
```
<xs:schema targetNamespace="http://www.fixprotocol.org/FIXML-4-4" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://www.fixprotocol.org/FIXML-4-4" elementFormDefault="qualified"
attributeFormDefault="unqualified">
            <xs:include schemaLocation="fixml-fields-impl-4-4.xsd"/>
            <!-- Base Header used by both the message and the batch headers -->
            <xs:group name="BaseHeaderElements">
                        <xs:sequence>
                                    <xs:element name="Hop" type="Hop_t" minOccurs="0" maxOccurs="unbounded"/>
                        </xs:sequence>
            </xs:group>
            <xs:attributeGroup name="BaseHeaderAttributes">
                        <xs:attribute name="SID" type="SenderCompID_t" use="optional"/>
                        <xs:attribute name="D2Loc" type="DeliverToLocationID_t" use="optional"/>
                        <xs:attribute name="PosDup" type="PossDupFlag_t" use="optional"/>
                        <xs:attribute name="PosRsnd" type="PossResend_t" use="optional"/>
                        <xs:attribute name="Snt" type="SendingTime_t" use="optional"/>
                        <xs:attribute name="OrigSnt" type="OrigSendingTime_t" use="optional"/>
                        <xs:attribute name="MsgEncd" type="MessageEncoding_t" use="optional"/>
            </xs:attributeGroup>
            <xs:complexType name="BaseHeader_t">
                        <xs:sequence>
                                    <x
                        </xs:sequence>
                        <xs:attributeG
            </xs:complexType>
```

FIXML Schema for FIX 4.4

Overview of the Schema Version of FIXML 4.4

Agenda

- Background
  - Optimizing FIXML timeline
  - Optimization Approach
- Schema Development
  - Design Objectives
  - FIXML Schema Working Group
- Examples
- Understanding the FIXML Schema
  - Structure and Organization
  - Versioning
  - Usage
  - Customization
- Documentation Provided
- FIXML Schema Distribution Package
- FPL XML Technology Road Map

Optimizing FIXML Timeline

- June 2002
  - CME approached FPL regarding using FIX for post trades
- July 2002
  - FIA formed the standards working group to drive the effort
  - FIXML was selected because
    - There was not an existing install base of FIX tag=value applications in the listed futures and options back office space
    - There was strong push back from firms to continue using the MQ Series transport instead of using the FIX Session layer
    - Post trade messages (allocation, trade capture, positions) have multiple levels of nesting - ideal for XML
  - CME started their pilot project

Timeline (continued)

- November 2002
  - CME quickly ran into problem that has plagued other FIXML initiatives - message size being too large for bandwidth and data storage requirements
    - Typical Trade Capture in FIXML was 3200 bytes
  - CME developed a transport optimized XML representation
    - Alternatives Examined
      - FIX tag=value
      - Convert from long descriptive element names to tag numbers
      - Convert to attributes
      - Convert to attributes and use abbreviations
    - Chose conversion to attributes using contextual abbreviations
    - Typical Trade Capture Report was reduced to 850 bytes
    - Message File Size was reduced from 25 MB to 9MB as a result of the optimization for a firm's daily trade file

Timeline (continued)

- December 2002 – January 2003
  - FIA Standards Working Group approached the Global Technical Committee regarding the transport optimized version of FIXML
- February 2003
  - The Global Technical Committee held discussions on the FIXML
    - Goal would be to have one version of FIXML
    - If message size is precluding usage we should consider converting to new version
    - Informally queried discussion groups about FIXML usage - mostly internal applications
    - Agreed any approach must provide some form of backward compatibility (via XSLT for instance)
- June 2003
  - Development of the FIX repository
  - Release of FIX 4.4 Errata version 20030618
  - Formation of the FIXML Schema Working Group
- January 2004
  - Release of the FIXML Schema version

Optimization Approach

- Eliminate elements that were used as holders for repeating groups
- Convert from elements to attributes where possible
  - Elements
    - Viewed conceptually as analagous to “objects”
    - From a FIX Perspective Elements represent messages and component blocks
  - Attributes
    - Viewed as properties (or attributes) of “objects”
    - Attributes represented fields within messages and component blocks
- Use contextual abbreviations
  - Standardized abbreviations using a dictionary of abbreviations
  - Contextual means – removing prefixes, such as “Trd” from fields on the Trade Capture messages
  - Permit manual overrides to mechanically generated abbreviations
- Choices were based upon experience of other organizations that have successfully deployed XML in production messaging application

Optimization Approach - Results

- Goal was not to necessarily have “human readable” XML
- Surprise: optimized version was deemed more readable
- Message size reduced from ~3200+ bytes to ~850+ bytes per trade record

Schema Development

The FIXML Schema Working Group was formed in July 2003 and chartered to define a transport optimized version of FIXML that was defined by an extensible XML Schema

The Working Group completed their work in December 2003

FIX 4.4 Work

- Modified the DTD based version of FIXML using the following techniques:
  - Roll up - eliminated extra levels of elements for repeating groups
  - Created standard abbreviation rules
  - Expanded meta data
    - Fullname
    - Category
    - ComponentType
      - Field
      - Message
      - Block
      - BlockRepeating
      - RepeatingGroup
    - Volume
      - Volume within FIX specification

FIXML Schema Working Group

- Formed to define the FIXML Schema
- Review proposed changes to FIXML for Transport Optimization
- Agree on FIXML (Instance document)
- Work on FIXML Schema Definition
  - Datatype usage
  - Use of types vs. elements
  - Integration with FpML

FIXML Schema Working Group Members

- *Jim Northey, Co-lead*
- *Kevin Houstoun, CitiGroup, Co-lead*
- Scott Atwell, American Century
- Robert Woodmansey, B2Bits
- Michael Timpone, Bloomberg
- Robert Stowsky, Brook Path Partners
- Lisa Taikitsadaporn, Brook Path Partners
- Matt Simpson, CME
- Mark Cox, CME
- Makoto Koizumi, Fujitsu
- Mary Ann Burns, FIA
- John Goeller, Lehman Brothers
- Satoru Mizukami, Nikko Citigroup
- Gerard Fritsch, The OCC
- Alex Olaru, The OCC
- Theresa Simon, The OCC
- Margaret Alfieri, Deloitte Consulting
- Peter Millington, OM Technology
- Mark Baumgardner, The OCC
- John Munro, Rolfe and Nolan
- Steve Wilkinson, Solution Forge
- Isabelle Cool, SWIFT
- Frank Van Damme, SWIFT
- Kris Ketels, SWIFT
- Sigrid Pousseur-Wiley, SWIFT
- Kevin Kobets, The OCC
- Karen Glad, The OCC
- Dean Kauffman, TradeWeb
- Sam Johnson, TransactTools
- Zach Zimmerer, TransactTools
- Nikhil Bose, AssistSoft
- Gerry Capone, State Street
- Paul Dreher, Administrative Answers

FIXML Schema Working Group Plan

- Consider Transport Optimized Approach
  - Attributes vs. Elements
  - Contextual Abbreviations
- Address component blocks built around limitations of FIX tag=value
  - InstrumentLeg, NestedParties, Nested2Parties,UnderlyingInstrument
- Develop XML Schema Design Approach
  - Leverage work already done by ISO/XML and FpML
- Address backward compatibility
- Establish implementation approach and timeline
- Produce FIXML Schema Version for FIX 4.4

Design Rules for FIXML Instances (Messages)

- FIXML implementation shall adhere to XML technology standards as specified by the W3C.
- FIXML implementation shall be suitable implementation for use in high volume transaction scenarios. Target applications:
  - Order Routing
  - Trade Reporting and Post Trade Processing
  - Distribution of product (instrument) information
  - Market making for lower volume applications
- FIXML implementation shall minimize bandwidth consumption (reduced message size). The goal is to have FIXML messages be less than 1.5 X the size of an equivalent FIX tag=value message.
- FIXML implementation shall maintain human readability of FIXML message, while still adhering to performance goals.

