CBOEDirect Release Notes

CBOEDIR_8.4

And Infra 13.2

August 4, 2010

Table of Contents

Overview of CD Server changes in this release

Work requests included in this release

6325 – HDE4, Phase 3 (features 1, 2, and 12 (AUDIT LOGGING))

6301 – Buffer support for current market

5959 – User Input monitor (UIM)

6443 – CBSX Contingency Cross Enhancements Phase 3

6448 – CPS Phase II – Market Order splitting functionality

6411 – Short Sales Marking

6185 – Global FE Deprecation

6154 – Trade Match Data Recovery

WR6325 – HDE4 Phase 3 (features 1 and 2)

Goal

The goal of this project is to add enhancements to the system to allow the Help Desk staff to do their jobs more efficiently.

Summary of changes for server side

Feature #1:

1. Cancel By spreadsheet

Today, cancels of orders can be performed at a BC level and also by spreadsheet. The purpose of this feature is to improve the spreadsheet capability by also including several additional fields to it. Those new fields are:

- *Firm*
- *Correspondent firm*
- *Order ID (ORSID or Firm/Corr/Branch/Seq/Date)*
- *Date*

2. Cancel orders by class for all users

Today, orders can be canceled by product class for a single user (group). With this feature, Help Desk will be able to cancel orders for a single product class or multiple product classes for all users.

3. Cancel orders by product for all users

With this feature, Help Desk will be able to cancel orders in multiple series for all users (e.g. cancel orders for ALL FIRMS for these IBM series).

4. MTS related bug fixes

As part of this feature, all issues introduced by the "Multiple Trade Servers" project have been fixed.

Feature #2:

1. Enhancements and fixes to the existing trade bust/reinstate functionality

Bust function

- Busting of trades involving complex orders is now allowed. Both package-level and leg by leg busts are permitted. Partial busts are permitted.
- Busting of trades involving cross-products is now allowed. Both package level and leg by leg busts can be performed from W_MAIN session side. No partial busts of equity leg are permitted.
- Busting of PAR and OMT trades is now allowed.

Reinstate function

- Orders reinstated at the top of the book will be allowed to HAL first to prevent NBBO violation..
- Complex order reinstate is not allowed.
- Reinstate of MKT orders is not allowed.
- The following contingency orders are allowed to be reinstated:
  - AON
  - STOP - The order will be reinstated in not-triggered state
  - STOP-LOSS - The order will be reinstated in not-triggered state
  - STOP-LIMIT - The order will be reinstated in not-triggered state
  - CLOSE - The order will be reinstated in not-triggered state
  - RESERVE
  - ISB

Note: The reinstate functionality requires a change to the CAS related to the processing of the internal product cache. The cache has the order removed when it is filled but it was not being added back to the cache on the reinstate.

2. New functionality

A new Trade Query screen has been added to the SAGUI that allowes Help Desk users to query the system for trades. The user then can mark the trades that need to be busted and then proceed to bust all the marked trades in a single operation. The results of the bust operation are displayed on a result screen with an indication next to each trade of whether or not the bust succeeded.

The new query will allow the user to query for trades using a combination of the following selection criteria:

- **Time** (e.g. all trades between 11:00:000 and 11:10:000).
- **Time/series** (e.g. all trades for IBM DEC 100 CALLS between 11:00:000 and 11:10:000).
- **Time/Class** (e.g. all IBM trades between 11:00:000 and 11:10:000).
- **Time/Underlying primary exchange** (e.g. all trades for the options classes whose underlying has NASDAQ as their primary exchange between 11:00:000 and 11:10:000).
- **Time/UserID and/or Acronym** (e.g. all trades where user 'XYZ' is on either side of the trade between 11:00:000 and 11:10:000).
  - Busts by UserID can only be for one user at a time.
- **Time/UserID Group** (e.g. all trades where user group 'Citadel' is on either side of the trade between 11:00:000 and 11:10:000).
  - User groups must be defined in the 'group service' (can have sub-groups as occurs today).
  - Busts by group can only be for one group at a time.

WR6301 – Buffer Support For Current Market

Goal

The goal of this project is to allow for the inclusion of the current market data in the MsgCodec buffers to be sent to clients currently receiving the CurrentMarket structs.

Summary of changes for server side

This release forms part of an incremental transition to the MsgCodec that includes CD8.0, CD8.1 and CD8.2 releases. CD8.4 will include the conversion of the existing arrays of CurrentMarketStruct over the CurrentMarket channel to MsgCodec-encoded entities co-mingled with book depth events in the DataBufferBlock over MarketBuffer channels. The codec includes all of the information currently encoded in the CurrentMarketStruct as well as the currently existing book depth messages. It will encode market updates, state changes, EOP events, and manual quotes (see Market Update Codec – Data Format document for more information).

There are a number of processes that currently receive CurrentMarketStructs over the CurrentMarket channel. These processes include:

- XTP
- MDX
- MDCAS
- CAS
- FIXCAS
- CFIX
- CoppBridge
- CoppAdapter
- StockCFNAdapter
- MDBroadcastAdapter

Once the conversion of the arrays of CurrentMarketStruct (over the CurrentMarket channel) to MsgCodec-encoded entities (grouped in the DataBufferBlock over MarketBuffer channels) happens, these processes will have to listen to the MarketBuffer channels for DataBufferBlocks in order to obtain the market updates, state changes, EOP events, and manual quotes information that they need. In order to allow a smother transition the CurrentMarketBridge process will act as a translator. The bridge will listen for MsgCodec-encoded entities (co-mingled with book depth events) in the DataBufferBlock over the MarketBuffer channels and transform them into arrays of CurrentMarketStruct, CurrentMarketStateChangeStruct, ExpectedOpeningPriceStruct and ManualsQuoteDetailInternalStruct. These will then be published to the CurrentMarket and CurrentMarketStateChange channels as they correspond. This will allow the current CurrentMarketStruct consumer processes to continue operating normally without realizing that the Trade Server is disseminating MsgCodecs. The Current Market Bridge is meant to be an interim process that will be removed progressively as the current CurrentMarketStruct consumer processes migrated to DataBufferBlocks consumer processes.

