FIX TRADING COMMUNITY
Nordic Trading Conference 2024

– How to use Orchestra Server? –

Thursday 16th May 2024

Hanno Klein
FIX Technical Director
GTC EMEA Co-Chair
Senior Standards Advisor, FIXdom

FIX TRADING COMMUNITY™
INDUSTRY-DRIVEN • INDEPENDENT • NEUTRAL

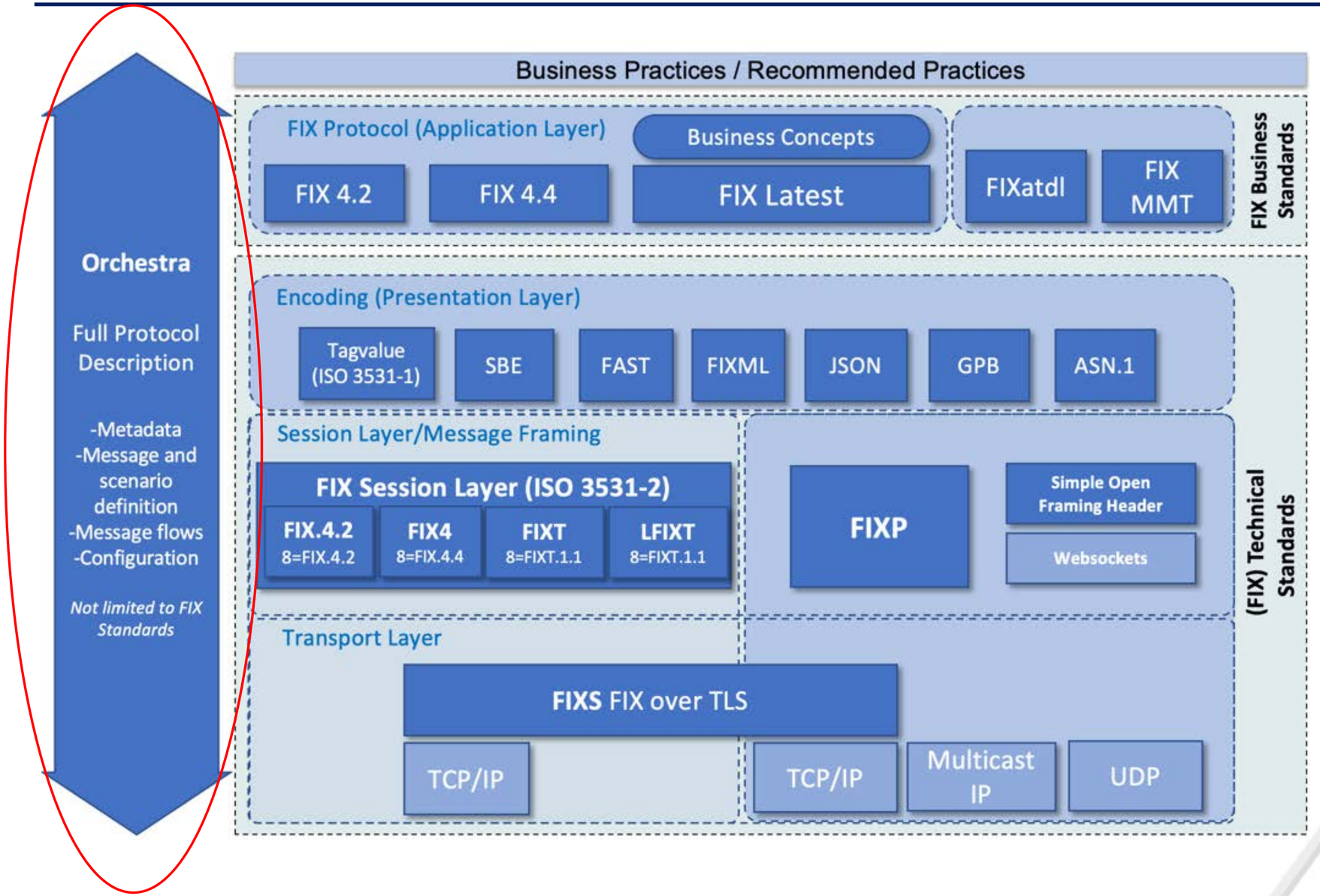
Agenda

- Orchestra Technical Standard
  - Objectives
  - Orchestra in a nutshell
- Rules of Engagement with Orchestra
  - Application Level
- Orchestra Scenarios

How to use Orchestra Server?
Orchestra Technical Standard

FIX TRADING COMMUNITY™
INDUSTRY-DRIVEN • INDEPENDENT • NEUTRAL

FIX Standards



FIX TRADING COMMUNITY™
INDUSTRY-DRIVEN • INDEPENDENT • NEUTRAL

Objectives for Orchestra

- Machine-readable standard for meta-data describing the content and behavior of an electronic messaging interface.
- Protocol agnostic to be applicable to FIX and non-FIX interfaces.
  - FIX Protocol (across all versions and flavors, including user-defined elements)
  - Regulatory protocols (e.g. US: SEC-CAT, Europe: ESMA/FCA, Asia: SFC-DS-OL)
  - Industry standard protocols (e.g. ISO 20022, FpML)
  - Proprietary protocols (trading venues, clearinghouses, buy/sell-side, vendors)
- Encoding agnostic to separate the business semantics from the wire format (standard/proprietary, ASCII/binary, with/without meta-data).
- Metadata for technical connectivity (counterparties, connections, sessions, versions, encodings, security,…)

Orchestra in a nutshell (application level)

- Basic features
  - Messages, groups, components, fields, code sets, codes, generic datatypes.
  - Nesting of groups/components inside messages, groups, components.
  - Simple presence rules (mandatory, optional, forbidden) for elements.
  - Unique identification and versioning (a.k.a. pedigree) of all elements.

- Advanced features
  - Conditional rules defined with expressions (e.g. Score DSL).
  - Scenarios for most elements to distinguish use cases.
  - Workflows to support request/response models or complex negotiations.
  - Actors and state machines to define transitions.

Orchestra in a nutshell (connection level)

- Basic features
  - An interface is a collection of services, protocols, sessions, and the transport exposed by a counterparty.
  - A service is an offering of an application (e.g. order entry) and requires the identification of an Orchestra XML file describing the messages etc.
  - A protocol relates to a specific layer of the technical stack of the interface, e.g. to the encoding or to the session protocol supported by the interface.
  - A session describes the connection with a counterparty (e.g. IP addresses).
  - A transport describes the lowest layer of the interface (e.g. TCP, UDP multicast).
- Advanced features
  - FIXatdl® can be supported as the protocol used for the user interface and requires the identification of an FIXatdl® XML file describing the GUI.
  - A session may have an effective time (start/end time) to support configuration prior to use.
  - A session definition may contain security keys (e.g. certificates, private keys) to be used when exchanging messages.

