FIX Recommended Practices

Onboarding with Orchestra

FIX TRADING COMMUNITY™
INDUSTRY-DRIVEN • INDEPENDENT • NEUTRAL

FINANCIAL INFORMATION EXCHANGE (FIX)

RECOMMENDED PRACTICES

FIX MOST Working Group

Onboarding with Orchestra

July 2023

Version 1.0 (FINAL)

TABLE OF CONTENTS

DOCUMENT HISTORY

Revision	Date	Author	Revision Comments
V1.0	8 May 2023	Francesco Lo Conte, Esprow	Final draft version
	15 June 2023	Hanno Klein, GTC co-chair	Version for public comment
	20 July 2023	Hanno Klein, GTC co-chair	Final version after public review (no changes)

1 Executive Summary

The process of connecting trading counterparties is mostly carried out manually by exchanging connectivity details and carrying out manual connectivity checks between counterparties.

One of the objectives of the Orchestra Technical Standard developed by FIX is to streamline this process to enable faster onboarding. Since the Orchestra standard allows encoding of the *rules of engagement* between two counterparties, in a machine-readable form, including connectivity and message-translation information, it offers many opportunities for automation and efficiencies.

This document covers how Orchestra can be used to introduce efficiencies in the onboarding process.

2 Objectives

The purpose of this document is to define industry practices for the establishment and certification of FIX and non-FIX connectivity between trading counterparties and to highlight the support provided by the new Orchestra standard in simplifying and streamlining this process.

3 Scope

This document covers the common steps involved in onboarding trading counterparties from a technical perspective, and how these aspects benefit from the new Orchestra standard.

Business onboarding is covered at a high level only.

4 Target Audience

This document is aimed at practitioners involved in certifying connectivity between trading counterparties during the onboarding process.

5 Authors

This document is based on work undertaken by the MOST Working Group in 2022, to identify areas in the monitoring, onboarding, simulation, and testing processes that can benefit from new capabilities enabled by Orchestra.

6 Background

The findings of the FIX MOST Working Group are that most market participants carry out similar steps when onboarding counterparties. These fall generally into two broad categories: *business* onboarding and *technical* onboarding.

The goal of the working group was to identify the common steps that make up the backbone of the onboarding process. They are documented and described below to provide practitioners with an actionable roadmap they can use to guide their onboarding activities.

Broadly speaking, these are the steps involved in the onboarding process:

1. Business Onboarding

1.1. Legal Onboarding, KYC

1.1.1. Ensuring a counterparty is good to deal with, signing necessary documents.

1.2. Project Initiation

1.2.1. Internal ticket, email with counterparty/vendor contacts and initial business requirements.

2. Technical Onboarding

2.1. Requirements Gathering

2.1.1. Kick-Off Call

2.1.1.1. Discuss connectivity, vendors involved, timelines, clarify business requirements.

2.1.2. API Specifications Exchange / Review ❶

2.1.2.1. Send out technical documentation.

2.1.3. Follow-Up Call

2.1.3.1. Discuss and address any issues.

2.2. UAT Setup

2.2.1. Network Connectivity Setup ❷

2.2.1.1. Configure the connection and share the details with the counterparty.

2.2.2. Trading Engine Setup ❸

2.2.2.1. Configure the trading gateway for the specific API and share the details with the counterparty.

2.2.3. Mappings ❹

2.2.3.1. Configure field mappings/translations if applicable.

2.2.4. EMS / OMS / Back-Office configuration

2.2.4.1. Configure EMS/OMS/Back-Office software, static data, to enable testing.

2.3. Connectivity Certification

2.3.1. Development

2.3.1.1. If required on the back of the requirement gathering stage.

2.3.2. Conformance Certification ❺

2.3.2.1. Follow test script(s).

2.3.3. Additional development

2.3.3.1. If required on the back of the conformance certification.

2.3.4. Re-certification ❺

2.3.4.1. If required post development.

2.3.5. Sign Off

2.3.5.1. Send out certification results and notes to a counterparty to sign off.

2.4. Production Setup

2.4.1. Network Connectivity Setup ❷

2.4.1.1. Configure the connection and share the details with the counterparty. Ideally configuration cloned from UAT.

2.4.2. Trading Engine Setup ❸

2.4.2.1. Configure the trading gateway for the specific API and share the details with the counterparty. Ideally configuration cloned from UAT.

