iLink 3 - Simple Binary Encoding

iLink 3 uses Simple Binary Encoding (SBE) optimized for low latency of encoding and decoding while keeping bandwidth utilization reasonably small. All FIX semantics are supported. This encoding standard describes the wire protocol for iLink 3 messages and is complimentary to other FIX standards for session protocol and application-level behavior.

This topic describes implementation of SBE in submitting orders on CME Globex and receiving order entry responses.

Contents

Binary Type System

To support traditional FIX semantics, all documented field types are supported. The binary type system binds to native binary data types, and defines derived types as needed.

The binary type system has been enhanced to:

- provide a means to specify precision of decimal numbers and timestamps, as well as valid ranges of numbers
- differentiate fixed-length character arrays from variable-length strings
- provide a consistent system of enumerations, Boolean switches, and multiple-choice fields.

Binary Encoding

Binary encoding provides direct data access without complex transformations or conditional logic by:

- Usage of native binary data types and simple types derived from native binaries, such as prices and timestamps.
- Fixed positions and fixed length fields, supporting direct access to data.

iLink 3 message schema supports the Simple Binary Encoding Specification - SBE version 1.0 release candidate 4

Users must be registered with FIX Protocol Organization to access the Simple Binary Encoding Documentation and will need a User ID provided by FPL to logon to the site.

The default data ranges and null indicator are listed below for each integer encoding.

Type	Min	Max	Null
uInt8	0	254	255
uInt16	0	65534	65535
uInt32	0	4294967294	4294967295
uInt64	0	9223372036854775807	18446744073709551615

Encoding for null values

For **charNULL** primitive data type, null value is represented by:

Encoding	Null Value
Enumeration	byte 0

String	byte 48 (character '0')

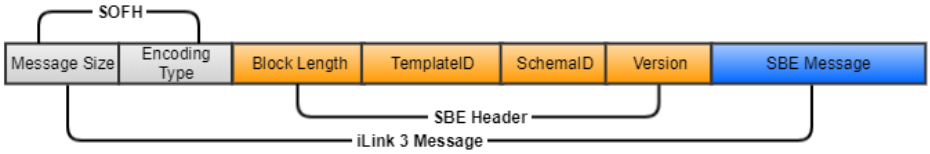
For example:

```
 <enum name="CmtaGiveUpCD" encodingType="charNULL">
        <validValue name="GiveUp" description="Give Up">G</validValue>
        <validValue name="SGXoffset" description="SGX offset">S</validValue>
 </enum>
```

The null value of enumerations of data type char is represented by byte '0', not a character '0'.

Message Structure

This diagram shows the structure of an SBE encoded iLink 3 message.



Simple Open Framing Header (SOFH)

FIX Protocol Ltd. offers the Simple Open Framing Header standard for framing messages encoded with binary wire formats, such as Simple Binary Encoding.
The framing header provides two features:

- An overall message length including headers to support framing.
- An identifier of the encoding used in the message payload. This supports selecting the correct decoder in the case where multiple message encodings are used on a session. It also aids tools such as protocol analyzers to identify message protocols contained in network packets.

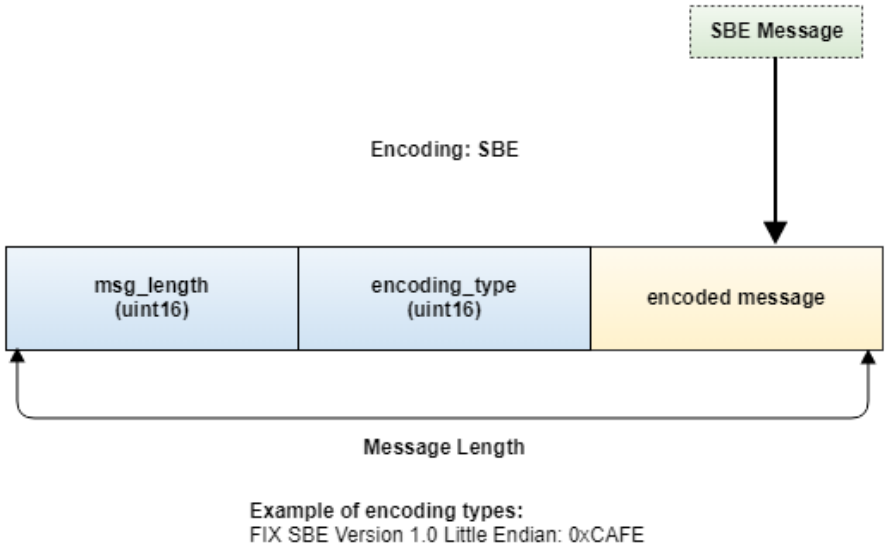
> The framing standard specifies that the framing header will always be encoded in little-endian byte order.

