FINANCIAL INFORMATION EXCHANGE PROTOCOL (FIX)

Version 4.4 with Errata 20030618

FIXML Transition and Migration Guide including FIXML 4.4 DTD Version Release Notes

Published in conjunction with FIX .4.4 Errata adjustments as of June 18, 2003

This document provides an overview of the direction being taken to enhance FIXML by providing a road map. The document also contains an overview on how to migrate FIX 4.2 and FIX 4.3 FIXML applications to use the FIX 4.4 FIXML DTD Version.

June 18, 2003

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.

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

PREFACE

The Financial Information eXchange (FIX) effort was initiated in 1992 by a group of institutions and brokers interested in streamlining their trading processes. These firms felt that they, and the industry as a whole, could benefit from efficiencies derived through the electronic communication of indications, orders and executions. The result is FIX, an open message standard controlled by no single entity, that can be structured to match the business requirements of each firm. The benefits are:

* From the business flow perspective, FIX provides institutions, brokers, and other market participants a means of reducing the clutter of unnecessary telephone calls and scraps of paper, and facilitates targeting high quality information to specific individuals.
* For technologists, FIX provides an open standard that leverages the development effort so that they can efficiently create links with a wide range of counter-parties.
* For vendors, FIX provides ready access to the industry, with the incumbent reduction in marketing effort and increase in potential client base.

Openness has been the key to FIX's success. For that reason, while encouraging vendors to participate with the standard, FIX has remained vendor neutral. Similarly, FIX avoids over-standardization. It does not demand a single type of carrier (e.g., it will work with leased lines, frame relay, Internet, etc.), nor a single security protocol. It leaves many of these decisions to the individual firms that are using it. We do expect that, over time, the rules of engagement in these non-standardized areas will converge as technologies mature.

FIX is now used by a variety of firms and vendors. It has clearly emerged as the inter-firm messaging protocol of choice. FIX has grown from its original buyside-to-sellside equity trading roots. It is now used by markets (exchanges, "ECNs", etc) and other market participants. In addition to equities, FIX currently supports four other products: Collective Investment Vehicles (CIVs), Derivatives, Fixed Income, and Foreign Exchange. The process for modifications to the specification is very open with input and feedback encouraged from the community. Those interested in providing input to the protocol are encouraged use the FIX website Discussion section or contact the FIX Global Technical Committee Chairpersons, Scottt Atwell, American Century Investments, (US) 816-340-7053 (scott_atwell@americancentury.com) or Dean Kauffman, TradeWeb LLC, (US) 201-536-5827 (dean.kauffman@tradeweb.com).. **The FIX website at *http://www.fixprotocol.org* is the main source of information, discussion, and notification of FIX-related events.**

We look forward to your participation.

FIX Protocol Ltd June 18, 2003

Introduction

The FIXML language is in a state of transition. It has been four years since the initial release of FIXML. XML technology has advanced considerably in those four years. The convergence across the information technology in the adoption of XML is incredible. Many concurrent efforts are ongoing to exploit XML as the basis for organization to organization communication. Because of this rapid rate of change and the wide scale adoption of XML, the FIX Global Technical Committee started in early 2003 to look to enhance FIXML to make it more usable for FIX applications that are known for requiring high message rates and high bandwidth consumption.

This document provides some background on FIXML and provides a road map for the future direction of FIXML. This document will be expanded as a result of the work of the FIXML Schema Working Group that will start at the end of June 2003.

FIXML Background

The FPL FIXML Working Group began investigating the XML format in 1998 and published a White Paper supporting an evolutionary approach to migrating the FIX Protocol to an XML format. The working group released an intial version of the FIXML DTDs on January 15th, 1999. There are currently DTDs based on FIX Protocol versions 4.1, 4.2 and 4.3. A FIXML Schema based version of FIXML will be provided after the release of FIX 4.4. The FIXML Schema version will be able to provide reduced message size via the use of attributes and contextual abbreviations.

FIXML Highlights

- FIXML is the XML vocabulary for creating FIX messages.
- Uses the same FIX data dictionary and business logic.
- Focuses primarily on the FIX Application Messages and does not provide a session layer.
- Can be encapsulated within the FIX Session Protocol or within another protocol like, MQ Series, TIBCO, SOAP, etc.

FIXML Transition

FIXML was initiated at a time when the only mechanism available to define and validate an XML syntax was the Document Type Definition (DTD) originally created as part of the Standardized General Markup Language (SGML). The DTD provided only minimal ability to define an XML syntax. The following limitations of DTDs determined much of the FIXML implementation;

1. Meta data could not be included in the DTD - so attributes were used for meta data.
2. Attributes could not be "typed" so this restricted datatyping to elements. Many XML syntax's then relied heavily on elements for data, attributes for meta-data. This is the approach taken for FIXML up through the FIX 4.4 Errata 20030618 release.

Since the initial release of FIXML in 1999, XML technology has advanced. The primary advancement has been in the area of standards that are used to define XML based languages. First among these is XML Schema - which has been adopted as a standard by the W3C. XML Schema addresses many of the limitations in DTDs, including:

1. Advanced datatyping, including datatyping for attributes.
2. Ability to include user defined meta data in addition to standardized annotation and documentation.
3. XML Schema is written in XML, permitting manipulation by XML tools, such as XSLT, Xpath, etc.

XML Usage within finance industry

Original approaches to defining XML languages for messaging primarily relied heavily on using elements, as was stated earlier this was due to limitations in the defining language, DTD.

One of the issues for high volume messaging applications was resultant message size from the inherent verbosity in having to repeat the element name as follows:

<Order>

<ClOrdID>AAA-0001</ClOrdID>

</Order>

Implementations that were sensitive to bandwidth and throughput requirements optimized their applications to use attributes within message elements to reduce overall message size:

<Order ClOrdID="AAA-0001"/>

The two messages are functionally equal - the message size however is significantly reduced in the version that uses attributes.

Much was given up for this optimization given the limitation of DTDs, with the advent of XML Schema, these limitations have been removed.

Current Direction

This has been in the works and much prototyping work has been done in conjunction with the FIA in the derivatives post trade space. There are a couple of major applications that will be going into production in the next several months using FIXML - both at the Options Clearing Corporation and the Chicago Mercantile Exchange.