How to use Orchestra Server?
Rules of Engagement

FIX TRADING COMMUNITY™
INDUSTRY-DRIVEN • INDEPENDENT • NEUTRAL

Rules of Engagement (application level)

- Task
  - Design a FIX Latest compliant interface with an order entry message and a response confirming a) the successful addition to the order book and b) the partial or full execution of the order.
  - The order needs to support ticker symbols and ISINs for the security.
  - Order attributes are type (market or limit), side, price, quantity, target venue and an optional custom field “MyUDF” for additional information.
- Approach
  - Use spreadsheet to design messages “top-down”.
  - Use Playlist to design messages “bottom-up”.
    1. Define code sets (SecurityIDSource(22), OrdType(40), Side(54), ExecType(150), OrdStatus(39), MsgType(34), BeginString(8))
    2. Define components (Instrument, OrdQtyData, ExecType, OrdStatus)
    3. Define messages (NewOrderSingle(35=D), ExecutionReport(35=8))
  - Use Orchestra Server to add custom field “MyUDF” and create specification document
  - Export Rules of Engagement as Orchestra XML file and PDF document

Step 1: Define messages in spreadsheet

Message	Component	Field	Value(s)
NewOrderSingle(35=D)		BeginString(8)	FIXT.1.1
		ClOrdID(11)	
	Instrument	Symbol(55)	
	Instrument	SecurityID(48)	
	Instrument	SecurityIDSource(22)	4=ISIN
		OrdType(40)	1=Market 2=Limit
		Price(44)	
	OrderQtyData	OrderQty(38)	
		Side(54)	1=Buy 2=Sell
		ExDestination(100)	
		MyUDF(20000)	

Message	Component	Field	Value(s)
ExecutionReport(35=8)		BeginString(8)	FIXT.1.1
		ClOrdID(11)	
		OrderID(37)	
	Instrument	Symbol(55)	
	Instrument	SecurityID(48)	
	Instrument	SecurityIDSource(22)	4=ISIN
		OrdType(40)	1=Market 2=Limit
		Price(44)	
	OrderQtyData	OrderQty(38)	
		Side(54)	1=Buy 2=Sell
		ExDestination(100)	
		ExecType(150)	0=New F=Trade
		OrdStatus(39)	0=New 1=Partially Filled 2=Filled
		LeavesQty(151)	
		CumQty(14)	
		LastQty(32)	
		LastPx(31)	
		MyUDF(20000)	

Step 2: Define code sets in Playlist

- BeginStringCodeSet - Type String
  - [ ] FIX.4.2=FIX42
  - [ ] FIX.4.4=FIX44
  - [x] FIXT.1.1=FIXT11
- MsgTypeCodeSet - Type String
  - [ ] 0=Heartbeat
  - [ ] 1=TestRequest
  - [ ] 2=ResendRequest
  - [ ] 3=Reject
  - [ ] 4=SequenceReset
  - [ ] 5=Logout
  - [ ] 6=IOI
  - [ ] 7=Advertisement
  - [x] 8=ExecutionReport
  - [ ] 9=OrderCancelReject
  - ...
  - [x] D=NewOrderSingle
  - ...
- SideCodeSet - Type char
  - [x] 1=Buy
  - [x] 2=Sell
  - [ ] 3=BuyMinus
  - [ ] 4=SellPlus
  - [ ] 5=SellShort
  - [ ] 6=SellShortExempt
  - [ ] 7=Undisclosed
  - [ ] 8=Cross
  - [ ] 9=CrossShort
  - [ ] A=CrossShortExempt
  - [ ] B=AsDefined
  - [ ] C=Opposite
  - [ ] D=Subscribe
  - [ ] E=Redeem
  - [ ] F=Lend
  - [ ] G=Borrow
  - [ ] H=SellUndisclosed
- OrdTypeCodeSet - Type char
  - [x] 1=Market
  - [x] 2=Limit
  - [ ] 3=Stop
  - [ ] 4=StopLimit
  - [ ] 5=MarketOnClose
  - [ ] 6=WithOrWithout
  - [ ] 7=LimitOrBetter
  - [ ] 8=LimitWithOrWithout
  - [ ] 9=OnBasis
  - [ ] A=OnClose
  - [ ] B=LimitOnClose
  - [ ] C=ForexMarket
  - [ ] D=PreviouslyQuoted
  - [ ] E=PreviouslyIndicated
  - [ ] F=ForexLimit
  - [ ] G=ForexSwap
  - [ ] H=ForexPreviouslyQuoted
  - [ ] I=Funari
  - [ ] J=MarketIfTouched
  - [ ] K=MarketWithLeftOverAsLimit
  - [ ] L=PreviousFundValuationPoint
  - [ ] M=NextFundValuationPoint
  - [ ] P=Pegged
  - [ ] Q=CounterOrderSelection
  - [ ] R=StopOnBidOrOffer
  - [ ] S=StopLimitOnBidOrOffer
- SecurityIDSourceCodeSet - Type String
  - [ ] 1=CUSIP
  - [ ] 2=SEDOL
  - [ ] 3=QUIK
  - [x] 4=ISINNumber
  - [ ] 5=RICCode
  - [ ] 6=ISOCurrencyCode
  - [ ] 7=ISOCountryCode
  - [ ] 8=ExchangeSymbol
  - [ ] 9=ConsolidatedTapeAssociation
  - [ ] A=BloombergSymbol
  - [ ] B=Wertpapier
  - [ ] C=Dutch
  - [ ] D=Valoren
  - [ ] E=Sicovam
  - [ ] F=Belgian
  - [ ] G=Common
  - [ ] H=ClearingHouse
  - [ ] I=ISDAFpMLSpecification
  - [ ] J=OptionPriceReportingAuthority
  - [ ] K=ISDAFpMLURL
  - [ ] L=LetterOfCredit
  - [ ] M=MarketplaceAssignedIdentifier
  - [ ] N=MarkitREDEntityCLIP
  - [ ] P=MarkitREDPairCLIP
  - [ ] Q=CFTCCommodityCode
  - [ ] R=ISDACommodityReferencePrice
  - [ ] S=FinancialInstrumentGlobalIdentifier
  - [ ] T=LegalEntityIdentifier
  - [ ] U=Synthetic
  - [ ] V=FidessaInstrumentMnemonic
  - [ ] W=IndexName
  - [ ] X=UniformSymbol
  - [ ] Y=DigitalTokenIdentifier
- ExecTypeCodeSet - Type char
  - [x] 0=New
  - [ ] 3=DoneForDay
  - [ ] 4=Canceled
  - [ ] 5=Replaced
  - [ ] 6=PendingCancel
  - [ ] 7=Stopped
  - [ ] 8=Rejected
  - [ ] 9=Suspended
  - [ ] A=PendingNew
  - [ ] B=Calculated
  - [ ] C=Expired
  - [ ] D=Restated
  - [ ] E=PendingReplace
  - [x] F=Trade
  - [ ] G=TradeCorrect
  - [ ] H=TradeCancel
  - [ ] I=OrderStatus
  - [ ] J=TradeInAClearingHold
  - [ ] K=TradeHasBeenReleasedToClearing
  - [ ] L=TriggeredOrActivatedBySystem
  - [ ] M=Locked
  - [ ] N=Released
- OrdStatusCodeSet - Type char
  - [x] 0=New
  - [x] 1=PartiallyFilled
  - [x] 2=Filled
  - [ ] 3=DoneForDay
  - [ ] 4=Canceled
  - [ ] 5=Replaced
  - [ ] 6=PendingCancel
  - [ ] 7=Stopped
  - [ ] 8=Rejected
  - [ ] 9=Suspended
  - [ ] A=PendingNew
  - [ ] B=Calculated
  - [ ] C=Expired
  - [ ] D=AcceptedForBidding
  - [ ] E=PendingReplace