Dependancies

- A XTP version must be rolled out that will accept current market in both the struct and market buffer configuration.
- A MDCAS, CAS, FIXCAS, CFIX version must be rolled out that will accept current market in both the struct and market buffer configuration.

WR5959 – User Input Monitoring (UIM)

Goal

The goal of this project is to make the enablement of the existing UIM feature at user ID level, thus individual firms could be on their own pace to extend the feature for their users.

Summary of changes for server side

The UIM feature is to cancel all quotes and quote like orders for a particular user if the trade server instance has not received a qualifying quote activity from that user in X milliseconds.

The existing implementation of the UIM feature has the following characteristics –

1. It offers all-or-none enablement per session for all users
2. It offers one fixed setting, the X milliseconds mentioned above, per session for all users
3. The actual action of "cancel" is never turned on in production, although the code has been in production for years and it logs alarms whenever an action would have occurred. In other words, the feature is never actually validated in production in terms of possible impact.

This enhancement project is to

1. Give flexibility, at the user ID level, to both the enablement of the feature and the granularity of control over the settings
2. Turn on the action of "cancel" when the conditions are met

To make the enhancement feasible, a new parameter, at user ID level in milliseconds, is introduced: the UIM interval, which would be controlled solely by the help desk via SA-GUI. The parameter is doubled as the enablement flag and the UIM setting: when it is set to be <=0, UIM is disabled for the user, otherwise, the number would be used for the UIM interval checking for the user.

To implement this new parameter, a new property category is added "UserProperty", and a new property group underneath the category is created [userID]+'\u0001'+"UIMInterval". The actual parameter is then a name-value pair simple property under this new property group, and the name would be the sessionName and the value would be the integer number in milliseconds.

It's worth noting that there are a few other attributes that controls the UIM behavior from the existing UIM implementation –

1. The trading property "UIM active", which could only set at the "default" level per session, is the flag that turns on/off the UIM feature for the session
2. The –DneverCancel system property for each trade server instance is the flag that turns on/off the actual action on "cancel". There is an ar command that could override the property manually at runtime.
3. The resource XML file for the UserInputMonitorHome has the following settings –
    a. numCancelThreads – determines the number of threads for the UIM executor pool
    b. secondsAfterOpen – determines when UIM would actually monitoring in seconds after a product open

   c. secondsBeforeClose – determines when UIM should stop monitoring in seconds before the pre-set closing time of a sub-session on all the trading products of the trading classes that belong to the sub-session. Note, the implementation assumes the sub-session closing time would not change once a session is started.
4. Any quote activity counts as valid UIM activity, including quote activities on test classes, but only when the trading product is in OPEN/FAST/HOLD/END_HOLDING states. Note, the quotes that are entered before OPEN would be counted as valid quote activity at the point the product is opened and its activity time is set to be the product opening time + the [secondsAfterOpen] parameter as described above.
5. UIM won't kick in if a product is NOT in OPEN/FAST/HOLD/END_HOLDING states, and, within the other parameters described above.

A best practice suggestion –

1. On the user side, if his UIM feature is enabled, and, the user wants to avoid UIM action under normal circumstances but keep the UIM guard for abnormal conditions, he should continuously quote on at least one product of a test class on each trade server instance that he would have quote activities on non-test classes.
2. On CBOE side, we should make sure
   a. All users are aware of the test classes on each trade server instance
   b. All the test classes on each trade server instances are open BEFORE the other trading classes

Additional notes:

1. The UIM enhancement is configured for all sessions, including HTS, C2, ETS, ONE, COF, CFE, but only HTS is tested, so there would be no guarantee on UIM functionality for any other sessions.
2. The attributes mentioned in the summary, except for the new UIMInterval user property and the "UIM active" trading property that can be updated intraday, all the others require minor release to make their changes into production.

Additional notes 2:

There is a change on setContext.database, per infra team's request, to enable redundant persistence connections. The change is independent from any functional features in 8.4, but, it is merged into 8.4 along with the UIM dev branch.

WR6443 – CBSX Contingency Cross Enhancements Phase 3

Goal

This goal of this project is to allow the Tied Sweep and Cross contingency types to violate the NBBO (no linkage routing necessary), but must cross at or within the CBOE BBO. If there are protected orders in the way, those orders must be swept first.

Summary of changes for server side

The order may come into the system as a paired list or as individual orders. Individual 'tied sweep and cross' orders that come in will start a timer and wait until another order comes in to cross with it (must match up contingency, price, size, and opposite side). If the timer expires for an order, the order will be canceled. Note that the quantity must be the same for the pair to begin, but it's possible that the 2nd order may only trade partially if the first order has to trade with the book first (i.e. sweep).

- The existing trade type of **"R"** (REGULAR_TRADE) should be used on these trades.
- The existing bill type code of **"C"** (Cross) should be used on these trades.

WR 6448 – CPS Phase II(Market Order splitting functionality)

Goal

The goal of this project is to split Cross product Spread Market orders that can not be fully traded in CBOEDirect, thus increasing the trading volume in CBOEdirect.

Summary of changes for server side

- Cross product market orders will trade using the current CBOEdirect's COA functionality.
- Any remainder quantity after the COA will be split into individual leg Market Orders
- Split leg orders for Option will be sent to CBOE(HyTS) exchange and split leg order for stock will be sent to CBSX for trading as a regular Market Order.
- Split leg orders will trade in CBOE and CBSX.
- SplitOrderService in HyTS will get all the fill reports for individual legs. When fills have been received in ratio for the CPS orders, CPS fill reports will be published to the user.
- CBOEdirect GUI will provide details of split leg order, when Order History for CPS order is requested from the GUI.
- If CPS order is not completely filled after certain time defined by "CPS order splitting Timeout" trading property, a text message will be sent to HelpDesk providing the CPS order and partiall fill details.
- CPS Market order splitting functionality can be turned on or off by "Enable CPS order Splitting" trading property.

WR 6411 – Short Sale Marking

Goal

The goal for this project is to provide a means to mark Quotes, Complex order with stock, Cancel / Replace requests for complex order with stock, Auction response, and CBSX internal (cd) reports with a short sell indicator.

