FIX PROTOCOL
INDUSTRY-DRIVEN MESSAGING STANDARD℠

FIX Repository Mapping to Google Protocol Buffers v0.9

RELEASE CANDIDATE 1: MAY 28, 2013

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.

High Performance Working Group

March 2013

Document History

Revison	Date	Author(s)	Comments
0.9	March 25, 2013	Greg Malatestinic, Jordan & Jordan Sara Rosen, EBS Alessandro Triglia, OSS Nokalva	Draft provided to HPWG for review.
0.9 RC1	May 28, 2013		Promoted to Release Candidate 1 for public release.

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

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

3 Definitions

GPB scalar type	A fundamental type provided by GPB. These include: • uint32 – a 32 bit signed integer • sint32 – a 32 bit signed inter
varint encoding	GPB optimization method of serializing integers using a variable number of bytes. Smaller numbers are encoded into fewer 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.
composite decimal	A GPB message type constructed for the purpose of transmitting decimal data as a pair of integers representing a mantissa and an exponent.

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, in this order. Within each of those clauses, GPB message and enumeration types shall be generated strictly in the order specified therein.

EXAMPLE

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

```
message HopGrp { // generated from a <component> element
      optional string hopCompId = 1;
      optional hopSendingTime = 2;
      optional hopRefId = 3;
}

mesaage StandardHeader { // generated from a <component> element
      optional string senderCompId = 1;
      optional string targetCompId = 2;
      optional string onBehalfOfCompId = 3;
      optional string deliverToCompId = 4;
      optional uint64 sendingTime = 5;
      repeated HopGrp hopGrp = 5;
}

enum IoiTransTypeEnum {
      IoiTransType_NEW = 0;
      IoiTransType_CANCEL = 1;
      IoiTransType_REPLACE = 2;
}
message Ioi { // generated from a <message> element
```

```
      optional StandardHeader standardHeader = 1;
      optional string ioiId = 2;
      optional IoiTransTypeEnum ioiTransType = 3;
      optional Instrument instrument = 4;
      optional SideEnum side = 5;
      optional IoiQty ioiQty = 6;
      optional uint32 price = 7;
      optional bool ioiNaturalFlag = 8;
}
```

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 sint32 and sint64 both satisfy a mapping from a FIX integral type, then the values of <encodingInfo> attributes @minValue and @maxValue will be taken into consideration in determining which integer type to select.

Also note that since GPB uses *varint* encoding for many of its scalar types, the <encodingInfo> attribute, *numBits*, does not define the actual number of bits used in the encoding. Rather, *numBits* will define the potential maximum number of bits to be encoded.

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.

There are no encoding attributes that are specific to GPB.

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	GPB uses varint encoding, so 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); if *isFixedPoint* is true, will generate a sint64; otherwise, will generate a composite decimal (a GPB message containing both a mantissa field and an exponent field). Composite decimals are defined in section 5.4.
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>	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 specifies the default exponent of a composite decimal. May be positive or negative. Scaled value = mantissa x $10^{exponent}$	0	Exponent value may be communicated in Rules of Engagement or provided as a default value of an exponent within a composite decimal in the GPB schema.

timeUnit	string	<datatype> <field> <fieldRef>	UTCTimeOnly UTCTimeStamp TZTimeOnly TZTimeStamp	Valid values are: • "day" • "second" • "millisecond" • "microsecond" • "nanosecond" • "picosecond"	nanosecond	Time unit for offset to epoch. May be specified in an encoding attribute or communicated in Rules of Engagement
epoch	string	<datatype> <field> <fieldRef>	UTCDateOnly UTCTimeStamp LocalMktDate TZTimeStamp	Reference epoch (either date or date and time) in ISO 8601 format. Timestamps indicate an integer offset to the epoch, for example: number of nanoseconds since 01/01/1970.	19700101	Epoch may be specified in an encoding attribute or communicated in Rules of Engagement

[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.

EXAMPLE 1

The following <datatype> element describes an integer datatype which, when mapped to GPB, will generate a uint32.

```
<datatype name="integer-with-bounds-ABC" ...>
      <encodingInfo minValue="2" maxValue="50"/>
</datatype>
```

EXAMPLE 2

The following <datatype> element describes a price datatype which, when mapped to GPB, will generate a sint32 representing an integer mantissa with an implicit integer exponent. The precision of the mantissa will be 32 bits, and the exponent (not encoded) will be fixed to -4.

```
<datatype name="Price-fixed-ABC" base="float">
```

```
        <encodingInfo numBits="32" isFixedPoint="true"
exponent="-4"/>
      </datatype>
```

EXAMPLE 3

The following <datatype> element describes a price datatype which, when mapped to GPB, will generate two fields - an sint64 representing an integer mantissa and an integer exponent. The maximum size of the mantissa will be 64 bits, and the exponent will have a default value of 4.

```
      <datatype name="Price-floating-ABC" base="float">
        <encodingInfo numBits="64" isFixedPoint="false" exponent="-4"/>
      </datatype>
```

EXAMPLE 4

The following <datatype> element describes a UTC timestamp datatype which, when mapped to GPB, will generate a uint64 representing a UTC timestamp. The integer will indicate the number of microseconds from the reference epoch and its precision will be 64 bits.

```
      <datatype name="Timestamp-ABC" base="UTCTimeStamp">
            <encodingInfo numBits=”64” timeUnit=”microsecond”/>
      </datatype>
```

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.

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 must be 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

Source substr	Target substr
UK	Uk
NERC	Nerc
CDS	Cds
IOI	Ioi
MD	Md

Source substr	Target substr
CFI	Cfi
ID	Id
XML	Xml
ISO	Iso
CP	Cp

RIC	Ric
USD	Usd
US	Us

EFP	Efp
GT	Gt
RFQ	Rfq

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 5.3, 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. The first character of a field name is converted to lower case.

The following table lists some examples.

Source field name	Target field name	Applied rule
Account	account	2
AdvRefID	advRefId	1,2
IOIID	ioiId	1,2
IOITransType	ioiTransType	1,2
LegIOIQty	legIoiQty	1, 2
MDEntryPx	mdEntryPx	1,2
GTBookingInst	gtBookingInst	1,2
CFICode	cfiCode	1,2
LegCFICode	legCfiCode	1,2

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. 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 upper case.
4. Prepend the value of the “name” attribute of the source <field> element followed by an underscore character, leaving its original case intact.

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 {
      HandlInst_AUTOMATED_EXECUTION_NO_INTERVENTION = 0;
      HandlInst_AUTOMATED_EXECUTION_INTERVENTION_OK = 1;
      HandlInst_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.

5 Mapping of FIX datatypes

5.1 Datatypes 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 table 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]	enumDataType attribute[c]	unionDataType attribute[e]	Effective datatype of the FIX field
1	no	no	no	no	the FIX datatype indicated in the type attribute of the a <field> element
3	no	**yes**	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	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	the FIX datatype (either MultipleCharValue or MultipleStringValue) with no restrictions on the items of the space-separated list
6	**yes**	**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 this <field> element
7	**yes**	no	**yes**	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	**yes**	a union between the type determined in case 1 above and the FIX datatype indicated in the unionDataType attribute
10	no	**yes**	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**	**yes**	a union between the type determined in case 4 above and the FIX datatype indicated in the `unionDataType` attribute
NOTE – there are no case numbers 2 and 9. 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 [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 5.1.2.2.

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, 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).

5.1.2.3 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 starting from 0.

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 message type:

```
enum AdvSideEnum {
      AdvSide_BUY = 0;
      AdvSide_SELL = 1;
      AdvSide_TRADE = 2;
      AdvSide_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 {
      AllocStatus_ACCEPTED = 0;
      AllocStatus_BLOCK_LEVEL_REJECT = 1;
      AllocStatus_ACCOUNT_LEVEL_REJECT = 2;
      // 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 5.1.3.1 to 5.1.3.2.

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).

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 {
      QuoteCondition_OPEN = 0;
      QuoteCondition_CLOSED = 1;
      QuoteCondition_EXCHANGE_BEST = 2;
      QuoteCondition_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 quoteCondition = 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, 9, 10, and 11 of table 3), a GPB message type shall be generated as specified in subclauses 5.1.4.1 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.

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

Case of table 3	**Type expression of the first alternative**	**Reference**
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
10	a reference to the enum generated from the <enum> child elements of this <field> element	subclause 5.1.
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 5.1.1

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	*uint32*
Reserved1000Plus	*uint32*
Reserved4000Plus	*uint32*
Qty	*sint64*
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 {
            PosMaintResult_SUCCESSFUL_COMPLETION = 0;
            PosMaintResult_REJECTED = 1;
            PosMaintResult_OTHER = 99;
      }

message PosMaintResultUnion {
      optional PosMaintResultEnum posMaintResult = 2;
      optional uint32 posMaintResultUint32 = 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 {
      SettlType_REGULAR = 0;
      SettlType_CASH = 1;
      SettlType_NEXT_DAY = 2;
      // …
}
message Tenor {
      optional uint32 days = 1;
      optional uint32 weeks = 2;
      optional uint32 months = 3;
      optional uint32 years = 4;
}

message SettlTypeUnion {
      optional SettlTypeEnum settlType = 1;
      optional Tenor settlTypeTenor = 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 = SettlType_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 {
      QuoteEntryRejectReason_UKNOWN_SYMBOL = 0;
      QuoteEntryRejectReason_EXCHANGE = 1;
      QuoteEntryRejectReason_QUOTE_REQUEST_EXCEEDS_LIMIT = 2;
```

```
      // …
}

message QuoteEntryRejectReasonUnion {
      optional QuoteEntryRejectReasonEnum quoteEntryRejectReason = 1;
      optional uint32 quoteEntryRejectReasonUint32 = 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 = QuoteEntryRejectReason_UNKNOWN_SYMBOL;

QuoteEntryRejectReasonUnion r22 = new QuoteEntryRejectReasonUnion();
r2.quoteEntryRejectReasonUint32 = 101;
```

5.2 Datatype Mapping Rules

5.2.1 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:

1) Define the "*source FIX datatype* " as the FIX datatype described by the <datatype> element referenced by the type attribute of the <field> element.
2) Scan table 6 from the top down, and select the first row such that the first cell of the row references either the *source FIX datatype* or one of its ancestor FIX datatypes.
3) Retreive the GPB type expression from the second cell of the selected row. (If the second cell contains multiple rows, take the type epression that satisfies the condition found in the adjacent cell of the row.)
4) If the GPB type expression references one of the *supporting GPB types* specified in a subclause of subclause 4.5, the subclause that specifies the supporting GPB type shall be invoked.

Table 6 – Datatype mapping rules

FIX datatype	GPB type expression	Comments
NumInGroup	N/A	NumInGroup fields are ignored.
DayOfMonth	uint32	Values are expected to be in the range [1, 31].
Reserved100Plus	uint32	
Reserved1000Plus	uint32	

Reserved4000Plus	unit32		
int	uint32	when *minValue* in the range [0, 2^32] and *maxValue* in the range [0, 2^32)	
	uint64	when *minValue* in the range [0, infinity	This case is applicable when maxValue is positive infinity and minValue is 0.
	sint32	when *minValue* in the range [-2^31, 0) and *maxValue* in the range (-2^31, 2^31]	
	sint64	when *(minValue* < 0 and *maxValue* >= 2^31) or minValue < -2^31	This case is applicable when maxValue is positive infinity and minValue is negative infinity.
float	double	when *isBinaryFloat* =true	SDecimal<N>EMinus<exp> and SDecimal<N>E<exp> are templates for supporting types and are defined in section 5.5.2. To generate the type expression: • replace <N> with the value of *numBits* • replace <exp> with the absolute value of *exponent*
	sint64	when isFixedPoint= true	
	SDecimal<N>EMinus<exp>	when isFixedPoint= false and *exponent* is negative	
	SDecimal<N>E<exp>	when isFixedPoint= false and *exponent* is positive	
UTCDateOnly	uint32		Number of days since epoch.
UTCTimeOnly	uint32	When *timeUnit* = seconds, milliseconds or microseconds	Number of time units since midnight UTC. The choice of type

	uint64	When *timeUnit* = nanoseconds or picoseconds, or when *timeUnit* is unspecified	expression is determined solely by *timeUnit*. Default is uint64 if *timeUnit* is unspecified.
UTCTimestamp	uint32	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*.
	uint64	when *numBits*=64 or is unspecified	
LocalMktDate	uint32		Number of days since epoch.
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	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	bytes	
boolean	bool	
char	bytes	The GPB bytes type is an arbitrary sequence of bytes. For our mapping we will restrict the sequence to just one byte.
String	string	
Tenor	Tenor	A supporting GPB message. See section 5.5.3.
MonthYear	uint32	The number of months since the epoch.
Pattern	string	

5.2.2 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:

1) 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.
2) Scan table 6 from the top down, and select the first row such that the first cell of the row references either the *source FIX datatype* or one of its ancestor FIX datatypes.

3) Retreive the GPB type expression from the second cell of the selected row. (If the second cell contains multiple rows, take the type epression that satisfies the condition found in the adjacent cell of the row.)
4) If the GPB type expression contains one or more names of GPB encoding attributes, replace each of them with the value of the *effective GPB encoding attribute* of the `<fieldRef>` element with that name.
5) If the GPB type expression references one of the *supporting GPB types* specified in a subclause of subclause 4.5, the subclause that specifies the supporting GPB type shall be invoked.

5.3 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 execInst = 21 [packed = true];
```

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

```
enum ExecInstEnum {
      ExecInst_STAY_ON_OFFER_SIDE = 0;
      ExecInst_NOT_HELD = 1;
      ExecInst_WORK = 2;
      ExecInst_GO_ALONG = 3;
      ExecInst_OVER_THE_DAY = 4;
      ExecInst_HELD = 5;
      // ...
}
```

5.4 Supporting GPB Types (including Composite Decimals)

While most datatypes will map to a single GPB field, some may require more than one field to represent the data. Here we present the GPB message types which must be constructed to support mapping of datatypes which result in multiple fields.

5.4.1 UDecimal<N>E[Minus]<exp>

A UDecimal<N>E[Minus]<exp> is a composite decimal type which holds a non-negative decimal value with a variable exponent. A default exponent is specified in the GPB schema and can be overridden in an instance of a

message over the wire. (Note that this default exponent is the value to be assigned to the exponent field when a value has not been specified and is unrelated to the default value of the encoding attribute, *exponent*.)

“UDecimal<N>E[Minus]<exp>" is a template for a dynamically created GPB message type, in which **N** is replaced by a number indicating the size of the mantissa (either 32 or 64 bits), **Minus** indicates the default exponent is negative, and **exp** is replaced by the absolute value of the default exponent.

On the first invocation of this subclause for given *N* and exponent, exactly one of the following GPB type assignments shall be generated, with N replaced by the size of the mantissa and <exp> replaced by the absolute value of the default exponent:

- when N = 32 and exp >= 0

```
message UDecimal32E<exp> {
    optional uint32 mantissa = 1;
    optional uint32 exponent = 2 [default = <exp>];
}
```

- when N = 64 and exp >= 0

```
message UDecimal64E<exp> {
    optional uint64 mantissa = 1;
    optional uint32 exponent = 2 [default = <exp>];
}
```

- when N = 32 and exp < 0

```
message UDecimal32EMinus<exp> {
    optional uint32 mantissa = 1;
    optional uint32 exponent = 2 [default = <exp>];
}
```

- when N =64 and exp < 0

```
message UDecimal64EMinus<exp> {
    optional uint64 mantissa = 1;
    optional uint32 exponent = 2 [default = <exp>];
}
```

5.4.2 SDecimal<N>E[Minus]<exp>

An SDecimal<N>E[Minus]<exp> is a composite decimal type which holds a signed decimal value with a variable exponent. A default exponent is specified in the GPB schema and can be overridden in an instance of a message over the wire.

“SDecimal<N>E[Minus]<exp>" is a template for a dynamically created GPB message type, in which **N** is replaced by a number indicating the size of the mantissa (either 32 or 64 bits), **Minus** indicates the default exponent is negative, and **exp** is replaced by the absolute value of the default exponent.

On the first invocation of this subclause for given *N* and exponent, exactly one of the following GPB type assignments shall be generated, with N replaced by the size of the mantissa and <exp> replaced by the absolute value of the default exponent:

- when N = 32 and exp >= 0

```
message SDecimal32E<exp> {
    optional sint32 mantissa = 1;
    optional uint32 exponent = 2 [default = <exp>];
}
```

- when N = 64 and exp >= 0

```
message SDecimal64E<exp> {
    optional sint64 mantissa = 1;
    optional uint32 exponent = 2 [default = <exp>];
}
```

- when N = 32 and exp < 0

```
message SDecimal32EMinus<exp> {
    optional sint32 mantissa = 1;
    optional uint32 exponent = 2 [default = <exp>];
}
```

- when N =64 and exp < 0

```
message SDecimal64EMinus<exp> {
    optional sint64 mantissa = 1;
    optional uint32 exponent = 2 [default = exp];
}
```

5.4.3 Tenor

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 uint32 days = 1;
      optional uint32 weeks = 2;
      optional uint32 months = 3;
      optional uint32 years = 4;
 }
```

5.5 Datatype mapping summary

Tables 7, 8 and 9 contain a summary of the mapping to GPB of all the datatypes defined in the FIX 5.0 SP2 Unified Repository using default encoding attribute values.

