Nasdaq Calypso

MTM Integration Guide

Version 4.3.0

Revision 10.0
August 2024
Approved

Document History

Revision	Published	Summary of Changes
1.0	July 2014	First edition for version 1.0.
2.0	November 2018	Updates for version 2.2.0.
3.0	December 2020	Updates for version 2.2.1, 2.2.2.
4.0	February 2021	Updates for version 2.2.3.
5.0	August 2021	Updates for version 2.3.0.
6.0	October 2021	Updates for version 2.4.0.
7.0	December 2021	Updates for version 2.5.0 is a technical release only.
8.0	January 2022	Updates for version 3.0.0, 3.0.1 - Technical release only – Compatibility for version 17
9.0	January 2024	Updates for version 4.3.0 - Technical release only – Compatibility for version 18
10.0	August 2024	Updates for version 3.3.0 – Release notes.

This document describes the Calypso MTM Interface (MarkitSERV Trade Manager).

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 MTM Interface setup. MarkitSERV Trade Manager (MTM) is an API Solution for the Buy Side Firms. MTM provides the Buy side firms a single gateway to connect to MARKITSERV suite of services such as Allocation Delivery, Affirmation, Confirmation and Clearing for the following Transaction types

- Trade
- Novations
- Terminations
- Amendment

This document assumes that the Buy Side firm is using Calypso as their front office system, and the sell side uses Markitwire as the affirmation platform to allege the trades. Once both trades are entered in the respective systems and sent to Markit Trade Manager, MTM would then match the trades and provide Clearing and Allocation delivery services to the Buy side.

Clearing the Trades

The trades sent to MTM for clearing should contain the following keywords.

- CCP
- CCPClearingBroker

The keywords are sent to MTM in XML via clearing details information as follows:

```
<ClearingDetails>
   <ClearingHouse>AUTO_SIM</ClearingHouse>
   <ClearingBroker>BSAPISEL01</ClearingBroker>
</ClearingDetails>
```

Once both parties allege the trades, MTM matches the trades and sends the trade for clearing to the selected CCP. Once the trade is cleared, MTM sends the responses back to the Buy side, resulting in a novation to CCP.

NOTE: Incoming allege messages are not currently supported as MTM does not send the trade details, and only sends the notification.

Allocation Delivery

If the account of the trade submitted to MTM is a Block account, it is treated as an allocation, in which case, after the trades are matched, MTM would deliver the allocation details to the sell side.

Interface Components

The interface comprises the MTMImportMessageEngine which listens to a Message Queue (MQ). It subscribes to the Message and Trade events. The events are triggered as soon as a message is posted into the input queue.

The Sender engine is responsible for sending out acknowledgements back to MQ depending on the status / state of the message / trade.

The MTM message flows through the configured workflows that route the message to appropriate stages until completion.

The document describes the configuration required to setup the workflows, engine etc. for the MTM module to run successfully.

Product Coverage

The Scope of this project is to build an interface to MTM and support Clearing workflows and Allocation Delivery of the following product types:

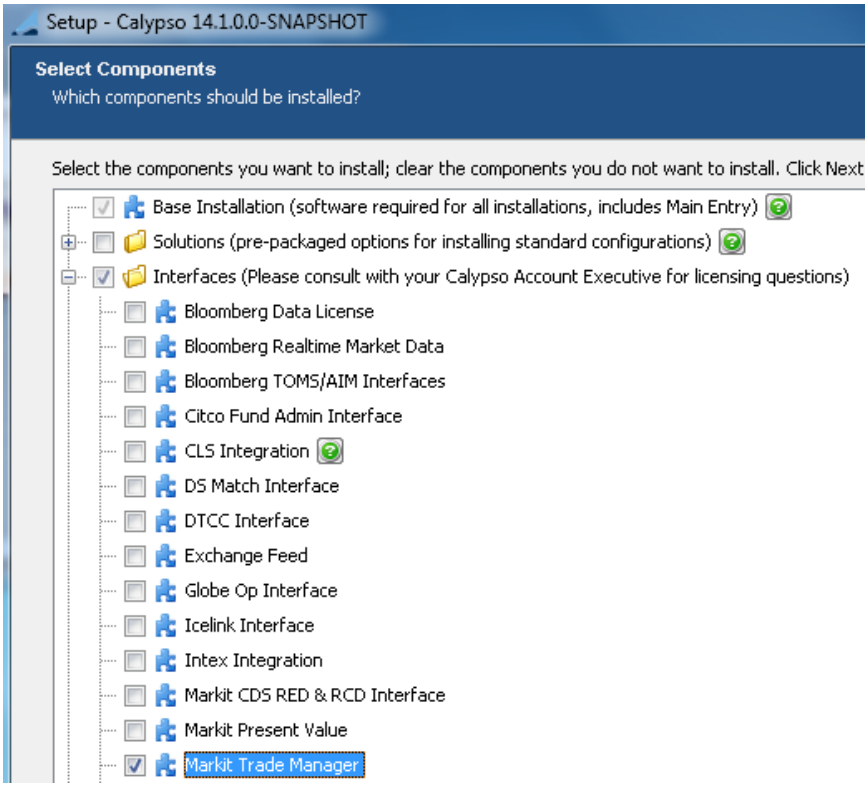
- InterestRateSwap
- CreditDefaultSwap
- CDSIndex

For IRS the family, the trade confirmation / clearing, and allocation delivery are handled by MTM. However, for Credit only the clearing and allocation delivery are handled, for bilateral confirmation the trades will be routed to DS Match by the MTM application.

Installation

2.1 Installation Instructions

The MTM module is installed as part of the Calypso Installer when you select the “Markit Trade Manager” interface:



▶ Please refer to the Calypso Installation Guide for complete details on the Calypso Installer.

If you are installing a Calypso Upgrade package instead, the instructions are also in the Calypso Installation Guide.

When you run Execute SQL as part of your installation, load the following files if they are not already loaded:

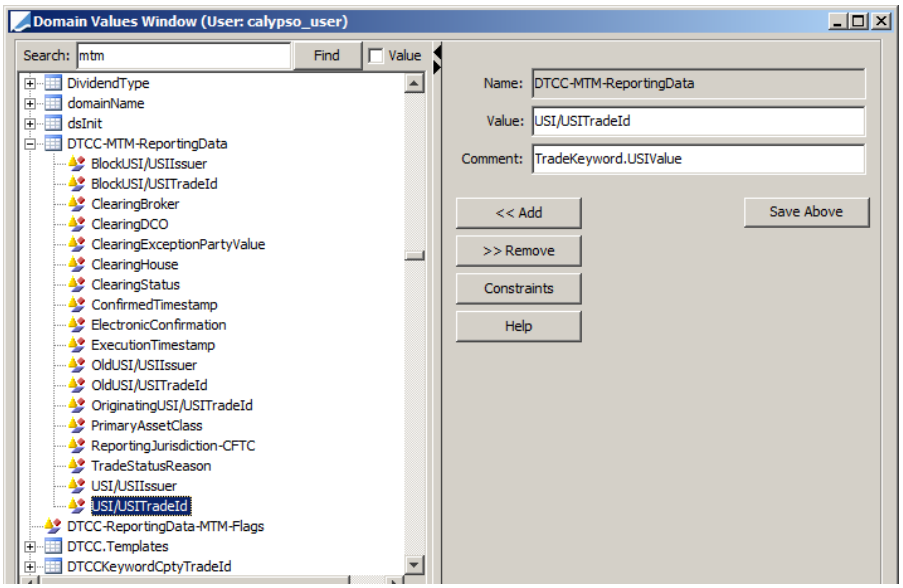
- GatewaySchemaBase.xml
- MTMSchemaData.xml

2.1.1 Domain Values

Running the MTM Schema Data will create the following data.

Domain "DTCC-MTM-ReportingData"

This domain contains the information that needs to map to trade keywords.



Domain "leAttributeType"

Values: DTCC_LE_ID, DTCC_LE_TYPE

Domain "creditDefaultSwapUpfrontFeeType"

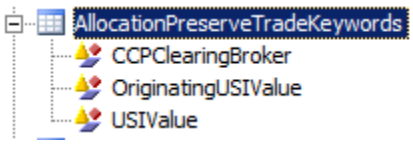
Value: UPFRONT_FEE

Domain "cdsIndexUpfrontFeeType"

Value "UPFRONT_FEE"

Domain "AllocationPreserveTradeKeywords"

The trade keyword specified in this domain will be propagated to child trade when an allocation occurs.



Domain "PropagateBlockTradeChangesAction"

This domain indicates the action to be set on the trade when propagating information during an allocation.

2.1.2 Libraries Dependencies

The following libraries are required as dependency for the MTM module, which can be downloaded from the download website.

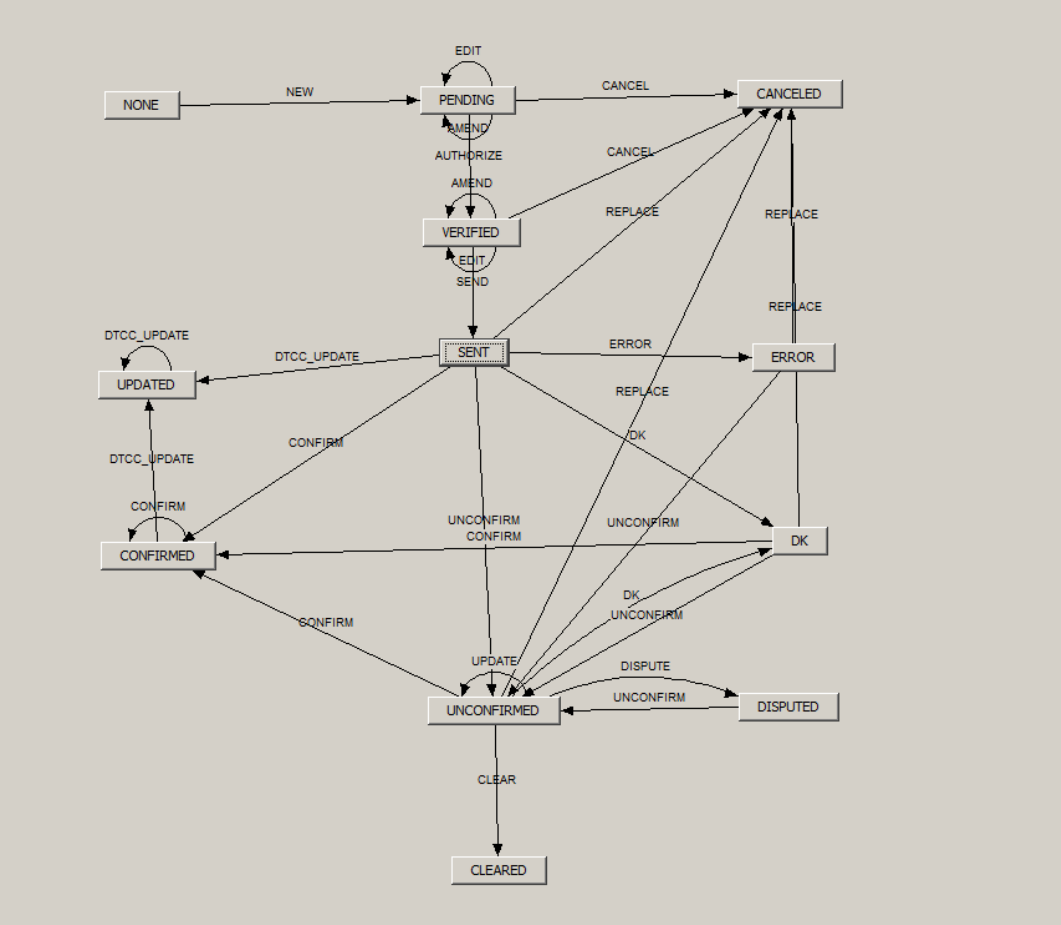
The libraries for the respective calypso version should be downloaded for each one mentioned below. Once downloaded based on the calypso version the libraries need to be installed following the installation process recommended.

