| SLOW_ON_ASK_SIDE | Market Indicator | 34 | |
| SLOW_ON_BID_SIDE | MarketIndicator | 33 | |
| SLOW_ON_BOTH_SIDES | MarketIndicator | 35 | |
| TRADING_HALT | MarketIndicator | 9 | |
| UNKNOWN | MarketIndicator | 11 | |

**MatchTypes**

Used to set the price for the Firms side of the internalization order.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AUTO_MATCH | MatchType | 3 | The internalization order can be set to AUTO_MATCH by using the value 3. |
| GUARANTEE_STARTING_PRICE | MatchType | 1 | This match type is not currently supported. |
| LIMIT_PRICE | MatchType | 2 | Used to enter a limit price for the internalization order. |
| UNSPECIFIED | MatchType | 0 | This constant is used as the default match type for internalization. - NOT currently supported. |

**MessageKeyTypes**

Contains constants for the types of messages in the Stock Intermarket Trading System.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ITS_GENERAL_ADMIN | MessageKey | 1 | |
| ITS_PRE_OPEN_CANCEL | MessageKey | 5 | |
| ITS_PRE_OPEN_INDICATION | MessageKey | 2 | |
| ITS_PRE_OPEN_RESPONSE | MessageKey | 3 | |
| ITS_PRE_OPEN_SECOND_LOOK | MessageKey | 4 | |

**MultiplePartiesIndicators**

Constants to indicate multiple parties.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NO | MultiplePartiesIndicator | 2 | |
| UNKNOWN | MultiplePartiesIndicator | 3 | |
| YES | MultiplePartiesIndicator | 1 | |

**OptionalDataFields**

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NAMES_LATER | OptionalDataField | NLTR | Used for Stock: Traders can indicate a non real-time clearing ("Names Later") order using const OptionalDataField NAMES_LATER = "NLTR". If the order gets executed, the trade will not automatically clear. The trader will have to provide the contra-party information at the end of the day |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | before the trade clears. |

**OptionTypes**

Contains option types

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CALL | OptionType | 'C' | |
| PUT | OptionType | 'P' | |

**OrderBookPriceViewTypes**

Constant that represent the type of order book price view.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BY_ORIGIN_TYPE | OrderBookPriceViewTypes | 1 | Order book price by origin type. |

**OrderBookStatusValues**

Contains the values used in the order book status.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ORDER_BOOK_OPENING_ROTATION_LOCKED | OrderBookStatus | 1 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ORDER_BOOK_OPENING_ROTATION_UNLOCKED | OrderBookStatus | 2 | |

**OrderBookStructTradableTypes**

The struct defines the different tradable types that can be returned as part of the book depth detail query. The struct that is returned as part of this query is "BookDepthDetailedStruct". This in turn contains the "OrderBookStruct". The OrderBookStruct returns the "tradableType". The User can use the values to change the display of the tradable as may be required

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BOOK_ITEM_ORDER | OrderBookTradableType | 'O' | |
| BOOK_ITEM_QUOTE | OrderBookTradableType | 'Q' | Quote side |
| BOOK_ITEM_QUOTE_TRIGGER | OrderBookTradableType | 'T' | |

**OrderCancelTypes**

Used to specify what cancellation method should be applied by the system.  Added to permit compatibility with both OSC (CMS) style and FIX style cancellations.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CANCEL_ALL_QUANTITY | CancelType | 3 | This cancellation type cancels all the remaining open quantity of an order.  Available to support FIX style order cancel requests. |
| DESIRED_CANCEL_QUANTITY | CancelType | 1 | This type of cancellation uses the quantity in the cancel request as the quantity of the order to be cancelled. |
| DESIRED_REMAINING_QUANTI | CancelType | 2 | This type of cancellation uses the quantity as the desired |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| TY | | | remaining quantity of the order after cancellation. It is not the quantity to be cancelled. This type of cancellation is compatible with FIX order handling semantics. |