The values of the encoding attributes provided here are more specific than the default values in table 1 and take precedence over them. In the absence of encoding attributes these are the default type expressions.

Encoding attribute which are not applicable to the particular FIX datatype have been left blank.

Table 7 – Datatype mapping summary (part 1)

FIX datatype	FIX base datatype	min Value	max Value	min Length	max Length	Default GPB type expression
int		-infinity	+infinity			sint64
Length	int					uint32
TagNum	int					uint32
SeqNum	int					uint32
NumInGroup	int					N/A
DayOfMonth	int					uint32
char	char					bytes
Boolean	char					bool
String				0	+infinity	string
MultipleCharValue	String			0	+infinity	(repeated enumerated type name)
MultipleStringValue	String			0	+infinity	(repeated enumerated type name)
Country	String					string
Currency	String					string
Exchange	String			0	+infinity	string
data	String					string
Pattern						string
Tenor	Pattern					Tenor
MonthYear	String					uint32
Reserved100Plus	Pattern					uint32
Reserved1000Plus	Pattern					uint32

Reserved4000Plus	Pattern					uint32
XMLData	String					string
Language	String			0	+infinity	string
field with enumerated values	(any)					(an enumerated type name)
field with a union datatype	(any)					(a GPB message with optional fields for each option of the union)

Table 8 – Datatype mapping summary (part 2)

FIX datatype	FIX base datatype	isBinaryFloat	isFixedPoint	exponent	numBits	Default GPB type expression
Qty	float	false	false	0	64	SDecimal64E0
Price	float	false	false	6	64	SDecimal64EMinus6
PriceOffset	float	false	false	6	64	SDecimal64EMinus6
Amt	float	false	false	0	64	SDecimal64E0
Percentage	float	false	false	2	32	SDecimal32EMinus2

Table 9 – Datatype mapping summary (part 3)

FIX datatype	FIX base datatype	timeUnit	epoch	numBits	Default GPB type expression
UTCDateOnly	String		19700101		sint32
UTCTimeOnly	String	nanosecond			uint64
UTCTimestamp	String	nanosecond	19700101	64	uint64
LocalMktDate	String		19700101		sint32
TZTimeOnly	String				string
TZTimestamp	String				string

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 7.4.610.

EXAMPLE

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

message NewOrderSingle {
      optional StandardHeader standardHeader = 1;
      optional string clOrdId = 2;
      optional string secondaryClOrdId = 3;
      optional string clOrdLinkId = 4;
      repeated Parties parties = 5;
      optional uint32 tradeOriginationDate = 6;
      optional uint32 tradeDate = 7;
      optional string account = 8;
      optional AcctIDSourceEnum acctIdSource = 9;
      optional AccountTypeEnum accountType = 10;
      optional DayBookingInstEnum dayBookingInst = 11;
      optional BookingUnitEnum bookingUnit = 12;
      optional PreallocMethodEnum preallocMethod = 13;
      optional string allocId = 14;
      repeated PreAllocGrp preAllocGrp = 15;
      optional SettlTypeUnion settlType = 16;
      optional uint32 settlDate = 17;
      optional CashMarginEnum cashMargin = 18;
      optional ClearingFeeIndicatorEnum clearingFeeIndicator = 19;
      optional HandlInstEnum handlInst = 20;
      repeated ExecInstEnum execInst = 21 [packed = true];
      optional UDecimal64E0 minQty = 22;
      optional UDecimal64E0 matchIncrement = 23;
      optional sint64 maxPriceLevels = 24;
      optional DisplayInstruction displayInstruction = 25;
      optional UDecimal64E0 maxFloor = 26;
      optional string exDestination = 27;
      optional ExDestinationIDSourceEnum exDestinationIdSource = 28;
      repeated TrdgSesGrp trdgSesGrp = 29;
```

```
      optional ProcessCodeEnum processCode = 30;
      optional Instrument instrument = 31;

      // . . .
      optional StandardTrailer standardTrailer = 94;
}
```

For a complete listing, which contains the messages generated from all the components required by a NewOrderSingle message, see Appendix A.

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:

a) a sequence of fields and/or child components, each either required or optional; or
b) 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, and a field option (which is optional itself).

7.4 On the first invocation of this clause 7 for a given `<component>` element a GPB message shall be generated as specified in subclauses 7.4.1 to 7.4.6.

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

7.4.2 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 order, with the exceptions specified in subclause 7.4.3.

7.4.3 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 `MsgType` field (FIX field id = `35`);
- the `CheckSum` field (FIX field id = `10`).

7.4.4 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.

7.4.5 The field rule and type expression of each field corresponding to a `<fieldRef>` element shall be determined as specified in table 10.

Table 10 – Determination of the type expression of a component of the field specification

Case of table 3	**Field rule**	**Type expression of the field**	**Reference**
1	optional	The type expression determined from the type attribute of the <fieldRef>	subclause 5.3

		element as specified in subclaue 5.3.	
3	optional	A reference to the name of the GPB enum generated from the referenced `<field>` element as specified in subclause 5.2.2.	subclause 5.1.2
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 5.2.2.	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 5.2.3.	Subclause 5.2.3
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 5.2.3.	Subclause 5.2.3
8 10, 11	optional	A reference to the GPB message generated from a union of two datatypes	subclause 5.1.4

7.4.6 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.

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

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

EXAMPLE 1

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

```
message NestedParties {
      optional string nestedPartyId = 1;
      optional string nestedPartyIdSource = 2;
      optional sint64 nestedPartyRole = 3;
      repeated NstdPtysSubGrp nstdPtysSubGrp = 4;
}
```

which contains a reference to the following GPB message generated from its `<field>` and `<datatype>` elements:

```
message NstdPtysSubGrp {
      optional string nestedPartySubId = 1;
      optional sint64 nestedPartySubIdType = 2;
```

```
}
```

7.4.10 The GPB field number of a field shall be shall be generated from the position within the list of <fieldRef> and <componentRef> elements of the <component> starting with 1.

Appendix A Full Listing of NewOrderSingle

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

```
message UDecimal64E0 {
      optional sint64 mantissa = 1;
      optional uint32 exponent = 2 [default = 0];
}

message UDecimal64EMinus2 {
      optional sint64 mantissa = 1;
      optional uint32 exponent = 2 [default = 2];
}

message SDecimal64EMinus6 {
      optional sint64 mantissa = 1;
      optional uint32 exponent = 2 [default = 6];
}

message UDecimal64EMinus6 {
      optional sint64 mantissa = 1;
      optional uint32 exponent = 2 [default = 6];
}

message Tenor {
      optional uint32 days = 1;
      optional uint32 weeks = 2;
      optional uint32 months = 3;
      optional uint32 years = 4;
}

message HopGrp {
      optional string hopCompId = 1;
      optional uint64 hopSendingTime = 2;
      optional uint32 hopRefId = 3;
}

enum MsgTypeEnum {
      MsgType_HEARTBEAT = 0;
      MsgType_TEST_REQUEST = 1;
      MsgType_RESEND_REQUEST = 2;
      MsgType_REJECT = 3;
      MsgType_SEQUENCE_RESET = 4;
      MsgType_LOGOUT = 5;
      MsgType_IOI = 6;
      MsgType_ADVERTISEMENT = 7;
      MsgType_EXECUTION_REPORT = 8;
      MsgType_ORDER_CANCEL_REJECT = 9;
      MsgType_LOGON = 10;
      MsgType_DERIVATIVE_SECURITY_LIST = 11;
      MsgType_NEW_ORDER_MULTILEG = 12;
```

```
MsgType_MULTILEG_ORDER_CANCEL_REPLACE = 13;
MsgType_TRADE_CAPTURE_REPORT_REQUEST = 14;
MsgType_TRADE_CAPTURE_REPORT = 15;
MsgType_ORDER_MASS_STATUS_REQUEST = 16;
MsgType_QUOTE_REQUEST_REJECT = 17;
MsgType_RFQ_REQUEST = 18;
MsgType_QUOTE_STATUS_REPORT = 19;
MsgType_QUOTE_RESPONSE = 20;
MsgType_CONFIRMATION = 21;
MsgType_POSITION_MAINTENANCE_REQUEST = 22;
MsgType_POSITION_MAINTENANCE_REPORT = 23;
MsgType_REQUEST_FOR_POSITIONS = 24;
MsgType_REQUEST_FOR_POSITIONS_ACK = 25;
MsgType_POSITION_REPORT = 26;
MsgType_TRADE_CAPTURE_REPORT_REQUEST_ACK = 27;
MsgType_TRADE_CAPTURE_REPORT_ACK = 28;
MsgType_ALLOCATION_REPORT = 29;
MsgType_ALLOCATION_REPORT_ACK = 30;
MsgType_CONFIRMATION_ACK = 31;
MsgType_SETTLEMENT_INSTRUCTION_REQUEST = 32;
MsgType_ASSIGNMENT_REPORT = 33;
MsgType_COLLATERAL_REQUEST = 34;
MsgType_COLLATERAL_ASSIGNMENT = 35;
MsgType_COLLATERAL_RESPONSE = 36;
MsgType_NEWS = 37;
MsgType_COLLATERAL_REPORT = 38;
MsgType_COLLATERAL_INQUIRY = 39;
MsgType_NETWORK_COUNTERPARTY_SYSTEM_STATUS_REQUEST = 40;
MsgType_NETWORK_COUNTERPARTY_SYSTEM_STATUS_RESPONSE = 41;
MsgType_USER_REQUEST = 42;
MsgType_USER_RESPONSE = 43;
MsgType_COLLATERAL_INQUIRY_ACK = 44;
MsgType_CONFIRMATION_REQUEST = 45;
MsgType_TRADING_SESSION_LIST_REQUEST = 46;
MsgType_TRADING_SESSION_LIST = 47;
MsgType_SECURITY_LIST_UPDATE_REPORT = 48;
MsgType_ADJUSTED_POSITION_REPORT = 49;
MsgType_ALLOCATION_INSTRUCTION_ALERT = 50;
MsgType_EXECUTION_ACKNOWLEDGEMENT = 51;
MsgType_CONTRARY_INTENTION_REPORT = 52;
MsgType_SECURITY_DEFINITION_UPDATE_REPORT = 53;
MsgType_SETTLEMENT_OBLIGATION_REPORT = 54;
MsgType_DERIVATIVE_SECURITY_LIST_UPDATE_REPORT = 55;
MsgType_TRADING_SESSION_LIST_UPDATE_REPORT = 56;
MsgType_MARKET_DEFINITION_REQUEST = 57;
MsgType_MARKET_DEFINITION = 58;
MsgType_MARKET_DEFINITION_UPDATE_REPORT = 59;
MsgType_APPLICATION_MESSAGE_REQUEST = 60;
MsgType_APPLICATION_MESSAGE_REQUEST_ACK = 61;
MsgType_APPLICATION_MESSAGE_REPORT = 62;
MsgType_ORDER_MASS_ACTION_REPORT = 63;
MsgType_EMAIL = 64;
MsgType_ORDER_MASS_ACTION_REQUEST = 65;
MsgType_USER_NOTIFICATION = 66;
MsgType_STREAM_ASSIGNMENT_REQUEST = 67;
MsgType_STREAM_ASSIGNMENT_REPORT = 68;
MsgType_STREAM_ASSIGNMENT_REPORT_ACK = 69;
```

```
      MsgType_NEW_ORDER_SINGLE = 70;
      MsgType_NEW_ORDER_LIST = 71;
      MsgType_ORDER_CANCEL_REQUEST = 72;
      MsgType_ORDER_CANCEL_REPLACE_REQUEST = 73;
      MsgType_ORDER_STATUS_REQUEST = 74;
      MsgType_ALLOCATION_INSTRUCTION = 75;
      MsgType_LIST_CANCEL_REQUEST = 76;
      MsgType_LIST_EXECUTE = 77;
      MsgType_LIST_STATUS_REQUEST = 78;
      MsgType_LIST_STATUS = 79;
      MsgType_ALLOCATION_INSTRUCTION_ACK = 80;
      MsgType_DONT_KNOW_TRADE = 81;
      MsgType_QUOTE_REQUEST = 82;
      MsgType_QUOTE = 83;
      MsgType_SETTLEMENT_INSTRUCTIONS = 84;
      MsgType_MARKET_DATA_REQUEST = 85;
      MsgType_MARKET_DATA_SNAPSHOT_FULL_REFRESH = 86;
      MsgType_MARKET_DATA_INCREMENTAL_REFRESH = 87;
      MsgType_MARKET_DATA_REQUEST_REJECT = 88;
      MsgType_QUOTE_CANCEL = 89;
      MsgType_QUOTE_STATUS_REQUEST = 90;
      MsgType_MASS_QUOTE_ACKNOWLEDGEMENT = 91;
      MsgType_SECURITY_DEFINITION_REQUEST = 92;
      MsgType_SECURITY_DEFINITION = 93;
      MsgType_SECURITY_STATUS_REQUEST = 94;
      MsgType_SECURITY_STATUS = 95;
      MsgType_TRADING_SESSION_STATUS_REQUEST = 96;
      MsgType_TRADING_SESSION_STATUS = 97;
      MsgType_MASS_QUOTE = 98;
      MsgType_BUSINESS_MESSAGE_REJECT = 99;
      MsgType_BID_REQUEST = 100;
      MsgType_BID_RESPONSE = 101;
      MsgType_LIST_STRIKE_PRICE = 102;
      MsgType_XMLNON_FIX = 103;
      MsgType_REGISTRATION_INSTRUCTIONS = 104;
      MsgType_REGISTRATION_INSTRUCTIONS_RESPONSE = 105;
      MsgType_ORDER_MASS_CANCEL_REQUEST = 106;
      MsgType_ORDER_MASS_CANCEL_REPORT = 107;
      MsgType_NEW_ORDER_CROSS = 108;
      MsgType_CROSS_ORDER_CANCEL_REPLACE_REQUEST = 109;
      MsgType_CROSS_ORDER_CANCEL_REQUEST = 110;
      MsgType_SECURITY_TYPE_REQUEST = 111;
      MsgType_SECURITY_TYPES = 112;
      MsgType_SECURITY_LIST_REQUEST = 113;
      MsgType_SECURITY_LIST = 114;
      MsgType_DERIVATIVE_SECURITY_LIST_REQUEST = 115;
}

enum ApplVerIdEnum {
      ApplVerId_FIX27 = 0;
      ApplVerId_FIX30 = 1;
      ApplVerId_FIX40 = 2;
      ApplVerId_FIX41 = 3;
      ApplVerId_FIX42 = 4;
      ApplVerId_FIX43 = 5;
      ApplVerId_FIX44 = 6;
      ApplVerId_FIX50 = 7;
```