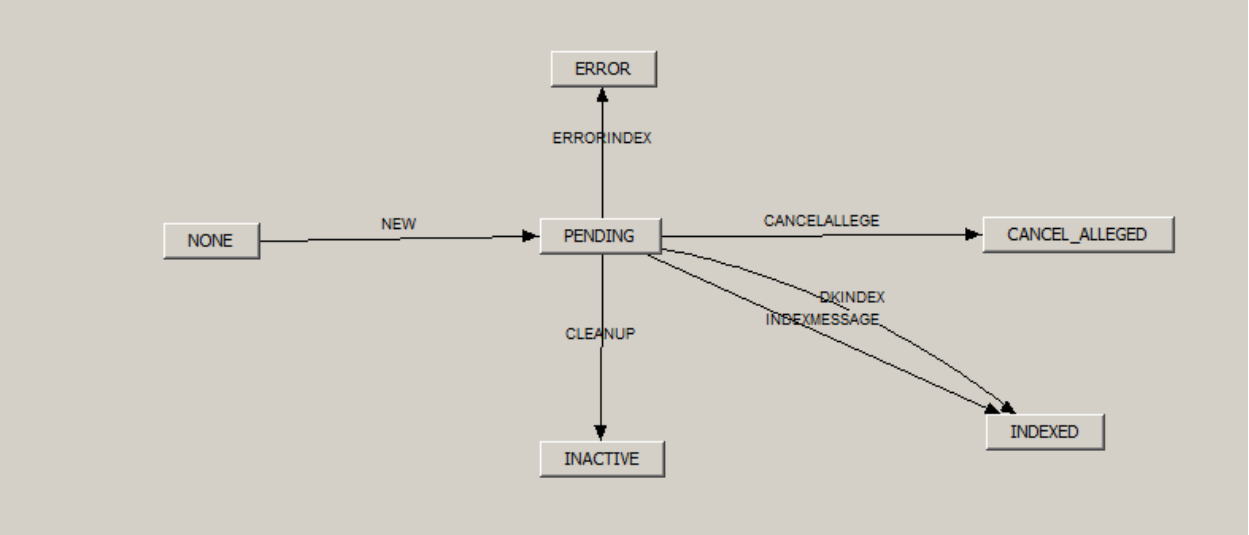
- data uploader
- dtcc-common
- derivserv-core
- derivserv-protocol

Setup Requirements

3.1 Message Workflows

The message workflows required by the MTM interface for incoming and outgoing messages can be imported into the system using the files "<calypso home>/clients/resources/MTM_CONFIRM.wf" and "<calypso home>/clients/resources/MTM_INCOMING.wf".

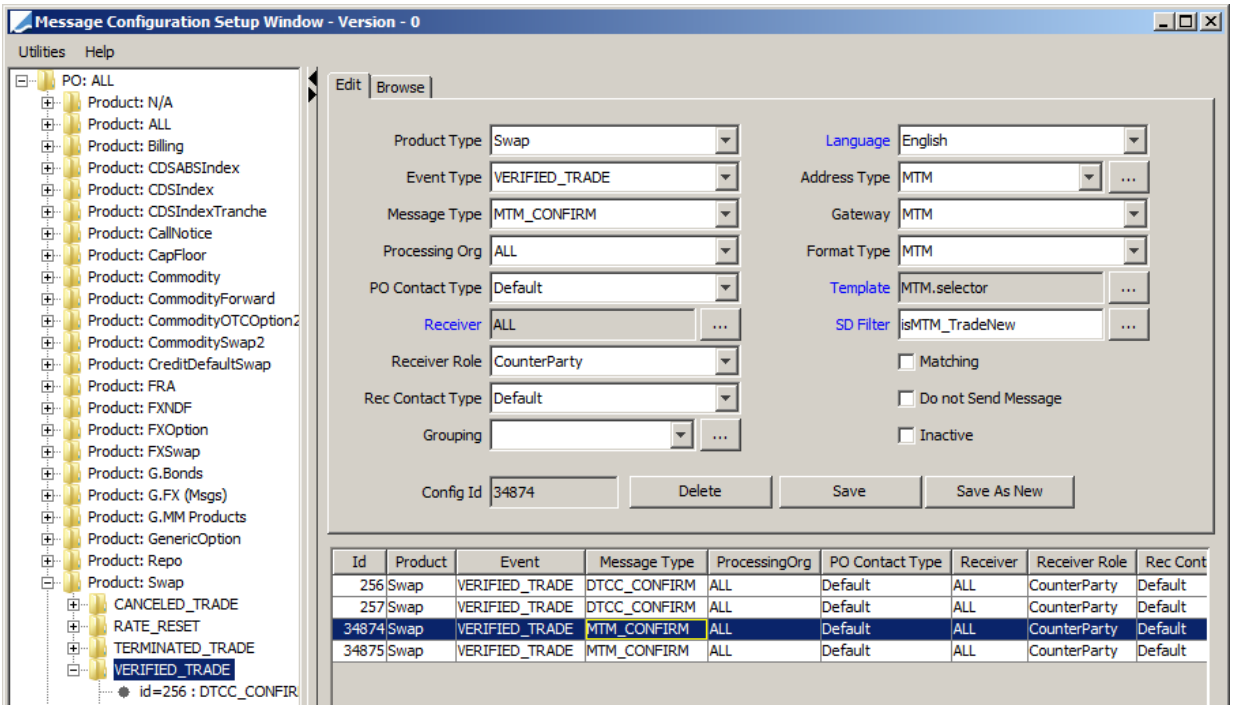


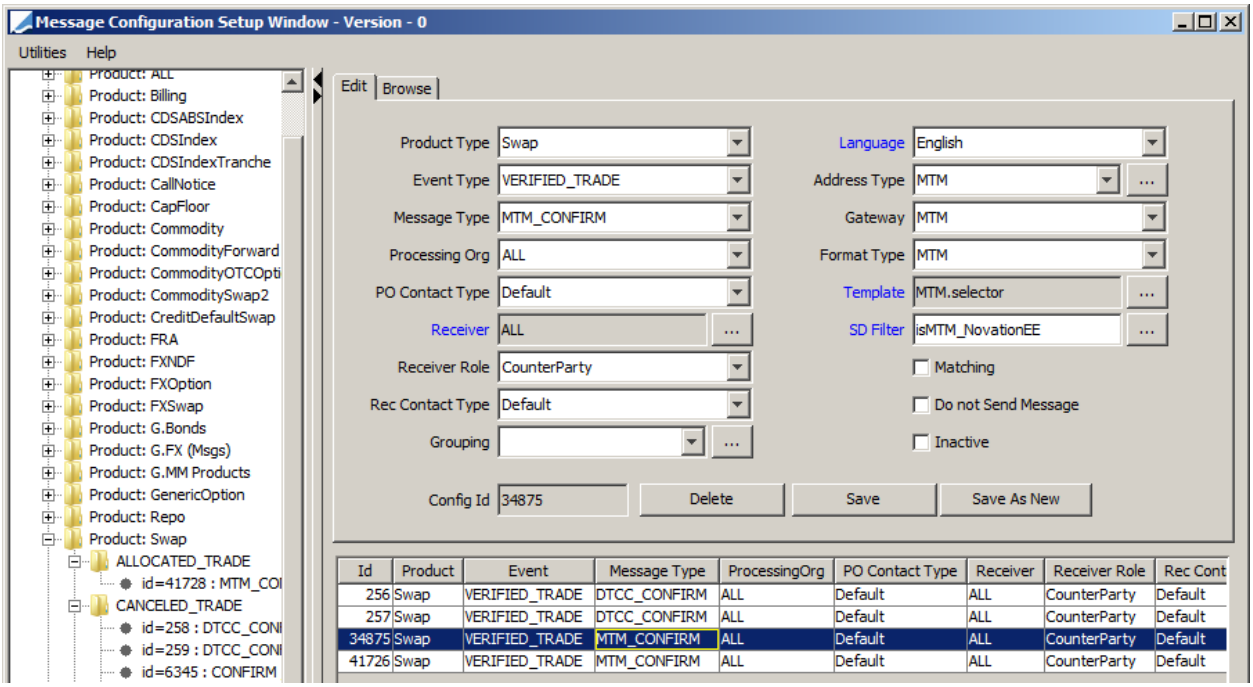


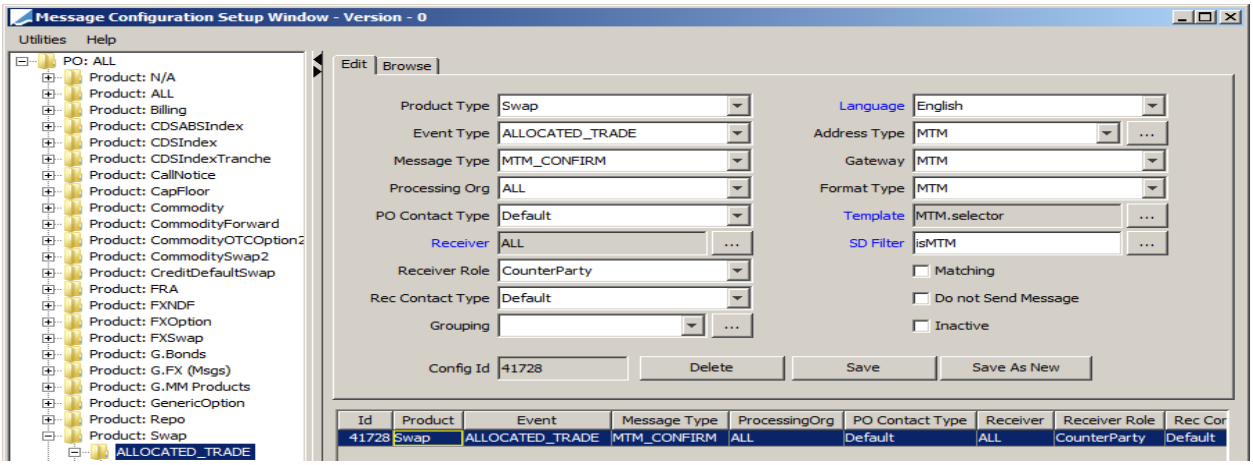
NOTE: The Allocation workflow is required for allocation. Please refer to the core calypso documentation of trade allocations and set up the workflow as required for allocations.