Summary of changes

Following areas of the system have been changed to allow for short sell indicator: -

- Order Entry for Complex Order
- Order Entry Cancel Repleace Complex Order
- Complex Order with stock legs to OMT
- Complex Order with stock leg to PAR
- Quotes

Short Sales Marking - Firm Property

This property will enforce a firm to mark their sells as a short sale. This is enforced in quote processing and Order Entry of Complex orders with a Stock leg.

If the “shortSalesMarking” flag is set to ‘True’ and if the quote entered is for Stock products then system will reject the quotes with a DataValidationException and cancel the current quote.

If the “ShortSalesMarking” flag is set to True and if a Complex Order entered contains a stock leg, The system will check to assure the side value for the stock leg has been specified. If the side is not marked buy or sell or sell short the system will reject the orders with error message as ‘SHORT_SELL_MARKING_NOT_SPECIFIED’.

WR 6687 – C2 Matching Algorithm Enhancements

Goal

The goal of the C2 Matching Algorithm Enhancements project is to add a new matching algorithm for C2_MAIN (customer priority, preferred MM and price- time priority) and disable the ability to configure any matching algorithm for C2_MAIN that combines customer priority and size-based allocation (like pro-rate).

Functionality changes

1. For C2_MAIN, users of the SA GUI will be prevented from configuring any algorithm that includes customer priority and size-based allocation (like prorata) combination.
2. Added a new overlay for Preferred MM where the PMM receives an allocation of any remainder of the order after filling others with priority.

3. Eliminate size based allocation from all Allocation Trade Types. The new configuration must be as follows:
    a. Allocation Trade Type = Regular (0)
    Default Strategy Code = Price Time (1)
    Prioritized Strategy = Customer (4), Preferred DPM For C2 (22)

    b. Allocation Trade Type = Internalization Auction Trade (5)
    Default Strategy Code = Price Time (1)
    Prioritized Strategy = Customer (4), Internalization Fixed pct (30)

    c. Allocation Trade Type = HAL Allocation (2)
    Default Strategy Code = Price Time (1)
    Prioritized Strategy = Customer (4)

    d. Allocation Trade Type = Strategy Auction Trade (8)
    Default Strategy Code = Price Time (1)
    Prioritized Strategy = Customer (4) , Preferred DPM For C2 (22)

Summary of Changes for Server Side

1. Added a new allocation strategy Element AllocationElectronicPreferredDPMStrategyElement.java" that does PMM allocation.

2. Added const tradingProperty::AllocationStrategyCode PREF_DPM_ELECTRONIC = 22; to constants.idl.

3. Added validation for allocation strategy configuration for C2_MAIN in TradingPropertyServerLocalImpl.java

4. Added time based sorting in AllocationPriceTimeStrategy.java

5. Added a new script initNewAllocationStrategiesForC2 to initialize the allocation strategy for C2_MAIN default class.

WR6444 –Faster Startup

Goal

The goal of this project is to speed up CBOE Direct trade engine starup

Summary of changes for server side

Create Business Day Changes

Currently business day creation is done in sequential for each session and session template which take between 20 to 25 minutes to complete. In this realease we make changes to start business day creation in parallel for each session and session template.

A thread will be started to process business day for each session, and within the session another thread will be started to process each session template. We use thread pool to limit number of thread started during business day creation. We have limitied the number of concurrent thread for trading sessions as well as for trading session templates.

The following changes is introduces in the script, xml and dtd files.

1. In setContext.main –**`DparallelCreateBusDayFlag`** argument is added to allow us to create business day in parallel or in regular way.
2. a new home ThreadingSessionThreadPoolHome,tradingSessionThreadPoolHome.xml and tradingSessiionThreadPoolHome.dtd is created to configure number of thread we start for session and session template.