**OrderFlowDirectionType**

Constants for order flow direction.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BOTH | OrderFlowDirection | 3 | The order is both inbound and outbound. |
| INBOUND | OrderFlowDirection | 1 | The order is inbound. |
| OUTBOUND | OrderFlowDirection | 2 | The order is outbound. |

**OrderNBBOProtectionTypes**

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| FULL | NBBOProtectionType | 2 | |
| NONE | NBBOProtectionType | 1 | |

**OrderOrigins**

Constants that specify the origin of an order. CFE uses origin types V, E, F, O, M, G, H and C.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BROKER_DEALER | OriginType | 'B' | |
| BROKER_DEALER_FBW_ICM | OriginType | 'K' | For CBSX, K orders will not link away. Any remaining balance that should link away will be cancelled. |
| MANUAL_QUOTE_ORDER | OriginType | 'K' | PAR client users submit K orders as manual quote orders |
| BROKER_DEALER_FBW_NON_CUSTOMER | OriginType | 'W' | For orders for the account of non-broker/dealer customer accounts who (i) place more than 390 orders in listed options per day on average during a calendar month for their own beneficial account(s) pursuant to Exchange Rule 1.1(ggg) (i.e. “Professionals”) or (ii) voluntarily have their orders categorized as broker/dealer orders for order handling, order execution and cancel fee calculation purposes pursuant to Exchange Rule 1.1(fff) (i.e. “Voluntary Professionals”) |
| CTI1Origin1 | OriginType | 'V' | Member, customer segregated account. OCC account type: Customer |
| CTI1Origin2 | OriginType | 'E' | Member, house non-segregated account. OCC account type: Firm |
| CTI1Origin5 | OriginType | 'Q' | Member, SIPC protected account. |
| CTI3Origin1 | OriginType | 'G' | User proxy for trader, customer segregated account. OCC account type: Customer |
| CTI3Origin2 | OriginType | 'H' | User proxy for trader, house non-segrated account. OCC account type: Firm |
| CTI3Origin5 | OriginType | 'R' | User proxy for trader, SIPC protected account. |
| CTI2Origin2 | OriginType | 'O' | Non-member, house account. OCC account type: Customer |
| CTI4Origin5 | OriginType | 'T' | Non-member SIPC protected account. |
| CUSTOMER | OriginType | 'C' | CTI Equivalent (CTI4Origin1) - non-broker/dealer customer account orders. |
| CUSTOMER_BROKER_DEALER | OriginType | 'X' | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CUSTOMER_FBW | OriginType | 'D' | |
| FIRM | OriginType | 'F' | CTI Equivalent (CTI2Origin2) - OCC clearing member firm proprietary account orders. |
| FIRM_FBW_ICM | OriginType | 'J' | For CBSX, J orders will not link away. Any remaining balance that should link away will be cancelled. |
| FIRM_FBW_NON_CUSTOMER | OriginType | 'L' | |
| ITS_POR | OriginType | 'a' | Order on behalf of ITS pre-opening response from away exchange. |
| M_N_Y_FBW | OriginType | 'U' | Market-Maker using a floor broker workstation. |
| MARKET_MAKER | OriginType | 'M' | CTI Equivalent (CTI2Origin6) - Member market-maker orders. OCC account type: Market Maker |
| MARKET_MAKER_AWAY | OriginType | 'N' | Orders in the account of a market-maker or specialist on another options exchange |
| MARKET_MAKER_IN_CROWD | OriginType | 'I' | The "I" order origin type is an in crowd Market Maker order that is treated like a quote in the book. Unlike quotes, "I" orders are not cancelled at system logout. |
| N_Y_FBW | OriginType | 'Z' | Non-CBOE Member or Specialist Away. |
| PRINCIPAL | OriginType | 'P' | A market maker order origin type for Linkage. |
| PRINCIPAL_ACTING_AS_AGENT | OriginType | 'A' | A market maker order origin type for linkage. |
| SATISFACTION | OriginType | 'S' | A market maker order origin type for Linkage. |
| UNDERLY_SPECIALIST | OriginType | 'Y' | Orders in the account of a specialist registered in the underlying stock at the primary exchange for trading the stock |