Simple Open Framing Header (SOFH) **(4 bytes)**	**SBE Header** **(8 bytes)**	**SBE Message** **(Variable Length)**
Message Length: 2 bytes • Overall message length including headers to support framing	**Block Length: 2 bytes** • The total space reserved for the root level of the Message	**Customer to CME Globex Messages** Example: • New Order Single • Order Cancel Replace Request • Order Cancel Request • Mass Quote
Encoding Type: 2 bytes • CME SBE Version 1.0 Little-endian: 0xCAFE • Identifier of the encoding used in the message payload	**TemplateID: 2 bytes** • Message Template Identifier	**CME Globex to Customer Messages** Example: • Execution Reports • Mass Quote Acknowledgement • Business Rejects

	SchemaID: 2 bytes • Identifier of the Message Schema that contains the Template	
	Version: 2 bytes • Version of the Message Schema in which the Message is defined	

The Simple Open Framing Header is four octets in length consisting of two fields, the Message_Length and Encoding_Type. The purpose of the Simple Open Framing Header will be to provide a simple mechanism to process messages from a stream and will always be encoded in little-endian byte order.

The Simple Open Framing Header is defined to contain the following information:



Message_Length Field

The Message_Length shall be defined to be the length in octets (i.e. bytes) of a message inclusive of the length of the Simple Open Framing Header.

The Message_Length field shall be the first field in the Simple Open Framing Header.

The Message_Length field shall be two octets in length (binary type uInt16), permitting a maximum message length of 2^16.

Encoding_Type Field

The Encoding_Type field shall be defined to be an integral enumeration.

The Encoding_Type field shall be the second field in the Simple Open Framing Header.

The values of encodingType used to indicate SBE payload is currently defined as:

Encoding	encodingType value
CME SBE version 1.0 little-endian	0xCAFE

Please note that the encoding type is a reserved keyword represented as a hexadecimal digits in Little-Endian byte order.

This framing header differs from the official SOFH specification in these ways:

- Byte order is little-endian and not big-endian.
- Message length is 2 bytes and not 4 bytes.
- Encoding type is private user defined value outside designated range.

SBE Message Header

The purpose of the SBE header encoding header is to tell which message template was used to encode the message and to give information about the length of the message body to aid in decoding.

The fields of the SBE header are:

- Block length of the message root - the total space reserved for the root level of the message not counting any repeating groups or variable-length fields
- Template ID - identifier of the message template.
- Schema ID - identifier of the message schema that contains the template.
- Schema version - the version of the message schema in which the message is defined.

SBE Message Body

The message body conveys the business information of the message and has the following attributes:

- Data only on the wire; without delimiters or metadata, such as tags.
- Access to data is positional, guided by a message schema that specifies a message type.

SBE has two features to control alignment of message elements:

- The length of a block is controlled with the **blockLength** attribute of a message or group. When applied to a message, it controls the length of the root level of the message, prior to any repeating groups or variable-length data. When applied to a group, it controls the length of each entry of the repeating group. The blockLength attribute is not required. By default, the length of a block is the sum of its field lengths. When specified, it must be at least that much, but can be greater. When greater than the field lengths, the extra space resides at the end of the block.
- The position of an individual field can be controlled with the **offset** attribute. Since individual field alignment is not considered crucial, this field attribute is optional. When not specified, the field is packed to the previous field without padding.