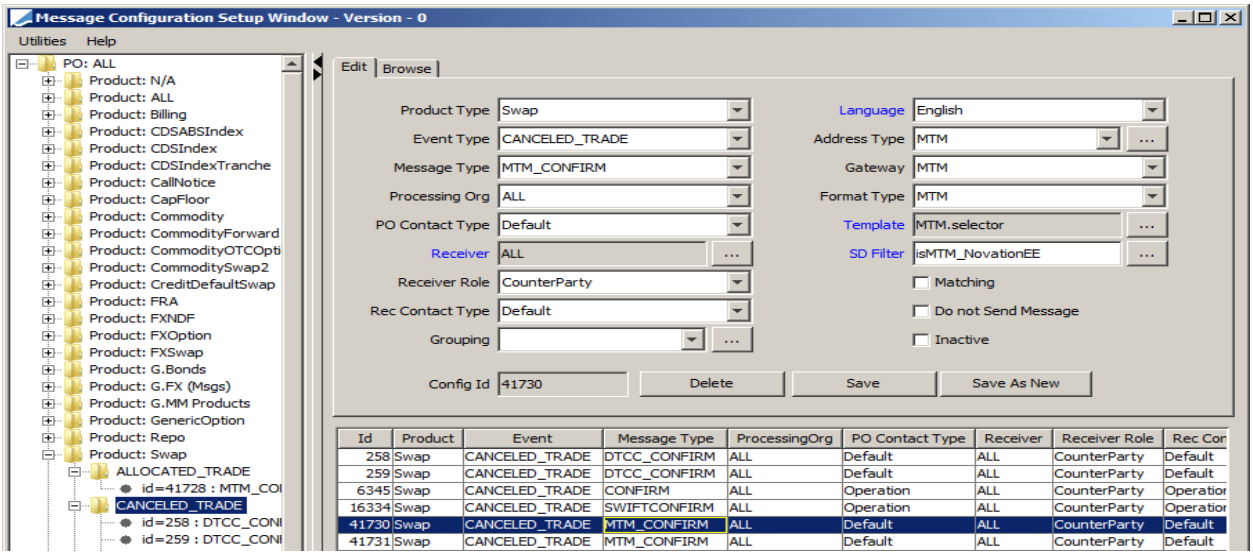
3.2 Message Setup

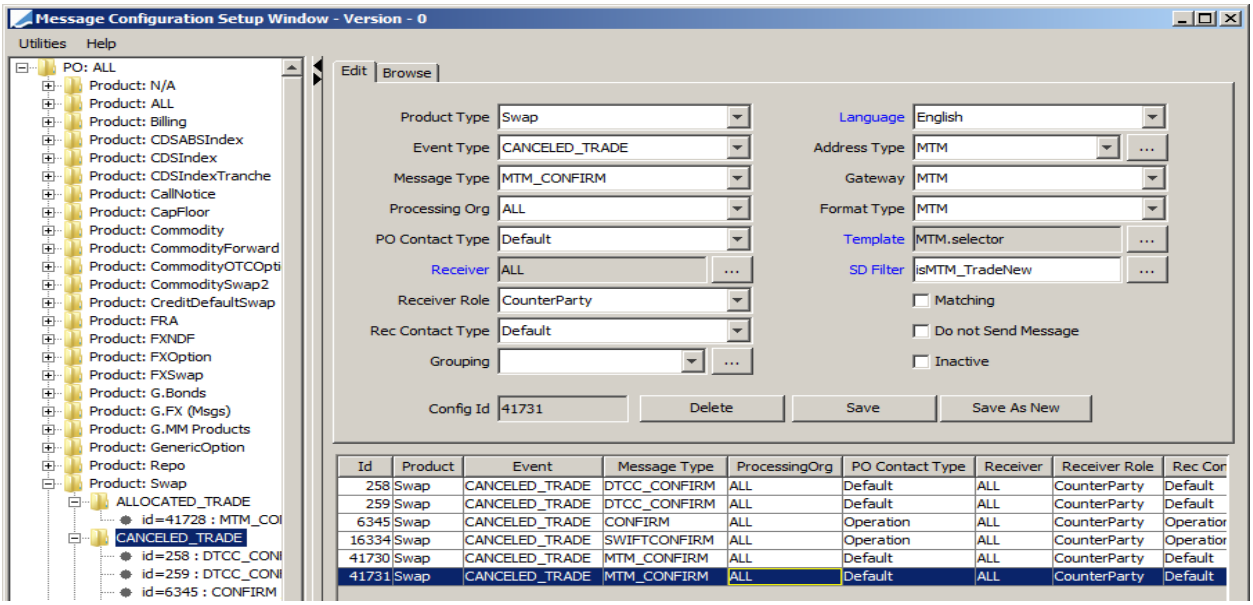
Message setup is required for each lifecycle of every product that MTM supports as shown below. Message setup will be added to the database by running the Execute SQL with the category “MTM Message Setup”.



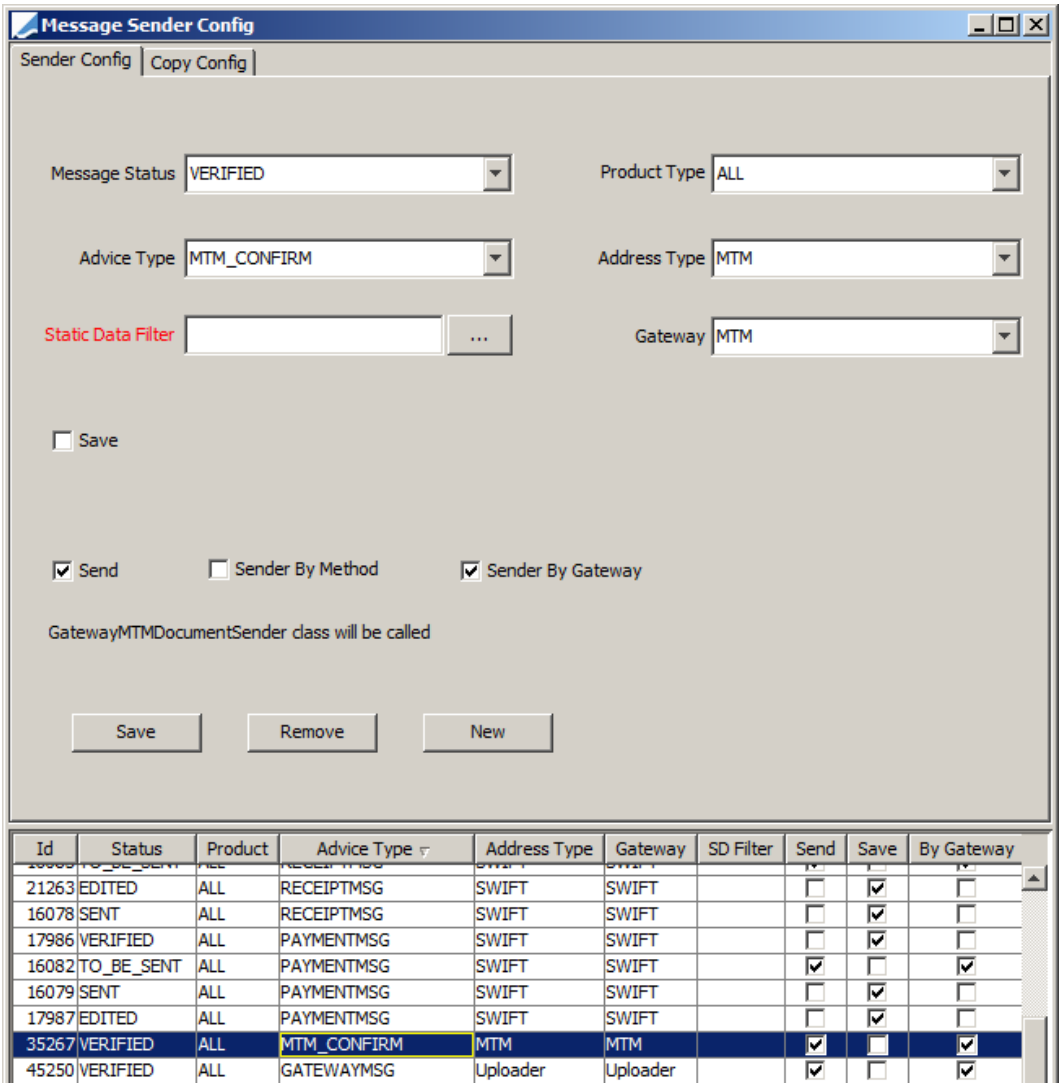

Message Configuration Setup Window - Version - 0
Utilities
Help
Product: ALL
Product: Billing
Product: CDSABSIndex
Product: CDSIndex
Product: CDSIndexTranche
Product: CallNotice
Product: CapFloor
Product: Commodity
Product: CommodityForward
Product: CommodityOTCOpti
Product: CommoditySwap2
Product: CreditDefaultSwap
Product: FRA
Product: FXNDF
Product: FXOption
Product: FXSwap
Product: G.Bonds
Product: G.FX (Msgs)
Product: G.MM Products
Product: GenericOption
Product: Repo
Product: Swap
ALLOCATED_TRADE
id=41728 : MTM_COI
CANCELED_TRADE
id=258 : DTCC_CONI
id=259 : DTCC_CONI
id=6345 : CONFIRM
Edit
Browse
Product Type
Swap
Event Type
VERIFIED_TRADE
Message Type
MTM_CONFIRM
Processing Org
ALL
PO Contact Type
Default
Receiver
ALL
Receiver Role
CounterParty
Rec Contact Type
Default
Grouping
Config Id
34875
Delete
Save
Save As New
Language
English
Address Type
MTM
Gateway
MTM
Format Type
MTM
Template
MTM.selector
SD Filter
isMTM_NovationEE
Matching
Do not Send Message
Inactive
Id
Product
Event
Message Type
ProcessingOrg
PO Contact Type
Receiver
Receiver Role
Rec Cont
256 Swap VERIFIED_TRADE DTCC_CONFIRM ALL Default ALL CounterParty Default
257 Swap VERIFIED_TRADE DTCC_CONFIRM ALL Default ALL CounterParty Default
34875 Swap VERIFIED_TRADE MTM_CONFIRM ALL Default ALL CounterParty Default
41726 Swap VERIFIED_TRADE MTM_CONFIRM ALL Default ALL CounterParty Default