Content of the TradingSessionThreadPoolHome.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE GlobalTradingSessionThreadPoolHome SYSTEM
"../dtd/TradingSessionThreadPoolHome.dtd">
<GlobalTradingSessionThreadPoolHome>
   <TradingSessionThreadPoolHomeImpl
name="TradingSessionServiceThreadPoolHomeImpl">
      <TradingSessionServiceThreadPoolHomeImplProperties>
       <sessionCorePoolSize>15</sessionCorePoolSize>
         <sessionMaximumPoolSize>50</sessionMaximumPoolSize>
         <sessionKeepAliveTime>10000</sessionKeepAliveTime>
       <sessionTemplateCorePoolSize>30</sessionTemplateCorePoolSize>
         <sessionTemplateMaximumPoolSize>50</sessionTemplateMaximumPoolSize>
         <sessionTemplateKeepAliveTime>10000</sessionTemplateKeepAliveTime>
      </TradingSessionServiceThreadPoolHomeImplProperties>
   </TradingSessionThreadPoolHomeImpl>
</GlobalTradingSessionThreadPoolHome>
```

WR6723 – C2 Linkage: PMM_Changes

Goal

The goal of this project is to enable to send linkage orders between C1 and C2 including the PMM information as part of the fix messages.

Summary of changes for server side

Include the PMM information stored in the extensions field with key ***PDPM*** from the original to the linkage derived order when its created. The information will be sent to the linkage server where it will be mapped according to the router selected.

WR6541 – Trade Match Data Recovery

Goal

The goal of the Trade Match Data Recovery project is to be able to "re-play" data into CTMr that is not already in CTMr or has not been processed by CTMr.

Summary of changes for server side

The scope and approach are as follows:

- CBOEdirect and Market Replay will develop scripts that format selected trade data to look like initial input to CTMr. Both CBOEdirect and Market Replay already have the ability to select trades and those existing capabilities will be used. The trade data will be output in Excel or text files.
- CTMr will accept the files through a 'back-door' and the system will ignore the date and duplicate record checks that it normally uses for data that comes through the front-door.

Bug fixes included in this release

SEDL	PITS	Test Plan	Assigned	Date	Origin	Status	Description
N/A	22099	No	Jay		Production	Assurance	Short sale indicator not showing up in market replay. This was fixed with Short sale marking project. We noticed this during development. I am adding this just to Mention that this ticket will be fixed wit SSM.
7099	12126	Yes	Mike H.	12/3	Production	Assurance	For CBSX, when a fill report is received for a linkage order that has been cancelled, product name needs to be shown, not just the product key.
7584	13402	Yes	Hemant		Production	Assurance	Duplicate strategies created
7746	13868	Yes	Sri		Production	Assurance	Complex reserve order resting in COB is getting taker billing type
7841	14018	No	Beckle			Assurance, handled as part of HDE4 requirements	didn't receive the Federatre REsponse of 'Cancel Orders By BCs' prompt after issuing the cancel command
7931	11483	No	Joel D.	12/3	Production - GUI	Assurance, handled as part of HDE4 requirements	In the SAGUI for the SPREAD SHEET download function, we need to add a correspondent and firm field to the sheet for canceling orders.
7966		No	Mike H.		Production	assigned	Global “unknown error creating event log entry”
8066	14420	Yes	Raghu		Production	Assurance	This morning we had a total of 149 order that OHS reflected as being stuck on OMT: 9IBFR1
8101	14571	Yes	Ellen	8/31	Production	assurance	The EOD script missed the exception trigger commit failure, which resulted in order not being flagged as purged.
8120	14498	Yes	Raghu	11/16	Production	assurance	TFL BEAP did not get a BTT notification. Par officials acronyms all start with an 'X' and there is logic that excludes X acronyms from BTT.
8181	14470	Yes	Brian	12/11	Production	Assurance	Auction end times are marked as null in SBT_AUCT table
8219	14892	Yes	Mike H.	4/10	Production	Assurance	When the par broker attempted to send a "Manual Quote" from W164 he received an error message:
8309	15154	Yes	Mike H.	4/10	Production	Assurance	Add Product key to information message when an order is the tradable to unbook and not found
8342		Existing product	Meng		Development?	Assurance	PRODUCT DATA CACHE HAS A POTENTIAL FLAW IN UPDATING

		data cache regression					A SPREAD TRADING CLASS.
8355	15263	Yes	Ellen	12/10	Production	Assurance	Order traded worse than limit price
8357	15139	Yes	Mike H		Production	Assurance	Clearing firm number got changed
8359	15431		Yonghua		Production	assurance	We are asking for a Next and Back button once the intial message is displayed
8410	15533	No	Kevin		Production	Closed	Object editor fix
8425	15317	Yes	Dave Dowat		Production	assurance	Server needs to add TTE emit point for getQuoteQueryDataForProducts calls made from OHS to TradeServers
8442	15572	Yes	Hemant		Production	Assurance	NTF6 routed a strategy order (EZO 2447). The order shows cancelled at 13:31:14. The user received a Fill report @ 14:22:48. If the order was cancelled, why/how did the user get filled
8455	13181	Yes	Ellen		Production	Assurance	On mdgc02, please arrange the response to the command 'globalExternalServices' sequentially. On Prdlc01, please arrange the response to the command 'checkOHSLinkageTestOrder' sequentially
8464	15929	Yes – ITG test plans	Sri	10/29	Holdover from 8.3	assurance	AIM Sweep order with both A:AIS in the Optional Data and contingency of SWEEP_BOOK on PO / SWEEP on MO is being accepted and is being processed for PO trading
8502	15987	No	Jim C.		Production	Assurance	Primary Order trade at a worse price
8518	16609	No	Jack		Production	closed	Query par trades resulting in MMTN generation
8621	19676	No	Peng		Production	assurance	race condition is within the CAS server between the cas product cache system and the cas routing proxy.
8651		Existing regression	Francisco		Development	Assurance	Global Server Processes to use new JDBC driver
8671	21005	No	Sushant			Assigned	The following list of new tables have been created in SBT Global database which caused the weekend export/import failure
8703	15292	No			Production	Unassigned	involving RESERVE orders the displayed qty in Current market and Book Depth is different than what is expected
8782	25310	No	Sri		Production	assurance	For carded trades, we should accept firm's time if it is in the future. The firm's time stamp should be overwritten with the CBOE's time stamp in this scenario.
8809	27334	Yes	Mike H		Production	assurance	Billing codes are incorrect on linkage orders

8882	25341	No			Production	Unassigned	after we cancel a booked order it looks like there is no market check is happening and hence the auction is not ending prematurely
8883	27485	Regression	Sri		Production	Assurance	After AIM auction trades in CBOE happen for an AIM Auto-Link order, the Top of the Book check is not reflecting the change in market
8922	34015	Yes	Hari		Production	Assurance	When an order is routed from the OMT and the routing attempt fails, an exeption should be thrown back to the OMT and the order should not be rem,oved from OMT
8925	34675	Yes	Mike H.		Assigned	Assurance	Firm receiving some trade reports without a billing flag.
8931		No	Ellen		Development	assurance	Run GC under compressed oops
9071	32802	Yes	Mike H		Production	Assurance	Billing Codes are incorrect on trades created from a quote moving and triggering the execution of an order resting in COB
	41598	Yes	Esquivel		Development	Unassigned	Cancel event from linkage added to stock order history
	41599	Yes	Esquivel		Development	Unassigned	Linkage fill not applied to sell short order when ETF Flag turned off.
	37298	Yes – Part of ITG Checkout	Yaowapa		Production	Assurance	Change quickstart mode for cas to use persistant cache.
Perfor-mance related		yes	Jim Cowell	9/3	Production	Assurance	IOC order query Market Data Service is taken out and replaced with query Order Book for CBOE market – latency project.
Perfor-mance related		regression	Jim Cowell	9/3	Production	Assurance	Fixing Cross Product Emit Point for ETS with leg orders to carry the TID to forward to ETS from HTS/STS in order to track and analyze TTE data cross ETS/HTS/STS on CROSS PRODUCT orders.

Overview of Infra changes in this release

Work requests included in this release

Infra 13.2

Infra 13.2

The following changes are included in the infra 13.2 release (I am expecting additional documentation as we approach the merge data):