The main things we are doing in the FIXML schema are:

Contextual Abbreviation - very short tag names where the identity of the tag is implied by the message that contains it. For instance - we don't need the "MD" prefixes on market data tags, or "TRD" prefixes on fields in the Trade Capture report.

Moving to attributes from elements. The number one issue that has prohibited adoption of FIXML is message size. We are addressing that in a couple of ways, the most important being moving from elements to attributes wherever possible. Schema gives us the ability to strongly type attributes - this was not possible with DTDs. This approach was based upon successful XML implementations at NYMEX and Lind-Waldock (a futures - FCM).

The resulting message size reduction about 75% smaller message sizes resulting from abbreviations and moving to attribute based messages. As an aside, the readability of the message (though not the primary goal of FIXML) actually has improved as a result of this optimization.

ISO Compatibility

A considerable amount of work was done in FIX 4.3 to work toward compliance with the overall ISO 15022 Working Group. The FIX 4.3 and FIX 4.4 pre-trade messages are based upon the reverse engineering work done by the the ISO 15022 Pre-Trade Working Group. FIX converted to several ISO standards for fields, such as BIC, MIC, ISO Currency Codes, and the CFI.

Work continues with the ISO/XML working group, which is part of the FPL Global Technical Committee.

FIXML Road Map

This section is especially important for those with existing or planned FIXML applications.

June 2003 FIXML 4.4 DTD Version

The FIXML 4.4 DTD that was released as part of the FIX 4.4 Errata 20030618 is based upon new design rules designed to reduce message size. Many of the optimizations applied in the FIXML 4.4 DTD version were conceived as part of the ISO/XML effort.

1. Eliminating extra elements that were used as containers. For instance - each repeating group had an outer list element, with a nested element for the group. The FIXML message itself had a FIXMLMessage element and an ApplicationMessage element before getting to the actual message element, such as Order. We have optimized these from the FIXML structure.
2. Applied a standard abbreviation algorithm using a dictionary that can be also overridden in specific cases. This again - greatly reduces message size.

August 2003 FIXML 4.4 Schema Representation

The FIXML 4.4 Schema representation is not intended to be backwardly compatible with the FIXML 4.4 DTD version. The Schema version will exploit the use of attributes and contextual abbreviations.

Released at that time to assist in the transition to the FIXML 4.4 Schema will be a FIX 4.4 Backward Compatible Schema - which will represent the FIXML 4.4 DTD version of FIXML. This is being provided to assist in providing interoperability and tools for migration to existing FIXML users. Other translations will be made available should the FIX user community request them, such as XSLT to convert from FIX 4.2, FIX 4.3 to FIX 4.4 Schema representation.

The FIXML Schema Working Group is being formed June 24th, 2003 and will continue the work started by the FIX Global Derivatives Committee in conjunction with the Futures Industry Association (FIA) Standards Working Group. Included in this effort will be inclusion of work completed as part of the ISO/XML working group and the FpML (Financial Products Markup Language) standard maintained by ISDA. The goal of this working group is to provide a highly optimized XML representation of FIX that can interoperate with FpML and builds upon the work done as part of the ISO/XML initiative.

Why release the FIX 4.4 DTD version at all - why not just jump to the schema directly?

There are a couple of answers to this question. First, was timing - it was not possible to complete the work required for the schema in concert with the FIX 4.4 release. Secondly, the GTC felt that the DTD

version would serve as a bridge for existing 4.3 DTD users to the new schema approach. It will at least give those who rely on DTD technology to have a way to use enhancements available in FIX 4.4.

FIXML 4.4 DTD Version Release Notes

The following major changes were introduced into the FIX4.4 FIXML DTD Version as part of the optimization of FIXML.

Modification to FIXML Outer Elements

The Application messages and the optional Header element are now contained directly within the <FIXML> element. The <FIXMLMessage> and the <ApplicationMessage> have been removed.

Explicit support for Custom Messages

FIX 4.4 now provides <CustomMessage> element that is defined as a child node of the root node <FIXML>. CustomMessages should be defined following the pattern for standard FIX messages. The new custom message should then be made a child of the <CustomMessage> element.

By default - the FIXML DTD defines the <CustomMessage> as an Empty tag:

```
<!--
      To add custom messages - define them below and
       replace the EMPTY keyword with your message(s)
       inside parentheses separated by the vertical bar
       character
 -->

<!ELEMENT CustomMsg EMPTY>
```

If we wanted to add a custom message MarginRequirement to report MarginRequirements for positions.

Step 1. Create the MarginRequirement message in DTD.

```
<!ENTITY % MarginRequirementCustom "">
<!ENTITY % MarginRequirementContent "MgnReqmntID, Instrmnt,
MgnAmount, Text?, EncTextLen?, EncText?
%MarginRequirementCustom;">
<!ELEMENT MarginRequirement (%MarginRequirementContent;)>
<!ATTLIST MarginRequirement
    FIXMsgType_ENUM CDATA #FIXED "AM"
    Category CDATA #FIXED "Custom"
    FIXSpecVolume CDATA #FIXED "N/A"
    FullName CDATA #FIXED "MarginRequirement"
    ComponentType CDATA #FIXED "Message"

>
```

Step 2. Define any custom fields that are required.

We added two fields, MgnReqID and MgnAmount - these will need to be defined in the DTD.

```
<!ELEMENT MgnReqmntID (#PCDATA)>
<!ATTLIST MgnReqmntID
    FIXTag CDATA #FIXED "9XXX"
    DataType CDATA #FIXED "String"
    FullName CDATA #FIXED "MarginRequriementID"
    ComponentType CDATA #FIXED "Field"
>
```

```
<!ELEMENT MgnAmount (#PCDATA)>
<!ATTLIST MgnAmount
    FIXTag CDATA #FIXED "9XXX"
    DataType CDATA #FIXED "Amt"
    FullName CDATA #FIXED "MarginAmount"
    ComponentType CDATA #FIXED "Field"
>
```

Step 3. Add the MarginRequirement element as a sub-element the <CustomMsg> element.

In this case we add MarginRequirement as an optional subelement of <CustomMsg>.

```
<!ELEMENT CustomMsg (MarginRequirement?)>
```

Step 4. Validate your DTD using a third party product.

Abbreviations

FIX 4.3 used full field names to provide a consistent mapping between the FIXML representation and the FIX data dictionary. This resulted in very verbose FIXML messages. A standard set of abbreviations is now applied to FIX names. The FULLNAME is provided as meta-data to permit mapping between the FIX specification and the FIXML representation.