Step 3: Define components in Playlist

- Instrument
  - ...
  - SecurityID(48) - Type String
  - SecurityIDSource(22) - SecurityIDSourceCodeSet - Type String - Union Reserved100Plus
  - SecurityReferenceDataSupplement(2962) - Type String
  - SecurityStatus(965) - SecurityStatusCodeSet - Type String
  - SecuritySubType(762) - Type String
  - SecurityType(167) - SecurityTypeCodeSet - Type String
  - SecurityXML - Component
  - Seniority(1450) - SeniorityCodeSet - Type String
  - SettlDisruptionProvision(2143) - SettlDisruptionProvisionCodeSet - Type int
  - SettlMethod(1193) - SettlMethodCodeSet - Type String
  - SettlRateIndex(1577) - Type String
  - SettlRateIndexLocation(1580) - Type String
  - SettlSubMethod(2579) - SettlSubMethodCodeSet - Type int - Union Reserved100Plus
  - SettleOnOpenFlag(966) - Type String
  - SettledEntityMatrixPublicationDate(1945) - LocalMktDate - Base Type String
  - SettledEntityMatrixSource(1944) - Type String
  - ShortSaleRestriction(1687) - ShortSaleRestrictionCodeSet - Type int
  - StateOrProvinceOfIssue(471) - Type String
  - StrategyType(2141) - StrategyTypeCodeSet - Type String
  - StreamGrp - Group
  - StrikeCurrency(947) - Currency - Base Type String
  - StrikeCurrencyCodeSource(2904) - CurrencyCodeSourceCodeSet - Type String
  - StrikeIndex(1866) - Type String
  - StrikeIndexCurvePoint(2600) - Type String
  - StrikeIndexQuote(2601) - StrikeIndexQuoteCodeSet - Type int
  - StrikeIndexSpread(2001) - PriceOffset - Base Type float
  - StrikeMultiplier(967) - Type float
  - StrikePrice(202) - Price - Base Type float
  - StrikePriceBoundaryMethod(1479) - StrikePriceBoundaryMethodCodeSet - Type int
  - StrikePriceBoundaryPrecision(1480) - Percentage - Base Type float
  - StrikePriceDeterminationMethod(1478) - StrikePriceDeterminationMethodCodeSet - Type ir
  - StrikePricePrecision(2577) - Type int
  - StrikeUnitOfMeasure(1698) - UnitOfMeasureCodeSet - Type String
  - StrikeValue(968) - Type float
  - SwapClass(1941) - SwapClassCodeSet - Type String
  - SwapSubClass(1575) - SwapSubClassCodeSet - Type String
  - Symbol(55) - Type String
  - ...

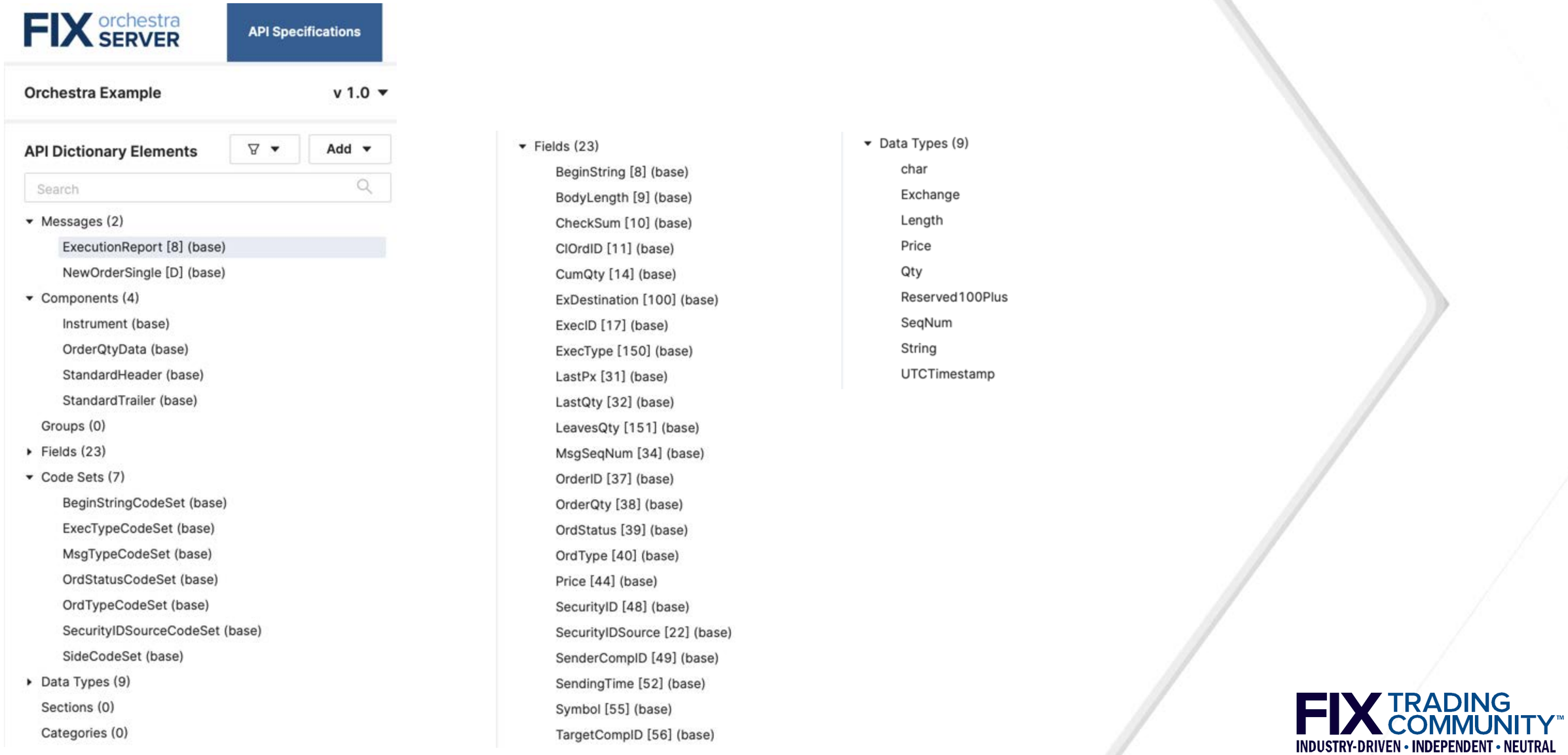
- OrderQtyData
  - CashOrderQty(152) - Qty - Base Type float
  - OrderPercent(516) - Percentage - Base Type float
  - OrderQty(38) - Qty - Base Type float
  - RoundingDirection(468) - RoundingDirectionCodeSet - Type char
  - RoundingModulus(469) - Type float

