Encoding FIX Using Google Protocol Buffers

Release Candidate 2

Technical Proposal

April 25, 2014

v0.6

Proposal Status: Public Comment

For Global Technical Committee Governance Internal Use Only

Submission Date	Dec 18, 2013	Control Number	
Submission Status	Submitted	Ratified Date	
Primary Contact Person	Greg Malatestinic	Release Identifier	

r0.2

DISCLAIMER

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

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

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

No proprietary or ownership interest of any kind is granted with respect to the FIX Protocol (or any rights therein).

Table of Contents

Document History

Revision	Date	Author	Revision Comments
v0.1	11/21/13	Greg Malatestinic, Jordan & Jordan Sara Rosen, EBS Joe Wood, Credit Suisse	Initial draft
v0.2	11/22/13	Greg Malatestinic, Jordan & Jordan Sara Rosen, EBS Joe Wood, Credit Suisse	For review by HPWG
v0.3	11/25/13	Greg Malatestinic, Jordan & Jordan Sara Rosen, EBS Joe Wood, Credit Suisse	Updated based on feedback from HPWG review.
v0.4	12/18/13	Fred Malabre	Updates based on GTC review.
v0.5	1/27/14	Greg Malatestinic	Further updates to formally align the documents with those of the other specs.
V0.6	4/25/14	Greg Malatestinic	Update to align with message encoding header specified in user guide.

1 Introduction

The High Performance Working Group was formed with the goal of improving the fit-for-purposefulness of FIX for high performance.

Recent improvements in the speed of hardware, software, and network connections (such as in co-location solutions) are putting pressure on the FIX protocol and highlighting some inefficiencies of the current version of the protocol (e.g., excessive echoing of input values, inefficient encoding). New financial applications such as high-frequency trading and market data feeds pose new performance requirements. In recent years, several financial organizations have avoided the performance limitations of FIX and introduced new proprietary protocols that are optimized for speed. These proprietary interfaces have been offered, sometimes along with a FIX interface, to support high-speed transactions and/or data feeds.

The current performance limitations of FIX can be removed by making changes and additions at multiple levels of the protocol. At the *application* level, there is a need to define less-verbose versions of some FIX messages and to streamline the message flow. At the *presentation* level, there is a need to provide new encodings that are faster and more compact than the traditional Tag=Value encoding of FIX. At the *session* level, there is a need to specify a new lightweight session protocol with basic recovery options. The High Performance Working Group is drafting a set of specifications and guideline documents to address all these aspects.

This proposal entails the use of a Google Protocol Buffers to produce fast and compact encodings of FIX messages.

The document describes the changes adopted as a result of the RC1 community feedback period.

1.1 Authors

Name	Affiliation	Contact	Role
Greg Malatestinic	Jordan & Jordan	gmala@jandj.com	GPB co-lead
Sara Rosen	EBS	sara.rosen@ebs.com	GPB co-lead
Joe Wood	Credit Suisse	joe.wood@credit-suisse.com	

2 Requirements

2.1 Business Requirements

The purpose of the RC2 release of the Google Protocol Buffers Encoding specification is to address the community feedback received during the public comment period on the RC1 proposal.

Changes were made to optimize performance, improve FIX repository cross-referencing, provide programmatic access to encoding metadata, and to better organize the .proto files.

2.1.1 Naming Conventions

In RC1, the GPB encoding adopted a camel-case naming convention for FIX fields. It was recommended that the GPB encoding comply with Google Protocol Buffers field naming style guidelines. This will allow the code generation tools to produce code compliant with the naming conventions of the target language. (C++, Java, .Net, etc.)

2.1.2 Programmatic Access to FIX Tag Cross Reference

In RC1, the FIX tag was carried in the comments of the .proto template. This supported human-access, but did not allow third-party tools programmatic access to these values. Suggestion was to utilize Google Custom Options to add metadata to the .proto to cross reference FIX tag numbers and other metadata from the FIX repository.

2.1.3 Proto file organization

The RC1 specification did not address the organization of .proto files into packages. Recommendation was to organize .proto files into packages which mirror the FIX categories.

2.1.4 Decimal numeric encoding

RC1 specification called for generation of an SDecimal<N>E message consisting of a required mantissa and an optional exponent to represent fixed point and floating point decimal fields. Due to overhead associated with embedded messages, recommendation is to use in-line fields or custom options to carry the exponent metadata.