```
      ApplVerId_FIX50SP1 = 8;
      ApplVerId_FIX50SP2 = 9;
}

enum PossDupFlagEnum {
      PossDupFlag_ORIGINAL_TRANSMISSION = 0;
      PossDupFlag_POSSIBLE_DUPLICATE = 1;
}

enum PossResendEnum {
      PossResend_ORIGINAL_TRANSMISSION = 0;
      PossResend_POSSIBLE_RESEND = 1;
}

message StandardHeader {
      optional string beginString = 1;
      optional uint32 bodyLength = 2;
      optional MsgTypeEnum msgType = 3;
      optional ApplVerIdEnum applVerId = 4;
      optional sint64 applExtId = 5;
      optional string cstmApplVerId = 6;
      optional string senderCompId = 7;
      optional string targetCompId = 8;
      optional string onBehalfOfCompId = 9;
      optional string deliverToCompId = 10;
      optional uint32 secureDataLen = 11;
      optional bytes secureData = 12;
      optional uint32 msgSeqNum = 13;
      optional string senderSubId = 14;
      optional string senderLocationId = 15;
      optional string targetSubId = 16;
      optional string targetLocationId = 17;
      optional string onBehalfOfSubId = 18;
      optional string onBehalfOfLocationId = 19;
      optional string deliverToSubId = 20;
      optional string deliverToLocationId = 21;
      optional PossDupFlagEnum possDupFlag = 22;
      optional PossResendEnum possResend = 23;
      optional uint64 sendingTime = 24;
      optional uint64 origSendingTime = 25;
      optional uint32 xmlDataLen = 26;
      optional bytes xmlData = 27;
      optional string messageEncoding = 28;
      optional uint32 lastMsgSeqNumProcessed = 29;
      repeated HopGrp hopGrp = 30;
}

enum PartySubIdTypeEnum {
      PartySubIdType_FIRM = 0;
      PartySubIdType_PERSON = 1;
      PartySubIdType_SYSTEM = 2;
      PartySubIdType_APPLICATION = 3;
      PartySubIdType_FULL_LEGAL_NAME_OF_FIRM = 4;
      PartySubIdType_POSTAL_ADDRESS = 5;
      PartySubIdType_PHONE_NUMBER = 6;
      PartySubIdType_EMAIL_ADDRESS = 7;
      PartySubIdType_CONTACT_NAME = 8;
```

```
	PartySubIdType_SECURITIES_ACCOUNT_NUMBER = 9;
	PartySubIdType_REGISTRATION_NUMBER = 10;
	PartySubIdType_REGISTERED_ADDRESS_FOR_CONFIRMATION = 11;
	PartySubIdType_REGULATORY_STATUS = 12;
	PartySubIdType_REGISTRATION_NAME = 13;
	PartySubIdType_CASH_ACCOUNT_NUMBER = 14;
	PartySubIdType_BIC = 15;
	PartySubIdType_CSDPARTICIPANT_MEMBER_CODE = 16;
	PartySubIdType_REGISTERED_ADDRESS = 17;
	PartySubIdType_FUND_ACCOUNT_NAME = 18;
	PartySubIdType_TELEX_NUMBER = 19;
	PartySubIdType_FAX_NUMBER = 20;
	PartySubIdType_SECURITIES_ACCOUNT_NAME = 21;
	PartySubIdType_CASH_ACCOUNT_NAME = 22;
	PartySubIdType_DEPARTMENT = 23;
	PartySubIdType_LOCATION_DESK = 24;
	PartySubIdType_POSITION_ACCOUNT_TYPE = 25;
	PartySubIdType_SECURITY_LOCATE_ID = 26;
	PartySubIdType_MARKET_MAKER = 27;
	PartySubIdType_ELIGIBLE_COUNTERPARTY = 28;
	PartySubIdType_PROFESSIONAL_CLIENT = 29;
	PartySubIdType_LOCATION = 30;
	PartySubIdType_EXECUTION_VENUE = 31;
	PartySubIdType_CURRENCY_DELIVERY_IDENTIFIER = 32;
}

message PartySubIdTypeUnion {
	optional PartySubIdTypeEnum partySubIdType = 1;
	optional uint32 partySubIdTypeUint32 = 2;
}

message PtysSubGrp {
	optional string partySubId = 1;
	optional PartySubIdTypeUnion partySubIdType = 2;
}

enum PartyIdSourceEnum {
	PartyIdSource_UK_NATIONAL_INSURANCE_OR_PENSION_NUMBER = 0;
	PartyIdSource_US_SOCIAL_SECURITY_NUMBER = 1;
	PartyIdSource_US_EMPLOYER_OR_TAX_ID_NUMBER = 2;
	PartyIdSource_AUSTRALIAN_BUSINESS_NUMBER = 3;
	PartyIdSource_AUSTRALIAN_TAX_FILE_NUMBER = 4;
	PartyIdSource_KOREAN_INVESTOR_ID = 5;
	PartyIdSource_TAIWANESE_FOREIGN_INVESTOR_ID = 6;
	PartyIdSource_TAIWANESE_TRADING_ACCT = 7;
	PartyIdSource_MALAYSIAN_CENTRAL_DEPOSITORY = 8;
	PartyIdSource_CHINESE_INVESTOR_ID = 9;
	PartyIdSource_ISITC_ACRONYM = 10;
	PartyIdSource_BIC = 11;
	PartyIdSource_GENERAL_IDENTIFIER = 12;
	PartyIdSource_PROPRIETARY = 13;
	PartyIdSource_ISO_COUNTRY_CODE = 14;
	PartyIdSource_SETTLEMENT_ENTITY_LOCATION = 15;
	PartyIdSource_MIC = 16;
	PartyIdSource_CSDPARTICIPANT = 17;
}
```

```
enum PartyRoleEnum {
      PartyRole_EXECUTING_FIRM = 0;
      PartyRole_BROKER_OF_CREDIT = 1;
      PartyRole_CLIENT_ID = 2;
      PartyRole_CLEARING_FIRM = 3;
      PartyRole_INVESTOR_ID = 4;
      PartyRole_INTRODUCING_FIRM = 5;
      PartyRole_ENTERING_FIRM = 6;
      PartyRole_LOCATE = 7;
      PartyRole_FUND_MANAGER_CLIENT_ID = 8;
      PartyRole_SETTLEMENT_LOCATION = 9;
      PartyRole_ORDER_ORIGINATION_TRADER = 10;
      PartyRole_EXECUTING_TRADER = 11;
      PartyRole_ORDER_ORIGINATION_FIRM = 12;
      PartyRole_GIVEUP_CLEARING_FIRM = 13;
      PartyRole_CORRESPONDANT_CLEARING_FIRM = 14;
      PartyRole_EXECUTING_SYSTEM = 15;
      PartyRole_CONTRA_FIRM = 16;
      PartyRole_CONTRA_CLEARING_FIRM = 17;
      PartyRole_SPONSORING_FIRM = 18;
      PartyRole_UNDERLYING_CONTRA_FIRM = 19;
      PartyRole_CLEARING_ORGANIZATION = 20;
      PartyRole_EXCHANGE = 21;
      PartyRole_CUSTOMER_ACCOUNT = 22;
      PartyRole_CORRESPONDENT_CLEARING_ORGANIZATION = 23;
      PartyRole_CORRESPONDENT_BROKER = 24;
      PartyRole_BUYER = 25;
      PartyRole_CUSTODIAN = 26;
      PartyRole_INTERMEDIARY = 27;
      PartyRole_AGENT = 28;
      PartyRole_SUB_CUSTODIAN = 29;
      PartyRole_BENEFICIARY = 30;
      PartyRole_INTERESTED_PARTY = 31;
      PartyRole_REGULATORY_BODY = 32;
      PartyRole_LIQUIDITY_PROVIDER = 33;
      PartyRole_ENTERING_TRADER = 34;
      PartyRole_CONTRA_TRADER = 35;
      PartyRole_POSITION_ACCOUNT = 36;
      PartyRole_CONTRA_INVESTOR_ID = 37;
      PartyRole_TRANSFER_TO_FIRM = 38;
      PartyRole_CONTRA_POSITION_ACCOUNT = 39;
      PartyRole_CONTRA_EXCHANGE = 40;
      PartyRole_INTERNAL_CARRY_ACCOUNT = 41;
      PartyRole_ORDER_ENTRY_OPERATOR_ID = 42;
      PartyRole_SECONDARY_ACCOUNT_NUMBER = 43;
      PartyRole_FOREIGN_FIRM = 44;
      PartyRole_THIRD_PARTY_ALLOCATION_FIRM = 45;
      PartyRole_CLAIMING_ACCOUNT = 46;
      PartyRole_ASSET_MANAGER = 47;
      PartyRole_PLEDGOR_ACCOUNT = 48;
      PartyRole_PLEDGEE_ACCOUNT = 49;
      PartyRole_LARGE_TRADER_REPORTABLE_ACCOUNT = 50;
      PartyRole_TRADER_MNEMONIC = 51;
      PartyRole_SENDER_LOCATION = 52;
      PartyRole_SESSION_ID = 53;
      PartyRole_ACCEPTABLE_COUNTERPARTY = 54;
      PartyRole_UNACCEPTABLE_COUNTERPARTY = 55;
```

```
      PartyRole_ENTERING_UNIT = 56;
      PartyRole_EXECUTING_UNIT = 57;
      PartyRole_INTRODUCING_BROKER = 58;
      PartyRole_QUOTE_ORIGINATOR = 59;
      PartyRole_REPORT_ORIGINATOR = 60;
      PartyRole_SYSTEMATIC_INTERNALISER = 61;
      PartyRole_MULTILATERAL_TRADING_FACILITY = 62;
      PartyRole_REGULATED_MARKET = 63;
      PartyRole_MARKET_MAKER = 64;
      PartyRole_INVESTMENT_FIRM = 65;
      PartyRole_HOST_COMPETENT_AUTHORITY = 66;
      PartyRole_HOME_COMPETENT_AUTHORITY = 67;
      PartyRole_COMPETENT_AUTHORITY_LIQUIDITY = 68;
      PartyRole_COMPETENT_AUTHORITY_TRANSACTION_VENUE = 69;
      PartyRole_REPORTING_INTERMEDIARY = 70;
      PartyRole_EXECUTION_VENUE = 71;
      PartyRole_MARKET_DATA_ENTRY_ORIGINATOR = 72;
      PartyRole_LOCATION_ID = 73;
      PartyRole_DESK_ID = 74;
      PartyRole_MARKET_DATA_MARKET = 75;
      PartyRole_ALLOCATION_ENTITY = 76;
      PartyRole_PRIME_BROKER = 77;
      PartyRole_STEP_OUT_FIRM = 78;
      PartyRole_BROKER_CLEARING_ID = 79;
      PartyRole_CENTRAL_REGISTRATION_DEPOSITORY = 80;
      PartyRole_CLEARING_ACCOUNT = 81;
      PartyRole_ACCEPTABLE_SETTLING_COUNTERPARTY = 82;
      PartyRole_UNACCEPTABLE_SETTLING_COUNTERPARTY = 83;
}

message Parties {
      optional string partyId = 1;
      optional PartyIdSourceEnum partyIdSource = 2;
      optional PartyRoleEnum partyRole = 3;
      repeated PtysSubGrp ptysSubGrp = 4;
}

message NstdPtysSubGrp {
      optional string nestedPartySubId = 1;
      optional sint64 nestedPartySubIdType = 2;
}

message NestedParties {
      optional string nestedPartyId = 1;
      optional string nestedPartyIdSource = 2;
      optional sint64 nestedPartyRole = 3;
      repeated NstdPtysSubGrp nstdPtysSubGrp = 4;
}

message PreAllocGrp {
      optional string allocAccount = 1;
      optional sint64 allocAcctIdSource = 2;
      optional string allocSettlCurrency = 3;
      optional string individualAllocId = 4;
      repeated NestedParties nestedParties = 5;
      optional UDecimal64E0 allocQty = 6;
}
```

```
enum DisplayWhenEnum {
      DisplayWhen_IMMEDIATE = 0;
      DisplayWhen_EXHAUST = 1;
}

enum DisplayMethodEnum {
      DisplayMethod_INITIAL = 0;
      DisplayMethod_NEW = 1;
      DisplayMethod_RANDOM = 2;
      DisplayMethod_UNDISCLOSED = 3;
}

message DisplayInstruction {
      optional UDecimal64E0 displayQty = 1;
      optional UDecimal64E0 secondaryDisplayQty = 2;
      optional DisplayWhenEnum displayWhen = 3;
      optional DisplayMethodEnum displayMethod = 4;
      optional UDecimal64E0 displayLowQty = 5;
      optional UDecimal64E0 displayHighQty = 6;
      optional UDecimal64E0 displayMinIncr = 7;
      optional UDecimal64E0 refreshQty = 8;
}

enum TradingSessionIdEnum {
      TradingSessionId_DAY = 0;
      TradingSessionId_HALF_DAY = 1;
      TradingSessionId_MORNING = 2;
      TradingSessionId_AFTERNOON = 3;
      TradingSessionId_EVENING = 4;
      TradingSessionId_AFTER_HOURS = 5;
}

enum TradingSessionSubIdEnum {
      TradingSessionSubId_PRE_TRADING = 0;
      TradingSessionSubId_OPENING_OR_OPENING_AUCTION = 1;
      TradingSessionSubId_CONTINUOUS = 2;
      TradingSessionSubId_CLOSING_OR_CLOSING_AUCTION = 3;
      TradingSessionSubId_POST_TRADING = 4;
      TradingSessionSubId_INTRADAY_AUCTION = 5;
      TradingSessionSubId_QUIESCENT = 6;
}

message TradingSessionIdUnion {
      optional TradingSessionIdEnum tradingSessionId = 1;
      optional uint32 tradingSessionIdUint32 = 2;
}

message TradingSessionSubIdUnion {
      optional TradingSessionSubIdEnum tradingSessionSubId = 1;
      optional uint32 tradingSessionSubIdUint32 = 2;
}

message TrdgSesGrp {
      optional TradingSessionIdUnion tradingSessionId = 1;
      optional TradingSessionSubIdUnion tradingSessionSubId = 2;
}
```

```
message SecAltIdGrp {
      optional string securityAltId = 1;
      optional string securityAltIdSource = 2;
}

message SecurityXml {
      optional uint32 securityXmlLen = 1;
      optional string securityXml = 2;
      optional string securityXmlSchema = 3;
}

enum EventTypeEnum {
      EventType_PUT = 0;
      EventType_CALL = 1;
      EventType_TENDER = 2;
      EventType_SINKING_FUND_CALL = 3;
      EventType_ACTIVATION = 4;
      EventType_INACTIVIATION = 5;
      EventType_LAST_ELIGIBLE_TRADE_DATE = 6;
      EventType_SWAP_START_DATE = 7;
      EventType_SWAP_END_DATE = 8;
      EventType_SWAP_ROLL_DATE = 9;
      EventType_SWAP_NEXT_START_DATE = 10;
      EventType_SWAP_NEXT_ROLL_DATE = 11;
      EventType_FIRST_DELIVERY_DATE = 12;
      EventType_LAST_DELIVERY_DATE = 13;
      EventType_INITIAL_INVENTORY_DUE_DATE = 14;
      EventType_FINAL_INVENTORY_DUE_DATE = 15;
      EventType_FIRST_INTENT_DATE = 16;
      EventType_LAST_INTENT_DATE = 17;
      EventType_POSITION_REMOVAL_DATE = 18;
      EventType_OTHER = 19;
}

message EventTypeUnion {
      optional EventTypeEnum eventType = 1;
      optional uint32 eventTypeUint32 = 2;
}

message EvntGrp {
      optional EventTypeUnion eventType = 1;
      optional uint32 eventDate = 2;
      optional uint64 eventTime = 3;
      optional UDecimal64EMinus6 eventPx = 4;
      optional string eventText = 5;
}

message InstrumentPtysSubGrp {
      optional string instrumentPartySubId = 1;
      optional sint64 instrumentPartySubIdType = 2;
}

message InstrumentParties {
      optional string instrumentPartyId = 1;
      optional string instrumentPartyIdSource = 2;
      optional sint64 instrumentPartyRole = 3;
```