2.4.3. Mappings ❹

2.4.3.1. Configure field mappings/translations, if applicable. Ideally configuration cloned from UAT.

2.4.4. EMS / OMS / Back-Office configuration

2.4.4.1. Configure EMS/OMS/Back-Office, static data, etc. Ideally configuration cloned from UAT.

2.4.5. Production Sign Off

2.4.5.1. Sending out a configuration summary.

2.5. Go Live

2.5.1. Pilot Order

2.5.1.1. First production order closely monitored.

2.5.2. Documentation Update

2.5.2.1. Updating necessary documentation internally for better support.

2.5.3. Hand Over to Support Teams ❻

2.5.3.1. Connectivity monitoring begins.

In the table above, we also identified several areas that would benefit from support from the new Orchestra standard. These areas are identified with a number, and they are described as it follows:

In the list above, we also identified several areas that would benefit from support from the new Orchestra standard. These areas are identified with a number, and they are described as follows:

❶ The PDF documents that describe the API can be generated automatically from an Orchestra file.

❷ Network setup can be described using Orchestra's Interfaces files.

❸ Trading gateways can be configured by loading a Orchestra file with all message definitions, including Scenarios and Workflows to validate different trading workflows (e.g., equities vs FX).

❹ Orchestra's Mappings files that describe the logic to transform counterparties' inbound messages into internal normalized messages (and vice versa) can be used to automate message translation.

❺ Using Orchestra's Workflows it is possible to generate automated API tests to check the API before releasing it to production, and then to generate automated onboarding scripts to onboard each counterparty automatically.

❻ Orchestra specifications can be loaded into monitoring systems to monitor message traffic and identify incompatibilities with the API standard (i.e., machine-driven message validation).

The goal of Orchestra is interoperability. Orchestra files can be used with various tools to inform counterparties of changes to the API, i.e., rather than relying on manual checks of the PDF specifications.

7 Implementation Details

In this section we provide details on how to use the Orchestra standard to implement automation in the onboarding process. It should be noted that several areas of the Orchestra standard are under development by the Orchestra and MOST working groups and may not be fully available at the time of this writing.

7.1 Definitions

1. **Orchestra Standard**: The schema definition that constitutes Orchestra. It includes the Specifications schema, Interfaces schema, and Mappings schema, described below.
2. **Orchestra Specifications ("Specifications")**: An XML file describing the elements of an API, including messages, components, groups, fields, datasets, etc.
3. **Orchestra Interfaces ("Interfaces")**: An XML file describing the connectivity details to an endpoint.
4. **Orchestra Mappings ("Mappings")**: An XML file describing how messages are converted between two related APIs, e.g., from inbound messages from counterparties to an internal normalized API.

7.2 Using Orchestra "Interfaces" for Network Connectivity Setup

✓ **Warning**

The Interfaces feature of Orchestra is under active development and not ready for production usage.

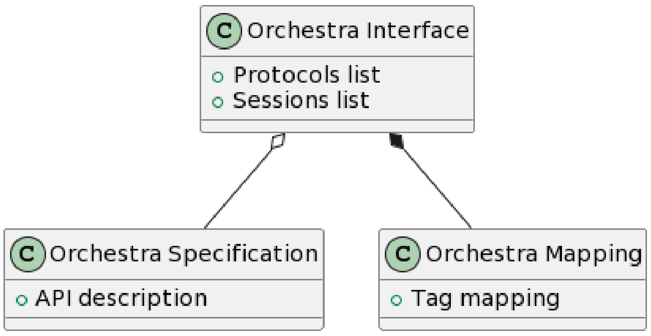
7.2.1 Orchestra Interfaces

The Orchestra Standard defines the Interfaces schema, a way of defining connectivity details to endpoints in XML (i.e., FIX gateways). Interfaces definitions are stored in their own XML files, separately from Specifications. Interfaces are used to describe the protocols available, and the (client) sessions that use those protocols, with specific details such as IP/port. They allow processing of connectivity details by internal systems, to increase automation, and avoid mistakes during the network connectivity setup. They also provide a standard format to exchange connectivity information with counterparties to allow the same level of automation by them.