**OrderStates**

Constants representing order states

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ACTIVE | OrderState | 6 | CBOE currently does not use this designation. |
| BOOKED | OrderState | 1 | The order was entered into the CBOE book. |
| CANCEL | OrderState | 2 | An order will have an order state of CANCEL only if the last action on the order is a cancellation. If an order is partially canceled and partially filled, but a portion of the order still remains in the book, the order state will be BOOKED. If the remaining BOOKED quantity is filled, then the order state will be FILL. If the remaining BOOKED quantity is canceled, then the order state will be CANCEL. If any portion of an order remains BOOKED at the close of trading, CBOE will automatically cancel the balance of that order, and change the order state to CANCEL. At the end of the day processing (around 4:00 p.m. Central Time), CBOE changes the state of the order from CANCEL to PURGED. PURGED orders will remain in the CBOE system until the end of the following business day. |
| EXPIRED | OrderState | 7 | CBOE currently does not use this designation. |
| FILL | OrderState | 3 | An order will have an order state of FILL only if the last action on the order is a fill report. If an order is partially canceled and partially filled, but a portion of the order still remains in the book, the order state will be BOOKED. If the remaining BOOKED quantity is filled, then the order state will be FILL. If the remaining BOOKED quantity on a partially filled order is canceled, then the order state will be CANCEL. |
| INACTIVE | OrderState | 5 | CBOE currently does not use this designation. |
| OPEN_OUTCRY | OrderState | 4 | CBOE currently does not use this designation. |
| PURGED | OrderState | 8 | During the trading day, or at the close of trading, an order moves into the CANCEL state. At the end of the day processing (around 4:00 p.m. Central time), CBOE changes |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | the state of the order from CANCEL to PURGED. PURGED orders will remain in the CBOE system until the end of the following business day. |
| REMOVED | OrderState | 9 | CBOE currently does not use this designation. |
| WAITING | OrderState | 10 | For purely electronic trading sessions such as ONE_MAIN and CFE_MAIN, the order state WAITING is the state that CBOE uses when a STP or STP Limit order is accepted by CBOE. When the order stop condition is met, the STP or STP Limit order will be triggered and it will either be immediately traded or it will be entered into the CBOE book. If it is entered into the book, CBOE will publish an order status update with the order state set to BOOKED. When the STP or STP Limit order is triggered, if it is partially filled, CBOE will publish a fill report, but the order will still have an order state of BOOKED. When the STP or STP Limit order is triggered, if it is entirely filled, CBOE will publish a fill report, and the order will have an order state of FILL. STP and STP Limit orders in Hybrid and non-Hybrid classes in the W_MAIN session will never receive an order status of WAITING. When a firm enters a STP or STP Limit order, CBOE will report the order with a status of BOOKED, regardless of whether the order is triggered or not. |

**OverrideIndicatorTypes**

Used by TPF for MDR display.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BOOK_OVERRIDE | OverrideIndicatorType | 'B' | Used by TPF. Manually enter last sale in ebook. |
| LINKAGE | OverrideIndicatorType | 'L' | Used by TPF. Last Sale displays as a Linkage order on MDR. |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NONE | OverrideIndicatorType | ' ' | Used by TPF. Normal last sale on the MDR display. |
| OFFER_OVERRIDE | OverrideIndicatorType | 'O' | Used by TPF. Manually override offer. |
| SUPERVISORY_OVERRIDE | OverrideIndicatorType | 'X' | Used by TPF. Manual override of price. |

**PositionEffects**

For omnibus accounts, orders can either result in closing or opening either the long or short side. An Open position indicates that the position is to be added.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CLOSED | PositionEffect | 'C' | |
| NOTAPPLICABLE | PositionEffect | 'N' | |
| OPEN | PositionEffect | 'O' | |

**PreOpeningIndicationTypes**

Constants used during pre-opening for the Stock Intermarket Trading System (ITS).