```
      repeated InstrumentPtysSubGrp instrumentPtysSubGrp = 4;
}

message ComplexEventTimes {
      optional uint64 complexEventStartTime = 1;
      optional uint64 complexEventEndTime = 2;
}

message ComplexEventDates {
      optional uint64 complexEventStartDate = 1;
      optional uint64 complexEventEndDate = 2;
      repeated ComplexEventTimes complexEventTimes = 3;
}

enum ComplexEventTypeEnum {
      ComplexEventType_CAPPED = 0;
      ComplexEventType_TRIGGER = 1;
      ComplexEventType_KNOCK_IN_UP = 2;
      ComplexEventType_KOCK_IN_DOWN = 3;
      ComplexEventType_KNOCK_OUT_UP = 4;
      ComplexEventType_KNOCK_OUT_DOWN = 5;
      ComplexEventType_UNDERLYING = 6;
      ComplexEventType_RESET_BARRIER = 7;
      ComplexEventType_ROLLING_BARRIER = 8;
}

enum ComplexEventPriceBoundaryMethodEnum {
      ComplexEventPriceBoundaryMethod_LESS_THAN_COMPLEX_EVENT_PRICE = 0;
      ComplexEventPriceBoundaryMethod_LESS_THAN_OR_EQUAL_TO_COMPLEX_EVENT_PRICE =
1;
      ComplexEventPriceBoundaryMethod_EQUAL_TO_COMPLEX_EVENT_PRICE = 2;
      ComplexEventPriceBoundaryMethod_GREATER_THAN_OR_EQUAL_TO_COMPLEX_EVENT_PRICE
= 3;
      ComplexEventPriceBoundaryMethod_GREATER_THAN_COMPLEX_EVENT_PRICE = 4;
}

enum ComplexEventPriceTimeTypeEnum {
      ComplexEventPriceTimeType_EXPIRATION = 0;
      ComplexEventPriceTimeType_IMMEDIATE = 1;
      ComplexEventPriceTimeType_SPECIFIED_DATE = 2;
}

enum ComplexEventConditionEnum {
      ComplexEventCondition_AND = 0;
      ComplexEventCondition_OR = 1;
}

message ComplexEvents {
      optional ComplexEventTypeEnum complexEventType = 1;
      optional UDecimal64E0 complexOptPayoutAmount = 2;
      optional UDecimal64EMinus6 complexEventPrice = 3;
      optional ComplexEventPriceBoundaryMethodEnum complexEventPriceBoundaryMethod
= 4;
      optional UDecimal64EMinus2 complexEventPriceBoundaryPrecision = 5;
      optional ComplexEventPriceTimeTypeEnum complexEventPriceTimeType = 6;
      optional ComplexEventConditionEnum complexEventCondition = 7;
      repeated ComplexEventDates complexEventDates = 8;
```

```
}

enum SymbolSfxEnum {
      SymbolSfx_EUCP_WITH_LUMP_SUM_INTEREST = 0;
      SymbolSfx_WHEN_ISSUED = 1;
}

enum SecurityIdSourceEnum {
      SecurityIdSource_CUSIP = 0;
      SecurityIdSource_SEDOL = 1;
      SecurityIdSource_QUIK = 2;
      SecurityIdSource_ISIN_NUMBER = 3;
      SecurityIdSource_RIC_CODE = 4;
      SecurityIdSource_ISO_CURRENCY_CODE = 5;
      SecurityIdSource_ISO_COUNTRY_CODE = 6;
      SecurityIdSource_EXCHANGE_SYMBOL = 7;
      SecurityIdSource_CONSOLIDATED_TAPE_ASSOCIATION = 8;
      SecurityIdSource_BLOOMBERG_SYMBOL = 9;
      SecurityIdSource_WERTPAPIER = 10;
      SecurityIdSource_DUTCH = 11;
      SecurityIdSource_VALOREN = 12;
      SecurityIdSource_SICOVAM = 13;
      SecurityIdSource_BELGIAN = 14;
      SecurityIdSource_COMMON = 15;
      SecurityIdSource_CLEARING_HOUSE = 16;
      SecurityIdSource_ISDA_FPML_SPECIFICATION = 17;
      SecurityIdSource_OPTION_PRICE_REPORTING_AUTHORITY = 18;
      SecurityIdSource_ISDA_FPML_URL = 19;
      SecurityIdSource_LETTER_OF_CREDIT = 20;
      SecurityIdSource_MARKETPLACE_ASSIGNED_IDENTIFIER = 21;
}

enum ProductEnum {
      Product_AGENCY = 0;
      Product_COMMODITY = 1;
      Product_CORPORATE = 2;
      Product_CURRENCY = 3;
      Product_EQUITY = 4;
      Product_GOVERNMENT = 5;
      Product_INDEX = 6;
      Product_LOAN = 7;
      Product_MONEYMARKET = 8;
      Product_MORTGAGE = 9;
      Product_MUNICIPAL = 10;
      Product_OTHER = 11;
      Product_FINANCING = 12;
}

enum SecurityTypeEnum {
      SecurityType_US_TREASURY_NOTE_OLD = 0;
      SecurityType_US_TREASURY_BILL_OLD = 1;
      SecurityType_EURO_SUPRANATIONAL_COUPONS = 2;
      SecurityType_FEDERAL_AGENCY_COUPON = 3;
      SecurityType_FEDERAL_AGENCY_DISCOUNT_NOTE = 4;
      SecurityType_PRIVATE_EXPORT_FUNDING = 5;
      SecurityType_USD_SUPRANATIONAL_COUPONS = 6;
      SecurityType_CORPORATE_BOND = 7;
```

```
SecurityType_CORPORATE_PRIVATE_PLACEMENT = 8;
SecurityType_CONVERTIBLE_BOND = 9;
SecurityType_DUAL_CURRENCY = 10;
SecurityType_EURO_CORPORATE_BOND = 11;
SecurityType_EURO_CORPORATE_FLOATING_RATE_NOTES = 12;
SecurityType_US_CORPORATE_FLOATING_RATE_NOTES = 13;
SecurityType_INDEXED_LINKED = 14;
SecurityType_STRUCTURED_NOTES = 15;
SecurityType_YANKEE_CORPORATE_BOND = 16;
SecurityType_FOREIGN_EXCHANGE_CONTRACT = 17;
SecurityType_NON_DELIVERABLE_FORWARD = 18;
SecurityType_FX_SPOT = 19;
SecurityType_FX_FORWARD = 20;
SecurityType_FX_SWAP = 21;
SecurityType_CREDIT_DEFAULT_SWAP = 22;
SecurityType_FUTURE = 23;
SecurityType_OPTION = 24;
SecurityType_OPTIONS_ON_FUTURES = 25;
SecurityType_OPTIONS_ON_PHYSICAL = 26;
SecurityType_INTEREST_RATE_SWAP = 27;
SecurityType_OPTIONS_ON_COMBO = 28;
SecurityType_COMMON_STOCK = 29;
SecurityType_PREFERRED_STOCK = 30;
SecurityType_REPURCHASE = 31;
SecurityType_FORWARD = 32;
SecurityType_BUY_SELLBACK = 33;
SecurityType_SECURITIES_LOAN = 34;
SecurityType_SECURITIES_PLEDGE = 35;
SecurityType_BRADY_BOND = 36;
SecurityType_CANADIAN_TREASURY_NOTES = 37;
SecurityType_CANADIAN_TREASURY_BILLS = 38;
SecurityType_EURO_SOVEREIGNS = 39;
SecurityType_CANADIAN_PROVINCIAL_BONDS = 40;
SecurityType_TREASURY_BILL = 41;
SecurityType_US_TREASURY_BOND = 42;
SecurityType_INTEREST_STRIP_FROM_ANY_BOND_OR_NOTE = 43;
SecurityType_US_TREASURY_BILL = 44;
SecurityType_TREASURY_INFLATION_PROTECTED_SECURITIES = 45;
SecurityType_PRINCIPAL_STRIP_OF_ACALLABLE_BOND_OR_NOTE = 46;
SecurityType_PRINCIPAL_STRIP_FROM_ANON_CALLABLE_BOND_OR_NOTE = 47;
SecurityType_US_TREASURY_NOTE = 48;
SecurityType_TERM_LOAN = 49;
SecurityType_REVOLVER_LOAN = 50;
SecurityType_REVOLVER = 51;
SecurityType_BRIDGE_LOAN = 52;
SecurityType_LETTER_OF_CREDIT = 53;
SecurityType_SWING_LINE_FACILITY = 54;
SecurityType_DEBTOR_IN_POSSESSION = 55;
SecurityType_DEFAULTED = 56;
SecurityType_WITHDRAWN = 57;
SecurityType_REPLACED = 58;
SecurityType_MATURED = 59;
SecurityType_AMENDED = 60;
SecurityType_RETIRED = 61;
SecurityType_BANKERS_ACCEPTANCE = 62;
SecurityType_BANK_DEPOSITORY_NOTE = 63;
SecurityType_BANK_NOTES = 64;
```

```
	SecurityType_BILL_OF_EXCHANGES = 65;
	SecurityType_CANADIAN_MONEY_MARKETS = 66;
	SecurityType_CERTIFICATE_OF_DEPOSIT = 67;
	SecurityType_CALL_LOANS = 68;
	SecurityType_COMMERCIAL_PAPER = 69;
	SecurityType_DEPOSIT_NOTES = 70;
	SecurityType_EURO_CERTIFICATE_OF_DEPOSIT = 71;
	SecurityType_EURO_COMMERCIAL_PAPER = 72;
	SecurityType_LIQUIDITY_NOTE = 73;
	SecurityType_MEDIUM_TERM_NOTES = 74;
	SecurityType_OVERNIGHT = 75;
	SecurityType_PROMISSORY_NOTE = 76;
	SecurityType_SHORT_TERM_LOAN_NOTE = 77;
	SecurityType_PLAZOS_FIJOS = 78;
	SecurityType_SECURED_LIQUIDITY_NOTE = 79;
	SecurityType_TIME_DEPOSIT = 80;
	SecurityType_TERM_LIQUIDITY_NOTE = 81;
	SecurityType_EXTENDED_COMM_NOTE = 82;
	SecurityType_YANKEE_CERTIFICATE_OF_DEPOSIT = 83;
	SecurityType_ASSET_BACKED_SECURITIES = 84;
	SecurityType_CANADIAN_MORTGAGE_BONDS = 85;
	SecurityType_CORP = 86;
	SecurityType_COLLATERALIZED_MORTGAGE_OBLIGATION = 87;
	SecurityType_IOETTE_MORTGAGE = 88;
	SecurityType_MORTGAGE_BACKED_SECURITIES = 89;
	SecurityType_MORTGAGE_INTEREST_ONLY = 90;
	SecurityType_MORTGAGE_PRINCIPAL_ONLY = 91;
	SecurityType_MORTGAGE_PRIVATE_PLACEMENT = 92;
	SecurityType_MISCELLANEOUS_PASS_THROUGH = 93;
	SecurityType_PFANDBRIEFE = 94;
	SecurityType_TO_BE_ANNOUNCED = 95;
	SecurityType_OTHER_ANTICIPATION_NOTES = 96;
	SecurityType_CERTIFICATE_OF_OBLIGATION = 97;
	SecurityType_CERTIFICATE_OF_PARTICIPATION = 98;
	SecurityType_GENERAL_OBLIGATION_BONDS = 99;
	SecurityType_MANDATORY_TENDER = 100;
	SecurityType_REVENUE_ANTICIPATION_NOTE = 101;
	SecurityType_REVENUE_BONDS = 102;
	SecurityType_SPECIAL_ASSESSMENT = 103;
	SecurityType_SPECIAL_OBLIGATION = 104;
	SecurityType_SPECIAL_TAX = 105;
	SecurityType_TAX_ANTICIPATION_NOTE = 106;
	SecurityType_TAX_ALLOCATION = 107;
	SecurityType_TAX_EXEMPT_COMMERCIAL_PAPER = 108;
	SecurityType_TAXABLE_MUNICIPAL_CP = 109;
	SecurityType_TAX_REVENUE_ANTICIPATION_NOTE = 110;
	SecurityType_VARIABLE_RATE_DEMAND_NOTE = 111;
	SecurityType_WARRANT = 112;
	SecurityType_MUTUAL_FUND = 113;
	SecurityType_MULTILEG_INSTRUMENT = 114;
	SecurityType_NO_SECURITY_TYPE = 115;
	SecurityType_WILDCARD = 116;
	SecurityType_CASH = 117;
}

enum SecurityStatusEnum {
	SecurityStatus_ACTIVE = 0;
```

```
      SecurityStatus_INACTIVE = 1;
}

enum RestructuringTypeEnum {
      RestructuringType_FULL_RESTRUCTURING = 0;
      RestructuringType_MODIFIED_RESTRUCTURING = 1;
      RestructuringType_MODIFIED_MOD_RESTRUCTURING = 2;
      RestructuringType_NO_RESTRUCTURING_SPECIFIED = 3;
}

enum SeniorityEnum {
      Seniority_SENIOR_SECURED = 0;
      Seniority_SENIOR = 1;
      Seniority_SUBORDINATED = 2;
}

enum StrikePriceDeterminationMethodEnum {
      StrikePriceDeterminationMethod_FIXED_STRIKE = 0;
      StrikePriceDeterminationMethod_STRIKE_SET_AT_EXPIRATION = 1;
      StrikePriceDeterminationMethod_STRIKE_SET_TO_AVERAGE_ACROSS_LIFE = 2;
      StrikePriceDeterminationMethod_STRIKE_SET_TO_OPTIMAL_VALUE = 3;
}

enum StrikePriceBoundaryMethodEnum {
      StrikePriceBoundaryMethod_LESS_THAN = 0;
      StrikePriceBoundaryMethod_LESS_THAN_OR_EQUAL = 1;
      StrikePriceBoundaryMethod_EQUAL = 2;
      StrikePriceBoundaryMethod_GREATER_THAN_OR_EQUAL = 3;
      StrikePriceBoundaryMethod_GREATER_THAN = 4;
}

enum UnderlyingPriceDeterminationMethodEnum {
      UnderlyingPriceDeterminationMethod_REGULAR = 0;
      UnderlyingPriceDeterminationMethod_SPECIAL_REFERENCE = 1;
      UnderlyingPriceDeterminationMethod_OPTIMAL_VALUE = 2;
}

enum ContractMultiplierUnitEnum {
      ContractMultiplierUnit_SHARES = 0;
      ContractMultiplierUnit_HOURS = 1;
      ContractMultiplierUnit_DAYS = 2;
}

enum FlowScheduleTypeEnum {
      FlowScheduleType_NERC_EASTERN_OFF_PEAK = 0;
      FlowScheduleType_NERC_WESTERN_OFF_PEAK = 1;
      FlowScheduleType_NERC_CALENDAR_ALL_DAYS_IN_MONTH = 2;
      FlowScheduleType_NERC_EASTERN_PEAK = 3;
      FlowScheduleType_NERC_WESTERN_PEAK = 4;
}

enum UnitOfMeasureEnum {
      UnitOfMeasure_BILLION_CUBIC_FEET = 0;
      UnitOfMeasure_MILLION_BARRELS = 1;
      UnitOfMeasure_ONE_MILLION_BTU = 2;
      UnitOfMeasure_MEGAWATT_HOURS = 3;
      UnitOfMeasure_BARRELS = 4;
```