FIX 4.3 Field Representation

```
<!ELEMENT UnderlyingCouponPaymentDate (#PCDATA)>
<!ATTLIST UnderlyingCouponPaymentDate
   FIXTag CDATA #FIXED "241"
   DataType CDATA #FIXED "UTCDate"
>
```

FIX 4.4 Equivalent

```
<!ELEMENT UndCpnPmtDt (#PCDATA)>
<!ATTLIST UndCpnPmtDt
   FIXTag CDATA #FIXED "241"
   DataType CDATA #FIXED "LocalMktDate"
   FullName CDATA #FIXED "UnderlyingCouponPaymentDate"
   ComponentType CDATA #FIXED "Field"
>
```

Refer to Appendix 1-A in Volume 1 for a complete list of FIXML abbreviations.

Repeating Group Structures Optimized

In FIX 4.3 and earlier versions, the Number In Group field was included as part of FIXML explicitly. While the FIX 4.4 DTD looks more verbose, in actuality - there is only one element that contains the repeating group of Parties (Ptys), the outer "*List" element that was present in FIX 4.3 has been removed. This "roll up" technique was developed as part of the ISO/XML initiative.

FIX 4.3 Repeating Group Structure

```
<!ELEMENT PartiesList (NoPartyIDs?, PartiesGroup+)>
<!ELEMENT PartiesGroup (PartyID?, PartyIDSource?, PartyRole?,
PartySubID?)>
```

FIX 4.4 Repeating Group Structure

```
<!ENTITY % PtysCustom "">
<!ENTITY % PtysContent "PtyID?, PtyIDSrc?, PtyRole?, PtySubIDsGrp*
%PtysCustom;">
<!ELEMENT Ptys (%PtysContent;)+>
<!ATTLIST Ptys
   FullName CDATA #FIXED "Parties"
   NumInGrp_FIELD CDATA #FIXED "NoPtyIDs"
   FIXTag CDATA #FIXED "453"
   ComponentType CDATA #FIXED "BlockRepeating"
   Category CDATA #FIXED "Common"
>
```

Explicit support for custom fields in Repeating Groups and Component Blocks

Building upon the work in previous FIXML releases, the design pattern using ENTITIES to include custom fields was expanded to Component Blocks and Repeating Groups.

You may recall that ENTITIES work similarly to macros in the C or C++ preprocessor. The XML ENTITY statement can be used include a file or define a Macro. We use it within FIXML to define macros.

In the following example for the <Parties> Component Block - the PtysCustom ENTITY is where addition custom fields are added. The PtysCustom ENTITY is then included as part of the PtysContent ENTITY - which contains the fields that make up the Message, Component Block, or Repeating Group.

```
<!ENTITY % PtysCustom "">
<!ENTITY % PtysContent "PtyID?, PtyIDSrc?, PtyRole?, PtySubIDsGrp*
%PtysCustom;">
<!ELEMENT Ptys (%PtysContent;)+>
<!ATTLIST Ptys
   FullName CDATA #FIXED "Parties"
   NumInGrp_FIELD CDATA #FIXED "NoPtyIDs"
   FIXTag CDATA #FIXED "453"
   ComponentType CDATA #FIXED "BlockRepeating"
   Category CDATA #FIXED "Common"
>
```

Additional meta data captured as attributes

Attached as an appendix to this document are the FIX 4.4 DTD Design Rules taken from Volume 1 of the FIX 4.4 specification. Provided in the design rules are the meta-data items provided as attributes in the DTD. Highlights of additional meta data items with FIX 4.4 include the category and volume for FIX messages. Each FIXML element now is defined as a ComponentType, such as Message, Field, Block (component block), RepeatingGroup, BlockRepeating (component block that is also a repeating group).

Example Messages

The following sections show an example FIX 4.4 FIXML DTD version message, followed by the same message in FIX 4.3 FIXML. You will notice the following differences.

1. FIX 4.4 makes extensive use of abbreviations
2. FIX 4.4 has eliminated several intermediate elements, such as FIXMLMessage and ApplicationMessage.
3. FIX 4.4 no longer uses the *List element to indicate a repeating group, instead it has flattened this to use only the *Grp element that was previously contained inside the *List element.

FIX 4.4 DTD Version: Advertisement FIXML Message

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSPY v5 rel. 2 U (http://www.xmlspy.com)-->
<!DOCTYPE FIXML SYSTEM "D:\home\jim\FIX\FIX44\fix-
44\FIXML\XMLDTDRC06182003\FIXML.4.4.dtd">
<FIXML>
    <Header>
            <Sender>
                    <CompID>Text</CompID>
                    <SubID>Text</SubID>
```

```
                <LocationID>Text</LocationID>
        </Sender>
        <OnBehalfOf>
                <CompID>Text</CompID>
                <SubID>Text</SubID>
                <LocationID>Text</LocationID>
        </OnBehalfOf>
        <Target>
                <CompID>Text</CompID>
                <SubID>Text</SubID>
                <LocationID>Text</LocationID>
        </Target>
        <DeliverTo>
                <CompID>Text</CompID>
                <SubID>Text</SubID>
                <LocationID>Text</LocationID>
        </DeliverTo>
        <SndgTm>Text</SndgTm>
        <PossDupFlag Value="Y"/>
        <PossResend Value="Y"/>
        <MsgEncoding>Text</MsgEncoding>
        <HopGrp>
                <HopCompID>Text</HopCompID>
                <HopSndgTm>Text</HopSndgTm>
                <HopRefID>Text</HopRefID>
        </HopGrp>
</Header>
<Adv>
        <AdvId>Text</AdvId>
        <AdvTransTyp Value="N"/>
        <AdvRefID>Text</AdvRefID>
        <Instrmt>
                <Sym>Text</Sym>
                <SymSfx Value="WI"/>
                <SecID>Text</SecID>
                <SecIDSrc Value="1"/>
                <SecAltIDGrp>
                        <SecAltID>Text</SecAltID>
                        <SecAltIDSrc>Text</SecAltIDSrc>
                </SecAltIDGrp>
                <Prod Value="1"/>
                <CFICode>Text</CFICode>
                <SecTyp Value="EUSUPRA"/>
                <SecSubTyp>Text</SecSubTyp>
                <MatMoYr>Text</MatMoYr>
                <MatDt>Text</MatDt>
                <CpnPmtDt>Text</CpnPmtDt>
                <IssDt>Text</IssDt>
                <RepoCollSecTyp Value="EUSUPRA"/>
                <RepoTrm>Text</RepoTrm>
                <RepoRt>Text</RepoRt>
                <Fctr>Text</Fctr>
                <CreditRtng>Text</CreditRtng>
                <InstrRgstry Value="BIC"/>
                <CtryOfIss>Text</CtryOfIss>
                <StOrProvncOfIss>Text</StOrProvncOfIss>
                <LocaleOfIss>Text</LocaleOfIss>
                <RedDt>Text</RedDt>
                <StrkPx>Text</StrkPx>
                <StrkCcy>Text</StrkCcy>
                <OptAttribute>Text</OptAttribute>
                <CntractMultiplier>Text</CntractMultiplier>
                <CpnRt>Text</CpnRt>
                <SecExch Value="CDATA"/>
                <Issr>Text</Issr>
                <EncIssrLen>Text</EncIssrLen>
                <EncIssr>Text</EncIssr>
                <SecDesc>Text</SecDesc>
                <EncSecDescLen>Text</EncSecDescLen>
                <EncSecDesc>Text</EncSecDesc>
```

