Nasdaq Calypso

FXall Integration Guide

Version 4.11.0

Revision 16.0
January 2025
Approved

Document History

Revision	Published	Summary of Changes
1.0	January 2020	First edition for version 1.0.1
2.0	August 2020	Updates for version 2.1.0
3.0	September 2020	Updates for version 2.1.1
4.0	November 2020	Updates for version 2.2.0 – Added Order integration.
5.0	February 2021	Updates for version 2.3.0, 2.4.0
6.0	May 2021	Updates for version 2.4.6 – Added Order Workflow Setup.
7.0	February 2022	Updates for version 3.0.0, 3.0.1
8.0	March 2022	Updates for version 2.9.0
9.0	July 2022	Updates for version 3.4.0, 3.4.1
10.0	December 2022	Updates for version 3.7.0
11.0	January 2023	Updates for version 3.8.0
12.0	August 2023	Updates for version 3.9.0 - Block orders are no longer using the AMBlockOrder workflow. They are using the AMOrder workflow instead.
13.0	January 2024	Updates for version 4.3.0 (compatibility with version 18).
14.0	March 2024	Updates for version 4.5.0 – Added TradeDate mapping.
15.0	October 2024	Updates for version 4.9.0 – Change to Spot / Forward identification.
16.0	January 2025	Updates for version 4.11.0 – Added engine parameter OPTIONAL_FEATURE.

This document provides setup information for the FXall interface.

NOTE: The Calypso License to use this Calypso Integration Module does not include a license for any third-party data services to which this module can interface. Clients are responsible for contracting with the appropriate third-party data service(s) prior to using this Calypso Integration Module

Table of Contents

Introduction

This document describes the Calypso FXall Interface setup.

The FXall interface allows an End User Firm (Buy-Side) to import trades which have been booked through the FXall Trading Terminal provided from Refinitiv. It allows sending Calypso orders to FXall. It also allows sending RFQ requests initiated on the Calypso ePortals platform (eDealing in version 16.1) to FXall.

The Calypso FXall integration connects to Refinitiv FXall using a FIX interface.

This document describes the configuration required to setup the workflows, etc. for the FXall module to run successfully.

Make sure that environment property ASSET_MANAGEMENT=true.

1.1 FXall Trade Interface

Once the connectivity is setup, a trader can book a trade through the FXall Trading Terminal, and FXall sends that as a FIX message to the Calypso FXall interface. The message then flows through the configured Calypso workflows which route the message through the appropriate stages to create a Calypso trade.

The FXall interface supports the following:

Product Types:

- FX Spot
- FX Outright
- FX NDF
- FX Swap
- FX NDF Swap
- FX Option

Trade lifecycle:

- NEW

Fix Messages:

The interface supports the following FXall messages:

- Execution Report (Trade): Once the trade is executed within the FXall Trading Terminal, FXall sends a message over the FIX interface. A Bilateral trade is created in Calypso to represent the trade between the two parties (Dealer vs. Buy-Side)

1.2 FXall Orders

Product Types:

Same as FXall Trade Interface

Trade lifecycle:

- NEW / CANCEL

1.3 FX all Request for Quote (RFQ) Interface

With this feature Calypso ePortals customers can request a quote from different liquidity providers:

- Provide ePortals customers the ability to trade directly with liquidity providers
- Automatically hedge / create back-to-back trades in real time basis to eliminate market risk

Supported Product Types:

- FX Spot
- FX Outright
- FX NDF
- FX Swap

Supporter Message Types:

Message	Type	Description
QuoteRequest <QuotReq>	R	Quote Request prior to placement of an order.
Quote <Quot>	S	A response to a Quote Request.
QuoteCancel <QuotCxl>	Z	Used to cancel a Quote Request.
QuoteRequestReject <QuotReqRej>	AG	Used to reject a Quote Request.
New Order Single <Order>	D	Submit a request that a trade be performed.
New Order MultiLeg <NewOrdMleg>	AB	Used to submit orders for securities that are made up of multiple legs. This message must be used for FX SWAP orders.
Execution Report <ExecRpt>	8	Indication of the execution of a trade.

The following messages are sent by the customers:

- Quote Request R
- Quote Cancel Z
- New Order Single D
- New Order MultiLeg AB

The following messages are sent to the customers:

- Quote S
- QuoteRequestReject AG
- Quote Cancel Z
- Execution Report 8

Please see below for additional setup requirements specific to FXall RFQ.

Installation

2.1 Software Requirements

2.1.1 Supported JRE Versions

Please use the appropriate JRE version depending on the supported version for the base Calypso release you are running.

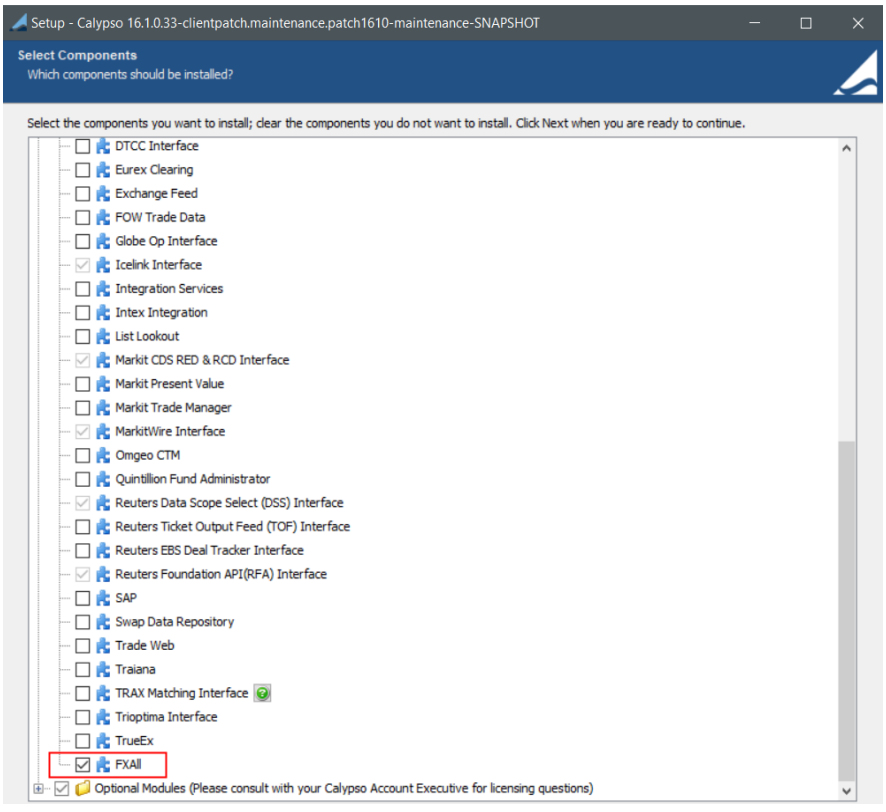
2.1.2 Supported Calypso Versions

The module supports specific versions of Calypso. In addition, your implementation must have the current Hotfixes applied. Before downloading the FXall module, please refer to the Calypso FXall Release Notes to determine which module versions are applicable for your implementation.

2.2 Installation Instructions

2.2.1 Calypso Components

Follow the Calypso Installation Guide to install Calypso. Check the FXall interface during the installation.