Design Rules for FIXML instances (cont'd)

- FIXML implementation shall support integration of FpML product specifications within the FIXML message in an equivalent manner to FIX 4.4 tag=value. This integration should use commonly agreed upon, de facto standard XML design patterns.
- FIXML implementation shall support a ready translation to and from FIX tag=value messages
- FIXML implementation shall provide a cross-reference to ISO 15022 repository for each message, element, and component
- FIXML implementation shall maintain the extensibility and customization available via the FIX tag=value message format, including:
- Ability to add custom messages
- Ability to add custom fields to messages, component blocks, and repeating groups

Design rules for FIXML Instances (conclusion)

- FIXML Implementation shall provide full transport level independence
- FIXML Implementation shall support version identification

Design Rules for FIXML Schema Document

- FIXML Schema shall be implemented using the current de facto industry best practices for XML Schema usage.
- FIXML Schema shall be implemented in such a way as to fully support the FIXML 4.4 "Schema Version" Instance Requirements defined above.
- FIXML Schema shall support version identification.
- FIXML Schema shall provide meta-data sufficient to identify the FIX field name, component type, tag number, ISO 15022 repository cross-reference.
- FIXML Schema shall be interoperable and compatible with the FpML schema.
- The FIXML Schema shall be based upon and be compatible with the current version of XML schema: ***http://www.w3.org/2001/XMLSchema***

Examples

A Very Simple Order in FIXML Schema Version

```
<FIXML v="4.4" r="20030618" s="20031218">
       <Order ID="123456" Side="1"
              TxnTm="2003-12-18T12:00:00" Typ="2" Px="85.00">
              <Hdr TID="SSB" SID="FCM"
              SeqNum="1" Snt="2003-12-18T12:00:00"/>
              <Instrmt Sym="IBM"/>
              <OrdQty Qty="100"/>
       </Order>
</FIXML>
```

Example Trade Capture Report

```
<?xml version="1.0" encoding="UTF-8"?>
<FIXML v="4.4" r="20030618" s="20031218">
<TrdCaptRpt RptID="60374" PrevlyRpted="N" LastQty="200" LastPx="-0.30"
        TrdDt="2003-12-03" TxnTm="2003-12-03T12:30:01" TransTyp="0"
        RptTyp="0" TrdTyp="0" BizDt="2003-12-03" MLEGRptTyp="3">
        <Instrmt ID="ED" SecTyp="MLEG" SubTyp="CAL" Exch="CME"/>
        <TrdLeg LastPx="97.55">
                <Leg ID="ED" CFI="FXXXSX" MMY="200312" Side="1"/>
        </TrdLeg>
        <TrdLeg LastPx="97.25">
                <Leg ID="ED" CFI="FXXXSX" MMY="200403" Side="2"/>
        </TrdLeg>
        <RptSide Side="1" OrdID="A456721" ClOrdID2="1234"  ClrFeeInd="B"
                InptSrc="MQM" CustCpcty="4" OrdTyp="L" TmBkt="A"
                SesID="RTH" SesSub="P" PosEfct="O" AllocInd="0">
                <Pty ID="600" R="1"/>
                <Pty ID="BAT" R="12"/>
                <Pty ID="052G0039" R="24">
                        <Sub ID="1" Typ="26"/>
                </Pty>
                <Pty ID="815" R="17"/>
                <Pty ID="TGK" R="37"/>
        </RptSide>
</TrdCaptRpt>
</FIXML>
```

Example Allocation Report

```
<?xml version="1.0" encoding="UTF-8"?>
<FIXML>
<AllocRpt AcrdIntAmt="55000.0" AvgPx="110.74080" Ccy="USD" GrossTrdAmt="11074080.0"
ID="20013212" LastFragment="Y" LastMkt="TRWB" NetMny="11129080.0" NoOrdsTyp="1" PxTyp="1"
Qty="10000000.000000" RptID="3349GCMXAG000671" RptTyp="3" SettlDt="2003-12-18" Side="1"
Stat="0" TransTyp="0" TrdDt="2003-12-15" TxnTm="2003-12-15T20:29:03">
          <Hdr SID="TRADEWEB" SeqNum="544" Snt="2003-12-15T20:29:05.827" TID="XYZFI"/>
          <OrdAlloc ClOrdID="NONREF" OrdID2="3349GCMXAG00067"/>
          <AllExc ExecID="3349GCMXAG000671" LastPx="110.74080" LastQty="10000000.000000"/>
          <Instrmt CpnRt="0.06" Desc="FNMA  6.000 05/15/2008" ID="31359MDU4" Issr="FNMA"
Issued="1998-05-11" MatDt="2008-05-15" Prod="1" SecTyp="FAC" Src="1" Sym="[N/A]"/>
          <Pty ID="GRNWUS33" R="1" Src="B"/>
          <Yield Typ="MATURITY" Yld="0.033575"/>
          <Alloc Acct="ACCT 2" AcrdIntAmt="44000.0" NetMny="8903264.0" Qty="8000000">
                    <Pty ID="DTC" R="4" Src="C"/>
          </Alloc>
</AllocRpt>
</FIXML>
```

Another Order Example

```
<FIXML>
        <Order ID="123456" Side="2"
               TxnTm="2001-09-11T09:30:47-05:00" Typ="2"
                Px="93.25" Acct="26522154">
               <Hdr Snt="2001-09-11T09:30:47-05:00" PosDup="N"
                      PosRsnd="N"
                      SeqNum="521"
                      SID="AFUNDMGR" TID="ABROKER"/>
               <Instrmt Sym="IBM" ID="459200101" Src="1"/>
               <OrdQty Qty="1000"/>
        </Order>
</FIXML>
```

Example courtesy Nikhil Bose (AssistSoft)

Understanding the FIXML Schema

This part of the presentation is an attempt to explain
the structure of the FIXML Schema

XML Schema

- W3C Standard use of XML to define the structure of other XML documents
  - Used to validate well formed XML documents
  - Designed as a replacement for the Document Type Definition (DTD) statements that were part of SGML and carried forward to XML
- Opinion: XML Schema can be difficult to learn and use
- Despite its *interesting* design, XML Schema is gaining wide spread acceptance as a standard
- Relax NG, another alternative schema language to define XML documents is gaining a resurgence of interest based upon the difficulty in using XML Schema and has become a W3C standard recently