Sure, here is a description of each element in the diagram using bullet points:

Orchestra Interface:

- Describes multiple protocols.
- Lists the Sessions that implement these protocols.
- Each protocol references a specific Orchestra Specification.
- Each session references a specific Orchestra Mapping.



7.2.2 How to use Orchestra Interfaces

An Interfaces file defines the top four (4) layers of the OSI connectivity model: Application (7), Presentation (6), Session (5), and Transport (4), together with additional details, such as TLS encryption keys.

Interface files can be stored in a central repository to create a hub for all connectivity information. They can be stored in a configuration management system (e.g., Git) to provide an audit trail of changes to connectivity details.

Since Interfaces files are written in machine-readable XML, they can be processed by systems, like FIX gateways, to self-configure, which allows large-scale DevOps changes to connectivity in an automated and efficient way.

Tip

During the onboarding process, an Interfaces file should be created for the counterparty being onboarded and shared with the counterparty to enable automation by them as well. The Interfaces file should be stored together with other documents related to the onboarding and preferably within a system that facilitates discovery and management of this information, including for DevOps and audit purposes.

7.2.3 Example of an Orchestra Interfaces (provisional)

An example of an Orchestra Infaces file can be found in the Orchestra GitHub repository at this URL.:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<fixi:interfaces xmlns:dcterms="http://purl.org/dc/terms/" xmlns:fixi="http://fixprotocol.io/2020/orchestra/interfaces"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://fixprotocol.io/2020/orchestra/interfaces ../../../main/resources/xsd/interfaces.xsd">
        <fixi:metadata>
                <dcterms:subject>Service offerings and sessions example file</dcterms:subject>
                <dcterms:date>2019-08-07T09:30:00Z</dcterms:date>
        </fixi:metadata>
        <fixi:interface name="Private">
                <!-- one or more service offerings, with local orchestration file or internet address -->
                <fixi:service name="orderEntry" orchestration="https://mydomain.com/orchestra/orderEntry.xml"/>
                <!-- the protcol stack -->
                <fixi:userInterface name="ATDL" orchestration="https://mydomain.com/orchestra/algo.xml"/>
                <fixi:encoding name="TagValue"/>
                <fixi:sessionProtocol name="FIXT.1.1" reliability="recoverable"
orchestration="https://mydomain.com/orchestra/session.xml">
                        <fixi:annotation>
                                <fixi:documentation langId="en-us">FIX session protocol</fixi:documentation>
                        </fixi:annotation>
                </fixi:sessionProtocol>
                <fixi:transport name="TCP"/>
                <fixi:sessions>
                        <fixi:session name="XYZ-ABC">
                                <!-- inherits services and protocols from interface -->
                                <!-- alternate addresses are supported -->
                                <fixi:transport address="10.96.1.2:567" use="primary"/>
                                <fixi:transport address="10.96.2.2:567" use="secondary"/>
                                <fixi:identifier name="SenderCompID">XYZ</fixi:identifier>
                                <fixi:identifier name="TargetCompID">ABC</fixi:identifier>
                                <fixi:activationTime>2019-08-07T09:30:00Z</fixi:activationTime>
                        </fixi:session>
                </fixi:sessions>
        </fixi:interface>
        <fixi:interface name="OrderRouting">
                <fixi:service name="orderRouting" orchestration="https://mydomain.com/orchestra/orderRouting.xml"/>
                <fixi:encoding name="GPB" messageSchema="file://something.proto"></fixi:encoding>
                <fixi:protocol name="TLS" version="1.2" layer="transport"></fixi:protocol>
                <fixi:sessions>
                        <fixi:session name="OR1">
                        <fixi:securityKeys><![CDATA[
-----BEGIN CERTIFICATE-----
<certificates omitted for brevity>
-----END PRIVATE KEY----- ]]></fixi:securityKeys>
                        </fixi:session>
                </fixi:sessions>
        </fixi:interface>
</fixi:interfaces>
```

The Interfaces file in the example describes the following configuration:

1. Three interfaces named Private, Public, and OrderRouting are configured.