2.2.2 Data Uploader Installation

The use of the FXall interface requires the Data Uploader. All subsequent instructions assume that all Data Uploader installation steps have been completed successfully. This includes:

- Choosing Data Uploader from within the Calypso Installer (or using the patch tool to add it to an existing installation)
- Applying the Gateway SchemaBase/SchemaData to your database
- Setting up the GATEWAYMSG and UPLOADESOURCEMSG workflows
- Setting up Task Station tabs for the workflows above

Please refer to the Calypso Data Uploader Integration Guide for specific installation and configuration information. You must install and configure Calypso Data Uploader prior to configuring FXall.

2.2.3 Setup Config Data using Execute SQL

The following files will be loaded when you run Execute SQL from <calypso home>/bin/dbscripts:

- SchemaBase.xml
- GatewaySchemaBase.xml
- FIXSchemaData.xml
- FXallSchemaData.xml

2.3 Message Workflow Setup

The FXall module uses the UPLOADSOURCEMSG and GATEWAYMSG workflows when importing messages. These should have been setup as part of the Data Uploader Setup Guide.

Messages from the UPLOADSOURCEMSG workflow are translated from the external message format into Calypso's internal format and placed in the GATEWAYMSG workflow. The GATEWAYMSG workflow then translates the internal format, performs verifications, and saves the trade to the database.

2.4 Task Station Setup

The FXall module uses the Data Uploader Framework to create task station entries for all the messages and exceptions that are encountered. The user can view / reprocess the messages that failed in validation from the task station.

Please refer to the Calypso Data Uploader Integration Guide for how to add the appropriate messages and exceptions to the Task Station.

Setup Requirements

3.1 Legal Entity Mapping

The incoming FXall FIX messages contain Legal Entity identifier for counterparty involved in the trade.

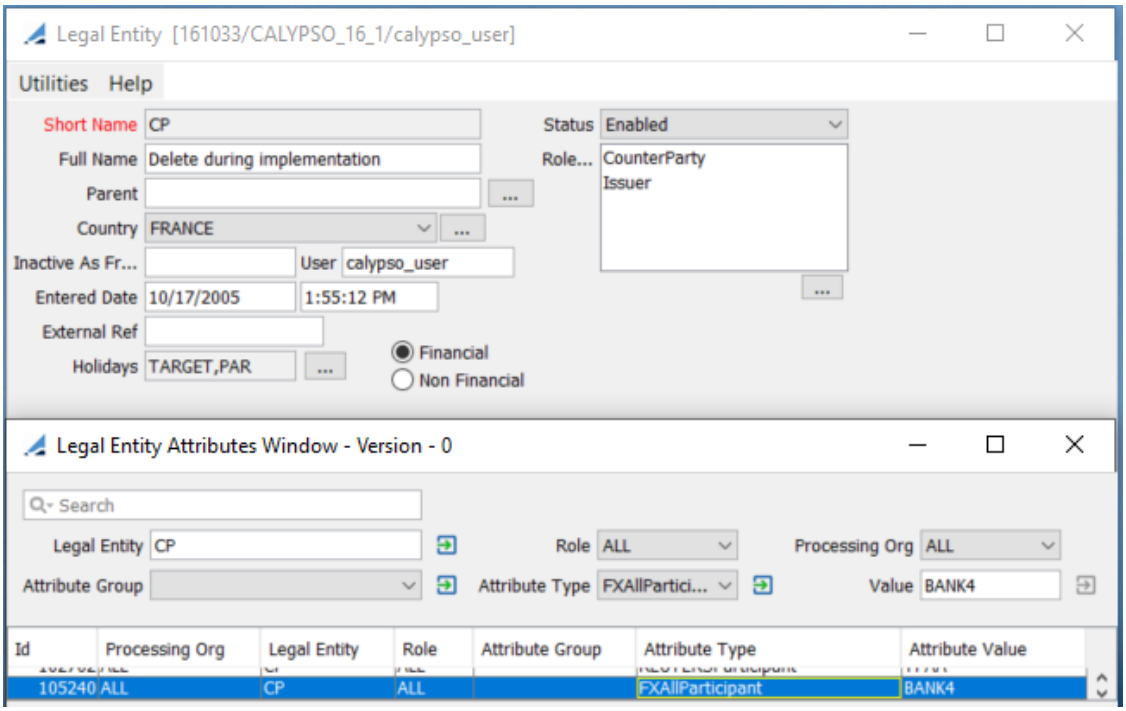
The Legal Entity for the Counterparty identifiers populated in the FIX message need to be configured in Calypso using the Legal Entity Attribute '**FXallParticipant**', and the value would match the bank provider value provided in FIX message.

The logic will look for the attribute value FXallParticipant configured on the Legal Entity involved in trade booking (CP-LE) which will be used to generate the counterparty for legal entity.

This lookup logic will be applied to Counterparty. If no Calypso Legal Entity is found using the rules above, an error will be raised.

Counterparty Mapping

For the Calypso legal entity which is intended to be used as counterparty in trade booking must be mapped to attribute **FXallParticipant** with the value configured in fxall platform. See the below illustration.



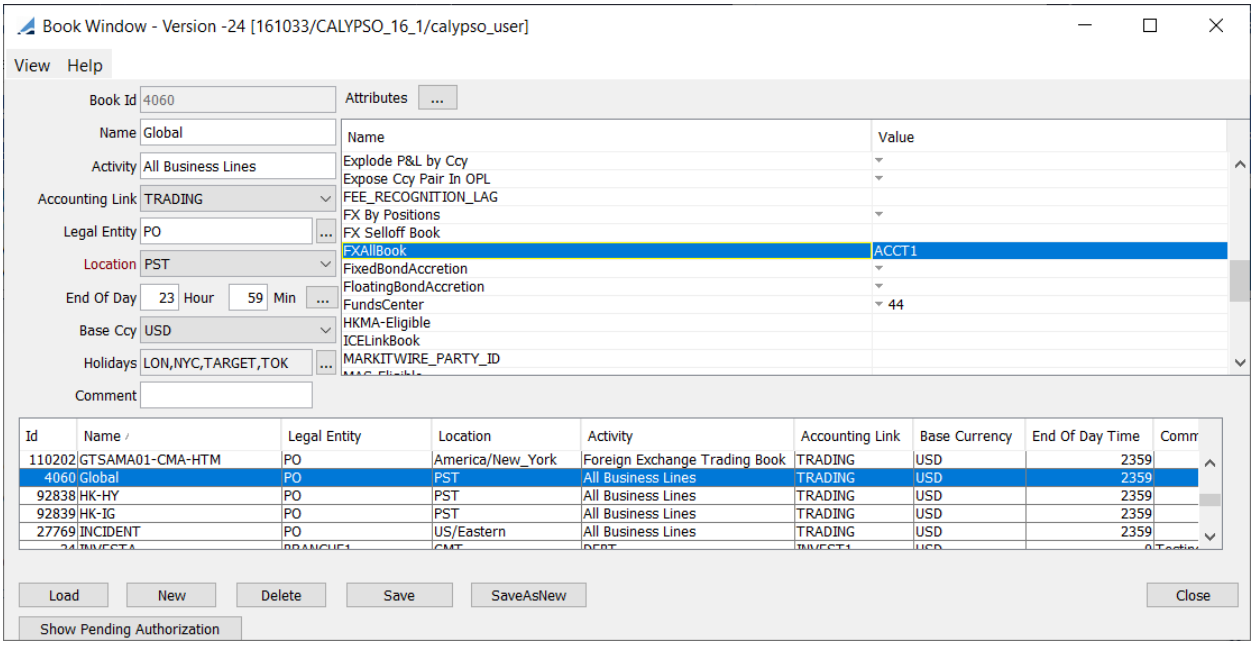
3.2 Book Mapping

When a trade is booked in the FXall Trading Terminal, the user is required to set an Account value.

