FIX PROTOCOL
INDUSTRY-DRIVEN MESSAGING STANDARD℠

Encoding FIX using Google Protocol Buffers

Release Candidate 2

RELEASE CANDIDATE 2 Document Revision 0.18: March 27, 2014

THIS DOCUMENT IS A RELEASE CANDIDATE FOR A PROPOSED FIX TECHNICAL STANDARD. A RELEASE CANDIDATE HAS BEEN APPROVED BY THE GLOBAL TECHNICAL COMMITTEE AS AN INITIAL STEP IN CREATING A NEW FIX TECHNICAL STANDARD. POTENTIAL ADOPTERS ARE STRONGLY ENCOURAGED TO BEGIN WORKING WITH THE RELEASE CANDIDATE AND TO PROVIDE FEEDBACK TO THE GLOBAL TECHNICAL COMMITTEE AND THE WORKING GROUP THAT SUBMITTED THE PROPOSAL. THE FEEDBACK TO THE RELEASE CANDIDATE WILL DETERMINE IF ANOTHER REVISION AND RELEASE CANDIDATE IS NECESSARY OR IF THE RELEASE CANDIDATE CAN BE PROMOTED TO BECOME A FIX TECHNICAL STANDARD DRAFT.

March 2014

DISCLAIMER

THE INFORMATION CONTAINED HEREIN AND THE FINANCIAL INFORMATION EXCHANGE PROTOCOL (COLLECTIVELY, THE "FIX PROTOCOL") ARE PROVIDED "AS IS" AND NO PERSON OR ENTITY ASSOCIATED WITH THE FIX PROTOCOL MAKES ANY REPRESENTATION OR WARRANTY, EXPRESS OR IMPLIED, AS TO THE FIX PROTOCOL (OR THE RESULTS TO BE OBTAINED BY THE USE THEREOF) OR ANY OTHER MATTER AND EACH SUCH PERSON AND ENTITY SPECIFICALLY DISCLAIMS ANY WARRANTY OF ORIGINALITY, ACCURACY, COMPLETENESS, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. SUCH PERSONS AND ENTITIES DO NOT WARRANT THAT THE FIX PROTOCOL WILL CONFORM TO ANY DESCRIPTION THEREOF OR BE FREE OF ERRORS. THE ENTIRE RISK OF ANY USE OF THE FIX PROTOCOL IS ASSUMED BY THE USER.

NO PERSON OR ENTITY ASSOCIATED WITH THE FIX PROTOCOL SHALL HAVE ANY LIABILITY FOR DAMAGES OF ANY KIND ARISING IN ANY MANNER OUT OF OR IN CONNECTION WITH ANY USER'S USE OF (OR ANY INABILITY TO USE) THE FIX PROTOCOL, WHETHER DIRECT, INDIRECT, INCIDENTAL, SPECIAL OR CONSEQUENTIAL (INCLUDING, WITHOUT LIMITATION, LOSS OF DATA, LOSS OF USE, CLAIMS OF THIRD PARTIES OR LOST PROFITS OR REVENUES OR OTHER ECONOMIC LOSS), WHETHER IN TORT (INCLUDING NEGLIGENCE AND STRICT LIABILITY), CONTRACT OR OTHERWISE, WHETHER OR NOT ANY SUCH PERSON OR ENTITY HAS BEEN ADVISED OF, OR OTHERWISE MIGHT HAVE ANTICIPATED THE POSSIBILITY OF, SUCH DAMAGES.

DRAFT OR NOT RATIFIED PROPOSALS (REFER TO PROPOSAL STATUS AND/OR SUBMISSION STATUS ON COVER PAGE) ARE PROVIDED "AS IS" TO INTERESTED PARTIES FOR DISCUSSION ONLY. PARTIES THAT CHOOSE TO IMPLEMENT THIS DRAFT PROPOSAL DO SO AT THEIR OWN RISK. IT IS A DRAFT DOCUMENT AND MAY BE UPDATED, REPLACED, OR MADE OBSOLETE BY OTHER DOCUMENTS AT ANY TIME. THE FPL GLOBAL TECHNICAL COMMITTEE WILL NOT ALLOW EARLY IMPLEMENTATION TO CONSTRAIN ITS ABILITY TO MAKE CHANGES TO THIS SPECIFICATION PRIOR TO FINAL RELEASE. IT IS INAPPROPRIATE TO USE FPL WORKING DRAFTS AS REFERENCE MATERIAL OR TO CITE THEM AS OTHER THAN “WORKS IN PROGRESS”. THE FPL GLOBAL TECHNICAL COMMITTEE WILL ISSUE, UPON COMPLETION OF REVIEW AND RATIFICATION, AN OFFICIAL STATUS ("APPROVED") OF/FOR THE PROPOSAL AND A RELEASE NUMBER.

Document History

Revison	Date	Author(s)	Comments
0.9	Mar 25, 2013	Greg Malatestinic, Jordan & Jordan Sara Rosen, EBS Alessandro Triglia, OSS Nokalva	Draft provided to HPWG for review.
0.91	Apr 19, 2013	Greg Malatestinic, Jordan & Jordan	Changes due to feedback from HPWG and GTC reviews.
RC2 0.15	Nov 22, 2013	Greg Malatestinic, Jordan & Jordan Sara Rosen, EBS Joe Wood, Credit Suisse	Incorporating new material and changes due to feedback during public comment period. Draft is now marked as Release Candidate 2. Submitted to HPWG for review.
RC2 0.16	Nov 25, 2013	Greg Malatestinic, Jordan & Jordan Sara Rosen, EBS Joe Wood, Credit Suisse	Changes based on feedback from HPWG review. Submitted to GTC.
RC2 0.17	Jan 27, 2014	Greg Malatestinic, Jordan & Jordan	Changes based on feedback from GTC.
RC2 0.18	Mar 27, 2014	Greg Malatestinic, Jordan & Jordan	Changes based on introduction of message encoding headers.

CONTENTS

1 Introduction

Google Protocol Buffers is a language-neutral, platform-neutral, extensible mechanism for serializing structured data. It was originally developed by Google to deal with an index server request/response protocol and has become their standard data interchange format. It was made available to the Open Source community in 2008. GBP allows programmers to define simple data structures in a specialized definition language and generate source code containing classes representing those structures. These classes come complete with optimized code to parse and serialize data in a compact format and they allow easy access to data fields via “get” and “set” methods.

This technical specification contains provisions for the mapping of the content of the FIX Unified Repository to the Google Protocol Buffers (GPB) language according to the GPB definition language (or “.proto”-file syntax). These provisions are for the purpose of enabling the exchange of FIX messages between two endpoints using the binary encoding defined by GPB.

The mapping procedure specified in this technical specification can be applied either to the original FIX Unified Repository or to any other XML document that contains a <fix> element with the same syntax as the one in the FIX Unified Repository. For example, a user of FIX could take a subset of the Repository, apply one or more scenarios to some messages, add encoding attributes to a few fields, and finally apply the mapping procedure to the resulting XML document, thus producing a GPB schema (or “.proto” file).

This technical specification is intended to be used in an algorithmic fashion. Clause 4 contains general provisions and drives the whole mapping procedure. Clause 5 (mapping of FIX datatypes) and clause 6 (mapping of FIX messages) are invoked by subclause 4.1.3. Clause 7 (mapping of a FIX component) is invoked by subclause 6.2.2 and by itself (recursively).

2 References

1. Google Protocol Buffers Development Guide and API Reference, https://developers.google.com/protocol-buffers

2. FIX Unified Repository 2010 Edition, http://www.fixtradingcommunity.org/pg/structure/tech-specs/fix-tools/fix-repository

3 Definitions

GPB scalar type	A fundamental type provided by GPB. These include: • double • float • uint32 – a 32 bit unsigned integer with variable-length encoding • uint64 – a 64 bit unsigned integer with variable-length encoding • sint32 – a 32 bit signed inter with variable-length encoding • sint64 – a 64 bit signed integer with variable-length encoding • fixed32 – a 32 bit unsigned integer always using four bytes • fixed64 – a 64 bit unsigned integer always using eight bytes • sfixed32 – a 32 bit signed integer always using four bytes • sfixed64 – a 64 bit signed integer always using eight bytes • bool • string • bytes – any arbitrary sequence of bytes
source <fix> element	An XML element named `<fix>` (usually, but not necessarily, located in the FIX Unified Repository) that is used as the source of the mapping to GPB.

4 General provisions

4.1 Generation of the Google Protocol Buffers schema

4.1.1 The mapping process specified in this technical specification takes as input a <fix> element (called the *source <fix> element*), whose syntax and semantics are identical to those of the <fix> element of the FIX Unified Repository v. 5.0 SP2. The mapping process generates a Google Protocol Buffers schema, consisting of a set of GPB message and enumeration types.

4.1.2 The *source <fix> element* can be one of the following:

a) a <fix> element that is part of a FIX Unified Repository instance;
b) a <fix> element derived from (a) by excluding part of its content (e.g., keeping only the messages belonging to one or more specified FIX sections and/or FIX categories);
c) a <fix> element derived from (a) or (b) by applying one or more specified scenarios to it;
d) any <fix> element whose syntax and semantics are identical to those of the <fix> element of the FIX Unified Repository v. 5.0 SP2.

4.1.3 GPB message types in the GPB schema shall be generated from the *source <fix> element* as specified in clauses 5 and 6,

EXAMPLE

The following set of GPB message types could be generated by the mapping process.

```
message HopGrp { // generated from a <component> element
      optional string hop_comp_id = 1           [(fix.tag)=628];
      optional fixed64 hop_sending_time = 2;    [(fix.tag)=629];
      optional fixed64 hop_ref_id = 3;          [(fix.tag)=630];
}

mesaage StandardHeader { // generated from a <component> element
      optional string sender_comp_id = 1        [(fix.tag)=49];
      optional string target_comp_id = 2        [(fix.tag)=56];
      optional string on_behalf_of_comp_id = 3  [(fix.tag)=115];
      optional string deliver_to_comp_id = 4    [(fix.tag)=128];
      optional fixed64 sending_time = 5         [(fix.tag)=52];
      optional bool poss_resend = 5             [(fix.tag)=97];
      repeated HopGrp hop_grp = 6               [(fix.tag)=627];
}

enum IoiTransTypeEnum { // generated from a <field> with <enum>s
      IOI_TRANS_TYPE_CANCEL = 0;                [(fix.enum)="C"];
      IOI_TRANS_TYPE_NEW = 1                    [(fix.enum)="N"];
      IOI_TRANS_TYPE_REPLACE = 2                [(fix.enum)="R"];
}

message Ioi { // generated from a <message> element
```



```
      option (fix.mag_type)="6";
      optional StandardHeader standard_header = 1;
      optional string ioi_id = 2                [(fix.tag)=23];
      optional IoiTransTypeEnum ioi_trans_type = 3    [(fix.tag)=28];
      optional Instrument instrument = 4;
      optional SideEnum side = 5                [(fix.tag)=54];
      optional IoiQty ioi_qty = 6               [(fix.tag)=27];
      optional sfixed64 price = 7               [(fix.tag)=44];
      optional sfixed32 price_exponent = 8;
      optional bool ioi_natural_flag = 9        [(fix.tag)=130];
}
```

4.2 GPB encoding attributes

As a way to provide metadata to assist in the mapping, the repository can be annotated with encoding attributes. These attributes may be used as the basis to determine the most favorable encoding when there are several options, or they may be considered when formulating mapping rules. For example, if the GPB scaler types, sfixed32 and sfixed64, both satisfy a mapping from a FIX integral type, then the values of <encodingInfo> attributes @minValue, @maxValue and @numBits will be taken into consideration in determining which integer type to select.

4.2.1 Within the *source* `<fix>` *element*, *encoding information* is carried by a set of *encoding attributes* within *encoding information* elements. The *encoding information* elements that affect the mapping to GPB are the elements `<encodingInfo>`.

4.2.2 The `<encodingInfo>` element may contain *encoding attributes* applicable to all standard FIX encodings. All the attributes of an *encoding information* element are optional. An *encoding information* element is allowed to be empty.

4.2.3 Each `<datatype>`, `<field>`, `<fieldRef>`, `<component>`, `<componentRef>`, or `<message>` element in the *source* `<fix>` *element* may contain zero or more *encoding information* elements, but it never contains two *encoding information* elements with the same name.

4.2.4 The encoding attributes are summarized in table 1.

Table 1 –Encoding Attributes

Name	Type	In Repository elements[a]	Applicable to FIX datatypes[b]	Description	Default value[c]	GPB-relevance[d]

minValue	integer	<datatype> <field> <fieldRef>	int Pattern	minimum permitted value of the integer datatype	negative infinity	If minValue ≥ 0, then an unsigned integer GPB datatype will be selected. Otherwise the minValue will be used to determine the size of the signed integer.
maxValue	integer	<datatype> <field> <fieldRef>	int Pattern	maximum permitted value of the integer datatype	positive infinity	Used to determine the size of the GBP integer data type.
isUTF8	boolean	<datatype> <field> <fieldRef>	data	whether the data consists of Unicode characters encoded in UTF-8	false	The FIX datatype, "data", is mapped to an arbitrary sequence of the GPB datatype "bytes". GPB encoding is un-affected by this attribute.
minLength	integer	<datatype> <field> <fieldRef>	String Pattern	minimum permitted length of the string datatype (minimum number of characters)	zero	Not enforced by GPB. GPB strings are of unbounded length.
maxLength	integer	<datatype>, <field>, <fieldRef>	String Pattern	maximum permitted length of the string datatype (maximum number of characters)	positive infinity	Same as above.
numBits	integer	<datatype> <field> <fieldRef>	float UTCTimeStamp TZTimeStamp	maximum size in bits of the data element	64	This attribute will provide an upper limit on the size of integers.

isBinaryFloat	boolean	<datatype> <field> <fieldRef>	Qty Price PriceOffset Amt Percentage float	Indicates whether to generate an IEEE-754 floating point. Mutually exclusive of *isFixedPoint*. (They both cannot be true.)	false	If *isBinaryFloat* is true, will generate a double (the GPB IEEE-754 equivilent). Otherwise one of sfixed32, sfixed64, fixed32, or fixed64 (depending on the range requirements given by minValue and maxValue) will be generated to represent a mantissa and a supplemental sfixed32 will be generated to represent an exponent.
isFixedPoint	boolean	<datatype> <field> <fieldRef>	Qty Price PriceOffset Amt Percentage float	Indicates whether the exponent is fixed (true) or variable (false). Mutually exclusive of *isBinaryFloat*. (They both cannot be true.)	false	
exponent	integer	<datatype> <field> <fieldRef>	Qty Price PriceOffset Amt Percentage float	When isFixedPoint is true this specifies the fixed exponent which is used to determine the decimal position of the integer representation. When isFixedPoint is false this encoding attribute should not be provided. May be positive or negative. Scaled value = mantissa x $10^{exponent}$	None – null value allowed	Exponent has no default value and, therefore, can be null. Used to determine default value of supplemental fields generated for decimal types.

timeUnit	string restricted to specific enum values	<datatype> <field> <fieldRef>	UTCTimeOnly UTCTimeStamp TZTimeOnly TZTimeStamp	Valid values are: • "day" • "second" • "millisecond" • "microsecond" • "nanosecond" • "picosecond"	None – null value allowed	Time unit for offset to epoch. May be specified in an encoding attribute or communicated in Rules of Engagement
epoch	string restricted to specific enum values	<datatype> <field> <fieldRef>	UTCDateOnly UTCTimeStamp LocalMktDate TZTimeStamp	Valid values are: • "midnight" – of any given date • "unix" – Jan 1, 1970 • "1900" – Jan 1, 1900 • "2000" – Jan 1, 2000	"unix"	Epoch may be specified in an encoding attribute or communicated in Rules of Engagement
isNumeric	boolean	<datatype> <field> <fieldRef>	String Pattern	Indicates whether a string field will always contain numeric characters.	false	Identifies an opportunity to optimize by mapping a String to an integer.
gpbType	String restricted to specific enum values	<datatype> <field> <fieldRef>	(All)	Generic mechanism to directly specify the type overriding all mapping rules. Valid values are any GPB scalar type.	None – null value allowed	Overriding type must be a GPB scaler type

[a] This column indicates the FIX Repository elements in which the encoding information element (<GPB> or <encodingInfo>) containing the encoding attribute may appear.

[b] This column indicates the FIX datatypes to which the encoding attribute applies. An encoding attribute is ignored when used with a FIX datatype to which it does not apply.

[c] The default value is used when an encoding attribute is applicable but is absent.

[d] This column indicates the GPB specific considerations of the encoding attribute.

4.3 Generation of GPB names

This subclause specifies rules for generation of the various names mapped from character strings in the Repository. The transformation of a name depends on the context in which it is used. These rules ensure that the generated GPB files conform to the GPB Style Guidelines as defined in Reference 2.1.

Acronyms require special handling to provide an asthetically pleasing result. We call these acronyms the case-sensitive acronyms. When these acronyms are found within a character string that is to be mapped, they are first converted to camel-case. These acronyms and their mappings are listed in the following table:

Table 2 - Case-sensitive acronym string mapping

Source substr	Target substr
ISDA	Isda
FpML	Fpml
CUSIP	Cusip
ISITC	Isitc
ISIN	Isin
RIC	Ric
USD	Usd
US	Us

Source substr	Target substr
UK	Uk
NERC	Nerc
CDS	Cds
IOI	Ioi
MD	Md
EFP	Efp
GT	Gt
RFQ	Rfq

Source substr	Target substr
CFI	Cfi
ID	Id
XML	Xml
ISO	Iso
CP	Cp
NT	Nt
FX	Fx

The mapping of these acronyms is to be applied according to the order to a top-down, left-to-right precedence. E.g. “USD” must be mapped before “US”.

4.3.1 Datatype names

Source datatypes will map either directly to GPB scalar types, or to supporting GPB types designed to carry the same information as the source type. In both cases, datatype names are defined by the mapping rules, found in section 5.2, 5.3 and 7.5.6, and are not subject to a naming convention.

4.3.2 Field names

The target name of fields declared within GPB messages are derived from the “name” attribute of the source <field> element and are transformed by the following rules:

1. All instances of the case-sensitive acronyms are converted to camel case according to the mapping provided by table 2.
2. Insert an underscore character (‘_’) between any two consecutive characters X and Y such that X is in lower case and Y is in upper case.
3. Convert all characters to lower-case.

The following table lists some examples.

Source field name	Target field name	Applied rule
Account	account	3

AdvRefID	adv_ref_id	1,2,3
IOIID	ioi_id	1,2,3
IOITransType	ioi_trans_type	1,2,3
LegIOIQty	leg_ioi_qty	1, 2,3
MDEntryPx	md_entry_px	1,2,3
GTBookingInst	gt_booking_inst	1,2,3
CFICode	cfi_code	1,2,3
LegCFICode	leg_cfi_code	1,2,3

4.3.3 Enumeration type and enumeration value names

<field> elements with <enum> element children will result in a comparable enumeration type definition in GPB. The name of the target GPB enumeration type follows the camel-case convention and will be derived from the "name" attribute of the source <field> element with the following transformation rules:

1. Maintain the first character as upper case.
2. Append the string value "Enum" to the name.

The target enumeration value names are derived from the "symbolicName" attribute of the <enum> element and the "name" attribute of the source <field> element. According to the following rules:

1. All instances of the case-sensitive acronyms are converted to camel case according to the mapping provided by table 2.
2. Prepend the value of the "name" attribute of the source <field> element followed by an underscore character, leaving its original case intact.
3. Insert an underscore character ('_') between any two consecutive characters X and Y such that X is in lower case and Y is in upper case.
4. Convert all characters to upper case.

For example, the following FIX field definition

```
<field id="21" name="HandlInst" type="char" textId="FIELD_21"
added="FIX.2.7"
                abbrName="HandlInst"
                notReqXML="0">
    <enum value="1" symbolicName="AutomatedExecutionNoIntervention"
                  sort="1"
                  added="FIX.2.7"
                  textId="ENUM_21_1"/>
    <enum value="2" symbolicName="AutomatedExecutionInterventionOK"
                  sort="2"
                  added="FIX.2.7"
```



```
                    textId="ENUM_21_2"/>
    <enum value="3" symbolicName="ManualOrder"
                    sort="3"
                    added="FIX.2.7"
                    textId="ENUM_21_3"/>
</field>
```

Will result in the following GPB enum definition within a GPB message containing a HandlInst field

```
enum HandlInstEnum {
      HANDL_INST_AUTOMATED_EXECUTION_NO_INTERVENTION = 0;
      HANDL_INST_AUTOMATED_EXECUTION_INTERVENTION_OK = 1;
      HANDL_INST_MANUAL_ORDER = 2;
}
```

4.3.4 Component names

All <component> elements will result in a comparable message definition in GPB. When mapping a <component> to a GPB message, transform the <component> “name” attribute as follows:

- All instances of the case-sensitive acronyms are converted to camel case according to the mapping provided by table 2.
- The hyphen character (-) shall be removed resulting in the concatenation of the string appearing before the hyphen and the string appearing after it.

4.3.5 Message names

All <message> elements will result in a comparable message definition in GPB. As with component names, when mapping a <message> to a GPB message, transform the <message> “name” attribute as follows:

- All instances of the case-sensitive acronyms are converted to camel case according to the mapping provided by table 2.
- The hyphen character (-) shall be removed resulting in the concatenation of the string appearing before the hyphen and the string appearing after it.

4.3.6 Package names

All GPB message definitions must be generated within a GPB package. The package specifier is listed within a “.proto” file and is taken directly from the “category” attribute of a <component> element.

4.3.7 File names

All GPB messages and enum definitions that belong to the same GPB package must be defined in the same “.proto” file. The name of the file is derived from (1) the “category” attribute of the <component> elements which generated the messages and enums, and (2) a version number of the .proto definition, as follows:

- Insert a hyphen character (‘-’) between any two consecutive characters X and Y such that X is in lower case and Y is in upper case.
- Convert all characters to lower-case.
- Append the string “.v<N>” where <N> refers to the file version umber
- Append the string “.proto”

For example, the file used to store all components of the category “SingleGeneralOrderHandling” for version 1 of the data definition will be named “single-general-order-handling.proto”.

4.4 Field and enumeration value ordering

When generating a GPB message type, the fields must be ordered according to specific attributes of the source elements as follows:

1) First by *whenAdded* attribute of the element. This attribute indicates the FIX version to which this field was added to the component. The lexigraphical order of FIX versions is based on the respective release dates: FIX.2.7, followed by FIX.3.0, FIX.4.0, FIX.4.1, FIX.4.2, FIX.4.3, FIX.4.4, FIX.5.0/ FIXT.1.1 (released together), FIX.5.0.SP, and FIX.5.0.SP2.
2) Then by *whenAddedEP* attribute with lower-numbered extension packs listed before higher-numbered ones.
3) Then alphabetically by *name*.

The ordering is enforced through the assignment of field numbers. The field number of the first field is to be assigned the value of 1. Field numbers of subsequent fields must increase by integral values.

When generating a GPB enumeration type, the enumeration values must be ordered according to the same criteria used for field ordering except that the first enumeration is to be assigned a value of 0.

5 Mapping of FIX datatypes

5.1 Datatypes implicitly defined via <field> elements

5.1.1 Within the *source* <fix> *element*, the *effective datatype* of a field is the FIX datatype determined from a <field> element, its attributes, and its child elements, as described in 3.

Table 3 – Determination of the effective datatype of a FIX field from a <field> element

Case	Multiple-valued type[a]	<enum> child elements[b]	enum DataType attribute[c]	Different encoding attributes[d]	union DataType attribute[e]	Effective datatype of the FIX field
1	No	no	no	no	no	the FIX datatype indicated in the type attribute of the a <field> element
2	No	no	no	**yes**	no	the FIX datatype indicated in the type attribute, modified by the *effective GPB encoding attributes* of the <field> element
3	No	**yes**	no	no	no	the FIX datatype indicated in the type attribute, restricted by the enumeration specified in the <enum> child elements of this <field> element
4	No	no	**yes**	no	no	the FIX datatype indicated in the type attribute, restricted by the enumeration specified in the <enum> child elements of the <field> element referenced by the enumDataType attribute
5	**Yes**	no	no	no	no	the FIX datatype (either MultipleCharValue or MultipleStringValue) with no restrictions on the items of the space-separated list
6	**Yes**	**yes**	no	no	no	the FIX datatype (either MultipleCharValue or MultipleStringValue) where each item of the space-separated list is required to belong to the enumeration specified in the <enum> child elements of this <field> element
7	**yes**	no	**yes**	no	no	the FIX datatype (either MultipleCharValue or MultipleStringValue) where each item of the space-separated list is required to belong to the enumeration specified in the <enum> child elements of the <field> element referenced by the enumDataType attribute
8	no	no	no	no	**yes**	a union between the type determined in case 1 above and the FIX datatype indicated in the unionDataType attribute

9	no	no	no	**yes**	**yes**	a union between the type determined in case 2 above and the FIX datatype indicated in the `unionDataType` attribute
10	no	**yes**	no	no	**yes**	a union between the type determined in case 3 above and the FIX datatype indicated in the `unionDataType` attribute
11	no	no	**yes**	no	**yes**	a union between the type determined in case 4 above and the FIX datatype indicated in the `unionDataType` attribute

NOTE – Other combinations of column values do not occur

[a] This column indicates whether the `type` attribute of the `<field>` element is a multiple-valued datatype (either `MultipleCharValue` or `MultipleStringValue`)

[b] This column indicates whether the `<field>` element has one or more `<enum>` child elements

[c] This column indicates whether the `<field>` element has an `enumDataType` attribute

[d] This column indicates whether the *effective GPB encoding attributes* of the `<field>` element differ from the *effective GPB encoding attributes* of the `<datatype>` element referenced by the `type` attribute

[e] This column indicates whether the `<field>` element has a `unionDataType` attribute

In all cases of table 3 except case 1, the effective datatype of a field differs from the datatype indicated in the `type` attribute and needs to be mapped separately. The following subclauses specify how to map such implicitly defined datatypes.