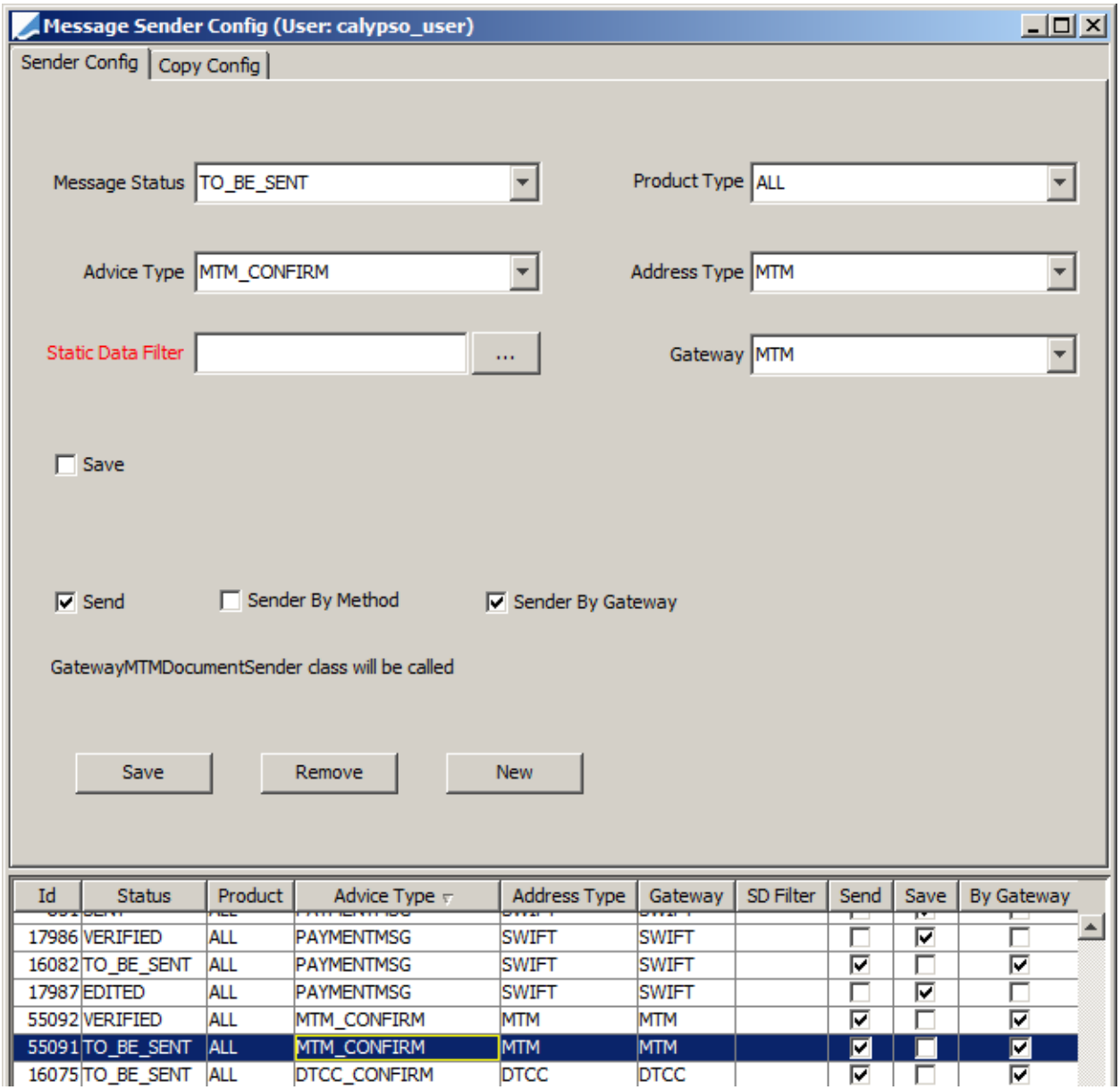

Message Configuration Setup Window - Version - 0
Utilities
Help
PO: ALL
Product: N/A
Product: ALL
Product: Billing
Product: CDSABSIndex
Product: CDSIndex
Product: CDSIndexTranche
Product: CallNotice
Product: CapFloor
Product: Commodity
Product: CommodityForward
Product: CommodityOTCOption2
Product: CommoditySwap2
Product: CreditDefaultSwap
Product: FRA
Product: FXNDF
Product: FXOption
Product: FXSwap
Product: G.Bonds
Product: G.FX (Msgs)
Product: G.MM Products
Product: GenericOption
Product: Repo
Product: Swap
ALLOCATED_TRADE
Edit
Browse
Product Type
Swap
Event Type
ALLOCATED_TRADE
Message Type
MTM_CONFIRM
Processing Org
ALL
PO Contact Type
Default
Receiver
ALL
Receiver Role
CounterParty
Rec Contact Type
Default
Grouping
Config Id
41728
Delete
Save
Save As New
Language
English
Address Type
MTM
Gateway
MTM
Format Type
MTM
Template
MTM.selector
SD Filter
isMTM
Matching
Do not Send Message
Inactive
Id
Product
Event
Message Type
ProcessingOrg
PO Contact Type
Receiver
Receiver Role
Rec Cor
41728 Swap ALLOCATED_TRADE MTM_CONFIRM ALL Default ALL CounterParty Default


Message Configuration Setup Window - Version - 0
Utilities
Help
PO: ALL
Product: N/A
Product: ALL
Product: Billing
Product: CDSABSIndex
Product: CDSIndex
Product: CDSIndexTranche
Product: CallNotice
Product: CapFloor
Product: Commodity
Product: CommodityForward
Product: CommodityOTCOpti
Product: CommoditySwap2
Product: CreditDefaultSwap
Product: FRA
Product: FXNDF
Product: FXOption
Product: FXSwap
Product: G.Bonds
Product: G.FX (Msgs)
Product: G.MM Products
Product: GenericOption
Product: Repo
Product: Swap
ALLOCATED_TRADE
id=41728 : MTM_COI
CANCELED_TRADE
id=258 : DTCC_CONI
id=259 : DTCC_CONI
Edit
Browse
Product Type
Swap
Event Type
CANCELED_TRADE
Message Type
MTM_CONFIRM
Processing Org
ALL
PO Contact Type
Default
Receiver
ALL
Receiver Role
CounterParty
Rec Contact Type
Default
Grouping
Language
English
Address Type
MTM
Gateway
MTM
Format Type
MTM
Template
MTM.selector
SD Filter
isMTM_NovationEE
Matching
Do not Send Message
Inactive
Config Id
41730
Delete
Save
Save As New
Id
Product
Event
Message Type
ProcessingOrg
PO Contact Type
Receiver
Receiver Role
Rec Cor
258 Swap CANCELED_TRADE DTCC_CONFIRM ALL Default ALL CounterParty Default
259 Swap CANCELED_TRADE DTCC_CONFIRM ALL Default ALL CounterParty Default
6345 Swap CANCELED_TRADE CONFIRM ALL Operation ALL CounterParty Operatio
16334 Swap CANCELED_TRADE SWIFTCONFIRM ALL Operation ALL CounterParty Operatio
41730 Swap CANCELED_TRADE MTM_CONFIRM ALL Default ALL CounterParty Default
41731 Swap CANCELED_TRADE MTM_CONFIRM ALL Default ALL CounterParty Default


Message Configuration Setup Window - Version - 0
Utilities
Help
PO: ALL
Product: N/A
Product: ALL
Product: Billing
Product: CDSABSIndex
Product: CDSIndex
Product: CDSIndexTranche
Product: CallNotice
Product: CapFloor
Product: Commodity
Product: CommodityForward
Product: CommodityOTCOpti
Product: CommoditySwap2
Product: CreditDefaultSwap
Product: FRA
Product: FXNDF
Product: FXOption
Product: FXSwap
Product: G.Bonds
Product: G.FX (Msgs)
Product: G.MM Products
Product: GenericOption
Product: Repo
Product: Swap
ALLOCATED_TRADE
id=41728 : MTM_COI
CANCELED_TRADE
id=258 : DTCC_CONI
id=259 : DTCC_CONI
id=6345 : CONFIRM
Edit
Browse
Product Type
Swap
Event Type
CANCELED_TRADE
Message Type
MTM_CONFIRM
Processing Org
ALL
PO Contact Type
Default
Receiver
ALL
Receiver Role
CounterParty
Rec Contact Type
Default
Grouping
Language
English
Address Type
MTM
Gateway
MTM
Format Type
MTM
Template
MTM.selector
SD Filter
isMTM_TradeNew
Matching
Do not Send Message
Inactive
Config Id
41731
Delete
Save
Save As New
Id
Product
Event
Message Type
ProcessingOrg
PO Contact Type
Receiver
Receiver Role
Rec Cor
258 Swap CANCELED_TRADE DTCC_CONFIRM ALL Default ALL CounterParty Default
259 Swap CANCELED_TRADE DTCC_CONFIRM ALL Default ALL CounterParty Default
6345 Swap CANCELED_TRADE CONFIRM ALL Operation ALL CounterParty Operatio
16334 Swap CANCELED_TRADE SWIFTCONFIRM ALL Operation ALL CounterParty Operatio
41730 Swap CANCELED_TRADE MTM_CONFIRM ALL Default ALL CounterParty Default
41731 Swap CANCELED_TRADE MTM_CONFIRM ALL Default ALL CounterParty Default