Step 4: Define messages in Playlist

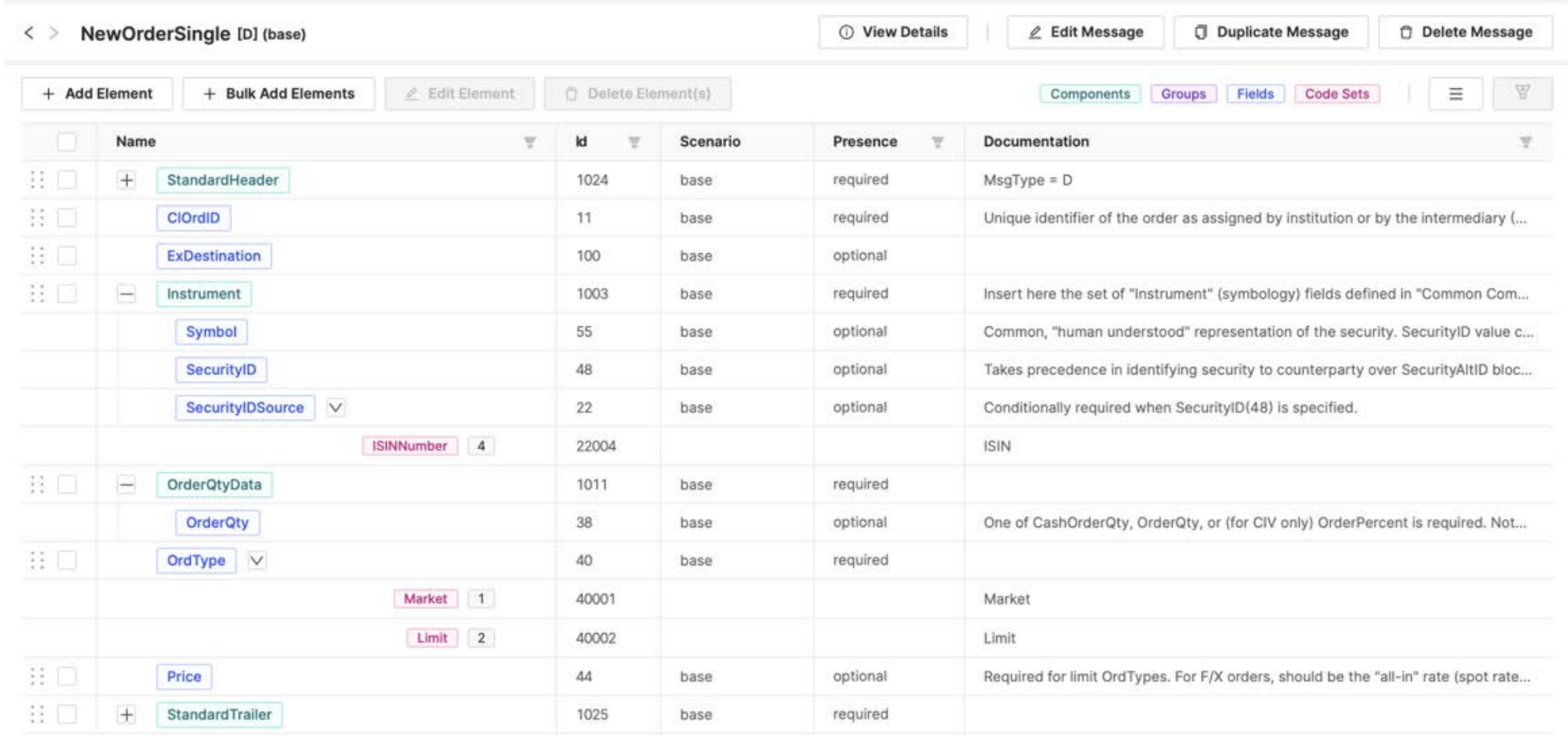
- NewOrderSingle(35=D)
  - ...
  - ClOrdID(11) - Type String
  - ...
  - ExDestination(100) - Exchange - Base Type String
  - ExDestinationIDSource(1133) - ExDestinationIDSourceCodeSet - Type char
  - ExDestinationType(2704) - ExDestinationTypeCodeSet - Type int
  - ExecInst(18) - ExecInstCodeSet - Type MultipleCharValue
  - ExpireDate(432) - LocalMktDate - Base Type String
  - ExpireTime(126) - UTCTimestamp - Base Type String
  - ExposureDuration(1629) - Type int
  - ExposureDurationUnit(1916) - OrderDelayUnitCodeSet - Type int
  - FinancingDetails - Component
  - ForexReq(121) - ForexReqCodeSet - Type Boolean
  - GTBookingInst(427) - GTBookingInstCodeSet - Type int
  - HandlInst(21) - HandlInstCodeSet - Type char
  - IOIID(23) - Type String
  - Instrument - Component
  - ...
  - OrdType(40) - OrdTypeCodeSet - Type char
  - OrderAttributeGrp - Group
  - OrderCapacity(528) - OrderCapacityCodeSet - Type char
  - OrderHandlingInstSource(1032) - OrderHandlingInstSourceCodeSet - Type int
  - OrderOrigination(1724) - OrderOriginationCodeSet - Type int
  - OrderQty2(192) - Qty - Base Type float
  - OrderQtyData - Component
  - ...
  - Price(44) - Price - Base Type float
  - ...