5.1.2 For each `<field>` element in the `<fields>` element of the *source* `<fix>` *element*, in order, that has one or more `<enum>` child elements and either

a) it has a `type` other than `MultipleCharValue` or `MultipleStringValue` (cases 3 and 10 of table3); or
b) it is referenced by the `enumDataType` attribute of a `<field>` element having a `type` other than `MultipleCharValue` or `MultipleStringValue` (cases 4 and 11 of table 3),

a GPB enumerated type shall be generated as specified in subclauses 5.1.2.1 to 5.1.2.3.

5.1.2.1 The name in the GPB enumerated type shall be generated from the name of the FIX field with the "Enum" suffix appended in accordance with subclause 4.3 (Generation of Names).

5.1.2.2 The type expression in the GPB enum type shall be an enumerated type expression containing one enumerator for each `<enum>` child element of the `<field>` element. Each enumerator identifier shall be generated from the value of the `symbolicName` attribute of the corresponding `<enum>` element in accordance with subclause 4.3 (Generation of Names).

5.1.2.3 The enumerator values of the GPB enum type shall be generated from the position of each of the <enum> child elements of the <field> element according to the sort order specified by clause 4.4, starting from 0.

5.1.2.4 The custom enum options for each enumeration type expression shall be generated according to table 10 found in section 8.5 entitled, “Generating values for custom enum options”.

NOTE – For a field having the FIX datatype `Currency`, if a list of `<enum>` child elements specifying a limited set of currency codes is added to the `<field>` element, the type of the field will be mapped to a GPB enumerated type. The default mapping of the `Currency` datatype (a string) will not be used for this particular field. The same considerations apply to other FIX datatypes that rely on an externally defined codelist, such as `Exchange`, `Country`, or `Language`.

EXAMPLE 1

The following `<field>` element:

```
<field ... name="AdvSide" type="char" ... abbrName="AdvSide"
notReqXML="0">
      <enum value="B" symbolicName="Buy" ... />
      <enum value="S" symbolicName="Sell" .../>
      <enum value="T" symbolicName="Trade" .../>
      <enum value="X" symbolicName="Cross" .../>
</field>
```

will generate the following GPB enum:

```
enum AdvSideEnum {
      ADV_SIDE_BUY = 0;
      ADV_SIDE_SELL = 1;
      ADV_SIDE_TRADE = 2;
      ADV_SIDE_CROSS = 3;
}
```

EXAMPLE 2

The following `<field>` element:

```
<field ... name="AllocStatus" type="int" ... abbrName="Stat"
notReqXML="0">
      <enum value="0" symbolicName="Accepted" .../>
      <enum value="1" symbolicName="BlockLevelReject" .../>
      <enum value="2" symbolicName="AccountLevelReject" .../>
      <enum value="3" symbolicName="Received" .../>
      <enum value="4" symbolicName="Incomplete" .../>
      <enum value="5" symbolicName="RejectedByIntermediary" .../>
      <enum value="6" symbolicName="AllocationPending" .../>
      <enum value="7" symbolicName="Reversed" .../>
      <enum value="8" symbolicName="CancelledByIntermediary" .../>
      <enum value="9" symbolicName="Claimed" .../>
```



```
      <enum value="10" symbolicName="Refused" .../>
      <enum value="11" symbolicName="PendingGiveUpApproval" .../>
      <enum value="12" symbolicName="Cancelled" .../>
      <enum value="13" symbolicName="PendingTakeUpApproval" .../>
      <enum value="14" symbolicName="ReversalPending" .../>
</field>
```

will generate the following GPB message type:

```
enum AllocStatusEnum {
      ALLOC_STATUS_ACCEPTED = 0;
      ALLOC_STATUS_BLOCK_LEVEL_REJECT = 1;
      ALLOC_STATUS_ACCOUNT_LEVEL_REJECT = 2;
      ALLOC_STATUS_RECEIVED = 3;
      // etc. etc.
}
```

5.1.3 For each `<field>` element in the `<fields>` element of the *source* `<fix>` *element*, in order, that has one or more `<enum>` child elements and either:

a) it has a `type` of `MultipleCharValue` or `MultipleStringValue` (case 6 of table3); or
a) it is referenced by the `enumDataType` attribute of a `<field>` element having a `type` of `MultipleCharValue` or `MultipleStringValue` (case 7 of table 3),

a GPB enumerated type shall be generated as specified in subclauses **Error! Reference source not found.** to 5.1.3.4.

5.1.3.1 The name in the GPB enumerated type shall be generated from the name of the FIX field with the "Enum" suffix appended in accordance with subclause 4.3 (Generation of Names).

5.1.3.2 The type expression in the GPB message type shall be an enumerated type expression containing one enumerator for each `<enum>` child element of the `<field>` element, in order. Each enumerator identifier shall be generated from the value of the `symbolicName` attribute of the corresponding `<enum>` element in accordance with subclause 4.3 (Generation of Names).

5.1.3.3 The enumerator values of the GPB enum type shall be generated from the position of each of the <enum> child elements of the <field> element according to the sort order specified by clause 4.4, starting from 0.

5.1.3.4 The custom enum options for each enumeration type expression shall be generated according to table 10 found in section 8.5 entitled, "Generating values for custom enum options".

EXAMPLE

The following `<field>` element:


```
<field ... id="276" name="QuoteCondition" type="MultipleStringValue"
...>
      <enum value="A" symbolicName="Open" .../>
      <enum value="B" symbolicName="Closed" ... />
      <enum value="C" symbolicName="ExchangeBest" ... />
      <enum value="D" symbolicName="ConsolidatedBest" .../>
</field>
```

will generate the following GPB enum:

```
enum QuoteConditionEnum {
      QUOTE_CONDITION_OPEN = 0;
      QUOTE_CONDITION_CLOSED = 1;
      QUOTE_CONDITION_EXCHANGE_BEST = 2;
      QUOTE_CONDITION_CONSOLIDATED_BEST = 3;
}
```

If we were to map a MDIncGrp component to, which contains a QuoteCondition field, the following may be generated:

```
Message MDIncGrp {
      // ...
      repeated QuoteConditionEnum quote_condition = 1;
      // ...
}
```

5.1.4 For each `<field>` element in the `<fields>` element of the *source* `<fix>` *element*, in order, that has a `unionDataType` attribute (cases 8, 10, and 11 of table 3), a GPB message type shall be generated as specified in subclauses **Error! Reference source not found.** to 5.1.4.4.

5.1.4.1 The name in the GPB message type shall be generated from the name of the FIX field with the "`Union`" suffix appended, in accordance with subclause 4.3.

5.1.4.2 The contents of the GPB message type shall consist of two optional fields. (In GPB, unions are emulated by creating a wrapper message containing an optional field for each possible variant with the understanding that only one of these fields will have been set.)

5.1.4.3 The identifier of the first alternative shall be derived from the *`name`* attribute of the <field> element according to the field naming rules specified in clause 4.3.2, and its type expression shall be determined as specified in table 4 **Error! Reference source not found.**.

Table 4 – Determination of the type expression of the first alternative of the GPB wrapper message

Case of table Error! Reference source not	Type expression of the first alternative	Reference

found.		
8	a reference to the GPB message type generated from the FIX datatype indicated in the `type` attribute of the `<field>` element	subclause 5.1.4 **Error! Reference source not found.**
10	a reference to the enum generated from the `<enum>` child elements of this `<field>` element	subclause 5.1.2
11	a reference to the enum generated from the `<enum>` child elements of the `<field>` element whose `id` is indicated in the `enumDataType` attribute of this `<field>` element	subclause 5.1.2

NOTE – Since the type expression of the first alternative is determined at this stage of the mapping from the `<field>` element, any *textual GPB encoding attribute* of a `<fieldRef>` element referencing this `<field>` element will have no effect on the type expression of the first alternative.

5.1.4.4 The identifier of the each alternative shall be derived from the identifier of the first alternative as follows:

The field name of the subsequent alternatives shall be set to the field name of the first alternative concatenated with the type expression of the particular subsequent alternative where the type expression of the particular subsequent alternative's first character is in upper-case.

The field will also include the "optional" qualifier. Its type expression shall be determined by the *unionDataType* attribute of the source <field> element as specified in table 5.

Table 5 – Determination of the type expression of the second alternative of the GPB wrapper message

Union Type	Type expression of the second alternative
Reserved100Plus	*fixed32*
Reserved1000Plus	*fixed32*
Reserved4000Plus	*fixed32*
Qty	*sfixed64*
Tenor	*Tenor**
* - Tenor is a GPB message that is used to support the mapping. It is not derived from any source FIX element. Its definition can be found in section 5.5.3.	

EXAMPLE 1

The following `<field>` element

```
<field ... id="723" name="PosMaintResult" type="int" ...
            abbrName="Rslt" notReqXML="0"
            unionDataType="Reserved100Plus">
      <enum value="0" symbolicName="SuccessfulCompletion" ... />
      <enum value="1" symbolicName="Rejected" .../>
      <enum value="99" symbolicName="Other" ... />
</field>
```

will generate the following GPB enum and message type

```
enum PosMaintResultEnum {
      POS_MAIN_RESULT_SUCCESSFUL_COMPLETION = 0;
      POS_MAIN_RESULT_REJECTED = 1;
      POS_MAIN_RESULT_OTHER = 99;
}

message PosMaintResultUnion {
      optional PosMaintResultEnum pos_maint_result = 2;
      optional fixed32 pos_maint_result_fixed32 = 3;
}
```

EXAMPLE 2

The following `<field>` element

```
<field ... id="63" name="SettlType" type="string" ...
            unionDataType="Tenor">
      <enum value="0" symbolicName="Regular" ... />
      <enum value="1" symbolicName="Cash" ... />
      <enum value="2" symbolicName="NextDay" ... />
      ...
</field>
```

will generate the following GPB message types

```
enum SettlTypeEnum {
      SETTL_TYPE_REGULAR = 0;
      SETTL_TYPE_CASH = 1;
      SETTL_TYPE_NEXT_DAY = 2;
      // …
}
message Tenor {
      optional fixed32 days = 1;
      optional fixed32 weeks = 2;
      optional fixed32 months = 3;
      optional fixed32 years = 4;
}
```



```
message SettlTypeUnion {
      optional SettlTypeEnum settl_type = 1;
      optional Tenor settl_type_tenor = 32
}
```

The following code snippet shows the two different ways a SettlTypeUnion object may be accessed programmatically in Java:

```
SettlTypeUnion u1 = new SettlTypeUnion();
u1.settlTypeTenor.weeks = 6;

SettlTypeUnion u2 = new SettlTypeUnion();
u2.settlType = SETTL_TYPE_NEXT_DAY;
```

EXAMPLE 3

The following `<field>` element:

```
<field ... id="368" name="QuoteEntryRejectReason" type="int" ...
      enumDatatype="300" unionDataType="Reserved100Plus"/>
```

will generate the following GPB enum and message type:

```
enum QuoteEntryRejectReasonEnum {
      QUOTE_ENTRY_REJECT_REASON_UKNOWN_SYMBOL = 0;
      QUOTE_ENTRY_REJECT_REASON_EXCHANGE = 1;
      QUOTE_ENTRY_REJECT_REASON_QUOTE_REQUEST_EXCEEDS_LIMIT = 2;
      // …
}

message QuoteEntryRejectReasonUnion {
      optional QuoteEntryRejectReasonEnum quote_entry_reject_reason =
1;
      optional fixed32 quote_entry_reject_reason_fixed32 = 2;
}
```

The following code snippet shows the two different ways a QuoteEntryRejectReasonUnion object may be accessed programmatically in Java. The first using a provided enumeration value, the second using a user-defined value:

```
QuoteEntryRejectReasonUnion r1 = new QuoteEntryRejectReasonUnion();
r1.quoteEntryRejectReason = QUOTE_ENTRY_REJECT_REASON_UKNOWN_SYMBOL;

QuoteEntryRejectReasonUnion r22 = new QuoteEntryRejectReasonUnion();
r2.quoteEntryRejectReason_fixed32 = 101;
```


5.2 Determination of the target GPB type expression corresponding to a <field> element

The *target GPB expression* corresponding to a `<field>` element shall be determined according the following procedure:

Define the "*source FIX datatype*" as the FIX datatype described by the <datatype> element referenced by the type attribute of the <field> element. Retrieve the GPB type expression from table 6 giving consideration to the values of relevant encoding attributes that are mentioned.

Table 6 – Datatype mapping rules

FIX datatype	GPB type expression		Comments
int	sfixed32	When -2^31<=*minValue* <0 and maxValue<2^31	If the encoding attributes *minValue* or *maxValue* are not provided then their default values of negative infinity and positive infinity are assumed. In this case the GBP type expression would default to sfixed64.
	sfixed64	When *minValue* <-2^31 or *maxValue* >= 2^31	
	fixed32	When *minValue*>=0 and *maxValue* < 2^32	
	fixed64	When *minValue* >= 0 and *maxValue* >= 2^32	
Length	fixed32		
TagNum	fixed32		
SeqNum	fixed32		
NumInGroup	N/A		NumInGroup fields are ignored.
DayOfMonth	fixed32		
Qty Price PriceOffset Amt Percentage float	double	When *isBinaryFloat*=true	
	sfixed32	When *isBinaryFloat*=false and *numBits*<=32 and the values of *minValue* a/nd *maxValue* allow for the use of a signed 32 bit	These types map to two fields: an integer field (sfixed32 or sfixed64) to hold a mantissa value and a supplemental integer field (sfixed32) to hold an exponent value. The GPB type expression specified here

		integer.	refers to the mantissa value. The mantissa field may have a custom field option named exponent. The value of this option is determined by the following rule: If *isFixedPoint*=true and the encoding attribute, *exponent,* is provided and is equal to some value E, then the value shall be set to E. Otherwise this custom field option shall not be specified. The default value of the supplemental field may be specified under the following conditions: If isFixedPoint=true then the default value of the supplemental field shall be set to: • E, where E is the value of the encoding attribute*, exponent* • 0, if *exponent* is not provided and FIX type is Qty, Amt or Percentage • -6, if *exponent* is not provided and FIX type is Price or PriceOffset If isFixedPoint=false then the default option shall not be specified. The rules governing supplemental exponent fields are described in section 7.5.4.
	sfixed64	When *isBinaryFloa*t=false and (*numBits*>32 or the values of *minValue* and *maxValue* require the use of a 64 bit integer).	
char	bytes		The GPB bytes type is an arbitrary sequence of bytes. For char mapping, the sequence will consist of just one byte.
Boolean	bool		
String	string	When isNumeric=false. Otherwise process this field as if it were an int.	If the sending entity knows that a String field will contain numeric characters exclusively then follow the mapping rule for an int.
MultipleCharValue	N/A		Always a repeated enumerated type. There is no direct mapping to a GPB scalar type.
MultipleStringValue	N/A		Always a repeated enumerated type. There is no direct mapping to a GPB scalar type.
Country	string		
Currency	string		

Exchange	string		
UTCDateOnly LocalMktDate	sfixed32		Number of time units since epoch as indicated by the encoding attributes *timeUnit* and *epoch.* Negative value indicates date before the epoch. Field expressions generated from UTCDateOnly types may include a custom field option named “time_unit”. The value of which is determined by the following rule: If the encoding attribute, *timeUnit,* is provided and is equal to some value U, then the value shall be set to U. Otherwise the value shall be set to “TIME_UNIT_DAYS”.
UTCTimeOnly	fixed32	When *timeUnit* = seconds, milliseconds or microseconds	Number of time units since midnight UTC as indicated by the encoding attribute *timeUnit*. The choice of type expression is determined solely by *timeUnit*. Default is fixed64 if *timeUnit* is unspecified.
	fixed64	When *timeUnit* = nanoseconds or picoseconds, or when *timeUnit* is unspecified	
UTCTimestamp	fixed32	when *numBits*=32	Number of time units since the epoch as indicated by the encoding attributes *timeUnits* and *epoch*. *numBits* should be set sufficiently large to cover the possible range of values specified by the values of *timeUnit* and *epoch*.
	fixed64	when *numBits*=64 or is unspecified	
TZTimeOnly	string		Value conforms to standard FIX: String field representing the time represented based on ISO 8601. This is the time with a UTC offset to allow identification of local time and timezone of that time. Format is HH:MM[:SS][Z | [+ | - hh[:mm]]] where HH = 00-23 hours, MM = 00-59 minutes, SS = 00-59 seconds, hh = 01-12 offset hours, mm = 00-59 offset minutes.
TZTimestamp	string		Value conforms to standard FIX:

		String field representing a time/date combination representing local time with an offset to UTC to allow identification of local time and timezone offset of that time. The representation is based on ISO 8601. Format is YYYYMMDD-HH:MM:SS[Z | [+ | - hh[:mm]]] where YYYY = 0000 to 9999, MM = 01-12, DD = 01-31 HH = 00-23 hours, MM = 00-59 minutes, SS = 00-59 seconds, hh = 01-12 offset hours, mm = 00-59 offset minutes.
data	string	
Pattern	string	
Tenor	Tenor	A supporting GPB message. See section 5.5.1.
MonthYear	sfixed32	The number of months since the epoch indicated by the encoding attribute *epoch*. Negative value indicates months before the epoch
Reserved100Plus	fixed32	
Reserved1000Plus	fixed32	
Reserved4000Plus	fixed32	
XMLData	string	
Language	string	
XID	string	
XIDRef	string	

5.3 Determination of the target GPB type expression corresponding to a <fieldRef> element

The *target GPB expression* corresponding to a `<field>` element shall be determined according the following procedure:

Define the "*source FIX datatype" as* the FIX datatype described by the <datatype> element referenced by the type attribute of the <field> element that is referenced by the name attribute of the <fieldRef> element.Retrieve the GPB type expression from table 6 giving consideration to the values of relevant encoding attributes that are mentioned.

5.4 Special handling of MultipleCharValue and MultipleStringValue FIX datatypes

Fields that are MutilpleCharValue or MultipleStringValue are mapped to GPB repeated fields of enumerated characters or strings, where inclusion of each element of the repeating group indicates selection of its associated optional value. MultipleCharValue maps to a repeated sequence of chars. MultipleStringValue maps to a repeated sequence of strings.

Usage of the GPB option [PACKED=true] can be used whenever the repeated field is a scaler type, such as an enum. This will result in a more efficient encoding.

EXAMPLE

The field ExecInst is a MultipleCharValue type where the choice of values is selected from a pool of enumerated values, it can include the [PACKED=true] option. The expression can be written as:

```
repeated ExecInstEnum exec_inst = 21 [packed = true];
```

where the enumerated values are defined in the following GPB enum:

```
enum ExecInstEnum {
      EXEC_INST_STAY_ON_OFFER_SIDE = 0;
      EXEC_INST_NOT_HELD = 1;
      EXEC_INST_WORK = 2;
      EXEC_INST_GO_ALONG = 3;
      EXEC_INST_OVER_THE_DAY = 4;
      EXEC_INST_HELD = 5;
      // ...
}
```

5.5 Supporting Types

5.5.1 Tenor

FIX fields of type Tenor require the generation of a supporting GPB type. A Tenor holds four optional fields where it is expected that one and only one of the fields will be populated.

On the first invocation of this subclause the following GPB type assignment shall be generated:

```
message Tenor {
      optional fixed32 days = 1;
      optional fixed32 weeks = 2;
      optional fixed32 months = 3;
      optional fixed32 years = 4;
}
```


6 Mapping of FIX messages

6.1 The <messages> element in the *source* <fix> *element* contains a list of <message> elements each describing one FIX message. A message consists of a sequence of fields and/or components, each either required or optional.

6.2 For each <message> element in the <messages> element of the *source* <fix> *element*, in order, a GPB message shall be generated as specified in subclauses 6.2.1 to 6.2.2.

6.2.1 The name in the GPB message shall be generated from the name of the FIX message in accordance with subclause 4.3.

6.2.2 The body of the GPB message type shall be determined as specified in subclauses 7.4.2 to 7.4.15.

EXAMPLE

The NewOrderSingle message of the FIX Repository v.5.0 SP2 will generate the following GPB message type:

```
message NewOrderSingle {
      optional string account = 1;
      optional string cl_ord_id = 2;
      optional string currency = 3;
      optional string ex_destination = 4;
      repeated ExecInstEnum exec_inst = 5 [packed=true];
      optional HandlInstEnum handl_inst = 6;
      optional string ioi_id = 7;
      optional sfixed64 max_floor = 8;
      optional sfixed32 max_floor_exponent = 9;
      optional sfixed64 min_qty = 10;
      optional sfixed32 min_qty_exponent = 11;
      optional OrdTypeEnum ord_type = 12;
      optional sfixed64 price = 13;
      optional sfixed32 price_exponent = 14;
      optional ProcessCodeEnum process_code = 15;
      optional sfixed32 settl_date = 16;
      optional SettlTypeUnion settl_type = 17;
      optional SideEnum side = 18;
      optional Session.StandardHeader standard_header = 19;
      optional Session.StandardTrailer standard_trailer = 20;
      optional sfixed64 stop_px = 21;
      optional sfixed32 stop_px_exponent = 22;
      optional string text = 23;
      optional TimeInForceEnum time_in_force = 24;
      optional Common.Instrument instrument = 57;
      //
      // ...
      //
}
```


7 Mapping of a FIX component

7.1 This clause applies as explicitly invoked by other clauses of this technical specification to generate a GPB messages corresponding to a FIX component.

7.2 The `<components>` element in the *source* `<fix>` *element* contains a list of `<component>` elements each describing one FIX component. A component consists of:

(i) a sequence of fields and/or child components, each either required or optional; or
(ii) a repeating group, which in turn contains a sequence of fields and/or child components, each either required or optional

7.3 GPB messages consist of a message name and a list of fields. A field expression consists of a field rule, a type expression, a field name, a field number, a list of standard or custom field options, and a field comment. Field comments must appear at the end of the line of the last line of the field expression inorder to exploit intellisence features of some Integrated Development Environments (IDE).

7.4 GPB messages may be declared within a specific GPB package. GPB packages are declared the “package” specifier in a “.proto” file. Only one package may be specified in any given “.proto” file. Names of the “.proto” files are given according to the package within and must follow the naming convention specified in subclause 4.3.7.

7.5 On the first invocation of this clause for a given `<component>` element, a GPB message shall be generated as specified in subclauses 7.5.1 to 7.5.14.

7.5.1 The target package where the GPB message is generated is determined from the value of the “category” attribute of the <component> element. The declaration of the GPB message must go in the appropriate .proto file.

7.5.2 The name of the GPB message shall be generated from the name of the FIX component in accordance with subclause 4.3.5.

7.5.3 The fields of the GPB message shall be generated from the members of the FIX component where one field is generated for each <fieldRef> and <componentRef> child element of the <component> element, in the order specifiefd by subclaue 4.4, with the exceptions specified in subclause 7.5.4.

7.5.4 No fields shall be generated for the following FIX fields:

- any FIX field of type `NumInGroup`;
- any FIX field of type `Length` that has a non-empty `associatedDataTag` attribute;
- the `BeginString` field (FIX field id = `8`);
- the `BodyLength` field (FIX field id = `9`);
- the `CheckSum` field (FIX field id = `10`).

7.5.5 The name of each field shall be generated from the name attribute of the <fieldRef> or <componentRef> element in accordance with subclause 4.3.2.

7.5.6 The field rule and type expression of each field corresponding to a `<fieldRef>` element shall be determined as specified in table 7.

Table 7 – Determination of the type expression of a component of the field specification

Case of table 3	Field rule	Type expression of the field
1	optional	The type expression determined from the type attribute of the <fieldRef> element as specified in subclause 5.2.
3	optional	For non-Boolean field types, a reference to the name of the GPB enum generated from the referenced `<field>` element as specified in subclause 5.1.2. For Boolean field types, the type expression determined from the type attribute of the <fieldRef>
4	optional	A reference to the name of the GPB enum generated from the `<field>` element whose `id` is indicated in the `enumDataType` attribute of the referenced `<field>` element as specified in subclause 5.1.2.
5	repeated	The type expression “bytes” if the type attribute of <fieldRef> is MultipleCharValue, otherwise for a MultipleStringValue the type expression is “string”.
6	repeated	A reference to the name of the GPB enum generated from the referenced `<field>` element as specified in subclause 5.1.3.
7	repeated	A reference to the name of the GPB enum generated from the `<field>` element whose `id` is indicated in the `enumDataType` attribute of the referenced `<field>` element as specified in subclause 5.1.3.
8 10, 11	optional	A reference to the GPB message generated from a union of two datatypes as specified in subclause 5.1.4.

7.5.7 The type expression of each field corresponding to a `<componentRef>` element shall be the name of the GPB message generated by invoking this clause 7 for the referenced `<component>` element.

7.5.8 The field rule of each field corresponding to a `<componentRef>` element, not consisting of a repeating group, shall be “optional”.

7.5.9 The field rule of each field corresponding to a `<componentRef>` element, where the component consists of a repeating group, shall be “repeated”.

7.5.10 The custom options of each field corresponding to a `<fieldRef>` element shall be determined as specified in table 11 which is found in section 8.

7.5.11 The custom options of each field corresponding to a <componentRef> element shall be determined as specified in table 12 whichis found in section 8.

7.5.12 In cases where the member of the FIX <component> is a <fieldRef> element, the field comment shall be generated as specified by clause 8.4.14.

7.5.13 In cases where the member of the FIX <component> is a <componentRef> element, the comment field shall be as specified by clause 8.4.15.