3.3 Message Sender Config



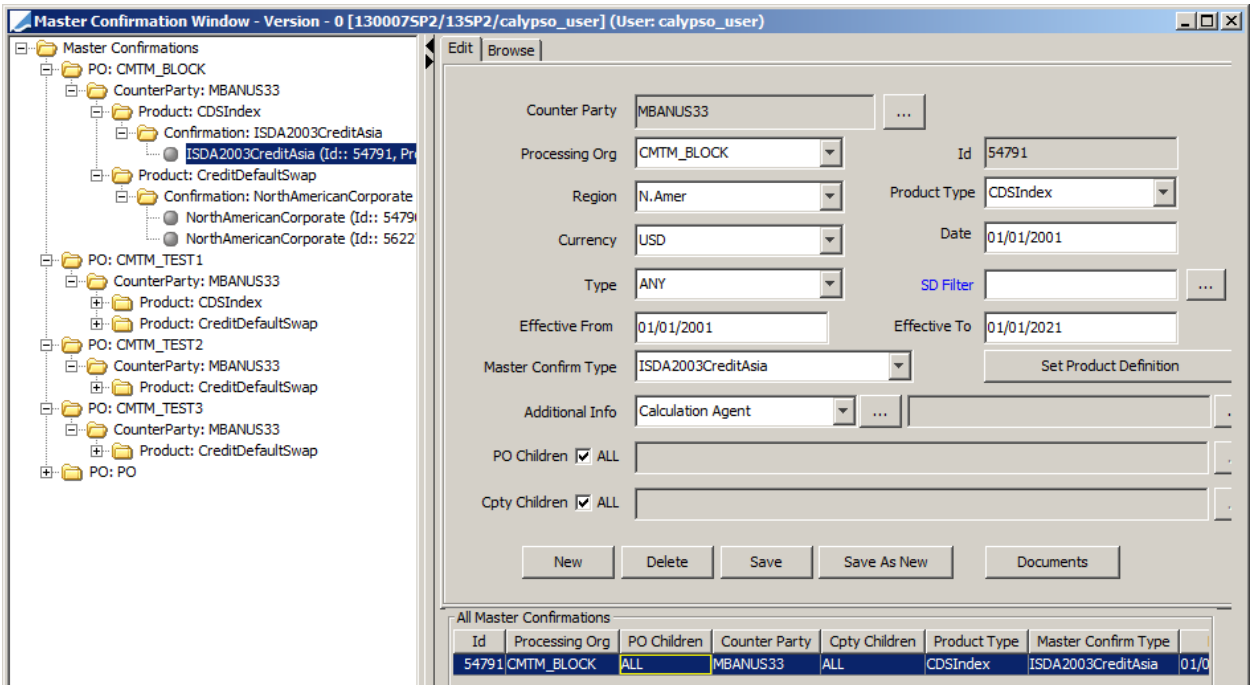


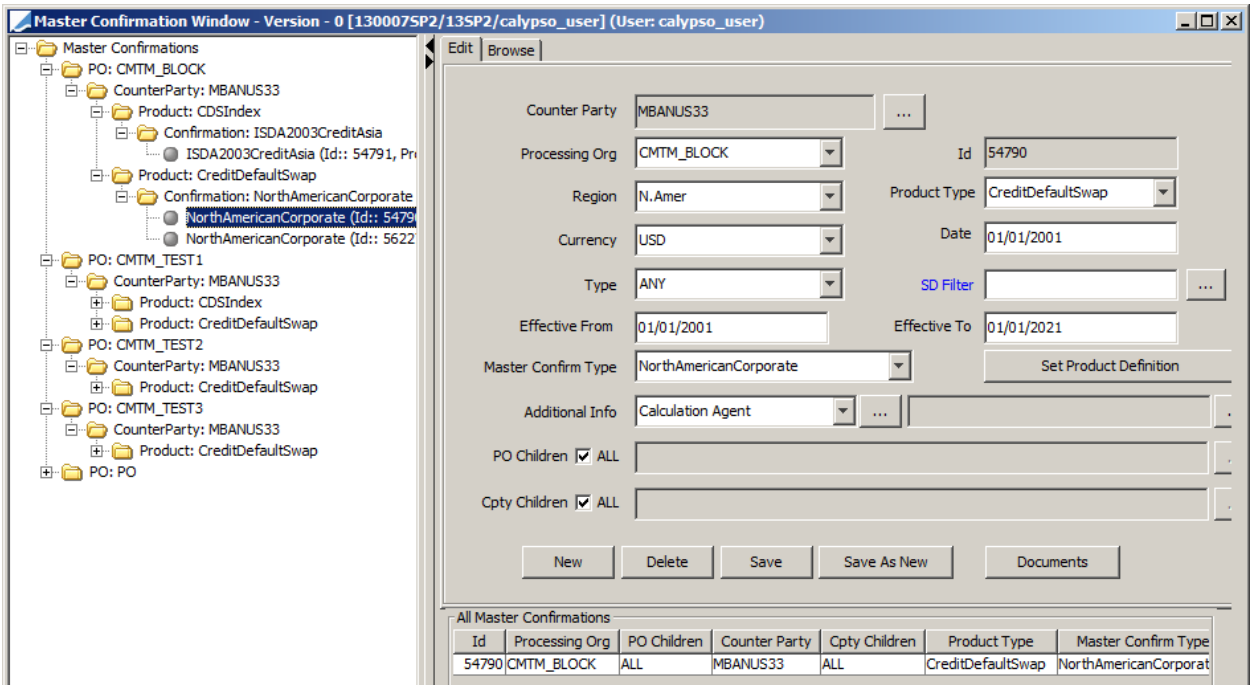
3.4 Master Confirmation Config & Legal Agreements

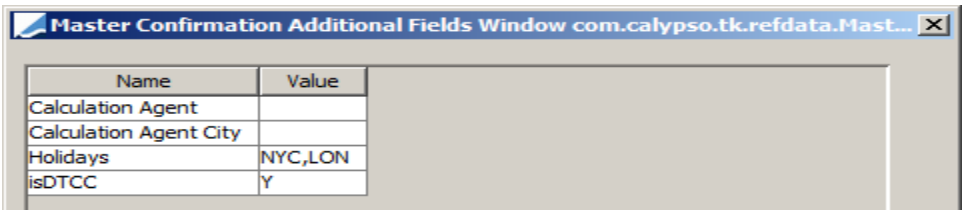
Master confirmation and Legal Agreements are required between the Processing Org and the Counterparty legal entities.

Master confirmations are required by Credit products, whereas Legal Agreements are required for both IRS and CDS products.

If MBANUS33 is the Counterparty and CMTM_BLOCK is the Processing Org the Confirmation should be as follows.


Master Confirmation Window - Version - 0 [130007SP2/13SP2/calypso_user] (User: calypso_user)
Master Confirmations
PO: CMTM_BLOCK
CounterParty: MBANUS33
Product: CDSIndex
Confirmation: ISDA2003CreditAsia
ISDA2003CreditAsia (Id:: 54791, Pr
Product: CreditDefaultSwap
Confirmation: NorthAmericanCorporate
NorthAmericanCorporate (Id:: 5479
NorthAmericanCorporate (Id:: 5622
PO: CMTM_TEST1
CounterParty: MBANUS33
Product: CDSIndex
Product: CreditDefaultSwap
PO: CMTM_TEST2
CounterParty: MBANUS33
Product: CreditDefaultSwap
PO: CMTM_TEST3
CounterParty: MBANUS33
Product: CreditDefaultSwap
PO: PO
Edit
Browse
Counter Party
MBANUS33
Processing Org
CMTM_BLOCK
Id
54791
Region
N.Amer
Product Type
CDSIndex
Currency
USD
Date
01/01/2001
Type
ANY
SD Filter
Effective From
01/01/2001
Effective To
01/01/2021
Master Confirm Type
ISDA2003CreditAsia
Set Product Definition
Additional Info
Calculation Agent
PO Children
ALL
Cpty Children
ALL
New
Delete
Save
Save As New
Documents
All Master Confirmations
Id
Processing Org
PO Children
Counter Party
Cpty Children
Product Type
Master Confirm Type
54791
CMTM_BLOCK
ALL
MBANUS33
ALL
CDSIndex
ISDA2003CreditAsia


Master Confirmation Window - Version - 0 [130007SP2/13SP2/calypso_user] (User: calypso_user)
Master Confirmations
PO: CMTM_BLOCK
CounterParty: MBANUS33
Product: CDSIndex
Confirmation: ISDA2003CreditAsia
ISDA2003CreditAsia (Id:: 54791, Pr
Product: CreditDefaultSwap
Confirmation: NorthAmericanCorporate
NorthAmericanCorporate (Id:: 5479
NorthAmericanCorporate (Id:: 5622
PO: CMTM_TEST1
CounterParty: MBANUS33
Product: CDSIndex
Product: CreditDefaultSwap
PO: CMTM_TEST2
CounterParty: MBANUS33
Product: CreditDefaultSwap
PO: CMTM_TEST3
CounterParty: MBANUS33
Product: CreditDefaultSwap
PO: PO
Edit
Browse
Counter Party
MBANUS33
Processing Org
CMTM_BLOCK
Id
54790
Region
N.Amer
Product Type
CreditDefaultSwap
Currency
USD
Date
01/01/2001
Type
ANY
SD Filter
Effective From
01/01/2001
Effective To
01/01/2021
Master Confirm Type
NorthAmericanCorporate
Set Product Definition
Additional Info
Calculation Agent
PO Children
ALL
Cpty Children
ALL
New
Delete
Save
Save As New
Documents
All Master Confirmations
Id
Processing Org
PO Children
Counter Party
Cpty Children
Product Type
Master Confirm Type
54790
CMTM_BLOCK
ALL
MBANUS33
ALL
CreditDefaultSwap
NorthAmericanCorporat


Master Confirmation Additional Fields Window com.calypso.tk.refdata.Mast...
Name
Value
Calculation Agent
Calculation Agent City
Holidays
NYC,LON
isDTCC
Y