What you need to know

- Working knowledge of FIX messages and fields
  - Understanding of XML
  - Basic understanding of XML Schema

XML Schema Prerequisites

XML datatypes
- Basic knowledge of builtin XML Schema datatypes and familiarity with the type hierarchy

Namespaces
- Basic knowledge of how Namespaces are used
- For instance using **`xs`** as a prefix to reference the XML Schema name space

How to define and later reference simple types
- <xs:simpleType>

How to define complex types to define FIXML messages and component blocks
- <xs:complextType>

How to define elements from FIXML datatypes
- <xsd:element>

How to define an element as abstract
- <xs:element … abstract="true"/>

How to defined named model groups and then later reference these to create complex types
- Also referred to as named model groups
- <xs:group>

How to define attribute groups and then later use them in complex type definitions
- <xs:attributeGroup>

Including and referencing other schema files from within a schema file
- <xs:include>

How to redefine components defined in another schema file
- <xs:redefine>

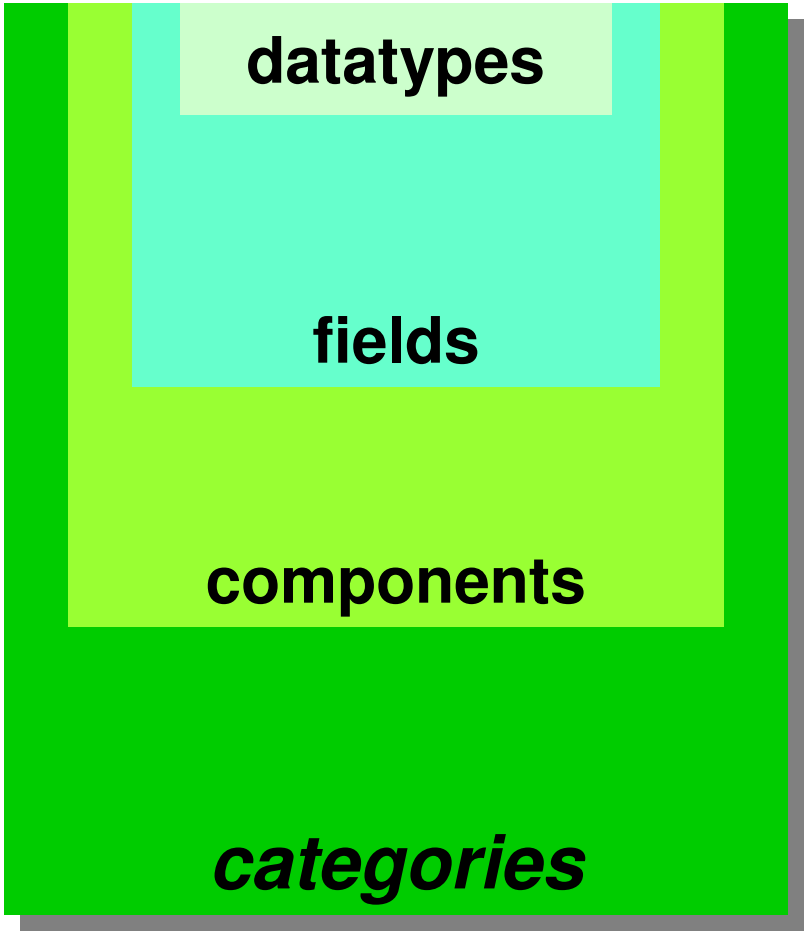
XML Schema References

- Online
  - W3C http://www.w3c.org/xmlschema
  - XXXXX
  - YYYYYY
- Books
  - We used the following three books extensively during the development of FIXML Schema
    - Alas, we actually had to read and compare the three, then do some experimenting to actually find out how schema worked or more importantly how software products supported the standard
  - O'Reilly Book
  - WROX Book
  - Prentice Hall Book

Organization of the FIXML Schema

- The FIXML Schema uses multiple schema files for the implementation
- A Layered approach was chosen
  - XML Schema for the FIXML Datatypes (and their mappings to XML Schema datatypes) are defined in a separate file
  - The definition of simple types and enumerations for FIXML fields include the datatypes schema
  - A schema file that contains the definition for components that make up FIX include the files file
  - A separate schema file is created for each category of FIXML messages as defined in the FIX specification
  - The following diagram shows the layers of schema files used to define FIXML

Layers of the FIXML Schema



Defines FIXML Datatypes

`fixml-4-4-datatypes.xsd`

Defines FIXML Fields and enumerations

`fixml-4-4-fields-*.xsd`

Defines FIXML root elements and global component blocks

`fixml-4-4-components-*.xsd`

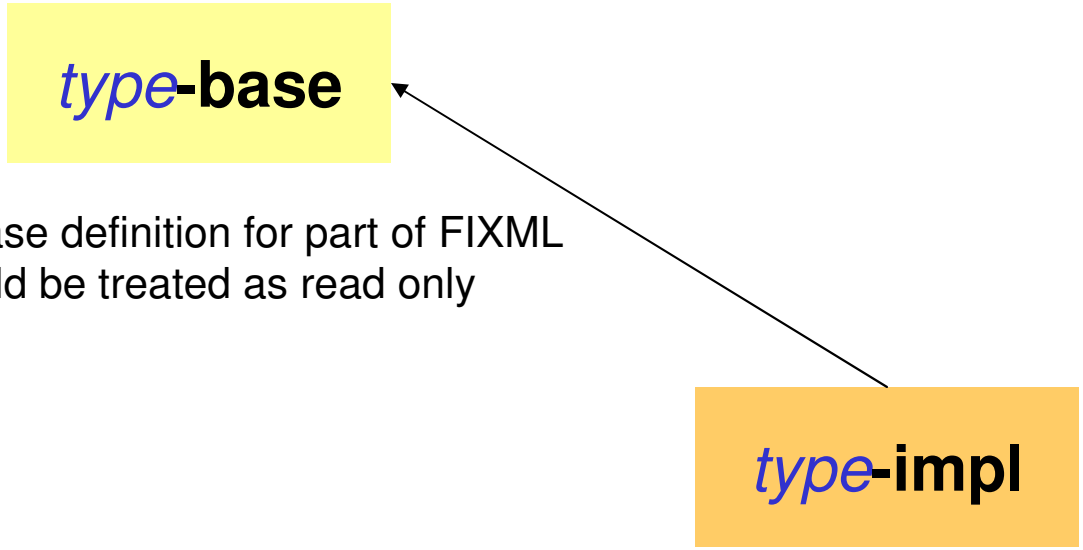
Defines FIXML messages and components for each message category in FIXML

`fixml-4-4-`*categoryName*`.xsd`

Extensibility Requirements

- FIXML needed to support the same customization capabilities provided in the FIX tag=value version
  - Add custom enumerations to existing fields
  - Add custom fields
  - Add custom messages
- Goal was to do so in such a way that the basic schema files could remain unchanged and all customizations would be isolated to separate implementation files