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CANCEL | PreOpeningIndicationTypes | 2 | Cancels pre-opening notification. |
| INDICATION | PreOpeningIndicationTypes | 1 | Notifies ITS participants of the opening. |
| SEC_LOOK | PreOpeningIndicationTypes | 3 | Second look sent after the opening for a better price. |

### PriceAdjustmentActions

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| PRICE_ADJUSTMENT_CREATE | PriceAdjustmentAction | 3 | |
| PRICE_ADJUSTMENT_DELETE | PriceAdjustmentAction | 2 | |
| PRICE_ADJUSTMENT_MOVE | PriceAdjustmentAction | 4 | |
| PRICE_ADJUSTMENT_UPDATE | PriceAdjustmentAction | 1 | |

### PriceAdjustmentTypes

Pending Price Adjustments

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| COMMON_DISTRIBUTION | PriceAdjustmentType | 8 | |
| DIVIDEND_CASH | PriceAdjustmentType | 2 | |
| DIVIDEND_PERCENT | PriceAdjustmentType | 3 | |
| DIVIDEND_STOCK | PriceAdjustmentType | 4 | |
| LEAP_ROLLOVER | PriceAdjustmentType | 5 | |
| MERGER | PriceAdjustmentType | 6 | |
| SPLIT | PriceAdjustmentType | 1 | |
| SYMBOL_CHANGE | PriceAdjustmentType | 7 | |

### PriceConstants

A special indicator that an IntegerPrice isn't defined.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NO_PRICE | IntegerPrice | -2147483648 | |

### PriceDisplayTypes

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| DECIMAL | PriceDisplayType | 2 | |
| FRACTION | PriceDisplayType | 1 | |

### PriceScale

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| DEFAULT_SCALE | long | 1000000000 | |

### PriceTypes

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CABINET | PriceType | 4 | Used for cabinet orders |
| LIMIT | PriceType | 2 | Used for limit orders and quotes. |
| MARKET | PriceType | 3 | Used for market orders.<br><br>For CBSX, market orders will not be allowed unless the product is in OPEN or FAST state. |
| NO_PRICE | PriceType | 1 | Only used to describe something that has no price such as a single sided quote or Market Data callback that has no price. |
| VALUED | PriceType | 2 | Used for limit orders and quotes. This is the exact same as LIMIT. |

**ProductClass**

Default Class Key

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| DEFAULT_CLASS_KEY | long | 0 | Default class key value |

**ProductStates**

Contains constants that represent state of a product

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CLOSED | ProductState | 1 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ENDING_HOLD | ProductState | 9 | Product hold is ending |
| FAST_MARKET | ProductState | 6 | |
| HALTED | ProductState | 5 | |
| NO_SESSION | ProductState | 7 | Product has not been assigned to a trading session |
| ON_HOLD | ProductState | 8 | Product has been placed on trading hold |
| OPEN | ProductState | 4 | |
| OPENING_ROTATION | ProductState | 3 | |
| PRE_OPEN | ProductState | 2 | |
| SUSPENDED | ProductState | 10 | Indicates that a failover situation has occurred affecting the product for which the callback is subscribed and that the product is suspended from trading. |
| UNKNOWN | ProductState | 0 | |

**ProductTypes**

Contains constants that represent product types. Not all of these products are currently supported by CBOE systems

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| COMMODITY | ProductType | 1 | |
| DEBT | ProductType | 2 | |
| EQUITY | ProductType | 3 | |
| FUTURE | ProductType | 4 | |
| INDEX | ProductType | 5 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| INTEREST_RATE_COMPOSITE | ProductType | 12 | |
| LINKED_NOTE | ProductType | 6 | |
| OPTION | ProductType | 7 | |
| STRATEGY | ProductType | 11 | |
| UNIT_INVESTMENT_TRUST | ProductType | 8 | |
| VOLATILITY_INDEX | ProductType | 9 | |
| WARRANT | ProductType | 10 | |

### QueryDirections

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| QUERY_BACKWARD | QueryDirection | 2 | |
| QUERY_FORWARD | QueryDirection | 1 | |