```
                <Pool>Text</Pool>
                <CntractSettlMo>Text</CntractSettlMo>
                <CPPgm Value="1"/>
                <CPRegTyp>Text</CPRegTyp>
                <EventsGrp>
                        <EventTyp Value="1"/>
                        <EventDt>Text</EventDt>
                        <EventPx>Text</EventPx>
                        <EventText>Text</EventText>
                </EventsGrp>
                <DtdDt>Text</DtdDt>
                <IntAcrlDt>Text</IntAcrlDt>
        </Instrmt>
        <AdvSide Value="B"/>
        <Qty>Text</Qty>
        <QtyTyp Value="0"/>
        <Px>Text</Px>
        <Ccy Value="CDATA"/>
        <TrdDt>Text</TrdDt>
        <TransactTm>Text</TransactTm>
        <Text>Text</Text>
        <EncTextLen>Text</EncTextLen>
        <EncText>Text</EncText>
        <URLLink>Text</URLLink>
        <LastMkt Value="CDATA"/>
        <TrdgSessID>Text</TrdgSessID>
        <TrdgSessSubID>Text</TrdgSessSubID>
    </Adv>
</FIXML>
```

FIX 4.3 FIXML Advertisement Message

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSPY v5 rel. 4 U (http://www.xmlspy.com)-->
<!DOCTYPE FIXML SYSTEM "D:\home\jim\FIX\FIX43-Errata\FIX43_Errata_20020920\fixml43.dtd">
<FIXML>
      <FIXMLMessage>
            <Header>
                  <Sender>
                        <CompID>Text</CompID>
                        <SubID>Text</SubID>
                        <LocationID>Text</LocationID>
                  </Sender>
                  <OnBehalfOf>
                        <CompID>Text</CompID>
                        <SubID>Text</SubID>
                        <LocationID>Text</LocationID>
                  </OnBehalfOf>
                  <Target>
                        <CompID>Text</CompID>
                        <SubID>Text</SubID>
                        <LocationID>Text</LocationID>
                  </Target>
                  <DeliverTo>
                        <CompID>Text</CompID>
                        <SubID>Text</SubID>
                        <LocationID>Text</LocationID>
                  </DeliverTo>
                  <SendingTime>Text</SendingTime>
                  <PossDupFlag Value="Y"/>
                  <PossResend Value="Y"/>
                  <MessageEncoding Value="ISO-2022-JP"/>
                  <HopList>
                        <NoHops>Text</NoHops>
                        <HopGroup>
                              <HopCompID>Text</HopCompID>
                              <HopSendingTime>Text</HopSendingTime>
                              <HopRefID>Text</HopRefID>
                        </HopGroup>
                  </HopList>
            </Header>
            <ApplicationMessage>
                  <Advertisement>
                        <AdvID>Text</AdvID>
                        <AdvTransType>
                              <AdvNew/>
                        </AdvTransType>
                        <Instrument>
                              <Symbol>Text</Symbol>
                              <SymbolSfx>Text</SymbolSfx>
                              <SecurityID>Text</SecurityID>
                              <SecurityIDSource Value="1"/>
                              <SecurityAltIDList>
                                    <NoSecurityAltID>Text</NoSecurityAltID>
                                    <SecurityAltIDGroup>
                                          <SecurityAltID>Text</SecurityAltID>
                                          <SecurityAltIDSource Value="1"/>
                                    </SecurityAltIDGroup>
                              </SecurityAltIDList>
                              <Product Value="1"/>
                              <CFICode>Text</CFICode>
                              <SecurityType>
                                    <Agency Value="FAC">
                                          <IssueDate>Text</IssueDate>
                                          <RedemptionDate>Text</RedemptionDate>

      <ContractMultiplier>Text</ContractMultiplier>
                                          <CouponRate>Text</CouponRate>
                                    </Agency>
                              </SecurityType>
```

```
                                    <MaturityMonthYear>Text</MaturityMonthYear>
                                    <MaturityDate>Text</MaturityDate>
                                    <InstrRegistry>Text</InstrRegistry>
                                    <CountryOfIssue>Text</CountryOfIssue>

      <StateOrProvinceOfIssue>Text</StateOrProvinceOfIssue>
                                    <LocaleOfIssue>Text</LocaleOfIssue>
                                    <SecurityExchange Value="CDATA"/>
                                    <Issuer>Text</Issuer>
                                    <EncodedIssuerGroup>
                                           <EncodedIssuerLen>Text</EncodedIssuerLen>
                                           <EncodedIssuer>Text</EncodedIssuer>
                                    </EncodedIssuerGroup>
                                    <SecurityDesc>Text</SecurityDesc>
                                    <EncodedSecurityDescGroup>

      <EncodedSecurityDescLen>Text</EncodedSecurityDescLen>

      <EncodedSecurityDesc>Text</EncodedSecurityDesc>
                                    </EncodedSecurityDescGroup>
                              </Instrument>
                              <AdvSide Value="B"/>
                              <Quantity>Text</Quantity>
                              <Price>Text</Price>
                              <Currency Value="CDATA"/>
                              <TradeDate>Text</TradeDate>
                              <TransactTime>Text</TransactTime>
                              <Text>Text</Text>
                              <EncodedTextGroup>
                                    <EncodedTextLen>Text</EncodedTextLen>
                                    <EncodedText>Text</EncodedText>
                              </EncodedTextGroup>
                              <URLLink>Text</URLLink>
                              <LastMkt Value="CDATA"/>
                              <TradingSessionID>Text</TradingSessionID>
                              <TradingSessionSubID>Text</TradingSessionSubID>
                        </Advertisement>
                  </ApplicationMessage>
      </FIXMLMessage>
</FIXML>
```

Appendix A - FIXML 4.3 Design Rules

FIXML Design Rules

1) Elements can contain other Elements, EMPTY content or PCDATA (text) content.

