Nasdaq Calypso

Tradeweb Integration Guide

Version 4.14.0

Revision 34
April 2025
Approved

Document History

Revision	Published	Summary of Changes
1.0	**Feb 2014**	Initial Version for 1.0.0 Release
2.0	**July 2014**	Updates due to packaging changes
3.0	**July 2014**	Updates for 3rd party jar installation procedures and uploading files via the GUI
4.0	**Sept 2014**	Updates for mina jar packaging changes
5.0	**Nov 2014**	Update details regarding daily start/stop of engine
6.0	**July 2015**	Added Performance and Stability Improvement details and Appendix section
7.0	**Dec 2015**	Update for data uploader version, supported product, workflows and keywords
8.0	**March 2016**	Update for supported calypso version and added allocation requirement
9.0	**July 2016**	Update for supported Calypso version
10.0	**May 2017**	Added SSL Connection and TSL Connection
11.0	**Sept 2017**	Added Proxy Server
12.0	**March 2018**	Added STunnel
13.0	**September 2020**	Updated for version 2.5.0
14.0	**March 2021**	Updated for version 2.6.0
15.0	**April 2021**	Updated for version 2.7.0
16.0	**May 2021**	Updated for version 2.7.2, 2.8.0
17.0	**August 2021**	Updated for version 2.10.0

Revision	Published	Summary of Changes
18.0	**October 2021**	Updated for version 2.12.0
19.0	**December 2021**	Updated sample property file
20.0	**February 2022**	Updated for version 3.0.0, 3.0.1, 3.0.2, 3.0.3 - Technical release only – Version 17.0 compatibility Updated for version 2.13.0, 2.13.1 – Added book mapping for single executed order, Trade Date mapping
21.0	**March 2022**	Updated for version 3.2.0 – Added OrderRateIndex mapping
22.0	**April 2022**	Updated for version 3.3.0 – Added Tag 1 book mapping for single trades
23.0	**June 2022**	Updated for version 3.5.0 – Added OrderAction mapping
24.0	**July 2022**	Updated for version 3.6.0 – Added platform PO mapping
25.0	**February 2023**	Updated for version 3.7.0 – Added trader mapping
26.0	**May 2023**	Updated for version 3.9.0 – Added support for block orders for Bonds
27.0	**August 2023**	Updated for version 3.11.0 - Block orders are no longer using the AMBlockOrder workflow. They are using the AMOrder workflow instead
28.0	**December 2023**	Updated for version 3.11.1 – Added nominal mapping.
29.0	**January 2024**	Updated for version 4.3.0 (compatibility with version 18)
30.0	**March 2024**	Updated for version 4.5.0
31.0	**May 2024**	Updated for version 4.6.0 – Added support for CANCEL action for all products and Counterparty Mapping using Tag 447=N
32.0	**July 2024**	Updated for version 4.7.1 – Added bi-directional interface for Fixed Rate Bonds - EFP - Future Bond.
33.0	**January 2025**	Updated for version 4.12.0 – Added engine parameter OPTIONAL_FEATURE.
34.0	**April 2025**	Updated for version 4.14 – Added import of Repo trades from Tradeweb.

This document describes how to integrate Calypso with Tradeweb.

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

1 Introduction

This document describes the Calypso Tradeweb Interface setup. The Tradeweb interface allows an End User Firm (Buy-Side) to import trades which have been booked through the Tradeweb Viewer using Tradeweb's SEF workflow.

The Calypso Tradeweb integration connects to Tradeweb using a FIX interface. Once the connectivity is setup, a trader can book a trade through the Tradeweb Viewer, and Tradeweb will send that as a FIX message to the Calypso Tradeweb interface. The message will then flow through the configured Calypso workflows which route the message through the appropriate stages to create a Calypso trade. Additional clearing lifecycle messages will also be sent over the FIX connection, and the appropriate lifecycle actions will be applied to the Calypso trade.

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

Supported Features

Calypso's Tradeweb integration supports trade messages for the following Clearing workflows:

- Non-Allocating
- Pre-Allocation
- Bunched Order / Post-Allocation

The Tradeweb module supports the following Tradeweb trade types:

- IR (Vanilla, IMM, Forward, OIS, Basis). These are all captured in Calypso as Swap trades.
- Derivatives (CDS Index, Credit Default Swap).
- Fixed Income Trade (Bond).
- Bilateral interface for Fixed Rate Bonds - EFP - Future Bond
- Repo (NEW, CANCEL, RERATE, AMEND to closeout TERM trades by amending the end of the trade, CALL EXERCISE to closeout OPEN trades by amending the end of the trade)
- Swaptions (Option type - Payer, Receiver and Straddle / Direction - Buy and Sell / Settlement type - Physical and Cash)

  Exchange Delta - true. Two trades are created: Swaption and Swap

  Allocations in Bilateral mode

  No clearing.

The module supports the following Tradeweb messages for the Clearing workflow (i.e. these message types can be consumed by Calypso):

- Execution Report: Once the trade is executed within the Tradeweb Viewer and the Dealer accepts, Tradeweb will send a message over the FIX interface. A Bilateral trade will be created in Calypso to represent the trade between the two parties (Dealer vs. Buy-Side), with the Calypso user's party as the PO.
- Confirmation (Cleared): When the CCP confirms the trade has been cleared, Tradeweb will send a FIX confirmation message. Within Calypso, the original Bilateral trade will be Novated to the CCP as counterparty. The Bilateral trade will be terminated and a new Cleared trade created.
- Confirmation (Rejected): If the CCP rejects the trade for any reason, the Bilateral trade will be cancelled.
- Allocation Report: If the trade is allocated, Tradeweb will send a FIX confirmation message. Within Calypso, the original block trade will be allocated and its respective allocated fund trade will be created.
- Cancelation Report: When receiving message with tag 35=8 and tag 150=H, both the trade and its associated allocations are canceled if applicable.

Order Integration

The Tradeweb interface supports the following products for Order integration.

Product Types:

- Vanilla IRS
- Basis Swap
- Cross Currency Swap
- Bond
- Fixed Rate Bonds - EFP - Future Bond
- CDS Index
- CDS

Order Type:

- Single Order
- Block Order

Order Lifecycle:

- NEW/CANCEL

Make sure that environment property ASSET_MANAGEMENT=true

2 Setup Instructions

2.1 Software Requirements

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

2.2 Installation Instructions

2.2.1 Data Uploader Installation

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

- Applying the Gateway SchemaBase/SchemaData to your database
- Setting up the GATEWAYMSG & UPLOADESOURCEMSG workflows
- Setting up Task Station tabs for the workflows above

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

2.2.2 Calypso Components

Choose the Tradeweb component from within the Calypso Installer or use the patch tool to add it to an existing installation.

2.2.3 Setup Config Data using Execute SQL

The following files are loaded when you run Execute SQL:

- SchemaBase.xml
- GatewaySchemaBase.xml
- FIXSchemaData.xml
- FpMLSchemaData.xml
- TradewebSchemaData.xml

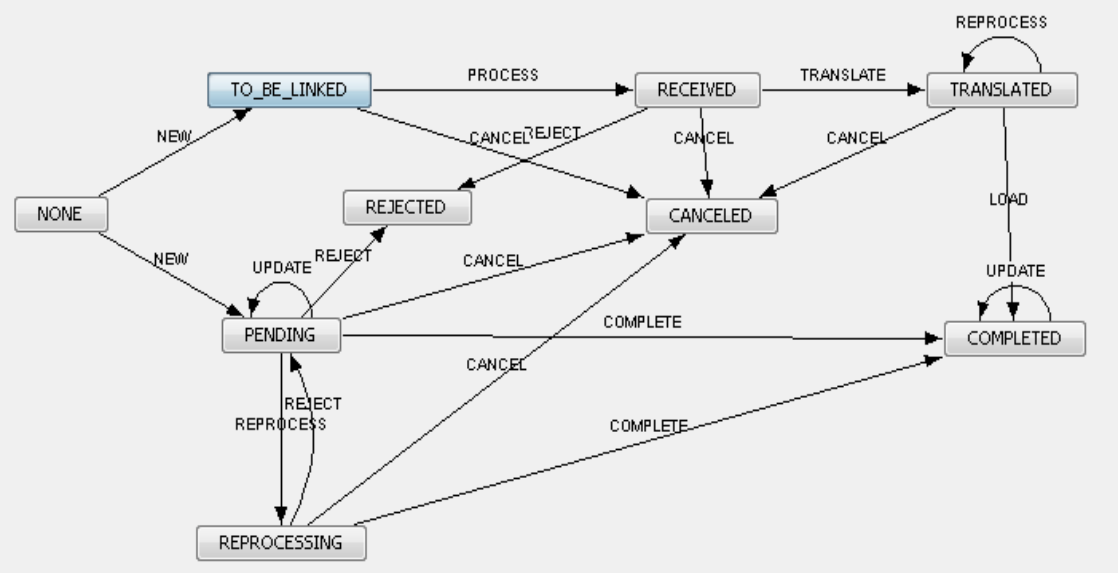
3 Configuration Requirements

3.1 Message Workflow Setup

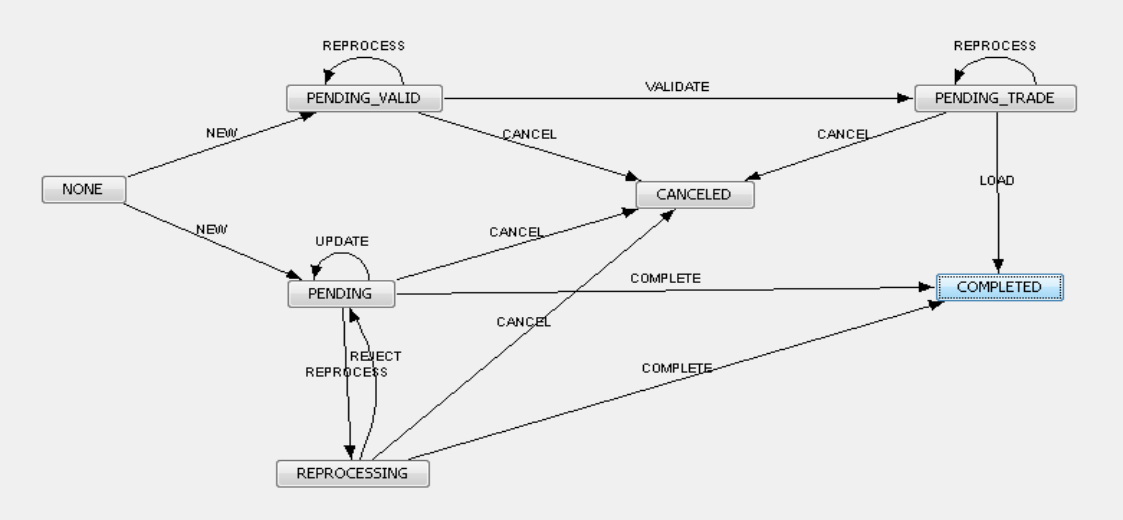
The Tradeweb module uses the UPLOADSOURCEMSG and GATEWAYMSG workflows when importing messages. These should have been setup as part of the Data Uploader Integration Guide.