2. The Private interface references three orchestrations, which would be Orchestra files in the repository2016 schema. Orchestration files can either be local files or accessible through a web interface.
3. Private interface defines a protocol stack with "orderRouting" application layer, a FIXatdl user interface for algorithm controls, a session layer, and a transport layer.
4. One session is configured for the Private interface. The configuration includes identifiers, and two transport addresses.
5. Session "XYZ-ABC" has an activation time. This supports sending a new or changed interfaces file in advance of the session being authorized.
6. The Public interface shows how a multicast is specified.
7. Session "OR1" shows an example of security keys, including public certificates and a private key. They are encoded in RFC 7468 format.

7.3 Using Orchestra “Mappings” for Message Translation

Warning

The Mappings feature of Orchestra is under active development and not ready for production usage.

7.3.1 Orchestra Mappings

An Orchestra Mappings file describes how messages received from a counterparty on a specific endpoint are translated to be adapted to the internal (normalized) message format. This allows, for example, support of counterparty-specific message fields that are mapped to internal standard tags (e.g., for algorithmic trading). Mappings also support the inverse translation when messages are sent back to the counterparty.

7.3.2 How to use Orchestra Mappings

As an example, a counterparty may be utilizing trading algorithms provided by our firm. However, they prefer to use their own FIX tags to specify certain parameters. To support this requirement, incoming messages will have to be translated from the counterparty’s own specific algo tags to our internal equivalent tags. The Mappings file will look something like this:

Mappings Details			
	“clientId”		0123456789
	“clientName”		"CLIENT123"
	“environment”		“UAT”
Incoming Translations			
	"messageType"		"D"
	Translations		
		9990	“(\d+)\.(\d{2})?(\d{0,2})$ -> $1.$2$3.00”
		9991	“^(.*)$ -> ISO_$1”
		9992	9995
Outgoing Translations			
	"messageType"		“8”
	Translations		
		7990	“\.\d{2}$”
		7991	“^ISO_”
		7995	7992

Mappings files can be stored in a central repository to create a hub for all connectivity information. They can be stored in a configuration management system (e.g., Git) to provide an audit trail of changes to connectivity details.

Since Mappings files are written in machine-readable XML, they can be processed by systems, like FIX gateways, to self-configure, which allows large-scale DevOps changes to connectivity in an automated and efficient way

Tip

During the onboarding process, a Mappings file should be created for the counterparty being onboarded and shared with the counterparty to enable automation by them as well. The Mappings file should be stored together with other documents related to the onboarding and preferably within a system that facilitates discovery and management of this information, including for DevOps and audit purposes.

7.4 Using Orchestra "Specifications" for Trading Engine Setup

7.4.1 Orchestra Specifications

An Orchestra Specifications file describes an API in XML, including its messages, components, groups, fields, and data types. It is the new equivalent of a FIX Unified Repository dictionary file, or, for example, of a QuickFIX XML file. However, a Specifications file contains much more information about the API than any other format before it, allowing for much more automation and interoperability. Refer to the Orchestra standard for more information.

7.4.2 How to use Orchestra Specifications

Trading gateways need to be configured with the rules of engagement (RoE), e.g., FIX gateways are configured with the FIX RoE. This can be done with a Specifications file describing the messages supported by the API.

Since Orchestra also has the concept of Scenarios, gateways for different APIs can be configured using the same Specifications file. For example, a FIX Specifications file can describe two scenarios for the NewOrderSingle(35=D) message, e.g., one for US equities and one for European equities, where the NewOrderSingle(35=D) message differs slightly for each case. FIX gateways for US equities and European equities would use the same Specifications file but only consider their corresponding message scenarios.

Tip