### QueueActions

Actions taken when there is queueing.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| DISCONNECT_CONSUMER | QueueAction | '3' | Automatically unregisters the callback object when the queue size is exceeded.  Currently, the queue size value for the Hybrid CAS is configured to 5,000. |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| FLUSH_QUEUE | QueueAction | '1' | Allows the end user to continue building large queues on the CAS. Once the queue reaches a certain size, it will automatically be flushed then allowed to rebuild. The queue size is currently configured to 1,000. |
| NO_ACTION | QueueAction | '0' | CBOE currently does not support this designation. It will return the same resulting actions as DISCONNECT_CONSUMER. |
| OVERLAY_LAST | QueueAction | '2' | The callback object receives the most recent market data information for a product, overlaying updates that are in progress. |

**QuoteUpdateControlValues**

Constants for quote update values.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CONTROL_DISABLED | QuoteUpdateControl | 0 | Remove the ability to update a quote. |

**ReportTypes**

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CANCEL_ORDER_REQUEST | ReportType | 103 | This report type is not used in the CancelReportStruct. |
| CANCEL_ORDER_REQUEST_REJECT | ReportType | 104 | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CANCEL_ORDER_REQUEST_REJECT_REJECTED | ReportType | 108 | |
| CANCEL_REPORT_REJECT | ReportType | 105 | |
| CANCEL_REPORT_REJECT_REJECTED | ReportType | 109 | |
| FILL_REJECT | ReportType | 102 | |
| FILL_REJECT_REJECTED | ReportType | 107 | |
| NEW_ORDER_REJECT | ReportType | 101 | |
| NEW_ORDER_REJECT_REJECTED | ReportType | 106 | |
| REGULAR_REPORT | ReportType | 1 | |
| STRATEGY_LEG_REPORT | ReportType | 3 | |
| STRATEGY_REPORT | ReportType | 2 | |

### RFQTypes

Constants that specify whether an RFQ was generated manually or via the autoquotation system

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| MANUAL | RFQType | 1 | |
| SYSTEM | RFQType | 2 | |

### SatisfactionOrderDispositions

Reasons for canceling incoming Satisfaction orders.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ORDER_SIZE_MORE_THAN_TRADE_THROUGH_SIZE | SatisfactionOrderDisposition | 2 | Cancel - order size is more than the trade through size. |
| PRO_RATA_DISTRIBUTION | SatisfactionOrderDisposition | 1 | Cancel due to pro-rata distribution. |
| SATISFIED_AS_SPECIFIED | SatisfactionOrderDisposition | 0 | Order was satisfied. |

**SatisfactionOrderRejectReasons**

Satisfaction order reject reasons (See cmiConstants::SatisfactionOrderRejectReasons)

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| COMM_DELAYS | SatisfactionOrderRejectReason | ActivityReasons::COMM_DELAYS | |
| CROWD_TRADE | SatisfactionOrderRejectReason | ActivityReasons::CROWD_TRADE | |
| INVALID_PRODUCT_TYPE | SatisfactionOrderRejectReason | ActivityReasons::INVALID_PRODUCT_TYPE | |
| LATE_PRINT | SatisfactionOrderRejectReason | ActivityReasons::LATE_PRINT | |
| NON_BLOCK_TRADE | SatisfactionOrderRejectReasons | ActivityReasons::NON_BLOCK_TRADE | |
| ORIGINAL_ORDER_REJECTED | SatisfactionOrderRejectReasons | ActivityReasons:: | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | ORIGINAL_ORDER_REJECTED | |
| PROCESSING_PROBLEMS | SatisfactionOrderRejectReason | ActivityReasons::PROCESSING_PROBLEMS | |
| TRADE_BUSTED | SatisfactionOrderRejectReasons | ActivityRejectReasons::TRADE_BUSTED | |
| TRADE_REJECTED | SatisfactionOrderRejectReasons | ActivityReasons::TRADE_REJECTED | |

### SessionNameValues

Contains constants that represent session name values.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ALL_SESSION_NAME | TradingSessionName | "ALL_SESSIONS" | |