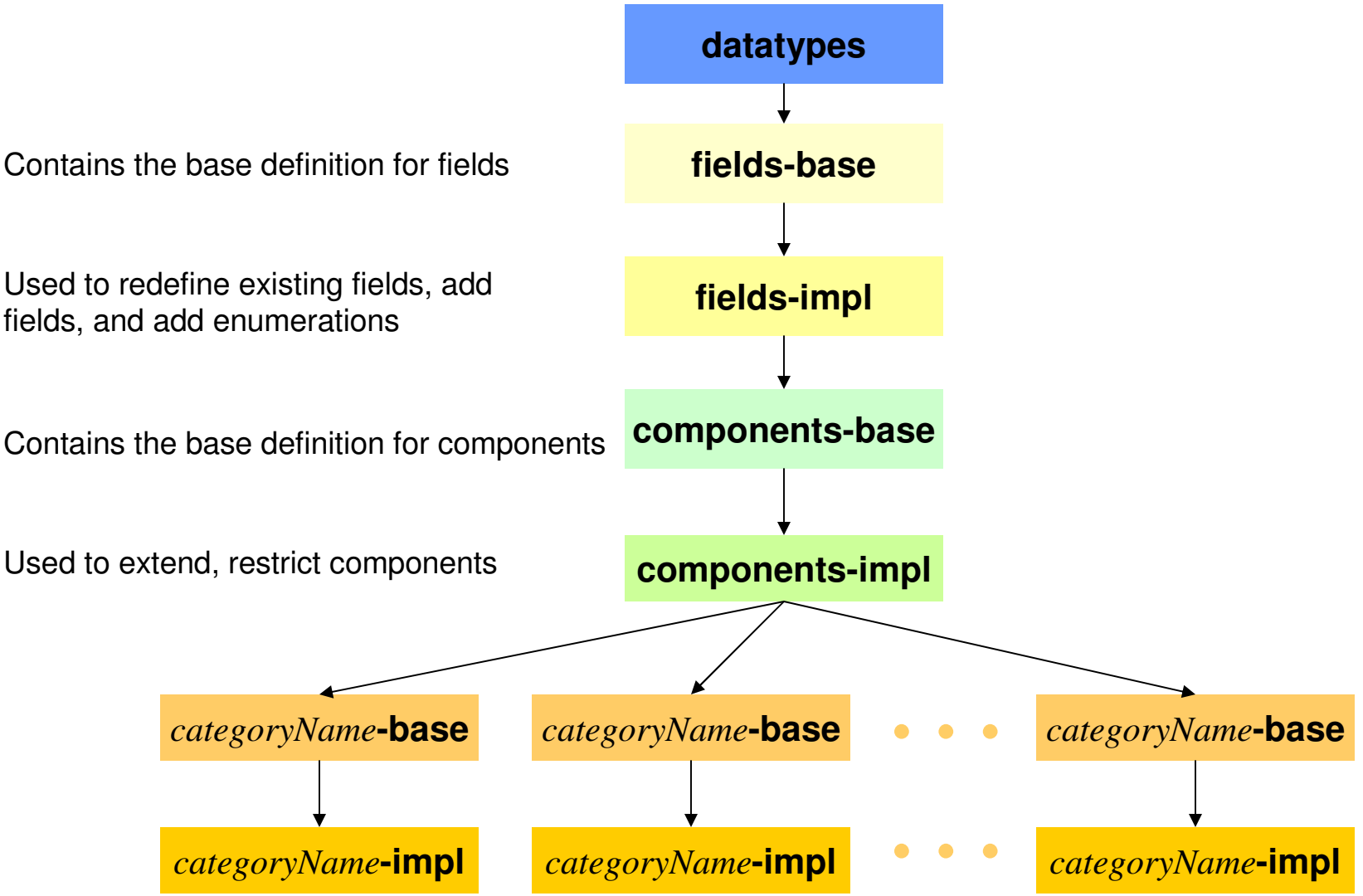
Extensibility Pattern



- Contains the base definition for part of FIXML
- Base files should be treated as read only

- Can be modified to restrict or extend the base definition
- Modification by agreement between counterparties or markets

Schema File Hierarchy Revisited



Schema File Naming Conventions

`fixml-`*Type*`-{base|impl}-`*m*`-`*n*`.xsd`

Type is one of
`datatypes`
`fields`
`components`

category -where category is one of the FIX message categories, such as `confirmation`, `listorder`, `order`, `settlement`, etc.

m is the FIX Major Version number, such as "4"

n is the FIX Minor Version number, such as "4"