- ExecutionReport(35=8)
  - ...
  - CashMargin(544) - CashMarginCodeSet - Type char
  - ClOrdID(11) - Type String
  - ClOrdLinkID(583) - Type String
  - ...
  - CrossedIndicator(2523) - CrossedIndicatorCodeSet - Type int
  - CumQty(14) - Qty - Base Type float
  - Currency(15) - Currency - Base Type String
  - CurrencyCodeSource(2897) - CurrencyCodeSourceCodeSet - Type String
  - ...
  - ExecID(17) - Type String
  - ExecInst(18) - ExecInstCodeSet - Type MultipleCharValue
  - ExecPriceAdjustment(485) - Type float
  - ExecPriceType(484) - ExecPriceTypeCodeSet - Type char
  - ExecRefID(19) - Type String
  - ExecRestatementReason(378) - ExecRestatementReasonCodeSet - Type int - Union Reserved100Plus
  - ExecType(150) - ExecTypeCodeSet - Type char
  - ...
  - LastPx(31) - Price - Base Type float
  - LastQty(32) - Qty - Base Type float
  - ...
  - LeavesQty(151) - Qty - Base Type float
  - ...
  - OrdStatus(39) - OrdStatusCodeSet - Type char
  - OrdStatusReqID(790) - Type String
  - OrdType(40) - OrdTypeCodeSet - Type char
  - OrderAttributeGrp - Group
  - OrderCapacity(528) - OrderCapacityCodeSet - Type char
  - OrderCategory(1115) - OrderCategoryCodeSet - Type char
  - OrderEventGrp - Group
  - OrderHandlingInstSource(1032) - OrderHandlingInstSourceCodeSet - Type int
  - OrderID(37) - Type String
  - OrderOrigination(1724) - OrderOriginationCodeSet - Type int
  - OrderOwnershipIndicator(2679) - OrderOwnershipIndicatorCodeSet - Type int
  - OrderQty2(192) - Qty - Base Type float
  - OrderQtyData - Component
  - ...
  - Price(44) - Price - Base Type float
  - ...

Step 5: Upload to Orchestra Server (1)

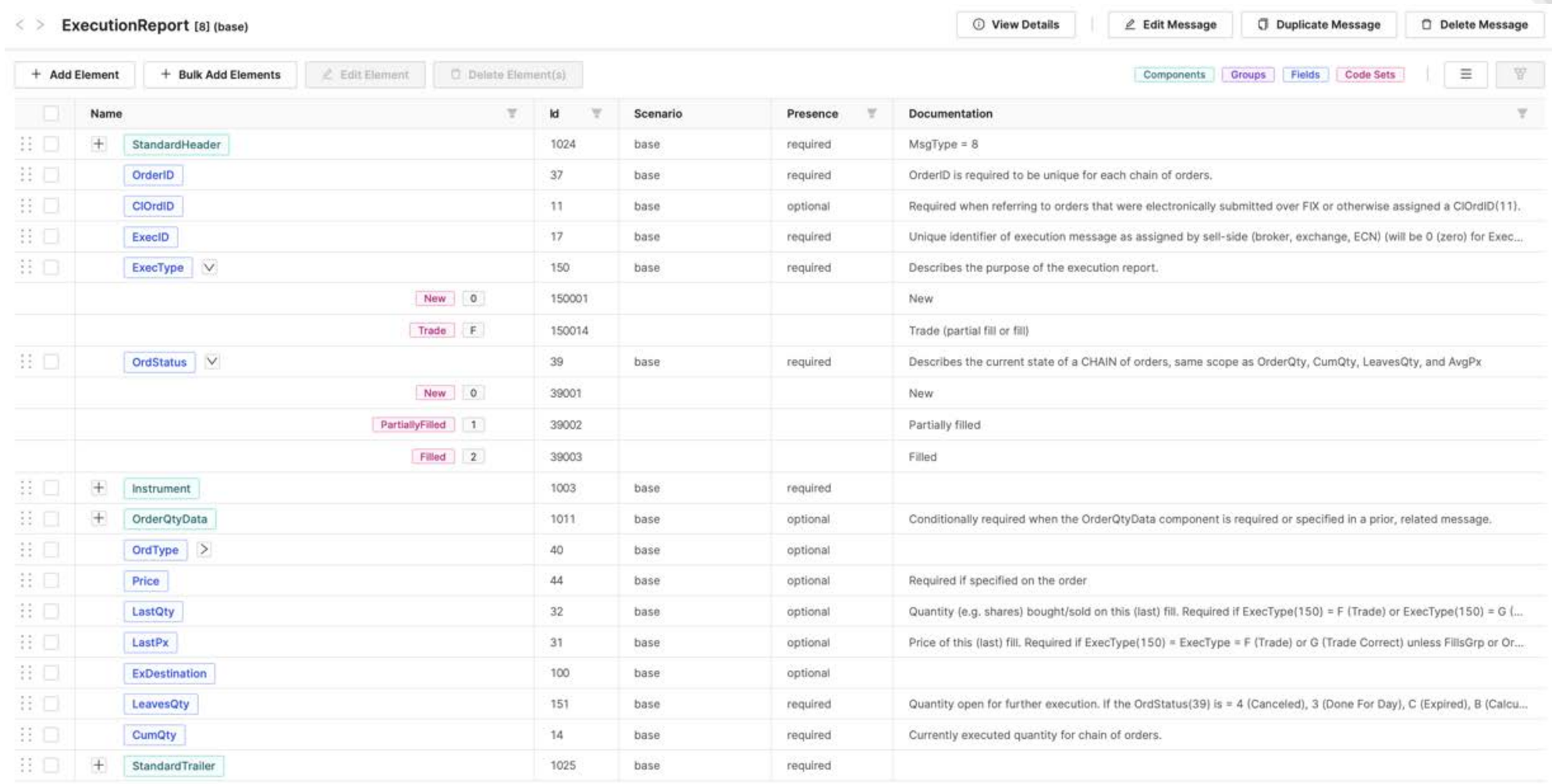


Step 5: Upload to Orchestra Server (2)



Name	Id	Scenario	Presence	Documentation
StandardHeader	1024	base	required	MsgType = D
ClOrdID	11	base	required	Unique identifier of the order as assigned by institution or by the intermediary (...
ExDestination	100	base	optional	
Instrument	1003	base	required	Insert here the set of "Instrument" (symbology) fields defined in "Common Com...
Symbol	55	base	optional	Common, "human understood" representation of the security. SecurityID value c...
SecurityID	48	base	optional	Takes precedence in identifying security to counterparty over SecurityAltID bloc...
SecurityIDSource	22	base	optional	Conditionally required when SecurityID(48) is specified.
ISINNumber 4	22004			ISIN
OrderQtyData	1011	base	required	
OrderQty	38	base	optional	One of CashOrderQty, OrderQty, or (for CIV only) OrderPercent is required. Not...
OrdType	40	base	required	
Market 1	40001			Market
Limit 2	40002			Limit
Price	44	base	optional	Required for limit OrdTypes. For F/X orders, should be the "all-in" rate (spot rate...
StandardTrailer	1025	base	required	

Step 5: Upload to Orchestra Server (3)



ExecutionReport [8] (base)

View Details | Edit Message | Duplicate Message | Delete Message

+ Add Element | + Bulk Add Elements | Edit Element | Delete Element(s)