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.

The UPLOADSOURCEMSG workflow:



The GATEWAYMSG workflow:



3.2 Task Station Setup

The Tradeweb 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 are failed in validation from the task station.

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

3.3 Legal Entity Mapping

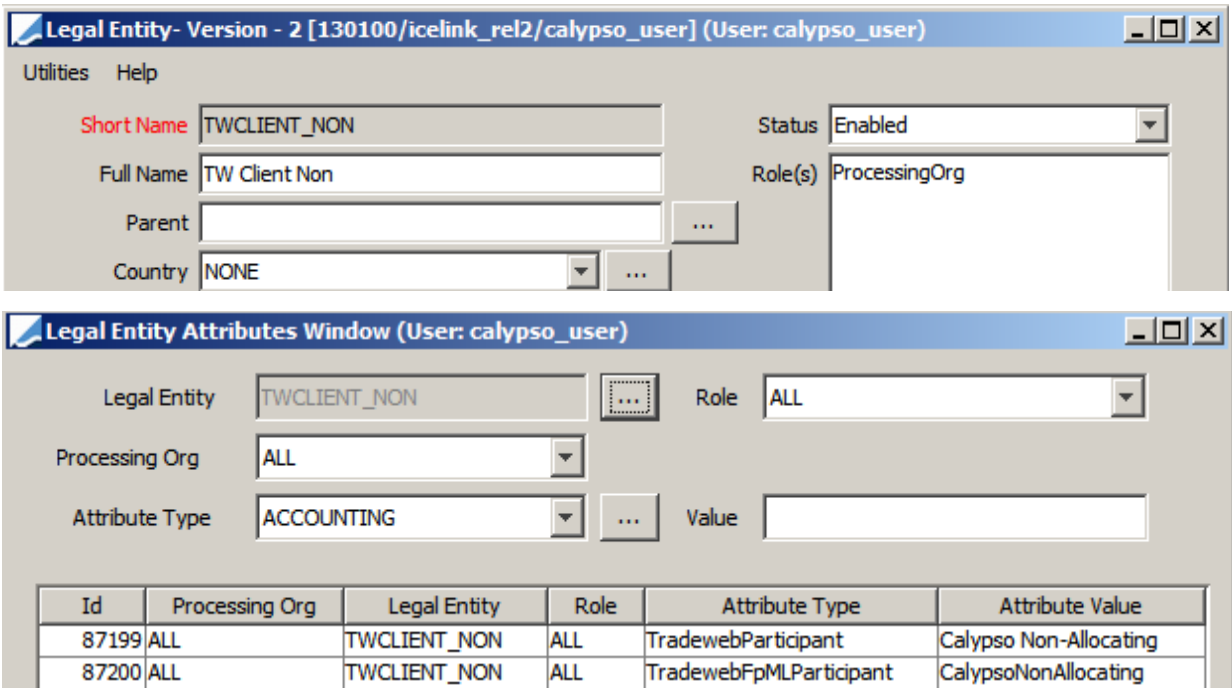
The incoming Tradeweb FIX messages contain Legal Entity identifiers for all parties involved in the trade (Party, Counterparty, FCM, and CCP). In addition, the FIX messages contain an embedded FpML message with product details, and this FpML message may contain **completely different** Legal Entity identifiers for the Party & Counterparty.

To address this, the Calypso Legal Entity setup for the Party & Counterparty requires **two** mappings to be setup in the Legal Entity Attribute screen.

The Legal Entity for the Party, Counterparty, FCM, and CCP identifiers within the FIX message will be identified in Calypso using the Legal Entity Attribute '**TradewebParticipant**', and the value would match the legal entity value provided in the FIX message.

The Legal Entity for the Party & Counterparty identifiers within the embedded FpML message will be identified in Calypso using the Legal Entity Attribute '**TradewebFpMLParticipant**', and the value would match the legal entity value provided in the embedded FpML message.

The fetching logic will first search for a Legal Entity with the attribute value matching the value specified in the message, and if not found then it will search for a Legal Entity having a matching Short Name (case-sensitive or all uppercase).



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

Tag 447 can contain "B=BIC" and "N=Legal Entity Identifier" for the CounterParty. Tag 447=N is prioritized if both are present, and the counterparty is taken from the corresponding Tag 448 and mapped to legal entity attribute TradewebParticipant.

[*Note* You must contact Tradeweb to provide you with the internal identifiers contained in the messages. Alternatively, you may use the Tradeweb FIX engine to consume test trades booked in the Tradeweb DEMO environment and use the details from the Task Station errors to determine the identifiers that must be setup as Calypso Legal Entity Attribute values]

[*Note* Even if the identifiers in the FIX and FpML messages match for the Party and Counterparty values, you must still setup both Legal Entity Attributes]

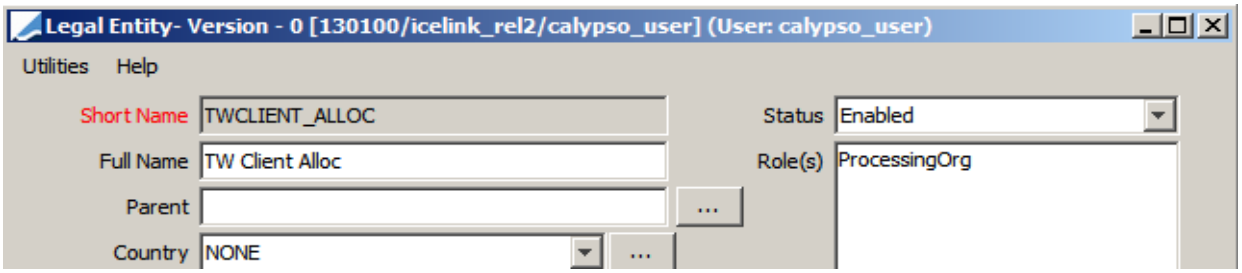
3.4 Parent Legal Entity Mapping

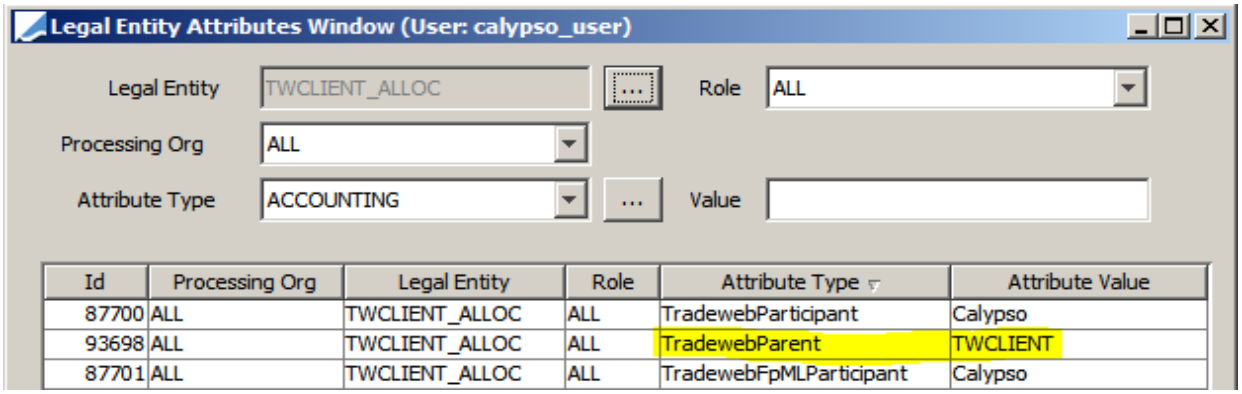
In some cases, there might be a need to map several separate Tradeweb identifiers to a single parent LE in Calypso.

To support this, first follow the procedure described above to setup separate Calypso Legal Entities each mapped to their corresponding Tradeweb identifier. Once that is complete, simply add the 'TradewebParent' attribute on each of the Calypso Legal Entities and populate it with the Calypso LE Short Name that you wish to be used as the parent.

When the lookup logic encounters a Legal Entity that has a parent defined, it will use the parent LE instead for all LE-related logic afterwards.

[*Note* This is only supported currently for the Party and Counterparty identifiers. In addition, when a parent is found, the child LE short name will be saved as a keyword on the trade; PlatformPO for the Party, PlatformCP for the Counterparty]





3.5 Book Mapping

Once the Tradeweb module determines the appropriate Legal Entity to use as the PO, it must then choose a Calypso Book for the trade. The section below outlines the logic used by the Tradeweb interface.