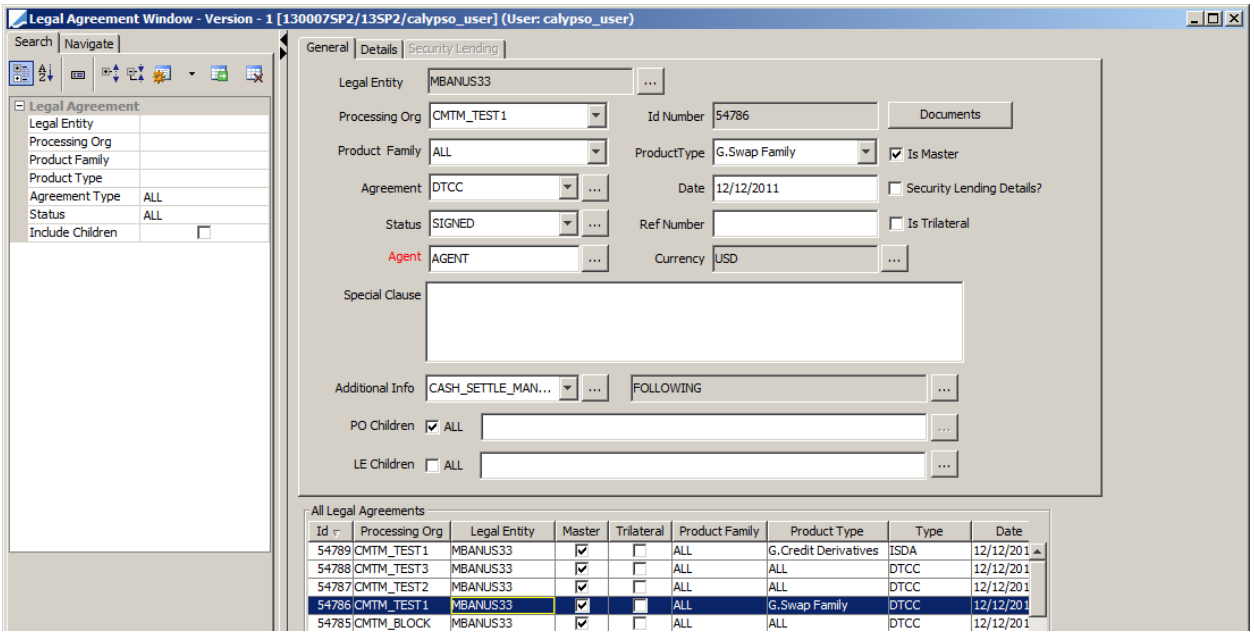
Please note that the MasterConfirmation Type should match as shown in the screen shots above.

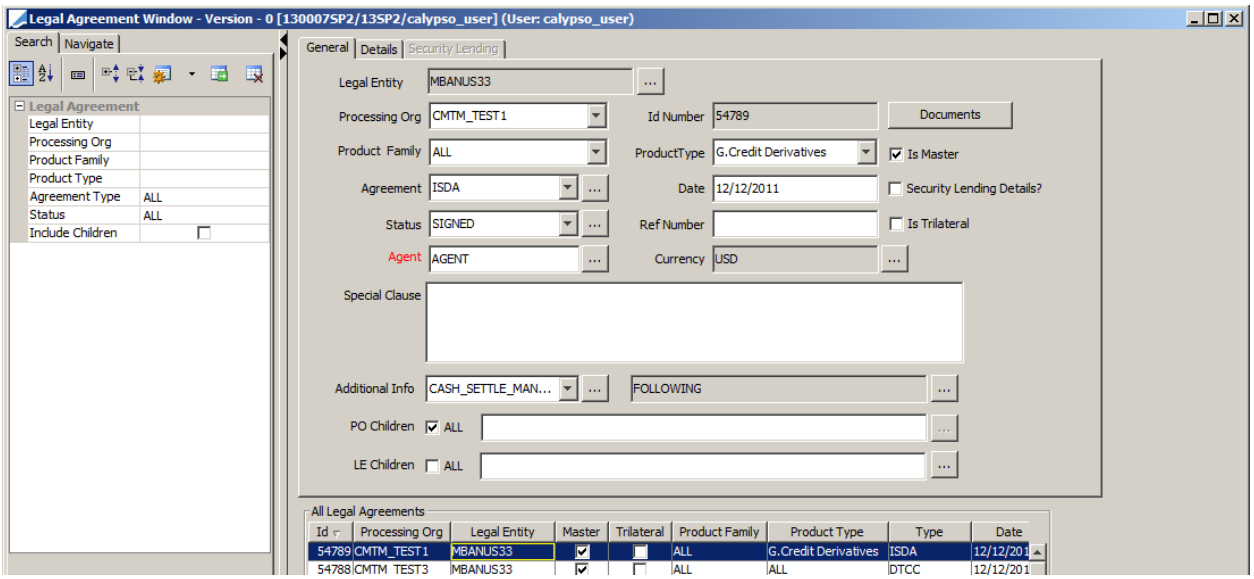
For CDSIndex the master confirmation type should be “ISDA2003CreditAsia”.

For CDS the master confirmation type should be “NorthAmericanCorporate”.

Also, it is important to set the Holidays and the isDTCC flag for each of these confirmation types.

Please note the agreement in the screen shots provided below for Legal Agreements (note that Data Uploader samples can be used to upload the legal agreements, however the master confirmation types have to be created manually as they are not yet supported by the Data Uploader).





3.5 Processes

3.5.1 Property File

You need to setup a property file for the MTM Import Message engine: “calypso_mtm_config.properties”.

This file contains the properties used to identify the input and output queues. The following are the properties in the config file:

```
# Sender Engine (MQ)
mq.dtcc.hostname=10.2.1.252
mq.dtcc.port=1414
mq.dtcc.appiddatauser=059T0101
mq.dtcc.appiddatapass=TIJA6D1J
mq.dtcc.queuemanager=U059T0101
mq.dtcc.channel=SYSTEM.ADMIN.SVRCONN
output.queue.name=DER.PART_DTCC.TRADE.INP01
DER.PART_DTCC.TRADE.INP01.queue.setContext=true

# Import Message Engine (JMS)
jms.url=file://localhost/c:/tools/IBM/binding/
jms.modetypeclass=com.sun.jndi.fscontext.RefFSContextFactory
jms.queue.connectionFactory=QueueConnectionFactory
input.queue.name=QUEUE.MTM.CALYPSO
QUEUE.MTM.CALYPSO.queue.ackType=auto
QUEUE.MTM.CALYPSO.queue.persist=true
QUEUE.MTM.CALYPSO.queue.transacted=false

DEBUG_XML=true
```

3.5.2 Engines

The following engines need to be running:

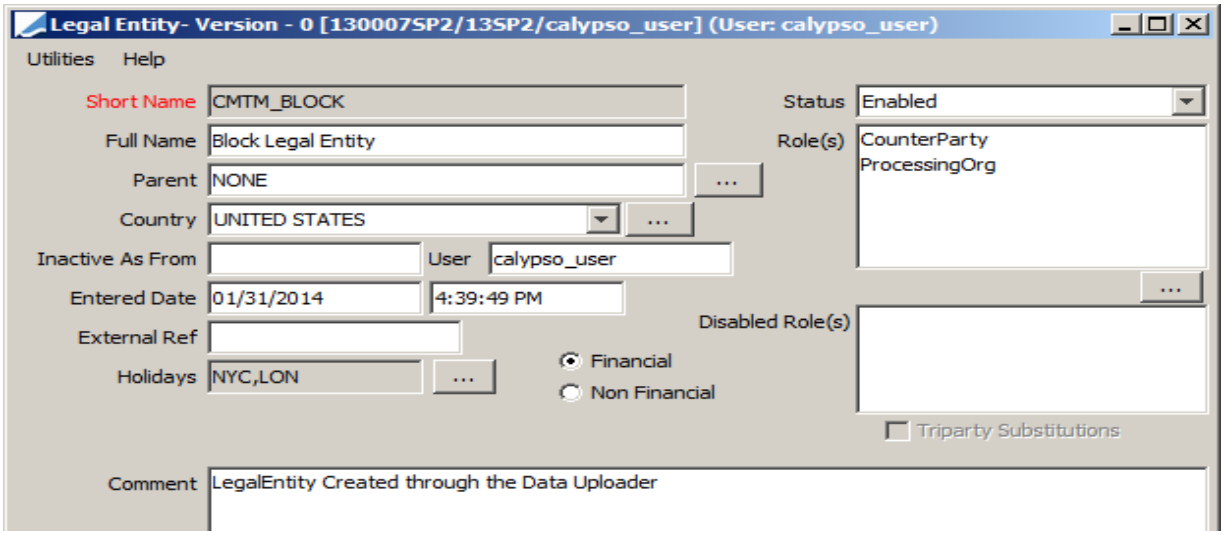
- Message engine, for outgoing message generation
- Sender engine, to send message to MQ
- Import message engines, to import message from MQ

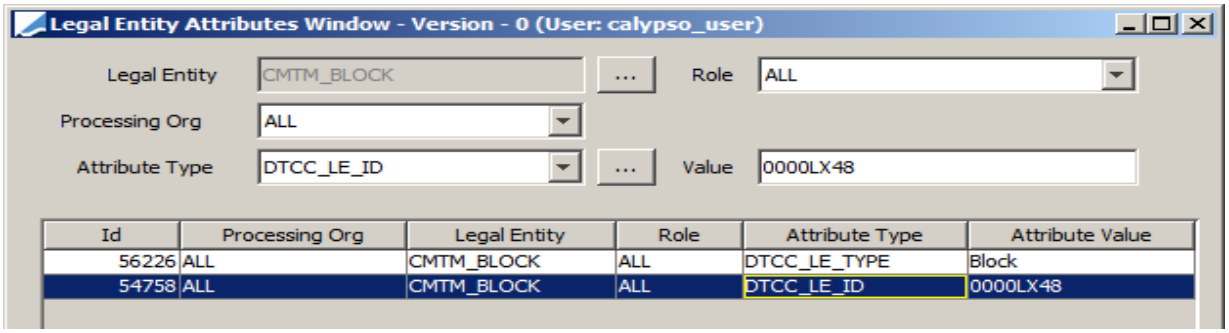
The setup of the engines is done using the Engine Manager in Web Admin.