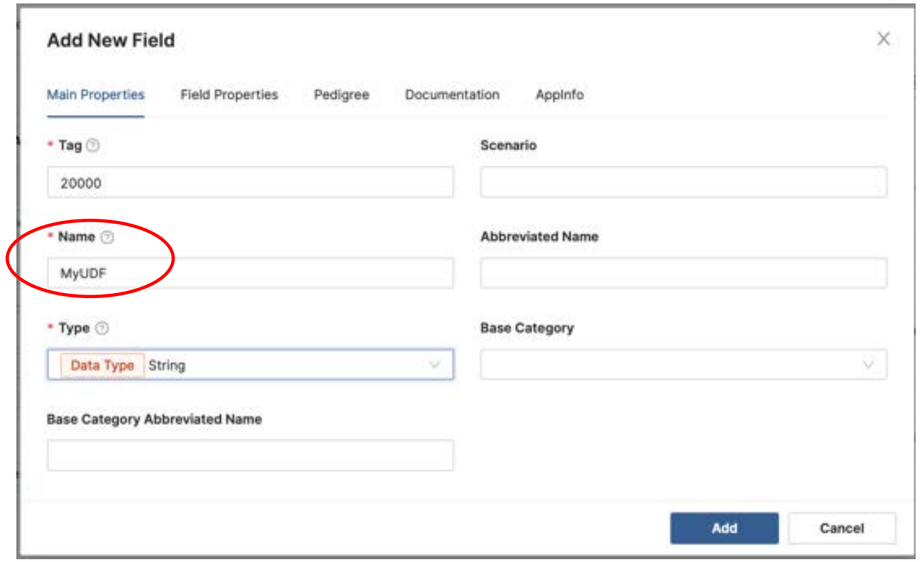
Components | Groups | Fields | Code Sets

Name	Id	Scenario	Presence	Documentation
+ StandardHeader	1024	base	required	MsgType = 8
OrderID	37	base	required	OrderID is required to be unique for each chain of orders.
ClOrdID	11	base	optional	Required when referring to orders that were electronically submitted over FIX or otherwise assigned a ClOrdID(11).
ExecID	17	base	required	Unique identifier of execution message as assigned by sell-side (broker, exchange, ECN) (will be 0 (zero) for Exec...
ExecType	150	base	required	Describes the purpose of the execution report.
New 0	150001			New
Trade F	150014			Trade (partial fill or fill)
OrdStatus	39	base	required	Describes the current state of a CHAIN of orders, same scope as OrderQty, CumQty, LeavesQty, and AvgPx
New 0	39001			New
PartiallyFilled 1	39002			Partially filled
Filled 2	39003			Filled
+ Instrument	1003	base	required	
+ OrderQtyData	1011	base	optional	Conditionally required when the OrderQtyData component is required or specified in a prior, related message.
OrdType	40	base	optional	
Price	44	base	optional	Required if specified on the order
LastQty	32	base	optional	Quantity (e.g. shares) bought/sold on this (last) fill. Required if ExecType(150) = F (Trade) or ExecType(150) = G (...
LastPx	31	base	optional	Price of this (last) fill. Required if ExecType(150) = ExecType = F (Trade) or G (Trade Correct) unless FillsGrp or Or...
ExDestination	100	base	optional	
LeavesQty	151	base	required	Quantity open for further execution. If the OrdStatus(39) is = 4 (Canceled), 3 (Done For Day), C (Expired), B (Calcu...
CumQty	14	base	required	Currently executed quantity for chain of orders.
+ StandardTrailer	1025	base	required	