SBE Message Example

The following is an example of the schema used to interpret the iLink 3 New Order - Single message:

```
<ns2:message id="514" description="NewOrderSingle" name="NewOrderSingle514" semanticType="D" blockLength="116">

    <field id="44" description="Price per share or contract. Conditionally required if the order type requires a price (not market orders)" name="
    Price" semanticType="Price" type="PRICENULL9" offset="0"/>
    <field id="38" description="Number of shares or contracts ordered" name="OrderQty" semanticType="int" type="uInt32" offset="8"/>
    <field id="48" description="Security ID as defined by CME. For the security ID list, see the security definition messages" name="SecurityID"
    semanticType="int" type="Int32" offset="12"/>
    <field id="54" description="Side of order" name="Side" semanticType="int" type="SideReq" offset="16"/>
    <field id="9726" description="Sequence number as assigned to message" name="SeqNum" semanticType="int" type="uInt32" offset="17"/>
    <field id="5392" description="Operator ID. Should be unique per Firm ID. Assigned value used to identify specific message originator.
    Represents last individual or team in charge of the system which modifies the order before submission to the Globex platform, or if not
    modified from initiator (party role=118), last individual or team in charge of the system, which submit the order to the Globex platform"
    name="SenderID" semanticType="String" type="String20Req" offset="21"/>
    <field id="11" description="Unique identifier for Order as assigned by the buy-side (institution, broker, intermediary etc.). Uniqueness must
    be guaranteed within a single trading day. Firms, particularly those which electronically submit multi-day orders, trade globally or throughout
    market close periods, should ensure uniqueness across days, for example by embedding a date within the ClOrdID field" name="ClOrdID"
    semanticType="String" type="String20Req" offset="41"/>
    <field id="1505" description="Refers to the ID of the related PartyDetailsDefinitionRequest message which will logically be tied to this
    message" name="PartyDetailsListReqID" semanticType="int" type="uInt64" offset="61"/>
    <field id="2422" description="Use OrderRequestID to identify a request to enter, modify or delete an order and echo the value on the
    ExecutionReport representing the response" name="OrderRequestID" semanticType="int" type="uInt64" offset="69"/>
    <field id="5297" description="Time when the message is sent. 64-bit integer expressing the number of nano seconds since midnight
    January 1, 1970." name="SendingTimeEpoch" semanticType="int" type="uInt64" offset="77"/>
    <field id="99" description="The stop price of a stop protect or stop limit order. (Conditionally required if OrdType = 3 or 4)." name="StopPx"
    semanticType="Price" type="PRICENULL9" offset="85"/>
    <field id="9537" description="Text describing sender's location (i.e. geopraphic location and/or desk)" name="Location" semanticType="
    String" type="String5Req" offset="93"/>
    <field id="110" description="Minimum quantity of an order to be executed" name="MinQty" semanticType="int" type="uInt32NULL" offset="
    98"/>
    <field id="1138" description="The quantity to be displayed . Required for iceberg orders. On orders specifies the qty to be displayed, on
    execution reports the currently displayed quantity" name="DisplayQty" semanticType="int" type="uInt32NULL" offset="102"/>
    <field id="432" description="Date of order expiration (last day the order can trade), always expressed in terms of the local market date.
    Applicable only to GTD orders which expire at the end of the trading session specified. This has to be a future or current session date and
    cannot be in the past." name="ExpireDate" semanticType="LocalMktDate" type="LocalMktDate" offset="106"/>
    <field id="40" description="Order type" name="OrdType" semanticType="char" type="OrderTypeReq" offset="108"/>
    <field id="59" description="Specifies how long the order remains in effect" name="TimeInForce" semanticType="int" type="TimeInForce"
    offset="109"/>
    <field id="1028" description="Indicates if the order was initially received manually (as opposed to electronically)" name="
    ManualOrderIndicator" semanticType="int" type="ManualOrdIndReq" offset="110"/>
    <field id="18" description="Instructions for order handling on exchange. Since more than one instruction is applicable to an order, this field
    can represent those using a bitset." name="ExecInst" semanticType="MultipleCharValue" type="ExecInst" offset="111"/>
    <field id="5906" description="Identifies whether the order should be treated as passive (will not match when entered) or aggressive (could
    match when entered); default behavior when absent is aggressive" name="ExecutionMode" semanticType="char" type="ExecMode" offset="
    112"/>
    <field id="9373" description="New field added to capture if an order was submitted for market making obligation or not. Applicable only for
    EU fixed income markets" name="LiquidityFlag" semanticType="int" type="BooleanNULL" offset="113"/>
    <field id="6881" description="Boolean: flags a managed order" name="ManagedOrder" semanticType="int" type="BooleanNULL" offset="
    114"/>
    <field id="5409" description="Indicates the type of short sale. Will not be used for Buy orders but Sell orders should have this tag populated
    for MiFID" name="ShortSaleType" semanticType="int" type="ShortSaleType" offset="115"/>

</ns2:message>
```