7.5.14 For each <fieldRef> child element of the <component> element that refers to a <field> element with a type attribute of "Qty", "Amt", "Percentage", "Price" or "PriceOffset", an additional field shall be generated. This additional field represents the exponent portion of a floating point number. It is generated as follows:

- The field rule shall be "optional".
- The type of the additional field shall be sfixed32.
- The name of the additional field is formed by the concatenation of the name of the field generated from the <fieldRef> element and the string "_exponent".
- The field number of the addition field shall be one more than the field number of the field generated from the <fieldRef> element. (All fields generated from a <component> must adhere to the sort order imposed by subclause 4.4.)

The generation of specific standard or custom GPB options of the field generated from the <fieldRef> element and the additional field are dependent on the values of several encoding attributes and the type of the <field> element referred to by the <fieldRef> element. The following table shows these dependencies and how they affect the generation of both the field generated from the <fieldRef> element and the additional field.

Table 8 – GPB field generation for FIX decimal types

Case	isFixedPt	exponent	FIX type	Resulting protobuf field declarations
1	True	Some value, e	Qty Amt Percentage Price PriceOffset	optional sfixed<N> <field_name> = <field_num> [(meta.exponent)=e]; optional sfixed32 <field_name>_exponent = <field_num+1> [default=e];
2	True	(not provided)	Qty Amt Percentage	optional sfixed<N> <field_name> = <field_num> ; optional sfixed32 <field_name>_exponent = <field_num+1> [default=0];
			Price PriceOffset	optional sfixed<N> <field_name> = <field_num> ; optional sfixed32 <field_name>_exponent = <field_num+1> [default=-6];

3	False	Some value, e	Qty Amt Percentage Price PriceOffset	optional sfixed<N> <field_name> = <field_num>; optional sfixed32 <field_name>_exponent = <field_num+1> ;
4	False	(not provided)	Qty Amt Percentage Price PriceOffset	optional sfixed<N> <field_name> = <field_num>; optional sfixed32 <field_name>_exponent = <field_num+1> ;

Note:

- in case 1 it is recommended that sending applications never set the value of the 2nd field. This is because receiving applications may not be inclined to get the value of the 2nd field since it is available as a custom optionof the first.
- In case 3 the combination of the encoding attribute values are contradictory. *isFixedPoint*=true implies that the exponent is variable and will always be included in the encoded data, while *exponent*=e implies the exponent is constant and will not be included in the encoded data. Therefore, in this case, the value of the encoding attribute, *exponent* shall be ignored.

EXAMPLE

The following example shows the two GPB fields that are generated from a particular <fieldRef> element of a component.

From the Unified Repository:

```
<field id="6" name="AvgPx" type="Price" added="FIX.2.7"/>

<fieldRef id="6" name="AvgPx" required="1" added="FIX.2.7">
      <encodingInfo
                  isBinaryFloat="false"
                  isFixedPoint="true"
                  exponent="-8"
      />
</fieldRef>
```

Generated GPB fields:

```
optional sfixed64 avx_px = 13 [(meta.exponent)=-8];
optional sfixed32 avx_px_exponent = 14 [default=-8];
```


8 Message Encoding Header for use with GPB Encodings

8.1 When a GPB messages generated from a FIX message is encoded, the encoded message does not carry any indication of the message type.

8.2 The lack of such an indication in an encoded message is a characteristic of GPB. In general, when a GPB schema (one or more .proto files) defines multiple top-level message types, it is assumed that the top-level type used in a particular instance of a message will be conveyed either in a message of a higher-level protocol containing the GPB-encoded message, or in a short header preceding the GPB-encoded message.

8.3 A message encoding header for use with GPB encodings of FIX messages is a data structure defined as follows:

Field	Size (bits)	Description
Proto ID	16	A number that identifies GPB schema
Proto Version Number	16	The version number of the GPB schema
GPB Message Type	32	A number or 4 character ASCII identifier used to identify the top-level GPB type to which the present message conforms

where all fields, when representing numeric values, are unsigned integers in big-endian order.

8.4 The fields *Proto ID* and *Proto Version*, taken together, identify a particular GPB schema containing the top-level GPB type definition to which the current message conforms.

8.5 The field *GPB Message Type* identifies a particular top-level GPB type definition present in the schema identified by *Proto ID* and *Proto Version Number*.

8.6 The mapping between *GPB Message TypeD* values and top-level GPB types within a schema is not standardized. The following two methods are recommended, under the assumption that the only top-level GPB have been derived from FIX messages (see **6.2**):

a) the message type field may be the ASCII code of the FIX message, as defined in the FIX repository, for example “AE” for TradeCaptureReport;
b) otherwise, the message type may reflect the integer enumerated values of the **MsgTypeEnum** enumeration, derived from the <field> element named “MsgType”.

9 Annotation with Repository meta-data and encoding attributes

The GPB mapping makes use of additional metadata in the form of GPB Options. Options are annotations used to add information about the message schema over and above the standard schema constructs offered by the proto schema file. This metadata can be used to decorate fields, messages, packages, enumerated types and enumerated values. Conceptually, GPB Custom Options are similar to Java's decorators or .NET's attributes.

9.1 Standard Options

Protocol Buffers provide a set of predefined options that can be used in any proto file. See reference [1] for more information on predefined options.

9.2 Custom Options

Attributes derived from the FIX repository and encoding attributes are used to embellish the proto schema with metadata in the form of custom options. This metadata helps define message usage and tie the GPB proto files back to the FIX repository.

The table below describes each custom option and its usage.

Table 9 – Custom Options

Option	Applied To	Defined In	Description
time_unit	A date or date/time field	meta.proto	Defines the unit of time of the applied scalar field. e.g. TIME_UNIT_DAYS, TIME_UNIT_MILLISECONDS.
epoch	A date or date/time field	meta.proto	The starting reference point used to calculate the date or date/time offset.
exponent	A fixed or floating point scalar field	meta.proto	Defines the exponent value (base 10) of the fixed or floating point field.
min_value, max_value	GPB scalar field	meta.proto	The minimum and maximum expected value of the field.
min_length, max_length	GPB string field	meta.proto	The minimum and maximum length of a string. When minLength= maxLength, indicates that the string is fixed length.
msg_type	Top-level GPB message	fix.proto	Indicates the FIX message type the GPB message has been generated from. Eg. "AE" for a Trade Capture Report.

tag	Any GPB field generated from a <fieldRef> element	fix.proto	Indicates the FIX tag the field was generated from.
type	Any GPB field	fix.proto	Indicates the FIX Repository data type of the field
field_added	Any GPB field	fix.proto	Indicates the version of FIX in which the field was first added to the component or message
field_added_ep	Any GPB field	fix.proto	Indicates the extension pack (if any) in which the field was first added to the component or message
field_deprecated	Any GPB field	fix.proto	If a field has been deprecated for a given component or message, this field option indicates in which version of FIX it was first deprecated.
enum	GPB enumerated value	fix.proto	Indicates the corresponding enumerated value in the FIX repository.
enum_added	GPB enumerated value	fix.proto	Indicates the version of FIX when the enumerated value was added to the component or message.
enum_added_ep	GPB enumerated value	fix.proto	Indicates the extension pack (if any) of FIX when the enumerated value was added to the component or message.

9.3 Required Types to support Custom Options

The following types are required to support the generation of custom field and enumeration options.

9.3.1 TimeUnit

The following GPB enum called “TimeUnit” shall be generated within the “meta” package.

```
enum TimeUnit {
        TIME_UNIT_DAYS = 0;
        TIME_UNIT_SECONDS = 1;
        TIME_UNIT_MILLISECONDS = 2;
        TIME_UNIT_MICROSECONDS = 3;
        TIME_UNIT_NANOSECONDS = 4;
        TIME_UNIT_PICOSECONDS = 5;
```



```
}
```

The generation of this enum is in addition to and independent from “TimeUnitEnum” which may be generated from the FIX field named “TimeUnit” (tag 997).

9.3.2 Epoch

The following GPB enum called “Epoch” shall be generated within the “meta” package.

```
enum Epoch {
        EPOCH_MIDNIGHT = 0;          // Midnight of any given date
        EPOCH_UNIX = 1;              // Midnight of Jan-01 1970
        EPOCH_1900 = 2;              // Midnight of Jan-01 1900
        EPOCH_2000 = 3;              // Midnight of Jan-01 2000
}
```

9.3.3 Datatype

A GPB enum called “Datatype” shall be generated within the “fix” package with enum expressions generated according to the following procedure:

For each child <datatype> element of the <datatypes> element generate a GPB enumeration expression where the enum name is derived from the “name” attribute of <datatype> according to the following naming convention:

1. All instances of the case-sensitive acronyms are converted to camel case according to the mapping provided by table 2.
2. Insert an underscore character (‘_’) between any tewo consecutive characters X and Y such that X is in lower case and Y is in upper case.
3. Convert all characters to upper case.

Enum values are generated according to the sorting rules of clause 4.4.

For example, the following enum may be generated:

```
enum Datatype {
        CHAR = 0                     [(fix.enum_added)=FIX_2_7];
        DATA = 1                     [(fix.enum_added)=FIX_2_7];
        FLOAT = 2                    [(fix.enum_added)=FIX_2_7];
        INT = 3                      [(fix.enum_added)=FIX_2_7];
        DAY_OF_MONTH = 4             [(fix.enum_added)=FIX_4_1];
        MONTH_YEAR = 5               [(fix.enum_added)=FIX_4_1];
        AMT = 6                      [(fix.enum_added)=FIX_4_2];
        BOOLEAN = 7                  [(fix.enum_added)=FIX_4_2];
        CURRENCY = 8                 [(fix.enum_added)=FIX_4_2];
        EXCHANGE = 9                 [(fix.enum_added)=FIX_4_2];
```



```
        LOCAL_MKT_DATE = 10             [(fix.enum_added)=FIX_4_2];
        MULTIPLE_STRING_VALUE = 11      [(fix.enum_added)=FIX_4_2];
        PRICE = 12                      [(fix.enum_added)=FIX_4_2];
        PRICE_OFFSET = 13               [(fix.enum_added)=FIX_4_2];
        QTY = 14                        [(fix.enum_added)=FIX_4_2];
        STRING = 15                     [(fix.enum_added)=FIX_4_2];
        UTC_TIME_ONLY = 16              [(fix.enum_added)=FIX_4_2];
        UTC_TIMESTAMP = 17              [(fix.enum_added)=FIX_4_2];
        LENGTH = 18                     [(fix.enum_added)=FIX_4_3];
        NUM_IN_GROUP = 19               [(fix.enum_added)=FIX_4_3];
        PERCENTAGE = 20                 [(fix.enum_added)=FIX_4_3];
        SEQ_NUM = 21                    [(fix.enum_added)=FIX_4_3];
        TAG_NUM = 22                    [(fix.enum_added)=FIX_4_3];
        COUNTRY = 23                    [(fix.enum_added)=FIX_4_4];
        MULTIPLE_CHAR_VALUE = 24        [(fix.enum_added)=FIX_4_4];
        PATTERN = 25                    [(fix.enum_added)=FIX_4_4];
        RESERVED1000PLUS = 26           [(fix.enum_added)=FIX_4_4];
        RESERVED100PLUS = 27            [(fix.enum_added)=FIX_4_4];
        RESERVED4000PLUS = 28           [(fix.enum_added)=FIX_4_4];
        TZ_TIME_ONLY = 29               [(fix.enum_added)=FIX_4_4];
        TZ_TIMESTAMP = 30               [(fix.enum_added)=FIX_4_4];
        TENOR = 31                      [(fix.enum_added)=FIX_4_4];
        UTC_DATE_ONLY = 32              [(fix.enum_added)=FIX_4_4];
        XML_DATA = 33                   [(fix.enum_added)=FIX_5_0];
        LANGUAGE = 34                   [(fix.enum_added)=FIX_5_0_SP_1, (fix.enum_added_ep)=90];
}
```

9.3.4 Version

Several custom field options refer to specific FIX versions. For example the custrom field option “added” (see clause 8.2) indicates the FIX version in which a field was added to a particular message or component. The following GPB enum called “Version” shall be generated within the “fix” package.

```
enum Version {
        FIX_2_7 = 0;
        FIX_3_0 = 1;
        FIX_4_0 = 2;
        FIX_4_1 = 3;
        FIX_4_2 = 4;
        FIX_4_3 = 5;
        FIX_4_4 = 6;
        FIX_5_0 = 7;
        FIXT_1_1 = 8;
        FIX_5_0_SP_1 = 9;
        FIX_5_0_SP_2 = 10;
}
```


Note that the FIX repository provides a simple type, Version_t, which is string-based and restricted to specific patterns. However, there is no association between the range of Version_t and their release dates. This association is required by the sorting criteria described in clause 4.4. It is for this reason that Version is statically generated and not mapped from the repository.

9.3.4.1 In the FIX repository attributes of type Version_t appear as strings and contain "." characters instead of "_" characters. To map an attribute that is of type Version_t to the GPB enum Version, the period characters (.) of the attribute value are replaced by underscore characters (_).

For example, the field OrderQty is defined in the Unfified FIX Repository as:

```
<field id="38" name="OrderQty" type="int" textId="Field38" added="FIX.2.7"/>
```

So an OrderQty field appearing as the 11th field of a GPB message would be declared as such:

```
optional sfixed64 order_qty = 11 [(fix.field_added)=FIX_2_7];
```

9.4 Custom Option Declarations

Custom optins listed in table 9 are defined by adding the following declarations within the meta package (file name"meta.proto") and the fix package (filename "fix.proto") as shown below.

meta.proto:

```
import "google/protobuf/descriptor.proto";
package meta;
extend google.protobuf.FieldOptions {
        optional TimeUnit time_unit = 51001;
        optional sfixed32 exponent = 51003;
        optional fixed32 min_length = 51004;
        optional fixed32 max_length = 51005;
        optional sfixed64 min_value = 51006;
        optional sfixed64 max_value = 51007;
}
```

fix.proto:

```
import "google/protobuf/descriptor.proto";
package fix;
extend google.protobuf.FileOptions {
        optional string category = 53002;
}
extend google.protobuf.MessageOptions {
        optional string msg_type = 55001;
}
```



```
extend google.protobuf.FieldOptions {
        optional fixed32 tag = 56003;
        optional Datatype type = 56004;
        optional Version field_added = 56005;
        optional sfixed32 field_added_ep = 56006;
        optional Version field_deprecated = 56007;
}
extend google.protobuf.EnumValueOptions {
        optional string enum = 72004;
        optional Version enum_added = 72005;
        optional sfixed32 enum_added_ep = 72006;
        optional Version enum_deprecated = 72007;
}
```

9.5 Generating values for custon enumeration options

The custom enum options for each enumeration value shall be generated according to table 10.

Table 10 - Determiniation of custom enum value options

Custom option	Type	Value
enum	string	The value attribute of the <enum> element.
enum_added	Version	The enumAdded attribute of the <enum> element mapped to a value of the Version enumeration type as specified in subclause 8.3.4.1.
enum_added_ep	sfixed32	The fieldAddedEP attribute of the <enum> element mapped to a value of the Version enumeration type as specified in subclause 8.3.4.1.
enum_deprecated	Version	The deprecated attribute of the <enum> element mapped to a value of the Version enumeration type as specified in subclause 8.3.4.1.

9.6 Generating values for custom field options

The custom field options for each field of a message shall be generated according to table 15. Note that some custom field options are not applicable for some corresponding repository elements.

Table 11 - Determiniation of custom field options

Custom option	Type	Applicable element	Value
tag	fixed32	<fieldRef>	The tag number of the associated FIX field.
type	Datatype	<fieldRef>	The type of the associated FIX field. Must be one of the enumerated defined in the GPB enum Datatype according to clause 5.4.2.

exponent	sfixed32	<fieldRef>	For <fieldRef> elements with type attributes of Qty, Amt, Percentage, Price or PriceOffset, if isFixedPoint=true and the encoding attribute *exponent* is provided, then this custom option is set to the value of *exponent*.
field_added	Version	<fieldRef> <component>	The fieldAdded attribute of the <fieldRef> element mapped to a value of the Version enumeration type as specified in subclause 8.3.4.1.
field_added_ep	sfixed32	<fieldRef> <component>	The fieldAddedEP attribute of the <fieldRef> element.
field_deprecated	Version	<fieldRef> <component>	The deprecated attribute of the <fieldRef> element mapped to a value of the Version enumeration type as specified in subclause 8.3.4.1.

9.7 Generating values for custom message options

The custom enum options for each enumeration value shall be generated according to table 12.

Table 12 - Determiniation of custom message options

Custom option	Type	Value
msg_type	string	The msgType attribute of the <message> element.

10 Field Descriptions

The FIX repository includes description of fields and components. These descriptions can be carried over to the GPB messages generated from the repository in the form a field-level comments. GPB uses C/C++ style // syntax. Each field comment must appear at the end of the line of the last line of the field expression in order to exploit intellisence features of some Integrated Development Environments (IDE).

10.1 Generating Descriptions for <fieldRef> elements

In cases where the field expression has been generated from a <fieldRef> element, the field comment shall be generated from the first <para> element of the <text> element of the <phrases> element where the textId attribute of the <phrase> element is equal to the concatenation of the Id attribute of the <fieldRef> element and "FIELD_". If there are more than one <para> elements then the comment shall be appended with the string "[cont…]".

For example, given the following <fieldRef> and <phrase> elements

```
<fieldRef id="100" name="ExDestination" required="0" added="FIX.2.7" textId="MSG_14_REF_100"/>
```

```
<phrase textId="FIELD_100">
  <text>
    <para>Execution destination as defined by institution when order id entered.</para>
    <para>Valud values:</para>
    <para>See "Appendix 6-C"</para>
  </text>
</phrase>
```

the following field comment will be generated.

```
// Execution destination as defined by institution when order id entered. [cont…]
```

10.2 Generating Descritions for <componentRef> elements

In cases where is the field expression has been generated from a <componentRef> element, the field comment shall be generated from the first <para> element of the <text> element of the <phrases> element where the textId attribute of the <phrase> element is equal to the textId attribute of the <componentRef> element. If there are more than one <para> elements then the comment shall be appended with the string "[cont…]".

For example, given the following <componentRef> and <phrase> elements

```
<componentRef id="1012" name="Parties" required="0" added="FIX.4.3" textId="MSG_14_REF_Parties"/>
```

```
<phrase textId="MSG_14_REF_Parties">
  <text>
    <para>Insert here the set of "Parties" (firm identification) fields defined in "Common Components of Application Messages"</para>
  </text>
```



```
</phrase>
```

the following field comment will be generated.

```
// Insert here the set of “Parties” (firm identification) fields defined in “Common Components of Application Messages
```


Appendix A Full Listing of IOI

The following listing shows the GPB enums and messages of which the IndicationOfInterest message and all its components are dependent. The IndicationOfInterest message is the final message of the listing.