```
      UnitOfMeasure_BUSHELS = 5;
      UnitOfMeasure_POUNDS = 6;
      UnitOfMeasure_GALLONS = 7;
      UnitOfMeasure_TROY_OUNCES = 8;
      UnitOfMeasure_METRIC_TONS = 9;
      UnitOfMeasure_TONS = 10;
      UnitOfMeasure_USDOLLARS = 11;
      UnitOfMeasure_ALLOWANCES = 12;
}

enum SettlMethodEnum {
      SettlMethod_CASH_SETTLEMENT_REQUIRED = 0;
      SettlMethod_PHYSICAL_SETTLEMENT_REQUIRED = 1;
}

enum ExerciseStyleEnum {
      ExerciseStyle_EUROPEAN = 0;
      ExerciseStyle_AMERICAN = 1;
      ExerciseStyle_BERMUDA = 2;
}

enum OptPayoutTypeEnum {
      OptPayoutType_VANILLA = 0;
      OptPayoutType_CAPPED = 1;
      OptPayoutType_BINARY = 2;
}

enum PriceQuoteMethodEnum {
      PriceQuoteMethod_STANDARD = 0;
      PriceQuoteMethod_INDEX = 1;
      PriceQuoteMethod_INTEREST_RATE_INDEX = 2;
      PriceQuoteMethod_PERCENT_OF_PAR = 3;
}

enum ValuationMethodEnum {
      ValuationMethod_PREMIUM_STYLE = 0;
      ValuationMethod_FUTURES_STYLE_MARK_TO_MARKET = 1;
      ValuationMethod_FUTURES_STYLE_WITH_AN_ATTACHED_CASH_ADJUSTMENT = 2;
      ValuationMethod_CDS_STYLE_COLLATERALIZATION = 3;
      ValuationMethod_CDS_IN_DELIVERY_USE_RECOVERY_RATE_TO_CALCULATE = 4;
}

enum ListMethodEnum {
      ListMethod_PRE_LISTED_ONLY = 0;
      ListMethod_USER_REQUESTED = 1;
}

enum PutOrCallEnum {
      PutOrCall_PUT = 0;
      PutOrCall_CALL = 1;
}

enum TimeUnitEnum {
      TimeUnit_HOUR = 0;
      TimeUnit_MINUTE = 1;
      TimeUnit_SECOND = 2;
      TimeUnit_DAY = 3;
```

```
      TimeUnit_WEEK = 4;
      TimeUnit_MONTH = 5;
      TimeUnit_YEAR = 6;
}

enum CpProgramEnum {
      CpProgram_PROGRAM3A3 = 0;
      CpProgram_PROGRAM42 = 1;
      CpProgram_OTHER = 2;
}

message StrikePriceDeterminationMethodUnion {
      optional StrikePriceDeterminationMethodEnum strikePriceDeterminationMethod =
1;
      optional uint32 strikePriceDeterminationMethodUint32 = 2;
}

message FlowScheduleTypeUnion {
      optional FlowScheduleTypeEnum flowScheduleType = 1;
      optional uint32 flowScheduleTypeUint32 = 2;
}

message CpProgramUnion {
      optional CpProgramEnum cpProgram = 1;
      optional uint32 cpProgramUint32 = 2;
}

message Instrument {
      optional string symbol = 1;
      optional SymbolSfxEnum symbolSfx = 2;
      optional string securityId = 3;
      optional SecurityIdSourceEnum securityIdSource = 4;
      repeated SecAltIdGrp secAltIdGrp = 5;
      optional ProductEnum product = 6;
      optional string productComplex = 7;
      optional string securityGroup = 8;
      optional string cfiCode = 9;
      optional SecurityTypeEnum securityType = 10;
      optional string securitySubType = 11;
      optional uint32 maturityMonthYear = 12;
      optional uint32 maturityDate = 13;
      optional string maturityTime = 14;
      optional string settleOnOpenFlag = 15;
      optional string instrmtAssignmentMethod = 16;
      optional SecurityStatusEnum securityStatus = 17;
      optional uint32 couponPaymentDate = 18;
      optional RestructuringTypeEnum restructuringType = 19;
      optional SeniorityEnum seniority = 20;
      optional UDecimal64EMinus2 notionalPercentageOutstanding = 21;
      optional UDecimal64EMinus2 originalNotionalPercentageOutstanding = 22;
      optional UDecimal64EMinus2 attachmentPoint = 23;
      optional UDecimal64EMinus2 detachmentPoint = 24;
      optional uint32 issueDate = 25;
      optional string repoCollateralSecurityType = 26;
      optional sint64 repurchaseTerm = 27;
      optional UDecimal64EMinus2 repurchaseRate = 28;
      optional double factor = 29;
```

```
      optional string creditRating = 30;
      optional string instrRegistry = 31;
      optional string countryOfIssue = 32;
      optional string stateOrProvinceOfIssue = 33;
      optional string localeOfIssue = 34;
      optional uint32 redemptionDate = 35;
      optional UDecimal64EMinus6 strikePrice = 36;
      optional string strikeCurrency = 37;
      optional double strikeMultiplier = 38;
      optional double strikeValue = 39;
      optional StrikePriceDeterminationMethodUnion strikePriceDeterminationMethod =
40;
      optional StrikePriceBoundaryMethodEnum strikePriceBoundaryMethod = 41;
      optional UDecimal64EMinus2 strikePriceBoundaryPrecision = 42;
      optional UnderlyingPriceDeterminationMethodEnum
underlyingPriceDeterminationMethod = 43;
      optional string optAttribute = 44;
      optional double contractMultiplier = 45;
      optional ContractMultiplierUnitEnum contractMultiplierUnit = 46;
      optional FlowScheduleTypeUnion flowScheduleType = 47;
      optional double minPriceIncrement = 48;
      optional UDecimal64E0 minPriceIncrementAmount = 49;
      optional UnitOfMeasureEnum unitOfMeasure = 50;
      optional UDecimal64E0 unitOfMeasureQty = 51;
      optional string priceUnitOfMeasure = 52;
      optional UDecimal64E0 priceUnitOfMeasureQty = 53;
      optional SettlMethodEnum settlMethod = 54;
      optional ExerciseStyleEnum exerciseStyle = 55;
      optional OptPayoutTypeEnum optPayoutType = 56;
      optional UDecimal64E0 optPayoutAmount = 57;
      optional PriceQuoteMethodEnum priceQuoteMethod = 58;
      optional ValuationMethodEnum valuationMethod = 59;
      optional ListMethodEnum listMethod = 60;
      optional UDecimal64EMinus6 capPrice = 61;
      optional UDecimal64EMinus6 floorPrice = 62;
      optional PutOrCallEnum putOrCall = 63;
      optional bool flexibleIndicator = 64;
      optional bool flexProductEligibilityIndicator = 65;
      optional TimeUnitEnum timeUnit = 66;
      optional UDecimal64EMinus2 couponRate = 67;
      optional string securityExchange = 68;
      optional sint64 positionLimit = 69;
      optional sint64 ntPositionLimit = 70;
      optional string issuer = 71;
      optional uint32 encodedIssuerLen = 72;
      optional bytes encodedIssuer = 73;
      optional string securityDesc = 74;
      optional uint32 encodedSecurityDescLen = 75;
      optional bytes encodedSecurityDesc = 76;
      optional SecurityXml securityXml = 77;
      optional string pool = 78;
      optional uint32 contractSettlMonth = 79;
      optional CpProgramUnion cpProgram = 80;
      optional string cpRegType = 81;
      repeated EvntGrp evntGrp = 82;
      optional uint32 datedDate = 83;
      optional uint32 interestAccrualDate = 84;
```

```
	repeated InstrumentParties instrumentParties = 85;
	repeated ComplexEvents complexEvents = 86;
}

enum TerminationTypeEnum {
	TerminationType_OVERNIGHT = 0;
	TerminationType_TERM = 1;
	TerminationType_FLEXIBLE = 2;
	TerminationType_OPEN = 3;
}

enum DeliveryTypeEnum {
	DeliveryType_VERSUS_PAYMENT = 0;
	DeliveryType_FREE = 1;
	DeliveryType_TRI_PARTY = 2;
	DeliveryType_HOLD_IN_CUSTODY = 3;
}

message FinancingDetails {
	optional string agreementDesc = 1;
	optional string agreementId = 2;
	optional uint32 agreementDate = 3;
	optional string agreementCurrency = 4;
	optional TerminationTypeEnum terminationType = 5;
	optional uint32 startDate = 6;
	optional uint32 endDate = 7;
	optional DeliveryTypeEnum deliveryType = 8;
	optional UDecimal64EMinus2 marginRatio = 9;
}

message UndSecAltIdGrp {
	optional string underlyingSecurityAltId = 1;
	optional string underlyingSecurityAltIdSource = 2;
}

message UnderlyingStipulations {
	optional string underlyingStipType = 1;
	optional string underlyingStipValue = 2;
}

message UndlyInstrumentPtysSubGrp {
	optional string underlyingInstrumentPartySubId = 1;
	optional sint64 underlyingInstrumentPartySubIdType = 2;
}

message UndlyInstrumentParties {
	optional string underlyingInstrumentPartyId = 1;
	optional string underlyingInstrumentPartyIdSource = 2;
	optional sint64 underlyingInstrumentPartyRole = 3;
	repeated UndlyInstrumentPtysSubGrp undlyInstrumentPtysSubGrp = 4;
}

enum UnderlyingSettlementTypeEnum {
	UnderlyingSettlementType_TPLUS1 = 0;
	UnderlyingSettlementType_TPLUS3 = 1;
	UnderlyingSettlementType_TPLUS4 = 2;
}
```

```
enum UnderlyingCashTypeEnum {
      UnderlyingCashType_FIXED = 0;
      UnderlyingCashType_DIFF = 1;
}

enum UnderlyingFxRateCalcEnum {
      UnderlyingFxRateCalc_DIVIDE = 0;
      UnderlyingFxRateCalc_MULTIPLY = 1;
}

message UnderlyingFlowScheduleTypeUnion {
      optional sint64 underlyingFlowScheduleType = 1; // No enums for this field
were found in source repository.
      optional uint32 underlyingFlowScheduleTypeUint32 = 2;
}

message UnderlyingInstrument {
      optional string underlyingSymbol = 1;
      optional string underlyingSymbolSfx = 2;
      optional string underlyingSecurityId = 3;
      optional string underlyingSecurityIdSource = 4;
      repeated UndSecAltIdGrp undSecAltIdGrp = 5;
      optional sint64 underlyingProduct = 6;
      optional string underlyingCfiCode = 7;
      optional string underlyingSecurityType = 8;
      optional string underlyingSecuritySubType = 9;
      optional uint32 underlyingMaturityMonthYear = 10;
      optional uint32 underlyingMaturityDate = 11;
      optional string underlyingMaturityTime = 12;
      optional uint32 underlyingCouponPaymentDate = 13;
      optional string underlyingRestructuringType = 14;
      optional string underlyingSeniority = 15;
      optional UDecimal64EMinus2 underlyingNotionalPercentageOutstanding = 16;
      optional UDecimal64EMinus2 underlyingOriginalNotionalPercentageOutstanding =
17;
      optional UDecimal64EMinus2 underlyingAttachmentPoint = 18;
      optional UDecimal64EMinus2 underlyingDetachmentPoint = 19;
      optional uint32 underlyingIssueDate = 20;
      optional string underlyingRepoCollateralSecurityType = 21;
      optional sint64 underlyingRepurchaseTerm = 22;
      optional UDecimal64EMinus2 underlyingRepurchaseRate = 23;
      optional double underlyingFactor = 24;
      optional string underlyingCreditRating = 25;
      optional string underlyingInstrRegistry = 26;
      optional string underlyingCountryOfIssue = 27;
      optional string underlyingStateOrProvinceOfIssue = 28;
      optional string underlyingLocaleOfIssue = 29;
      optional uint32 underlyingRedemptionDate = 30;
      optional UDecimal64EMinus6 underlyingStrikePrice = 31;
      optional string underlyingStrikeCurrency = 32;
      optional string underlyingOptAttribute = 33;
      optional double underlyingContractMultiplier = 34;
      optional sint64 underlyingContractMultiplierUnit = 35;
      optional UnderlyingFlowScheduleTypeUnion underlyingFlowScheduleType = 36;
      optional string underlyingUnitOfMeasure = 37;
      optional UDecimal64E0 underlyingUnitOfMeasureQty = 38;
```

```
	optional string underlyingPriceUnitOfMeasure = 39;
	optional UDecimal64E0 underlyingPriceUnitOfMeasureQty = 40;
	optional string underlyingTimeUnit = 41;
	optional sint64 underlyingExerciseStyle = 42;
	optional UDecimal64EMinus2 underlyingCouponRate = 43;
	optional string underlyingSecurityExchange = 44;
	optional string underlyingIssuer = 45;
	optional uint32 encodedUnderlyingIssuerLen = 46;
	optional bytes encodedUnderlyingIssuer = 47;
	optional string underlyingSecurityDesc = 48;
	optional uint32 encodedUnderlyingSecurityDescLen = 49;
	optional bytes encodedUnderlyingSecurityDesc = 50;
	optional string underlyingCpProgram = 51;
	optional string underlyingCpRegType = 52;
	optional UDecimal64EMinus2 underlyingAllocationPercent = 53;
	optional string underlyingCurrency = 54;
	optional UDecimal64E0 underlyingQty = 55;
	optional UnderlyingSettlementTypeEnum underlyingSettlementType = 56;
	optional UDecimal64E0 underlyingCashAmount = 57;
	optional UnderlyingCashTypeEnum underlyingCashType = 58;
	optional UDecimal64EMinus6 underlyingPx = 59;
	optional UDecimal64EMinus6 underlyingDirtyPrice = 60;
	optional UDecimal64EMinus6 underlyingEndPrice = 61;
	optional UDecimal64E0 underlyingStartValue = 62;
	optional UDecimal64E0 underlyingCurrentValue = 63;
	optional UDecimal64E0 underlyingEndValue = 64;
	repeated UnderlyingStipulations underlyingStipulations = 65;
	optional UDecimal64E0 underlyingAdjustedQuantity = 66;
	optional double underlyingFxRate = 67;
	optional UnderlyingFxRateCalcEnum underlyingFxRateCalc = 68;
	optional UDecimal64E0 underlyingCapValue = 69;
	repeated UndlyInstrumentParties undlyInstrumentParties = 70;
	optional string underlyingSettlMethod = 71;
	optional sint64 underlyingPutOrCall = 72;
}

message UndInstrmtGrp {
	optional UnderlyingInstrument underlyingInstrument = 1;
}

enum StipulationTypeEnum {
	StipulationType_ALTERNATIVE_MINIMUM_TAX = 0;
	StipulationType_AUTO_REINVESTMENT = 1;
	StipulationType_BANK_QUALIFIED = 2;
	StipulationType_BARGAIN_CONDITIONS = 3;
	StipulationType_COUPON_RANGE = 4;
	StipulationType_ISO_CURRENCY_CODE = 5;
	StipulationType_CUSTOM_START = 6;
	StipulationType_GEOGRAPHICS = 7;
	StipulationType_VALUATION_DISCOUNT = 8;
	StipulationType_INSURED = 9;
	StipulationType_ISSUE_DATE = 10;
	StipulationType_ISSUER = 11;
	StipulationType_ISSUE_SIZE_RANGE = 12;
	StipulationType_LOOKBACK_DAYS = 13;
	StipulationType_EXPLICIT_LOT_IDENTIFIER = 14;
	StipulationType_LOT_VARIANCE = 15;
```