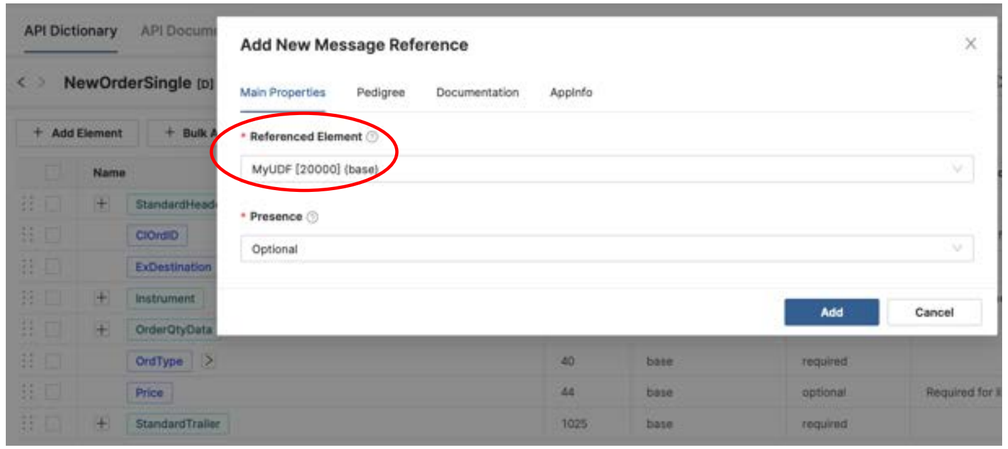
Step 6: Add UDF with Orchestra Server

1

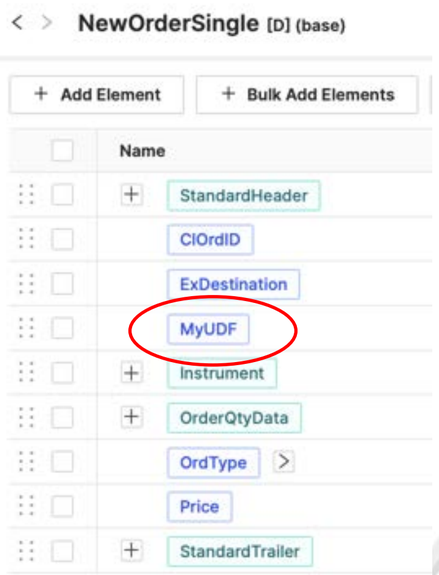
2



3

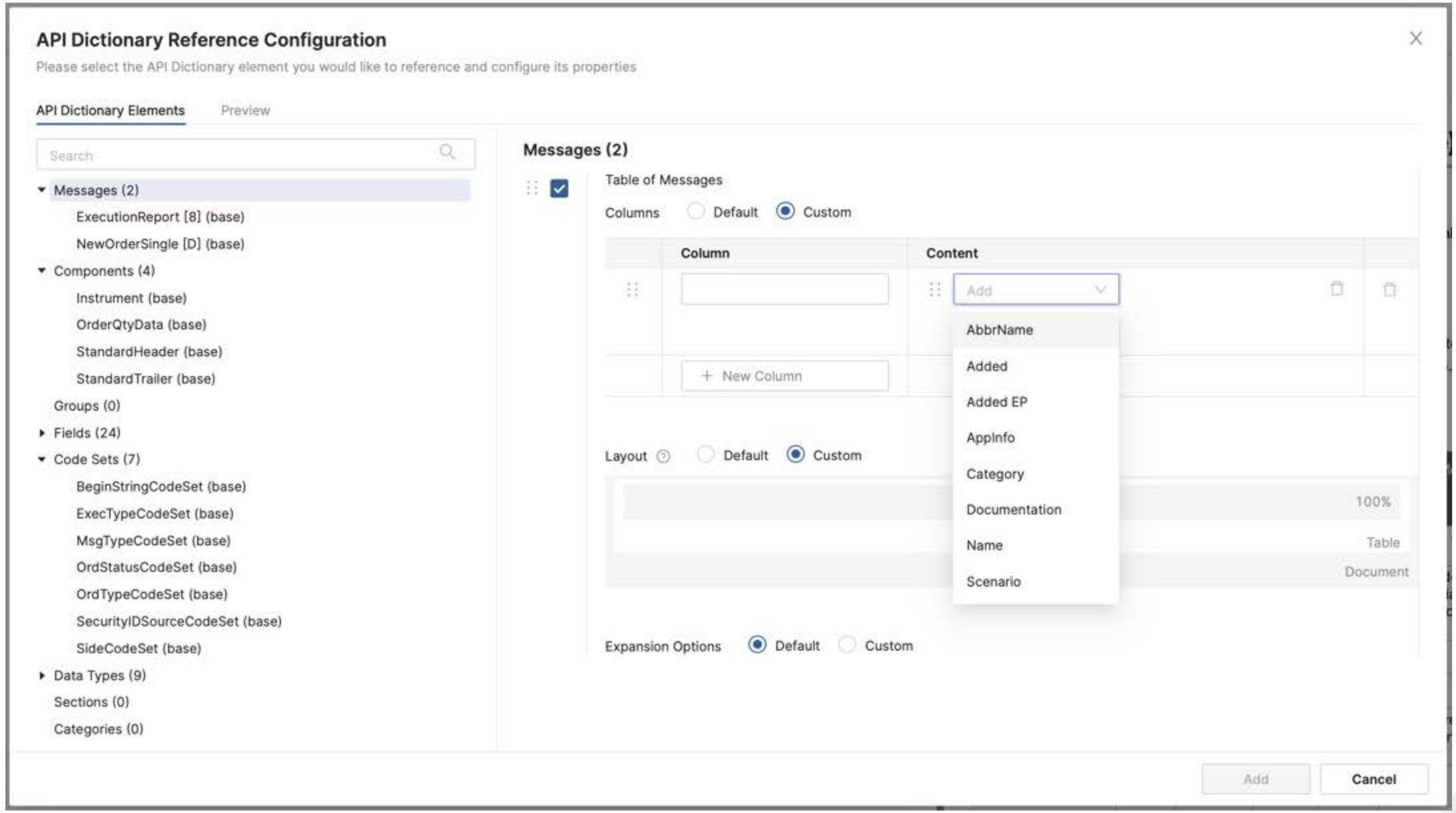


4

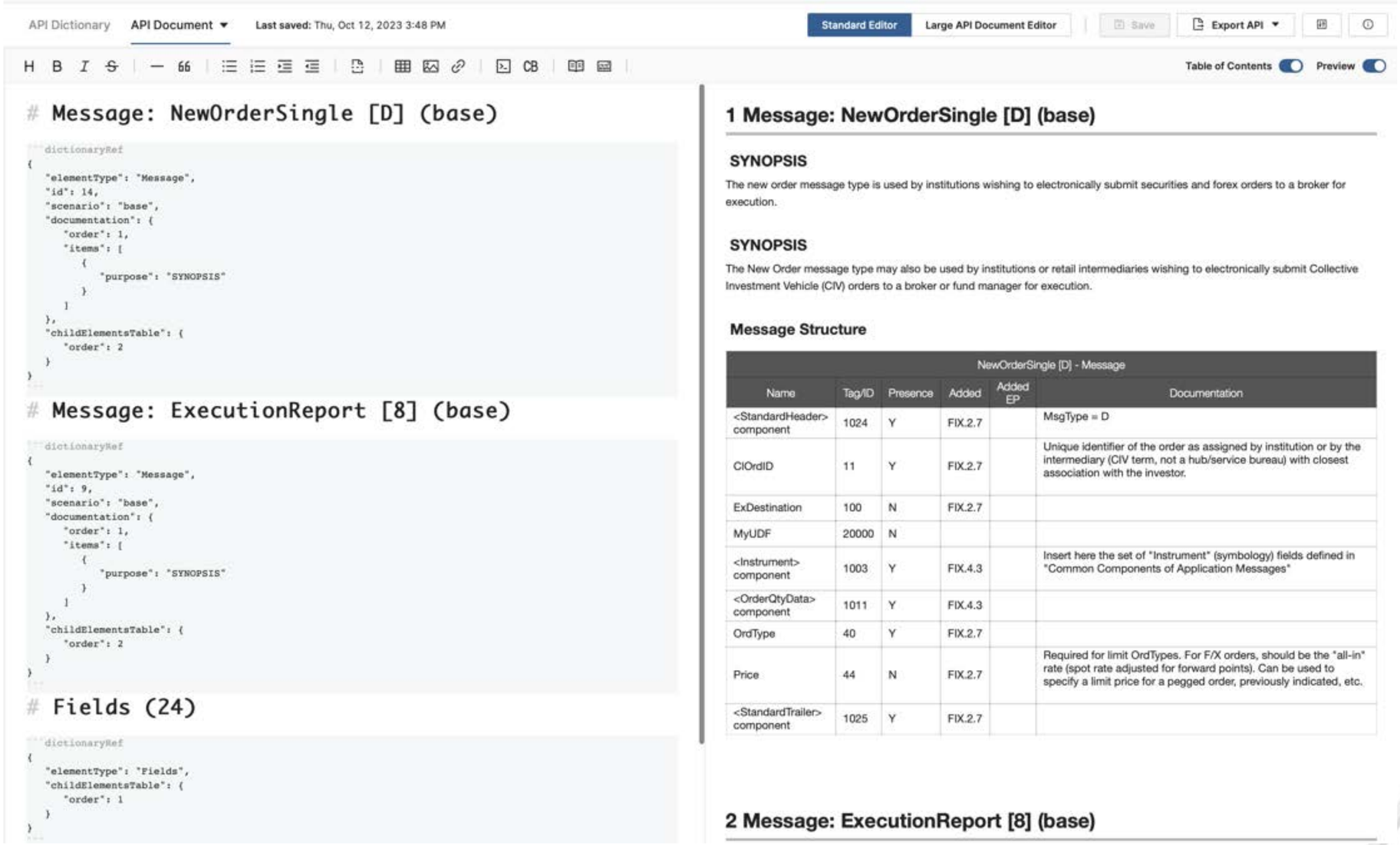


FIX TRADING COMMUNITY™
INDUSTRY-DRIVEN • INDEPENDENT • NEUTRAL

Step 7: Create specification document



Step 7: Create specification document



1 Message: NewOrderSingle [D] (base)

SYNOPSIS

The new order message type is used by institutions wishing to electronically submit securities and forex orders to a broker for execution.