-JGrinder ObjectEditor race condition / bugfix

- SEDL 7981: JGrinder Redundant Broker support within Foundation Framework Containers
- AMQBroker support for cached GMD subjects (the “Big G” GMD support)
- Linux support in all infra scripting
- Automated ProcessWatcher / SMS registrations for Infrastructure processes
- DN Filter Re-propagation at restart (intended to mostly help in DEV/ATG/API since these environments sometimes have irregular restart sequences performed)
- multicast event channel support

Overview of CD Client changes in this release

Work requests included in this release

6301 – Buffer support for current market

WR6301 – Buffer Support For Current Market

Goal

The goal of this project is to allow for the inclusion of the current market data in the MsgCodec buffers to be sent to clients currently receiving the CurrentMarket structs.

Summary of changes for Client side

CFIX (for W_MAIN) and MDCAS (for W_STOCK) will read CurrentMarket and MarketBuffer channels. They should be installed **before** their respective TradeServers, and they will read CurrentMarket from the 8.3 servers and MarketBuffer from the 8.4 servers; no Bridge will be needed for these two sessions.

CAS (for ONE_MAIN) and FIXCAS (for ONE_MAIN and CFE_MAIN) will be configured to read MarketBuffer channels (admin commands or configuration could change them to read CurrentMarket instead). They should be installed **after** the TradeServers. The Bridge processes will provide continuity until all new CAS and FIXCAS engines have been installed.

WR 6185 – Global FE Deprecation

Goal

This project will eliminate the call to Global v20frontend. Instead, the routing logic will reside in the SACAS, which itself will delegate the requests to Global or Trade server.

.

Summary of changes

The following areas of the system have been changed to allow for Global FE deprecation: -

- **Local Mode**

The SACAS will be configured to run in LOCAL mode. It will thus bypass the Global v20frontend and makes all calls directly to the GC or Trade server.

- **SACAS Services Requests**

Currently, the SACAS will route incoming requests to Global FE. The routing proxy class in Global Fe will delegate incoming requests to appropriate services on the BC or the GC. To skip going to Global FE, the proposed change will move the routing proxy classes to the SACAS. These classes maintain a table of maps for every service to their respective routes on the BC or the GC.

- **Required FECache**

The new SACAS will get currentBusinessDay, PQS and PCS Cache from FECache and we will thus need the FECache to be up on frontend host, before the SACAS is brought up. SACAS1 will tide with FE03/FE04 and SACAS2 will tide with FE05/06 sides on the frontend hosts.

- **SACAS Startup**

As far as startup procedure are concerned, the SACAS will access FECache first to get the previously cached data. In the case that both FECaches are not available, SACAS will query directly to Global to get those caches.

- **New Environment Variable**

A new environment variable is introduced to config/bin/setContext.v2sacas01:

SACAS1

export CAS_REMOTE=false

export GLOBAL_IDENTIFIER=Global

export FRONTEND1_HOSTNAME=prdfe01xport FRONTEND2_HOSTNAME=prdfe02export FRONTEND_IDENTIFIER=PRDFE01

Deleted: ¶
e

Deleted: ¶

SACAS2

export CAS_REMOTE=false

export GLOBAL_IDENTIFIER=Global

xport FRONTEND1_HOSTNAME=prdfe03xport FRONTEND2_HOSTNAME=prdfe04export FRONTEND_IDENTIFIER=PRDFE03

Deleted: *e*

Deleted: ¶
e

Deleted: ¶

If this variable is not set up, SACAS will fail to start with a Foundation FrameWork error.

SAGUI:

SAGUI was modified to remove the AgentQuery window. This functionality was no longer being used and contained queries to NBBOAgentAdminService which is being eliminated. The gui AgentQuery package is removed and the corresponding internalTranslator api interfaces and impls.

Appendices

Appendix A – Support notes

UIM specific AR commands

ar GlobalServer getProperties UserProperty "*W_MAIN*|UIMInterval"

ar *HybridTradeServer1* showUIMParams

ar *HybridTradeServer1* setUIMParam help then go from there

ar *EquityTradeServer1* displayUserPropertyCache ALL

ar *EquityTradeServer1* clearUserPropertyCache

Note, anything in *"italics"* means it can be changed with other session names or other trade server instance names.

Buffer support for current market AR commands

Trade Server related AR commands:

The below two AR commands are supported on all trade servers i.e. C1, C2, ONE, CFE and COF. By default trade servers publish the CM events exclusively over Market Buffer (MB) event channel(s) and basically the two AR commands allow the trade server to switch the CM events publishing between default MB and legacy CM event channels.

enableLegacyCMPublisher - Switches CM events publishing from codec based MB event channel(s) to legacy CM structure based CM event channel.

disableLegacyCMPublisher - Switches CM events publishing from legacy CM structure based CM event channel to codec based MB event channel(s).

showLegacyCMPublisherStatus – Indicates whether the legacy CM publisher is enabled or disabled.

CoppBridge AR commands:

The CoppBridge by default consumes the CM events exclusively from the MB event channel(s) and basically the two AR commands allow the CoppBridge to switch the CM events consuming between default MB and legacy CM event channels.

enableBridgeLegacyCMConsumer - Switches CM events consuming from codec based MB event channel(s) to legacy CM structure based CM event channel.

disableBridgeLegacyCMConsumer - Switches CM events consuming from legacy CM structure based CM event channel to codec based MB event channel(s).

showBridgeLegacyCMConsumerStatus – Indicates whether the legacy CM consumer is enabled or disabled.

CoppAdapter AR commands:

The CoppAdapter by default consumes the CM events exclusively from the MB event channel(s) and basically the two AR commands allow the CoppAdapter to switch the CM events consuming between default MB and legacy CM event channels.

enableCFELegacyCMConsumer - Switches CFE_MAIN CM events consuming from codec based MB event channel(s) to legacy CM structure based CM event channel.

disableCFELegacyCMConsumer - Switches CFE_MAIN CM events consuming from legacy CM structure based CM event channel to codec based MB event channel(s).

showCFELegacyCMConsumerStatus – Indicates whether the legacy CM consumer is enabled or disabled for CFE session.

enableONELegacyCMConsumer - Switches ONE_MAIN CM events consuming from codec based MB event channel(s) to legacy CM structure based CM event channel.

disableONELegacyCMConsumer - Switches ONE_MAIN CM events consuming from legacy CM structure based CM event channel to codec based MB event channel(s).

showONELegacyCMConsumerStatus – Indicates whether the legacy CM consumer is enabled or disabled for ONE session.

enableBridgeLegacyCMConsumer - Switches COF_MAIN CM events consuming from codec based MB event channel(s) to legacy CM structure based CM event channel.

disableBridgeLegacyCMConsumer - Switches COF_MAIN CM events consuming from legacy CM structure based CM event channel to codec based MB event channel(s).

showBridgeLegacyCMConsumerStatus – Indicates whether the legacy CM consumer is enabled or disabled for COF session.

StockCFNAdapter AR Commands

The StockCFNAdapter can process data from either source CurrentMarket or MarketBuffer channels. The default behavior is determined by the value of the **useMarketBufferChannels** boolean parameter in the **StockCFNQuoteConsumer.xml** file. If this parameter is set to “true" the adapter process events received on the MarketBuffer event channel. If the parameter is set to “false” then events from the CurrentMarket channel are processed. This behavior can be modified at run time by using the following ar commands:

getEventChannelProcessingFrom - Indicates what channel the StockCFNAdapter process is currently processing from (listening to), either CurrentMarket or MarketBuffer event channels.

switchEventChannelListener - Switches StockCFNAdapter process listener from CurrentMarket to MarketBuffer event channels and vice versa.

Global FE Deprecation

New AR commands for RateMonitor

A script **casadmin** invokes the **ar** command with the appropriate name for the Quote/Order RateMonitor process.

Quote RateMonitor