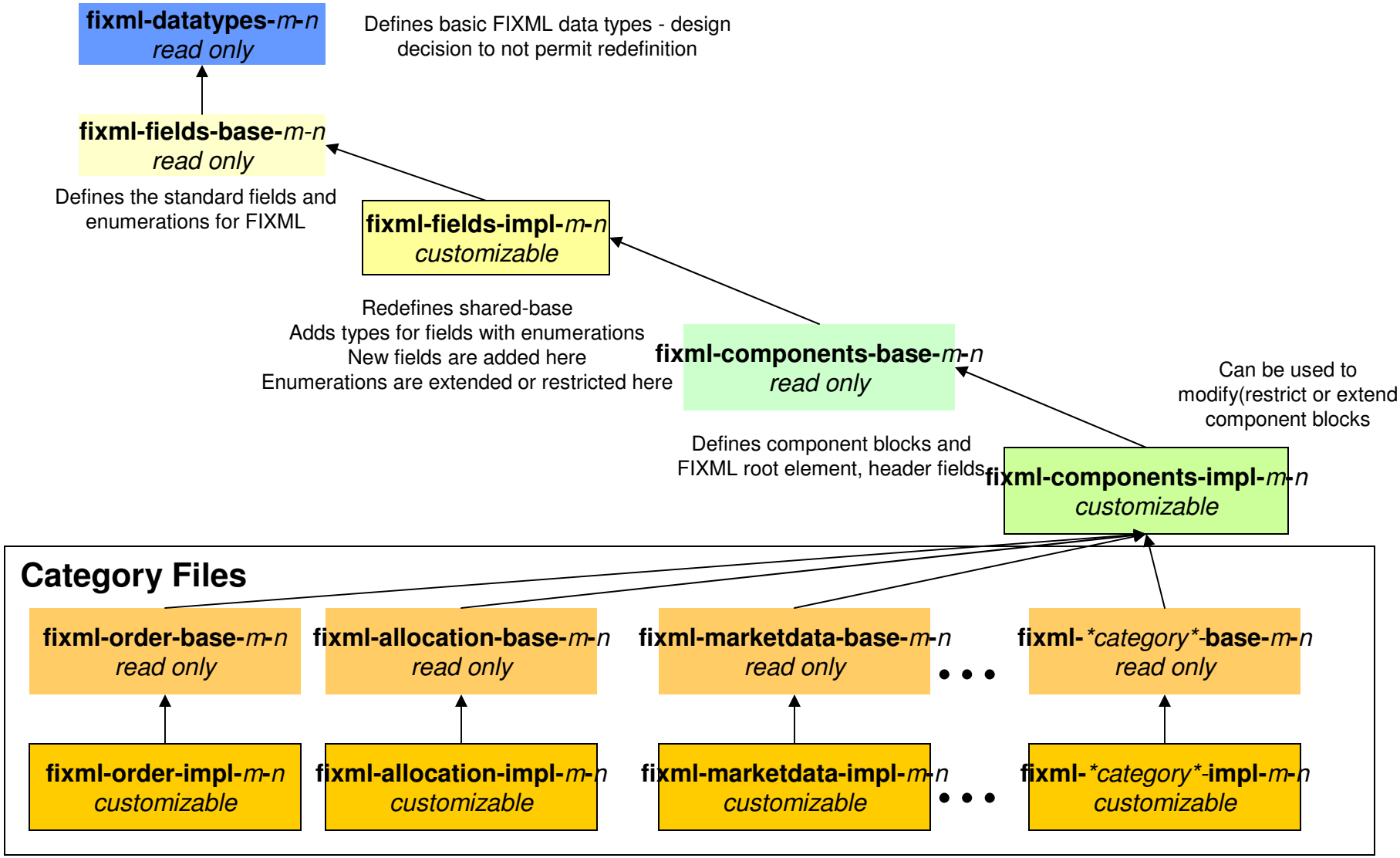
Example File Names

Fields base file for FIX Version 4.4: fixml-fields-base-4-4.xsd

Order Category base file for FIX Version 4.4: fixml-order-base-4-4.xsd

Component implementation file for FIX Version 4.4: fixml-components-impl-4-4.xsd

Customizable and read-only schema files



File Structure

- fixml-main-4-4.xsd
  - fixml-pretrade-4-4.xsd
    - indication
    - newsevents
    - quotation
    - marketdata
    - securitystatus
  - fixml-trade-4-4.xsd
    - singleorder
    - crossorder
    - multilegorder
    - listorder
  - fixml-posttrade-4-4.xsd
    - allocation
    - collateral
    - confirmation
    - positions
    - settlement
    - tradecapture
    - registration

`fixml-datatypes-m-n.xsd`

FIXML Datatypes

- Decision was made to use XML Schema datatypes
  - Compatible with XML software deemed more important than compatibility with FIX tag=value datatype formats
- Where does this decision hurt?
  - Date and time formats
  - XML date formats are based upon ISO 8601 standard
  - XML tools have implemented a very strict interpretation of this standard, meaning
    - dashes are required between date components
    - colons are required between time components
    - Time is separated from date portion via a "T"
    - Example

```
2003-12-18T12:30:001
```

fixml-datatypes-*m*-*n*.xsd

FIX Datatype	Description	FIXML Schema Implementation
int	Negative or Nonnegative	`xs:int`

Length	int field (see definition of "int" above) representing the length in bytes. Value must be positive.	`<xs:simpleType name="Length">`
		`  <xs:restriction base="xs:nonNegativeInteger"/>`
		`</xs:simpleType>`

SeqNum	int field (see definition of "int" above) representing a message sequence number. Value must be positive.	`<xs:simpleType name="SeqNum">`
		`  <xs:restriction base="xs:positiveInteger"/>`
		`</xs:simpleType>`

FIX Datatype	Description	FIXML Schema Implementation
float	Sequence of digits with optional decimal point and sign character (ASCII characters "**-**", "**0**" - "**9**" and "**.**"); the absence of the decimal point within the string will be interpreted as the float representation of an integer value. All float fields must accommodate up to fifteen significant digits. The number of decimal places used should be a factor of business/market needs and mutual agreement between counterparties. Note that float values may contain leading zeros (e.g. "00023.23" = "23.23") and may contain or omit trailing zeros after the decimal point (e.g. "23.0" = "23.0000" = "23" = "23."). Note that fields which are derived from float may contain negative values unless explicitly specified otherwise.	`xs:decimal` XML Schema datatype supports 18 significant digits

FIX Datatype	Description	FIXML Schema Implementation

Qty	float field (see definition of "float" above) capable of storing either a whole number (no decimal places) of "shares" (securities denominated in whole units) or a decimal value containing decimal places for non-share quantity asset classes (securities denominated in fractional units).	`<xs:simpleType name="Qty">`
		`<xs:restriction base="xs:decimal"/>`
		`</xs:simpleType>`

Price	float field (see definition of "float" above) representing a price. Note the number of decimal places may vary. For certain asset classes prices may be negative values. For example, prices for options strategies can be negative under certain market conditions. Refer to ***Volume 7: FIX Usage by Product*** for asset classes that support negative price values.	`<xs:simpleType name="Price">`
		`<xs:restriction base="xs:decimal"/>`
		`</xs:simpleType>`

fixml-datatypes-*m*-*n*.xsd

FIX Datatype	Description	FIXML Schema Implementation

Price	float field (see definition of "float" above) representing a price. Note the number of decimal places may vary. For certain asset classes prices may be negative values. For example, prices for options strategies can be negative under certain market conditions. Refer to ***Volume 7: FIX Usage by Product*** for asset classes that support negative price values.	`<xs:simpleType name="Price">`
		
		`<xs:restriction base="xs:decimal"/>`
		`</xs:simpleType>`

PriceOffset	float field (see definition of "float" above) representing a price offset, which can be mathematically added to a "Price". Note the number of decimal places may vary and some fields such as LastForwardPoints may be negative.	`<xs:simpleType name="PriceOffset">`
		`<xs:restriction base="xs:decimal"/>`
		`</xs:simpleType>`

fixml-datatypes-*m*-*n*.xsd

FIX Datatype	Description	FIXML Schema Implementation

Amt	float field (see definition of "float" above) typically representing a Price times a Qty.	`<xs:simpleType name="Amt">`
		`  <xs:restriction base="xs:decimal"/>`
		`</xs:simpleType>`

Percentage	float field (see definition of "float" above) representing a percentage (e.g. .05 represents 5% and .9525 represents 95.25%). Note the number of decimal places may vary.	`<xs:simpleType name="Percentage">`
		`  <xs:restriction base="xs:decimal"/>`
		`</xs:simpleType>`

char	Single character value, can include any alphanumeric character or punctuation except the delimiter. All char fields are case sensitive (i.e. **m** ≠ **M**).	`<xs:simpleType name="char">`
		`  <xs:restriction base="xs:string">`
		`  <xs:pattern value=".{1}"/>`
		`  </xs:restriction>`
		`</xs:simpleType>`

Boolean	a char field (see definition of "char" above) containing one of two values: 'Y' = True/Yes 'N' = False/No	xs:boolean

fixml-datatypes-*m*-*n*.xsd

FIX Datatype	Description	FIXML Schema Implementation
String	Alphanumeric free format strings can include any character or punctuation except the delimiter. All char fields are case sensitive (i.e. **morstatt** ≠ **Morstatt**).	xs:string
MultipleValueString	String field (see definition of "String" above) containing one or more space delimited values.	NMTOKENS