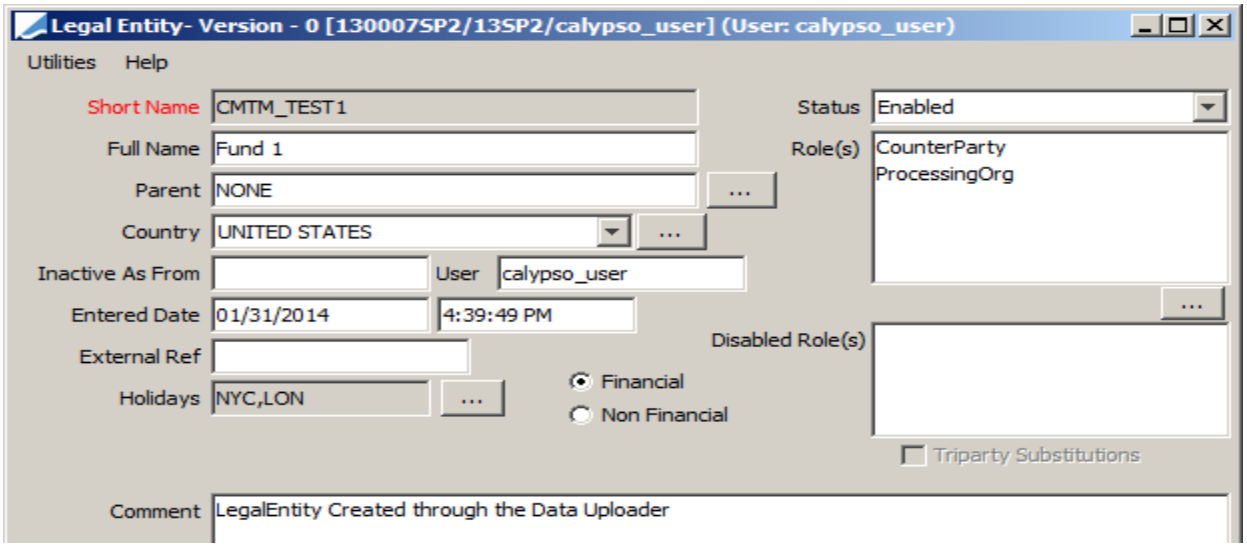
They are started as part of the Engine server using "`<calypso home>/startEngineserver.bat`" on Windows platforms or "`<calypso home>/startEngineerver.sh`" on *nix platforms.

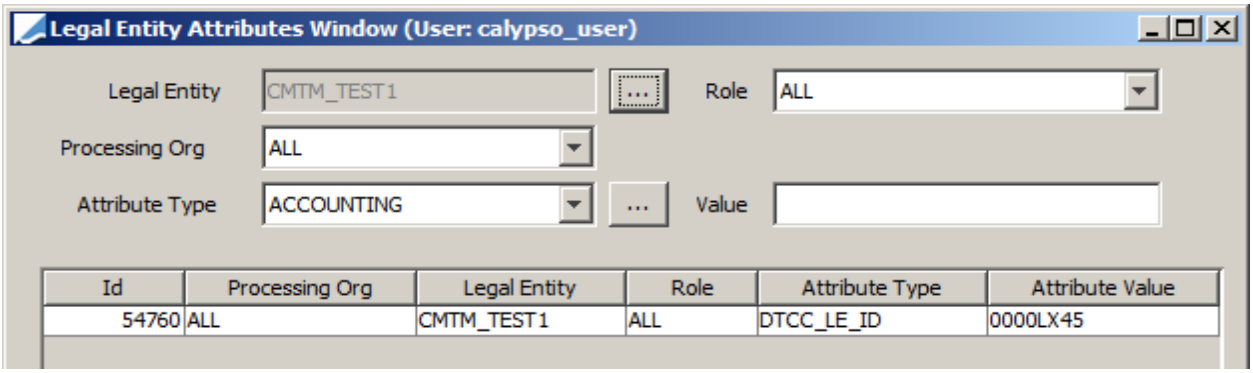
3.6 Book and Legal Entity Mapping

Below legal entity setup is required to generate the outgoing message. Legal entity should have LE attribute DTCC_LE_ID and DTCC_LE_TYPE as shown in the screenshots.

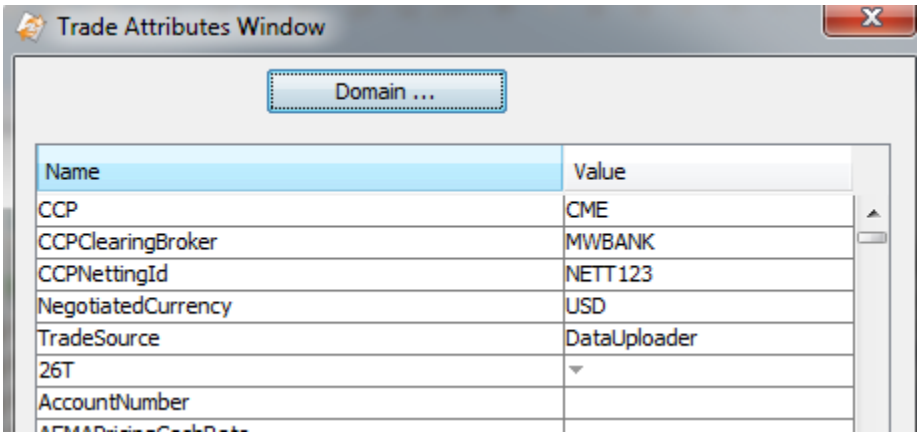






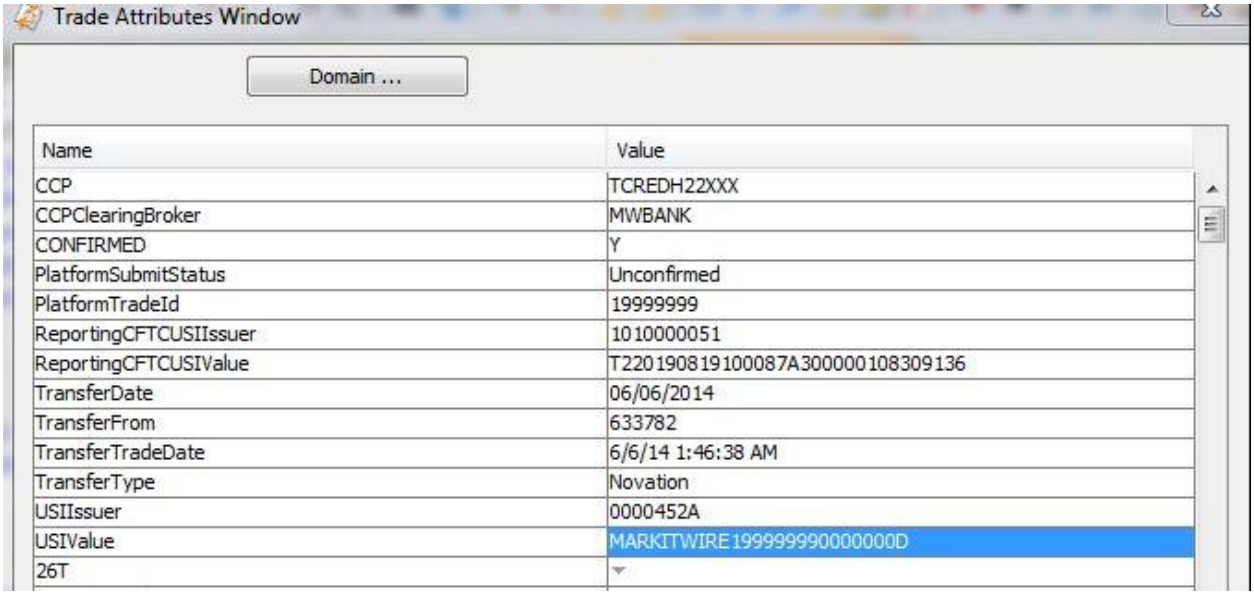


The Trade keywords will be populated as follows:



After clearing:

```
- <USI>
    <USIIssuer issuerIdScheme="http://www.fpml.org/coding-scheme/cftc/issuer-identifier">1010000051</USIIssuer>
    <USITradeId tradeIdScheme="http://www.fpml.org/coding-scheme/cftc/issuer-identifier">T220190819100087A300000108309136</USITradeId>
  </USI>
- <USI>
    <USIIssuer issuerIdScheme="http://www.fpml.org/coding-scheme/external/unique-transaction-identifier">0000452A</USIIssuer>
    <USITradeId tradeIdScheme="http://www.fpml.org/coding-scheme/external/unique-transaction-identifier">MARKITWIRE199999990000000D</USITradeId>
  </USI>
```


Trade Attributes Window
Domain ...
Name
Value
CCP
TCREDH22XXX
CCPClearingBroker
MWBANK
CONFIRMED
Y
PlatformSubmitStatus
Unconfirmed
PlatformTradeId
19999999
ReportingCFTCUSIIssuer
1010000051
ReportingCFTCUSIValue
T220190819100087A300000108309136
TransferDate
06/06/2014
TransferFrom
633782
TransferTradeDate
6/6/14 1:46:38 AM
TransferType
Novation
USIIssuer
0000452A
USIValue
MARKITWIRE199999990000000D
26T

MQ Setup

The MTM module reads messages from a JMS Provider (like IBM MQ) via the MTM Import Message Engine and translates them into CalypsoUploadDocument format. The Following are the steps to be followed to support real-time messaging via IBM MQ or any other JMS Provider like Apache Active MQ.

4.1 Prerequisites

- Installation - IBM MQ / Active MQ (or another JMS Provider)
- Queues / Queue manager should already be created by customer.
- Calypso with MTM configured.

4.2 MTM Configuration for JMS Support

This section covers installation and configuration of JMS Support in MTM.

4.2.1 Installation

Ensure that the appropriate jars are set in the Classpath. Each JMS provider will have different jars required. We are providing an example for Web sphere MQ and Active MQ below.

The MTM jar should already be in classpath. Details are not covered since that is discussed in the MTM setup guide.

For IBM WebShere MQ

com.ibm.mq.commonservices.jar

com.ibm.mq.headers.jar

com.ibm.mq.jar

com.ibm.mq.jmqi.jar

com.ibm.mq.jms.Nojndi.jar

com.ibm.mq.pcf.jar

com.ibm.mqjms.jar

connector.jar

fscontext.jar

jms.jar

jndi.jar

providerutil.jar

com.ibm.mq.jmqi.jar

dhbcore.jar

For Active MQ

activemq-core-5.5.0.jar (or whatever version the customer is using)

kahadb-5.5.0.jar

4.2.2 Calypso Configuration

The following configuration needs to be done in Calypso.

- MTM Import Message Engine for incoming Messages (Receiver Queue)
- Acknowledgements: MTM Import Message Engine to send ACK / NACK messages to Sender Queue.

The Import Message Engine listens to JMS Queue (Receiver Queue) based on the Message Engine configuration. Any SOAP (having fpml embedded) message placed on the queue is read by the Engine and then translated to CalypsoUploadDocument and finally Uploaded into Calypso. Acknowledgements generated are sent back to the Sender Queue using the same Engine.

Configuring MTM Import Message Engine

To configure the Import Message Engine, please change the following in the file "calypso_uploader_config.properties".