```
StipulationType_MATURITY_YEAR_AND_MONTH = 16;
StipulationType_MATURITY_RANGE = 17;
StipulationType_MAXIMUM_SUBSTITUTIONS = 18;
StipulationType_MINIMUM_DENOMINATION = 19;
StipulationType_MINIMUM_INCREMENT = 20;
StipulationType_MINIMUM_QUANTITY = 21;
StipulationType_PAYMENT_FREQUENCY = 22;
StipulationType_NUMBER_OF_PIECES = 23;
StipulationType_POOLS_MAXIMUM = 24;
StipulationType_POOLS_PER_LOT = 25;
StipulationType_POOLS_PER_MILLION = 26;
StipulationType_POOLS_PER_TRADE = 27;
StipulationType_PRICE_RANGE = 28;
StipulationType_PRICING_FREQUENCY = 29;
StipulationType_PRODUCTION_YEAR = 30;
StipulationType_CALL_PROTECTION = 31;
StipulationType_PURPOSE = 32;
StipulationType_BENCHMARK_PRICE_SOURCE = 33;
StipulationType_RATING_SOURCE_AND_RANGE = 34;
StipulationType_TYPE_OF_REDEMPTION = 35;
StipulationType_RESTRICTED = 36;
StipulationType_MARKET_SECTOR = 37;
StipulationType_SECURITY_TYPE_INCLUDED_OR_EXCLUDED = 38;
StipulationType_STRUCTURE = 39;
StipulationType_SUBSTITUTIONS_FREQUENCY = 40;
StipulationType_SUBSTITUTIONS_LEFT = 41;
StipulationType_FREEFORM_TEXT = 42;
StipulationType_TRADE_VARIANCE = 43;
StipulationType_WEIGHTED_AVERAGE_COUPON = 44;
StipulationType_WEIGHTED_AVERAGE_LIFE_COUPON = 45;
StipulationType_WEIGHTED_AVERAGE_LOAN_AGE = 46;
StipulationType_WEIGHTED_AVERAGE_MATURITY = 47;
StipulationType_WHOLE_POOL = 48;
StipulationType_YIELD_RANGE = 49;
StipulationType_AVERAGE_FICOSCORE = 50;
StipulationType_AVERAGE_LOAN_SIZE = 51;
StipulationType_MAXIMUM_LOAN_BALANCE = 52;
StipulationType_POOL_IDENTIFIER = 53;
StipulationType_TYPE_OF_ROLL_TRADE = 54;
StipulationType_REFERENCE_TO_ROLLING_OR_CLOSING_TRADE = 55;
StipulationType_PRINCIPAL_OF_ROLLING_OR_CLOSING_TRADE = 56;
StipulationType_INTEREST_OF_ROLLING_OR_CLOSING_TRADE = 57;
StipulationType_AVAILABLE_OFFER_QUANTITY_TO_BE_SHOWN_TO_THE_STREET = 58;
StipulationType_BROKER_CREDIT = 59;
StipulationType_OFFER_PRICE_TO_BE_SHOWN_TO_INTERNAL_BROKERS = 60;
StipulationType_OFFER_QUANTITY_TO_BE_SHOWN_TO_INTERNAL_BROKERS = 61;
StipulationType_THE_MINIMUM_RESIDUAL_OFFER_QUANTITY = 62;
StipulationType_MAXIMUM_ORDER_SIZE = 63;
StipulationType_ORDER_QUANTITY_INCREMENT = 64;
StipulationType_PRIMARY_OR_SECONDARY_MARKET_INDICATOR = 65;
StipulationType_BROKER_SALES_CREDIT_OVERRIDE = 66;
StipulationType_TRADER_CREDIT = 67;
StipulationType_DISCOUNT_RATE = 68;
StipulationType_YIELD_TO_MATURITY = 69;
StipulationType_ABSOLUTE_PREPAYMENT_SPEED = 70;
StipulationType_CONSTANT_PREPAYMENT_PENALTY = 71;
StipulationType_CONSTANT_PREPAYMENT_RATE = 72;
```

```
      StipulationType_CONSTANT_PREPAYMENT_YIELD = 73;
      StipulationType_FINAL_CP_ROF_HOME_EQUITY_PREPAYMENT_CURVE = 74;
      StipulationType_PERCENT_OF_MANUFACTURED_HOUSING_PREPAYMENT_CURVE = 75;
      StipulationType_MONTHLY_PREPAYMENT_RATE = 76;
      StipulationType_PERCENT_OF_PROSPECTUS_PREPAYMENT_CURVE = 77;
      StipulationType_PERCENT_OF_BMAPREPAYMENT_CURVE = 78;
      StipulationType_SINGLE_MONTHLY_MORTALITY = 79;
}

message Stipulations {
      optional StipulationTypeEnum stipulationType = 1;
      optional string stipulationValue = 2;
}

enum RoundingDirectionEnum {
      RoundingDirection_ROUND_TO_NEAREST = 0;
      RoundingDirection_ROUND_DOWN = 1;
      RoundingDirection_ROUND_UP = 2;
}

message OrderQtyData {
      optional UDecimal64E0 orderQty = 1;
      optional UDecimal64E0 cashOrderQty = 2;
      optional UDecimal64EMinus2 orderPercent = 3;
      optional RoundingDirectionEnum roundingDirection = 4;
      optional double roundingModulus = 5;
}

enum TriggerTypeEnum {
      TriggerType_PARTIAL_EXECUTION = 0;
      TriggerType_SPECIFIED_TRADING_SESSION = 1;
      TriggerType_NEXT_AUCTION = 2;
      TriggerType_PRICE_MOVEMENT = 3;
}

enum TriggerActionEnum {
      TriggerAction_ACTIVATE = 0;
      TriggerAction_MODIFY = 1;
      TriggerAction_CANCEL = 2;
}

enum TriggerPriceTypeEnum {
      TriggerPriceType_BEST_OFFER = 0;
      TriggerPriceType_LAST_TRADE = 1;
      TriggerPriceType_BEST_BID = 2;
      TriggerPriceType_BEST_BID_OR_LAST_TRADE = 3;
      TriggerPriceType_BEST_OFFER_OR_LAST_TRADE = 4;
      TriggerPriceType_BEST_MID = 5;
}

enum TriggerPriceTypeScopeEnum {
      TriggerPriceTypeScope_NONE = 0;
      TriggerPriceTypeScope_LOCAL = 1;
      TriggerPriceTypeScope_NATIONAL = 2;
      TriggerPriceTypeScope_GLOBAL = 3;
}
```

```
enum TriggerPriceDirectionEnum {
      TriggerPriceDirection_UP = 0;
      TriggerPriceDirection_DOWN = 1;
}

enum TriggerOrderTypeEnum {
      TriggerOrderType_MARKET = 0;
      TriggerOrderType_LIMIT = 1;
}

message TriggeringInstruction {
      optional TriggerTypeEnum triggerType = 1;
      optional TriggerActionEnum triggerAction = 2;
      optional UDecimal64EMinus6 triggerPrice = 3;
      optional string triggerSymbol = 4;
      optional string triggerSecurityId = 5;
      optional string triggerSecurityIdSource = 6;
      optional string triggerSecurityDesc = 7;
      optional TriggerPriceTypeEnum triggerPriceType = 8;
      optional TriggerPriceTypeScopeEnum triggerPriceTypeScope = 9;
      optional TriggerPriceDirectionEnum triggerPriceDirection = 10;
      optional UDecimal64EMinus6 triggerNewPrice = 11;
      optional TriggerOrderTypeEnum triggerOrderType = 12;
      optional UDecimal64E0 triggerNewQty = 13;
      optional string triggerTradingSessionId = 14;
      optional string triggerTradingSessionSubId = 15;
}

enum BenchmarkCurveNameEnum {
      BenchmarkCurveName_EONIA = 0;
      BenchmarkCurveName_EUREPO = 1;
      BenchmarkCurveName_EURIBOR = 2;
      BenchmarkCurveName_FUTURE_SWAP = 3;
      BenchmarkCurveName_LIBID = 4;
      BenchmarkCurveName_LIBOR = 5;
      BenchmarkCurveName_MUNI_AAA = 6;
      BenchmarkCurveName_OTHER = 7;
      BenchmarkCurveName_PFANDBRIEFE = 8;
      BenchmarkCurveName_SONIA = 9;
      BenchmarkCurveName_SWAP = 10;
      BenchmarkCurveName_TREASURY = 11;
}

message SpreadOrBenchmarkCurveData {
      optional SDecimal64EMinus6 spread = 1;
      optional string benchmarkCurveCurrency = 2;
      optional BenchmarkCurveNameEnum benchmarkCurveName = 3;
      optional string benchmarkCurvePoint = 4;
      optional UDecimal64EMinus6 benchmarkPrice = 5;
      optional sint64 benchmarkPriceType = 6;
      optional string benchmarkSecurityId = 7;
      optional string benchmarkSecurityIdSource = 8;
}

enum YieldTypeEnum {
      YieldType_AFTER_TAX_YIELD = 0;
      YieldType_ANNUAL_YIELD = 1;
```

```
	YieldType_YIELD_AT_ISSUE = 2;
	YieldType_YIELD_TO_AVERAGE_MATURITY = 3;
	YieldType_BOOK_YIELD = 4;
	YieldType_YIELD_TO_NEXT_CALL = 5;
	YieldType_YIELD_CHANGE_SINCE_CLOSE = 6;
	YieldType_CLOSING_YIELD = 7;
	YieldType_COMPOUND_YIELD = 8;
	YieldType_CURRENT_YIELD = 9;
	YieldType_GVNT_EQUIVALENT_YIELD = 10;
	YieldType_TRUE_GROSS_YIELD = 11;
	YieldType_YIELD_WITH_INFLATION_ASSUMPTION = 12;
	YieldType_INVERSE_FLOATER_BOND_YIELD = 13;
	YieldType_MOST_RECENT_CLOSING_YIELD = 14;
	YieldType_CLOSING_YIELD_MOST_RECENT_MONTH = 15;
	YieldType_CLOSING_YIELD_MOST_RECENT_QUARTER = 16;
	YieldType_CLOSING_YIELD_MOST_RECENT_YEAR = 17;
	YieldType_YIELD_TO_LONGEST_AVERAGE_LIFE = 18;
	YieldType_MARK_TO_MARKET_YIELD = 19;
	YieldType_YIELD_TO_MATURITY = 20;
	YieldType_YIELD_TO_NEXT_REFUND = 21;
	YieldType_OPEN_AVERAGE_YIELD = 22;
	YieldType_PREVIOUS_CLOSE_YIELD = 23;
	YieldType_PROCEEDS_YIELD = 24;
	YieldType_YIELD_TO_NEXT_PUT = 25;
	YieldType_SEMI_ANNUAL_YIELD = 26;
	YieldType_YIELD_TO_SHORTEST_AVERAGE_LIFE = 27;
	YieldType_SIMPLE_YIELD = 28;
	YieldType_TAX_EQUIVALENT_YIELD = 29;
	YieldType_YIELD_TO_TENDER_DATE = 30;
	YieldType_TRUE_YIELD = 31;
	YieldType_YIELD_VALUE_OF32NDS = 32;
	YieldType_YIELD_TO_WORST = 33;
}

message YieldData {
	optional YieldTypeEnum yieldType = 1;
	optional UDecimal64EMinus2 yield = 2;
	optional uint32 yieldCalcDate = 3;
	optional uint32 yieldRedemptionDate = 4;
	optional UDecimal64EMinus6 yieldRedemptionPrice = 5;
	optional sint64 yieldRedemptionPriceType = 6;
}

enum CommTypeEnum {
	CommType_PER_UNIT = 0;
	CommType_PERCENT = 1;
	CommType_ABSOLUTE = 2;
	CommType_PERCENTAGE_WAIVED_CASH_DISCOUNT = 3;
	CommType_PERCENTAGE_WAIVED_ENHANCED_UNITS = 4;
	CommType_POINTS_PER_BOND_OR_CONTRACT = 5;
}

enum FundRenewWaivEnum {
	FundRenewWaiv_NO = 0;
	FundRenewWaiv_YES = 1;
}
```

```
message CommissionData {
      optional UDecimal64E0 commission = 1;
      optional CommTypeEnum commType = 2;
      optional string commCurrency = 3;
      optional FundRenewWaivEnum fundRenewWaiv = 4;
}

enum PegPriceTypeEnum {
      PegPriceType_LAST_PEG = 0;
      PegPriceType_MID_PRICE_PEG = 1;
      PegPriceType_OPENING_PEG = 2;
      PegPriceType_MARKET_PEG = 3;
      PegPriceType_PRIMARY_PEG = 4;
      PegPriceType_PEG_TO_VWAP = 5;
      PegPriceType_TRAILING_STOP_PEG = 6;
      PegPriceType_PEG_TO_LIMIT_PRICE = 7;
}

enum PegMoveTypeEnum {
      PegMoveType_FLOATING = 0;
      PegMoveType_FIXED = 1;
}

enum PegOffsetTypeEnum {
      PegOffsetType_PRICE = 0;
      PegOffsetType_BASIS_POINTS = 1;
      PegOffsetType_TICKS = 2;
      PegOffsetType_PRICE_TIER = 3;
}

enum PegLimitTypeEnum {
      PegLimitType_OR_BETTER = 0;
      PegLimitType_STRICT = 1;
      PegLimitType_OR_WORSE = 2;
}

enum PegRoundDirectionEnum {
      PegRoundDirection_MORE_AGGRESSIVE = 0;
      PegRoundDirection_MORE_PASSIVE = 1;
}

enum PegScopeEnum {
      PegScope_LOCAL = 0;
      PegScope_NATIONAL = 1;
      PegScope_GLOBAL = 2;
      PegScope_NATIONAL_EXCLUDING_LOCAL = 3;
}

message PegInstructions {
      optional double pegOffsetValue = 1;
      optional PegPriceTypeEnum pegPriceType = 2;
      optional PegMoveTypeEnum pegMoveType = 3;
      optional PegOffsetTypeEnum pegOffsetType = 4;
      optional PegLimitTypeEnum pegLimitType = 5;
      optional PegRoundDirectionEnum pegRoundDirection = 6;
      optional PegScopeEnum pegScope = 7;
      optional string pegSecurityIdSource = 8;
```