SYNOPSIS

The New Order message type may also be used by institutions or retail intermediaries wishing to electronically submit Collective Investment Vehicle (CIV) orders to a broker or fund manager for execution.

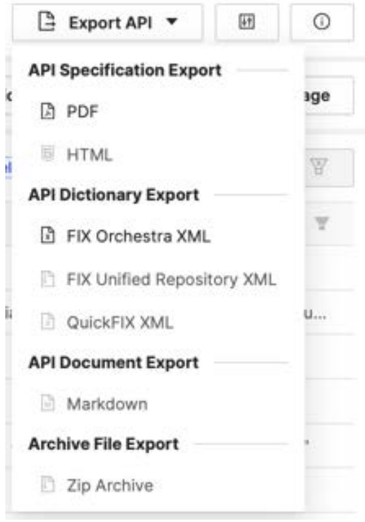
Message Structure

NewOrderSingle [D] - Message					
Name	Tag/ID	Presence	Added	Added EP	Documentation
<StandardHeader> component	1024	Y	FIX.2.7		MsgType = D
ClOrdID	11	Y	FIX.2.7		Unique identifier of the order as assigned by institution or by the intermediary (CIV term, not a hub/service bureau) with closest association with the investor.
ExDestination	100	N	FIX.2.7		
MyUDF	20000	N			
<Instrument> component	1003	Y	FIX.4.3		Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
<OrderQtyData> component	1011	Y	FIX.4.3		
OrdType	40	Y	FIX.2.7		
Price	44	N	FIX.2.7		Required for limit OrdTypes. For F/X orders, should be the "all-in" rate (spot rate adjusted for forward points). Can be used to specify a limit price for a pegged order, previously indicated, etc.
<StandardTrailer> component	1025	Y	FIX.2.7		

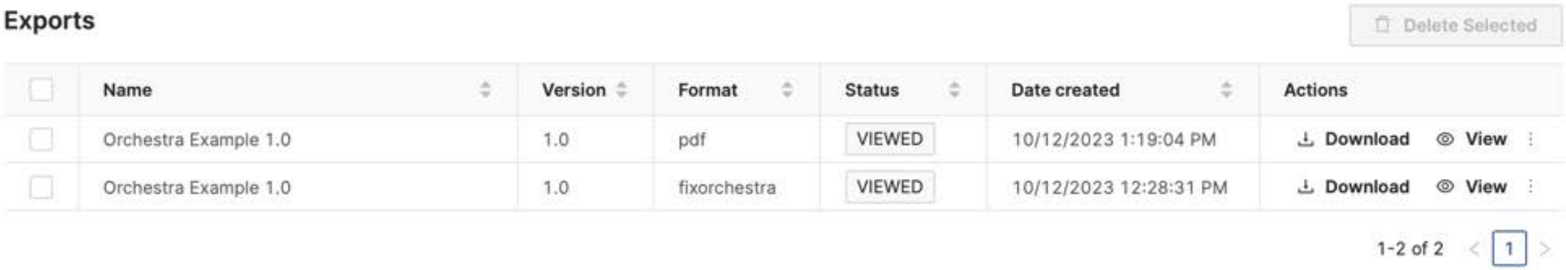
2 Message: ExecutionReport [8] (base)

Step 8: Export Rules of Engagement (XML)

1



2



3

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<fixr:repository xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:fixr="http://fixprotocol.io/2020/orchestra/repository" name="Orchestra Example" version="1.0">
    <fixr:metadata>
    <fixr:categories/>
    <fixr:sections/>
    <fixr:datatypes>
    <fixr:codeSets>
    <fixr:fields>
    <fixr:actors/>
    <fixr:components>
    <fixr:groups/>
    <fixr:messages>
    <fixr:concepts/>
</fixr:repository>
```

FIX TRADING COMMUNITY™
INDUSTRY-DRIVEN • INDEPENDENT • NEUTRAL

Step 8: Export Rules of Engagement (PDF)

NewOrderSingle [D] - Message					
Name	**Tag/ID**	**Presence**	**Added**	**Added EP**	**Documentation**
<StandardHeader> component	1024	Y	FIX.2.7		MsgType = D
ClOrdID	11	Y	FIX.2.7		Unique identifier of the order as assigned by institution or by the intermediary (CIV term, not a hub/service bureau) with closest association with the investor.
ExDestination	100	N	FIX.2.7		
MyUDF	20000	N			
<Instrument> component	1003	Y	FIX.4.3		Insert here the set of "Instrument" (symbology) fields defined in "Common Components of Application Messages"
<OrderQtyData> component	1011	Y	FIX.4.3		
OrdType	40	Y	FIX.2.7		
Price	44	N	FIX.2.7		Required for limit OrdTypes. For F/X orders, should be the "all-in" rate (spot rate adjusted for forward points). Can be used to specify a limit price for a pegged order, previously indicated, etc.
<StandardTrailer> component	1025	Y	FIX.2.7		

How to use Orchestra Server?
Orchestra Scenarios

FIX TRADING COMMUNITY™
INDUSTRY-DRIVEN • INDEPENDENT • NEUTRAL

Orchestra Scenarios

- Orchestra supports scenarios for the main elements of the schema:
  - Messages, Groups, Components, Fields, Code Sets, Datatypes (as of v1.1).

- A scenario is a copy of another element with different characteristics
  - Message/group/component scenarios can have different subsets of fields and different presence attributes.
  - Field scenarios can have different code sets.
  - Datatype scenarios can have different value ranges.

- Scenarios are the machine-readable equivalent to manual text, e.g.
  - Execution report fields and repeating groups that only apply to trades.
  - Instrument fields and values that only apply to specific asset classes.

Orchestra Scenarios

- Scenarios support the definition of granular workflows
  - ExecutionReport(35=8) messages to confirm a new order do not need fields like LastQty(32) and LastPx(31) or ExecType(150) values F=Trade, G=Trade Correct, H=Trade Cancel.
- Conditional requirements may be a better choice for single fields
  - Example: Stop price only relevant when order type identifies a stop order
  - Using scenarios requires a message level scenario only for stop orders
  - Better: StopPx(99) conditionally required when OrdType(40)=3 or 4
- Datatype scenarios (Orchestra v1.1):
  - Orchestra v1.0 only supports a base type, e.g. “int”, for derived datatypes, e.g. “SeqNum” which has to be positive
  - Orchestra v1.1 allows to define scenarios for datatypes, e.g. for “int”
    - scenario=“base” for all integer values
    - Scenario=“SeqNum” for positive integer values