```
casadmin quoteRateMonitor username sessionname
```

This ar command will retrieve quote ratelimit from propertyService from global and display quote calls, quote rate, and quote block size for user and session name.

Order RateMonitor

```
casadmin orderRateMonitor username sessionname
```

This ar command will retrieve order ratelimit from propertyService from global and display order calls for user and session name.

Appendix B – Database Changes

HDE-4 Database Changes

1. sbt_tradereport table
    a. Add column
        i. Column name: CLASSKEY
        ii. Data Type: NUMBER(20)

2. bulk_action_dtl
    a. Add column
        i. Column name: TRADEID
        ii. Data Type: NUMBER(20)
    b. Modify column
        i. Column name: TARGETDBID
        ii. Modification: Remove NOT NULL constaint

3. Add new table
    a. Table name: CROSSPRODUCTTRADERELATION
        i. Columns
            1. DATABASEIDENTIFIER, NUMBER(20)
            2. LOCALTRADEREPORTID, NUMBER(20)
            3. REMOTETRADEREPORTID, NUMBER(20)
            4. REMOTEPRODUCTKEY, NUMBER(20)
            5. REMOTETRADETIME, NUMBER(20)
            6. REMOTETRADEQUANTITY, NUMBER(10)
            7. REMOTESESSIONNAME, VARCHAR2(30)

4. Add index in sbt_tradereport
    a. Index name: SBT_TRADEREPORT_I3
        i. Columns
            1. TIME
            2. CLASSKEY
            3. PRODUCT

5. Add index in sbt_tradereportentry
    a. Index name: SBT_TRADEREPORTENTRY_I4
        i. Columns
            1. BUYORDERID
            2. SELLORDERID

Short Sales Marking Database Changes

6. Add column in sbt_tradereportentry
    a. Add column
        i. Column name: sell_side_ind
        ii. Data Type: VARCHAR2(1)

alterTradeReportEntry_ShortSale.sql is the alter script for making this changes.

Appendix C – Operator Procedure Changes or Additions

Global FE Deprecation

Normal start

SACAS will get currentBusinessDay, PQS and PCS Cache from FECache if these are not previously cached. SACAS1 will use the FECache from the FE03/FE04 pair and SACAS2 will use the FECache from the FE05/FE06 pair.

In the case that both FECaches are not available, SACAS will query directly to Global to get those caches.

Quickstart

SACAS will use its in-memory cache, and not query the Server for currentBusinessDay, PQS and PCS on the startup except when cache files do not exist on disk. In such a case, there is no difference in the behavior for normal start vs. quick start.

Appendix D – Installation Procedures

Installation procedures GC02A/B

SPECIAL IMPLEMENTATION DEPENDENCY
IIOP Conversion

We will be enabling IIOP ports for the following processes on the IGC machine:

- ***Central Logging, new IIOP port = 18155***
- ***Channel Admin, new IIOP port = 18129***
- ***Security Service, new IIOP port = 18130***
- ***Tran Timing Registration, new IIOP port = 18162***

NOTE: we must enable these ports on the slave-side IGC prior to the 13.2 install on the master-side.

QA Steps

At 3:15 have qa load the new software.

Server Group steps

Most of the steps here can be done after 3:15.

Only if needed fix the setContext file in /sbt/prod/tradeeng directory. After you login, if the setContext version has changed then the setContext is not run and you will not be able to start any process. In this case just copy the /sbt/prod/tradeeng/CBOEDIR_8.3/setContext.template file into /sbt/prod/tradeeng as setContext and then correct all the variables in it. You can use the old setContext file as an example to update the new file.

Change run_dir links for previous release.

Change the run_dir link in /sbt/prod/tradeeng to point to the new release.

Logout and log back in as tradengp.

Run script $RUN_DIR/bin/genWatchedProcessList. Verify that $RUN_DIR/properties/ WatchedProcessListServer.out is generated and that all processes are listed correctly in this file.

Do a diff against the old and new WatchedProcessListServer.out file to ensure they are the same. **DO NOT go any further if this does not work.**

Installation procedures GC01A/B

- ### QA steps

1. At 3:15 have qa load the new software using the QA setup steps.

- ### Infra group steps after end of all sessions.

2. Load new ACL

- ### Server group steps after end of all sessions.

4. Install new software CBOEDIR_8.4 release,
5. Update setContext in home directory with changes
6. Change run_dir links and log back in.
7. Do any database conversions if needed..
8. Start the GC processes.
9. Add new trading properties and event channel if any:
   - TradingPropertyServiceClient setPropertyDefinitions ${RUN_DIR}/properties/TradingPropertyDefinitions.csv
10. execute “$RUN_DIR/bin/ rollout/addCPSSplittingTradingProperties “. This will set the default trading property for CPS splitting.
    - Execute “getTradingProperty W_MAIN default EnableCPSMktSplitting” and verify that output is “EnableCPSMktSplitting = false”
11. **This step needs to be executed only for C2** .Execute

    “ ${RUN_DIR}/bin/initNewAllocationStrategiesForC2 –run “

    to initialize the allocation strategy for default class for C2_MAIN.
12. **This step needs to be executed only for C2** . Execute

    “ getTradingProperty C2_MAIN default AllocationStrategies “

    to verify the trading property was set properly. The output should be:

    Result:

    Results from "get":

    Property value = 0=1;4;22,2=1;4,3=14,5=1;4;30,6=1,8=1;4;22

    ClassKey = 0

    Sequence = 20
13. Enable global external connections on the GC.

14. Start all sessions, do a quick quote and order test on one class on each bc.
15. Ops runs an "IPD Resync" report (Window staff would verify the data, a clean report is a good report)
16. End all the sessions.

- **Other Verification after GC Upgrade**

13. Check all files (.log, .debug, .out, .err) for errors, exception's and high system alarms.
14. If you are installing the Slave side box then perform a fail-over so that the upgraded box becomes Master and then continue on with the remainder of the plan.
15. Start all sessions using the SA GUI
16. Verify on prdgc01a/b that there are no products in NO_SESSION state.
17. Use SAGUI to open test products on all the BC's. You can get the list of test classes from operations (This list is taped to one of the monitors in the basement)
18. Login to 2 trader GUI's and bring up the market display window and status window on each of the GUI's. NOTE: Status Window is a scrolling display that shows status messages for orders and quotes as and when they occur. (This window can be used to verify that OrderStatus and QuoteStatus events are working).
19. Do test trades using quotes/orders on at least one BC for all sessions.
20. For Futures TradesServers (TradeServer3) – You can use the same user to enter Orders/Quotes.
21. For Hybrid TradeServers (you will need to use quotes to trade with each other) + Enter a couple of orders.
22. Verify that CurrentMarket (Mkt Bid and Mkt ask) is showing up on the market display window. This tests out that current market events are working.
23. Verify that the last sale price and quantity fields are showing up on the Market display window. This tests out that Last Sale and Recap events are working.
24. Verify that Order Status messages (New Order, Order filled are showing up on the Status window screen)
25. Check all files on prdgc01 (.log, .debug, .out, .err) for errors, exception's and high system alarms.
26. Specific to the Faster startup first install:

```
Verification:After system starup please check
      Business day created properly.
      Session started successfully.
      No product in no session state.
```

Verification of MDGC1a/b installation

mdgc1 is verified by the opening of products via XTP replay. If the products open properly, it proves XTP data flowed through the TipsAdapters on mdgc1 and reached the trade servers.

Additionaly, the CFN Adapters are verified as follows:

1. Open three terminals to mdgc01a
2. In each, run
    a. **cfnAdmin monitor raw CFE_MAIN C 1 | grep ‘ u ‘**
    b. **cfnAdmin monitor raw CFE_MAIN C 2 | grep ‘ u ‘**
    c. **cfnAdmin monitor raw CFE_MAIN C 3 | grep ‘ u ‘**
3. Enter quotes and orders for some production CFE class (Saturday only), verify that book depth messages are seen on one of these windows.
4. *[FYI: CFE multicast host&port is idenfified as group ‘C’ (on the .8 network) and ‘D’ (on the .12 network)]*
5. *[FYI: The grep for ‘ u ‘ will display update messages only, ignoring any refreshes.]*

1. Open three terminals to mdgc01a
2. In each, run
    a. **cfnAdmin monitor raw ONE_MAIN A 1 | grep ‘ u ‘**
    b. **cfnAdmin monitor raw ONE_MAIN A 1 | grep ‘ u ‘**
    c. **cfnAdmin monitor raw ONE_MAIN A 1 | grep ‘ u ‘**
3. Enter quotes and orders for some production CFE class (Saturday only), verify that book depth messages are seen on one of these windows.
4. *[FYI: ONE multicast host&port is idenfified as group ‘A’ (on the .8 network) and ‘B’ (on the .12 network)]*
5. *[FYI: The grep for ‘ u ‘ will display update messages only, ignoring any refreshes.]*

Example output: (line sizes cause the example to wrap lines. This is all one line in the output of the test)

```
2010.09.15 14:50:08 514ms 28004531 00062 'line2'       E u
14694081145008MORN1C U0         B  0501R01B0000402400010
                                                    |’u’ means update

             |”MORN1C” is the symbol
                    | ‘U’ is the month code
                                  | “R01” is “replace level 1”

Other examples.