2) FIXML uses camel case notation in which elements and attributes may be made up of multiple words with each word beginning with a capital letter.

3) Certain commonly used and well-known acronyms like IOI and DK are capitalized and separated from the rest of the tag by an underscore. (i.e. IOI_Qty).

4) FIXML requires ordered content model. This differs from the traditional FIX approach ("tag=value" syntax) which allows fields to be in any order (other than the first couple and last).

5) FIXML supports conditionally required content models. Options must contain a Strike Price.

```
<!ELEMENT Option (StrikePrice, OptAttribute?)>
<!ATTLIST Option  FIXTag   CDATA  #FIXED '167'
                  DataType CDATA  #FIXED 'String'
                  Value    CDATA  #FIXED 'OPT' >
```

6) Content models of business messages contain entities that allow for customization. For example, all application messages have a custom entity that can be redefined to extend the content model of the particular message. The following illustrates the ListExecute message:

```
<!ENTITY % ListExecuteCustom "">
<!ENTITY % ListExecuteContent "ListID,ClientBidID?,BidID?,TransactTime,Text?,EncodedTextGroup?
%ListExecuteCustom;" >
<!ELEMENT ListExecute (%ListExecuteContent;)>
<!ATTLIST ListExecute FIXTag   CDATA  #FIXED '35'
  DataType CDATA  #FIXED 'String'
  Value    CDATA  #FIXED 'L' >
```

To extend the content model of the ListExecute message, add the following to the internal subset of a FIXML message.

```
<!DOCTYPE fixml SYSTEM "fixmlmain.dtd" [
  <!ENTITY % ListExecuteCustom ", InternalTransNumber?">
  <!ELEMENT InternalTransNumber (#PCDATA)>
]>
```

After entity reference resolution the Indication content model will look like:

```
<!ELEMENT ListExecute (ListID,ClientBidID?,BidID?,TransactTime,Text?,EncodedTextGroup?,
InternalTransNumber? )>
```

instead of

```
<!ELEMENT ListExecute (ListID,ClientBidID?,BidID?,TransactTime,Text?,EncodedTextGroup? )>
```

7) FIXML elements have attributes, which contain referential information related to the FIX Field ID, Data type, and numeric constraints. **Validation of these attributes must happen at the application level.**

FIXTag - contains the FIX Protocol Field ID (Tag).

DataType - reflects data types (char, int, float, month-year, day-of-month, time, date) from the FIX specification.

Example:

```
<!ELEMENT ForexReq EMPTY>
        <!ATTLIST ForexReq  FIXTag   CDATA  #FIXED '121'
            DataType CDATA  #FIXED 'Boolean'
            Value     (Y | N )  #REQUIRED
    SDValue   (Yes | No )  #IMPLIED >
```

8) FIX defines message types with the **MsgType** field (tag "35"). Since the existence of a particular element indicates the message type (ie <ExecutionReport>), **MsgType** is reflected as meta-data information. Each FIX message contains the attribute **FIXTag** with a fixed value equal to "35" and a **Value** attribute equal to the corresponding **MsgType** value.

```
<!ELEMENT QuoteReq (%QuoteReqContent; )>
<!ATTLIST QuoteReq
            FIXTag CDATA #FIXED "35"
           DataType CDATA #FIXED "char"
           Value CDATA #FIXED "R"
>
```

9) FIXML allows for the XML parser to validate enumerations from the FIX Specification. These elements are defined with EMPTY content models and an attribute called Value. The acceptable values for FIXML attribute enumerations come from the FIX Specification. **An optional attribute list call SDValue (SelfDescribingValue) contains the human-readable equivalent of the FIX specification values.**

```
<!ELEMENT ProcessCode EMPTY>
<!ATTLIST ProcessCode  FIXTag   CDATA  #FIXED '81'
              DataType CDATA  #FIXED 'char'
              Value     (0 | 1 | 2 | 3 | 4 | 5 | 6 )  #REQUIRED
              SDValue   (Regular |
                    SoftDollar |
                    StepIn |
                    StepOut |
                    StepInSoft |
                    StepOutSoft |
                        PlanSponsor )  #IMPLIED >
```

The linkage between Value and SDValue cannot be validated.

10) When fields are conditionally required based on the value of other fields, the Tag=Value pair becomes an element. For example, ExecRefID is required when ExecTransType = Cancel. The attribute **Value** is added and contains the valid FIX Specification value.

```
<!ELEMENT ExecTransType (ExecNew | ExecCancel | ExecCorrect | ExecStatus)>

<!ELEMENT ExecCancel (ExecRefID, LastQty, LastPx)>
        <!ATTLIST ExecCancel
                        FIXTag CDATA #FIXED "20"
                        Value CDATA #FIXED "1">
        >
```

20=1 (ExecTransType=Cancel)

becomes

```
<ExecTransType><ExecNew FIXTag="20" Value="1"> … </ExecTransType>
```

Applies to:

ExecNew, ExecCancel, ExecCorrect, ExecStatus, AllocStatusAccept, AllocStatusReject, AllocPartialAccept, AllocStatusReceived, AdvNew, AdvCancel, AdvReplace, IOINew, IOICancel, IOIReplace

11) FIXML has elements that serve as containers and do not map directly to FIX tag=value pairs.

```
<!ELEMENT MiscFeeList (NoMiscFees? , MiscFeeGroup+)>
```