The incoming FXall FIX message contains Account value for the account involved in the trade.

The account value populated in the FIX message needs to be configured in Calypso using the Book attribute '**FXallBook**', and the value would match the account value provided in FIX message.

FXall account value coming in the FIX message is mapped to book attribute **FXallBook** in Calypso as shown below:



If the incoming book cannot be matched to any book using the book attribute FXAllBook, the following mapping is used:

Name = FXAll/FXAllParticipant

Interface Value = <incoming book>

Calypso Value = <processing org>

In this case, the incoming book is mapped to the book defined in the PO legal entity attribute FXAllBook.

When no book is found using book attribute or PO attribute FXallBook, the FXall account value coming in FIX message is searched in Calypso as **book name**.

If no Calypso Book is found using the rules above, an error will be raised.

If you want to map FXall books to multiple Calypso books, you can use the following mapping:

Name = FXAll/FXAllBook

Interface Value = <incoming book>

Calypso Value = <Calypso book>

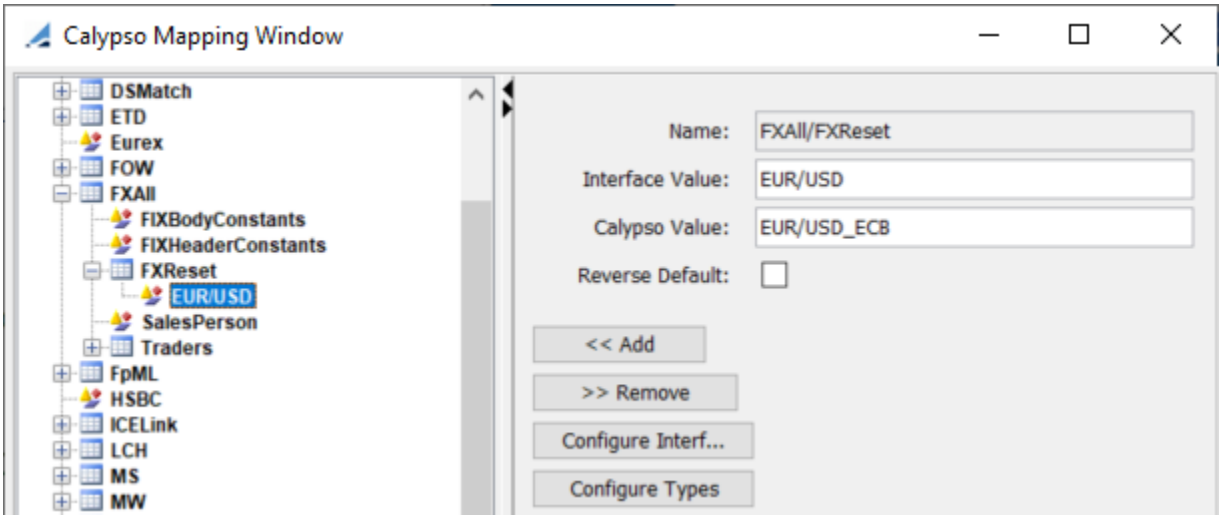
In this case, the incoming book is directly mapped to a Calypso book.

3.3 Product Mapping

When importing product-based trade types into Calypso, mappings must be setup so that the FXall interface can match the incoming product details to objects contained within the client's instance of Calypso. The sections below outline the various mappings required for the different trade types supported by the FXall interface.

FX-Reset

For saving the non-deliverable FX trades in Calypso, FX-Reset is required. Users can map the given reset with the Calypso FX-Reset using CalypsoMapping window.



FX Spot / FX Forward

If Max Spot Days (3 for example) > Spot Days in Currency Defaults, then T+1, T+2 and T+3 will be considered as Spot dates. However user can change the trade type to FXForward under some conditions:

If Min Spot Days = Max Spot Days, then SettleDate = SpotDate, trade is a Spot trade and cannot be changed.

If Min Spot Days < Max Spot Days, trade can be saved as Spot trade or FXForward trade based on MapSpotTradeOnSpotDateAs mapping.

Name = FXAll/TradeType

Interface Value = MapSpotTradeOnSpotDateAs

Calypso Value = Spot or Forward

When set to Spot, Spot Rate = Final Rate and SPOT_MARGIN fee is not created.

If not set, it defaults to Forward.

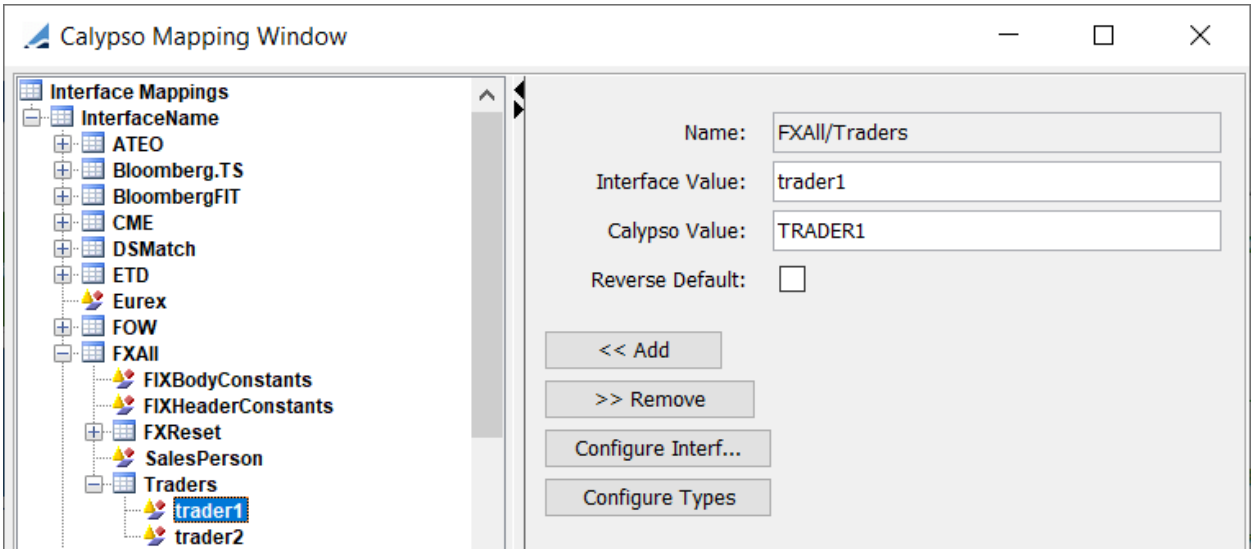
If primary currency and secondary currency default spot days are not equal, checking currency pair attribute SpotDateCalcByCcySpotDays.

If true, get the maximum value of the currency spot days. If false, get the minimum value.