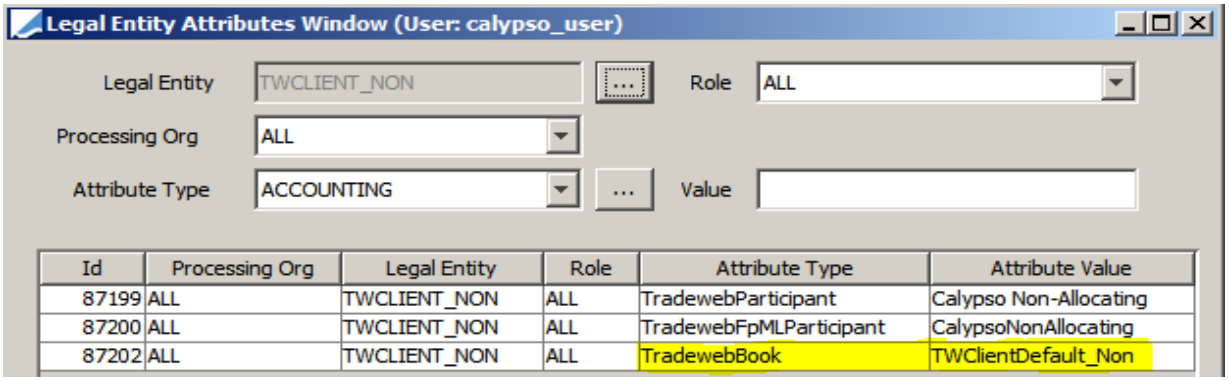
If no Calypso Book is found using the rules below, an error will be raised.

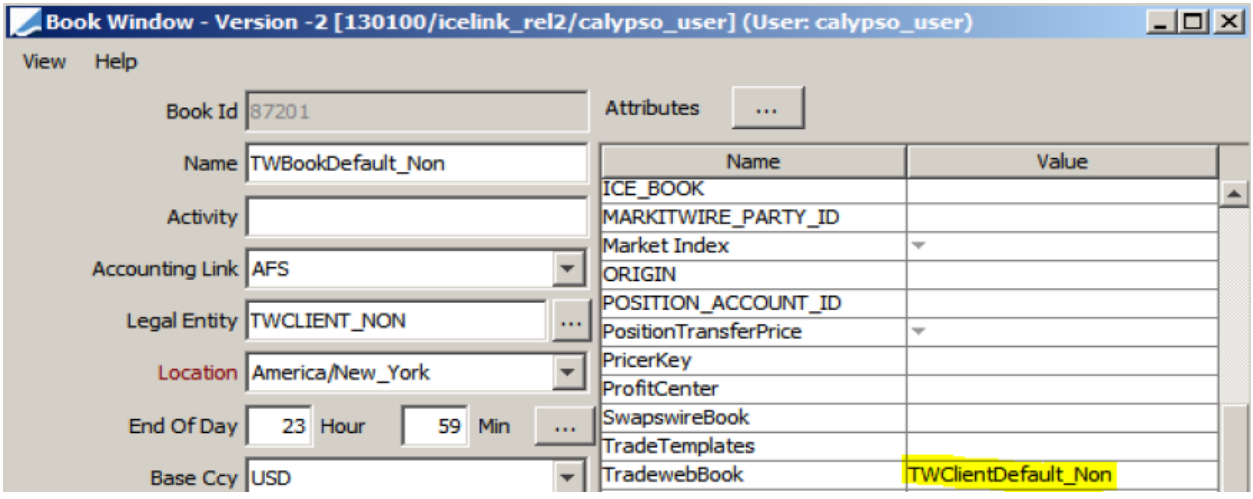
[*Note* When a Calypso Book is determined based on the lookup rules described below, the module will verify that the Book's Legal Entity matches the Calypso Legal Entity found using the rules specified in the previous section. This is especially important to note when using a parent LE; the Book must belong to the parent LE, not the child]

In a Tradeweb Non-Allocating workflow, when a trade is booked in the Tradeweb Viewer the user is not able to set any Book or Account details. However, in Calypso we require the trade to be assigned to a Calypso Book for the PO.

To address this, the Tradeweb interface allows you to set a default book at the Legal Entity level, which will be used for all incoming Tradeweb messages where that Legal Entity is the PO. Specifically, the Calypso Book is determined using the Legal Entity attribute 'TradewebBook' on the Legal Entity retrieved using the lookup rules described in the previous section.

The value in the 'TradewebBook' Legal Entity attribute is matched to a Calypso Book using the Book attribute 'TradewebBook'. If no Book is found in Calypso with that attribute value, the fetching logic will look for a Book with the short name matching the 'TradewebBook' attribute value directly (case-sensitive).





For orders, when an order is created in Calypso with a book, it is sent to the Tradeweb Platform. When the order is executed on the Tradeweb Platform and sent back to Calypso, the book from the order is set on the executed trade if the order exists in Calypso.

For a Single Executed Order from Tradeweb to Calypso:

- If Tag 35=AS, the book is set to the book with book attribute TradewebBook = Tag 79. If it is not found, the book from the PO attribute TradewebBook is used.
- If Tag 35=8 and Tag 1 contains a value, the book is set to the book with book attribute TradewebBook = Tag 1. If it is not found, an error is raised.

  If Tag 1 does not contain a value, the book from the PO attribute TradewebBook is used.

3.6 Product Mapping

When importing product-based trade types into Calypso, mappings must be setup so that the Tradeweb module 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 Tradeweb module.

The default product types are: Swap for IRS, CDSIndex for CDS and Bond for TNOTE.

You can use the following mapping to customize the products types:

Interface = Tradeweb/ProductType

Interface Value = <trade web product type>

Calypso Value = <calypso product type>

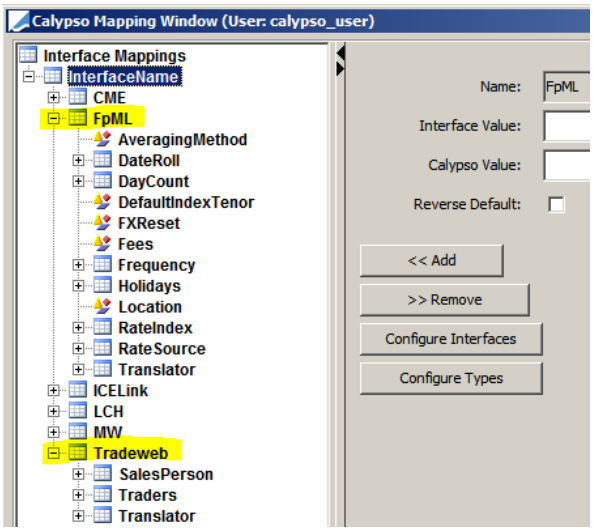
3.7 Calypso Mapping Window

[*Note* Please refer to the Data Uploader Integration Guide for information on how to add the Calypso Mapping Window to your Calypso Navigator]

The Calypso Mapping Window is required to map the Tradeweb values used in the incoming messages to the Calypso values.

The Tradeweb interface uses the Data Uploader FpML framework; therefore, most of the mapping can be done under the 'FpML' mapping interface, which allows it to be shared across other interfaces that make use of the FpML framework.

Some standard mappings are already provided for you out-of-the-box, such as DayCount and Frequency.



Rate Index Mapping

If trade keyword PlatformContractualDefinition = ISDA2021 is set on the Trade, the RateIndex_ISDA2021 mapping is used to determine the rate index instead of RateIndex.

Example:

Name = FpML/RateIndex_ISDA2021

Interface Value = USD-SOFR-OIS Compound

Calypso Value = USD~SOFR~COMPOUND

Order Rate Index Mapping

Name = Tradeweb/OrderRateIndex

Interface Value = <tradeweb rate index>

Calypso Value = <calypso rate index>

Example:

Interface Value = SOFRCMPD

Calypso Value = SOFR

Platform PO Mapping

The platform PO is retrieved from the FIX message in tag 448 when 447 = C and 452 =3. It is stored in the message attribute PlatformProcessingOrg,

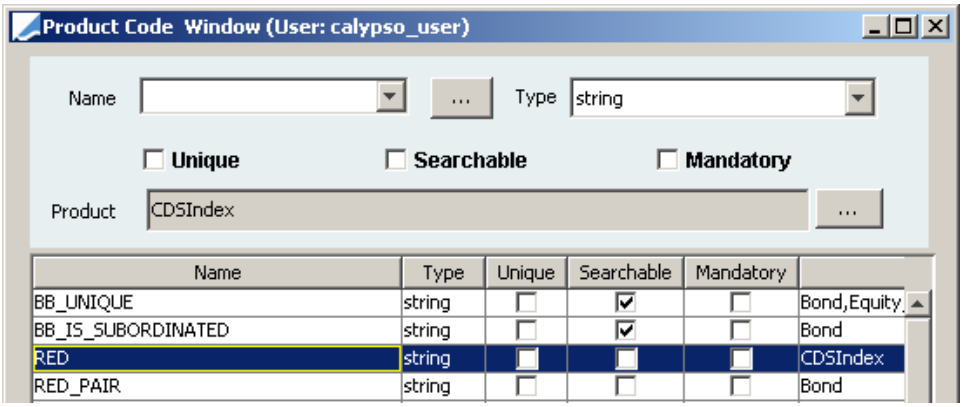
Mappings can be defined by platform PO as Name = “Tradeweb.<platform PO>” instead of Name = Tradeweb. Example, if the platform PO is PPO, you can create mapping values under Name = Tradeweb.PPO.

NOTE: If the mapping is not available for Name = Tradeweb, the system looks for a mapping for Name = "Tradeweb.<platform PO>". Tradeweb mapping takes precedence over "Tradeweb.<platform PO>" mapping.