The following listing shows the GPB enums and messages of which the IOI message and all its components are dependent. The IOI message is the final message of the listing.

```
//
//     FIX Unified Repository mapping to Google Protocol Buffer
//
//     Copyright (c) FIX Protocol Ltd. All Rights Reserved.
//
//     File: meta.proto
//

import "google/protobuf/descriptor.proto";

package meta;

enum TimeUnit {
       TIME_UNIT_DAYS = 1;
       TIME_UNIT_SECONDS = 2;
       TIME_UNIT_MILLISECONDS = 3;
       TIME_UNIT_MICROSECONDS = 4;
       TIME_UNIT_NANOSECONDS = 5;
       TIME_UNIT_PICOSECONDS = 6;
}

enum Epoch {
       EPOCH_MIDNIGHT = 1;        // For time-only fields, midnight of a given date
       EPOCH_UNIX = 2;            // Midnight of Jan-01 1970
       EPOCH_1900 = 3;            // Midnight of Jan-01 1900
       EPOCH_2000 = 4;            // Midnight of Jan-01 2000
}

extend google.protobuf.FieldOptions {
       optional TimeUnit time_unit = 51001;
       optional sfixed32 exponent = 51003;
       optional fixed32 min_length = 51004;
       optional fixed32 max_length = 51005;
       optional sfixed64 min_value = 51006;
       optional sfixed64 max_value = 51007;

}

//
//     FIX Unified Repository mapping to Google Protocol Buffer
//
//     Copyright (c) FIX Protocol Ltd. All Rights Reserved.
//
//     File: fix.proto
//

import "google/protobuf/descriptor.proto";

package fix;
```

```
extend google.protobuf.FileOptions {
	optional string category = 53002;
}

extend google.protobuf.MessageOptions {
	optional string msg_type = 55001;
}

extend google.protobuf.FieldOptions {
	optional int32 tag = 56003;
	optional Datatype type = 56004;
	optional Version field_added = 56005;
	optional sfixed32 field_added_ep = 56006;
	optional Version field_deprecated = 56007;
}

extend google.protobuf.EnumValueOptions {
	optional string enum = 72004;
	optional Version enum_added = 72005;
	optional sfixed32 enum_added_ep = 72006;
	optional Version enum_deprecated = 72007;
}

enum Datatype {
	CHAR = 0                    [(fix.enum_added)=FIX_2_7];
	DATA = 1                    [(fix.enum_added)=FIX_2_7];
	FLOAT = 2                   [(fix.enum_added)=FIX_2_7];
	INT = 3                     [(fix.enum_added)=FIX_2_7];
	DAY_OF_MONTH = 4            [(fix.enum_added)=FIX_4_1];
	MONTH_YEAR = 5              [(fix.enum_added)=FIX_4_1];
	AMT = 6                     [(fix.enum_added)=FIX_4_2];
	BOOLEAN = 7                 [(fix.enum_added)=FIX_4_2];
	CURRENCY = 8                [(fix.enum_added)=FIX_4_2];
	EXCHANGE = 9                [(fix.enum_added)=FIX_4_2];
	LOCAL_MKT_DATE = 10         [(fix.enum_added)=FIX_4_2];
	MULTIPLE_STRING_VALUE = 11 [(fix.enum_added)=FIX_4_2];
	PRICE = 12                  [(fix.enum_added)=FIX_4_2];
	PRICE_OFFSET = 13           [(fix.enum_added)=FIX_4_2];
	QTY = 14                    [(fix.enum_added)=FIX_4_2];
	STRING = 15                 [(fix.enum_added)=FIX_4_2];
	UTC_TIME_ONLY = 16          [(fix.enum_added)=FIX_4_2];
	UTC_TIMESTAMP = 17          [(fix.enum_added)=FIX_4_2];
	LENGTH = 18                 [(fix.enum_added)=FIX_4_3];
	NUM_IN_GROUP = 19           [(fix.enum_added)=FIX_4_3];
	PERCENTAGE = 20             [(fix.enum_added)=FIX_4_3];
	SEQ_NUM = 21                [(fix.enum_added)=FIX_4_3];
	TAG_NUM = 22                [(fix.enum_added)=FIX_4_3];
	COUNTRY = 23                [(fix.enum_added)=FIX_4_4];
	MULTIPLE_CHAR_VALUE = 24    [(fix.enum_added)=FIX_4_4];
	PATTERN = 25                [(fix.enum_added)=FIX_4_4];
	RESERVED1000PLUS = 26       [(fix.enum_added)=FIX_4_4];
	RESERVED100PLUS = 27        [(fix.enum_added)=FIX_4_4];
	RESERVED4000PLUS = 28       [(fix.enum_added)=FIX_4_4];
	TZ_TIME_ONLY = 29           [(fix.enum_added)=FIX_4_4];
	TZ_TIMESTAMP = 30           [(fix.enum_added)=FIX_4_4];
	TENOR = 31                  [(fix.enum_added)=FIX_4_4];
	UTC_DATE_ONLY = 32          [(fix.enum_added)=FIX_4_4];
	XML_DATA = 33               [(fix.enum_added)=FIX_5_0];
	LANGUAGE = 34               [(fix.enum_added)=FIX_5_0_SP_1, (fix.enum_added_ep)=90];
}

enum Version {
```



```
	FIX_2_7 = 0;
	FIX_3_0 = 1;
	FIX_4_0 = 2;
	FIX_4_1 = 3;
	FIX_4_2 = 4;
	FIX_4_3 = 5;
	FIX_4_4 = 6;
	FIX_5_0 = 7;
	FIXT_1_1 = 8;
	FIX_5_0_SP_1 = 9;
	FIX_5_0_SP_2 = 10;
}

message Tenor {
	optional fixed32 days = 1;
	optional fixed32 weeks = 2;
	optional fixed32 months = 3;
	optional fixed32 years = 4;
}

//
//	FIX Unified Repository mapping to Google Protocol Buffer
//
//	Copyright (c) FIX Protocol Ltd. All Rights Reserved.
//
//	File: session.proto
//
//	Category: Session
//

import "meta.proto";
import "fix.proto";

option java_outer_classname = "Session";
option java_package = "org.fixprotocol.components";
package Session;

message HopGrp {
	optional string hop_comp_id = 1		[(fix.tag)=628, (fix.type)=STRING];
	optional fixed32 hop_ref_id = 2		[(fix.tag)=630, (fix.type)=SEQ_NUM];
	optional sfixed64 hop_sending_time = 3	[(fix.tag)=629, (fix.type)=UTC_TIMESTAMP];
}

enum MsgTypeEnum {
	MSG_TYPE_ADJUSTED_POSITION_REPORT = 0		[(fix.enum)="BL"];
	MSG_TYPE_ADVERTISEMENT = 1			[(fix.enum)="7"];
	MSG_TYPE_ALLOCATION_INSTRUCTION = 2		[(fix.enum)="J"];
	MSG_TYPE_ALLOCATION_INSTRUCTION_ACK = 3		[(fix.enum)="P"];
	MSG_TYPE_ALLOCATION_INSTRUCTION_ALERT = 4	[(fix.enum)="BM"];
	MSG_TYPE_ALLOCATION_REPORT = 5			[(fix.enum)="AS"];
	MSG_TYPE_ALLOCATION_REPORT_ACK = 6		[(fix.enum)="AT"];
	MSG_TYPE_APPLICATION_MESSAGE_REPORT = 7		[(fix.enum)="BY"];
	MSG_TYPE_APPLICATION_MESSAGE_REQUEST = 8	[(fix.enum)="BW"];
	MSG_TYPE_APPLICATION_MESSAGE_REQUEST_ACK = 9	[(fix.enum)="BX"];
	MSG_TYPE_ASSIGNMENT_REPORT = 10			[(fix.enum)="AW"];
	MSG_TYPE_BID_REQUEST = 11			[(fix.enum)="k"];
	MSG_TYPE_BID_RESPONSE = 12			[(fix.enum)="l"];
	MSG_TYPE_BUSINESS_MESSAGE_REJECT = 13		[(fix.enum)="j"];
	MSG_TYPE_COLLATERAL_ASSIGNMENT = 14		[(fix.enum)="AY"];
	MSG_TYPE_COLLATERAL_INQUIRY = 15		[(fix.enum)="BB"];
	MSG_TYPE_COLLATERAL_INQUIRY_ACK = 16		[(fix.enum)="BG"];
	MSG_TYPE_COLLATERAL_REPORT = 17			[(fix.enum)="BA"];
	MSG_TYPE_COLLATERAL_REQUEST = 18		[(fix.enum)="AX"];
```

```
MSG_TYPE_COLLATERAL_RESPONSE = 19             [(fix.enum)="AZ"];
MSG_TYPE_CONFIRMATION = 20                    [(fix.enum)="AK"];
MSG_TYPE_CONFIRMATION_ACK = 21                [(fix.enum)="AU"];
MSG_TYPE_CONFIRMATION_REQUEST = 22            [(fix.enum)="BH"];
MSG_TYPE_CONTRARY_INTENTION_REPORT = 23       [(fix.enum)="BO"];
MSG_TYPE_CROSS_ORDER_CANCEL_REPLACE_REQUEST = 24      [(fix.enum)="t"];
MSG_TYPE_CROSS_ORDER_CANCEL_REQUEST = 25              [(fix.enum)="u"];
MSG_TYPE_DERIVATIVE_SECURITY_LIST = 26                [(fix.enum)="AA"];
MSG_TYPE_DERIVATIVE_SECURITY_LIST_REQUEST = 27        [(fix.enum)="z"];
MSG_TYPE_DERIVATIVE_SECURITY_LIST_UPDATE_REPORT = 28  [(fix.enum)="BR"];
MSG_TYPE_DONT_KNOW_TRADE = 29                         [(fix.enum)="Q"];
MSG_TYPE_EMAIL = 30                           [(fix.enum)="C"];
MSG_TYPE_EXECUTION_ACKNOWLEDGEMENT = 31       [(fix.enum)="BN"];
MSG_TYPE_EXECUTION_REPORT = 32                [(fix.enum)="8"];
MSG_TYPE_HEARTBEAT = 33                       [(fix.enum)="0"];
MSG_TYPE_IOI = 34                             [(fix.enum)="6"];
MSG_TYPE_LIST_CANCEL_REQUEST = 35             [(fix.enum)="K"];
MSG_TYPE_LIST_EXECUTE = 36                    [(fix.enum)="L"];
MSG_TYPE_LIST_STATUS = 37                     [(fix.enum)="N"];
MSG_TYPE_LIST_STATUS_REQUEST = 38             [(fix.enum)="M"];
MSG_TYPE_LIST_STRIKE_PRICE = 39               [(fix.enum)="m"];
MSG_TYPE_LOGON = 40                           [(fix.enum)="A"];
MSG_TYPE_LOGOUT = 41                          [(fix.enum)="5"];
MSG_TYPE_MARKET_DATA_INCREMENTAL_REFRESH = 42 [(fix.enum)="X"];
MSG_TYPE_MARKET_DATA_REQUEST = 43             [(fix.enum)="V"];
MSG_TYPE_MARKET_DATA_REQUEST_REJECT = 44              [(fix.enum)="Y"];
MSG_TYPE_MARKET_DATA_SNAPSHOT_FULL_REFRESH = 45       [(fix.enum)="W"];
MSG_TYPE_MARKET_DEFINITION = 46                       [(fix.enum)="BU"];
MSG_TYPE_MARKET_DEFINITION_REQUEST = 47               [(fix.enum)="BT"];
MSG_TYPE_MARKET_DEFINITION_UPDATE_REPORT = 48         [(fix.enum)="BV"];
MSG_TYPE_MASS_QUOTE = 49                      [(fix.enum)="i"];
MSG_TYPE_MASS_QUOTE_ACKNOWLEDGEMENT = 50      [(fix.enum)="b"];
MSG_TYPE_MULTILEG_ORDER_CANCEL_REPLACE = 51           [(fix.enum)="AC"];
MSG_TYPE_NETWORK_COUNTERPARTY_SYSTEM_STATUS_REQUEST = 52    [(fix.enum)="BC"];
MSG_TYPE_NETWORK_COUNTERPARTY_SYSTEM_STATUS_RESPONSE = 53   [(fix.enum)="BD"];
MSG_TYPE_NEW_ORDER_CROSS = 54                 [(fix.enum)="s"];
MSG_TYPE_NEW_ORDER_LIST = 55                  [(fix.enum)="E"];
MSG_TYPE_NEW_ORDER_MULTILEG = 56              [(fix.enum)="AB"];
MSG_TYPE_NEW_ORDER_SINGLE = 57                [(fix.enum)="D"];
MSG_TYPE_NEWS = 58                            [(fix.enum)="B"];
MSG_TYPE_ORDER_CANCEL_REJECT = 59             [(fix.enum)="9"];
MSG_TYPE_ORDER_CANCEL_REPLACE_REQUEST = 60    [(fix.enum)="G"];
MSG_TYPE_ORDER_CANCEL_REQUEST = 61            [(fix.enum)="F"];
MSG_TYPE_ORDER_MASS_ACTION_REPORT = 62                [(fix.enum)="BZ"];
MSG_TYPE_ORDER_MASS_ACTION_REQUEST = 63               [(fix.enum)="CA"];
MSG_TYPE_ORDER_MASS_CANCEL_REPORT = 64                [(fix.enum)="r"];
MSG_TYPE_ORDER_MASS_CANCEL_REQUEST = 65               [(fix.enum)="q"];
MSG_TYPE_ORDER_MASS_STATUS_REQUEST = 66               [(fix.enum)="AF"];
MSG_TYPE_ORDER_STATUS_REQUEST = 67                    [(fix.enum)="H"];
MSG_TYPE_POSITION_MAINTENANCE_REPORT = 68             [(fix.enum)="AM"];
MSG_TYPE_POSITION_MAINTENANCE_REQUEST = 69            [(fix.enum)="AL"];
MSG_TYPE_POSITION_REPORT = 70                         [(fix.enum)="AP"];
MSG_TYPE_QUOTE = 71                           [(fix.enum)="S"];
MSG_TYPE_QUOTE_CANCEL = 72                    [(fix.enum)="Z"];
MSG_TYPE_QUOTE_REQUEST = 73                           [(fix.enum)="R"];
MSG_TYPE_QUOTE_REQUEST_REJECT = 74                    [(fix.enum)="AG"];
MSG_TYPE_QUOTE_RESPONSE = 75                          [(fix.enum)="AJ"];
MSG_TYPE_QUOTE_STATUS_REPORT = 76             [(fix.enum)="AI"];
MSG_TYPE_QUOTE_STATUS_REQUEST = 77                    [(fix.enum)="a"];
MSG_TYPE_RFQ_REQUEST = 78                     [(fix.enum)="AH"];
MSG_TYPE_REGISTRATION_INSTRUCTIONS = 79               [(fix.enum)="o"];
MSG_TYPE_REGISTRATION_INSTRUCTIONS_RESPONSE = 80      [(fix.enum)="p"];
MSG_TYPE_REJECT = 81                                  [(fix.enum)="3"];
```



```
        MSG_TYPE_REQUEST_FOR_POSITIONS = 82                  [(fix.enum)="AN"];
        MSG_TYPE_REQUEST_FOR_POSITIONS_ACK = 83              [(fix.enum)="AO"];
        MSG_TYPE_RESEND_REQUEST = 84                         [(fix.enum)="2"];
        MSG_TYPE_SECURITY_DEFINITION = 85              [(fix.enum)="d"];
        MSG_TYPE_SECURITY_DEFINITION_REQUEST = 86            [(fix.enum)="c"];
        MSG_TYPE_SECURITY_DEFINITION_UPDATE_REPORT = 87             [(fix.enum)="BP"];
        MSG_TYPE_SECURITY_LIST = 88                          [(fix.enum)="y"];
        MSG_TYPE_SECURITY_LIST_REQUEST = 89                  [(fix.enum)="x"];
        MSG_TYPE_SECURITY_LIST_UPDATE_REPORT = 90            [(fix.enum)="BK"];
        MSG_TYPE_SECURITY_STATUS = 91                        [(fix.enum)="f"];
        MSG_TYPE_SECURITY_STATUS_REQUEST = 92                [(fix.enum)="e"];
        MSG_TYPE_SECURITY_TYPE_REQUEST = 93                  [(fix.enum)="v"];
        MSG_TYPE_SECURITY_TYPES = 94                         [(fix.enum)="w"];
        MSG_TYPE_SEQUENCE_RESET = 95                         [(fix.enum)="4"];
        MSG_TYPE_SETTLEMENT_INSTRUCTION_REQUEST = 96         [(fix.enum)="AV"];
        MSG_TYPE_SETTLEMENT_INSTRUCTIONS = 97                [(fix.enum)="T"];
        MSG_TYPE_SETTLEMENT_OBLIGATION_REPORT = 98           [(fix.enum)="BQ"];
        MSG_TYPE_STREAM_ASSIGNMENT_REPORT = 99               [(fix.enum)="CD"];
        MSG_TYPE_STREAM_ASSIGNMENT_REPORT_ACK = 100          [(fix.enum)="CE"];
        MSG_TYPE_STREAM_ASSIGNMENT_REQUEST = 101             [(fix.enum)="CC"];
        MSG_TYPE_TEST_REQUEST = 102                          [(fix.enum)="1"];
        MSG_TYPE_TRADE_CAPTURE_REPORT = 103                  [(fix.enum)="AE"];
        MSG_TYPE_TRADE_CAPTURE_REPORT_ACK = 104              [(fix.enum)="AR"];
        MSG_TYPE_TRADE_CAPTURE_REPORT_REQUEST = 105          [(fix.enum)="AD"];
        MSG_TYPE_TRADE_CAPTURE_REPORT_REQUEST_ACK = 106             [(fix.enum)="AQ"];
        MSG_TYPE_TRADING_SESSION_LIST = 107                  [(fix.enum)="BJ"];
        MSG_TYPE_TRADING_SESSION_LIST_REQUEST = 108          [(fix.enum)="BI"];
        MSG_TYPE_TRADING_SESSION_LIST_UPDATE_REPORT = 109    [(fix.enum)="BS"];
        MSG_TYPE_TRADING_SESSION_STATUS = 110                [(fix.enum)="h"];
        MSG_TYPE_TRADING_SESSION_STATUS_REQUEST = 111        [(fix.enum)="g"];
        MSG_TYPE_USER_NOTIFICATION = 112               [(fix.enum)="CB"];
        MSG_TYPE_USER_REQUEST = 113                          [(fix.enum)="BE"];
        MSG_TYPE_USER_RESPONSE = 114                         [(fix.enum)="BF"];
        MSG_TYPE_XMLNON_FIX = 115                      [(fix.enum)="n"];
}

enum ApplVerIdEnum {
        APPL_VER_ID_FIX27 = 0                                [(fix.enum)="0"];
        APPL_VER_ID_FIX30 = 1                                [(fix.enum)="1"];
        APPL_VER_ID_FIX40 = 2                                [(fix.enum)="2"];
        APPL_VER_ID_FIX41 = 3                                [(fix.enum)="3"];
        APPL_VER_ID_FIX42 = 4                                [(fix.enum)="4"];
        APPL_VER_ID_FIX43 = 5                                [(fix.enum)="5"];
        APPL_VER_ID_FIX44 = 6                                [(fix.enum)="6"];
        APPL_VER_ID_FIX50 = 7                                [(fix.enum)="7"];
        APPL_VER_ID_FIX50SP1 = 8                       [(fix.enum)="8"];
        APPL_VER_ID_FIX50SP2 = 9                       [(fix.enum)="9"];
}

message StandardHeader {
        optional string deliver_to_comp_id = 1         [(fix.tag)=128, (fix.type)=STRING];
        optional string deliver_to_sub_id = 2          [(fix.tag)=129, (fix.type)=STRING];
        optional fixed32 msg_seq_num = 3               [(fix.tag)=34, (fix.type)=SEQ_NUM];
        optional string on_behalf_of_comp_id = 4       [(fix.tag)=115, (fix.type)=STRING];
        optional string on_behalf_of_sub_id = 5        [(fix.tag)=116, (fix.type)=STRING];
        optional sfixed64 orig_sending_time = 6 [(fix.tag)=122, (fix.type)=UTC_TIMESTAMP];
        optional bool poss_dup_flag = 7                [(fix.tag)=43, (fix.type)=BOOLEAN];
        optional bool poss_resend = 8                  [(fix.tag)=97, (fix.type)=BOOLEAN];
        optional bytes secure_data = 9                 [(fix.tag)=91, (fix.type)=DATA,
(fix.field_deprecated)=FIXT_1_1];
        optional fixed32 secure_data_len = 10   [(fix.tag)=90, (fix.type)=LENGTH,
(fix.field_deprecated)=FIXT_1_1];
        optional string sender_comp_id = 11     [(fix.tag)=49, (fix.type)=STRING];
```



```
	optional string sender_sub_id = 12	[(fix.tag)=50, (fix.type)=STRING];
	optional sfixed64 sending_time = 13	[(fix.tag)=52, (fix.type)=UTC_TIMESTAMP];
	optional string target_comp_id = 14	[(fix.tag)=56, (fix.type)=STRING];
	optional string target_sub_id = 15	[(fix.tag)=57, (fix.type)=STRING];
	optional string deliver_to_location_id = 16	[(fix.tag)=145, (fix.type)=STRING];
	optional string on_behalf_of_location_id = 17	[(fix.tag)=144, (fix.type)=STRING];
	optional string sender_location_id = 18	[(fix.tag)=142, (fix.type)=STRING];
	optional string target_location_id = 19	[(fix.tag)=143, (fix.type)=STRING];
	optional fixed32 last_msg_seq_num_processed = 20	[(fix.tag)=369,
(fix.type)=SEQ_NUM];
	optional string message_encoding = 21	[(fix.tag)=347,
(fix.type)=STRING];
	optional bytes xml_data = 22	[(fix.tag)=213, (fix.type)=DATA];
	optional fixed32 xml_data_len = 23	[(fix.tag)=212,
(fix.type)=LENGTH];
	optional ApplVerIdEnum appl_ver_id = 24	[(fix.tag)=1128, (fix.type)=STRING];
	optional string cstm_appl_ver_id = 25	[(fix.tag)=1129, (fix.type)=STRING];
	repeated HopGrp hop_grp = 26	[(fix.tag)=627];
	optional sfixed64 appl_ext_id = 27	[(fix.tag)=1156, (fix.type)=INT];
}

message StandardTrailer {
	optional bytes signature = 1	[(fix.tag)=89, (fix.type)=DATA,
(fix.field_deprecated)=FIXT_1_1];
	optional fixed32 signature_length = 2	[(fix.tag)=93, (fix.type)=LENGTH,
(fix.field_deprecated)=FIXT_1_1];
}


//
//	FIX Unified Repository mapping to Google Protocol Buffer
//
//	Copyright (c) FIX Protocol Ltd. All Rights Reserved.
//
//	File: common.proto
//
//	Category: Common
//

import "meta.proto";
import "fix.proto";

option java_outer_classname = "Common";
option java_package = "org.fixprotocol.components";
package Common;

message ApplicationSequenceControl {
	optional string appl_id = 1	[(fix.tag)=1180, (fix.type)=STRING];
	optional fixed32 appl_last_seq_num = 2	[(fix.tag)=1350, (fix.type)=SEQ_NUM];
	optional bool appl_resend_flag = 3	[(fix.tag)=1352, (fix.type)=BOOLEAN];
	optional fixed32 appl_seq_num = 4	[(fix.tag)=1181, (fix.type)=SEQ_NUM];
}

message SecAltIdGrp {
	optional string security_alt_id = 1	[(fix.tag)=455, (fix.type)=STRING];
	optional string security_alt_id_source = 2	[(fix.tag)=456, (fix.type)=STRING];
}

message SecurityXml {
	optional string security_xml = 1	[(fix.tag)=1185, (fix.type)=XML_DATA];
	optional fixed32 security_xml_len = 2	[(fix.tag)=1184, (fix.type)=LENGTH];
	optional string security_xml_schema = 3	[(fix.tag)=1186, (fix.type)=STRING];
}
```

```
enum EventTypeEnum {
       EVENT_TYPE_CALL = 0                              [(fix.enum)="2"];
       EVENT_TYPE_OTHER = 1                             [(fix.enum)="99"];
       EVENT_TYPE_PUT = 2                               [(fix.enum)="1"];
       EVENT_TYPE_SINKING_FUND_CALL = 3                 [(fix.enum)="4"];
       EVENT_TYPE_TENDER = 4                            [(fix.enum)="3"];
       EVENT_TYPE_ACTIVATION = 5                        [(fix.enum)="5"];
       EVENT_TYPE_INACTIVIATION = 6                     [(fix.enum)="6"];
       EVENT_TYPE_LAST_ELIGIBLE_TRADE_DATE = 7          [(fix.enum)="7"];
       EVENT_TYPE_FINAL_INVENTORY_DUE_DATE = 8          [(fix.enum)="16"];
       EVENT_TYPE_FIRST_DELIVERY_DATE = 9               [(fix.enum)="13"];
       EVENT_TYPE_FIRST_INTENT_DATE = 10                [(fix.enum)="17"];
       EVENT_TYPE_INITIAL_INVENTORY_DUE_DATE = 11       [(fix.enum)="15"];
       EVENT_TYPE_LAST_DELIVERY_DATE = 12               [(fix.enum)="14"];
       EVENT_TYPE_LAST_INTENT_DATE = 13                 [(fix.enum)="18"];
       EVENT_TYPE_POSITION_REMOVAL_DATE = 14            [(fix.enum)="19"];
       EVENT_TYPE_SWAP_END_DATE = 15                    [(fix.enum)="9"];
       EVENT_TYPE_SWAP_NEXT_ROLL_DATE = 16              [(fix.enum)="12"];
       EVENT_TYPE_SWAP_NEXT_START_DATE = 17             [(fix.enum)="11"];
       EVENT_TYPE_SWAP_ROLL_DATE = 18                   [(fix.enum)="10"];
       EVENT_TYPE_SWAP_START_DATE = 19                  [(fix.enum)="8"];
}

message EventTypeUnion {
       optional EventTypeEnum event_type = 1;
       optional sfixed64 event_type_sfixed64 = 2;
}

message EvntGrp {
       optional sfixed32 event_date = 1        [(fix.tag)=866, (fix.type)=LOCAL_MKT_DATE];
       optional sfixed64 event_px = 2          [(fix.tag)=867, (fix.type)=PRICE];
       optional sfixed32 event_px_exponent = 3;
       optional string event_text = 4          [(fix.tag)=868, (fix.type)=STRING];
       optional EventTypeUnion event_type = 5  [(fix.tag)=865, (fix.type)=INT];
       optional sfixed64 event_time = 6        [(fix.tag)=1145, (fix.type)=UTC_TIMESTAMP];
}

message InstrumentPtysSubGrp {
       optional string instrument_party_sub_id = 1    [(fix.tag)=1053, (fix.type)=STRING];
       optional sfixed64 instrument_party_sub_id_type = 2    [(fix.tag)=1054, (fix.type)=INT];
}

message InstrumentParties {
       optional string instrument_party_id = 1        [(fix.tag)=1019, (fix.type)=STRING];
       optional string instrument_party_id_source = 2 [(fix.tag)=1050, (fix.type)=CHAR];
       optional sfixed64 instrument_party_role = 3    [(fix.tag)=1051, (fix.type)=INT];
       repeated InstrumentPtysSubGrp instrument_ptys_sub_grp = 4   [(fix.tag)=1052];
}

message ComplexEventTimes {
       optional sfixed64 complex_event_end_time = 1         [(fix.tag)=1496,
(fix.type)=UTC_TIME_ONLY];
       optional sfixed64 complex_event_start_time = 2       [(fix.tag)=1495,
(fix.type)=UTC_TIME_ONLY];
}

message ComplexEventDates {
       optional sfixed64 complex_event_end_date = 1         [(fix.tag)=1493,
(fix.type)=UTC_TIMESTAMP];
       optional sfixed64 complex_event_start_date = 2       [(fix.tag)=1492,
(fix.type)=UTC_TIMESTAMP];