During the onboarding process, a Specifications file should be created for the counterparty being onboarded and shared with the counterparty to enable automation by them as well. The Specifications file should be stored together with other documents related to the onboarding and preferably within a system that facilitates discovery and management of this information, including for DevOps and audit purposes.

7.5 Using Orchestra API Specifications for OMS/EMS Configuration

Although OMS/EMS/Back-Office systems are usually proprietary in nature, they share common workflows, such as STP, DMA, DSA, and so on. The Orchestra standard, with its support for rules of engagement, scenarios, workflows, interfaces, and mappings, has the potential to provide these systems with most configuration they need, in a standardized and interoperable format. This makes collaboration easier across teams and skills portable across systems.

7.6 Using Orchestra API Specifications for Connectivity Monitoring

Since a Specifications file describes the rules of engagement between counterparties, it is suitable to be used as a blueprint for monitoring and checking the connectivity with counterparties, to validate their compliance with the rules of engagement. Systems that monitor connectivity should consider supporting Specifications files to verify, highlight inconsistencies with the RoE, auto-repair messages (where possible), and alert production support teams.

8 Considerations

The latest release of the Orchestra standard is version 1.0. A release candidate for version 1.1 is being worked on (https://github.com/FIXTradingCommunity/fix-orchestra/tree/v1.1RC1). Some of the features described in this document may not have been fully released at the time of writing. We would invite the reader to check support for the features described here when approaching this.

9 Conclusion

The Orchestra standard is evolving to integrate as much information as possible to support automation of the onboarding pipeline. Onboarding practitioners should strive to adopt this new machine-readable standard and the tools in its ecosystem, to achieve as much automation as possible. Besides the obvious savings in costs and time, Orchestra also offers opportunities for creating information hubs for client connectivity, for algo tags, and important audit trails that improve every firm's regulatory compliance and audit requirements.

Today, the Orchestra standard supports rich expressiveness for API features, scenarios, workflows, interfaces, and mappings. This is a leap forward from the previous generation, such as the FIX Unified Repository standard. The vision for Orchestra is to become the standard technology platform for trading connectivity interoperability, for the FIX community, by the FIX community, for the FIX protocol and beyond, to support the evolving trading industry.

10 References

10.1 Technical Onboarding Checklist

This checklist can be used to support technical onboarding of counterparties.

Requirements Gathering		
☐	Contact a counterparty to arrange a kick-off call.	
☐	Send API specifications to the counterparty, including machine-readable Orchestra file.	
☐	Agenda for the kickoff call:	
	☐	Review, discuss, and agree API specifications.
	☐	Discuss physical connectivity requirements.
	☐	Discuss vendors involved.
UAT Setup		
☐	Connectivity Setup *[Consider the same for Production Setup]*	
	☐	Order lines.
	☐	Configure VPN tunnels.
	☐	Configure firewalls.
☐	FIX Engine Setup *[Consider the same for Production Setup]*	
	☐	Provision FIX gateway instance.
	☐	Configure FIX gateway.
	☐	Share details of connectivity to FIX gateway with the counterparty.
☐	OMS/EMS/Back-Office Configuration *[Consider the same for Production Setup]*	
	☐	Configure OMS.
	☐	Configure EMS.
	☐	Configure back-office systems.
	☐	Configure other systems.
Connectivity Certification		
☐	Carry out development as required.	
☐	Configure certification system.	
☐	Contact the counterparty to arrange connectivity certification.	
☐	Complete connectivity certification.	
☐	*Carry out additional development as required.*	
☐	*Contact the counterparty to arrange connectivity re-certification if necessary.*	
☐	Sign off on certification.	

[continued on next page]

Production Setup		
☐	Connectivity Setup	
	☐	Order lines.
	☐	Configure VPN tunnels.
	☐	Configure firewalls.
☐	FIX Engine Setup	
	☐	Provision FIX gateway instance.
	☐	Configure FIX gateway.
	☐	Share details of connectivity to FIX gateway with the counterparty.
☐	OMS/EMS/Back-Office Configuration	
	☐	Configure OMS.
	☐	Configure EMS.
	☐	Configure back-office systems.
	☐	Configure other systems.
Go Live		
☐	Contact the counterparty to arrange pilot order.	
☐	Update internal databases.	
☐	Hand over connectivity to the support team.	

www.fixtrading.org