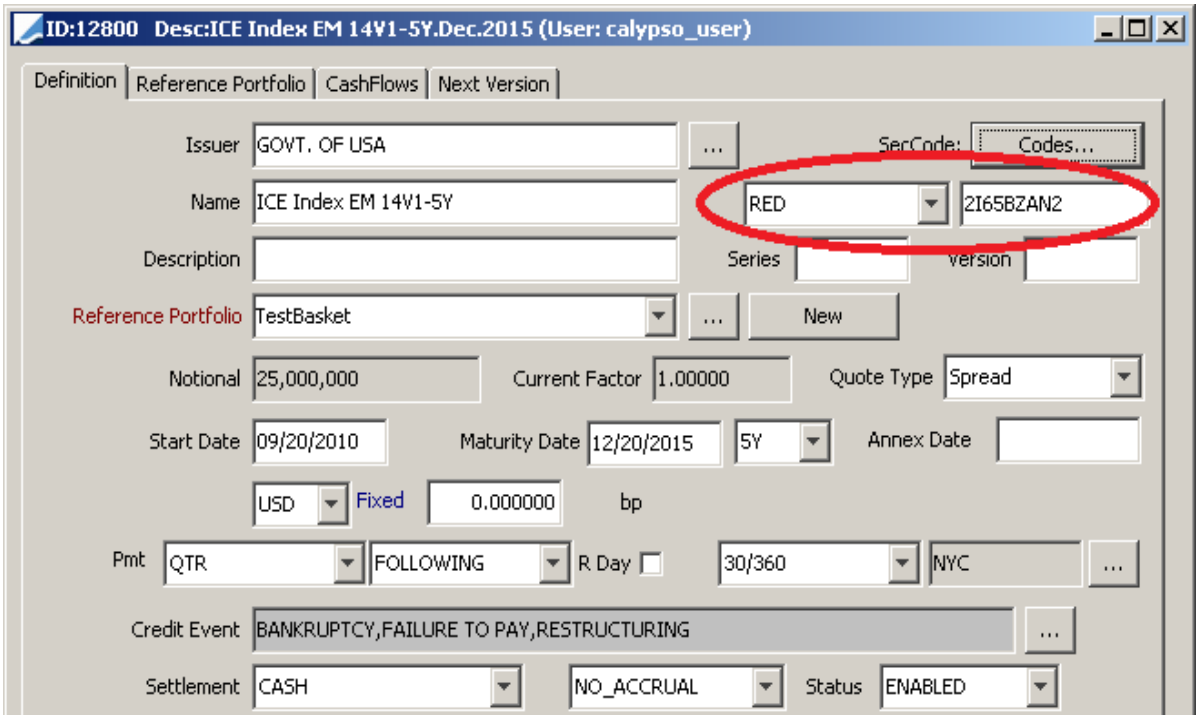
3.8 CDS Index Mapping

When importing CDS Index trades, the Tradeweb module needs a way to map the incoming CDS Index key with the corresponding CDS Index Definition in Calypso. To accomplish this, it uses the Product Code in Calypso.

The Tradeweb module can map the incoming index to the Calypso index using the 'RED' code. Please ensure that this code is setup as a valid Product Code type for CDS Index using the Product Code window under Configuration > Product > Code (screenshot below).



Then for each index you intend to receive messages for, you will need to populate the 'RED' Product Code value. Below is a screenshot of the code being set on an index called 'ICE Index EM 14V1-5Y'.



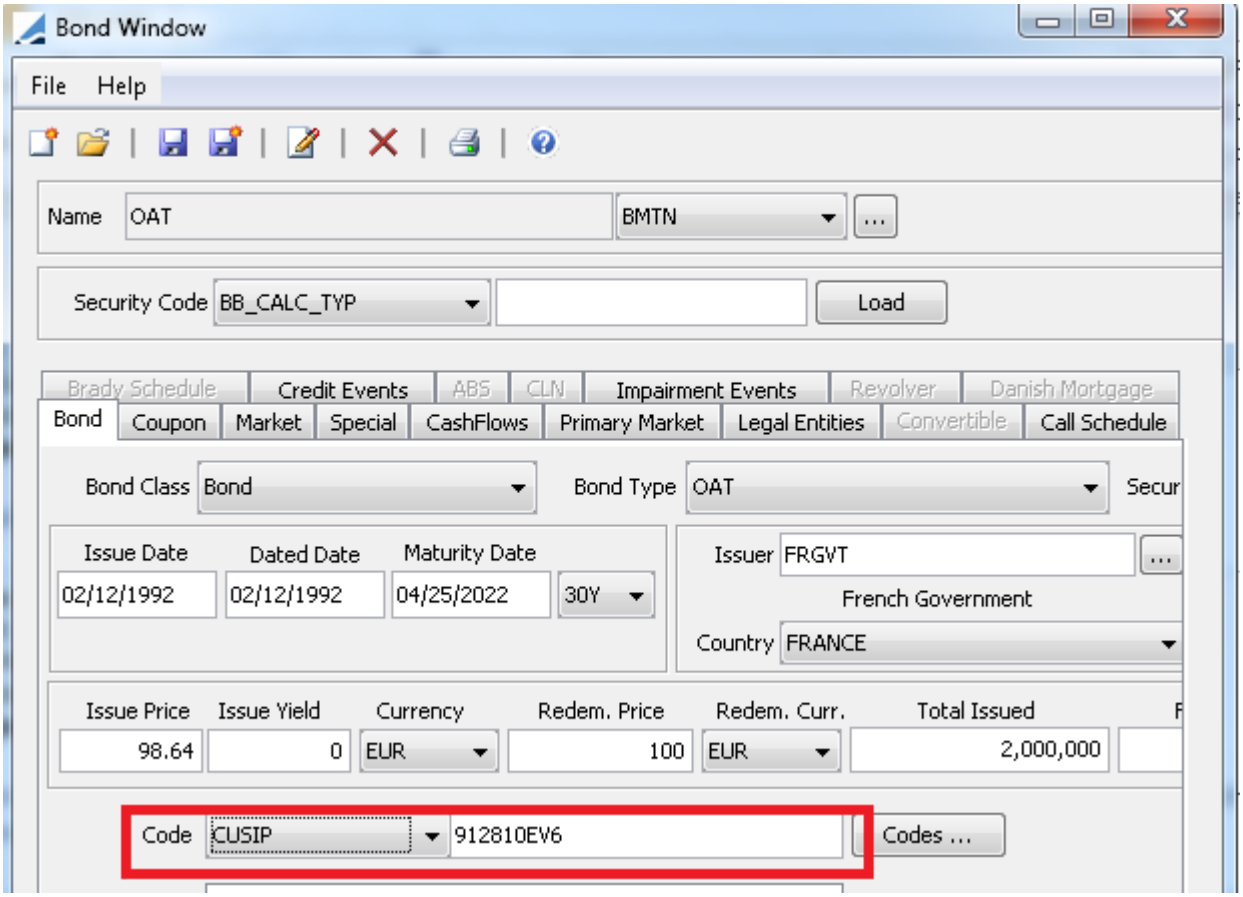
Once the code type & value is identified, the module will search for a corresponding index definition with the matching code type/value combination and the same Maturity Date as the trade. If found, the module will create the trade using that index as the underlying. Otherwise it will create an error message in the Task Station clearly identifying the code type, code value, and maturity date it tried to.

3.9 Bond Mapping

When importing Fixed Income-Bond trades, the Tradeweb module needs a way to map the incoming Bond key with the corresponding Bond Definition in Calypso. To accomplish this, it uses the Product Code in Calypso.

The Tradeweb module can map the incoming bond to the Calypso Bond using the 'CUSIP' or 'ISIN' code. Please ensure that this code is setup as a valid Product Code type for Bond using the Product Code window under Configuration > Product > Code (as mentioned above).

Then for each Bond you intend to receive messages for, you will need to populate the 'CUSIP' or 'ISIN' Product Code value. Below is a screenshot of the code being set on a fictitious bond called 'OAT' Bond.



Once the code type & value is identified, the module will search for a corresponding bond definition with the matching code type/value combination. If found, the module will create the trade using that bond as the underlying. Otherwise it will create an error message in the Task Station clearly identifying the code type, code value it tried to map.

3.10 Trade Date Mapping

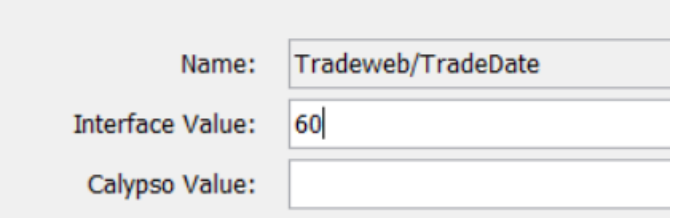
The following mapping can be used to determine the time of the Trade Date:

Name = Tradeweb/TradeDate

Interface Value = 60 or 75 (for Tag 60 or Tag 75) - Default value is 60.

Tag 60 indicates Trade Date with time

Tag 75 indicates Trade Date with no time



3.11 Trader Mapping

The following mapping can be used to determine the trader:

Name = Tradeweb/Traders

Interface Value = <Tradeweb trader>

Calypso Value = <Calypso trader>

If not set, the User Defaults trader is used.

4 Trade Workflow Setup

This section describes the Tradeweb module trade workflow. It is important to understand these details so that the Calypso Trade Workflow can be customized accordingly. Please read & follow all setup instructions carefully to ensure a successful installation.

4.1 Clearing Workflow Overview

[*Note* This sub-section contains a brief overview of the Tradeweb platform clearing workflow and how the Calypso Tradeweb integration interacts with it. For further details on the Tradeweb flow, please consult with Tradeweb]

Within the Tradeweb Viewer, the trader will setup a trade with a CCP and Block FCM, request a Check Certainty, choose a specific Dealer price, and finally execute the trade.

For a simple non-allocating clearing workflow, once the trade is executed from the Tradeweb Viewer, the Tradeweb platform will send a FIX message to both the Dealer and End User Firm (Buy-Side). At that point, the Calypso Tradeweb integration will process the FIX message and create a corresponding Bilateral trade in Calypso between the Dealer and Client.

The Tradeweb platform will also send a message to the CCP to affirm or reject the trade for clearing. Once the CCP responds, the Tradeweb platform will send a FIX message to both the Dealer and Client with the CCP's acceptance or rejection of the trade. Again, the Calypso Tradeweb integration will process the FIX message and update the Bilateral trade accordingly:

- If the CCP has accepted the trade, the Bilateral will be novated to the CCP as Counterparty. This will terminate the Bilateral trade in Calypso and create a new trade between the Client and the CCP for the cleared trade.
- If, however, the CCP rejects the trade, the Bilateral trade will be cancelled in Calypso with the rejection reason set as a keyword on the trade.

Additional keywords are added to the trade after each message is received to indicate Tradeweb status details, as well as clearing details such as the CCP, Clearing Broker, and Bilateral/Cleared USIs.