       repeated ComplexEventTimes complex_event_times = 3   [(fix.tag)=1494];
```



```
}

enum ComplexEventTypeEnum {
       COMPLEX_EVENT_TYPE_CAPPED = 0                          [(fix.enum)="1"];
       COMPLEX_EVENT_TYPE_KNOCK_IN_UP = 1                     [(fix.enum)="3"];
       COMPLEX_EVENT_TYPE_KNOCK_OUT_DOWN = 2                  [(fix.enum)="6"];
       COMPLEX_EVENT_TYPE_KNOCK_OUT_UP = 3                    [(fix.enum)="5"];
       COMPLEX_EVENT_TYPE_KOCK_IN_DOWN = 4                    [(fix.enum)="4"];
       COMPLEX_EVENT_TYPE_RESET_BARRIER = 5                   [(fix.enum)="8"];
       COMPLEX_EVENT_TYPE_ROLLING_BARRIER = 6                 [(fix.enum)="9"];
       COMPLEX_EVENT_TYPE_TRIGGER = 7                         [(fix.enum)="2"];
       COMPLEX_EVENT_TYPE_UNDERLYING = 8               [(fix.enum)="7"];
}

enum ComplexEventPriceBoundaryMethodEnum {
       COMPLEX_EVENT_PRICE_BOUNDARY_METHOD_EQUAL_TO_COMPLEX_EVENT_PRICE = 0
       [(fix.enum)="3"];
       COMPLEX_EVENT_PRICE_BOUNDARY_METHOD_GREATER_THAN_COMPLEX_EVENT_PRICE = 1
       [(fix.enum)="5"];
       COMPLEX_EVENT_PRICE_BOUNDARY_METHOD_GREATER_THAN_OR_EQUAL_TO_COMPLEX_EVENT_PRICE = 2
       [(fix.enum)="4"];
       COMPLEX_EVENT_PRICE_BOUNDARY_METHOD_LESS_THAN_COMPLEX_EVENT_PRICE = 3
       [(fix.enum)="1"];
       COMPLEX_EVENT_PRICE_BOUNDARY_METHOD_LESS_THAN_OR_EQUAL_TO_COMPLEX_EVENT_PRICE = 4
       [(fix.enum)="2"];
}

enum ComplexEventPriceTimeTypeEnum {
       COMPLEX_EVENT_PRICE_TIME_TYPE_EXPIRATION = 0           [(fix.enum)="1"];
       COMPLEX_EVENT_PRICE_TIME_TYPE_IMMEDIATE = 1            [(fix.enum)="2"];
       COMPLEX_EVENT_PRICE_TIME_TYPE_SPECIFIED_DATE = 2       [(fix.enum)="3"];
}

enum ComplexEventConditionEnum {
       COMPLEX_EVENT_CONDITION_AND = 0                        [(fix.enum)="1"];
       COMPLEX_EVENT_CONDITION_OR = 1                         [(fix.enum)="2"];
}

message ComplexEvents {
       optional ComplexEventConditionEnum complex_event_condition = 1     [(fix.tag)=1490,
(fix.type)=INT];
       repeated ComplexEventDates complex_event_dates = 2    [(fix.tag)=1491];
       optional sfixed64 complex_event_price = 3              [(fix.tag)=1486,
(fix.type)=PRICE];
       optional sfixed32 complex_event_price_exponent = 4;
       optional ComplexEventPriceBoundaryMethodEnum complex_event_price_boundary_method = 5
       [(fix.tag)=1487, (fix.type)=INT];
       optional sfixed64 complex_event_price_boundary_precision = 6       [(fix.tag)=1488,
(fix.type)=PERCENTAGE];
       optional sfixed32 complex_event_price_boundary_precision_exponent = 7;
       optional ComplexEventPriceTimeTypeEnum complex_event_price_time_type = 8
       [(fix.tag)=1489, (fix.type)=INT];
       optional ComplexEventTypeEnum complex_event_type = 9  [(fix.tag)=1484, (fix.type)=INT];
       optional sfixed64 complex_opt_payout_amount = 10      [(fix.tag)=1485, (fix.type)=AMT];
       optional sfixed32 complex_opt_payout_amount_exponent = 11;
}

enum SymbolSfxEnum {
       SYMBOL_SFX_EUCP_WITH_LUMP_SUM_INTEREST = 0             [(fix.enum)="CD"];
       SYMBOL_SFX_WHEN_ISSUED = 1                      [(fix.enum)="WI"];
}

enum SecurityIdSourceEnum {
```



```
        SECURITY_ID_SOURCE_CUSIP = 0                            [(fix.enum)="1"];
        SECURITY_ID_SOURCE_QUIK = 1                             [(fix.enum)="3"];
        SECURITY_ID_SOURCE_SEDOL = 2                            [(fix.enum)="2"];
        SECURITY_ID_SOURCE_ISIN_NUMBER = 3                      [(fix.enum)="4"];
        SECURITY_ID_SOURCE_RIC_CODE = 4                         [(fix.enum)="5"];
        SECURITY_ID_SOURCE_ISO_COUNTRY_CODE = 5                 [(fix.enum)="7"];
        SECURITY_ID_SOURCE_ISO_CURRENCY_CODE = 6                [(fix.enum)="6"];
        SECURITY_ID_SOURCE_CONSOLIDATED_TAPE_ASSOCIATION = 7    [(fix.enum)="9"];
        SECURITY_ID_SOURCE_EXCHANGE_SYMBOL = 8                  [(fix.enum)="8"];
        SECURITY_ID_SOURCE_BELGIAN = 9                          [(fix.enum)="F"];
        SECURITY_ID_SOURCE_BLOOMBERG_SYMBOL = 10                [(fix.enum)="A"];
        SECURITY_ID_SOURCE_COMMON = 11                          [(fix.enum)="G"];
        SECURITY_ID_SOURCE_DUTCH = 12                           [(fix.enum)="C"];
        SECURITY_ID_SOURCE_SICOVAM = 13                         [(fix.enum)="E"];
        SECURITY_ID_SOURCE_VALOREN = 14                         [(fix.enum)="D"];
        SECURITY_ID_SOURCE_WERTPAPIER = 15                      [(fix.enum)="B"];
        SECURITY_ID_SOURCE_CLEARING_HOUSE = 16                  [(fix.enum)="H"];
        SECURITY_ID_SOURCE_ISDA_FPML_SPECIFICATION = 17             [(fix.enum)="I"];
        SECURITY_ID_SOURCE_OPTION_PRICE_REPORTING_AUTHORITY = 18    [(fix.enum)="J"];
        SECURITY_ID_SOURCE_LETTER_OF_CREDIT = 19                [(fix.enum)="L"];
        SECURITY_ID_SOURCE_ISDA_FPML_URL = 20                   [(fix.enum)="K"];
        SECURITY_ID_SOURCE_MARKETPLACE_ASSIGNED_IDENTIFIER = 21     [(fix.enum)="M"];
}

enum ProductEnum {
        PRODUCT_AGENCY = 0                                  [(fix.enum)="1"];
        PRODUCT_COMMODITY = 1                                       [(fix.enum)="2"];
        PRODUCT_CORPORATE = 2                                       [(fix.enum)="3"];
        PRODUCT_CURRENCY = 3                                        [(fix.enum)="4"];
        PRODUCT_EQUITY = 4                                  [(fix.enum)="5"];
        PRODUCT_GOVERNMENT = 5                                      [(fix.enum)="6"];
        PRODUCT_INDEX = 6                                   [(fix.enum)="7"];
        PRODUCT_LOAN = 7                                    [(fix.enum)="8"];
        PRODUCT_MONEYMARKET = 8                                     [(fix.enum)="9"];
        PRODUCT_MORTGAGE = 9                                        [(fix.enum)="10"];
        PRODUCT_MUNICIPAL = 10                                      [(fix.enum)="11"];
        PRODUCT_OTHER = 11                                  [(fix.enum)="12"];
        PRODUCT_FINANCING = 12                                      [(fix.enum)="13"];
}

enum SecurityTypeEnum {
        SECURITY_TYPE_BANKERS_ACCEPTANCE = 0                    [(fix.enum)="BA"];
        SECURITY_TYPE_CERTIFICATE_OF_DEPOSIT = 1                [(fix.enum)="CD"];
        SECURITY_TYPE_COLLATERALIZED_MORTGAGE_OBLIGATION = 2    [(fix.enum)="CMO"];
        SECURITY_TYPE_COMMERCIAL_PAPER = 3                      [(fix.enum)="CP"];
        SECURITY_TYPE_COMMON_STOCK = 4                          [(fix.enum)="CS"];
        SECURITY_TYPE_CORPORATE_BOND = 5                [(fix.enum)="CORP"];
        SECURITY_TYPE_CORPORATE_PRIVATE_PLACEMENT = 6           [(fix.enum)="CPP"];
        SECURITY_TYPE_FOREIGN_EXCHANGE_CONTRACT = 7             [(fix.enum)="FOR",
(fix.enum_deprecated)=FIX_5_0_SP_1];
        SECURITY_TYPE_FUTURE = 8                            [(fix.enum)="FUT"];
        SECURITY_TYPE_MISCELLANEOUS_PASS_THROUGH = 9            [(fix.enum)="MPT"];
        SECURITY_TYPE_MORTGAGE_INTEREST_ONLY = 10               [(fix.enum)="MIO"];
        SECURITY_TYPE_MORTGAGE_PRINCIPAL_ONLY = 11              [(fix.enum)="MPO"];
        SECURITY_TYPE_MORTGAGE_PRIVATE_PLACEMENT = 12           [(fix.enum)="MPP"];
        SECURITY_TYPE_MUTUAL_FUND = 13                          [(fix.enum)="MF"];
        SECURITY_TYPE_NO_SECURITY_TYPE = 14                     [(fix.enum)="NONE"];
        SECURITY_TYPE_OPTION = 15                           [(fix.enum)="OPT"];
        SECURITY_TYPE_PREFERRED_STOCK = 16                      [(fix.enum)="PS"];
        SECURITY_TYPE_TIME_DEPOSIT = 17                         [(fix.enum)="TD"];
        SECURITY_TYPE_US_TREASURY_BILL_OLD = 18                 [(fix.enum)="USTB",
(fix.enum_deprecated)=FIX_4_4];
        SECURITY_TYPE_WARRANT = 19                          [(fix.enum)="WAR"];
```



```
      SECURITY_TYPE_CONVERTIBLE_BOND = 20                  [(fix.enum)="CB"];
      SECURITY_TYPE_IOETTEMORTGAGE = 21              [(fix.enum)="IET"];
      SECURITY_TYPE_WILDCARD = 22                          [(fix.enum)="?"];
      SECURITY_TYPE_AMENDED = 23                     [(fix.enum)="AMENDED"];
      SECURITY_TYPE_ASSET_BACKED_SECURITIES = 24           [(fix.enum)="ABS"];
      SECURITY_TYPE_BANK_NOTES = 25                        [(fix.enum)="BN"];
      SECURITY_TYPE_BILL_OF_EXCHANGES = 26                 [(fix.enum)="BOX"];
      SECURITY_TYPE_BRADY_BOND = 27                        [(fix.enum)="BRADY"];
      SECURITY_TYPE_BRIDGE_LOAN = 28                       [(fix.enum)="BRIDGE"];
      SECURITY_TYPE_CALL_LOANS = 29                        [(fix.enum)="CL"];
      SECURITY_TYPE_CERTIFICATE_OF_OBLIGATION = 30         [(fix.enum)="COFO"];
      SECURITY_TYPE_CERTIFICATE_OF_PARTICIPATION = 31            [(fix.enum)="COFP"];
      SECURITY_TYPE_CORP = 32                              [(fix.enum)="CMBS"];
      SECURITY_TYPE_DEBTOR_IN_POSSESSION = 33              [(fix.enum)="DINP"];
      SECURITY_TYPE_DEFAULTED = 34                         [(fix.enum)="DEFLTED"];
      SECURITY_TYPE_DEPOSIT_NOTES = 35               [(fix.enum)="DN"];
      SECURITY_TYPE_DUAL_CURRENCY = 36               [(fix.enum)="DUAL"];
      SECURITY_TYPE_EXTENDED_COMM_NOTE = 37                [(fix.enum)="XCN"];
      SECURITY_TYPE_FEDERAL_AGENCY_COUPON = 38             [(fix.enum)="FAC"];
      SECURITY_TYPE_FEDERAL_AGENCY_DISCOUNT_NOTE = 39            [(fix.enum)="FADN"];
      SECURITY_TYPE_GENERAL_OBLIGATION_BONDS = 40          [(fix.enum)="GO"];
      SECURITY_TYPE_INDEXED_LINKED = 41              [(fix.enum)="XLINKD"];
      SECURITY_TYPE_INTEREST_STRIP_FROM_ANY_BOND_OR_NOTE = 42     [(fix.enum)="TINT"];
      SECURITY_TYPE_LETTER_OF_CREDIT = 43                  [(fix.enum)="LOFC"];
      SECURITY_TYPE_LIQUIDITY_NOTE = 44              [(fix.enum)="LQN"];
      SECURITY_TYPE_MANDATORY_TENDER = 45                  [(fix.enum)="MT"];
      SECURITY_TYPE_MATURED = 46                     [(fix.enum)="MATURED"];
      SECURITY_TYPE_MEDIUM_TERM_NOTES = 47                 [(fix.enum)="MTN"];
      SECURITY_TYPE_MORTGAGE_BACKED_SECURITIES = 48        [(fix.enum)="MBS"];
      SECURITY_TYPE_MULTILEG_INSTRUMENT = 49               [(fix.enum)="MLEG"];
      SECURITY_TYPE_OTHER_ANTICIPATION_NOTES = 50          [(fix.enum)="AN"];
      SECURITY_TYPE_OVERNIGHT = 51                         [(fix.enum)="ONITE"];
      SECURITY_TYPE_PLAZOS_FIJOS = 52                      [(fix.enum)="PZFJ"];
      SECURITY_TYPE_PRINCIPAL_STRIP_FROM_ANON_CALLABLE_BOND_OR_NOTE = 53
      [(fix.enum)="TPRN"];
      SECURITY_TYPE_PRINCIPAL_STRIP_OF_ACALLABLE_BOND_OR_NOTE = 54
      [(fix.enum)="TCAL"];
      SECURITY_TYPE_PRIVATE_EXPORT_FUNDING = 55            [(fix.enum)="PEF"];
      SECURITY_TYPE_PROMISSORY_NOTE = 56                   [(fix.enum)="PN"];
      SECURITY_TYPE_REPLACED = 57                          [(fix.enum)="REPLACD"];
      SECURITY_TYPE_RETIRED = 58                     [(fix.enum)="RETIRED"];
      SECURITY_TYPE_REVENUE_ANTICIPATION_NOTE = 59         [(fix.enum)="RAN"];
      SECURITY_TYPE_REVENUE_BONDS = 60               [(fix.enum)="REV"];
      SECURITY_TYPE_REVOLVER = 61                          [(fix.enum)="RVLVTRM"];
      SECURITY_TYPE_REVOLVER_LOAN = 62               [(fix.enum)="RVLV"];
      SECURITY_TYPE_SHORT_TERM_LOAN_NOTE = 63              [(fix.enum)="STN"];
      SECURITY_TYPE_SPECIAL_ASSESSMENT = 64                [(fix.enum)="SPCLA"];
      SECURITY_TYPE_SPECIAL_OBLIGATION = 65                [(fix.enum)="SPCLO"];
      SECURITY_TYPE_SPECIAL_TAX = 66                       [(fix.enum)="SPCLT"];
      SECURITY_TYPE_STRUCTURED_NOTES = 67                  [(fix.enum)="STRUCT"];
      SECURITY_TYPE_SWING_LINE_FACILITY = 68               [(fix.enum)="SWING"];
      SECURITY_TYPE_TAX_ALLOCATION = 69              [(fix.enum)="TAXA"];
      SECURITY_TYPE_TAX_ANTICIPATION_NOTE = 70             [(fix.enum)="TAN"];
      SECURITY_TYPE_TAX_EXEMPT_COMMERCIAL_PAPER = 71       [(fix.enum)="TECP"];
      SECURITY_TYPE_TAX_REVENUE_ANTICIPATION_NOTE = 72     [(fix.enum)="TRAN"];
      SECURITY_TYPE_TERM_LOAN = 73                         [(fix.enum)="TERM"];
      SECURITY_TYPE_TO_BE_ANNOUNCED = 74                   [(fix.enum)="TBA"];
      SECURITY_TYPE_TREASURY_INFLATION_PROTECTED_SECURITIES = 75  [(fix.enum)="TIPS"];
      SECURITY_TYPE_US_TREASURY_BOND = 76                  [(fix.enum)="TBOND"];
      SECURITY_TYPE_US_TREASURY_NOTE_OLD = 77              [(fix.enum)="UST",
(fix.enum_deprecated)=FIX_4_4];
      SECURITY_TYPE_VARIABLE_RATE_DEMAND_NOTE = 78         [(fix.enum)="VRDN"];
      SECURITY_TYPE_WITHDRAWN = 79                         [(fix.enum)="WITHDRN"];
```



```
        SECURITY_TYPE_YANKEE_CORPORATE_BOND = 80             [(fix.enum)="YANK"];
        SECURITY_TYPE_BUY_SELLBACK = 81                      [(fix.enum)="BUYSELL"];
        SECURITY_TYPE_EURO_CERTIFICATE_OF_DEPOSIT = 82       [(fix.enum)="EUCD"];
        SECURITY_TYPE_EURO_COMMERCIAL_PAPER = 83             [(fix.enum)="EUCP"];
        SECURITY_TYPE_EURO_CORPORATE_BOND = 84               [(fix.enum)="EUCORP"];
        SECURITY_TYPE_EURO_SOVEREIGNS = 85                   [(fix.enum)="EUSOV"];
        SECURITY_TYPE_EURO_SUPRANATIONAL_COUPONS = 86        [(fix.enum)="EUSUPRA"];
        SECURITY_TYPE_FORWARD = 87                    [(fix.enum)="FORWARD"];
        SECURITY_TYPE_PFANDBRIEFE = 88                       [(fix.enum)="PFAND"];
        SECURITY_TYPE_REPURCHASE = 89                        [(fix.enum)="REPO"];
        SECURITY_TYPE_SECURITIES_LOAN = 90                   [(fix.enum)="SECLOAN"];
        SECURITY_TYPE_SECURITIES_PLEDGE = 91                 [(fix.enum)="SECPLEDGE"];
        SECURITY_TYPE_USD_SUPRANATIONAL_COUPONS = 92         [(fix.enum)="SUPRA"];
        SECURITY_TYPE_US_TREASURY_BILL = 93                  [(fix.enum)="TBILL"];
        SECURITY_TYPE_US_TREASURY_NOTE = 94                  [(fix.enum)="TNOTE"];
        SECURITY_TYPE_YANKEE_CERTIFICATE_OF_DEPOSIT = 95     [(fix.enum)="YCD"];
        SECURITY_TYPE_OPTIONS_ON_FUTURES = 96                [(fix.enum)="OOF"];
        SECURITY_TYPE_OPTIONS_ON_PHYSICAL = 97               [(fix.enum)="OOP"];
        SECURITY_TYPE_CASH = 98                              [(fix.enum)="CASH"];
        SECURITY_TYPE_OPTIONS_ON_COMBO = 99                  [(fix.enum)="OOC"];
        SECURITY_TYPE_INTEREST_RATE_SWAP = 100               [(fix.enum)="IRS"];
        SECURITY_TYPE_BANK_DEPOSITORY_NOTE = 101             [(fix.enum)="BDN"];
        SECURITY_TYPE_CANADIAN_MONEY_MARKETS = 102           [(fix.enum)="CAMM"];
        SECURITY_TYPE_CANADIAN_MORTGAGE_BONDS = 103          [(fix.enum)="CMB"];
        SECURITY_TYPE_CANADIAN_PROVINCIAL_BONDS = 104        [(fix.enum)="PROV"];
        SECURITY_TYPE_CANADIAN_TREASURY_BILLS = 105          [(fix.enum)="CTB"];
        SECURITY_TYPE_CANADIAN_TREASURY_NOTES = 106          [(fix.enum)="CAN"];
        SECURITY_TYPE_CREDIT_DEFAULT_SWAP = 107              [(fix.enum)="CDS"];
        SECURITY_TYPE_EURO_CORPORATE_FLOATING_RATE_NOTES = 108      [(fix.enum)="EUFRN"];
        SECURITY_TYPE_SECURED_LIQUIDITY_NOTE = 109           [(fix.enum)="SLQN"];
        SECURITY_TYPE_TAXABLE_MUNICIPAL_CP = 110             [(fix.enum)="TMCP"];
        SECURITY_TYPE_TERM_LIQUIDITY_NOTE = 111              [(fix.enum)="TLQN"];
        SECURITY_TYPE_TREASURY_BILL = 112             [(fix.enum)="TB"];
        SECURITY_TYPE_US_CORPORATE_FLOATING_RATE_NOTES = 113 [(fix.enum)="FRN"];
        SECURITY_TYPE_FX_FORWARD = 114                       [(fix.enum)="FXFWD"];
        SECURITY_TYPE_FX_SPOT = 115                          [(fix.enum)="FXSPOT"];
        SECURITY_TYPE_FX_SWAP = 116                          [(fix.enum)="FXSWAP"];
        SECURITY_TYPE_NON_DELIVERABLE_FORWARD = 117          [(fix.enum)="FXNDF"];
}

enum SecurityStatusEnum {
        SECURITY_STATUS_ACTIVE = 0                    [(fix.enum)="1"];
        SECURITY_STATUS_INACTIVE = 1                         [(fix.enum)="2"];
}

enum RestructuringTypeEnum {
        RESTRUCTURING_TYPE_FULL_RESTRUCTURING = 0            [(fix.enum)="FR"];
        RESTRUCTURING_TYPE_MODIFIED_MOD_RESTRUCTURING = 1    [(fix.enum)="MM"];
        RESTRUCTURING_TYPE_MODIFIED_RESTRUCTURING = 2        [(fix.enum)="MR"];
        RESTRUCTURING_TYPE_NO_RESTRUCTURING_SPECIFIED = 3    [(fix.enum)="XR"];
}

enum SeniorityEnum {
        SENIORITY_SENIOR = 0                                 [(fix.enum)="SR"];
        SENIORITY_SENIOR_SECURED = 1                         [(fix.enum)="SD"];
        SENIORITY_SUBORDINATED = 2                    [(fix.enum)="SB"];
}

enum StrikePriceDeterminationMethodEnum {
        STRIKE_PRICE_DETERMINATION_METHOD_FIXED_STRIKE = 0   [(fix.enum)="1"];
        STRIKE_PRICE_DETERMINATION_METHOD_STRIKE_SET_AT_EXPIRATION = 1    [(fix.enum)="2"];
        STRIKE_PRICE_DETERMINATION_METHOD_STRIKE_SET_TO_AVERAGE_ACROSS_LIFE = 2
        [(fix.enum)="3"];
```



```
       STRIKE_PRICE_DETERMINATION_METHOD_STRIKE_SET_TO_OPTIMAL_VALUE = 3  [(fix.enum)="4"];
}

enum StrikePriceBoundaryMethodEnum {
       STRIKE_PRICE_BOUNDARY_METHOD_EQUAL = 0                [(fix.enum)="3"];
       STRIKE_PRICE_BOUNDARY_METHOD_GREATER_THAN = 1         [(fix.enum)="5"];
       STRIKE_PRICE_BOUNDARY_METHOD_GREATER_THAN_OR_EQUAL = 2      [(fix.enum)="4"];
       STRIKE_PRICE_BOUNDARY_METHOD_LESS_THAN = 3            [(fix.enum)="1"];
       STRIKE_PRICE_BOUNDARY_METHOD_LESS_THAN_OR_EQUAL = 4   [(fix.enum)="2"];
}

enum UnderlyingPriceDeterminationMethodEnum {
       UNDERLYING_PRICE_DETERMINATION_METHOD_OPTIMAL_VALUE = 0     [(fix.enum)="3"];
       UNDERLYING_PRICE_DETERMINATION_METHOD_REGULAR = 1     [(fix.enum)="1"];
       UNDERLYING_PRICE_DETERMINATION_METHOD_SPECIAL_REFERENCE = 2 [(fix.enum)="2"];
}

enum ContractMultiplierUnitEnum {
       CONTRACT_MULTIPLIER_UNIT_DAYS = 0              [(fix.enum)="2"];
       CONTRACT_MULTIPLIER_UNIT_HOURS = 1                    [(fix.enum)="1"];
       CONTRACT_MULTIPLIER_UNIT_SHARES = 2                   [(fix.enum)="0"];
}

enum FlowScheduleTypeEnum {
       FLOW_SCHEDULE_TYPE_NERC_CALENDAR_ALL_DAYS_IN_MONTH = 0      [(fix.enum)="2"];
       FLOW_SCHEDULE_TYPE_NERC_EASTERN_OFF_PEAK = 1          [(fix.enum)="0"];
       FLOW_SCHEDULE_TYPE_NERC_EASTERN_PEAK = 2              [(fix.enum)="3"];
       FLOW_SCHEDULE_TYPE_NERC_WESTERN_OFF_PEAK = 3          [(fix.enum)="1"];
       FLOW_SCHEDULE_TYPE_NERC_WESTERN_PEAK = 4              [(fix.enum)="4"];
}

enum UnitOfMeasureEnum {
       UNIT_OF_MEASURE_BARRELS = 0                           [(fix.enum)="Bbl"];
       UNIT_OF_MEASURE_BILLION_CUBIC_FEET = 1                [(fix.enum)="Bcf"];
       UNIT_OF_MEASURE_BUSHELS = 2                           [(fix.enum)="Bu"];
       UNIT_OF_MEASURE_GALLONS = 3                           [(fix.enum)="Gal"];
       UNIT_OF_MEASURE_MEGAWATT_HOURS = 4                    [(fix.enum)="MWh"];
       UNIT_OF_MEASURE_METRIC_TONS = 5                       [(fix.enum)="t"];
       UNIT_OF_MEASURE_MILLION_BARRELS = 6                   [(fix.enum)="MMbbl",
(fix.enum_deprecated)=FIX_5_0_SP_1];
       UNIT_OF_MEASURE_ONE_MILLION_BTU = 7                   [(fix.enum)="MMBtu"];
       UNIT_OF_MEASURE_POUNDS = 8                     [(fix.enum)="lbs"];
       UNIT_OF_MEASURE_TONS = 9                       [(fix.enum)="tn"];
       UNIT_OF_MEASURE_TROY_OUNCES = 10               [(fix.enum)="oz_tr"];
       UNIT_OF_MEASURE_USDOLLARS = 11                        [(fix.enum)="USD"];
       UNIT_OF_MEASURE_ALLOWANCES = 12                       [(fix.enum)="Alw"];
}

enum SettlMethodEnum {
       SETTL_METHOD_CASH_SETTLEMENT_REQUIRED = 0             [(fix.enum)="C"];
       SETTL_METHOD_PHYSICAL_SETTLEMENT_REQUIRED = 1         [(fix.enum)="P"];
}

enum ExerciseStyleEnum {
       EXERCISE_STYLE_AMERICAN = 0                           [(fix.enum)="1"];
       EXERCISE_STYLE_BERMUDA = 1                     [(fix.enum)="2"];
       EXERCISE_STYLE_EUROPEAN = 2                           [(fix.enum)="0"];
}

enum OptPayoutTypeEnum {
       OPT_PAYOUT_TYPE_BINARY = 0                     [(fix.enum)="3"];
       OPT_PAYOUT_TYPE_CAPPED = 1                     [(fix.enum)="2"];
       OPT_PAYOUT_TYPE_VANILLA = 2                           [(fix.enum)="1"];
```



```
}

enum PriceQuoteMethodEnum {
       PRICE_QUOTE_METHOD_INDEX = 0                          [(fix.enum)="INX"];
       PRICE_QUOTE_METHOD_INTEREST_RATE_INDEX = 1            [(fix.enum)="INT"];
       PRICE_QUOTE_METHOD_STANDARD = 2                       [(fix.enum)="STD"];
       PRICE_QUOTE_METHOD_PERCENT_OF_PAR = 3                 [(fix.enum)="PCTPAR"];
}

enum ValuationMethodEnum {
       VALUATION_METHOD_FUTURES_STYLE_MARK_TO_MARKET = 0     [(fix.enum)="FUT"];