3.4 Calypso Mapping

Trader Mapping

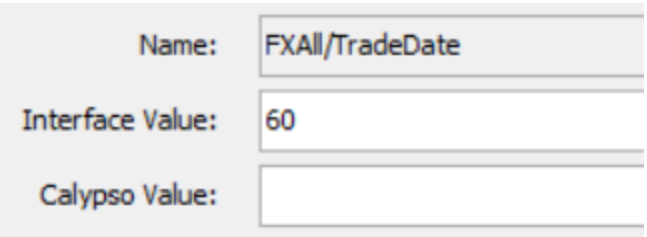
The Processing-Org trader present in fix message can be mapped to the TRADER available in Calypso. In Calypso Window mapping, the incoming trader name can be mapped to the existing TRADER in Calypso as shown below.



But if no mapping is found, then by default Trader is set with what is present in the incoming message.

Trade Date Mapping

By default, the trade is created using Trade Date = Tag 75 which is expressed in GMT timezone. If you want to use Tag 60 instead (entered user date and time in user timezone), add the following mapping:



Name = FXAll/TradeDate

Interface Value = 60

3.5 Fix-Engine Configuration

The FXall FIX Engine is responsible for getting messages from the FXall platform and handing it off to the appropriate workflows.

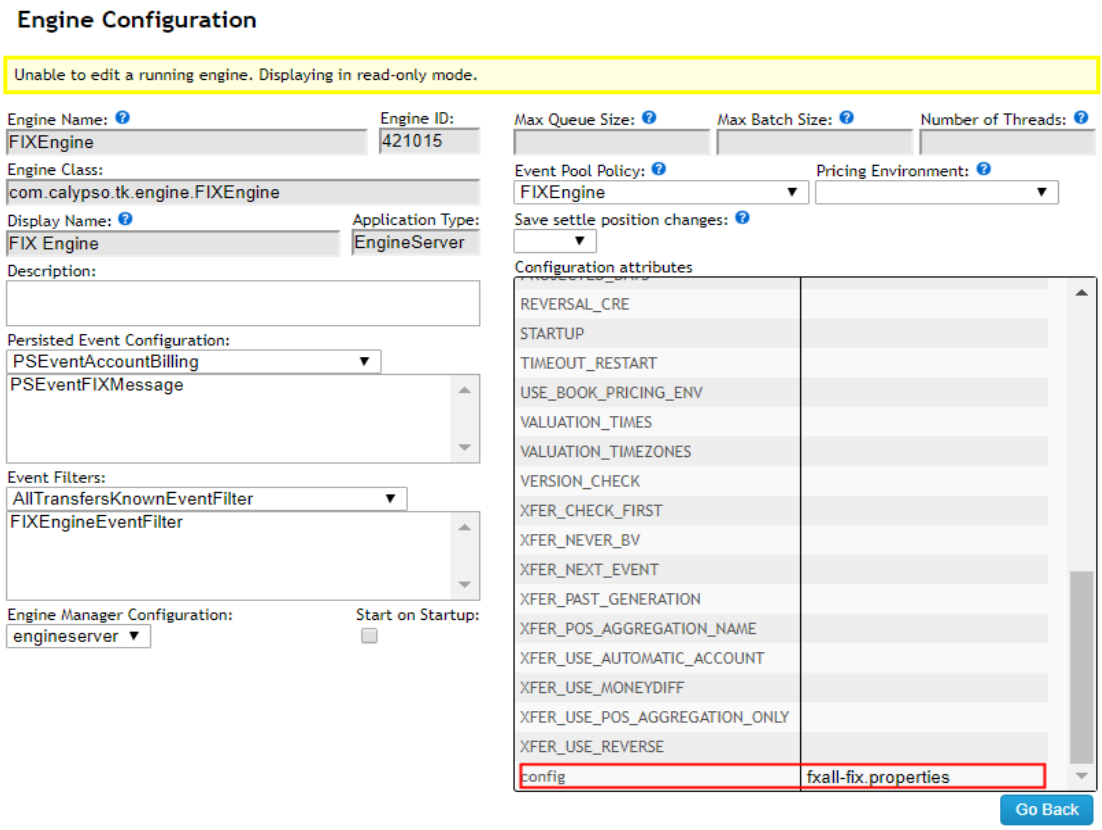
The FXall FIX Engine is built on the Calypso FIX Engine framework, and therefore while setting it up you will find generic FIX setup vs. FXall specific setup. For clarity, all required steps are listed below.

Please review the standard Calypso documentation for Engine setup to read about several useful engine parameters (such as thread count) and how to set them.

3.5.1 Configure the Engine

All the database-based Engine configuration is completed as part of applying the schema, including the engine name, event subscription, event filter, event policy, as well as assigning a unique id to the Engine.

You may refer to the schema file for more details.



To handle RFQs, the Fix engine also needs to subscribe to PSEventRFQRequestMessage events. It then publishes PSEventRFQResponseMessage events for the Calypso ePortals platform to consume.

Engine parameters:

config = fxall-fix.properties

OPTIONAL_FEATURE = fxall

3.5.2 Setup the FIX Config File

To run the FXall FIX Engine out-of-the-box you will need a properties file with the name “fxall-fix.properties” with the appropriate FIX connection settings.

A sample file is included under <calypso home>/client/resources with the name “fxall-fix.properties.sample”.

You will need to rename the file to “fxall-fix.properties”.

Once the file is customized, copy the file to “`<calypso home>/ tools/calypso-templates/resources`”.

Note that, as previously mentioned, the FXall FIX Engine uses the QuickFIXJ library for the FIX connectivity implementation. The QuickFIXJ library has many options that can be configured on a FIX session, using a standard properties file. The Calypso FXall FIX Engine uses this same file for internal settings as well.

For simplicity, we have provided a sample fxall-fix.properties file and will only refer to the minimum settings that must be changed to work with FXall connectivity. You can view all the available settings on the QuickFIXJ

Configuration page located at their documentation site at:
http://www.quickfixj.org/quickfixj/usermanual/1.5.3/usage/configuration.html

Sample Properties File

The sample “fxall-fix.properties” file appears similar to the following example:

```
# Default settings for sessions.
# These are inherited by each session defined below
# unless they are overridden in the session settings.
[DEFAULT]
ConnectionType=initiator
ReconnectInterval=10
HeartBtInt=20
LogonTimeout=20
LogoutTimeout=20
Calypso.LogOnInterval=5000
Calypso.LogOnRetryCount=5

# SSL Support
SocketUseSSL=Y
SocketKeyStore=fxall_dropcopy.jks
SocketKeyStorePassword= password123
EnabledProtocols=TLSv1.2

Calypso.UploadMode=Local
Calypso.PersistMessages=All

# DUMMY_FXALL session definition (FXall client)
[SESSION]
Calypso.FIXMessageType= FXALL
BeginString=FIX.4.4
DefaultApplVerID=7
SenderCompID= calypso
TargetCompID= FXALL
DataDictionary=DD_FXAll.xml
AppDataDictionary=DD_FXAll.xml
```

```
SocketConnectHost= 216.109.74.40
SocketConnectPort=443
FileLogHeartbeats=Y
FileIncludeMilliseconds=Y
FileIncludeTimeStampForMessages=Y
ValidateIncomingMessage=N
RequiresOrigSendingTime=N
ResetOnLogon=Y
ResetOnLogout=Y
ResetOnDisconnect=Y
StartTime=01:00:00
EndTime=23:00:00
TimeZone=America/New_York
```

QuickFIXJ Settings

To connect with FXall platform successfully, you will need to change the SenderCompID, TargetCompID, SocketConnectHost and SocketConnectPort connection properties to the correct values for your setup. Please contact FXall support for these details.