Country	String field (see definition of "String" above) representing a country using ISO 3166 Country code (2 character) values.	`<xs:simpleType name="Country">`
		`  <xs:restriction base="xs:string"/>`
		`</xs:simpleType>`

Currency	String field (see definition of "String" above) representing a currency type using ISO 4217 Currency code (3 character) values.	`<xs:simpleType name="Currency">`
		`  <xs:restriction base="xs:string"/>`
		`</xs:simpleType>`

fixml-datatypes-*m*-*n*.xsd

FIX Datatype	Description	FIXML Schema Implementation

	Market Identifier Code - ISO 10383 Market Identifier Code (MIC)	`<xs:simpleType name="MIC">`
		`  <xs:restriction base="xs:string"/>`
		`</xs:simpleType>`

Exchange	String field (see definition of "String" above) representing a market or exchange.- ISO 10383 Market Identifier Code (MIC)	`<xs:simpleType name="Exchange">`
		`  <xs:restriction base="MIC"/>`
		`</xs:simpleType>`

fixml-datatypes-*m*-*n*.xsd

FIX Datatype	Description	FIXML Schema Implementation

month-year	String field representing month of a year. An optional day of the month can be appended or an optional week code.
	Valid formats:
	YYYYMM
	YYYYMMDD
	YYYYMMWW
	YYYY = 0000-9999, MM = 01-12, DD = 01-31, WW = w1, w2, w3, w4, w5.
		`<xs:simpleType name="MonthYear">`
		`<xs:restriction base="xs:string">`
		`<xs:pattern value="\d{4}(0|1)\d([0-3wW]\d)?"/>`
		`</xs:restriction>`
		`</xs:simpleType>`

fixml-datatypes-*m*-*n*.xsd

FIX Datatype	Description	FIXML Schema Implementation

UTCTimestamp	Time/date combination represented in UTC (Universal Time Coordinated, also known as "GMT") in **either** YYYYMMDD-HH:MM:SS (whole seconds) or YYYYMMDD-HH:MM:SS.sss (milliseconds) format, colons, dash, and period required.
	Valid values:
	YYYY = 0000-9999, MM = 01-12, DD = 01-31, HH = 00-23, MM = 00-59, SS = 00-60 (60 only if UTC leap second) (without milliseconds).
	YYYY = 0000-9999, MM = 01-12, DD = 01-31, HH = 00-23, MM = 00-59, SS = 00-60 (60 only if UTC leap second), sss=000-999 (indicating milliseconds).
	Leap Seconds: Note that UTC includes corrections for leap seconds, which are inserted to account for slowing of the rotation of the earth. Leap second insertion is declared by the International Earth Rotation Service (IERS) and has, since 1972, only occurred on the night of Dec. 31 or Jun 30. The IERS considers March 31 and September 30 as secondary dates for leap second insertion, but has never utilized these dates. During a leap second insertion, a *UTCTimestamp* field may read "19981231-23:59:59", "19981231-23:59:**60**", "19990101-00:00:00". (see http://tycho.usno.navy.mil/leapsec.html)
		`<xs:simpleType name="UTCTimestamp">`
		
		`<xs:restriction base="xs:dateTime"/>`
		
		`</xs:simpleType>`

FIX Datatype	Description	FIXML Schema Implementation

UTCTimeOnly	Time-only represented in UTC (Universal Time Coordinated, also known as "GMT") in **either** HH:MM:SS (whole seconds) or HH:MM:SS.sss (milliseconds) format, colons, and period required. This special-purpose field is paired with UTCDateOnly to form a proper UTCTimestamp for bandwidth-sensitive messages.
	Valid values:
	HH = 00-23, MM = 00-60), SS = 00-59. (60 only if UTC leap second (without milliseconds)
	HH = 00-23, MM = 00-59, SS = 00-60 (60 only if UTC leap second), sss=000-999 (indicating milliseconds).
		`<xs:simpleType name="UTCTimeOnly">`
		`<xs:restriction base="xs:time"/>`
		`</xs:simpleType>`

UTCDateOnly	Date represented in UTC (Universal Time Coordinated, also known as "GMT") in YYYYMMDD format. This special-purpose field is paired with UTCTimeOnly to form a proper UTCTimestamp for bandwidth-sensitive messages.
	Valid values:
	YYYY = 0000-9999, MM = 01-12, DD = 01-31.
		`<xs:simpleType name="UTCDateOnly">`
		`<xs:restriction base="xs:date"/>`
		`</xs:simpleType>`

fixml-datatypes-*m*-*n*.xsd

FIX Datatype	Description	FIXML Schema Implementation

LocalMktDate	Date of Local Market (vs. UTC) in YYYYMMDD format. This is the "normal" date field used by the FIX protocol.
	Valid values:
	YYYY = 0000-9999, MM = 01-12, DD = 01-31.
		`<xs:simpleType name="LocalMktDate">`
		`<xs:restriction base="xs:date"/>`
		`</xs:simpleType>`

data	Raw data with no format or content restrictions. Data fields are always immediately preceded by a length field. The length field should specify the number of bytes of the value of the *data* field (up to but not including the terminating SOH). *Caution: the value of one of these fields may contain the delimiter (SOH) character. Note that the value specified for this field should be followed by the delimiter (SOH) character as all fields are terminated with an "SOH"*	`</xs:simpleType>`
		`<xs:simpleType name="data">`
		`<xs:restriction base="xs:string"/>`
		`</xs:simpleType>`

fixml-fields-base-*m*-*n*.xsd

fixml-fields-impl-*m*-*n*.xsd

Defining FIXML fields

Each FIX field is defined as a simple data type in the fields-base and the fields-impl files

`fixml-fields-base-m-n.xsd`

Field Representation

- A simple datatype is defined for each field in the FIX Specification used in FIXML

`<xs:simpleType>`

- The datatype name is derived from the full name of the FIX field with an "_t" appended
  - Some Examples
    - ClOrdID_t
    - OrderID_t
    - MsgSeqNum_t

fixml-fields-base-*m*-*n*.xsd

Example Field Definition for ClOrdID(tag 11)

```
<xs:simpleType name="ClOrdID_t">
   <xs:annotation>
      <xs:documentation xml:lang="en">Unique identifier for Order as
assigned by the buy side  institution  broker  intermediary  etc
identified by SenderCompID  49  or OnBehalfOfCompID  5  as appropriate
Uniqueness must be guaranteed within a single trading day   Firms
particularly those  which electronically submit multi day orders  trade
globally or throughout market close periods  should ensure  uniqueness
across days  for example by embedding a date within the ClOrdID field
      </xs:documentation>
      <xs:appinfo xmlns:x="http://www.fixprotocol.org/fixml/metadata.xsd">
         <xs:Xref Protocol="FIX" name="ClOrdID" tag="11" datatype="String"
ComponentType="Field" StdAbbrev="ClOrdID" QualifiedAbbrev="ClOrdID"
Category="SingleGeneralOrderHandling" CategoryAbbrev="ID"/>
         <xs:Xref Protocol="ISO_15022_XML"/>
      </xs:appinfo>
  </xs:annotation>
  <xs:restriction base="xs:string"/>
</xs:simpleType>
```