       VALUATION_METHOD_FUTURES_STYLE_WITH_AN_ATTACHED_CASH_ADJUSTMENT = 1
       [(fix.enum)="FUTDA"];
       VALUATION_METHOD_PREMIUM_STYLE = 2                    [(fix.enum)="EQTY"];
       VALUATION_METHOD_CDS_IN_DELIVERY_USE_RECOVERY_RATE_TO_CALCULATE = 3
       [(fix.enum)="CDSD"];
       VALUATION_METHOD_CDS_STYLE_COLLATERALIZATION = 4      [(fix.enum)="CDS"];
}

enum ListMethodEnum {
       LIST_METHOD_PRE_LISTED_ONLY = 0                       [(fix.enum)="0"];
       LIST_METHOD_USER_REQUESTED = 1                        [(fix.enum)="1"];
}

enum PutOrCallEnum {
       PUT_OR_CALL_CALL = 0                                  [(fix.enum)="1"];
       PUT_OR_CALL_PUT = 1                                [(fix.enum)="0"];
}

enum TimeUnitEnum {
       TIME_UNIT_DAY = 0                                  [(fix.enum)="D"];
       TIME_UNIT_HOUR = 1                                 [(fix.enum)="H"];
       TIME_UNIT_MINUTE = 2                                  [(fix.enum)="Min"];
       TIME_UNIT_MONTH = 3                                [(fix.enum)="Mo"];
       TIME_UNIT_SECOND = 4                                  [(fix.enum)="S"];
       TIME_UNIT_WEEK = 5                                 [(fix.enum)="Wk"];
       TIME_UNIT_YEAR = 6                                 [(fix.enum)="Yr"];
}

enum CpProgramEnum {
       CP_PROGRAM_OTHER = 0                                  [(fix.enum)="99"];
       CP_PROGRAM_PROGRAM3A3 = 1                          [(fix.enum)="1"];
       CP_PROGRAM_PROGRAM42 = 2                           [(fix.enum)="2"];
}

message StrikePriceDeterminationMethodUnion {
       optional StrikePriceDeterminationMethodEnum strike_price_determination_method = 1;
       optional sfixed64 strike_price_determination_method_sfixed64 = 2;
}

message FlowScheduleTypeUnion {
       optional FlowScheduleTypeEnum flow_schedule_type = 1;
       optional sfixed64 flow_schedule_type_sfixed64 = 2;
}

message CpProgramUnion {
       optional CpProgramEnum cp_program = 1;
       optional sfixed64 cp_program_sfixed64 = 2;
}

message Instrument {
       optional string cfi_code = 1                          [(fix.tag)=461,
(fix.type)=STRING];
```



```
      optional double contract_multiplier = 2              [(fix.tag)=231,
(fix.type)=FLOAT];
      optional string country_of_issue = 3                 [(fix.tag)=470,
(fix.type)=COUNTRY];
      optional sfixed32 coupon_payment_date = 4            [(fix.tag)=224,
(fix.type)=LOCAL_MKT_DATE];
      optional sfixed64 coupon_rate = 5              [(fix.tag)=223, (fix.type)=PERCENTAGE];
      optional sfixed32 coupon_rate_exponent = 6;
      optional string credit_rating = 7              [(fix.tag)=255, (fix.type)=STRING];
      optional bytes encoded_issuer = 8              [(fix.tag)=349, (fix.type)=DATA];
      optional fixed32 encoded_issuer_len = 9              [(fix.tag)=348,
(fix.type)=LENGTH];
      optional bytes encoded_security_desc = 10            [(fix.tag)=351, (fix.type)=DATA];
      optional fixed32 encoded_security_desc_len = 11            [(fix.tag)=350,
(fix.type)=LENGTH];
      optional double factor = 12                          [(fix.tag)=228,
(fix.type)=FLOAT];
      optional string instr_registry = 13                  [(fix.tag)=543,
(fix.type)=STRING];
      optional sfixed32 issue_date = 14              [(fix.tag)=225,
(fix.type)=LOCAL_MKT_DATE];
      optional string issuer = 15                          [(fix.tag)=106,
(fix.type)=STRING];
      optional string locale_of_issue = 16                 [(fix.tag)=472,
(fix.type)=STRING];
      optional sfixed32 maturity_date = 17                 [(fix.tag)=541,
(fix.type)=LOCAL_MKT_DATE];
      optional fixed32 maturity_month_year = 18            [(fix.tag)=200,
(fix.type)=MONTH_YEAR];
      optional string opt_attribute = 19                   [(fix.tag)=206, (fix.type)=CHAR];
      optional ProductEnum product = 20              [(fix.tag)=460, (fix.type)=INT];
      optional sfixed32 redemption_date = 21               [(fix.tag)=240,
(fix.type)=LOCAL_MKT_DATE, (fix.field_deprecated)=FIX_4_4];
      optional string repo_collateral_security_type = 22   [(fix.tag)=239,
(fix.type)=STRING, (fix.field_deprecated)=FIX_4_4];
      optional sfixed64 repurchase_rate = 23               [(fix.tag)=227,
(fix.type)=PERCENTAGE, (fix.field_deprecated)=FIX_4_4];
      optional sfixed32 repurchase_rate_exponent = 24;
      optional sfixed64 repurchase_term = 25               [(fix.tag)=226, (fix.type)=INT,
(fix.field_deprecated)=FIX_4_4];
      optional string security_desc = 26                   [(fix.tag)=107,
(fix.type)=STRING];
      optional string security_exchange = 27               [(fix.tag)=207,
(fix.type)=EXCHANGE];
      optional string security_id = 28               [(fix.tag)=48, (fix.type)=STRING];
      optional SecurityIdSourceEnum security_id_source = 29[(fix.tag)=22,
(fix.type)=STRING];
      optional SecurityTypeEnum security_type = 30         [(fix.tag)=167,
(fix.type)=STRING];
      optional string state_or_province_of_issue = 31            [(fix.tag)=471,
(fix.type)=STRING];
      optional sfixed64 strike_price = 32                  [(fix.tag)=202,
(fix.type)=PRICE];
      optional sfixed32 strike_price_exponent = 33;
      optional string symbol = 34                          [(fix.tag)=55,
(fix.type)=STRING];
      optional SymbolSfxEnum symbol_sfx = 35               [(fix.tag)=65,
(fix.type)=STRING];
      optional CpProgramUnion cp_program = 36              [(fix.tag)=875, (fix.type)=INT];
      optional string cp_reg_type = 37               [(fix.tag)=876, (fix.type)=STRING];
      optional fixed32 contract_settl_month = 38           [(fix.tag)=667,
(fix.type)=MONTH_YEAR];
```



```
      optional sfixed32 dated_date = 39             [(fix.tag)=873,
(fix.type)=LOCAL_MKT_DATE];
      repeated EvntGrp evnt_grp = 40                       [(fix.tag)=864];
      optional string instrmt_assignment_method = 41       [(fix.tag)=1049,
(fix.type)=CHAR];
      repeated InstrumentParties instrument_parties = 42   [(fix.tag)=1018];
      optional sfixed32 interest_accrual_date = 43         [(fix.tag)=874,
(fix.type)=LOCAL_MKT_DATE];
      optional string maturity_time = 44                   [(fix.tag)=1079,
(fix.type)=TZ_TIME_ONLY];
      optional double min_price_increment = 45             [(fix.tag)=969,
(fix.type)=FLOAT];
      optional sfixed64 nt_position_limit = 46                   [(fix.tag)=971,
(fix.type)=INT];
      optional string pool = 47                     [(fix.tag)=691, (fix.type)=STRING];
      optional sfixed64 position_limit = 48                [(fix.tag)=970, (fix.type)=INT];
      optional PutOrCallEnum put_or_call = 49              [(fix.tag)=201, (fix.type)=INT];
      repeated SecAltIdGrp sec_alt_id_grp = 50             [(fix.tag)=454];
      optional SecurityStatusEnum security_status = 51     [(fix.tag)=965,
(fix.type)=STRING];
      optional string security_sub_type = 52               [(fix.tag)=762,
(fix.type)=STRING];
      optional string settle_on_open_flag = 53             [(fix.tag)=966,
(fix.type)=STRING];
      optional string strike_currency = 54                 [(fix.tag)=947,
(fix.type)=CURRENCY];
      optional double strike_multiplier = 55               [(fix.tag)=967,
(fix.type)=FLOAT];
      optional double strike_value = 56             [(fix.tag)=968, (fix.type)=FLOAT];
      optional TimeUnitEnum time_unit = 57                 [(fix.tag)=997,
(fix.type)=STRING];
      optional UnitOfMeasureEnum unit_of_measure = 58            [(fix.tag)=996,
(fix.type)=STRING];
      optional sfixed64 cap_price = 59              [(fix.tag)=1199, (fix.type)=PRICE];
      optional sfixed32 cap_price_exponent = 60;
      optional ExerciseStyleEnum exercise_style = 61       [(fix.tag)=1194, (fix.type)=INT];
      optional bool flex_product_eligibility_indicator = 62[(fix.tag)=1242,
(fix.type)=BOOLEAN];
      optional bool flexible_indicator = 63                [(fix.tag)=1244,
(fix.type)=BOOLEAN];
      optional sfixed64 floor_price = 64                   [(fix.tag)=1200,
(fix.type)=PRICE];
      optional sfixed32 floor_price_exponent = 65;
      optional ListMethodEnum list_method = 66             [(fix.tag)=1198, (fix.type)=INT];
      optional sfixed64 min_price_increment_amount = 67    [(fix.tag)=1146, (fix.type)=AMT];
      optional sfixed32 min_price_increment_amount_exponent = 68;
      optional sfixed64 opt_payout_amount = 69             [(fix.tag)=1195, (fix.type)=AMT];
      optional sfixed32 opt_payout_amount_exponent = 70;
      optional PriceQuoteMethodEnum price_quote_method = 71[(fix.tag)=1196,
(fix.type)=STRING];
      optional string price_unit_of_measure = 72           [(fix.tag)=1191,
(fix.type)=STRING];
      optional sfixed64 price_unit_of_measure_qty = 73     [(fix.tag)=1192, (fix.type)=QTY];
      optional sfixed32 price_unit_of_measure_qty_exponent = 74;
      optional string product_complex = 75                 [(fix.tag)=1227,
(fix.type)=STRING];
      optional string security_group = 76                  [(fix.tag)=1151,
(fix.type)=STRING];
      optional SecurityXml security_xml = 77               [(fix.field_added)=FIX_5_0];
      optional SettlMethodEnum settl_method = 78           [(fix.tag)=1193,
(fix.type)=CHAR];
      optional sfixed64 unit_of_measure_qty = 79           [(fix.tag)=1147, (fix.type)=QTY];
      optional sfixed32 unit_of_measure_qty_exponent = 80;
```



```
	optional ValuationMethodEnum valuation_method = 81    [(fix.tag)=1197,
(fix.type)=STRING];
	optional ContractMultiplierUnitEnum contract_multiplier_unit = 82  [(fix.tag)=1435,
(fix.type)=INT];
	optional FlowScheduleTypeUnion flow_schedule_type = 83      [(fix.tag)=1439,
(fix.type)=INT];
	optional sfixed64 attachment_point = 84              [(fix.tag)=1457,
(fix.type)=PERCENTAGE];
	optional sfixed32 attachment_point_exponent = 85;
	optional sfixed64 detachment_point = 86              [(fix.tag)=1458,
(fix.type)=PERCENTAGE];
	optional sfixed32 detachment_point_exponent = 87;
	optional sfixed64 notional_percentage_outstanding = 88      [(fix.tag)=1451,
(fix.type)=PERCENTAGE];
	optional sfixed32 notional_percentage_outstanding_exponent = 89;
	optional sfixed64 original_notional_percentage_outstanding = 90    [(fix.tag)=1452,
(fix.type)=PERCENTAGE];
	optional sfixed32 original_notional_percentage_outstanding_exponent = 91;
	optional RestructuringTypeEnum restructuring_type = 92      [(fix.tag)=1449,
(fix.type)=STRING];
	optional SeniorityEnum seniority = 93                [(fix.tag)=1450,
(fix.type)=STRING];
	repeated ComplexEvents complex_events = 94           [(fix.tag)=1483];
	optional OptPayoutTypeEnum opt_payout_type = 95             [(fix.tag)=1482,
(fix.type)=INT];
	optional StrikePriceBoundaryMethodEnum strike_price_boundary_method = 96
	[(fix.tag)=1479, (fix.type)=INT];
	optional sfixed64 strike_price_boundary_precision = 97      [(fix.tag)=1480,
(fix.type)=PERCENTAGE];
	optional sfixed32 strike_price_boundary_precision_exponent = 98;
	optional StrikePriceDeterminationMethodUnion strike_price_determination_method = 99
	[(fix.tag)=1478, (fix.type)=INT];
	optional UnderlyingPriceDeterminationMethodEnum underlying_price_determination_method
= 100  [(fix.tag)=1481, (fix.type)=INT];
}

enum PartySubIdTypeEnum {
	PARTY_SUB_ID_TYPE_APPLICATION = 0             [(fix.enum)="4"];
	PARTY_SUB_ID_TYPE_BIC = 1                     [(fix.enum)="16"];
	PARTY_SUB_ID_TYPE_CSDPARTICIPANT_MEMBER_CODE = 2      [(fix.enum)="17"];
	PARTY_SUB_ID_TYPE_CASH_ACCOUNT_NAME = 3               [(fix.enum)="23"];
	PARTY_SUB_ID_TYPE_CASH_ACCOUNT_NUMBER = 4             [(fix.enum)="15"];
	PARTY_SUB_ID_TYPE_CONTACT_NAME = 5                    [(fix.enum)="9"];
	PARTY_SUB_ID_TYPE_DEPARTMENT = 6              [(fix.enum)="24"];
	PARTY_SUB_ID_TYPE_EMAIL_ADDRESS = 7                   [(fix.enum)="8"];
	PARTY_SUB_ID_TYPE_FAX_NUMBER = 8              [(fix.enum)="21"];
	PARTY_SUB_ID_TYPE_FIRM = 9                    [(fix.enum)="1"];
	PARTY_SUB_ID_TYPE_FULL_LEGAL_NAME_OF_FIRM = 10        [(fix.enum)="5"];
	PARTY_SUB_ID_TYPE_FUND_ACCOUNT_NAME = 11              [(fix.enum)="19"];
	PARTY_SUB_ID_TYPE_LOCATION_DESK = 12                  [(fix.enum)="25"];
	PARTY_SUB_ID_TYPE_PERSON = 13                         [(fix.enum)="2"];
	PARTY_SUB_ID_TYPE_PHONE_NUMBER = 14                   [(fix.enum)="7"];
	PARTY_SUB_ID_TYPE_POSITION_ACCOUNT_TYPE = 15          [(fix.enum)="26"];
	PARTY_SUB_ID_TYPE_POSTAL_ADDRESS = 16                 [(fix.enum)="6"];
	PARTY_SUB_ID_TYPE_REGISTERED_ADDRESS = 17             [(fix.enum)="18"];
	PARTY_SUB_ID_TYPE_REGISTERED_ADDRESS_FOR_CONFIRMATION = 18  [(fix.enum)="12"];
	PARTY_SUB_ID_TYPE_REGISTRATION_NAME = 19              [(fix.enum)="14"];
	PARTY_SUB_ID_TYPE_REGISTRATION_NUMBER = 20            [(fix.enum)="11"];
	PARTY_SUB_ID_TYPE_REGULATORY_STATUS = 21              [(fix.enum)="13"];
	PARTY_SUB_ID_TYPE_SECURITIES_ACCOUNT_NAME = 22        [(fix.enum)="22"];
	PARTY_SUB_ID_TYPE_SECURITIES_ACCOUNT_NUMBER = 23      [(fix.enum)="10"];
	PARTY_SUB_ID_TYPE_SYSTEM = 24                         [(fix.enum)="3"];
	PARTY_SUB_ID_TYPE_TELEX_NUMBER = 25                   [(fix.enum)="20"];
```



```
        PARTY_SUB_ID_TYPE_SECURITY_LOCATE_ID = 26            [(fix.enum)="27"];
        PARTY_SUB_ID_TYPE_ELIGIBLE_COUNTERPARTY = 27         [(fix.enum)="29"];
        PARTY_SUB_ID_TYPE_EXECUTION_VENUE = 28               [(fix.enum)="32"];
        PARTY_SUB_ID_TYPE_LOCATION = 29                      [(fix.enum)="31"];
        PARTY_SUB_ID_TYPE_MARKET_MAKER = 30                  [(fix.enum)="28"];
        PARTY_SUB_ID_TYPE_PROFESSIONAL_CLIENT = 31           [(fix.enum)="30"];
        PARTY_SUB_ID_TYPE_CURRENCY_DELIVERY_IDENTIFIER = 32  [(fix.enum)="33"];
}

message PartySubIdTypeUnion {
        optional PartySubIdTypeEnum party_sub_id_type = 1;
        optional sfixed64 party_sub_id_type_sfixed64 = 2;
}

message PtysSubGrp {
        optional string party_sub_id = 1               [(fix.tag)=523, (fix.type)=STRING];
        optional PartySubIdTypeUnion party_sub_id_type = 2   [(fix.tag)=803, (fix.type)=INT];
}

enum PartyIdSourceEnum {
        PARTY_ID_SOURCE_AUSTRALIAN_BUSINESS_NUMBER = 0       [(fix.enum)="9"];
        PARTY_ID_SOURCE_AUSTRALIAN_TAX_FILE_NUMBER = 1       [(fix.enum)="A"];
        PARTY_ID_SOURCE_BIC = 2                              [(fix.enum)="B"];
        PARTY_ID_SOURCE_CHINESE_INVESTOR_ID = 3              [(fix.enum)="5"];
        PARTY_ID_SOURCE_GENERAL_IDENTIFIER = 4               [(fix.enum)="C"];
        PARTY_ID_SOURCE_ISO_COUNTRY_CODE = 5                 [(fix.enum)="E"];
        PARTY_ID_SOURCE_KOREAN_INVESTOR_ID = 6               [(fix.enum)="1"];
        PARTY_ID_SOURCE_MALAYSIAN_CENTRAL_DEPOSITORY = 7     [(fix.enum)="4"];
        PARTY_ID_SOURCE_PROPRIETARY = 8                      [(fix.enum)="D"];
        PARTY_ID_SOURCE_SETTLEMENT_ENTITY_LOCATION = 9       [(fix.enum)="F"];
        PARTY_ID_SOURCE_TAIWANESE_FOREIGN_INVESTOR_ID = 10   [(fix.enum)="2"];
        PARTY_ID_SOURCE_TAIWANESE_TRADING_ACCT = 11          [(fix.enum)="3"];
        PARTY_ID_SOURCE_UK_NATIONAL_INSURANCE_OR_PENSION_NUMBER = 12      [(fix.enum)="6"];
        PARTY_ID_SOURCE_US_EMPLOYER_OR_TAX_ID_NUMBER = 13    [(fix.enum)="8"];
        PARTY_ID_SOURCE_US_SOCIAL_SECURITY_NUMBER = 14       [(fix.enum)="7"];
        PARTY_ID_SOURCE_CSDPARTICIPANT = 15                  [(fix.enum)="H"];
        PARTY_ID_SOURCE_ISITCACRONYM = 16             [(fix.enum)="I"];
        PARTY_ID_SOURCE_MIC = 17                      [(fix.enum)="G"];
}

enum PartyRoleEnum {
        PARTY_ROLE_BROKER_OF_CREDIT = 0                      [(fix.enum)="2"];
        PARTY_ROLE_CLEARING_FIRM = 1                         [(fix.enum)="4"];
        PARTY_ROLE_CLIENT_ID = 2                      [(fix.enum)="3"];
        PARTY_ROLE_CONTRA_CLEARING_FIRM = 3                  [(fix.enum)="18"];
        PARTY_ROLE_CONTRA_FIRM = 4                    [(fix.enum)="17"];
        PARTY_ROLE_CORRESPONDANT_CLEARING_FIRM = 5           [(fix.enum)="15"];
        PARTY_ROLE_ENTERING_FIRM = 6                         [(fix.enum)="7"];
        PARTY_ROLE_EXECUTING_FIRM = 7                        [(fix.enum)="1"];
        PARTY_ROLE_EXECUTING_SYSTEM = 8                      [(fix.enum)="16"];
        PARTY_ROLE_EXECUTING_TRADER = 9                      [(fix.enum)="12"];
        PARTY_ROLE_FUND_MANAGER_CLIENT_ID = 10               [(fix.enum)="9"];
        PARTY_ROLE_GIVEUP_CLEARING_FIRM = 11                 [(fix.enum)="14"];
        PARTY_ROLE_INTRODUCING_FIRM = 12              [(fix.enum)="6"];
        PARTY_ROLE_INVESTOR_ID = 13                          [(fix.enum)="5"];
        PARTY_ROLE_LOCATE = 14                               [(fix.enum)="8"];
        PARTY_ROLE_ORDER_ORIGINATION_FIRM = 15               [(fix.enum)="13"];
        PARTY_ROLE_ORDER_ORIGINATION_TRADER = 16             [(fix.enum)="11"];
        PARTY_ROLE_SETTLEMENT_LOCATION = 17                  [(fix.enum)="10"];
        PARTY_ROLE_SPONSORING_FIRM = 18                      [(fix.enum)="19"];
        PARTY_ROLE_UNDERLYING_CONTRA_FIRM = 19               [(fix.enum)="20"];
        PARTY_ROLE_CONTRA_INVESTOR_ID = 20                   [(fix.enum)="39"];
        PARTY_ROLE_TRANSFER_TO_FIRM = 21              [(fix.enum)="40"];
```



```
        PARTY_ROLE_AGENT = 22                                   [(fix.enum)="30"];
        PARTY_ROLE_BENEFICIARY = 23                             [(fix.enum)="32"];
        PARTY_ROLE_BUYER = 24                                   [(fix.enum)="27"];
        PARTY_ROLE_CLEARING_ORGANIZATION = 25                   [(fix.enum)="21"];
        PARTY_ROLE_CONTRA_TRADER = 26                           [(fix.enum)="37"];
        PARTY_ROLE_CORRESPONDENT_BROKER = 27                    [(fix.enum)="26"];
        PARTY_ROLE_CORRESPONDENT_CLEARING_ORGANIZATION = 28     [(fix.enum)="25"];
        PARTY_ROLE_CUSTODIAN = 29                           [(fix.enum)="28"];
        PARTY_ROLE_CUSTOMER_ACCOUNT = 30                    [(fix.enum)="24"];
        PARTY_ROLE_ENTERING_TRADER = 31                         [(fix.enum)="36"];
        PARTY_ROLE_EXCHANGE = 32                            [(fix.enum)="22"];
        PARTY_ROLE_INTERESTED_PARTY = 33                    [(fix.enum)="33"];
        PARTY_ROLE_INTERMEDIARY = 34                            [(fix.enum)="29"];
        PARTY_ROLE_LIQUIDITY_PROVIDER = 35                      [(fix.enum)="35"];
        PARTY_ROLE_POSITION_ACCOUNT = 36                    [(fix.enum)="38"];
        PARTY_ROLE_REGULATORY_BODY = 37                         [(fix.enum)="34"];
        PARTY_ROLE_SUB_CUSTODIAN = 38                           [(fix.enum)="31"];
        PARTY_ROLE_INTRODUCING_BROKER = 39                      [(fix.enum)="60"];
        PARTY_ROLE_CONTRA_EXCHANGE = 40                         [(fix.enum)="42"];
        PARTY_ROLE_CONTRA_POSITION_ACCOUNT = 41                 [(fix.enum)="41"];
        PARTY_ROLE_INTERNAL_CARRY_ACCOUNT = 42                  [(fix.enum)="43"];
        PARTY_ROLE_ORDER_ENTRY_OPERATOR_ID = 43                 [(fix.enum)="44"];
        PARTY_ROLE_SECONDARY_ACCOUNT_NUMBER = 44                [(fix.enum)="45"];
        PARTY_ROLE_ASSET_MANAGER = 45                           [(fix.enum)="49"];
        PARTY_ROLE_CLAIMING_ACCOUNT = 46                    [(fix.enum)="48"];
        PARTY_ROLE_FOREIGN_FIRM = 47                            [(fix.enum)="46"];
        PARTY_ROLE_LARGE_TRADER_REPORTABLE_ACCOUNT = 48                [(fix.enum)="52"];
        PARTY_ROLE_PLEDGEE_ACCOUNT = 49                         [(fix.enum)="51"];
        PARTY_ROLE_PLEDGOR_ACCOUNT = 50                         [(fix.enum)="50"];
        PARTY_ROLE_SENDER_LOCATION = 51                         [(fix.enum)="54"];
        PARTY_ROLE_SESSION_ID = 52                          [(fix.enum)="55"];
        PARTY_ROLE_THIRD_PARTY_ALLOCATION_FIRM = 53             [(fix.enum)="47"];
        PARTY_ROLE_TRADER_MNEMONIC = 54                         [(fix.enum)="53"];
        PARTY_ROLE_ACCEPTABLE_COUNTERPARTY = 55                 [(fix.enum)="56"];
        PARTY_ROLE_ENTERING_UNIT = 56                           [(fix.enum)="58"];
        PARTY_ROLE_EXECUTING_UNIT = 57                          [(fix.enum)="59"];
        PARTY_ROLE_UNACCEPTABLE_COUNTERPARTY = 58               [(fix.enum)="57"];
        PARTY_ROLE_COMPETENT_AUTHORITY_LIQUIDITY = 59           [(fix.enum)="70"];
        PARTY_ROLE_COMPETENT_AUTHORITY_TRANSACTION_VENUE = 60 [(fix.enum)="71"];
        PARTY_ROLE_DESK_ID = 61                                 [(fix.enum)="76"];
        PARTY_ROLE_EXECUTION_VENUE = 62                         [(fix.enum)="73"];