12) Special containers are used when enumeration values of a FIX field must be split into two elements to handle conditionally required elements.

```
<!ELEMENT OrderDuration (TimeInForce | GTDTimeInForce)>

<!ELEMENT TimeInForce          EMPTY>
<!ATTLIST TimeInForce
    FIXTag CDATA #FIXED "59"
    DataType CDATA #FIXED "char"
    Value (0|1|2|3|4|5) #REQUIRED
    SDValue (Day|GoodTillCancel|AtTheOpening|ImmediateOrCancel|FillOrKill|
    GoodTillCrossing) #IMPLIED
    >

    <!ELEMENT GTDTimeInForce          (ExpireTime)>
    <!ATTLIST GTDTimeInForce
            FIXTag CDATA #FIXED "59"
            DataType CDATA #FIXED "char"
            Value CDATA #FIXED "6"
            SDValue CDATA #FIXED "GoodTillDate" >
```

13) Certain FIX Fields are grouped into parent/child relationships. Referential information is contained in two places. The attribute **FIXTags** contains a list of valid tags in the content model and each field has its own attribute.

```
<!ELEMENT Sender (CompID, SubID?, LocationID?)>
<!ELEMENT CompID (#PCDATA)>
<!ATTLIST CompID
    FIXTag CDATA #FIXED "49-56-115-128"
    SenderFIXTag CDATA #FIXED "49"
    TargetFIXTag CDATA #FIXED "56"
     OnBehalfOfFIXTag CDATA #FIXED "115"
    DeliverToFIXTag CDATA #FIXED "128"
    DataType CDATA #FIXED "char">
```

For example:

```
49=ssmb

becomes

<Sender><CompID SenderFIXTag="49">ssmb</CompID></Sender>
```

Applies to:

Sender, Target, Location, OnBehalfOf, DeliverTo

14) FIX repeating groups are supported through the use of collection elements. To support conversions between FIX and FIXML, fields that identify the number of repeating elements are contained in the content model of the collection element.

```
<!ELEMENT BidComponentList (NoBidComponents? , BidComponentGroup+)>
<!ELEMENT BidComponentGroup (  ListID?, Side?,TradingSessionID?,
TradingSessionSubID?,NetGrossInd?, Settlement?, Account? )>
```

Appendix B - FIXML 4.4 Design Rules

FIXML 4.4 Design Rules

This section was taken from the FIX Specification 4.4 Errata 20030618, Volume 1.

General

Elements can contain otherElements, EMPTY content, or PCDATA (text) content and Attributes.
Element names within FIXML start with the full names from the FIX specification and are abbreviated using the abbreviations specified as an appendix to this volume.FIXML uses camel case notation in which elements and attributes may be made up of multiple abbreviated words with each abbreviation beginning with a capital letter
FIXML requires content to be ordered. This differs from the traditional FIX approach ("tag=value" syntax) which allows fields to be in any order (other than the first couple and last).
The FIXML is composed of ComponentTypes that correspond to the components that make up the FIX protocol specification.
The following component types are defined for FIXML.

FIXML ComponentType	Description
Message	Corresponds to a FIX message from the FIX specification
Field	FIX Field
Block	FIX Component Block - group of fields that are related and always appear together. Examples: Instrument Component Block Component Blocks for FIX and FIXML are specified within this volume.
RepeatingGroup	A repeating group of fields within a FIX Message The cardinality of the repeating group is defined with a field of type NumInGroup. FIXML does not require a separate data item to store the cardinality (number of items in the group). However, meta data for the NumInGroup field associated with a repeating group is provided to assist in mapping between FIX and FIXML.
BlockRepeating	FIX Component Block that is also a repeating group, such as Parties Component Block.

FIXML provides metadata for each ComponentType that provides a mapping to the FIX document (such as Fullname of the field, tag value, FIX datatype). Metadata is provided as XML attributes within the DTD for FIXML. Metadata is provided as elements to the data type definitions for each component within the FIXML Schema version. **Validation of these attributes must happen at the application level.**
The meta data items for each ComponentType are defined below.

ComponentType - Field

Fields are the basic data elements that make up a FIX message. Each field is given a tag number and a FIX data type. Enumerations and value ranges are optionally used for fields.
Meta data provided for FIXML Fields

FIXML Field Metadata	
Metadata Item	**Description**
FIXTag	contains the FIX Protocol Field ID (Tag number)
DataType	One of the FIX datatypes defined in this specification (char, int, float, etc.).
FullName	Full name of the field within the FIX specification prior to abbreviation

ComponentType	"Field"
Value	Enumerated values - if applicable
SDValue	Self Describing values that correspond to the enumerated values in the Value attribute

Templates used to generate Fields in DTD

Fields that use enums and values

```
<!ELEMENT **XMLName** EMPTY>
  <!ATTLIST **XMLName**
    FIXTag CDATA #FIXED '**Tag**'
    DataType CDATA  #FIXED '**Type**'
    FullName CDATA #FIXED '**FieldName**'
    ComponentType CDATA #FIXED 'Field'
    Value **Value** #REQUIRED
    SDValue **SDValue** #IMPLIED >
```

Fields that use others standards or enums only

```
<!ELEMENT **XMLName** EMPTY>
  <!ATTLIST **XMLName**
    FIXTag CDATA #FIXED '**Tag**'
    DataType CDATA  #FIXED '**Type**'
    FullName CDATA #FIXED '**FieldName**'
    ComponentType CDATA #FIXED 'Field'
    Value **Value** #REQUIRED >
```

Other fields

```
<!ELEMENT **XMLName** (#PCDATA)>
  <!ATTLIST **XMLName**
    FIXTag CDATA #FIXED '**Tag**'
    DataType CDATA  #FIXED '**Type**'
    FullName CDATA #FIXED '**FieldName**'
    ComponentType CDATA #FIXED 'Field' >
```

Field Example in DTD

```
<!ELEMENT OrdQty (#PCDATA)>
<!ATTLIST OrdQty
     FIXTag CDATA #FIXED "38"
     DataType CDATA #FIXED "Qty"
     FullName CDATA #FIXED "OrderQty"
     ComponentType CDATA #FIXED "Field"
```

```
>
```