### Sides

Side of the order - buy or sell, bid or ask

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AS_DEFINED | side | 'D' | |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ASK | Side | 'A' | |
| BID | Side | 'B' | |
| BUY | Side | 'B' | |
| BUY_MINUS | Side | 'M' | |
| OPPOSITE | Side | '0' | |
| SELL | Side | 'S' | |
| SELL_PLUS | Side | 'P' | |
| SELL_SHORT | Side | 'H' | Used for Stock. |
| SELL_SHORT_EXEMPT | Side | 'X' | Used for Stock. |
| UNSPECIFIED | Side | 'U' | Used when the side of the order is not specified. |

**Sources**

Source of order handling. System order was processed by - The screen based system, the TPF Trading Engine, COMPASS, Linkage or ITS.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| COMPASS | Source | 'C' | Order processed by the COMPASS system. |
| ITS | Source | 'I' | Order was processed by the Intermarket Trading System. |
| LINKAGE | Sources | 'L' | Order processed by the Linkage system. |
| SBT | Source | 'S' | Order processed by the SBT system. |
| TPF | Source | 'T' | Order processed by the TPF system. |

**StatusUpdateReasons**

Contains constants that represent the reason a status message was published.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BOOKED | UpdateStatusReason | 1 | CBOE entered the order or quote into the CBOE book. |
| BUST | UpdateStatusReason | 8 | CBOE partially or completely busted (or bust-reinstated) an order or quote trade. |
| CANCEL | UpdateStatusReason | 2 | The user or CBOE has partially or completely cancelled the order or quote. |
| FILL | UpdateStatusReason | 3 | The order or quote is partially or completely filled. |
| NEW | UpdateStatusReason | 7 | CBOE created a new order or quote. |
| OPEN_OUTCRY | UpdateStatusReason | 6 | CBOE currently does note use this designation. |
| POSSIBLE_RESEND | UpdateStatusReason | 10 | CBOE may or may not be resending the order or quote status message. CBOE requires firms to set the GMD field to "TRUE" in their order and quote status subscriptions to avoid unnecessary resending of unacknowledged messages with the POSSIBLE_RESEND designation. |
| QUERY | UpdateStatusReason | 4 | When a firm initially subscribes for orders or quotes, CBOE sends the firm a copy of each order or quote. Since the firm is not receiving the order or quote because of a market event (fill, cancel, bust, etc.), CBOE places the state QUERY on each order or quote. |
| QUOTE_TRIGGER_BUY | UpdateStatusReason | 11 | Quote trigger on the buy side of the quote. |
| QUOTE_TRIGGER_SELL | UpdateStatusReason | 12 | Quote trigger on the sell side of the quote. |
| REINSTATE | UpdateStatusReason | 9 | When an order bust-reinstate takes place, CBOE busts all (or a portion) of an order trade, and then reinstates (re-enters) the user's order (or a portion of the user's order) into the book. |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| | | | During an order bust-reinstate, the order trade bust and the order reinstate will always take place at the exact same time. The reinstate quantity will always equal the bust quantity. |
| UPDATE | TradableStatus | 5 | This is the resultant status when the firm successfully calls the acceptOrderUpdateRequest message. |

**StrategyTypes**

Constants for strategy types in CBOEdirect.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BUY_WRITE | StrategyType | 9 | |
| COMBO | StrategyType | 8 | |
| DIAGONAL | StrategyType | 7 | |
| PSEUDO_STRADDLE | StrategyType | 3 | |
| RATIO | StrategyType | 5 | |
| STRADDLE | StrategyType | 2 | |
| TIME | StrategyType | 6 | |
| UNKNOWN | StrategyType | 1 | |
| VERTICAL | StrategyType | 4 | |