```
      optional string pegSecurityId = 9;
      optional string pegSymbol = 10;
      optional string pegSecurityDesc = 11;
}

enum DiscretionInstEnum {
      DiscretionInst_RELATED_TO_DISPLAYED_PRICE = 0;
      DiscretionInst_RELATED_TO_MARKET_PRICE = 1;
      DiscretionInst_RELATED_TO_PRIMARY_PRICE = 2;
      DiscretionInst_RELATED_TO_LOCAL_PRIMARY_PRICE = 3;
      DiscretionInst_RELATED_TO_MIDPOINT_PRICE = 4;
      DiscretionInst_RELATED_TO_LAST_TRADE_PRICE = 5;
      DiscretionInst_RELATED_TO_VWAP = 6;
      DiscretionInst_AVERAGE_PRICE_GUARANTEE = 7;
}

enum DiscretionMoveTypeEnum {
      DiscretionMoveType_FLOATING = 0;
      DiscretionMoveType_FIXED = 1;
}

enum DiscretionOffsetTypeEnum {
      DiscretionOffsetType_PRICE = 0;
      DiscretionOffsetType_BASIS_POINTS = 1;
      DiscretionOffsetType_TICKS = 2;
      DiscretionOffsetType_PRICE_TIER = 3;
}

enum DiscretionLimitTypeEnum {
      DiscretionLimitType_OR_BETTER = 0;
      DiscretionLimitType_STRICT = 1;
      DiscretionLimitType_OR_WORSE = 2;
}

enum DiscretionRoundDirectionEnum {
      DiscretionRoundDirection_MORE_AGGRESSIVE = 0;
      DiscretionRoundDirection_MORE_PASSIVE = 1;
}

enum DiscretionScopeEnum {
      DiscretionScope_LOCAL = 0;
      DiscretionScope_NATIONAL = 1;
      DiscretionScope_GLOBAL = 2;
      DiscretionScope_NATIONAL_EXCLUDING_LOCAL = 3;
}

message DiscretionInstructions {
      optional DiscretionInstEnum discretionInst = 1;
      optional double discretionOffsetValue = 2;
      optional DiscretionMoveTypeEnum discretionMoveType = 3;
      optional DiscretionOffsetTypeEnum discretionOffsetType = 4;
      optional DiscretionLimitTypeEnum discretionLimitType = 5;
      optional DiscretionRoundDirectionEnum discretionRoundDirection = 6;
      optional DiscretionScopeEnum discretionScope = 7;
}

enum StrategyParameterTypeEnum {
```

```
	StrategyParameterType_INT = 0;
	StrategyParameterType_LENGTH = 1;
	StrategyParameterType_NUM_IN_GROUP = 2;
	StrategyParameterType_SEQ_NUM = 3;
	StrategyParameterType_TAG_NUM = 4;
	StrategyParameterType_FLOAT = 5;
	StrategyParameterType_QTY = 6;
	StrategyParameterType_PRICE = 7;
	StrategyParameterType_PRICE_OFFSET = 8;
	StrategyParameterType_AMT = 9;
	StrategyParameterType_PERCENTAGE = 10;
	StrategyParameterType_CHAR = 11;
	StrategyParameterType_BOOLEAN = 12;
	StrategyParameterType_STRING = 13;
	StrategyParameterType_MULTIPLE_CHAR_VALUE = 14;
	StrategyParameterType_CURRENCY = 15;
	StrategyParameterType_EXCHANGE = 16;
	StrategyParameterType_MONTH_YEAR = 17;
	StrategyParameterType_UTCTIMESTAMP = 18;
	StrategyParameterType_UTCTIME_ONLY = 19;
	StrategyParameterType_LOCAL_MKT_DATE = 20;
	StrategyParameterType_UTCDATE_ONLY = 21;
	StrategyParameterType_DATA = 22;
	StrategyParameterType_MULTIPLE_STRING_VALUE = 23;
	StrategyParameterType_COUNTRY = 24;
	StrategyParameterType_LANGUAGE = 25;
	StrategyParameterType_TZTIME_ONLY = 26;
	StrategyParameterType_TZTIMESTAMP = 27;
	StrategyParameterType_TENOR = 28;
}

message StrategyParametersGrp {
	optional string strategyParameterName = 1;
	optional StrategyParameterTypeEnum strategyParameterType = 2;
	optional string strategyParameterValue = 3;
}

enum TrdRegTimestampTypeEnum {
	TrdRegTimestampType_EXECUTION_TIME = 0;
	TrdRegTimestampType_TIME_IN = 1;
	TrdRegTimestampType_TIME_OUT = 2;
	TrdRegTimestampType_BROKER_RECEIPT = 3;
	TrdRegTimestampType_BROKER_EXECUTION = 4;
	TrdRegTimestampType_DESK_RECEIPT = 5;
	TrdRegTimestampType_SUBMISSION_TO_CLEARING = 6;
}

enum DeskTypeEnum {
	DeskType_AGENCY = 0;
	DeskType_ARBITRAGE = 1;
	DeskType_DERIVATIVES = 2;
	DeskType_INTERNATIONAL = 3;
	DeskType_INSTITUTIONAL = 4;
	DeskType_OTHER = 5;
	DeskType_PREFERRED_TRADING = 6;
	DeskType_PROPRIETARY = 7;
	DeskType_PROGRAM_TRADING = 8;
```

```
	DeskType_SALES = 9;
	DeskType_TRADING = 10;
}

enum DeskTypeSourceEnum {
	DeskTypeSource_NASDOATS = 0;
}

enum DeskOrderHandlingInstEnum {
	DeskOrderHandlingInst_ADD_ON_ORDER = 0;
	DeskOrderHandlingInst_ALL_OR_NONE = 1;
	DeskOrderHandlingInst_CASH_NOT_HELD = 2;
	DeskOrderHandlingInst_DIRECTED_ORDER = 3;
	DeskOrderHandlingInst_EXCHANGE_FOR_PHYSICAL_TRANSACTION = 4;
	DeskOrderHandlingInst_FILL_OR_KILL = 5;
	DeskOrderHandlingInst_IMBALANCE_ONLY = 6;
	DeskOrderHandlingInst_IMMEDIATE_OR_CANCEL = 7;
	DeskOrderHandlingInst_LIMIT_ON_OPEN = 8;
	DeskOrderHandlingInst_LIMIT_ON_CLOSE = 9;
	DeskOrderHandlingInst_MARKET_AT_OPEN = 10;
	DeskOrderHandlingInst_MARKET_AT_CLOSE = 11;
	DeskOrderHandlingInst_MARKET_ON_OPEN = 12;
	DeskOrderHandlingInst_MARKET_ON_CLOSE = 13;
	DeskOrderHandlingInst_MINIMUM_QUANTITY = 14;
	DeskOrderHandlingInst_NOT_HELD = 15;
	DeskOrderHandlingInst_OVER_THE_DAY = 16;
	DeskOrderHandlingInst_PEGGED = 17;
	DeskOrderHandlingInst_RESERVE_SIZE_ORDER = 18;
	DeskOrderHandlingInst_STOP_STOCK_TRANSACTION = 19;
	DeskOrderHandlingInst_SCALE = 20;
	DeskOrderHandlingInst_TIME_ORDER = 21;
	DeskOrderHandlingInst_TRAILING_STOP = 22;
	DeskOrderHandlingInst_WORK = 23;
}

message TrdRegTimestamps {
	optional uint64 trdRegTimestamp = 1;
	optional TrdRegTimestampTypeEnum trdRegTimestampType = 2;
	optional string trdRegTimestampOrigin = 3;
	optional DeskTypeEnum deskType = 4;
	optional DeskTypeSourceEnum deskTypeSource = 5;
	optional DeskOrderHandlingInstEnum deskOrderHandlingInst = 6;
}

message StandardTrailer {
	optional uint32 signatureLength = 1;
	optional bytes signature = 2;
	optional string checkSum = 3;
}

enum AcctIdSourceEnum {
	AcctIdSource_BIC = 0;
	AcctIdSource_SID_CODE = 1;
	AcctIdSource_TFM = 2;
	AcctIdSource_OMGEO = 3;
	AcctIdSource_DTCCCODE = 4;
	AcctIdSource_OTHER = 5;
```

```
}

enum AccountTypeEnum {
      AccountType_CARRIED_CUSTOMER_SIDE = 0;
      AccountType_CARRIED_NON_CUSTOMER_SIDE = 1;
      AccountType_HOUSE_TRADER = 2;
      AccountType_FLOOR_TRADER = 3;
      AccountType_CARRIED_NON_CUSTOMER_SIDE_CROSS_MARGINED = 4;
      AccountType_HOUSE_TRADER_CROSS_MARGINED = 5;
      AccountType_JOINT_BACK_OFFICE_ACCOUNT = 6;
}

enum DayBookingInstEnum {
      DayBookingInst_AUTO = 0;
      DayBookingInst_SPEAK_WITH_ORDER_INITIATOR_BEFORE_BOOKING = 1;
      DayBookingInst_ACCUMULATE = 2;
}

enum BookingUnitEnum {
      BookingUnit_EACH_PARTIAL_EXECUTION_IS_ABOOKABLE_UNIT = 0;
      BookingUnit_AGGREGATE_PARTIAL_EXECUTIONS_ON_THIS_ORDER = 1;
      BookingUnit_AGGREGATE_EXECUTIONS_FOR_THIS_SYMBOL = 2;
}

enum PreallocMethodEnum {
      PreallocMethod_PRO_RATA = 0;
      PreallocMethod_DO_NOT_PRO_RATA = 1;
}

enum SettlTypeEnum {
      SettlType_REGULAR = 0;
      SettlType_CASH = 1;
      SettlType_NEXT_DAY = 2;
      SettlType_TPLUS2 = 3;
      SettlType_TPLUS3 = 4;
      SettlType_TPLUS4 = 5;
      SettlType_FUTURE = 6;
      SettlType_WHEN_AND_IF_ISSUED = 7;
      SettlType_SELLERS_OPTION = 8;
      SettlType_TPLUS5 = 9;
      SettlType_BROKEN_DATE = 10;
      SettlType_FX_SPOT_NEXT_SETTLEMENT = 11;
}

enum CashMarginEnum {
      CashMargin_CASH = 0;
      CashMargin_MARGIN_OPEN = 1;
      CashMargin_MARGIN_CLOSE = 2;
}

enum ClearingFeeIndicatorEnum {
      ClearingFeeIndicator_FIRST_YEAR_DELEGATE = 0;
      ClearingFeeIndicator_SECOND_YEAR_DELEGATE = 1;
      ClearingFeeIndicator_THIRD_YEAR_DELEGATE = 2;
      ClearingFeeIndicator_FOURTH_YEAR_DELEGATE = 3;
      ClearingFeeIndicator_FIFTH_YEAR_DELEGATE = 4;
      ClearingFeeIndicator_SIXTH_YEAR_DELEGATE = 5;
```

```
      ClearingFeeIndicator_CBOEMEMBER = 6;
      ClearingFeeIndicator_NON_MEMBER_AND_CUSTOMER = 7;
      ClearingFeeIndicator_EQUITY_MEMBER_AND_CLEARING_MEMBER = 8;
      ClearingFeeIndicator_FULL_AND_ASSOCIATE_MEMBER = 9;
      ClearingFeeIndicator_FIRMS106HAND106J = 10;
      ClearingFeeIndicator_GIM = 11;
      ClearingFeeIndicator_LESSEE106FEMPLOYEES = 12;
      ClearingFeeIndicator_ALL_OTHER_OWNERSHIP_TYPES = 13;
}

enum HandlInstEnum {
      HandlInst_AUTOMATED_EXECUTION_NO_INTERVENTION = 0;
      HandlInst_AUTOMATED_EXECUTION_INTERVENTION_OK = 1;
      HandlInst_MANUAL_ORDER = 2;
}

enum ExecInstEnum {
      ExecInst_STAY_ON_OFFER_SIDE = 0;
      ExecInst_NOT_HELD = 1;
      ExecInst_WORK = 2;
      ExecInst_GO_ALONG = 3;
      ExecInst_OVER_THE_DAY = 4;
      ExecInst_HELD = 5;
      ExecInst_PARTICIPATE_DO_NOT_INITIATE = 6;
      ExecInst_STRICT_SCALE = 7;
      ExecInst_TRY_TO_SCALE = 8;
      ExecInst_STAY_ON_BID_SIDE = 9;
      ExecInst_NO_CROSS = 10;
      ExecInst_OKTO_CROSS = 11;
      ExecInst_CALL_FIRST = 12;
      ExecInst_PERCENT_OF_VOLUME = 13;
      ExecInst_DO_NOT_INCREASE = 14;
      ExecInst_DO_NOT_REDUCE = 15;
      ExecInst_ALL_OR_NONE = 16;
      ExecInst_REINSTATE_ON_SYSTEM_FAILURE = 17;
      ExecInst_INSTITUTIONS_ONLY = 18;
      ExecInst_REINSTATE_ON_TRADING_HALT = 19;
      ExecInst_CANCEL_ON_TRADING_HALT = 20;
      ExecInst_LAST_PEG = 21;
      ExecInst_MID_PRICE_PEG = 22;
      ExecInst_NON_NEGOTIABLE = 23;
      ExecInst_OPENING_PEG = 24;
      ExecInst_MARKET_PEG = 25;
      ExecInst_CANCEL_ON_SYSTEM_FAILURE = 26;
      ExecInst_PRIMARY_PEG = 27;
      ExecInst_SUSPEND = 28;
      ExecInst_FIXED_PEG_TO_LOCAL_BEST_BID_OR_OFFER_AT_TIME_OF_ORDER = 29;
      ExecInst_CUSTOMER_DISPLAY_INSTRUCTION = 30;
      ExecInst_NETTING = 31;
      ExecInst_PEG_TO_VWAP = 32;
      ExecInst_TRADE_ALONG = 33;
      ExecInst_TRY_TO_STOP = 34;
      ExecInst_CANCEL_IF_NOT_BEST = 35;
      ExecInst_TRAILING_STOP_PEG = 36;
      ExecInst_STRICT_LIMIT = 37;
      ExecInst_IGNORE_PRICE_VALIDITY_CHECKS = 38;
      ExecInst_PEG_TO_LIMIT_PRICE = 39;
```

```
      ExecInst_WORK_TO_TARGET_STRATEGY = 40;
      ExecInst_INTERMARKET_SWEEP = 41;
      ExecInst_EXTERNAL_ROUTING_ALLOWED = 42;
      ExecInst_EXTERNAL_ROUTING_NOT_ALLOWED = 43;
      ExecInst_IMBALANCE_ONLY = 44;
      ExecInst_SINGLE_EXECUTION_REQUESTED_FOR_BLOCK_TRADE = 45;
      ExecInst_BEST_EXECUTION = 46;
      ExecInst_SUSPEND_ON_SYSTEM_FAILURE = 47;
      ExecInst_SUSPEND_ON_TRADING_HALT = 48;
      ExecInst_REINSTATE_ON_CONNECTION_LOSS = 49;
      ExecInst_CANCEL_ON_CONNECTION_LOSS = 50;
      ExecInst_SUSPEND_ON_CONNECTION_LOSS = 51;
      ExecInst_RELEASE_FROM_SUSPENSION = 52;
      ExecInst_EXECUTE_AS_DELTA_NEUTRAL = 53;
      ExecInst_EXECUTE_AS_DURATION_NEUTRAL = 54;
      ExecInst_EXECUTE_AS_FX_NEUTRAL = 55;
}

enum ExDestinationIdSourceEnum {
      ExDestinationIdSource_BIC = 0;
      ExDestinationIdSource_GENERAL_IDENTIFIER = 1;
      ExDestinationIdSource_PROPRIETARY = 2;
      ExDestinationIdSource_ISO_COUNTRY_CODE = 3;
      ExDestinationIdSource_MIC = 4;
}

enum ProcessCodeEnum {
      ProcessCode_REGULAR = 0;
      ProcessCode_SOFT_DOLLAR = 1;
      ProcessCode_STEP_IN = 2;
      ProcessCode_STEP_OUT = 3;
      ProcessCode_SOFT_DOLLAR_STEP_IN = 4;
      ProcessCode_SOFT_DOLLAR_STEP_OUT = 5;
      ProcessCode_PLAN_SPONSOR = 6;
}

enum SideEnum {
      Side_BUY = 0;
      Side_SELL = 1;
      Side_BUY_MINUS = 2;
      Side_SELL_PLUS = 3;
      Side_SELL_SHORT = 4;
      Side_SELL_SHORT_EXEMPT = 5;
      Side_UNDISCLOSED = 6;
      Side_CROSS = 7;
      Side_CROSS_SHORT = 8;
      Side_CROSS_SHORT_EXEMPT = 9;
      Side_AS_DEFINED = 10;
      Side_OPPOSITE = 11;
      Side_SUBSCRIBE = 12;
      Side_REDEEM = 13;
      Side_LEND = 14;
      Side_BORROW = 15;
}