Notice the use of an XML datatype

fixml-fields-base-*m-n*.xsd

Example Field Definition for AvgPx(tag=6)

```
<xs:simpleType name="AvgPx_t">
    <xs:annotation>
      <xs:documentation xml:lang="en">
      Calculated average price of all fills on this order
      For Fixed Income trades AvgPx is always expressed as percent of
par
regardless of the PriceType  423  of LastPx  3
AvgPx will contain an average of percent of par values  see LastParPx
669
for issues traded in Yield  Spread or Discount
      </xs:documentation>
      <xs:appinfo
xmlns:x="http://www.fixprotocol.org/fixml/metadata.xsd">
        <xs:Xref Protocol="FIX" name="AvgPx" tag="6" datatype="Price"
ComponentType="Field"/>
        <xs:Xref Protocol="ISO_15022_XML"/>
      </xs:appinfo>
    </xs:annotation>
    <xs:restriction base="Price"/>
</xs:simpleType>
```

Notice the use of a FIXML defined datatype

`fixml-fields-base-m-n.xsd`

Representing enumerated fields in FIXML

Defining enumerated fields to permit customization without modifying the fields-base file proved to be a challenge

`fixml-fields-base-m-n.xsd`

Representing enumerations wasn't easy

- XML Schema supports defining a set of enumerations for a data type simple enough
- XML Schema does not make it easy to restrict / extend those enumerations – so we had to come up with a pattern for this purpose
- Enumerations are supported by
  - Defining a data type for the enumerations for a field in the fields-base file
  - Defining data type for the field that references the enumerations datatype in the fields-impl file
  - The fields-impl file then gives us a place to either restrict or extend the set of enumeration values
    - Use a **union** to extend the set of enumerations
    - Use redefinition to restrict the set of enumerations

fixml-fields-base-*m*-*n*.xsd

```
<xs:simpleType name="CommType_enum_t">
   <xs:annotation>
     <xs:documentation xml:lang="en">Commission type Valid values:  = per unit  implying shares  par
currency  etc   2 = percentage 3 = absolute  total monetary amount  4 =   for CIV buy orders
percentage waived   cash discount 5 =   for CIV buy orders  percentage waived   enhanced units 6 =
points per bond or or contract   Supply ContractMultiplier  23  in the  Instrument  component block if
the object security is denominated in a size other than the industry default   000 par for bonds
     </xs:documentation>
      <xs:appinfo xmlns:x="http://www.fixprotocol.org/fixml/metadata.xsd">
         <xs:Xref Protocol="FIX" name="CommType" tag="13" datatype="char"
ComponentType="Field"/>
         <xs:Xref Protocol="ISO_15022_XML"/>
      </xs:appinfo>
      <xs:appinfo xmlns:x="http://www.fixprotocol.org/fixml/metadata.xsd">
          <x:EnumDoc value="1" desc="PerShare"/>
          <x:EnumDoc value="2" desc="Percent"/>
          <x:EnumDoc value="3" desc="Absolute"/>
          <x:EnumDoc value="4" desc="PctWaivedCshDisc"/>
          <x:EnumDoc value="5" desc="PctWaivedEnUnits"/>
          <x:EnumDoc value="6" desc="PerBond"/>
      </xs:appinfo>
   </xs:annotation>
   <xs:restriction base="xs:string">
      <xs:enumeration value="1"/>
      <xs:enumeration value="2"/>
      <xs:enumeration value="3"/>
      <xs:enumeration value="4"/>
      <xs:enumeration value="5"/>
      <xs:enumeration value="6"/>
   </xs:restriction>
</xs:simpleType>
```

Documentation

Actual Enumerations

fixml-fields-impl-*m*-*n*.xsd

The enumerated field is defined in the impl file

```
<xs:simpleType name="CommType_t">
        <xs:restriction base="CommType_enum_t"/>
</xs:simpleType>
```

The field definition refers back to the enumeration

fixml-components-base-*m*-*n*.xsd

fixml-components-impl-*m*-*n*.xsd

Components

Components are defined as complex types

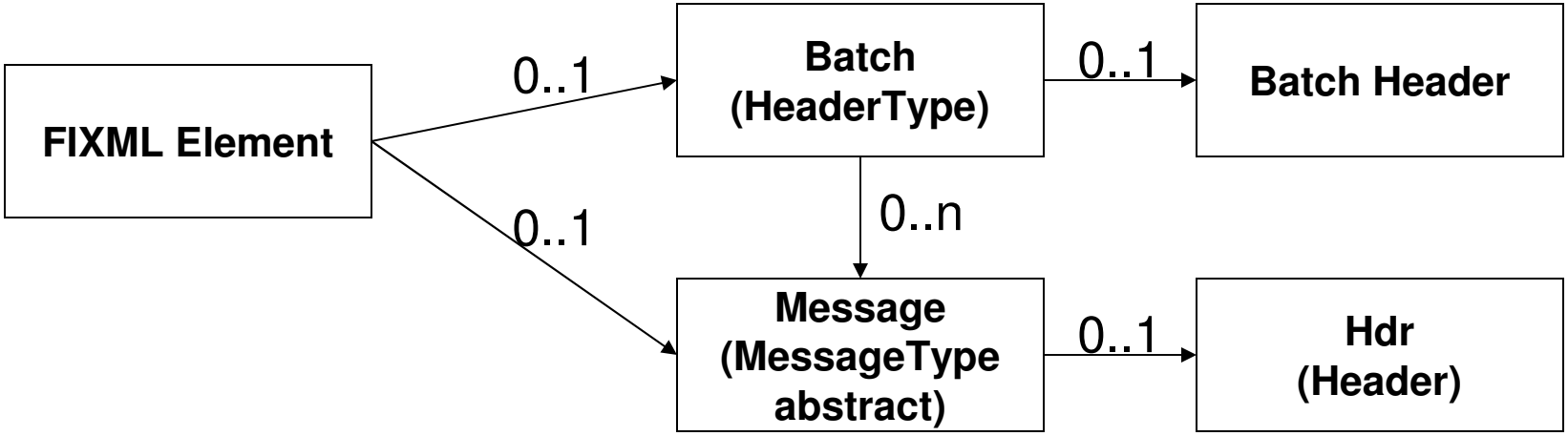
Two types of components: Explicit and Implicit

fixml-*category*-base-*m*-*n*.xsd

fixml-*category*-impl-*m*-*n*.xsd

Message Categories

FIXML Document Structure



Single Message Usage

```
<FIXML>
    <Order>
       <Hdr/>
     </Order>
</FIXML>
```