Additional points to note regarding the core QuickFIXJ settings:

- The QuickFIXJ settings allow you to configure multiple sessions in a single properties file. This means if you have multiple FXall session logins, you can use a single FXall FIX Engine to connect to all of them.
- FileStorePath and FileLogPath are defaulted to USER_HOME/Calypso/FIXEngine/Store and $USER_HOME/Calypso/FIXEngine/Log respectively. These may be overridden at the DEFAULT or SESSION level within the config file. There is no support for other Store or Log mechanisms currently.

3.5.3 Launching the FXall FIX Engine

Adding Logging Categories

To see logging messages for the Data Uploader and FXall modules you need to set the following log categories:

- UPLOADER: Set this to see logging for the Data Uploader translation from the internal Calypso xml format to the actual trade object.
- FXALL: Set this to see logging for the FXall translation from the external format to the internal Calypso xml format.
- FIX: Set this to see logging for the shared FIX connectivity & message processing pieces of the FXall FIX Engine.

Running the FXall FIX Engine

With the previous steps completed, you are now ready to run the FXall FIX Engine.

To start/stop the engine use the Calypso Engine Server Admin Web Console.

With the FXall Engine operating, you can then allege trades through the FXall Trading Terminal. The FXall Engine will process the trade messages and create corresponding trades in Calypso.

The Task Station will display any errors that may occur.

Daily Stop/Restart

The FXall FIX server is shut down daily after business hours and startup again at the start of business the next day. As part of this daily cycle, the Sequence Numbers for the FIX connections are reset as well.

The Calypso FXall FIX Engine handles this for you automatically, based on the values set in the fxall-fix.properties settings file for the properties StartTime, EndTime, and TimeZone. These properties control when the engine determines that a new session should be started & the Sequence Numbers reset.

For more details on these settings, please refer to the QuickFIXJ documentation site.

Order Workflow Setup

4.1 Order Processor Field in Order Pricing Sheet

An order destined to FXall must have Order Processor = FXAll.

4.2 Workflow Description

This section describes the FXall Interface order workflow. It is important to understand these details so that the Calypso Order workflow can be customized accordingly. Please read and follow all setup instructions carefully to ensure a successful installation.

Bi-directional functionality allows a user to perform actions in Calypso and have the appropriate FIX message sent to FXall using the FIX Message API.

The bi-directional functionality is implemented using the PlatformAllege and PlatformCancel workflow rules. Simply add the appropriate workflow rules to an action of the Calypso Order workflow and applying that action to an order will prompt the FIXEngine to create the external message and send it to the FXall platform. A BOMessage of type PLATFORMMSG is created and moved to SENT status if successfully sent, otherwise it remains in PENDING status.

For example, add the workflow rule PlatformAllege on the SEND order action. If any error occurs while sending the message to FXall, whether due to missing details or invalid action applied, the PLATFORMMSG message will have errors connected to it which can be reviewed to determine the error, address the specific issue, and resend the workflow message.

Once the message has been acknowledged by FXall, the following happens:

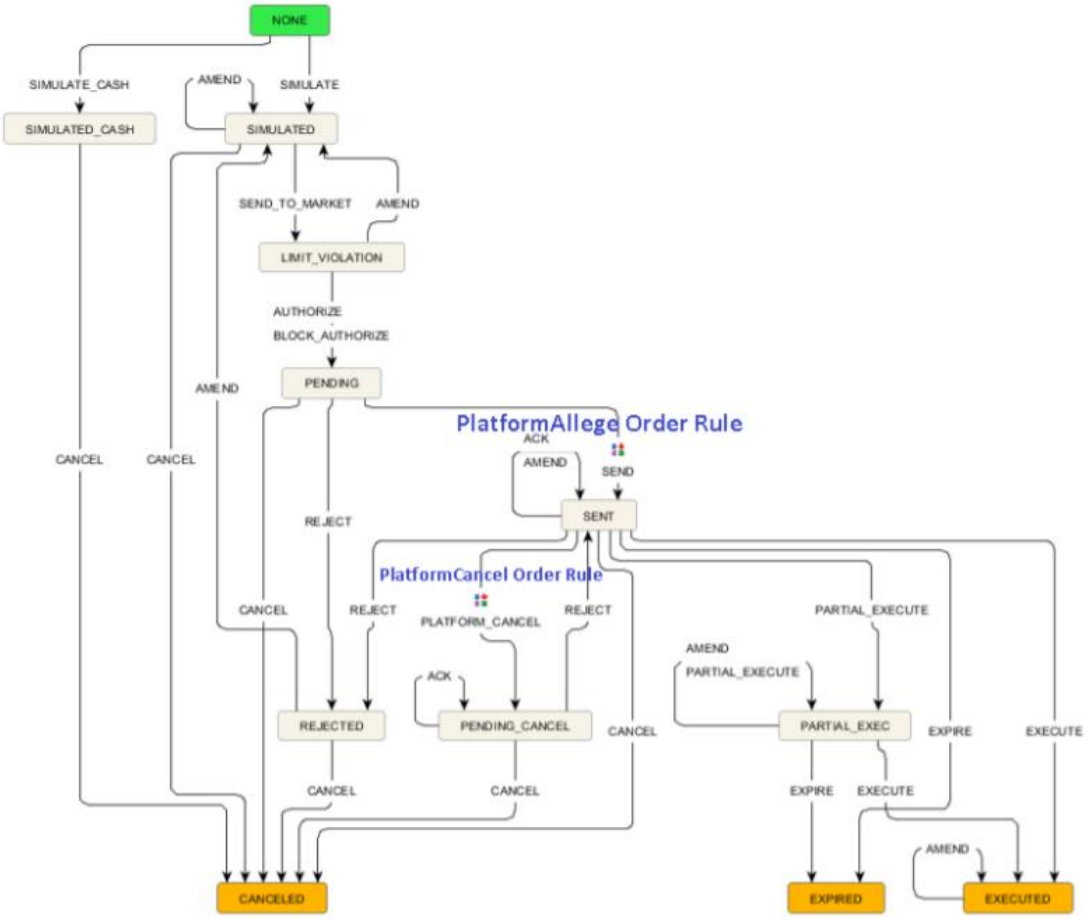
- The corresponding PLATFORMMSG message moves to COMPLETED.
- The order comment PlatformSubmitStatus is updated with '<Action>Successful'. Example SENDSuccessful if the order is alleged to the platform.
- FXall sends the appropriate FIX message to Calypso, which is used to update the order comment with FXall Order Id.

4.3 Workflow Setup

The following actions need to be configured on the Order workflow:

- Allege Order to FXall (PlatformAllege rule) – Example, action SEND below.
- Cancel sent order (PlatformCancel rule) – Example, action PLATFORM_CANCEL below.

Sample workflow in resources/FXAllAMOrder.wf.



4.4 Message Flow

The table below describes the sequence of FXall FIX messages that flow in and how they are handled.