FIXML allows for the XML parser to validate enumerations from the FIX Specification. These elements are defined with EMPTY content models and an attribute called **Value**. The acceptable values for FIXML attribute enumerations come from the FIX Specification. An optional attribute list named **SDValue** (SelfDescribingValue) contains the human-readable equivalent of the FIX specification values. The linkage between Value and SDValue cannot be validated using the DTD.

```
<!ELEMENT SettlTyp EMPTY>
<!ATTLIST SettlTyp
    FIXTag CDATA #FIXED "63"
    DataType CDATA #FIXED "char"
    FullName CDATA #FIXED "SettlType"
    ComponentType CDATA #FIXED "Field"
    Value (0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9) #REQUIRED
    SDValue (Regular | Cash | NextDay | T2 | T3 | T4 | Future | WhenIssued | T5 | T1) #IMPLIED
>
```

ComponentType - Block

The FIX Component Blocks defined in Volume 1 that do not start off with a repeating group are identified as "Block" within FIXML

Meta data provided for FIXML Blocks

FIXML Block Metadata	
Metadata Item	**Description**
FullName	Full name of the field within the FIX specification prior to abbreviation
Category	Name of the category to which the element belongs
ComponentType	"Block"

Template used to generate Component Blocks in DTD

```
<!ENTITY % **CompName**Custom "" >
<!ENTITY % **CompName**Content "**FieldList** %**CompName**Custom;" >
<!ELEMENT **CompName** (%**CompName**Content;)>
<!ATTLIST **CompName**
    FullName CDATA #FIXED "**ComponentName**"
    ComponentType CDATA #FIXED "Block"
    Category CDATA #FIXED "Common">
```

Block Example

```
<!ENTITY % InstrmtCustom "">
<!ENTITY % InstrmtContent "Sym?, SymSfx?, SecID?, SecIDSrc?, SecAltIDGrp*, Prod?, CFICode?, SecTyp?, SecSubTyp?, MatMoYr?, MatDt?, CpnPmtDt?, IssDt?, RepoCollSecTyp?, RepoTrm?, RepoRt?, Fctr?, CreditRtng?, InstrRgstry?, CtryOfIss?, StOrProvinceOfIss?, LocaleOfIss?, RedDt?, StrkPx?, StrkCcy?, OptAttribute?, CntractMultiplier?, CpnRt?, SecExch?, Issr?, EncIssrLen?,
```

```
EncIssr?, SecDesc?, EncSecDescLen?, EncSecDesc?, Pool?, CntractSettlMo?, CPProgram?, CPRegTyp?, EventsGrp*, DtdDt?, IntAcrlDt? %InstrmtCustom;">

<!ELEMENT Instrmt (%InstrmtContent;)>

<!ATTLIST Instrmt

    FullName CDATA #FIXED "Instrument"

    ComponentType CDATA #FIXED "ComponentBlock"

    Category CDATA #FIXED "Common"

>
```

ComponentType - RepeatingGroup

Repeating groups from FIX messages have been identified. Many of the repeating groups are the same across multiple messages - even though they are not declared explicitly as component blocks. Repeating groups from within the spec that occur in multiple places have been identified in the repository as being implicit components. This permits the common naming and reuse of repeating group definitions across messages.

Meta data provided for FIXML Repeating Groups

FIXML RepeatingGroup Metadata	
Metadata Item	**Description**
NumInGrp_FIELD	Name of the NumInGrp field that is used in the FIX tag=value version of FIX for the repeating group.
FIXTag	Tag # of the NumInGrp field
Category	Name of the category to which the element belongs
ComponentType	"RepeatingGroup"

Template used to generate RepeatingGroup component in DTD

```
<!ENTITY % **ElementName**Custom "" >

<!ENTITY % **ElementName**Content "**FieldList** %**ElementName**Custom;" >

<!ELEMENT **ElementName** (%**ElementName**Content;)+>

<!ATTLIST **ElementName**

    NumInGrp_FIELD CDATA #FIXED '**CounterName**'

    FIXTag CDATA #FIXED '**CounterTag**'

    ComponentType CDATA #FIXED '**ComponentType**'

    Category CDATA #FIXED '**Category**' >
```

RepeatingGroup Example

```
<!ENTITY % AllocGrpCustom "">

<!ENTITY % AllocGrpContent "AllocAcct?, AllocAcctIDSrc?, MtchStat?, AllocPx?, AllocQty?, IndAllocID?, ProcessCode?, NstPtys?, NotifyBrkrOfCredit?, AllocHandlInst?, AllocText?, EncAllocTextLen?, EncAllocText?, CommData?, AllocAvgPx?, AllocNetMny?, SettlCurrAmt?, AllocSettlCurrAmt?, SettlCcy?, AllocSettlCcy?, SettlCurrFxRt?, SettlCurrFxRtCalc?, AllocAcrdIntAmt?, AllocIntAtMat?, MiscFeesGrp*, ClrInstGrp*, AllocSettlInstTyp? %AllocGrpCustom;">

<!ELEMENT AllocGrp (%AllocGrpContent;)+>
```

```
<!ATTLIST AllocGrp
    NumInGrp_FIELD CDATA #FIXED "NoAllocs"
    FIXTag CDATA #FIXED "78"
    ComponentType CDATA #FIXED "RepeatingGroup"
    Category CDATA #FIXED "Allocation"
>
```

ComponentType - BlockRepeating

Component Blocks that themselves are also repeating groups have been designated as BlockRepeating components within FIXML. This differentiation was done in order to minimize nesting of elements and to accommodate the additional meta data required for Repeating Groups (NumInGrp_FIELD, FIXTag).

Meta data provided for FIXML BlockRepeating

FIXML BlockRepeating Metadata	
Metadata Item	**Description**
NumInGrp_FIELD	Name of the NumInGrp field that is used in the FIX tag=value version of FIX for the repeating group.
FIXTag	Tag # of the NumInGrp field
DataType	One of the FIX datatypes defined in this specification (char, int, float, etc.).
FullName	Full name of the field within the FIX specification prior to abbreviation
Category	Name of the category to which the element belongs
ComponentType	"BlockRepeating"

Template used to generate BlockRepeating Components in DTD (Same as RepeatingGroups template above)

```
<!ENTITY % **ElementName**Custom "" >
<!ENTITY % **ElementName**Content "**FieldList** %**ElementName**Custom;" >
<!ELEMENT **ElementName** (%**ElementName**Content;)+>
<!ATTLIST **ElementName**
    FullName CDATA #FIXED "**ComponentName**"
    NumInGrp_FIELD CDATA #FIXED '**CounterName**'
    FIXTag CDATA #FIXED '**CounterTag**'
    ComponentType CDATA #FIXED '**ComponentType**'
    Category CDATA #FIXED '**Category**' >
```