This standard describes how fields are encoded and the general structure of messages. The content of a message type is specified by a message schema. A message schema tells which fields belong to a message and their location within a message. Additionally, the metadata describes valid value ranges and information that need not be sent on the wire, such as constant values. For iLink 3 Simple Binary Encoding, message schemas are expressed as an XML template.Message Schema.

A FIX template corresponds to a FIX message type, and uniquely identifies an ordered collection of fields. The template also includes syntax indicating the type of field and transfer decoding to apply. A template is communicated between CME Group and client systems in XML syntax using the Simple Binary Encoding XML schema. The XML format is human- and machine-readable and can be used for authoring and storing FIX templates

Each template has a unique Template ID that describes the format of the binary encoded message. A Template ID is present in every message to provide a reference to the correct template. The Template ID is unique and is included in every message header, allowing the client system to apply the correct schema to the message upon receiving it.

New Order Single Message

8000FECA74000202080000000E876481700000001000000CBA70D000101000000437563756D6265720000000000000000000000000595A373334
0000000000000000000000000000007B00000000000000DE02000000000000D7096EAB9B27BB15FFFFFFFFFFFFFF7F4D696E736B00000000000
000000FFFF3200000000FFFFFF

The following is the translation of the New Order Single message:

Binary Values	Field Name	Binary Length	Values
80 00	Message size	2	128
FE CA	Encoding type	2	CME SBE version 1.0 little-endian
74 00	Block length	2	116
02 02	Template ID	2	514
08 00	Schema ID	2	8
00 00	Schema version	2	0
00 E8 76 48 17 00 00 00	Price	8	100000000000,-9
01 00 00 00	OrderQty	4	1
CB A7 0D 00	SecurityID	4	894923
01	Side	1	Buy
01 00 00 00	SeqNum	4	1
43 75 63 75 6D 62 65 72 00 00 00 00 00 00 00 00 00 00 00 00	SenderID	20	Cucumber
59 5A 37 33 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	ClOrdID	20	YZ734
7B 00 00 00 00 00 00 00	PartyDetailsListRequestID	8	123
DE 02 00 00 00 00 00 00	OrderRequestID	8	734
D7096EAB9B27BB15	SendingTimeEpoch	8	1565888844990908887
FF FF FF FF FF FF FF 7F	StopPx	8	9223372036854775807,-9
4D 69 6E 73 6B	Location	5	Minsk
00 00 00 00	MinQty	4	0
00 00 00 00	DisplayQty	4	0
FF FF	ExpireDate	2	65535
32	OrdType	1	Limit
00	TimeInForce	1	Day
00	ManualOrderIndicator	1	Automated
00	ExecInst	1	00000000
00	ExecutionMode	1	null
FF	LiquidityFlag	1	null
FF	ManagedOrder	1	null
FF	ShortSaleType	1	null

Schema Distribution

iLink 3 schema file will be available for download via the SFTP site (sftpng.cmegroup.com), CME Globex network direct connection. This SFTP site contains the schema files for all environments. The SFTP site is a secure site that requires a user name and password for access.