Batch Message Usage

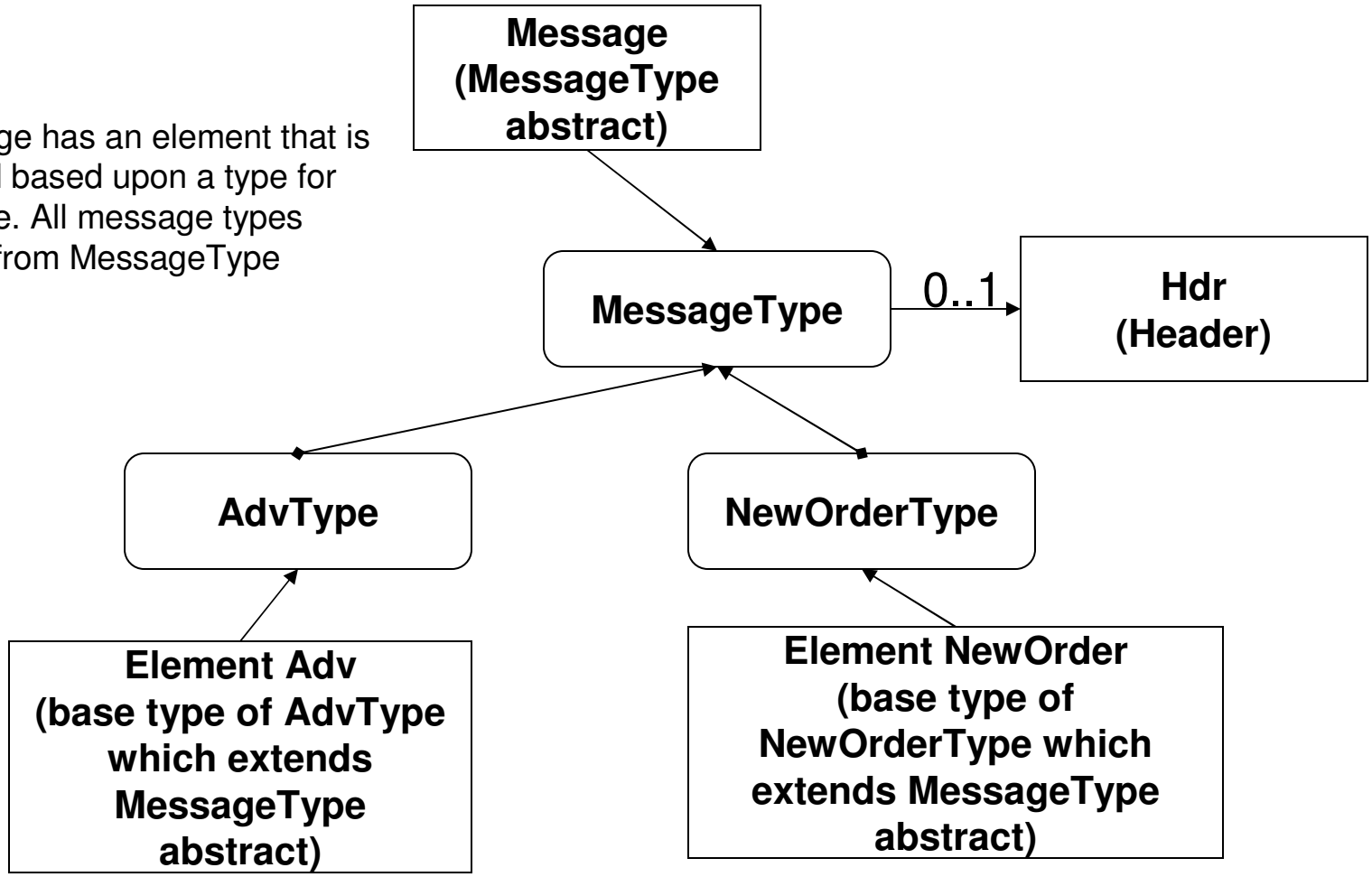
```
<FIXML>
   <Batch>
      <Hdr/>
      <Order>
        <Hdr/>
      </Order>
      <Order>
         <Hdr/>
      </Order>
    </Batch>
</FIXML>
```

Abstract Message Type

```
<!-- Message -->

<xs:complexType name="Abstract_message_t">

        <xs:sequence>
                <xs:element     name="Hdr"
                                type="MessageHeader_t"
                                minOccurs="0"/>
        </xs:sequence>

</xs:complexType>

<xs:element     name="Message"
                type="Abstract_message_t"
                abstract="true"/>
```

FIXML Message Structure

Each message has an element that is implemented based upon a type for that message. All message types must inherit from MessageType



Versioning

- Explicit versioning in file name
  - Unlike other uses of XML - we felt that the version should be explicitly identified
- Attributes on <FIXML> element

v FIX Version “4.4”

r FIX Version release date “20030618”

s Schema release date “20031218”

```
<FIXML v="4.4" r="20030618" s="20031218"/>
```

Usage

- Recommendations on use of Schema
  - Do not reference on <FIXML> element
  - Use for testing and certification

Customizations

Add a custom field to a message
Add a custom field to a component block
Add a custom message to a category
Add a custom enumeration
Restrict enumerations

Documentation

Documentation provided with FIXML Schema

- FIXML Schema Guide
  - Distributed as PDF document
  - Source in Microsoft Word
- FIXML Schema Overview Presentation
  - This PowerPoint presentation
  - Distributed as PDF document
- Example Library
  - XML text files

FIXML Schema Distribution Package

- FIXML Schema (fixml-4-4-schema-20040109.zip)
  - Version is specified as a YYYYMMDD date of the release – expect versioning for errata and enhancements to be on a different schedule than the base FIX specification
  - Schema files (.xsd)
  - Documentation directory
- Examples (fixml-4-4-schema-examples-20040109.zip)
  - Separate zip file from website – will be updated more frequently than the Schema distribution package
  - .xml examples
  - May later be expanded to include sample source code applications
- Supplement (fixml-4-4-schema-supplement-20040115.zip)
  - Schemas for previous versions of FIX
  - The backward compatible schema for the DTD version of FIX 4.4
  - XSLT translation examples

FIX XML Technology Road Map

Plenty of work remains in the areas of integration and adoption

FIX XML Technology Road Map

- January 2004 – June 2004
  - FIXML and FpML Interoperability
    - FpML-FIXML Working Group
  - FIXML Transport Layer Standardization
    - Industry Standard Middleware
      - FIX Session Layer
      - IP Multicast
      - CORBA
      - Web Services (SOAP, UDDI, WSDL)
    - Commercial Middleware
      - Tibco
      - MQ Series (With FIA Standards Working Group)
      - Others?
  - FIXML Language Bindings
    - Java
    - C++
    - SOAP
    - .NET

FIX XML Technology Road Map

- July 2004 – December 2004
  - Advanced FIXML Capabilities
    - Encoding business rules
      - Conditionally required fields
      - Semantic message correctness
      - Application of Schematron
    - Relax-NG representation of FIXML 4.4. Syntax
  - Integrate FpML Schema with cross reference mapping to ISO 15022 Repository

FIX XML Technology Road Map

- 2005
  - Advanced Transport Research
    - Binary Compatible XML formats
      - Applicability to FIX
      - Applicability to FIXML
      - FIXML – FIX Interoperability
  - Next round of interoperability and integration with FpML and ISO 15022 XML