[A full list of keywords is available below]

Note that the trade external reference will be formulated as:

TW_<PO Shortname>_<Tradeweb Trade Id>

4.2 Calypso Trade Workflow Setup

To support all the transitions required by the Tradeweb workflow, the trade workflow setup for processing Tradeweb trades must support the following transitions:

- NEW (for creation of new trades)

- CANCEL (if a trade is rejected by the CCP / cancel message is received)
- TERMINATE (when the bilateral is novated after clearing)

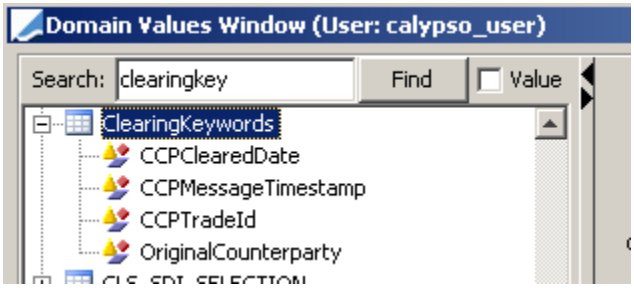
4.3 Clearing Novation Requirements

4.3.1 UpdateTermination Rule

By default, the Calypso Novation API will keep the same external reference on both the original trade and the new cleared trade. This will cause an issue if updates are received from Tradeweb for the trade. To avoid this, you *must* add the UpdateTermination trade rule to the TERMINATE transition in your trade workflow for Tradeweb trades, so that the bilateral external reference gets updated.

4.3.2 ClearingKeywords Domain

In addition, the Calypso Novation API will keep all the same keywords on both the original and cleared trades. In some cases, clients will want to remove certain clearing-specific keywords from the terminated bilateral trade. This can be controlled using the ClearingKeywords domain. Any keywords added to this domain will be removed from the bilateral trade when it is Novated.



[*Note* This domain is added and pre-populated for you by the Data Uploader + Tradeweb schema installation scripts. Please review it as part of the Tradeweb module setup]

4.3.3 Fees

By default, the Calypso Novation API will keep the trade fees on the original bilateral trade. If you wish to have the fees moved from the bilateral to the cleared trade, please add the fee name to the propagateFees domain.

[*Note* Please ensure that you restart the Calypso DataServer and EventServer after this change]

If you have mapped the default fee name (e.g. UPFRONT_FEE) using the Calypso Mapping Window, please ensure that you specify the new fee name in the propagateFees domain.

[*Note* This will only affect fees with a date after the Clearing Novation date. Also, the fees will remain on the original trade, but the Transfers will be cancelled by the Transfer Engine]

4.4 Allocation Requirements

This section describes how allocations are handled for Pre-Allocation and Bunched - Order supported by the Tradeweb module, and how each are handled.

4.4.1 Lifecycle

In case of Pre-Allocation in Tradeweb, the allocation needs to be performed before submitting it to CCP, after that the trade is cleared. The corresponding trade life cycle will be executed in Calypso for Pre-Allocation:

Tradeweb Business	Calypso-Action
Book a block trade with Allocations.	Calypso receives two messages MsgType '8': New trade is created. MsgType 'AS': Trade is Allocated.
Trade is Cleared by CCP	Calypso receives message (MsgType as 'AK' and Confirmation Status (665=4), Trade is novated.
Trade is Rejected by CCP	Calypso receives message (MsgType as 'AK' and Confirmation Status (665=5), Trade is Canceled.
Trade is canceled	Calypso receives message with tag 35=8 and tag 150=H. The trade is cancele.

Whereas in case of Bunched-Order in Tradeweb, the allocation can only be performed once the trade is submitted and cleared by CCP. The corresponding trade life cycle will be executed in Calypso for Bunched-Order:

Tradeweb Business	Calypso-Action
Book a block trade.	Calypso receives two messages MsgType '8': New trade is created.
Trade is Cleared by CCP	Calypso receives message (MsgType as 'AK' and Confirmation Status (665=4), Trade is novated.
Allocate a trade	Calypso receives a message with MsgType as 'AS', Trade is Allocated.
Trade is Cleared by CCP	Calypso receives message (MsgType as 'AK' and Confirmation Status (665=4), Trade is novated.

Tradeweb Business	Calypso-Action
Trade is Rejected by CCP	Calypso receives message (MsgType as 'AK' and Confirmation Status (665=5), Trade is Canceled.
Trade is canceled	Calypso receives message with tag 35=8 and tag 150=H. The trade and its associated allocations are canceled.

To skip allocation messages, you can use the following mapping:

Name = Tradeweb/Translator

Interface Value = IgnoreAllocation.Rekey

Calypso Value = true or false

If true, skip incoming Allocation messages

If false, process incoming Allocation messages

4.4.2 UpdateAllocationChild Rule

By default, the Calypso Allocation API will keep the same external reference on the original trade and the generated new allocated fund trade. This will cause an issue if updates are received from Tradeweb for the trade. To avoid this, you ***must*** add the UpdateAllocationChild trade rule to ***all*** the ALLOCATE transition leading to Allocated status in your trade workflow for Tradeweb trades, so that the fund trade external reference gets updated.

4.5 Workflow Keywords

Throughout a trade's lifecycle, trade status keywords will be updated on the trade to reflect its approval state. These keywords can be used together with Static Data Filters to move the trade through any custom workflow / status you create in Calypso.

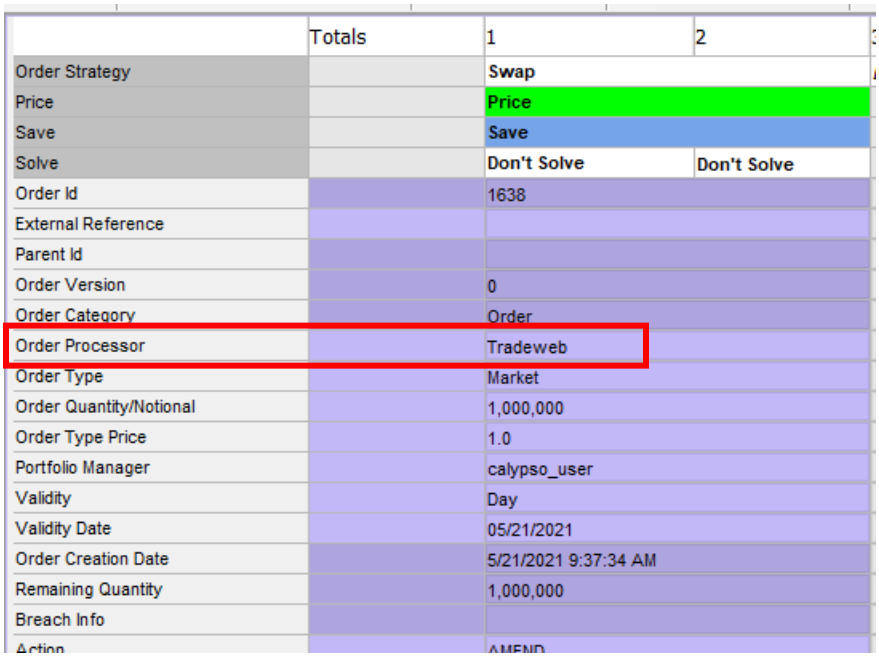
The pertinent keywords are described below:

- PlatformStatus: This keyword reflects the status of the trade from the Tradeweb platform's perspective. The trade will initially be created with a status value of 'Filled', and then 'Confirmed' or 'Rejected' depending on whether the CCP clears or rejects the trade.
- CCPStatus: Valid values are either Cleared or Rejected.

5 Order Workflow Setup

5.1 Order Processor

An order destined for the Tradeweb Platform must have **Order Processor = Tradeweb**.



	Totals	1	2
Order Strategy		Swap	
Price		Price	
Save		Save	
Solve		Don't Solve	Don't Solve
Order Id		1638	
External Reference			
Parent Id			
Order Version		0	
Order Category		Order	
Order Processor		Tradeweb	
Order Type		Market	
Order Quantity/Notional		1,000,000	
Order Type Price		1.0	
Portfolio Manager		calypso_user	
Validity		Day	
Validity Date		05/21/2021	
Order Creation Date		5/21/2021 9:37:34 AM	
Remaining Quantity		1,000,000	
Breach Info			
Action		AMEND	

5.2 Workflow Description

This section describes the Tradeweb 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 Tradeweb 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 Tradeweb 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 Tradeweb, 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.

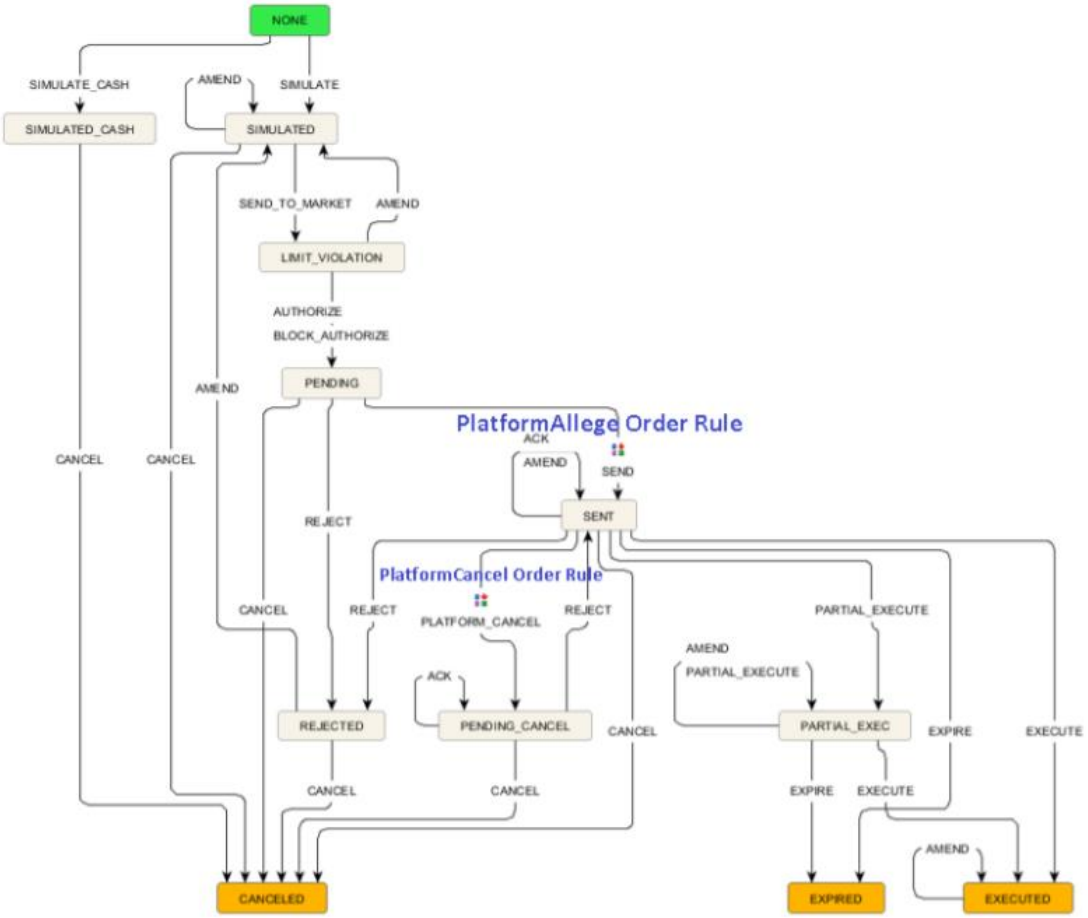
Once the message has been acknowledged by Tradeweb, the following happens:

- 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.
- Tradeweb sends the appropriate FIX message to Calypso, which is used to update the order comment with Tradeweb Order Id.

5.3 Workflow Setup

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

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



5.4 Order Capture Requirements

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

Tradeweb Business	Calypso Action
Send Order to Tradeweb	Calypso sends message: FIX Message of MsgType 'D' is sent to Tradeweb.
Confirmation message	Calypso receives message: FIX message with MsgType as '8' and with Order-Id tag (tag 11) Action: Update the Calypso order with Tradeweb status.
Execute the order in Tradeweb	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.
Cancel the order in Tradeweb	Calypso receives a FIX message with Tag 35=8 and Tag 39=4. The default action applied to the corresponding order in Calypso is ACK. However, since this is a cancelled order, you can apply a different action using the following mapping: Name = Tradeweb/OrderAction Interface Value = CANCELLED Calypso Value = <action to be applied, example CANCEL>

For bonds:

In case of block order, Tag 38 = Quantity

In case of single Order, the following mapping is used:

Name = Tradeweb/Translator

Interface Value = Nominal

Calypso Value = 38

If set, Tag 38 = Nominal ((Quantity * Face Value of Bond), otherwise Tag 38 = Quantity.

6 Bi-directional Interface Setup

6.1 Prerequisites

1. Add EFP Sell Bond Buy Future Bond in the bundleType domain

2. Install Order Bundle workflow (resources/OrderBundle.wf)

3. Ensure that all bundle actions in the OrderBundle workflow are present in OrderBundleAction domain

4. Ensure that the PlatformAllege workflow rule is present in OrderBundle workflow on BUNDLE_SEND_TO_MARKET action

5. Add the following transition to the AMOrder workflow:

PICKUP_READY - BUNDLE_SEND_TO_MARKET – SENT

6.2 Process

1. Create a new OrderBundle in Calypso having 2 legs - Bond and Future Bond

Apply BUNDLE_SEND_TO_MARKET action to send a 35=AB message to Tradeweb. This will be a PlatformMsg in Message Report.

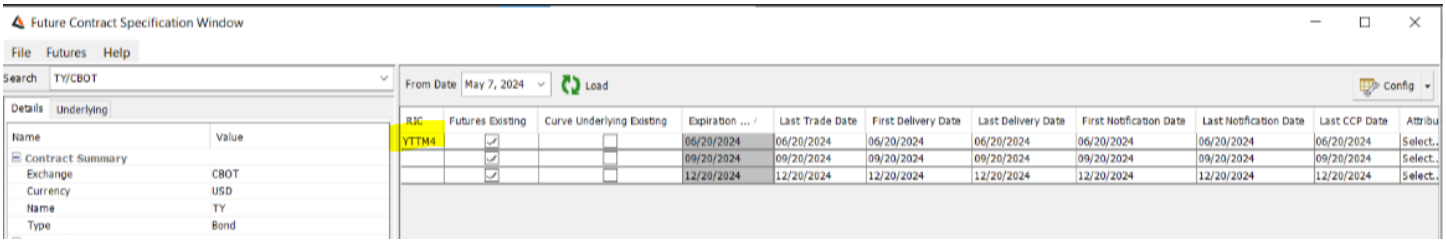
2. Ack from Tradeweb. On receiving the Ack, one CUD is created having 1 OrderBundle attributes update and 2 CalypsoOrder updates. The attributes are updated on OrderBundle and both Orders by applying action ACK. These are UploadSourceMsg and GatewayMsg

3. Execute the Order in Tradeweb

4. Trades are created in Calypso. The execution report is received for each trade separately. OrderBundle execution completes when both orders are executed

Allocation is out of scope.

5. For Future Bond Mapping, ensure that the security code is set on the future contract. Example below is of RIC security code. Similar mapping should be done in case of ISIN/CUSIP.



7 Tradeweb FIX Engine Setup

The Tradeweb FIX Engine is responsible for getting messages from the Tradeweb platform and handing it off to the appropriate workflows. It can also be run in 'Test' mode to only read FIX messages from files, which can be useful for replaying problematic messages or for Unit Testing.

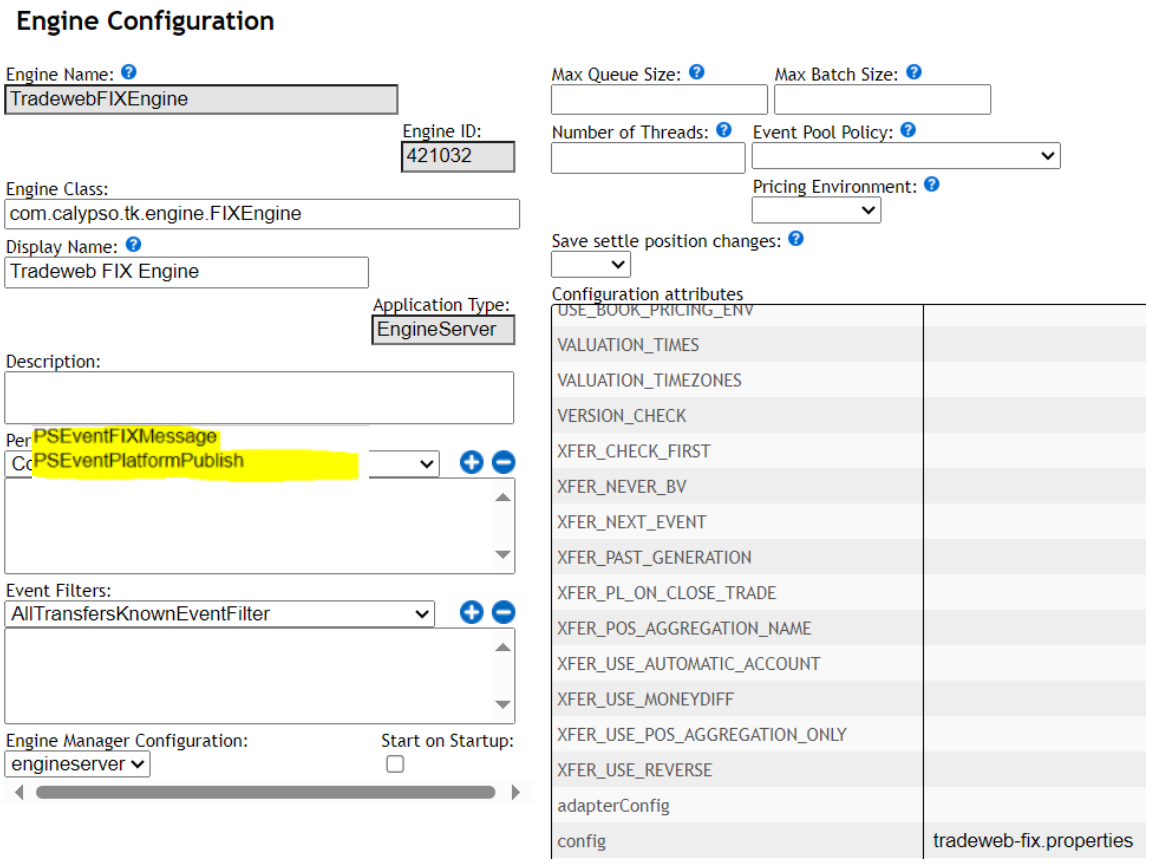
The Tradeweb FIX Engine is built on the Calypso FIX Engine framework, and therefore while setting it up you will find generic FIX setup vs. Tradeweb specific setup. For clarity, all required steps are listed below.

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.

7.1 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.



Engine parameters:

config = tradeweb-fix.properties

OPTIONAL_FEATURE = tradeweb

7.2 Setup the FIX Config File

To run the Tradeweb FIX Engine out-of-the-box you will need a properties file with the name “fix.properties” with the appropriate FIX connection settings. [You may change the filename, but it must end with “.properties”]

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

Note that, as previously mentioned, the Tradeweb 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 Tradeweb FIX Engine uses this same file for internal settings as well.

For simplicity, we have provided a sample fix.properties file and will only refer to the minimum settings that must be changed to work with Tradeweb 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

7.2.1 SSL Connection

Tradeweb module supports secure communication over SSL, using Quick-FixJ library

For SSL enabled fix session, Tradeweb platform provides SSL certificates in either JKS or PEM format.

Adding 'SocketUseSSL=Y' setting to your fix property file, enables the SSL on FIX session.

If you need to use a specific SSL certificate, configure your session as shown below.

```
# JavaKeyStore [JKS] Certificate
SocketUseSSL=Y
SocketKeyStore=/path/to/JKS/file/<your_key_identifier_name>. jks SocketKeyStorePassword=
[jks password]


# PEM Certificate
```

Convert the PEM certificate to JKS certificate and use the generated JKS as mentioned above.