Connection Details for the JMS

Please uncomment the appropriate block for Active MQ or Websphere MQ. The sample below shows the connection for Active MQ, whereas Websphere MQ is commented. So, please keep one of the blocks below.

```
# Connection Details for JMS
# Start Connection Details for ActiveMQ
jms.url=tcp://localhost:61616
jms.modetypeclass=org.apache.activemq.jndi.ActiveMQInitialContextFactory
jms.queue.connectionFactory=ConnectionFactory
# End Connection Details for ActiveMQ

# Start Connection Details for IBM Websphere MQ
# Please see documentation (appendix) on how to generate
# this bindings files
#jms.url=file://localhost/usr/local/calypso/resources/MTM/binding
#jms.modetypeclass=com.sun.jndi.fscontext.RefFSContextFactory
#jms.queue.connectionFactory=QueueConnectionFactory
# End Connection Details for IBM Websphere MQ
```

Input Queue Name. Please keep one of the blocks below based on the JMS provider.

```
# Input Queue Name (Active MQ)
# Note that for ActiveMQ the queue name has to prefix with
# 'dynamicQueues/', so the following is needed for Active MQ
input.queue.name=dynamicQueues/inputqueue
dynamicQueues/inputqueue.queue.setContext=true

# Input Queue Name (Websphere MQ)
input.queue.name=inputqueue
inputqueue.queue.setContext=true
```

Output Queue Name: Please keep one of the blocks below based on the JMS provider.

```
# Output Queue Name (Active MQ)
# Note that for ActiveMQ the queue name has to prefix with
# 'dynamicQueues/', so the following is needed for Active MQ
output.queue.name=dynamicQueues/outputqueue
dynamicQueues/outputqueue.queue.ackType=auto
dynamicQueues/outputqueue.queue.persist=true
dynamicQueues/outputqueue.queue.transacted=false

# Output Queue Name (Websphere MQ)
output.queue.name=outputqueue
outputqueue.queue.ackType=auto
outputqueue.queue.persist=true
outputqueue.queue.transacted=false
```

The below step is to be done for connecting to a remote queue manager having the user configuration done as mentioned previously.

MQ username

```
# Set the following property having the username of the MQ user setup on remote queue manager.
jms.queue.connectionUserName=mq_user
```

MQ user password

```
# Set the following property having the password of the MQ user setup on remote queue manager.

jms.queue.connectionPassword=<password>
```

App Id Data user and password

```
# Set the following properties having the username and password given by DTCC which will be set as
ApplIdenttityData field in the message header.

       <OutputQueueName>.queue.appiddatauser=<username>

<OutputQueueName>.queue.appiddatapass=<password>
```

4.3 Appendix: Websphere MQ Setup

The following steps are to be performed only when using IBM MQ Series.

IBM WebSphere MQ Installation

- Install IBM MQ Series
- Create Queues

  Launch MQ Explorer and create the following.

  Queue Manager

  Input Queue
- Output Queue
- Create Binding File for JMS Connectivity

Binding File for JMS Connectivity

Websphere MQ requires a bindings file to be created for JMS usage. This file has configuration which allows Java to connect to Websphere as a MQ. The process is as follows.

- Generate Bindings file as explained below, which is used as PROVIDER_URL to connect to the Middleware.
- Link the Bindings file in the MTM Import Message Engine for JMS Provider setup. E.g. see the line below to be changes in the MTM Import Message Engine configuration file.

```
#jms.url=file://localhost/usr/local/calypso/resources/MTM/binding
```

Generating Bindings

- Backup your **JMSAdmin.config** file if already present in environment. Rename the file **src/main/resources/MTMJMSAdmin.config** to **JMSAdmin.config**

  This is required since IBM tools to generate bindings only read JMSAdmin.config. So, please be sure to backup any earlier JMSAdmin.config file so that it can be restored.

- Edit the file **src/main/resources/JMSAdmin.config** and change the following properties.

  PROVIDER_URL=file:/usr/local/calypso/resources/MTM/binding (this indicates the path to the bindings file and the name of the file will be ".bindings";)

  SECURITY_AUTHENTICATION=none (authentication model)

- Edit the file **src/main/bin/MTMMQSeries.bat** and change the highlighted text (which are the Queue Manager Name, Input queue name and Output queue name).

```
@echo off


rem -------------------------------------------------------------
rem  this batch file is used to generate the bindings that are used by the connection factory.
rem  replace InputQueueName with the name of the input queue
rem  replace OutputQueueName with the name of the output queue
rem  replace QM_Name with the name of the Queue Manager
rem  change the classpath to the appropriate path
rem -------------------------------------------------------------


echo + Creating script for object creation within JMSAdmin
echo del qcf(QueueConnectionFactory) > MTMmqsetup.scp
echo del q(CalypsoToClient) >> MTMmqsetup.scp
echo del q(ClientToCalypso) >> MTMmqsetup.scp
echo def qcf(QueueConnectionFactory) TRAN(CLIENT) HOST(127.0.0.1) PORT(1414) qmgr(QM.JM)
CHANNEL(SYSTEM.ADMIN.SVRCONN)>> MTMmqsetup.scp
echo def q(CalypsoToClient) qu(CalypsoToClient) qmgr(QM_CALYPSO) tc(JMS)>> MTMmqsetup.scp
echo def q(ClientToCalypso) qu(ClientToCalypso) qmgr(QM_CALYPSO) tc(JMS)>> MTMmqsetup.scp
echo end >> MTMmqsetup.scp


set CLASSPATH="D:\\calypso\\software\\rel12sp6\\mq-
jars\\com.ibm.mq.jar;D:\\calypso\\software\\rel12sp6\\mq-jars\\com.ibm.mqjms.jar"
echo + CLASSPATH=%CLASSPATH%
echo + Calling JMSAdmin in batch mode to create objects
java -DMQJMS_LOG_DIR="%MQ_JAVA_DATA_PATH%"\log -DMQJMS_TRACE_DIR="%MQ_JAVA_DATA_PATH%"\errors -
DMQJMS_INSTALL_PATH="%MQ_JAVA_INSTALL_PATH%" com.ibm.mq.jms.admin.JMSAdmin < MTMmqsetup.scp


echo + Administration done; tidying up files
del MTMmqsetup.scp
```

where,

- QM_CALYPSO is Queue Manager
- CalypsoToClient is Input Queue
- ClientToCalypso is Output Queue

We are giving sample for Windows, please create a similar script for Unix and change to Unix format (% should change to $ for variable names, change CLASSPATH to use Unix syntax)

Sample script for Linux environment:

```
#!/bin/sh

#This script is used to create a MQ binding file. Please make sure that the directory specified in
JMSAdmin.config PROVIDER_URL exists in the disk. The usages of the script is as below:

# sudo -u <user> ./createMQbinding.sh QM TEST.SENDQ TEST.RECVQ  <hostname> <port> <client_name>
<client_env_name>


QUEUE_MANAGER=$1

INPUT_QUEUE=$2

OUTPUT_QUEUE=$3

MQ_HOST=$4

MQ_PORT=$5

CLIENT=$6

ENV=$7


USER_NAME=$USER


#USER_NAME=who

BINARY_PATH="/usr/local/calypso/clients/${CLIENT}/${ENV}/jars"


SCP_FILE="${ENV}_`date +'%Y%m%d%H%M%S'`.scp"


echo "del q($INPUT_QUEUE)" >> $SCP_FILE

echo "del q($OUTPUT_QUEUE)" >> $SCP_FILE

echo "def qcf(QueueConnectionFactory) TRAN(CLIENT) HOST($MQ_HOST) PORT($MQ_PORT)
qmgr($QUEUE_MANAGER) CHANNEL(SYSTEM.ADMIN.SVRCONN)" >> $SCP_FILE

echo "def q($OUTPUT_QUEUE) qu($OUTPUT_QUEUE) qmgr($QUEUE_MANAGER) tc(JMS)" >> $SCP_FILE
```

```
echo "def q($INPUT_QUEUE) qu($INPUT_QUEUE) qmgr($QUEUE_MANAGER) tc(JMS)" >> $SCP_FILE
echo "end" >> $SCP_FILE


CLASSPATH="${BINARY_PATH}/com.ibm.mq.jar:${BINARY_PATH}/com.ibm.mqjms.jar"
echo "+ CLASSPATH=$CLASSPATH"
echo "+ Calling JMSAdmin in batch mode to create objects"
java -cp $CLASSPATH -DMQJMS_LOG_DIR="/home/$USER_NAME" -DMQJMS_TRACE_DIR="/home/$USER_NAME" -
DMQJMS_INSTALL_PATH="/home/$USER_NAME" com.ibm.mq.jms.admin.JMSAdmin < $SCP_FILE


echo "+ Administration done; tidying up files"
# rm $SCP_FILE
```

- This batch file or the shell scipt calls the IBM JMSAdmin tool that looks for the property file JMSAdmin.config in the current directory. So, please run the script in the same directory where the JMSAdmin.config is present. This will create the binding file.
- After generating the bindings, you can set the MTM Import Message Engine configuration appropriately.
- Run the batch file MTMMQSeries.bat as shown below.

```
C:\calypso\software\MTM\src\main\resources>C:\calypso\software\MTM\src\main\bin\MTMMQSeries.bat
+ Creating script for object creation within JMSAdmin
+ Calling JMSAdmin in batch mode to create objects.
5724-H72, 5655-L82, 5724-L26 (c) Copyright IBM Corp. 2002,2005. All Rights Reserved.
Starting Websphere MQ classes for Java(tm) Message Service Administration
JNDI initialization failed, please check your JNDI settings and service.
For additional information on the cause of this problem run with the -v argument
+ Administration done; tidying up files.
+ Done!
```

Release Notes

5.1 Changes in Version 3.3.0

Component	Case	APL	Issue	Type	Description
Markit TM 3.3.0		APL-10316	MTM-125	Sec	Issue – Security - Upgrade org.xmlunit:xmlunit-core library to version 2.10.0.

5.2 Changes in Version 2.4.0

Component	HD/Case	RPM	Issue	Type	Description
Markit TM 2.4.0		RPM-6307	MTM-97	Enh	Issue – Security - Upgrade xmlunit libraries to version 2.8.2.

5.3 Changes in Version 2.3.0

Component	HD/Case	RPM	Issue	Type	Description
Markit TM 2.3.0		RPM-4313	MTM-93	Enh	Issue – Technical issue - Changes for using “internal” Data Uploader module.

5.4 Changes in Version 2.2.3

Component	HD/Case	Issue	Type	Description
Markit TM 2.2.3		MTM-89	Issue	Issue – Duplicate entries in MTMSchemaData.xml for domain xmlFormatType. Fix – Duplicate entries have been removed.

5.5 Changes in Version 2.2.1, 2.2.2

Component	HD/Case	Issue	Type	Description
Markit TM 2.2.2		MTM-76	Issue	Issue – Duplicate entries in MTMSchemaData.xml.