2.1.5 Varint performance overhead

RC1 specification utilized GPB varints for most scalar values due to its optimized use of message bandwidth. Proof of concept testing indicated a significant performance overhead for varints (e.g. uint32) when compared to the GPB fixed-length datatypes (e.g. sfixed32). Recommendation was to optimize for performance rather than bandwidth.

2.1.6 Integer encoding of decimal strings

Automated generation of proto files from FIX Repository is supported by a set of encoding attributes which are used in conjunction with the FIX datatype to determine the optimal GPB field mapping. Examples are minValue, maxValue, numBits, isBinaryFloat.

It was suggested to introduce an additional encoding attribute for FIX strings which are constrained to integer values to cause these fields to be mapped to GPB integers in the proto schema.

2.1.7 Proto field ordering

RC1 specification did not address the issue of field ordering upon addition of new fields to the FIX Repository. When new fields are added, auto-generation of .proto files would cause previously allocated field numbers to be overridden, resulting in a breaking change to the schema. Release Candidate 2 introduces a field ordering convention.

2.2 Technical Requirements

2.2.1 Naming Conventions

In RC1, the GPB encoding adopted a camel-case naming convention for FIX fields. It was recommended that the GPB encoding comply with Google Protocol Buffers field naming style guidelines. This will allow the code generation tools to produce code compliant with the naming conventions of the target language. (C++, Java, .Net, etc.)

Determination:

It was decided to change the naming conventions utilized by the GPB encoding to conform to Google Protocol Buffers Style Guidelines. Field names in .proto files are lower-case underscore-delimited, e.g. transact_time, account_type, cl_ord_id. Enumerations are upper-case underscore-delimited, e.g. CXL_REJ_REASON_TOO_LATE_TO_CANCEL, CXL_REJ_REASON_UNKNOWN_ORDER.

2.2.2 Programmatic Access to FIX Tag Cross Reference

In RC1, the FIX tag was carried in the comments of the .proto template. This supported human-access, but did not allow third-party tools programmatic access to these values. Suggestion was to utilize Google Custom Options to add metadata to the .proto to cross reference FIX tag numbers and other metadata from the FIX repository.

Determination:

GPB custom options are to be utilized to carry FIX cross reference information (FIX tag number, FIX enumeration value, FIX version added, FIX version deprecated) and encoding attributes (min_value, max_value, is_numeric, is_binary_float, epoch, time_unit, exponent, etc.) to provide programmatic access to this metadata.

Example:

```
message ListOrdGrp {
…
optional sfixed64 stop_px = 27  [(fix.tag)=99, (fix.type)=PRICE, (fix.field_added)=FIX_2_7];
optional sfixed32 stop_px_exponent = 28 [default=-6];
optional string settl_currency = 29 [(fix.tag)=120, (fix.type)=CURRENCY, (meta.minLength)=3, (meta.maxLength)=3,
(fix.field_added)=FIX_4_0];
…
}
```

2.2.3 Proto file organization

RC1 specification did not address the organization of .proto files into packages. Recommendation was to organize .proto files into packages which mirror the FIX categories.

Determination:

GPB .proto files are to be subdivided into packages mirroring the FIX categories. Proto files for categories such as Session and Common which are utilized by multiple schemas are imported utilizing the GPB “import” mechanism. These fields can then be utilized in the imported schema prefixed with the namespace of the imported file.

Example:

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

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

message OrderCancelReject {
        …
        optional string order_id = 4;
        optional Session.StandardHeader standard_header = 5;
        optional Session.StandardTrailer standard_trailer = 6;
        optional string text = 7;
        optional OrdStatusEnum ord_status = 8;
        optional string orig_cl_ord_id = 9;
        …
}
```

2.2.4 Decimal encoding

RC1 specification called for generation of an SDecimal<N>E message consisting of a required mantissa and an optional exponent to represent fixed point and floating point decimal fields. Due to overhead associated with embedded messages, recommendation is to use in-line fields or custom options to carry the exponent metadata.

Determination:

For Release Candidate 2, when an exponent is to be applied to an integer scalar value, an additional GPB field with the name of the original field and concatenated with the string "_exponent" will be generated to carry the value of the exponent.
Additionally, if a static exponent value is known at the time of .proto generation, the static exponent will be encoded as a custom option on the mantissa field, and the value of exponent will appear as the default value of the associated exponent field.

Example 1: Exponent value varies:

```
message EvntGrp {
        …
        optional sfixed64 event_px = 2                  [(fix.tag)=867, (fix.type)=PRICE];
        optional sfixed32 event_px_exponent = 3;
        …
}
```

Example 2: Static exponent value known at time of .proto generation:

```
message EvntGrp {
        …
        optional sfixed64 event_px = 2 [(meta.exponent)=-6,(fix.tag)=867,(fix.type)=PRICE];
        optional sfixed32 event_px_exponent = 3 [default=-6];
        …
}
```


2.2.5 Varint performance overhead

RC1 specification utilized GPB varints for most scalar values due to its optimized use of message bandwidth. Proof of concept testing indicated a significant performance overhead for varints (e.g. uint32) when compared to the GPB fixed-length datatypes (e.g. sfixed32). Recommendation was to optimize for performance rather than bandwidth.

Determination:

Release Candidate 2 specification indicates that all integer fields be mapped to fixed-length GPB scalar fields - fixed32, sfixed32, fixed64, or sfixed64, with choice dependent on the range of possible values for the field.

2.2.6 Integer encoding of decimal strings

Automated generation of proto files from FIX Repository is supported by a set of encoding attributes which are used in conjunction with the FIX datatype to determine the optimal GPB field mapping. Examples are minValue, maxValue, numBits, isBinaryFloat.

It was suggested to introduce an additional encoding attribute for FIX strings which are constrained to integer values to cause these fields to be mapped to GPB integers in the proto schema.

Determination:

Release Candidate 2 introduces a new “isNumeric” encoding attribute applicable to FIX string fields. When the auto-mapper encounters such a field in the FIX repository, it disregards the FIX string datatype and generates an integer field in the proto schema.

This designation is particularly useful for identifiers such as SecurityID, OrderID, PartyID, etc. which can then use integer operations for copying and comparing rather than the more computationally expensive string operations.

2.2.7 Proto field ordering

RC1 specification did not address the issue of field ordering upon addition of new fields to the FIX Repository. When new fields are added, auto-generation of .proto files would cause previously allocated field numbers to be overridden, resulting in a breaking change to the schema.

Determination:

Release Candidate 2 auto-mapping specification makes provisions for ordering of GPB fields according to the following sort criteria:

1. FIX version
2. EP number
3. Alphabetic order of fields added within the same FIX version/EP.

This sort order makes use of the Repository “added” and “addedEP” fields to assure that newly added fields are appended to the end of protobuf messages without overriding the field number of previously existing fields.

2.2.8 Message encoding header

Two fields are added as the message encoding header for GPB messages:

- **Proto ID** – identifier of the GPB schema.
- **Proto Version Number** - the version of the proto file containing the message definition.
- **GPB Message Type** – provides an association with the FIX message type from which the message was generated.

3 Issues and Discussion Points

None

4 References

Reference	Version	Relevance	Normative
Encoding FIX using Google Protocol Buffers - RC1 Specification	Final	Specification as approved for RC1 in June 2013 by the FPL GTC.	
Encoding FIX using Google Protocol Buffers - RC1 User Guide	Final	User guide as approved for RC1 in June 2013 by the FPL GTC.	
Encoding FIX using GPB - Release Candidate 2	RC2 rev0.18	Full specifications based on RC1 with the addition to the technical solutions from this technical proposal.	
Encoding FIX using GPB - Release Candidate 2 - User Guide	RC2 rev0.22	User Guide based on RC1 with the addition to the technical solutions from this technical proposal.	

5 Relevant and Related Standards

Related Standard	Version	Reference location	Relationship	Normative
Google Protocol Buffers Developer Guide		https://developers.google.com/protocol-buffers/docs/proto		

6 Intellectual Property Disclosure

Related Intellection Property	Type of IP (copyright, patent)	IP Owner	Relationship to proposed standard
None			

7 Definitions

Term	Definition
None	

8 Encoding FIX Using Google Protocol Buffers

Revision 0.18 of the full specification for the GPB encoding of FIX is available as “Encoding FIX using GPB – Release Candidate 2.docx”.

Revision 0.22 of the user guide for the GPB encoding of FIX is available as “Encoding FIX using GPB – Release Candidate 2 – User Guide.docx”.

Appendix A - Usage Examples

None

Appendix B – Compliance Strategy

None