FXall Business	Calypso Action
Send Order to FXall	Calypso sends message: FIX Message of MsgType 'D' is sent to FXall. You can prevent sending all allocations from Calypso to FXALL in case of block order, using the following mapping: Name = FXAll/Translator Interface Value = AggregateAllocations Calypso Value = true When set to true, allocations are aggregated per account. If not set, default is false.
Confirmation message	Calypso receives message: FIX message with MsgType as '8' and with Order-Id tag (tag 11) Action: Update the Calypso order with FXall status. When tag 35=8 and tag 39=4, the message type is CANCELLED. The action to be applied in case of CANCELLED message can be set in the following mapping: Name = FXAll/OrderAction Interface Value = CANCELLED Calypso Value = CANCEL If the mapping is not defined, action ACK is applied by default.
Execute the order in FXall	Calypso receives message: FIX message with MsgType as '8' along with Product Details Action: Move the Calypso order to EXECUTED status and create a trade.

RFQ – Additional Setup Requirements

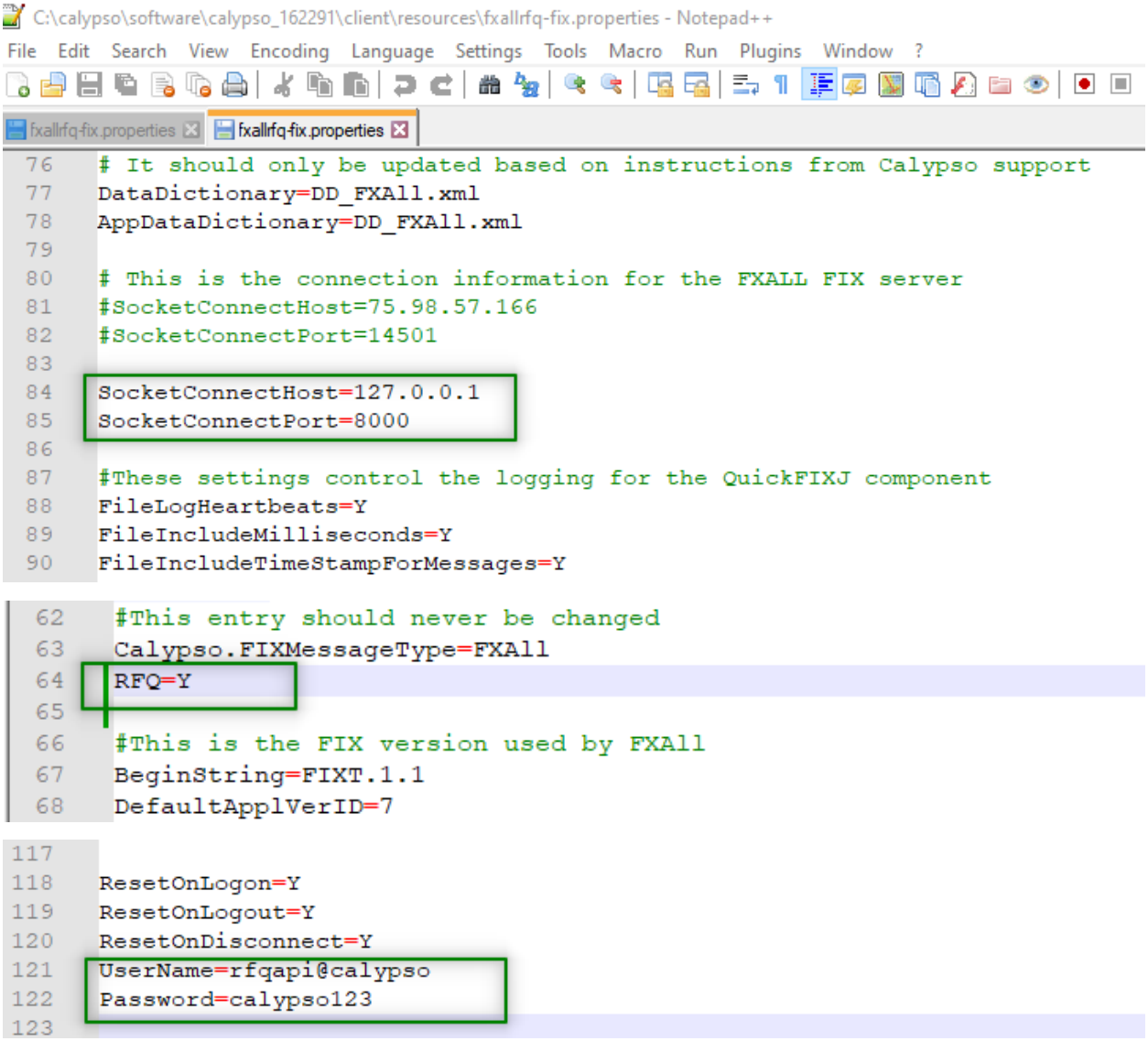
5.1 Properties File "fxallrfq-fix.properties"

A sample file is included under `<calypso home>/client/resources/ fxallrfq-fix.properties.sample`.

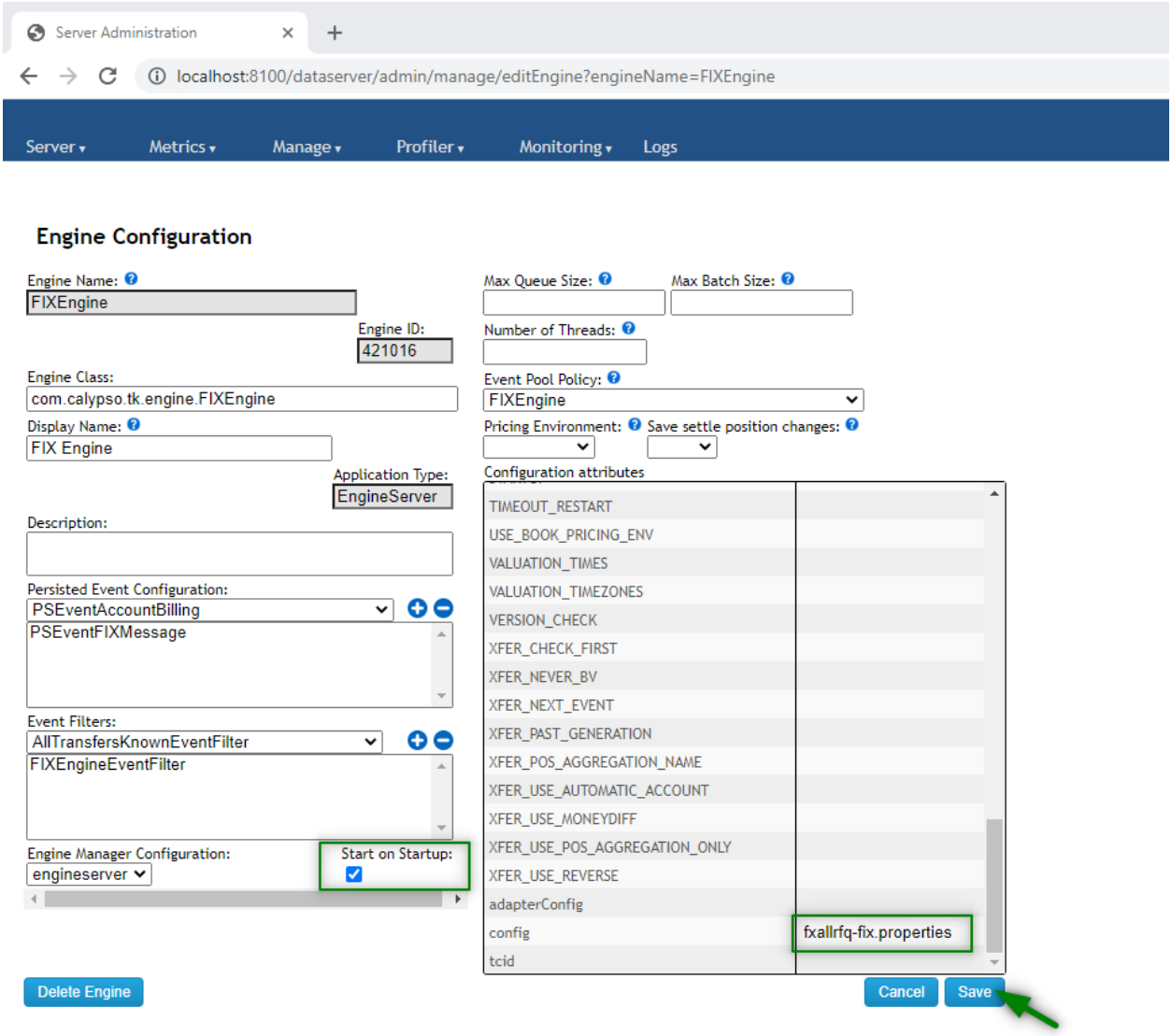
You need to rename the file to "`fxallrfq-fix.properties`".