        PARTY_ROLE_HOME_COMPETENT_AUTHORITY = 63                [(fix.enum)="69"];
        PARTY_ROLE_HOST_COMPETENT_AUTHORITY = 64                [(fix.enum)="68"];
        PARTY_ROLE_INVESTMENT_FIRM = 65                         [(fix.enum)="67"];
        PARTY_ROLE_LOCATION_ID = 66                             [(fix.enum)="75"];
        PARTY_ROLE_MARKET_DATA_ENTRY_ORIGINATOR = 67            [(fix.enum)="74"];
        PARTY_ROLE_MARKET_DATA_MARKET = 68                      [(fix.enum)="77"];
        PARTY_ROLE_MARKET_MAKER = 69                            [(fix.enum)="66"];
        PARTY_ROLE_MULTILATERAL_TRADING_FACILITY = 70           [(fix.enum)="64"];
        PARTY_ROLE_QUOTE_ORIGINATOR = 71                    [(fix.enum)="61"];
        PARTY_ROLE_REGULATED_MARKET = 72                    [(fix.enum)="65"];
        PARTY_ROLE_REPORT_ORIGINATOR = 73                   [(fix.enum)="62"];
        PARTY_ROLE_REPORTING_INTERMEDIARY = 74                  [(fix.enum)="72"];
        PARTY_ROLE_SYSTEMATIC_INTERNALISER = 75                 [(fix.enum)="63"];
        PARTY_ROLE_ALLOCATION_ENTITY = 76                   [(fix.enum)="78"];
        PARTY_ROLE_BROKER_CLEARING_ID = 77                      [(fix.enum)="81"];
        PARTY_ROLE_PRIME_BROKER = 78                            [(fix.enum)="79"];
        PARTY_ROLE_STEP_OUT_FIRM = 79                           [(fix.enum)="80"];
        PARTY_ROLE_CENTRAL_REGISTRATION_DEPOSITORY = 80               [(fix.enum)="82"];
        PARTY_ROLE_ACCEPTABLE_SETTLING_COUNTERPARTY = 81        [(fix.enum)="84"];
        PARTY_ROLE_CLEARING_ACCOUNT = 82                    [(fix.enum)="83"];
        PARTY_ROLE_UNACCEPTABLE_SETTLING_COUNTERPARTY = 83      [(fix.enum)="85"];
}
```



```
message Parties {
       optional string party_id = 1                          [(fix.tag)=448,
(fix.type)=STRING];
       optional PartyIdSourceEnum party_id_source = 2        [(fix.tag)=447, (fix.type)=CHAR];
       optional PartyRoleEnum party_role = 3                 [(fix.tag)=452, (fix.type)=INT];
       repeated PtysSubGrp ptys_sub_grp = 4                  [(fix.tag)=802];
}

enum TerminationTypeEnum {
       TERMINATION_TYPE_FLEXIBLE = 0                         [(fix.enum)="3"];
       TERMINATION_TYPE_OPEN = 1                      [(fix.enum)="4"];
       TERMINATION_TYPE_OVERNIGHT = 2                        [(fix.enum)="1"];
       TERMINATION_TYPE_TERM = 3                      [(fix.enum)="2"];
}

enum DeliveryTypeEnum {
       DELIVERY_TYPE_FREE = 0                                [(fix.enum)="1"];
       DELIVERY_TYPE_HOLD_IN_CUSTODY = 1              [(fix.enum)="3"];
       DELIVERY_TYPE_TRI_PARTY = 2                           [(fix.enum)="2"];
       DELIVERY_TYPE_VERSUS_PAYMENT = 3               [(fix.enum)="0"];
}

message FinancingDetails {
       optional string agreement_currency = 1                [(fix.tag)=918,
(fix.type)=CURRENCY];
       optional sfixed32 agreement_date = 2                  [(fix.tag)=915,
(fix.type)=LOCAL_MKT_DATE];
       optional string agreement_desc = 3                    [(fix.tag)=913,
(fix.type)=STRING];
       optional string agreement_id = 4               [(fix.tag)=914, (fix.type)=STRING];
       optional DeliveryTypeEnum delivery_type = 5           [(fix.tag)=919, (fix.type)=INT];
       optional sfixed32 end_date = 6                        [(fix.tag)=917,
(fix.type)=LOCAL_MKT_DATE];
       optional sfixed64 margin_ratio = 7                    [(fix.tag)=898,
(fix.type)=PERCENTAGE];
       optional sfixed32 margin_ratio_exponent = 8;
       optional sfixed32 start_date = 9               [(fix.tag)=916,
(fix.type)=LOCAL_MKT_DATE];
       optional TerminationTypeEnum termination_type = 10    [(fix.tag)=788, (fix.type)=INT];
}

message UndSecAltIdGrp {
       optional string underlying_security_alt_id = 1        [(fix.tag)=458,
(fix.type)=STRING];
       optional string underlying_security_alt_id_source = 2 [(fix.tag)=459,
(fix.type)=STRING];
}

message UnderlyingStipulations {
       optional string underlying_stip_type = 1              [(fix.tag)=888,
(fix.type)=STRING];
       optional string underlying_stip_value = 2             [(fix.tag)=889,
(fix.type)=STRING];
}

message UndlyInstrumentPtysSubGrp {
       optional string underlying_instrument_party_sub_id = 1      [(fix.tag)=1063,
(fix.type)=STRING];
       optional sfixed64 underlying_instrument_party_sub_id_type = 2      [(fix.tag)=1064,
(fix.type)=INT];
}
```



```
message UndlyInstrumentParties {
	optional string underlying_instrument_party_id = 1	[(fix.tag)=1059,
(fix.type)=STRING];
	optional string underlying_instrument_party_id_source = 2	[(fix.tag)=1060,
(fix.type)=CHAR];
	optional sfixed64 underlying_instrument_party_role = 3	[(fix.tag)=1061,
(fix.type)=INT];
	repeated UndlyInstrumentPtysSubGrp undly_instrument_ptys_sub_grp = 4
	[(fix.tag)=1062];
}

enum UnderlyingSettlementTypeEnum {
	UNDERLYING_SETTLEMENT_TYPE_TPLUS1 = 0		[(fix.enum)="2"];
	UNDERLYING_SETTLEMENT_TYPE_TPLUS3 = 1		[(fix.enum)="4"];
	UNDERLYING_SETTLEMENT_TYPE_TPLUS4 = 2		[(fix.enum)="5"];
}

enum UnderlyingCashTypeEnum {
	UNDERLYING_CASH_TYPE_DIFF = 0			[(fix.enum)="DIFF"];
	UNDERLYING_CASH_TYPE_FIXED = 1			[(fix.enum)="FIXED"];
}

enum UnderlyingFxRateCalcEnum {
	UNDERLYING_FX_RATE_CALC_DIVIDE = 0		[(fix.enum)="D"];
	UNDERLYING_FX_RATE_CALC_MULTIPLY = 1		[(fix.enum)="M"];
}

message UnderlyingFlowScheduleTypeUnion {
	optional sfixed64 underlying_flow_schedule_type = 1; // No enums for this field were
found in source repository.
	optional sfixed64 underlying_flow_schedule_type_sfixed64 = 2;
}

message UnderlyingInstrument {
	optional bytes encoded_underlying_issuer = 1		[(fix.tag)=363, (fix.type)=DATA];
	optional fixed32 encoded_underlying_issuer_len = 2	[(fix.tag)=362,
(fix.type)=LENGTH];
	optional bytes encoded_underlying_security_desc = 3	[(fix.tag)=365, (fix.type)=DATA];
	optional fixed32 encoded_underlying_security_desc_len = 4	[(fix.tag)=364,
(fix.type)=LENGTH];
	optional string underlying_cfi_code = 5		[(fix.tag)=463,
(fix.type)=STRING];
	optional double underlying_contract_multiplier = 6	[(fix.tag)=436,
(fix.type)=FLOAT];
	optional string underlying_country_of_issue = 7		[(fix.tag)=592,
(fix.type)=COUNTRY];
	optional sfixed32 underlying_coupon_payment_date = 8	[(fix.tag)=241,
(fix.type)=LOCAL_MKT_DATE];
	optional sfixed64 underlying_coupon_rate = 9		[(fix.tag)=435,
(fix.type)=PERCENTAGE];
	optional sfixed32 underlying_coupon_rate_exponent = 10;
	optional string underlying_credit_rating = 11		[(fix.tag)=256,
(fix.type)=STRING];
	optional double underlying_factor = 12		[(fix.tag)=246,
(fix.type)=FLOAT];
	optional string underlying_instr_registry = 13		[(fix.tag)=595,
(fix.type)=STRING];
	optional sfixed32 underlying_issue_date = 14		[(fix.tag)=242,
(fix.type)=LOCAL_MKT_DATE];
	optional string underlying_issuer = 15		[(fix.tag)=306,
(fix.type)=STRING];
	optional string underlying_locale_of_issue = 16		[(fix.tag)=594,
(fix.type)=STRING];
```



```
      optional sfixed32 underlying_maturity_date = 17            [(fix.tag)=542,
(fix.type)=LOCAL_MKT_DATE];
      optional fixed32 underlying_maturity_month_year = 18 [(fix.tag)=313,
(fix.type)=MONTH_YEAR];
      optional string underlying_opt_attribute = 19        [(fix.tag)=317, (fix.type)=CHAR];
      optional sfixed64 underlying_product = 20                  [(fix.tag)=462,
(fix.type)=INT];
      optional sfixed64 underlying_put_or_call = 21        [(fix.tag)=315, (fix.type)=INT];
      optional sfixed32 underlying_redemption_date = 22    [(fix.tag)=247,
(fix.type)=LOCAL_MKT_DATE, (fix.field_deprecated)=FIX_4_4];
      optional string underlying_repo_collateral_security_type = 23     [(fix.tag)=243,
(fix.type)=STRING, (fix.field_deprecated)=FIX_4_4];
      optional sfixed64 underlying_repurchase_rate = 24    [(fix.tag)=245,
(fix.type)=PERCENTAGE, (fix.field_deprecated)=FIX_4_4];
      optional sfixed32 underlying_repurchase_rate_exponent = 25;
      optional sfixed64 underlying_repurchase_term = 26          [(fix.tag)=244,
(fix.type)=INT, (fix.field_deprecated)=FIX_4_4];
      optional string underlying_security_desc = 27        [(fix.tag)=307,
(fix.type)=STRING];
      optional string underlying_security_exchange = 28    [(fix.tag)=308,
(fix.type)=EXCHANGE];
      optional string underlying_security_id = 29          [(fix.tag)=309,
(fix.type)=STRING];
      optional string underlying_security_id_source = 30   [(fix.tag)=305,
(fix.type)=STRING];
      optional string underlying_security_type = 31        [(fix.tag)=310,
(fix.type)=STRING];
      optional string underlying_state_or_province_of_issue = 32  [(fix.tag)=593,
(fix.type)=STRING];
      optional sfixed64 underlying_strike_price = 33       [(fix.tag)=316,
(fix.type)=PRICE];
      optional sfixed32 underlying_strike_price_exponent = 34;
      optional string underlying_symbol = 35               [(fix.tag)=311,
(fix.type)=STRING];
      optional string underlying_symbol_sfx = 36           [(fix.tag)=312,
(fix.type)=STRING];
      repeated UndSecAltIdGrp und_sec_alt_id_grp = 37            [(fix.tag)=457];
      optional sfixed64 underlying_adjusted_quantity = 38  [(fix.tag)=1044, (fix.type)=QTY];
      optional sfixed32 underlying_adjusted_quantity_exponent = 39;
      optional sfixed64 underlying_allocation_percent = 40 [(fix.tag)=972,
(fix.type)=PERCENTAGE];
      optional sfixed32 underlying_allocation_percent_exponent = 41;
      optional string underlying_cp_program = 42           [(fix.tag)=877,
(fix.type)=STRING];
      optional string underlying_cp_reg_type = 43          [(fix.tag)=878,
(fix.type)=STRING];
      optional sfixed64 underlying_cap_value = 44          [(fix.tag)=1038, (fix.type)=AMT];
      optional sfixed32 underlying_cap_value_exponent = 45;
      optional sfixed64 underlying_cash_amount = 46        [(fix.tag)=973, (fix.type)=AMT];
      optional sfixed32 underlying_cash_amount_exponent = 47;
      optional UnderlyingCashTypeEnum underlying_cash_type = 48   [(fix.tag)=974,
(fix.type)=STRING];
      optional string underlying_currency = 49             [(fix.tag)=318,
(fix.type)=CURRENCY];
      optional sfixed64 underlying_current_value = 50            [(fix.tag)=885,
(fix.type)=AMT];
      optional sfixed32 underlying_current_value_exponent = 51;
      optional sfixed64 underlying_dirty_price = 52        [(fix.tag)=882,
(fix.type)=PRICE];
      optional sfixed32 underlying_dirty_price_exponent = 53;
      optional sfixed64 underlying_end_price = 54          [(fix.tag)=883,
(fix.type)=PRICE];
      optional sfixed32 underlying_end_price_exponent = 55;
```



```
      optional sfixed64 underlying_end_value = 56          [(fix.tag)=886, (fix.type)=AMT];
      optional sfixed32 underlying_end_value_exponent = 57;
      optional double underlying_fx_rate = 58              [(fix.tag)=1045,
(fix.type)=FLOAT];
      optional UnderlyingFxRateCalcEnum underlying_fx_rate_calc = 59     [(fix.tag)=1046,
(fix.type)=CHAR];
      optional sfixed64 underlying_px = 60                 [(fix.tag)=810,
(fix.type)=PRICE];
      optional sfixed32 underlying_px_exponent = 61;
      optional sfixed64 underlying_qty = 62                [(fix.tag)=879, (fix.type)=QTY];
      optional sfixed32 underlying_qty_exponent = 63;
      optional string underlying_security_sub_type = 64    [(fix.tag)=763,
(fix.type)=STRING];
      optional string underlying_settl_method = 65         [(fix.tag)=1039,
(fix.type)=STRING];
      optional UnderlyingSettlementTypeEnum underlying_settlement_type = 66
      [(fix.tag)=975, (fix.type)=INT];
      optional sfixed64 underlying_start_value = 67        [(fix.tag)=884, (fix.type)=AMT];
      optional sfixed32 underlying_start_value_exponent = 68;
      repeated UnderlyingStipulations underlying_stipulations = 69       [(fix.tag)=887];
      optional string underlying_strike_currency = 70             [(fix.tag)=941,
(fix.type)=CURRENCY];
      optional string underlying_time_unit = 71            [(fix.tag)=1000,
(fix.type)=STRING];
      optional string underlying_unit_of_measure = 72             [(fix.tag)=998,
(fix.type)=STRING];
      repeated UndlyInstrumentParties undly_instrument_parties = 73      [(fix.tag)=1058];
      optional sfixed64 underlying_exercise_style = 74            [(fix.tag)=1419,
(fix.type)=INT];
      optional string underlying_maturity_time = 75        [(fix.tag)=1213,
(fix.type)=TZ_TIME_ONLY];
      optional string underlying_price_unit_of_measure = 76[(fix.tag)=1424,
(fix.type)=STRING];
      optional sfixed64 underlying_price_unit_of_measure_qty = 77 [(fix.tag)=1425,
(fix.type)=QTY];
      optional sfixed32 underlying_price_unit_of_measure_qty_exponent = 78;
      optional sfixed64 underlying_unit_of_measure_qty = 79[(fix.tag)=1423, (fix.type)=QTY];
      optional sfixed32 underlying_unit_of_measure_qty_exponent = 80;
      optional sfixed64 underlying_contract_multiplier_unit = 81  [(fix.tag)=1437,
(fix.type)=INT];
      optional UnderlyingFlowScheduleTypeUnion underlying_flow_schedule_type = 82
      [(fix.tag)=1441, (fix.type)=INT];
      optional sfixed64 underlying_attachment_point = 83   [(fix.tag)=1459,
(fix.type)=PERCENTAGE];
      optional sfixed32 underlying_attachment_point_exponent = 84;
      optional sfixed64 underlying_detachment_point = 85   [(fix.tag)=1460,
(fix.type)=PERCENTAGE];
      optional sfixed32 underlying_detachment_point_exponent = 86;
      optional sfixed64 underlying_notional_percentage_outstanding = 87  [(fix.tag)=1455,
(fix.type)=PERCENTAGE];
      optional sfixed32 underlying_notional_percentage_outstanding_exponent = 88;
      optional sfixed64 underlying_original_notional_percentage_outstanding = 89
      [(fix.tag)=1456, (fix.type)=PERCENTAGE];
      optional sfixed32 underlying_original_notional_percentage_outstanding_exponent = 90;
      optional string underlying_restructuring_type = 91   [(fix.tag)=1453,
(fix.type)=STRING];
      optional string underlying_seniority = 92            [(fix.tag)=1454,
(fix.type)=STRING];
}

message UndInstrmtGrp {
      optional UnderlyingInstrument underlying_instrument = 1
      [(fix.field_added)=FIX_4_4];
```



```
}

enum RoundingDirectionEnum {
       ROUNDING_DIRECTION_ROUND_DOWN = 0              [(fix.enum)="1"];
       ROUNDING_DIRECTION_ROUND_TO_NEAREST = 1               [(fix.enum)="0"];
       ROUNDING_DIRECTION_ROUND_UP = 2                       [(fix.enum)="2"];
}

message OrderQtyData {
       optional sfixed64 cash_order_qty = 1                  [(fix.tag)=152, (fix.type)=QTY];
       optional sfixed32 cash_order_qty_exponent = 2;
       optional sfixed64 order_percent = 3                   [(fix.tag)=516,
(fix.type)=PERCENTAGE];
       optional sfixed32 order_percent_exponent = 4;
       optional sfixed64 order_qty = 5                       [(fix.tag)=38, (fix.type)=QTY];
       optional sfixed32 order_qty_exponent = 6;
       optional RoundingDirectionEnum rounding_direction = 7 [(fix.tag)=468, (fix.type)=CHAR];
       optional double rounding_modulus = 8                  [(fix.tag)=469,
(fix.type)=FLOAT];
}

enum StipulationTypeEnum {
       STIPULATION_TYPE_ABSOLUTE_PREPAYMENT_SPEED = 0        [(fix.enum)="ABS"];
       STIPULATION_TYPE_CONSTANT_PREPAYMENT_PENALTY = 1      [(fix.enum)="CPP"];
       STIPULATION_TYPE_CONSTANT_PREPAYMENT_RATE = 2         [(fix.enum)="CPR"];
       STIPULATION_TYPE_CONSTANT_PREPAYMENT_YIELD = 3        [(fix.enum)="CPY"];
       STIPULATION_TYPE_FINAL_CP_ROF_HOME_EQUITY_PREPAYMENT_CURVE = 4     [(fix.enum)="HEP"];
       STIPULATION_TYPE_GEOGRAPHICS = 5              [(fix.enum)="GEOG"];
       STIPULATION_TYPE_ISSUE_DATE = 6                       [(fix.enum)="ISSUE"];
       STIPULATION_TYPE_LOT_VARIANCE = 7             [(fix.enum)="LOTVAR"];
       STIPULATION_TYPE_MATURITY_YEAR_AND_MONTH = 8          [(fix.enum)="MAT"];
       STIPULATION_TYPE_MONTHLY_PREPAYMENT_RATE = 9          [(fix.enum)="MPR"];
       STIPULATION_TYPE_NUMBER_OF_PIECES = 10                [(fix.enum)="PIECES"];
       STIPULATION_TYPE_PERCENT_OF_BMAPREPAYMENT_CURVE = 11  [(fix.enum)="PSA"];
       STIPULATION_TYPE_PERCENT_OF_MANUFACTURED_HOUSING_PREPAYMENT_CURVE = 12
       [(fix.enum)="MHP"];
       STIPULATION_TYPE_PERCENT_OF_PROSPECTUS_PREPAYMENT_CURVE = 13       [(fix.enum)="PPC"];
       STIPULATION_TYPE_POOLS_MAXIMUM = 14                   [(fix.enum)="PMAX"];
       STIPULATION_TYPE_POOLS_PER_LOT = 15                   [(fix.enum)="PPL"];
       STIPULATION_TYPE_POOLS_PER_MILLION = 16               [(fix.enum)="PPM"];
       STIPULATION_TYPE_POOLS_PER_TRADE = 17                 [(fix.enum)="PPT"];
       STIPULATION_TYPE_PRODUCTION_YEAR = 18                 [(fix.enum)="PROD"];
       STIPULATION_TYPE_SINGLE_MONTHLY_MORTALITY = 19        [(fix.enum)="SMM"];
       STIPULATION_TYPE_TRADE_VARIANCE = 20                  [(fix.enum)="TRDVAR"];
       STIPULATION_TYPE_WEIGHTED_AVERAGE_COUPON = 21         [(fix.enum)="WAC"];
       STIPULATION_TYPE_WEIGHTED_AVERAGE_LIFE_COUPON = 22    [(fix.enum)="WAL"];
       STIPULATION_TYPE_WEIGHTED_AVERAGE_LOAN_AGE = 23             [(fix.enum)="WALA"];
       STIPULATION_TYPE_WEIGHTED_AVERAGE_MATURITY = 24             [(fix.enum)="WAM"];
       STIPULATION_TYPE_ALTERNATIVE_MINIMUM_TAX = 25         [(fix.enum)="AMT"];
       STIPULATION_TYPE_AUTO_REINVESTMENT = 26               [(fix.enum)="AUTOREINV"];
       STIPULATION_TYPE_BANK_QUALIFIED = 27                  [(fix.enum)="BANKQUAL"];
       STIPULATION_TYPE_BARGAIN_CONDITIONS = 28              [(fix.enum)="BGNCON"];
       STIPULATION_TYPE_BENCHMARK_PRICE_SOURCE = 29          [(fix.enum)="PXSOURCE"];
       STIPULATION_TYPE_CALL_PROTECTION = 30                 [(fix.enum)="PROTECT"];
       STIPULATION_TYPE_COUPON_RANGE = 31                    [(fix.enum)="COUPON"];
       STIPULATION_TYPE_CUSTOM_START = 32                    [(fix.enum)="CUSTOMDATE"];
       STIPULATION_TYPE_EXPLICIT_LOT_IDENTIFIER = 33         [(fix.enum)="LOT"];
       STIPULATION_TYPE_FREEFORM_TEXT = 34                   [(fix.enum)="TEXT"];
       STIPULATION_TYPE_ISO_CURRENCY_CODE = 35               [(fix.enum)="CURRENCY"];
       STIPULATION_TYPE_INSURED = 36                         [(fix.enum)="INSURED"];
       STIPULATION_TYPE_ISSUE_SIZE_RANGE = 37                [(fix.enum)="ISSUESIZE"];
       STIPULATION_TYPE_ISSUER = 38                          [(fix.enum)="ISSUER"];
       STIPULATION_TYPE_LOOKBACK_DAYS = 39                   [(fix.enum)="LOOKBACK"];
```



```
        STIPULATION_TYPE_MARKET_SECTOR = 40                  [(fix.enum)="SECTOR"];
        STIPULATION_TYPE_MATURITY_RANGE = 41                 [(fix.enum)="MATURITY"];
        STIPULATION_TYPE_MAXIMUM_SUBSTITUTIONS = 42          [(fix.enum)="MAXSUBS"];
        STIPULATION_TYPE_MINIMUM_DENOMINATION = 43           [(fix.enum)="MINDNOM"];
        STIPULATION_TYPE_MINIMUM_INCREMENT = 44              [(fix.enum)="MININCR"];
        STIPULATION_TYPE_MINIMUM_QUANTITY = 45               [(fix.enum)="MINQTY"];
        STIPULATION_TYPE_PAYMENT_FREQUENCY = 46              [(fix.enum)="PAYFREQ"];
        STIPULATION_TYPE_PRICE_RANGE = 47              [(fix.enum)="PRICE"];
        STIPULATION_TYPE_PRICING_FREQUENCY = 48              [(fix.enum)="PRICEFREQ"];
        STIPULATION_TYPE_PURPOSE = 49                        [(fix.enum)="PURPOSE"];
        STIPULATION_TYPE_RATING_SOURCE_AND_RANGE = 50        [(fix.enum)="RATING"];
        STIPULATION_TYPE_RESTRICTED = 51               [(fix.enum)="RESTRICTED"];
        STIPULATION_TYPE_SECURITY_TYPE_INCLUDED_OR_EXCLUDED = 52    [(fix.enum)="SECTYPE"];
        STIPULATION_TYPE_STRUCTURE = 53                      [(fix.enum)="STRUCT"];
        STIPULATION_TYPE_SUBSTITUTIONS_FREQUENCY = 54        [(fix.enum)="SUBSFREQ"];
        STIPULATION_TYPE_SUBSTITUTIONS_LEFT = 55             [(fix.enum)="SUBSLEFT"];
        STIPULATION_TYPE_TYPE_OF_REDEMPTION = 56             [(fix.enum)="REDEMPTION"];
        STIPULATION_TYPE_VALUATION_DISCOUNT = 57             [(fix.enum)="HAIRCUT"];
        STIPULATION_TYPE_WHOLE_POOL = 58               [(fix.enum)="WHOLE"];
        STIPULATION_TYPE_YIELD_RANGE = 59              [(fix.enum)="YIELD"];
        STIPULATION_TYPE_AVAILABLE_OFFER_QUANTITY_TO_BE_SHOWN_TO_THE_STREET = 60
        [(fix.enum)="AVAILQTY"];
        STIPULATION_TYPE_AVERAGE_FICOSCORE = 61              [(fix.enum)="AVFICO"];
        STIPULATION_TYPE_AVERAGE_LOAN_SIZE = 62              [(fix.enum)="AVSIZE"];
        STIPULATION_TYPE_BROKER_CREDIT = 63                  [(fix.enum)="BROKERCREDIT"];
        STIPULATION_TYPE_BROKER_SALES_CREDIT_OVERRIDE = 64   [(fix.enum)="SALESCREDITOVR"];
        STIPULATION_TYPE_DISCOUNT_RATE = 65                  [(fix.enum)="DISCOUNT"];