7.2.2 TLS Connection

Tradeweb module support secure communications over TLS using Quick-FixJ library

For TLS enabled fix session, Tradeweb platform provide TLS certificates in following format CMS, JKS, PEM and PKCS12 format.

Tradeweb module is certified for TLS with PKCS12 format.

Adding 'SocketUseSSL=Y' setting to your fix property file, enables the TLS on FIX session.

If you need to use a specific SSL certificate, configure your session as shown below.

```
# PKCS12 Certificate
```

```
SocketUseSSL=Y

SocketKeyStore=/path/to/JKS/file/<your_key_identifier_name>.pfx SocketKeyStorePassword=
[pkcs12 password] KeyStoreType=PKCS12

EnabledProtocols=TLSv1.2

# PEM Certificate
```

Convert the PEM certificate to PKCS12 certificate and use the generated PKCS12 as mentioned above.

7.2.3 STunnel

Tradeweb module support Proxy connection using Quick-FixJ library. That means FIX connection can be established via STunnel.

Assuming STunnel is installed. In trade-fix.properties, instead of pointing to Tradeweb platform; point to IP and port where STunnel is running. And in STunnel route this incoming to Tradeweb platform.

```
# In case of STunnel, this should point to IP where STunnel is running.
SocketConnectHost=127.0.0.1
# In case of STunnel, this should point to Port where STunnel is running.
SocketConnectPort=8822
# This should be set to your SenderCompID, as provided by Platform
SenderCompID=<SENDER_ID>
# This should be set to Tradeweb's TargetCompID, as provided by Platform
TargetCompID=<TARGET_ID>
```

The SSL certificates also can be added to the STunnel to encrypt the outgoing fix message.

For STunnel from stunnel.org, add the incoming source IP address in ‘accept’ and put IP of destination i.e. Tradeweb platform in ‘connect’ property. User can also specify the SSL/TLS certificates if required.

```
[tradeweb-proxy]
client = yes
accept = 127.0.0.1:8822 <IP of source>
connect = 208.226.148.184:443 <IP of destination (Tradeweb)>
cert = <Optional; SSL or TLS certificate path>
key = <Optional; SSL or TLS certificate password>
```

7.2.4 Proxy Connection

Tradeweb module support Proxy connection using Quick-FixJ library. That means FIX connection can be established via Proxy Server.

QuickFixJ-1.6.4 supports two types of proxy:

- Socks Proxy
- HTTP Proxy

Socks proxy can be used with and without SSL, whereas HTTP proxy only works without SSL. Below is the quick-fix-j configuration for proxy.

```
# The [DEFAULT] section contains default settings for all sessions
# These are inherited by each session defined below unless they are overridden in the session settings
# If you're not sure about something, use the default for now
[DEFAULT]
ConnectionType=initiator
ReconnectInterval=10
HeartBtInt=20
LogonTimeout=20
LogoutTimeout=20
Calypso.LogOnInterval=5000
Calypso.LogOnRetryCount=5

# PROXY Support
ProxyType=socks
ProxyVersion=5
ProxyHost=<set_proxy_host>
ProxyPort=<set_proxy_port>
ProxyUser=<set_proxy_user_password_if_any>
ProxyPassword=<set_proxy_user_password_if_any>
```

Since application software for HTTP and SOCKS proxy can be different and they are hosted in different IP and PORT.

So, like HTTP, we have a separate configuration property for SOCKS in CalypsoEnv property file.

HTTP Configurations	SOCKS Configurations
HTTP_PROXY_HOST HTTP_PROXY_PORT	SOCKS_PROXY_HOST SOCKS_PROXY_PORT
HTTP_PROXY_USERNAME	SOCKS_PROXY_USERNAME
HTTP_PROXY_PASSWORD	SOCKS_PROXY_PASSWORD

For proxy connectivity, our FIXEngine logic will be,

- If property **ProxyType** in tradeweb-fix.properties not empty
  - Check for proxy information present in fix properties - Use that to connect to Tradeweb via proxy.
  - Else check for proxy information in calypsouesr.properties - Use that to connect to Tradeweb via proxy.
  - Else throw an error.
- Else
  - Connect to Tradeweb directly.

[Note: For proxy configuration, first preference will be given to tradeweb-fix.properties]

7.2.5 Sample properties file

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

```
# This file is used by the QuickFIXJ library for FIX Connection/Session Settings
# It also contains custom Calypso properties prefixed with "Calypso."
#
# Please read the Calypso Tradeweb documentation for more details on the relevant
```

```
# settings below.
# You can also review all available QuickFIXJ settings at
# http://www.quickfixj.org/quickfixj/usermanual/1.5.3/usage/configuration.html

# The [DEFAULT] section contains default settings for all sessions
# These are inherited by each session defined below unless they are overridden in the
# session settings
# If you're not sure about something, use the default for now
[DEFAULT]
ConnectionType=initiator
ReconnectInterval=10
HeartBtInt=20
LogonTimeout=20
LogoutTimeout=20
Calypso.LogOnInterval=5000
Calypso.LogOnRetryCount=5

# Turn on test mode so that the FIX Engine does not try to connect to any
# of the sessions listed in this file.
# It will still process messages generated from the File Watcher component.
#Calypso.TestMode=false

# This controls which mode the uploader will run in.
# See the Data Uploader documentation for more details.
Calypso.UploadMode=Local
Calypso.PersistMessages=All

# Each [SESSION] section contains session specific settings
# A single fix.properties file can be used for multiple sessions,
# and they will be handled in parallel by the FIX Engine

[SESSION]
# This entry should never be changed
Calypso.FIXMessageType=Tradeweb

# This is the FIX version used by Tradeweb
BeginString=FIX.4.4
```

```
# This should be set to your SenderCompID, as provided by Tradeweb
SenderCompID=<TradeWebSenderCompId>

# This should be set to Tradeweb's TargetCompID, as provided by Tradeweb
TargetCompID=<TradeWebTargetCompId>

# This is the Data Dictionary for Tradeweb's dialect of FIX
# It should only be updated based on instructions from Calypso support
DataDictionary=DD_Tradeweb_FIX.xml

# This is the connection information for the Tradeweb FIX server
SocketUseSSL=Y
SocketConnectHost=<TradeWebFIXServerIPAddress>
SocketConnectPort=<TradeWebFIXServerPortNumber>
SocketKeyStore=<Certificate jks>
SocketKeyStorePassword=<password>

# These settings control the logging for the QuickFIXJ component
ResetOnLogon=Y
ResetOnLogout=Y
ResetOnDisconnect=Y
RequiresOrigSendingTime=N
ValidateIncomingMessage=N
FileLogHeartbeats=Y
FileIncludeMilliseconds=Y
FileIncludeTimeStampForMessages=Y

# These settings control when the engine will start & end a new
# session daily with Tradeweb, and should be based on the daily
# start/end time provided by Tradeweb
# Please see the Calypso Tradeweb setup documentation for more details
# Please see the QuickFIXJ documentation for appropriate values
StartTime=07:00:00
EndTime=23:00:00
TimeZone=America/New_York
```

To send on multiple sessions using a single properties file, you need to map tags 56 and 49 for each trader as below.

Name = <TraderName>FIXSession

Example: JOHNDFIXSession for trader name JOHND

Interface Value = 49

Calypso Value = CALYPSO1

Name = <TraderName>FIXSession

Interface Value = 56

Calypso Value = TRADEWEB

7.2.6 QuickFIXJ Settings

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

Additional points to note regarding the core QuickFIXJ settings:

- SocketUseSSL=Y is required for Tradeweb connectivity.
- The QuickFIXJ settings allow you to configure multiple sessions in a single properties file. This means if you have multiple Tradeweb session logins, you can use a single Tradeweb 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 at this time.

7.2.7 Calypso Settings

There are two Calypso settings to note from the sample:

- Calypso.TestMode: This setting should be placed in the DEFAULT section of the properties file. If set to 'true', the Tradeweb FIX Engine will **not** connect to ANY session defined within the properties file. This can be used when using the File Watcher component to load the FIX messages via files rather than connecting to Tradeweb. If not set to 'true', the engine will support **both** messages from Tradeweb as well as from files.
- Calypso.FIXMessageType: This setting should be placed in the SESSION section(s) of the properties file. The value ***must*** be 'Tradeweb' for Tradeweb sessions.

7.3 Setup the gatewayservice config file (Advanced)

In addition to viewing exception messages in the Task Station, you may also configure Tradeweb to write all Success or Reject messages to the file system. This is configured within the gatewayservice.properties file distributed with Data Uploader.

A sample file is included under <calypso home>/client/resources with the name ".sample" as the suffix. To configure messages to be written to the file system, set the following in the config file:

```
TradewebPublishers=File,RejectedFile
```

At the least, you must have an empty entry (ie. `TradewebPublishers=`) to avoid any spurious errors in the log.

You can customize the `FilePublishToLocation` property to control where these files are written.

7.4 Launching the Tradeweb FIX Engine

7.4.1 Adding Logging Categories

To see logging messages for the Data Uploader and Tradeweb 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.
- Tradeweb: Set this to see logging for the Tradeweb 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 Tradeweb FIX Engine.

Additional debugging categories are listed in the Troubleshooting section of this document.

7.4.2 Running the Tradeweb FIX Engine

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

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

With the Tradeweb Engine operating, you can then allege/affirm trades through the Tradeweb Viewer. The Tradeweb Engine will process the trade messages and create corresponding trades in Calypso.

The Task Station will display any errors that may occur.

7.4.3 Daily Stop/Restart

The Tradeweb FIX server's shutdown 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 Tradeweb FIX Engine handles this for you automatically, based on the values set in the "tradeweb-fix.properties" 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.

8 Performance Improvement

To improve the performance for message processing we have now the provision to perform the upload process in the Engine side via the API instead of DataServer via the workflow rules. Workflow rules are executed in the data server thus keeping DataServer busy every time data is uploaded. Client side execution has the advantage of using the API and cache. And it can thus perform better and DataServer is free for performing other tasks.