**TickDirectionTypes**

Contains tick direction symbols.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| MINUS_TICK | TickDirectionTypes | '-' | MINUS_TICK '-'<br><br>Means the current last sale price is lower than the previous last sale price. Also known as a "downtick". |
| PLUS_TICK | TickDirectionType | '+' | PLUS_TICK '+'<br><br>Means the current last sale price is higher than the previous last sale price. Also known as an "uptick". |
| UNKNOWN_TICK | TickDirectionType | ' ' | UNKNOWN_TICK ' 'Means the tick direction field is not used or not known.<br><br>For options, the tick direction field is always set to UNKNOWN_TICK. |
| ZERO_MINUS_TICK | TickDirectionTypes | '_' | ZERO_MINUS_TICK '_'<br><br>Means the current last sale price is the same as the previous last sale price, but is lower than the most recent different last sale price. |
| ZERO_PLUS_TICK | TickDirectionTypes | '*' | ZERO_PLUS_TICK '*'<br><br>Means the current last sale price is the same as the previous last sale price, but is higher than the most recent different last sale price. |

**TimesInForce**

Times order is in force - orders can be specifed as being good until cancel (GTC) or until the end of the day (DAY).

Can be used to specify the time an order is in force in an order entry struct.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| ALL | TimesInForce | 'A' | Stands for all orders (i.e. GTC+DTD+DAY) |
| DAY | TimeInForce | 'D' | Day order |
| GTC | TimeInForce | 'G' | Good Til Cancel orders. CBSX does not support GTC orders. |
| GTD | TimeInForce | 'T' | Good until datetime |

**TradeReportHandlingInstructions**

Trade report instructions used by One Chicago for Block Trade functionality

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NO_TICKER | short | 2 | Indicates not to publish to Ticker. No last sales data will go to COPP and CFN. Used by One Chicago for block trade functionality. |
| NO_TRADEREPORT | short | 1 | Indicates not to publish the trade to CTM. No trade reports will go to CTM-R and OCC. Used by OneChicago for block trade functionality. |
| NO_TRADEREPORT_NO_TICKER | short | 3 | Indicates not to publish the Trade to CTM. No trade reports will go to CTM-R and OCC and last sale data will not go to COPP and CFN. Used by One Chicago for block trade functionality. |
| REGULAR | short | 0 | Default value. Used by One Chicago for Block Trade functionality. Indicates the current default functionality. |

**TradingSessionMethod**

Method of trading to be used in a trading session.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| OPENOUTCRY | TradingSessionState | 2 | |
| SBT | TradingSessionState | 1 | |

### TradingSessionStates

Contains constants that represents the state of a trading session.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| CLOSED | TradingSessionState | 1 | |
| OPEN | TradingSessionState | 2 | |

### TradingSessionType

Contains constants that represent state of a session

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| DAY | TradingSessionState | 1 | |
| EVENING | TradingSessionState | 2 | |

### UserRoles

User Role in system - Firm, Broker Dealer, Market Maker (See cmiConstants::UserRoles)

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| BROKER_DEALER | UserRole | 'B' | |
| CLASS_DISPLAY | UserRole | 'C' | |
| CUSTOMER_BROKER_DEALER | UserRole | 'X' | |
| DPM_ROLE | UserRole | 'D' | |
| EXCHANGE_BROKER | UserRole | 'E' | |
| FIRM | UserRole | 'F' | |
| FIRM_DISPLAY | UserRole | 'R' | |
| HELP_DESK | UserRole | 'H' | |
| MARKET_MAKER | UserRole | 'M' | |
| PRODUCT_MAINTENANCE | UserRole | 'P' | |
| TFL_ROLE | UserRole | 'T' | User role for a Trading Floor Liaison. |
| UNKNOWN_ROLE | UserRole | 'K' | |

### VolumeTypes

Constants for the differenct contingency types.

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| AON | VolumeType | 2 | All or none |
| CUSTOMER_ORDER | VolumeType | 6 | Customer |
| FOK | VolumeType | 3 | Fill or Kill |
| IOC | VolumeType | 4 | Immediate or Cancel |
| LIMIT | VolumeType | 1 | Limit (no contingency) |

| Name | Type | Initial Value | Documentation |
|---|---|---|---|
| NO_CONTINGENCY | VolumeType | 5 | |
| ODD_LOT | VolumeType | 9 | All non-round lot amounts |
| PROFESSIONAL_ORDER | VolumeType | 7 | Broker Dealer |
| QUOTES | VolumeType | 8 | In-crowd Market-Maker. |