enum LocateReqdEnum {
      LocateReqd_NO = 0;
```

```
      LocateReqd_YES = 1;
}

enum QtyTypeEnum {
      QtyType_UNITS = 0;
      QtyType_CONTRACTS = 1;
      QtyType_UNITS_OF_MEASURE_PER_TIME_UNIT = 2;
}

enum OrdTypeEnum {
      OrdType_MARKET = 0;
      OrdType_LIMIT = 1;
      OrdType_STOP = 2;
      OrdType_STOP_LIMIT = 3;
      OrdType_MARKET_ON_CLOSE = 4;
      OrdType_WITH_OR_WITHOUT = 5;
      OrdType_LIMIT_OR_BETTER = 6;
      OrdType_LIMIT_WITH_OR_WITHOUT = 7;
      OrdType_ON_BASIS = 8;
      OrdType_ON_CLOSE = 9;
      OrdType_LIMIT_ON_CLOSE = 10;
      OrdType_FOREX_MARKET = 11;
      OrdType_PREVIOUSLY_QUOTED = 12;
      OrdType_PREVIOUSLY_INDICATED = 13;
      OrdType_FOREX_LIMIT = 14;
      OrdType_FOREX_SWAP = 15;
      OrdType_FOREX_PREVIOUSLY_QUOTED = 16;
      OrdType_FUNARI = 17;
      OrdType_MARKET_IF_TOUCHED = 18;
      OrdType_MARKET_WITH_LEFT_OVER_AS_LIMIT = 19;
      OrdType_PREVIOUS_FUND_VALUATION_POINT = 20;
      OrdType_NEXT_FUND_VALUATION_POINT = 21;
      OrdType_PEGGED = 22;
      OrdType_COUNTER_ORDER_SELECTION = 23;
}

enum PriceTypeEnum {
      PriceType_PERCENTAGE = 0;
      PriceType_PER_UNIT = 1;
      PriceType_FIXED_AMOUNT = 2;
      PriceType_DISCOUNT = 3;
      PriceType_PREMIUM = 4;
      PriceType_SPREAD = 5;
      PriceType_TEDPRICE = 6;
      PriceType_TEDYIELD = 7;
      PriceType_YIELD = 8;
      PriceType_FIXED_CABINET_TRADE_PRICE = 9;
      PriceType_VARIABLE_CABINET_TRADE_PRICE = 10;
      PriceType_PRODUCT_TICKS_IN_HALFS = 11;
      PriceType_PRODUCT_TICKS_IN_FOURTHS = 12;
      PriceType_PRODUCT_TICKS_IN_EIGHTS = 13;
      PriceType_PRODUCT_TICKS_IN_SIXTEENTHS = 14;
      PriceType_PRODUCT_TICKS_IN_THIRTY_SECONDS = 15;
      PriceType_PRODUCT_TICKS_IN_SIXTY_FORTHS = 16;
      PriceType_PRODUCT_TICKS_IN_ONE_TWENTY_EIGHTS = 17;
}
```

```
enum PriceProtectionScopeEnum {
      PriceProtectionScope_NONE = 0;
      PriceProtectionScope_LOCAL = 1;
      PriceProtectionScope_NATIONAL = 2;
      PriceProtectionScope_GLOBAL = 3;
}

enum SolicitedFlagEnum {
      SolicitedFlag_WAS_NOT_SOLICITED = 0;
      SolicitedFlag_WAS_SOLICITED = 1;
}

enum TimeInForceEnum {
      TimeInForce_DAY = 0;
      TimeInForce_GOOD_TILL_CANCEL = 1;
      TimeInForce_AT_THE_OPENING = 2;
      TimeInForce_IMMEDIATE_OR_CANCEL = 3;
      TimeInForce_FILL_OR_KILL = 4;
      TimeInForce_GOOD_TILL_CROSSING = 5;
      TimeInForce_GOOD_TILL_DATE = 6;
      TimeInForce_AT_THE_CLOSE = 7;
      TimeInForce_GOOD_THROUGH_CROSSING = 8;
      TimeInForce_AT_CROSSING = 9;
}

enum GtBookingInstEnum {
      GtBookingInst_BOOK_OUT_ALL_TRADES_ON_DAY_OF_EXECUTION = 0;
      GtBookingInst_ACCUMULATE_UNTIL_FILLED_OR_EXPIRED = 1;
      GtBookingInst_ACCUMULATE_UNTIL_VERBALLLY_NOTIFIED_OTHERWISE = 2;
}

enum OrderCapacityEnum {
      OrderCapacity_AGENCY = 0;
      OrderCapacity_PROPRIETARY = 1;
      OrderCapacity_INDIVIDUAL = 2;
      OrderCapacity_PRINCIPAL = 3;
      OrderCapacity_RISKLESS_PRINCIPAL = 4;
      OrderCapacity_AGENT_FOR_OTHER_MEMBER = 5;
}

enum OrderRestrictionsEnum {
      OrderRestrictions_PROGRAM_TRADE = 0;
      OrderRestrictions_INDEX_ARBITRAGE = 1;
      OrderRestrictions_NON_INDEX_ARBITRAGE = 2;
      OrderRestrictions_COMPETING_MARKET_MAKER = 3;
      OrderRestrictions_ACTING_AS_MARKET_MAKER_OR_SPECIALIST_IN_SECURITY = 4;
      OrderRestrictions_ACTING_AS_MARKET_MAKER_OR_SPECIALIST_IN_UNDERLYING = 5;
      OrderRestrictions_FOREIGN_ENTITY = 6;
      OrderRestrictions_EXTERNAL_MARKET_PARTICIPANT = 7;
      OrderRestrictions_EXTERNAL_INTER_CONNECTED_MARKET_LINKAGE = 8;
      OrderRestrictions_RISKLESS_ARBITRAGE = 9;
      OrderRestrictions_ISSUER_HOLDING = 10;
      OrderRestrictions_ISSUE_PRICE_STABILIZATION = 11;
      OrderRestrictions_NON_ALGORITHMIC = 12;
      OrderRestrictions_ALGORITHMIC = 13;
      OrderRestrictions_CROSS = 14;
}
```

```
enum CustOrderCapacityEnum {
      CustOrderCapacity_MEMBER_TRADING_FOR_THEIR_OWN_ACCOUNT = 0;
      CustOrderCapacity_CLEARING_FIRM_TRADING_FOR_ITS_PROPRIETARY_ACCOUNT = 1;
      CustOrderCapacity_MEMBER_TRADING_FOR_ANOTHER_MEMBER = 2;
      CustOrderCapacity_ALL_OTHER = 3;
}

enum ForexReqEnum {
      ForexReq_DO_NOT_EXECUTE_FOREX_AFTER_SECURITY_TRADE = 0;
      ForexReq_EXECUTE_FOREX_AFTER_SECURITY_TRADE = 1;
}

enum BookingTypeEnum {
      BookingType_REGULAR_BOOKING = 0;
      BookingType_CFD = 1;
      BookingType_TOTAL_RETURN_SWAP = 2;
}

enum PositionEffectEnum {
      PositionEffect_CLOSE = 0;
      PositionEffect_FIFO = 1;
      PositionEffect_OPEN = 2;
      PositionEffect_ROLLED = 3;
      PositionEffect_CLOSE_BUT_NOTIFY_ON_OPEN = 4;
      PositionEffect_DEFAULT = 5;
}

enum CoveredOrUncoveredEnum {
      CoveredOrUncovered_COVERED = 0;
      CoveredOrUncovered_UNCOVERED = 1;
}

enum TargetStrategyEnum {
      TargetStrategy_VWAP = 0;
      TargetStrategy_PARTICIPATE = 1;
      TargetStrategy_MININIZE_MARKET_IMPACT = 2;
}

enum CancellationRightsEnum {
      CancellationRights_YES = 0;
      CancellationRights_NO_EXECUTION_ONLY = 1;
      CancellationRights_NO_WAIVER_AGREEMENT = 2;
      CancellationRights_NO_INSTITUTIONAL = 3;
}

enum MoneyLaunderingStatusEnum {
      MoneyLaunderingStatus_PASSED = 0;
      MoneyLaunderingStatus_NOT_CHECKED = 1;
      MoneyLaunderingStatus_EXEMPT_BELOW_LIMIT = 2;
      MoneyLaunderingStatus_EXEMPT_MONEY_TYPE = 3;
      MoneyLaunderingStatus_EXEMPT_AUTHORISED = 4;
}

enum CustOrderHandlingInstEnum {
      CustOrderHandlingInst_ADD_ON_ORDER = 0;
      CustOrderHandlingInst_ALL_OR_NONE = 1;
```

```
	CustOrderHandlingInst_CASH_NOT_HELD = 2;
	CustOrderHandlingInst_DIRECTED_ORDER = 3;
	CustOrderHandlingInst_EXCHANGE_FOR_PHYSICAL_TRANSACTION = 4;
	CustOrderHandlingInst_FILL_OR_KILL = 5;
	CustOrderHandlingInst_IMBALANCE_ONLY = 6;
	CustOrderHandlingInst_IMMEDIATE_OR_CANCEL = 7;
	CustOrderHandlingInst_LIMIT_ON_OPEN = 8;
	CustOrderHandlingInst_LIMIT_ON_CLOSE = 9;
	CustOrderHandlingInst_MARKET_AT_OPEN = 10;
	CustOrderHandlingInst_MARKET_AT_CLOSE = 11;
	CustOrderHandlingInst_MARKET_ON_OPEN = 12;
	CustOrderHandlingInst_MARKET_ON_CLOSE = 13;
	CustOrderHandlingInst_MINIMUM_QUANTITY = 14;
	CustOrderHandlingInst_NOT_HELD = 15;
	CustOrderHandlingInst_OVER_THE_DAY = 16;
	CustOrderHandlingInst_PEGGED = 17;
	CustOrderHandlingInst_RESERVE_SIZE_ORDER = 18;
	CustOrderHandlingInst_STOP_STOCK_TRANSACTION = 19;
	CustOrderHandlingInst_SCALE = 20;
	CustOrderHandlingInst_TIME_ORDER = 21;
	CustOrderHandlingInst_TRAILING_STOP = 22;
	CustOrderHandlingInst_WORK = 23;
}

enum OrderHandlingInstSourceEnum {
	OrderHandlingInstSource_NASDOATS = 0;
}

enum RefOrderIdSourceEnum {
	RefOrderIdSource_SECONDARY_ORDER_ID = 0;
	RefOrderIdSource_ORDER_ID = 1;
	RefOrderIdSource_MD_ENTRY_ID = 2;
	RefOrderIdSource_QUOTE_ENTRY_ID = 3;
	RefOrderIdSource_ORIGINAL_ORDER_ID = 4;
}

message AcctIdSourceUnion {
	optional AcctIdSourceEnum acctIdSource = 1;
	optional uint32 acctIdSourceUint32 = 2;
}

message SettlTypeUnion {
	optional SettlTypeEnum settlType = 1;
	optional Tenor settlTypeTenor = 2;
}

message TargetStrategyUnion {
	optional TargetStrategyEnum targetStrategy = 1;
	optional uint32 targetStrategyUint32 = 2;
}

message NewOrderSingle {
	optional StandardHeader standardHeader = 1;
	optional string clOrdId = 2;
	optional string secondaryClOrdId = 3;
	optional string clOrdLinkId = 4;
	repeated Parties parties = 5;
```

```
optional uint32 tradeOriginationDate = 6;
optional uint32 tradeDate = 7;
optional string account = 8;
optional AcctIdSourceUnion acctIdSource = 9;
optional AccountTypeEnum accountType = 10;
optional DayBookingInstEnum dayBookingInst = 11;
optional BookingUnitEnum bookingUnit = 12;
optional PreallocMethodEnum preallocMethod = 13;
optional string allocId = 14;
repeated PreAllocGrp preAllocGrp = 15;
optional SettlTypeUnion settlType = 16;
optional uint32 settlDate = 17;
optional CashMarginEnum cashMargin = 18;
optional ClearingFeeIndicatorEnum clearingFeeIndicator = 19;
optional HandlInstEnum handlInst = 20;
repeated ExecInstEnum execInst = 21 [packed = true];
optional UDecimal64E0 minQty = 22;
optional UDecimal64E0 matchIncrement = 23;
optional sint64 maxPriceLevels = 24;
optional DisplayInstruction displayInstruction = 25;
optional UDecimal64E0 maxFloor = 26;
optional string exDestination = 27;
optional ExDestinationIdSourceEnum exDestinationIdSource = 28;
repeated TrdgSesGrp trdgSesGrp = 29;
optional ProcessCodeEnum processCode = 30;
optional Instrument instrument = 31;
optional FinancingDetails financingDetails = 32;
repeated UndInstrmtGrp undInstrmtGrp = 33;
optional UDecimal64EMinus6 prevClosePx = 34;
optional SideEnum side = 35;
optional LocateReqdEnum locateReqd = 36;
optional uint64 transactTime = 37;
repeated Stipulations stipulations = 38;
optional QtyTypeEnum qtyType = 39;
optional OrderQtyData orderQtyData = 40;
optional OrdTypeEnum ordType = 41;
optional PriceTypeEnum priceType = 42;
optional UDecimal64EMinus6 price = 43;
optional PriceProtectionScopeEnum priceProtectionScope = 44;
optional UDecimal64EMinus6 stopPx = 45;
optional TriggeringInstruction triggeringInstruction = 46;
optional SpreadOrBenchmarkCurveData spreadOrBenchmarkCurveData = 47;
optional YieldData yieldData = 48;
optional string currency = 49;
optional string complianceId = 50;
optional SolicitedFlagEnum solicitedFlag = 51;
optional string ioiId = 52;
optional string quoteId = 53;
optional TimeInForceEnum timeInForce = 54;
optional uint64 effectiveTime = 55;
optional uint32 expireDate = 56;
optional uint64 expireTime = 57;
optional GtBookingInstEnum gtBookingInst = 58;
optional CommissionData commissionData = 59;
optional OrderCapacityEnum orderCapacity = 60;
repeated OrderRestrictionsEnum orderRestrictions = 61 [packed = true];
optional bool preTradeAnonymity = 62;
```

```
    optional CustOrderCapacityEnum custOrderCapacity = 63;
    optional ForexReqEnum forexReq = 64;
    optional string settlCurrency = 65;
    optional BookingTypeEnum bookingType = 66;
    optional string text = 67;
    optional uint32 encodedTextLen = 68;
    optional bytes encodedText = 69;
    optional uint32 settlDate2 = 70;
    optional UDecimal64E0 orderQty2 = 71;
    optional UDecimal64EMinus6 price2 = 72;
    optional PositionEffectEnum positionEffect = 73;
    optional CoveredOrUncoveredEnum coveredOrUncovered = 74;
    optional UDecimal64E0 maxShow = 75;
    optional PegInstructions pegInstructions = 76;
    optional DiscretionInstructions discretionInstructions = 77;
    optional TargetStrategyUnion targetStrategy = 78;
    repeated StrategyParametersGrp strategyParametersGrp = 79;
    optional string targetStrategyParameters = 80;
    optional UDecimal64EMinus2 participationRate = 81;
    optional CancellationRightsEnum cancellationRights = 82;
    optional MoneyLaunderingStatusEnum moneyLaunderingStatus = 83;
    optional string registId = 84;
    optional string designation = 85;
    optional bool manualOrderIndicator = 86;
    optional bool custDirectedOrder = 87;
    optional string receivedDeptId = 88;
    optional CustOrderHandlingInstEnum custOrderHandlingInst = 89;
    optional OrderHandlingInstSourceEnum orderHandlingInstSource = 90;
    repeated TrdRegTimestamps trdRegTimestamps = 91;
    optional string refOrderId = 92;
    optional RefOrderIdSourceEnum refOrderIdSource = 93;
    optional StandardTrailer standardTrailer = 94;
}
```