        STIPULATION_TYPE_INTEREST_OF_ROLLING_OR_CLOSING_TRADE = 66  [(fix.enum)="REFINT"];
        STIPULATION_TYPE_MAXIMUM_LOAN_BALANCE = 67           [(fix.enum)="MAXBAL"];
        STIPULATION_TYPE_MAXIMUM_ORDER_SIZE = 68             [(fix.enum)="MAXORDQTY"];
        STIPULATION_TYPE_OFFER_PRICE_TO_BE_SHOWN_TO_INTERNAL_BROKERS = 69
        [(fix.enum)="INTERNALPX"];
        STIPULATION_TYPE_OFFER_QUANTITY_TO_BE_SHOWN_TO_INTERNAL_BROKERS = 70
        [(fix.enum)="INTERNALQTY"];
        STIPULATION_TYPE_ORDER_QUANTITY_INCREMENT = 71       [(fix.enum)="ORDRINCR"];
        STIPULATION_TYPE_POOL_IDENTIFIER = 72                [(fix.enum)="POOL"];
        STIPULATION_TYPE_PRIMARY_OR_SECONDARY_MARKET_INDICATOR = 73 [(fix.enum)="PRIMARY"];
        STIPULATION_TYPE_PRINCIPAL_OF_ROLLING_OR_CLOSING_TRADE = 74 [(fix.enum)="REFPRIN"];
        STIPULATION_TYPE_REFERENCE_TO_ROLLING_OR_CLOSING_TRADE = 75 [(fix.enum)="REFTRADE"];
        STIPULATION_TYPE_THE_MINIMUM_RESIDUAL_OFFER_QUANTITY = 76   [(fix.enum)="LEAVEQTY"];
        STIPULATION_TYPE_TRADER_CREDIT = 77                  [(fix.enum)="TRADERCREDIT"];
        STIPULATION_TYPE_TYPE_OF_ROLL_TRADE = 78             [(fix.enum)="ROLLTYPE"];
        STIPULATION_TYPE_YIELD_TO_MATURITY = 79              [(fix.enum)="YTM"];
}

message Stipulations {
        optional StipulationTypeEnum stipulation_type = 1    [(fix.tag)=233,
(fix.type)=STRING];
        optional string stipulation_value = 2                [(fix.tag)=234,
(fix.type)=STRING];
}

message LegSecAltIdGrp {
        optional string leg_security_alt_id = 1              [(fix.tag)=605,
(fix.type)=STRING];
        optional string leg_security_alt_id_source = 2       [(fix.tag)=606,
(fix.type)=STRING];
}

message LegFlowScheduleTypeUnion {
        optional sfixed64 leg_flow_schedule_type = 1; // No enums for this field were found in
source repository.
        optional sfixed64 leg_flow_schedule_type_sfixed64 = 2;
```



```
}

message InstrumentLeg {
       optional bytes encoded_leg_issuer = 1                [(fix.tag)=619, (fix.type)=DATA];
       optional fixed32 encoded_leg_issuer_len = 2          [(fix.tag)=618,
(fix.type)=LENGTH];
       optional bytes encoded_leg_security_desc = 3         [(fix.tag)=622, (fix.type)=DATA];
       optional fixed32 encoded_leg_security_desc_len = 4   [(fix.tag)=621,
(fix.type)=LENGTH];
       optional string leg_cfi_code = 5               [(fix.tag)=608, (fix.type)=STRING];
       optional double leg_contract_multiplier = 6          [(fix.tag)=614,
(fix.type)=FLOAT];
       optional string leg_country_of_issue = 7             [(fix.tag)=596,
(fix.type)=COUNTRY];
       optional sfixed32 leg_coupon_payment_date = 8        [(fix.tag)=248,
(fix.type)=LOCAL_MKT_DATE];
       optional sfixed64 leg_coupon_rate = 9                [(fix.tag)=615,
(fix.type)=PERCENTAGE];
       optional sfixed32 leg_coupon_rate_exponent = 10;
       optional string leg_credit_rating = 11               [(fix.tag)=257,
(fix.type)=STRING];
       optional double leg_factor = 12                      [(fix.tag)=253,
(fix.type)=FLOAT];
       optional string leg_instr_registry = 13              [(fix.tag)=599,
(fix.type)=STRING];
       optional sfixed32 leg_issue_date = 14                [(fix.tag)=249,
(fix.type)=LOCAL_MKT_DATE];
       optional string leg_issuer = 15                      [(fix.tag)=617,
(fix.type)=STRING];
       optional string leg_locale_of_issue = 16             [(fix.tag)=598,
(fix.type)=STRING];
       optional sfixed32 leg_maturity_date = 17             [(fix.tag)=611,
(fix.type)=LOCAL_MKT_DATE];
       optional fixed32 leg_maturity_month_year = 18        [(fix.tag)=610,
(fix.type)=MONTH_YEAR];
       optional string leg_opt_attribute = 19               [(fix.tag)=613, (fix.type)=CHAR];
       optional sfixed64 leg_product = 20                   [(fix.tag)=607, (fix.type)=INT];
       optional double leg_ratio_qty = 21                   [(fix.tag)=623,
(fix.type)=FLOAT];
       optional sfixed32 leg_redemption_date = 22           [(fix.tag)=254,
(fix.type)=LOCAL_MKT_DATE, (fix.field_deprecated)=FIX_4_4];
       optional string leg_repo_collateral_security_type = 23      [(fix.tag)=250,
(fix.type)=STRING, (fix.field_deprecated)=FIX_4_4];
       optional sfixed64 leg_repurchase_rate = 24           [(fix.tag)=252,
(fix.type)=PERCENTAGE, (fix.field_deprecated)=FIX_4_4];
       optional sfixed32 leg_repurchase_rate_exponent = 25;
       optional sfixed64 leg_repurchase_term = 26           [(fix.tag)=251, (fix.type)=INT,
(fix.field_deprecated)=FIX_4_4];
       optional string leg_security_desc = 27               [(fix.tag)=620,
(fix.type)=STRING];
       optional string leg_security_exchange = 28           [(fix.tag)=616,
(fix.type)=EXCHANGE];
       optional string leg_security_id = 29                 [(fix.tag)=602,
(fix.type)=STRING];
       optional string leg_security_id_source = 30          [(fix.tag)=603,
(fix.type)=STRING];
       optional string leg_security_type = 31               [(fix.tag)=609,
(fix.type)=STRING];
       optional string leg_side = 32                        [(fix.tag)=624, (fix.type)=CHAR];
       optional string leg_state_or_province_of_issue = 33  [(fix.tag)=597,
(fix.type)=STRING];
       optional sfixed64 leg_strike_price = 34              [(fix.tag)=612,
(fix.type)=PRICE];
```



```
	optional sfixed32 leg_strike_price_exponent = 35;
	optional string leg_symbol = 36 [(fix.tag)=600,
(fix.type)=STRING];
	optional string leg_symbol_sfx = 37 [(fix.tag)=601,
(fix.type)=STRING];
	optional fixed32 leg_contract_settl_month = 38 [(fix.tag)=955,
(fix.type)=MONTH_YEAR];
	optional string leg_currency = 39 [(fix.tag)=556, (fix.type)=CURRENCY];
	optional sfixed32 leg_dated_date = 40 [(fix.tag)=739,
(fix.type)=LOCAL_MKT_DATE];
	optional sfixed32 leg_interest_accrual_date = 41 [(fix.tag)=956,
(fix.type)=LOCAL_MKT_DATE];
	optional double leg_option_ratio = 42 [(fix.tag)=1017,
(fix.type)=FLOAT];
	optional string leg_pool = 43 [(fix.tag)=740,
(fix.type)=STRING];
	optional sfixed64 leg_price = 44 [(fix.tag)=566, (fix.type)=PRICE];
	optional sfixed32 leg_price_exponent = 45;
	repeated LegSecAltIdGrp leg_sec_alt_id_grp = 46 [(fix.tag)=604];
	optional string leg_security_sub_type = 47 [(fix.tag)=764,
(fix.type)=STRING];
	optional string leg_strike_currency = 48 [(fix.tag)=942,
(fix.type)=CURRENCY];
	optional string leg_time_unit = 49 [(fix.tag)=1001,
(fix.type)=STRING];
	optional string leg_unit_of_measure = 50 [(fix.tag)=999,
(fix.type)=STRING];
	optional sfixed64 leg_exercise_style = 51 [(fix.tag)=1420,
(fix.type)=INT];
	optional string leg_maturity_time = 52 [(fix.tag)=1212,
(fix.type)=TZ_TIME_ONLY];
	optional string leg_price_unit_of_measure = 53 [(fix.tag)=1421,
(fix.type)=STRING];
	optional sfixed64 leg_price_unit_of_measure_qty = 54 [(fix.tag)=1422, (fix.type)=QTY];
	optional sfixed32 leg_price_unit_of_measure_qty_exponent = 55;
	optional sfixed64 leg_put_or_call = 56 [(fix.tag)=1358, (fix.type)=INT];
	optional sfixed64 leg_unit_of_measure_qty = 57 [(fix.tag)=1224, (fix.type)=QTY];
	optional sfixed32 leg_unit_of_measure_qty_exponent = 58;
	optional sfixed64 leg_contract_multiplier_unit = 59 [(fix.tag)=1436, (fix.type)=INT];
	optional LegFlowScheduleTypeUnion leg_flow_schedule_type = 60 [(fix.tag)=1440,
(fix.type)=INT];
}

message LegStipulations {
	optional string leg_stipulation_type = 1 [(fix.tag)=688,
(fix.type)=STRING];
	optional string leg_stipulation_value = 2 [(fix.tag)=689,
(fix.type)=STRING];
}

message InstrmtLegIoiGrp {
	optional InstrumentLeg instrument_leg = 1 [(fix.field_added)=FIX_4_4];
	optional string leg_ioi_qty = 2 [(fix.tag)=682,
(fix.type)=STRING];
	repeated LegStipulations leg_stipulations = 3 [(fix.tag)=683];
}

enum RoutingTypeEnum {
	ROUTING_TYPE_BLOCK_FIRM = 0 [(fix.enum)="3"];
	ROUTING_TYPE_BLOCK_LIST = 1 [(fix.enum)="4"];
	ROUTING_TYPE_TARGET_FIRM = 2 [(fix.enum)="1"];
	ROUTING_TYPE_TARGET_LIST = 3 [(fix.enum)="2"];
}
```



```
message RoutingGrp {
	optional string routing_id = 1 [(fix.tag)=217,
(fix.type)=STRING];
	optional RoutingTypeEnum routing_type = 2 [(fix.tag)=216, (fix.type)=INT];
}

enum BenchmarkCurveNameEnum {
	BENCHMARK_CURVE_NAME_EURIBOR = 0 [(fix.enum)="Euribor"];
	BENCHMARK_CURVE_NAME_FUTURE_SWAP = 1 [(fix.enum)="FutureSWAP"];
	BENCHMARK_CURVE_NAME_LIBID = 2 [(fix.enum)="LIBID"];
	BENCHMARK_CURVE_NAME_LIBOR = 3 [(fix.enum)="LIBOR"];
	BENCHMARK_CURVE_NAME_MUNI_AAA = 4 [(fix.enum)="MuniAAA"];
	BENCHMARK_CURVE_NAME_OTHER = 5 [(fix.enum)="OTHER"];
	BENCHMARK_CURVE_NAME_PFANDBRIEFE = 6 [(fix.enum)="Pfandbriefe"];
	BENCHMARK_CURVE_NAME_SWAP = 7 [(fix.enum)="SWAP"];
	BENCHMARK_CURVE_NAME_TREASURY = 8 [(fix.enum)="Treasury"];
	BENCHMARK_CURVE_NAME_EONIA = 9 [(fix.enum)="EONIA"];
	BENCHMARK_CURVE_NAME_EUREPO = 10 [(fix.enum)="EUREPO"];
	BENCHMARK_CURVE_NAME_SONIA = 11 [(fix.enum)="SONIA"];
}

message SpreadOrBenchmarkCurveData {
	optional string benchmark_curve_currency = 1 [(fix.tag)=220,
(fix.type)=CURRENCY];
	optional BenchmarkCurveNameEnum benchmark_curve_name = 2 [(fix.tag)=221,
(fix.type)=STRING];
	optional string benchmark_curve_point = 3 [(fix.tag)=222,
(fix.type)=STRING];
	optional sfixed64 spread = 4 [(fix.tag)=218,
(fix.type)=PRICE_OFFSET];
	optional sfixed32 spread_exponent = 5;
	optional sfixed64 benchmark_price = 6 [(fix.tag)=662,
(fix.type)=PRICE];
	optional sfixed32 benchmark_price_exponent = 7;
	optional sfixed64 benchmark_price_type = 8 [(fix.tag)=663, (fix.type)=INT];
	optional string benchmark_security_id = 9 [(fix.tag)=699,
(fix.type)=STRING];
	optional string benchmark_security_id_source = 10 [(fix.tag)=761,
(fix.type)=STRING];
}

enum YieldTypeEnum {
	YIELD_TYPE_AFTER_TAX_YIELD = 0 [(fix.enum)="AFTERTAX"];
	YIELD_TYPE_ANNUAL_YIELD = 1 [(fix.enum)="ANNUAL"];
	YIELD_TYPE_BOOK_YIELD = 2 [(fix.enum)="BOOK"];
	YIELD_TYPE_CLOSING_YIELD = 3 [(fix.enum)="CLOSE"];
	YIELD_TYPE_CLOSING_YIELD_MOST_RECENT_MONTH = 4 [(fix.enum)="LASTMONTH"];
	YIELD_TYPE_CLOSING_YIELD_MOST_RECENT_QUARTER = 5 [(fix.enum)="LASTQUARTER"];
	YIELD_TYPE_CLOSING_YIELD_MOST_RECENT_YEAR = 6 [(fix.enum)="LASTYEAR"];
	YIELD_TYPE_COMPOUND_YIELD = 7 [(fix.enum)="COMPOUND"];
	YIELD_TYPE_CURRENT_YIELD = 8 [(fix.enum)="CURRENT"];
	YIELD_TYPE_GVNT_EQUIVALENT_YIELD = 9 [(fix.enum)="GOVTEQUIV"];
	YIELD_TYPE_INVERSE_FLOATER_BOND_YIELD = 10 [(fix.enum)="INVERSEFLOATER"];
	YIELD_TYPE_MARK_TO_MARKET_YIELD = 11 [(fix.enum)="MARK"];
	YIELD_TYPE_MOST_RECENT_CLOSING_YIELD = 12 [(fix.enum)="LASTCLOSE"];
	YIELD_TYPE_OPEN_AVERAGE_YIELD = 13 [(fix.enum)="OPENAVG"];
	YIELD_TYPE_PREVIOUS_CLOSE_YIELD = 14 [(fix.enum)="PREVCLOSE"];
	YIELD_TYPE_PROCEEDS_YIELD = 15 [(fix.enum)="PROCEEDS"];
	YIELD_TYPE_SEMI_ANNUAL_YIELD = 16 [(fix.enum)="SEMIANNUAL"];
	YIELD_TYPE_SIMPLE_YIELD = 17 [(fix.enum)="SIMPLE"];
	YIELD_TYPE_TAX_EQUIVALENT_YIELD = 18 [(fix.enum)="TAXEQUIV"];
	YIELD_TYPE_TRUE_GROSS_YIELD = 19 [(fix.enum)="GROSS"];
```



```
	YIELD_TYPE_TRUE_YIELD = 20 [(fix.enum)="TRUE"];
	YIELD_TYPE_YIELD_AT_ISSUE = 21 [(fix.enum)="ATISSUE"];
	YIELD_TYPE_YIELD_CHANGE_SINCE_CLOSE = 22 [(fix.enum)="CHANGE"];
	YIELD_TYPE_YIELD_TO_AVERAGE_MATURITY = 23 [(fix.enum)="AVGMATURITY"];
	YIELD_TYPE_YIELD_TO_LONGEST_AVERAGE_LIFE = 24 [(fix.enum)="LONGAVGLIFE"];
	YIELD_TYPE_YIELD_TO_MATURITY = 25 [(fix.enum)="MATURITY"];
	YIELD_TYPE_YIELD_TO_NEXT_CALL = 26 [(fix.enum)="CALL"];
	YIELD_TYPE_YIELD_TO_NEXT_PUT = 27 [(fix.enum)="PUT"];
	YIELD_TYPE_YIELD_TO_NEXT_REFUND = 28 [(fix.enum)="NEXTREFUND"];
	YIELD_TYPE_YIELD_TO_SHORTEST_AVERAGE_LIFE = 29 [(fix.enum)="SHORTAVGLIFE"];
	YIELD_TYPE_YIELD_TO_TENDER_DATE = 30 [(fix.enum)="TENDER"];
	YIELD_TYPE_YIELD_TO_WORST = 31 [(fix.enum)="WORST"];
	YIELD_TYPE_YIELD_WITH_INFLATION_ASSUMPTION = 32 [(fix.enum)="INFLATION"];
	YIELD_TYPE_YIELD_VALUE_OF32NDS = 33 [(fix.enum)="VALUE1_32"];
}

message YieldData {
	optional sfixed64 yield = 1 [(fix.tag)=236,
(fix.type)=PERCENTAGE];
	optional sfixed32 yield_exponent = 2;
	optional YieldTypeEnum yield_type = 3 [(fix.tag)=235,
(fix.type)=STRING];
	optional sfixed32 yield_calc_date = 4 [(fix.tag)=701,
(fix.type)=LOCAL_MKT_DATE];
	optional sfixed32 yield_redemption_date = 5 [(fix.tag)=696,
(fix.type)=LOCAL_MKT_DATE];
	optional sfixed64 yield_redemption_price = 6 [(fix.tag)=697,
(fix.type)=PRICE];
	optional sfixed32 yield_redemption_price_exponent = 7;
	optional sfixed64 yield_redemption_price_type = 8 [(fix.tag)=698,
(fix.type)=INT];
}


//
//     FIX Unified Repository mapping to Google Protocol Buffer
//
//     Copyright (c) FIX Protocol Ltd. All Rights Reserved.
//
//     File: indication.proto
//
//     Category: Indication
//

import "meta.proto";
import "fix.proto";
import "session.proto";
import "common.proto";

option java_outer_classname = "Indication";
option java_package = "org.fixprotocol.components";
package Indication;

enum IoiQualifierEnum {
	IOI_QUALIFIER_ALL_OR_NONE = 0 [(fix.enum)="A"];
	IOI_QUALIFIER_AT_THE_CLOSE = 1 [(fix.enum)="C"];
	IOI_QUALIFIER_AT_THE_MARKET = 2 [(fix.enum)="Q"];
	IOI_QUALIFIER_AT_THE_OPEN = 3 [(fix.enum)="O"];
	IOI_QUALIFIER_CROSSING_OPPORTUNITY = 4 [(fix.enum)="X"];
	IOI_QUALIFIER_IN_TOUCH_WITH = 5 [(fix.enum)="I"];
	IOI_QUALIFIER_INDICATION = 6 [(fix.enum)="W"];
	IOI_QUALIFIER_LIMIT = 7 [(fix.enum)="L"];
	IOI_QUALIFIER_MORE_BEHIND = 8 [(fix.enum)="M"];
```

```
        IOI_QUALIFIER_PORTFOLIO_SHOWN = 9              [(fix.enum)="S"];
        IOI_QUALIFIER_TAKING_APOSITION = 10            [(fix.enum)="P"];
        IOI_QUALIFIER_THROUGH_THE_DAY = 11             [(fix.enum)="T"];
        IOI_QUALIFIER_VERSUS = 12                      [(fix.enum)="V"];
        IOI_QUALIFIER_AT_THE_MIDPOINT = 13             [(fix.enum)="Y"];
        IOI_QUALIFIER_PRE_OPEN = 14                    [(fix.enum)="Z"];
        IOI_QUALIFIER_READY_TO_TRADE = 15              [(fix.enum)="R"];
        IOI_QUALIFIER_MARKET_ON_CLOSE = 16             [(fix.enum)="B"];
        IOI_QUALIFIER_VWAP = 17                        [(fix.enum)="D"];
}

message IoiQualGrp {
        optional IoiQualifierEnum ioi_qualifier = 1          [(fix.tag)=104, (fix.type)=CHAR];
}

enum IoiTransTypeEnum {
        IOI_TRANS_TYPE_CANCEL = 0                      [(fix.enum)="C"];
        IOI_TRANS_TYPE_NEW = 1                         [(fix.enum)="N"];
        IOI_TRANS_TYPE_REPLACE = 2                     [(fix.enum)="R"];
}

enum SideEnum {
        SIDE_BUY = 0                                   [(fix.enum)="1"];
        SIDE_BUY_MINUS = 1                             [(fix.enum)="3"];
        SIDE_SELL = 2                                  [(fix.enum)="2"];
        SIDE_SELL_PLUS = 3                             [(fix.enum)="4"];
        SIDE_SELL_SHORT = 4                            [(fix.enum)="5"];
        SIDE_SELL_SHORT_EXEMPT = 5                     [(fix.enum)="6"];
        SIDE_CROSS = 6                                 [(fix.enum)="8"];
        SIDE_UNDISCLOSED = 7                           [(fix.enum)="7"];
        SIDE_CROSS_SHORT = 8                           [(fix.enum)="9"];
        SIDE_AS_DEFINED = 9                            [(fix.enum)="B"];
        SIDE_CROSS_SHORT_EXEMPT = 10                   [(fix.enum)="A"];
        SIDE_OPPOSITE = 11                             [(fix.enum)="C"];
        SIDE_BORROW = 12                               [(fix.enum)="G"];
        SIDE_LEND = 13                                 [(fix.enum)="F"];
        SIDE_REDEEM = 14                               [(fix.enum)="E"];
        SIDE_SUBSCRIBE = 15                            [(fix.enum)="D"];
}

enum QtyTypeEnum {
        QTY_TYPE_CONTRACTS = 0                         [(fix.enum)="1"];
        QTY_TYPE_UNITS = 1                             [(fix.enum)="0"];
        QTY_TYPE_UNITS_OF_MEASURE_PER_TIME_UNIT = 2    [(fix.enum)="2"];
}

enum IoiQtyEnum {
        IOI_QTY_LARGE = 0                              [(fix.enum)="L"];
        IOI_QTY_MEDIUM = 1                             [(fix.enum)="M"];
        IOI_QTY_SMALL = 2                              [(fix.enum)="S"];
        IOI_QTY_UNDISCLOSED_QUANTITY = 3               [(fix.enum)="U"];
}

enum PriceTypeEnum {
        PRICE_TYPE_FIXED_AMOUNT = 0                    [(fix.enum)="3"];
        PRICE_TYPE_PER_UNIT = 1                        [(fix.enum)="2"];
        PRICE_TYPE_PERCENTAGE = 2                      [(fix.enum)="1"];
        PRICE_TYPE_DISCOUNT = 3                        [(fix.enum)="4"];
        PRICE_TYPE_PREMIUM = 4                         [(fix.enum)="5"];
        PRICE_TYPE_SPREAD = 5                          [(fix.enum)="6"];
        PRICE_TYPE_TEDPRICE = 6                        [(fix.enum)="7"];
        PRICE_TYPE_TEDYIELD = 7                        [(fix.enum)="8"];
        PRICE_TYPE_FIXED_CABINET_TRADE_PRICE = 8       [(fix.enum)="10"];
```



```
        PRICE_TYPE_VARIABLE_CABINET_TRADE_PRICE = 9    [(fix.enum)="11"];
        PRICE_TYPE_YIELD = 10                          [(fix.enum)="9"];
        PRICE_TYPE_PRODUCT_TICKS_IN_EIGHTS = 11        [(fix.enum)="15"];
        PRICE_TYPE_PRODUCT_TICKS_IN_FOURTHS = 12       [(fix.enum)="14"];
        PRICE_TYPE_PRODUCT_TICKS_IN_HALFS = 13         [(fix.enum)="13"];
        PRICE_TYPE_PRODUCT_TICKS_IN_ONE_TWENTY_EIGHTS = 14    [(fix.enum)="19"];
        PRICE_TYPE_PRODUCT_TICKS_IN_SIXTEENTHS = 15    [(fix.enum)="16"];
        PRICE_TYPE_PRODUCT_TICKS_IN_SIXTY_FORTHS = 16  [(fix.enum)="18"];
        PRICE_TYPE_PRODUCT_TICKS_IN_THIRTY_SECONDS = 17             [(fix.enum)="17"];
}

enum IoiQltyIndEnum {
        IOI_QLTY_IND_HIGH = 0                          [(fix.enum)="H"];
        IOI_QLTY_IND_LOW = 1                           [(fix.enum)="L"];
        IOI_QLTY_IND_MEDIUM = 2                        [(fix.enum)="M"];
}

enum IoiNaturalFlagEnum {
        IOI_NATURAL_FLAG_NATURAL = 0                   [(fix.enum)="Y"];
        IOI_NATURAL_FLAG_NOT_NATURAL = 1               [(fix.enum)="N"];
}

message Ioi {
        option (fix.msg_type) = "6";
        optional string currency = 1                   [(fix.tag)=15, (fix.type)=CURRENCY];
        optional string ioi_id = 2                     [(fix.tag)=23, (fix.type)=STRING];
        optional IoiQltyIndEnum ioi_qlty_ind = 3       [(fix.tag)=25, (fix.type)=CHAR];
        optional IoiQtyEnum ioi_qty = 4                [(fix.tag)=27, (fix.type)=STRING];
        optional string ioi_ref_id = 5                 [(fix.tag)=26, (fix.type)=STRING];
        optional IoiTransTypeEnum ioi_trans_type = 6   [(fix.tag)=28, (fix.type)=CHAR];
        optional sfixed64 price = 7                    [(fix.tag)=44, (fix.type)=PRICE];
        optional sfixed32 price_exponent = 8;
        optional SideEnum side = 9                     [(fix.tag)=54, (fix.type)=CHAR];
        optional Session.StandardHeader standard_header = 10 [(fix.field_added)=FIX_2_7];
        optional Session.StandardTrailer standard_trailer = 11
        [(fix.field_added)=FIX_2_7];
        optional string text = 12                      [(fix.tag)=58, (fix.type)=STRING];
        optional sfixed64 valid_until_time = 13        [(fix.tag)=62,
(fix.type)=UTC_TIMESTAMP];
        optional IoiNaturalFlagEnum ioi_natural_flag = 14    [(fix.tag)=130,
(fix.type)=BOOLEAN];
        optional sfixed64 transact_time = 15           [(fix.tag)=60,
(fix.type)=UTC_TIMESTAMP];
        optional string urllink = 16                   [(fix.tag)=149, (fix.type)=STRING];
        optional bytes encoded_text = 17               [(fix.tag)=355, (fix.type)=DATA];
        optional fixed32 encoded_text_len = 18         [(fix.tag)=354, (fix.type)=LENGTH];
        optional Common.Instrument instrument = 19     [(fix.field_added)=FIX_4_3];
        optional PriceTypeEnum price_type = 20         [(fix.tag)=423, (fix.type)=INT];
        optional Common.SpreadOrBenchmarkCurveData spread_or_benchmark_curve_data = 21
        [(fix.field_added)=FIX_4_3];
        optional Common.FinancingDetails financing_details = 22
        [(fix.field_added)=FIX_4_4];
        repeated IoiQualGrp ioi_qual_grp = 23                [(fix.tag)=199];
        repeated Common.InstrmtLegIoiGrp instrmt_leg_ioi_grp = 24   [(fix.tag)=555];
        optional Common.OrderQtyData order_qty_data = 25     [(fix.field_added)=FIX_4_4];
        repeated Common.Parties parties = 26                 [(fix.tag)=453];
        optional QtyTypeEnum qty_type = 27                   [(fix.tag)=854, (fix.type)=INT];
        repeated Common.RoutingGrp routing_grp = 28          [(fix.tag)=215];
        repeated Common.Stipulations stipulations = 29       [(fix.tag)=232];
        repeated Common.UndInstrmtGrp und_instrmt_grp = 30   [(fix.tag)=711];
        optional Common.YieldData yield_data = 31            [(fix.field_added)=FIX_4_4];
        optional Common.ApplicationSequenceControl application_sequence_control = 32
        [(fix.field_added)=FIX_5_0];
```



```
}
```