BlockRepeating Example

```
<!ENTITY % PtysCustom "">
<!ENTITY % PtysContent "PtyID?, PtyIDSrc?, PtyRole?, PtySubIDsGrp* %PtysCustom;">
<!ELEMENT Ptys (%PtysContent;)+>
<!ATTLIST Ptys
    NumInGrp_FIELD CDATA #FIXED "NoPtyIDs"
```

```
    FIXTag CDATA #FIXED "453"
    ComponentType CDATA #FIXED "BlockRepeating"
    Category CDATA #FIXED "Common"
>
```

ComponentType - Message

FIX Messages are represented in FIXML using the same name.
Meta data provided for FIXML Messages

FIXML Field Metadata	
Metadata Item	**Description**
FullName	Full name of the field within the FIX specification prior to abbreviation
Category	Name of the category to which the element belongs
ComponentType	"Message"
FixSpecVolume	Number of the FIX volume where the message is documented
FIXMsgType	The enumeration (Value) of the MsgType (Tag 35) field in the FIX specification for this message

Template used to generate FIXML DTD

```
<!ENTITY % **MsgName**Custom "" >
<!ENTITY % **MsgName**Content "**FieldList** %**MsgName**Custom;" >
<!ELEMENT **MsgName** (%**MsgName**Content;)>
<!ATTLIST **MsgName**
    FIXMsgType_ENUM CDATA #FIXED "**MsgType**"
    Category CDATA #FIXED "**Category**"
    FIXSpecVolume CDATA #FIXED '**Volume**'
    FullName CDATA #FIXED "**MessageName**"
    ComponentType CDATA #FIXED "Message" >
```

Message Example

```
<!ENTITY % NewOrdSingleCustom "">
<!ENTITY % NewOrdSingleContent "ClOrdID, ScndClOrdID?, ClOrdLinkID?, TrdOriginationDt?, TrdDt?, Acct?, AcctIDSrc?, AcctTyp?, DayBkngInst?, BkngUnit?, PreallocMethod?, AllocID?, PreAllocGrp*, SettlTyp?, SettlDt?, CshMgn?, ClearingFeeInd?, HandlInst?, ExecInst?, MinQty?, MaxFloor?, ExDest?, TrdgSesGrp*, ProcessCode?, PrevClsPx?, Side, LocReqd?, TransactTm, QtyTyp?, OrdTyp, PxTyp?, Px?, StopPx?, Ccy?, ComplianceID?, SolicitedFlag?, IOIID?, QuotID?, TmInForce?, EfctvTm?, ExpireDt?, ExpireTm?, GTBkngInst?, OrdCpcty?, OrdRstctions?, CustOrdCpcty?, ForexReq?, SettlCcy?, BkngTyp?, Text?, EncTextLen?, EncText?, SettlDt2?, OrdQty2?, Px2?, PosEffect?, CoveredOrUncovered?, MaxShow?, TgtStrategy?, TgtStrategyParameters?, ParticipationRt?, CxllationRights?, MnyLaunderingStat?, RegistID?, Designation? %NewOrdSingleCustom;">
<!ELEMENT NewOrdSingle (%NewOrdSingleContent;)>
<!ATTLIST NewOrdSingle
    FIXMsgType CDATA #FIXED "D"
    Category CDATA #FIXED "SingleGeneralOrderHandling"
    FIXSpecVolume CDATA #FIXED "Volume4"
    FullName CDATA #FIXED "NewOrderSingle"
    ComponentType CDATA #FIXED "Message"
>
```

FIXML permits the inclusion of custom (user defined fields) in order to be compliant with the FIX specificiation. ComponentTypes: Block, RepeatingGroup, BlockRepeating, and Message are all implemented using entities to permit this customization. For example, all application messages have a custom entity that can be redefined to extend the content model of the particular message. In the following example, the Position Maintenance Request message has two Entities - PosMntReqCustom and PosMntReqContent. PosMntReqCustom is defined as an empty string. PosMntReqContent is a string of all the components that make up the PosMntReq message. PosMntReqContent also includes the PosMntReqCustom entity.
Custom fields are added to the PosMntReqCustom entity between the double quotes, each custom field must be preceeded by a comma. Once added, the custom fields are automatically picked up as part of the PosMntReqContent entity.

```
<!ENTITY % PosMntReqCustom "">
```

```
<!ENTITY % PosMntReqContent "PosReqID, PosTransTyp, PosMaintActn, OrigPosReqRefID?, PosMaintRptRefID?, ClearingBizDt, SettlSessID?, SettlSessSubID?, Acct, AcctIDSrc?, AcctTyp, Ccy?, TrdgSesGrp*, TransactTm, AdjmentTyp?, CntraryInstrctnInd?, PriorSpreadInd?, ThresholdAmt?, Text?, EncTextLen?, EncText? %PosMntReqCustom;">

<!ELEMENT PosMntReq (%PosMntReqContent;)>

<!ATTLIST PosMntReq

    FixMsgType CDATA #FIXED "AL"

    Category CDATA #FIXED "PositionMaintenance"

    FIXSpecVolume CDATA #FIXED "Volume5"

    FullName CDATA #FIXED "PositionMaintenanceRequest"

    ComponentType CDATA #FIXED "Message"
>
```

Certain FIX Fields are grouped into parent/child relationships. Referential information is contained in two places. The attribute **FIXTags** contains a list of valid tags in the content model and each field has its own attribute.

```
<!ELEMENT Sender (CompID, SubID?, LocationID?)>

<!ELEMENT CompID (#PCDATA)>

<!ATTLIST CompID

    FIXTag CDATA #FIXED "49-56-115-128"

    SenderFIXTag CDATA #FIXED "49"

    TargetFIXTag CDATA #FIXED "56"

     OnBehalfOfFIXTag CDATA #FIXED "115"

    DeliverToFIXTag CDATA #FIXED "128"

    DataType CDATA #FIXED "char">
```

For example:

49=ssmb

becomes

```
<Sender><CompID SenderFIXTag="49">ssmb</CompID></Sender>
```

Applies to:

Sender, Target, Location, OnBehalfOf, DeliverTo