The new enhancement provides the equivalent features that were available via the workflow based approach such as:

- Persistence of external messages as BO messages.
- Ability to Re-process failed messages.
- Maintain the order in which the messages are received.
- Acknowledgement generation.

To enable the clients to use the new approach, we have introduced two new properties in “tradeweb-fix.properties” and “tradewebfixuploader.properties”. Please note that if these are not set we will default to the current way of processing in DataServer via the workflow rules which is the BOMessage mode.

The following are the properties:

- uploadMode
- persistMessages

```
#Valid combination for uploadMode and persistMessage
#
#                        |-------------------------------------------------------------------|
#                        |                          PERSIST MESSAGES                         |
#                        |-----------------|-------------------------------|-------------------|
#                        |       None      |     Failure                   |     All           |
#       |---|------------|-----------------|-------------------------------|-------------------|
#       | M | Local      | No BOMessage    |  BOMessage will be created    |  BOMessage will be |
#       | O |            | will be created |   only in case of failure     |  always created    |
#       | D |------------|-----------------|-------------------------------|-------------------|
#       | E | BOMessage  | Not Applicable  |     Not Applicable            | BOMessage will be  |
#       | S |            |                 |                               | always created     |
#       |---|------------|-----------------|-------------------------------|-------------------|
#
#   For better performance it is recommended to use uploadMode as Local with persistMessages as Failure.
```

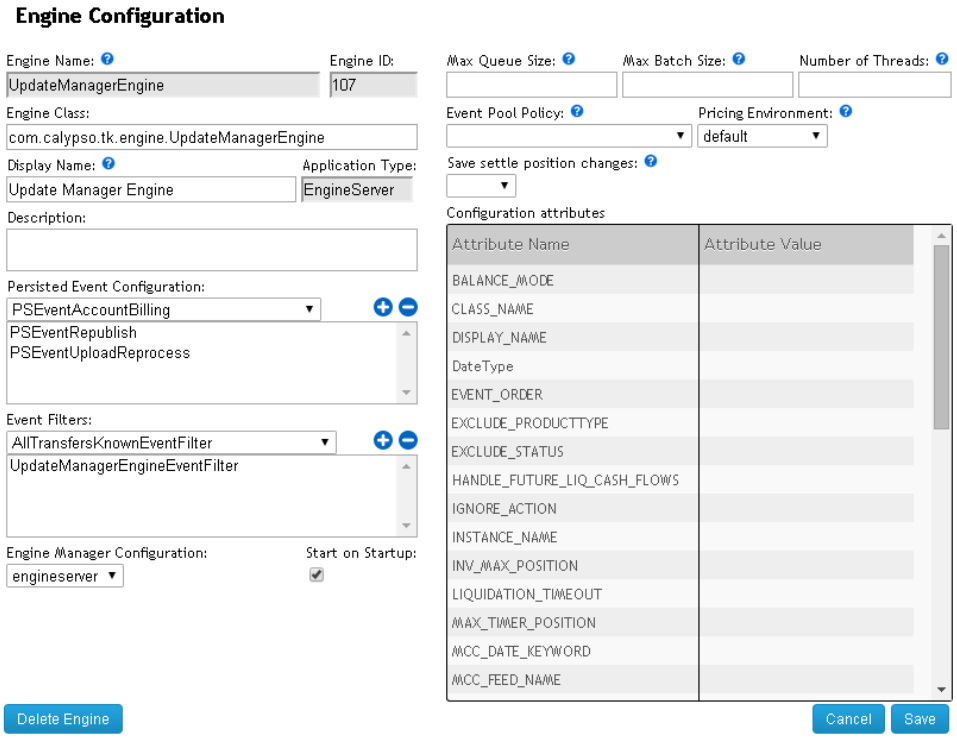
- uploadMode = Local
- persistMessages: Possible values are ‘All’, ‘None’ and ‘Failure’, by default it is set to ‘None’.
  - All: External messages are always persisted as BO messages.
  - None: External messages are _NOT_ persisted as BO messages. If the message fails in translation or validation, the message needs to be resent or handled via custom code.
  - Failure: External Messages are persisted only in case of failure in translation or validation. This is the _recommended_ configuration for the ‘Local’ mode as this will not save any BO messages in case it is all

processed fine and only create messages in case of any translation failures which will enables the failed messages to be reprocessed as before and also improve performance by not saving the success BO messages.

Please note that for message reprocessing, "**UpdateManagerEngine**" needs to be running. Failed messages will be stuck in "PENDING" status and pending Messages are re-processed via the UpdateManagerEngine. The message workflows are changed to generate an event 'PSEventUploadReprocess', every time a failed message is re-processed. Update Manager Engine would then receive these events and processes them again and generate acknowledgement if needed.

The following Events are required by the Update Manager Engine.

- PSEventUploadReprocess



9 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 Tradeweb FIX Engine also supports processing FIX messages from files. To achieve this, you must run the Data Uploader FileWatcher in 'tradeweb' mode, so that it will load files from a specified location and pass them on to the Tradeweb FIX Engine.

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

9.1 Setup the FileWatcher Config File

To run the FileWatcher in 'tradeweb' mode, you will need a properties file with the appropriate settings. A sample file "tradewebuploader.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 Tradeweb FIX Engine's property file.

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

9.2 Launching FileWatcher

9.2.1 Adding Logging Categories

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

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

9.2.2 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 Tradeweb FIX Engine.

The Task Station will display any errors that may occur. Please also review requirements for FileWatcher from the Data Uploader Integration Guide.

10 Test Tool Setup - GUI

[*Note* The details in this section are provided for testing purposes only, and not recommended for production use]

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

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

10.1 Setup the GUI Config File

[*Note* The need for this step will be removed in a future release]

To upload Tradeweb FIX files through the GUI, you will need a properties file with the appropriate settings. A sample file "tradeweb-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 Tradeweb 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, as previously described.

10.2 Uploading via the GUI

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

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

[*Note* The uploaded file must have a '.fix' extension, not '.xml']

11 Trade Keywords

In addition to the keywords listed above in the "Trade Workflow" section, the Calypso Standard Clearing keywords – as well as Tradeweb specific keywords – will be populated for incoming trades accordingly. A full list is provided in the table below.

[*Note* This is an early release version and keyword names are subject to change. Such changes will be documented in the Release Notes, so please ensure you review them whenever upgrading from previous Tradeweb module versions]

Keyword Name	Description	Comments
TradeSource	Always set to 'Tradeweb'	Used for engine logic behind the scenes; do not change.
PlatformAPIUser	FIX Session Id	Used for engine logic behind the scenes; do not change.
USIPrefix	Tradeweb Regulatory Reporting value	
USIValue	Tradeweb Regulatory Reporting value	
ReportingRepository	Trade Repository that this trade will be reported to.	
CCP	LE short name of CCP	
CCPClearingBroker	The clearing broker (when available in the trade)	
OriginalCounterparty	Bilateral counterparty	Set on the cleared trade, to show the original counterparty before novation
Platform	Always set to 'Tradeweb'	
PlatformStatus	Tradeweb Trade Status	
PlatformTransactionId	Tradeweb Transaction Id	Set to SecondaryOrderID which connects all messages for a single trade
PlatformTradeId	Tradeweb Trade Id	As per Tradeweb internal identifier

Keyword Name	Description	Comments
CCPStatus	Sending, Cleared, Rejected	
CCPClearedDate	Date when trade was cleared by CCP	
CCPMessageTimestamp	GMT timestamp of last message to/from CCP	
CCPTradeID	CCP assigned deal id	
PriorUSIPrefix	Tradeweb Regulatory Reporting value	Set on a cleared trade, with the USI Prefix of the original bilateral trade
PriorUSIValue	Tradeweb Regulatory Reporting value	Set on a cleared trade, with the USI Value of the original bilateral trade
PlatformRejectReason	Reject reason provided in Tradeweb message	
CCPAccount	Is this trade in a CLIENT or HOUSE account at the CCP	Always set to 'CLIENT'
CCPOriginCode	Did this trade originate due to CLIENT or HOUSE activity	Always set to 'CLIENT'
PlatformPO	The PO Legal Entity short name which maps to the original Tradeweb message value	This is only set if the Calypso LE was pointing to a Parent LE, and therefore the child LE is preserved as a keyword
PlatformCP	The CP Legal Entity short name which maps to the original Tradeweb message value	This is only set if the Calypso LE was pointing to a Parent LE, and therefore the child LE is preserved as a keyword
AllocationClearingType	Specifies whether this trade is part of a Pre-Allocation or Post-Allocation allocation workflow	Set to "Pre-Allocation" if pre-allocation, "Bunched-Order" if bunched-order, and empty if this is not part of an allocation workflow at all
BlockUSIPrefix	Tradeweb RegReporting value	Set on an allocation leg trade, with the USI Prefix of the allocation block trade
BlockUSIValue	Tradeweb RegReporting value	Set on an allocation leg trade, with the USI Value of the allocation block trade

Keyword Name	Description	Comments
CCPConfirmationId	Tradeweb CCP confirmation Id	
InstrumentUPI	UPI	It also populates the UPI product code
ReportTrackingNumber	Report tracking number	

12 Troubleshooting

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

12.1 Connectivity

The Tradeweb FIX Engine automatically attempts to reconnect if a connection with the Tradeweb 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 Tradeweb FIX Engine logs.

12.2 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.

In addition, if you have configured the gatewayservice.properties file to log message processing errors to files, you can review those files as well.

12.3 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 Tradeweb platform as well as the interim Calypso xml format created by the translation under <user home>/Calypso/Tradeweb. 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 Tradeweb platform before they are interpreted by the Tradeweb FIX Engine.

12.4 Reporting Issues to Calypso

If after attempting all of 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
  - Tradeweb FIX Engine logs

  - Tradweb FileWatcher logs
  - DataServer & EventServer logs
- FIX messages, if applicable
- Clear description of the issue, including:
  - What steps were executed to produce the issue, both in the Tradeweb Viewer and Calypso
  - What attempts, if any, were made to debug the issue, and what were the results