Information applies as follows in the table:

- Environment - specific environment (i.e., Certification, New Release, Production)

> The AutoCert+ tool will indicate which environment you need to connect to for certification. For additional information on AutoCert+ access, refer to the AutoCert+ Access Guide.

- Service - the Schema services
- SFTP Site - address of SFTP site.
- User Name - identifies the user name.
- Password - identifies the password.
- Directory Location - identifies the directory.
- Client System Update Schedule - Client systems should download updates according to the schedule specified.

Environment*	Service	SFTP Site	User Name	Password	Directory Location	Client System Update Schedule
Certification	Schema	sftpng.cmegroup.com	cmeconfig	G3t(0nnect3d	/MSGW/Cert/Templates	Sunday prior to market open

New Release					/MSGW/NRCert/Templates	Sunday prior to market open
Production					/MSGW/Production/Templates	Sunday prior to market open

In addition to the generic User Name/Password, client systems can connect using the same credential currently used for CME SFTP site. Additional information pertaining to the CME Secure SFTP site is available in CME Clearing Advisory Notice 15-105.

Schema Versioning

The schema.xml file is versioned each time an update is made. All elements in a message schema are of the same version. The first version of a schema is version zero, and the version number is incremented each time a schema is changed.

iLink 3 message schema is backward compatible. Customers can send messages using the preceding version of a schema, however CME Globex will only send messages using the latest version of a schema.

Schema Header

The following is the layout of the SBE Schema header:

<ns2:messageSchema id="8" xsi:schemaLocation="http://www.fixtradingcommunity.org/pg/file/fplpo/read/1196759/simple-binary-encoding-rc2xsd SimpleBinary-RC2.xsd" byteOrder="**littleEndian**"

description="**20180225**"

semanticVersion="**FIX5.0**"

version="**0**"

package="**iLinkBinary**"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ns2="http://www.fixprotocol.org/ns/simple/1.0">

Schema Format

The schema file is constructed of several sections including the following:

Primitive Type Encodings - Valid encoding specifications for each of the integer primitive types.

Simple Encodings - Definition of specific values.

Composite Encodings - Specifies the range of valid values for a decimal field.

Enumerations - Conveys a single choice of mutually exclusive valid values.

Schema:

- **Message Description**
  - ID - The Schema ID used identify the Template ID.
  - Field Description - Field name
  - Name - FIX tag name
  - Semantic Type - FIX Message Type value for the message
  - Block Length - The length of the root level or the repeating group of the message in bytes of the field
- **Field Descriptions**
  - ID - FIX tag
  - Field Description - Field name
  - Name - FIX tag name.
  - Semantic Type - FIX semantic data type
  - Type - Binary encoding type
  - Offset - The number of bytes from the start of the message body or group to the first byte of the field
- **Repeating Group Definitions**
  - ID - FIX Tag.
  - Field Description - Field Name.
  - Name - FIX tag name.
  - Block Length - the length of the root level or the repeating group of the message in bytes
  - Dimension Type - Dimensions of the repeating group

Template Extension

In the event that a message template is updated, the change can be made as an *appended* extension, which requires no immediate action on the part of the client, or a *non-appended* extension, which requires implementation of the new template.

Template extensions can be applied to the body and/or repeating groups of an SBE message. When a template is extended as an appended extension, customers can choose to continue using the prior template version or process the new data with the new schema version. Customers who support template extension should verify any SBE schema update in New Release prior to a Production schema launch to ensure compatibility.

For the **body** and **repeating groups** of SBE messages, appended template extension provides the following benefits:

- Templates are backward compatible
- Template ID changes are less frequent
- Expedited development time

In the event of an appended template extension, the following conditions will apply:

- Tags added to the **body** of the message will be added to the end of the body, before the repeating groups. **In this scenario, the TemplateID will not change, but the schema version will change.**
- The BlockLength field in the Binary Header or Group will increase in value to include the length of the new tag.
- Data fields can also be appended at the end of the repeating groups without updating the TemplateID, although the contents of the template and the schema version number will change. Each repeating group will include:
    - the new length of each entry in the repeating group
    - the number of entries in the repeating group