Once the file is customized, copy the file to "`<calypso home>/ tools/calypso-templates/resources`".

Update the following properties with your own values as needed:



5.2 FIX Engine Configuration



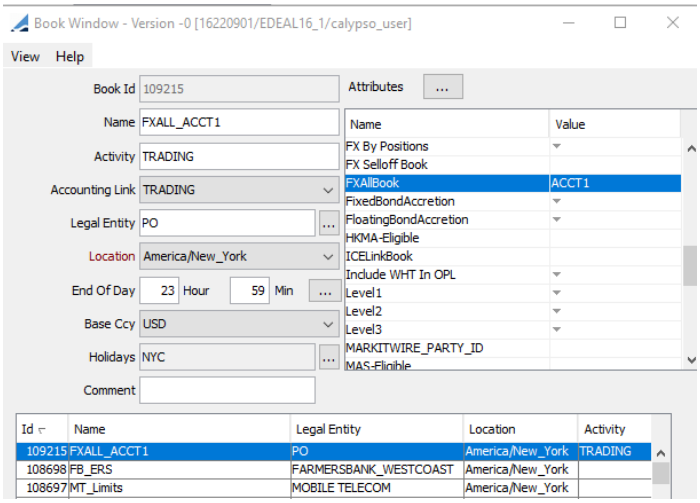
Start the FIX engine.

5.3 Trading Book

Create a new trading book which will be used for booking the trades using the external pricing mechanism (FXALL).

Set the book attribute FXAllBook as needed.

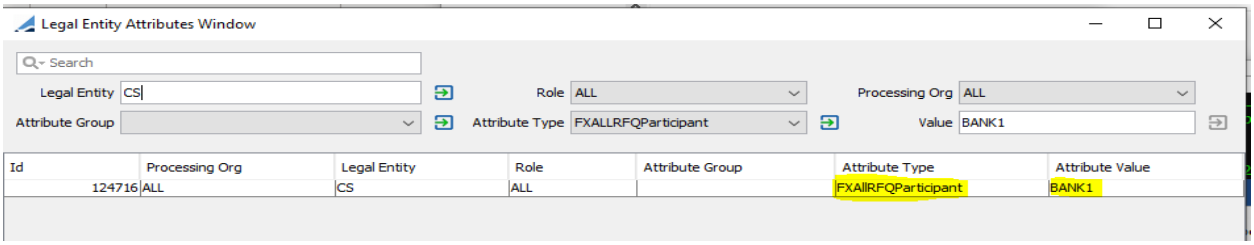
Example:



5.4 Legal Entities

Set the legal entity attribute FXAllRFQParticipant for the legal enities that represent liquidity providers on the FXall platform.

Example:



5.5 Message Workflows

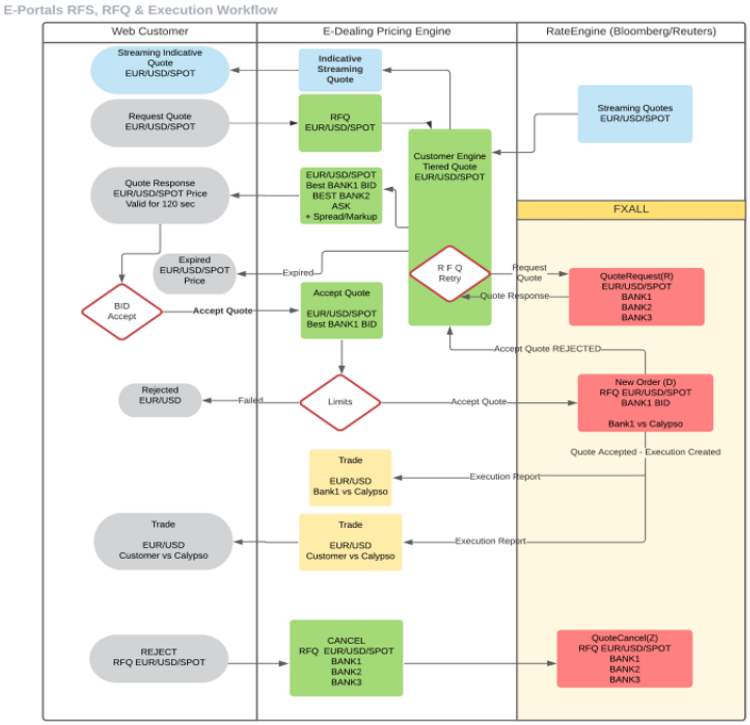
Import the following messages workflows:

- resources/RFQREQUESTMSG.wf
- resources/RFQRESPONSEMSG.wf

For Outgoing RFQ and Order a RFQREQUESTMSG message is created.

For Incoming Quotes and Execution Report, a RFQRESPONSEMSG message is created.

Below is the business workflow for RFQ and RFS execution.



On Successful verification of the quote, two back-to-back trades are executed, one with Liquidity Provider vs Calypso and other between ePortals user vs Calypso.

Calypso ePortals	Calypso-FXAll	FXAll Platform
1 - PSEventRFQRequestMessage event is created and published with payload of type 'QuoteRequest'	2 - FixEngine receives the PSEventRFQRequestMessage with payload of type 'QuoteRequest'. Create Fix message 35=R and send to FXall platform.	3 - Stream Quotes 35=S
	4 - Stream Quotes 35=S received by fix engine. 5 - FixEngine applies translation on the FIX Message and create a Quote JSON data. FixEngine publishes event PSEventRFQResponseMessage with the JSON data as payload.	

Calypso ePortals	Calypso-FXAll	FXAll Platform
6 - ePortals consumes the event PSEventRFQResponseMessage with payload data of type 'Quote' and fetches the quote information from the payload. 7 - Customer accepts the Quote, ePortals creates & publishes event “PSEventRFQRequestMessage” with payload of type as ‘NewOrderSingle’ / 'NewOrderMultiLeg'.		
	8 - FixEngine gets the event and converts the Order information present in the payload to FIX message 35=D. 9 - FixEngine sends the fix message 35=D to FXall. 10 - Receives ACK for order. 35=8 & 39=0 (Order Status = New)	11- Order executed in FXall.
	12 - FixEngine receives Execution Report 35=8 & 39=2 (Order Status = Filled). 13 - Trade is created in Calypso. Also translation is applied on the FIX message and JSON data of type execution report is created and published as PSEventRFQResponseMessage	

Once FXall has been installed and configured, you can configure an External Pricing Rule and a Pricing Rule in Calypso ePortals. Then you can send requests for quotes.

Please refer to the Calypso eDealing Web Portal for details.

Test Tool Setup: FileWatcher

Note: The details in this section are provided for testing purposes only, and not recommended for production use.

As mentioned in the previous section, the FXall FIX Engine also supports processing FIX messages from files. To achieve this, you must run the DataUploader FileWatcher in 'fxall' mode, so that it will load files from a specified location and pass them on to the FXall FIX Engine.

The steps below assume you've already setup the DataUploader module as per the Data Uploader Setup Guide.

6.1 Setup the FileWatcher Config File

To run the FileWatcher in 'fxall' mode, you will need a properties file with the appropriate settings. A sample file "fxalluploader.properties" is included under <calypso home>/client/resources with the name ".sample" as the suffix.

Please change the fileDir as required for the polling directory. Also note that the fixSettings property must point to the FXall FIX Engine's property file.

For other details, please refer to the Data Uploader Setup Guide.

Once the file is customized, you will need to install it to the engineserver war using the standard Calypso installation procedures for installing resource files.

6.2 Launching FileWatcher

6.2.1 Adding Logging Categories

To see logging messages for the Data Uploader and FXall modules you need to set the following log categories:

- UPLOADER: Set this to see logging for the Data Uploader FileWatcher component.
- FXall: Set this to see logging for the FXall FileWatcher component.
- FIX: Set this to see logging for the shared FIX FileWatcher component.

6.2.2 Running FileWatcher

With the previous steps completed, you are now ready to run the FileWatcher.

To start/stop the engine use the Calypso Engine Server Admin Web Console.

With the FileWatcher operating, you can then place a '.fix' file to import in the watched directory (specified by fileDir in the properties file). At the end of the current wait interval, the FileWatcher will notify the Data Uploader which will then load the file and hand it off to the FXall FIX Engine.

The Task Station will display any errors that may occur.

Please also review requirements for FileWatcher from the Data Uploader Setup Guide.

Test Tool Setup: GUI

Note: The details in this section are provided for testing purposes only, and not recommended for production use.

The Calypso FXall Interface is built on the Data Uploader framework, and therefore supports uploading FXall FIX files through the Data Uploader GUI. This can be useful for testing and does not require you to run the FXall FIX Engine.

The steps below assume you've already setup the DataUploader module as per the Data Uploader Setup Guide, including adding the GUI window to your menu.

7.1 Setup the GUI config file

Note: The need for this step will be removed in a future release.

To upload FXall FIX files through the GUI, you will need a properties file with the appropriate settings. A sample file "fxall-datauploader-gui.properties" is included under <calypso home>/client/resources with the name ".sample" as the suffix.

Note that the fixSettings property must point to the FXall FIX Engine's property file, although the engine itself does not need to be running.

You must also ensure that the 3rd party jars have been installed on the client side.

7.2 Uploading via the GUI

With the previous steps completed, you are now ready to upload FXall FIX files using the GUI. Simply launch the Data Uploader GUI from the menu, choose the Source/Format, browse to select your '. fix' file, and upload.

For further details on using the Data Uploader GUI, please refer to the Data Uploader Setup Guide.

Note: The uploaded file must have a '. fix' extension, not '.xml'

Troubleshooting

This section contains details on how to troubleshoot any issues you may encounter with the FXall Engine or processing messages, and what to do if you need to report an issue to Calypso.

8.1 Connectivity

The FXall FIX Engine automatically attempts to reconnect if a connection with the FXall platform is lost. On reconnect, it will first process any queued messages, and then be available to process new messages.

In case messages are not being received by the engine, please check the log files produced by the QuickFIXJ library. As mentioned previously, the default location for these logs is $USER_HOME/Calypso/FIXEngine/Log. You can also check the Calypso FXall FIX Engine logs.

8.2 Message Processing

For message processing failures, check the Task Station of UPLOADSOURCEMSG or GATEWAYMSG workflow errors. Please review the installation section of this document for details.

8.3 Debug Logging

Additional logging can be configured to help with debugging errors; just set the following log categories:

- FIX_DEBUG_XML: Set this category to create xml files of the incoming messages from the FXall platform as well as the interim Calypso xml format created by the translation under $USER_HOME/Calypso/FXall. Note that these files can be used for loading via the File Watcher.
- FIX_DEBUG_API: Set this category to see additional logging statements for the FIX connectivity, including Login/Logout and Admin messages (such as resend requests). These logging statements are helpful to debug any FIX connectivity issues, as well as see headers of the messages coming in from the FXall platform before they are interpreted by the FXall FIX Engine.

8.4 Reporting Issues to Calypso

If, after attempting all the above steps you still need to contact Calypso, please ensure that your ticket contains the following information:

- All available logs, including:
  - QuickFIXJ connectivity logs
  - FXall FIX Engine logs
  - FXall FileWatcher logs
  - DataServer and EventServer logs
- FIX messages, if applicable

- Clear description of the issue, including:
  - What steps were executed to produce the issue, both in the FXall Trading Terminal and Calypso
  - What attempts, if any, were made to debug the issue, and what were the results

Frequently Asked Questions

1. Unable to connect to FXall platform

It can be a network issue. Try Telnet on the IP and port provided by FXall platform. If Telnet fails kindly check firewall and contact Refinitiv support.

2. FIX Session frequently disconnect

It is mostly due to poor network connectivity. Kindly contact FXall support and ask them to confirm session heartbeats.

3. How to enable logs for FXall

Add debug log categories 'FXall, FIX' on EngineServer/Navigator.

4. Unable to connect to FIX session during certain time

OR FIX connectivity stops after a certain period.

FXall provides StartTime and EndTime, only during which the user can connect to FIX session. The user will not be able to connect to fix session before StartTime and after EndTime.

This connectivity time frame can be controlled via fix properties.

Example:

StartTime=07:00:00

EndTime=23:00:00

TimeZone=America/New_York

For StartTime and EndTime value, please contact FXall support.

5. FIX message gets rejected due to missing or unorder tag

OR Data-Dictionary outdated / out of sync.

FXall module tries to keep its data-dictionary in sync with FXall platform. But in case, the data-dictionary is outdated, and fix messages started getting rejected from FIX session, then the user is advised to disable the data-dictionary validation by adding property 'ValidateIncomingMessage=N' in fix config file. Disabling the validation check will stop the fix messages from getting rejected. FXall module has its own data validation check and if the required data is missing or invalid on fix message, FXall module will log an error in task-station.