2010.09.15 14:50:08 514ms 28004532 00062 'line2'       E u
14694082145008MORN1C Z0         B  0501R01B0000402600010
2010.09.15 14:50:08 514ms 28004533 00079 'line1'       E u
16571480145008ETN1C  U0         B  0502R01A0000792200005R02A0000792300025
```

Verification of MDGC2a/b installation

MDBroadcastAdapter Verification:

XML:

command: grep com.cboe.server.events.MarketBufferConsumerHomeImpl ./properties/xml/MDBroadcastAdapter.xml
passing result:
<homeImpl>com.cboe.server.events.MarketBufferConsumerHomeImpl</homeImpl>

Log files:

command: grep "Adding consumer" ../log/MDBroadcastAdapter1.log
passing result:

…"MarketBufferConsumerHome>>> Adding consumer com.cboe.externalIntegrationServices.mdbAdapter.currentMarket.CurrentMarketConsumerHomeImpl$MarketBufferChannelReader@... to channel index 0 (…MarketBuffer1)" 0
…"MarketBufferConsumerHome>>> Adding consumer com.cboe.externalIntegrationServices.mdbAdapter.currentMarket.CurrentMarketConsumerHomeImpl$MarketBufferChannelReader@... to channel index 1 (…MarketBuffer2)" 0
…"MarketBufferConsumerHome>>> Adding consumer com.cboe.externalIntegrationServices.mdbAdapter.currentMarket.CurrentMarketConsumerHomeImpl$MarketBufferChannelReader@... to channel index 2 (…MarketBuffer3)" 0
:
:
command: grep "Adding consumer" ../log/MDBroadcastAdapter20.log
passing result:
…"MarketBufferConsumerHome>>> Adding consumer com.cboe.externalIntegrationServices.mdbAdapter.currentMarket.CurrentMarketConsumerHomeImpl$MarketBufferChannelReader@... to channel index 0 (…MarketBuffer1)" 0
…"MarketBufferConsumerHome>>> Adding consumer com.cboe.externalIntegrationServices.mdbAdapter.currentMarket.CurrentMarketConsumerHomeImpl$MarketBufferChannelReader@... to channel index 1 (…MarketBuffer2)" 0
…"MarketBufferConsumerHome>>> Adding consumer com.cboe.externalIntegrationServices.mdbAdapter.currentMarket.CurrentMarketConsumerHomeImpl$MarketBufferChannelReader@... to channel index 2 (…MarketBuffer3)" 0

Note: the above verification should be repeated for any additionally installed and enabled MDBroadcastAdapter process(e.g. MDBroadcastAdapter21) byond the currently enabled 20 processes.

The mdbAdapters are verified by watching an RCN overhead. Ops can configure their overhead to watch a class to be used during testing. The class should be closed and then it should be run through the various states to reach the open state. The states should be reflected on the RCN overhead as they are changed in CBOEdirect. Additionally, a quote and trade should be entered against a series being display to make sure the values update on the RCN overhead.

- **GC - Saturday verification after upgrade**

CBOEDirect verification after Global Cluster upgrade

#	Description	PASS/FAIL
1	**GC01 - Verification test**	
2	loadOpenInterest test (performed by Ops at startup- requested at weekend test meeting)	
3	Product download test – CAS (performed by Ops at startup)	
4	Restart all CAS & Fix Engines. Verify CAS startup time. (performed by Ops at startup)	
5	Start all sessions using the SA-GUI (performed by Ops at startup)	
6	Verify all products are assigned to sessions and no product is in NO_STATE (Ops at startup)	
7	Run ITG Checkout scrpt on all BC's (performed by Ops at startup)	
8	Transition all products for all sessions to PRE-OPEN state.	
9	Run the XTP replay, replaying the Friday 8:29-8:45 traffic at a 1.5 times rate. Verify that all W_MAIN classes transitioned to OPENING-ROTATION, verify ticker and recap on GUI, Verify underlying price in MDH. Verify broadcast to a PDS. Verifies Data from MDGC1.	
10	OPEN all products for all sessions.	
11	Kill a CAS to verify users are logged out - SMS test	
12	Enter Quote - Book Depth update (dynamic) - CFE/ONE_MAIN Verify that data goes ot of CfnAdapter1 on mdgc01 (3 outbound lines for CFE and 3 outbound lines for ONE). **ar cfnAdmin monitor raw CFE_MAIN C 1**	
13	Generate and verify Half Hourly reports for News Wire and HVOL. Actually sending the file to OPRA can only be verified on the weekend. Only on the weekend can you use the –opra option. * Verify hourly reports are generated every half an hour by Control-M * Run script **createHalfHourlyReport – nw W_MAIN**. * Verify that new report file is generated. log/HalfHourly_NewsWireReport_latest.xml.log * Repeat above for –hvol.	
14	Examine global log to verify open interest was loaded	
15	Using SA_GUI, spot check some open interest for some of the products. There should be non-zero values for most of the products. It is OK if a few of them have zero values because it might be a result of new series additions.	
16	Run **ar MarketDataReportServer1 showStatistics** and verify number messages received	
17	Do test trade on any production class for W_MAIN. Note : SATURDAY TESTING ONLY	
18	Repeat step 16 and verify total number of events received increased	
19	Repeat step 13 and verify the reported numbers have increased	
20	Pause XTP Replay.	
21	Perform four BCXX failovers with GC01a is master. Where XX is a single hybrid, CFE, ONE, and STOCK BC.	

22	Run ITG Checkout scripts on the failed BCs (only). While running proceed to step 23.	
23	Verify that data CM , LS and Product states are going out to PDSs. The PDSs are verified as follows: Verify the PDS Overhead is updating with values from the class being used. Quote and last sale information should change. This verifies the MDB broadcast. Depending on the type of display OPS has configured on their test PDS Overhead, the last sale might not show up. Have them switch screen types until they find the one that shows the last sale. **Make sure XTP replay is NOT running during the ITG checkouts**. XTP Replay can cause spooling and there would be a possibility of dropped quotes/trades which could cause the test to appear to fail.	
24	Failover MDGC01 and MDGC02 with new GC01a as master.	
25	Verify CFE/ONE_MAIN book depth data is being published by repeating step 12.	
26	FE failover test with GC01a as master. KILL FE03	
27 28	Restart CAS2011. Verify the cas reinitializes. Run ITG Checkouts on any of the following BC/TE combinations that use FE03. BC04/TS-4 (cas2011) BC10/TS-3 (cas2011) BC30/TS-2 (cas2011) BC82/TS-1 (cas2011) BC90/TS-3 (cas2011) BC93/TS-4 (cas2011) BC98/TS-1 (cas2011) Start XTP Replay (1.5 rate(. Open the *Market Display For Underlying* window and verify ticker and recap.	
29	VERIFY BROADCAST TO A PDS. THE PDSS ARE VERIFIED AS FOLLOWS:Verify the PDS Overhead is updating with values from the class being used. Quote and last sale information should change. This verifies the MDB broadcast. Depending on the type of display OPS has configured on their test PDS Overhead, the last sale might not show up. Have them switch screen types until they find the one that shows the last sale.	
30	GC01 Fail over (new to old)	
31	Pause XTP Replay.	
32	RUN CMi and FIX ITG Checkout scripts (all BCs) after the fail over.	
33	Verify Failover times to see how long it takes to do the complete failover (Stop + goMaster + Pre-open products)	
34	Close->PreOpen->Open all products in all sessions. Time the transition from Close to PreOpen	
35	Bounce a CAS to verify it “re-inits”	
36	Verify the CAS startup time against the previous morning startup time using the CAS Start Times script.	
37	Failover GC2.	
38	Failback GC01 to CD 8.3.5 codebase.	
39 40	Close all products and run End Of Sale and updateClose after all other tests have completed. Run and verify any **new** end of day procedures as listed in the operator procedures	

Run and verify new end of day procedure as listed in operator procedure.

Fallback

1. See operator procedures on how to failover GC's.

Installation procedures for BC

- **QA steps**

  1. At 3:15 have qa load the new software using the QA setup steps.

  For the first, follow the *Evening Installation Procedures* below and then follow the *Saturday BC Verification After Upgrade* procedures on the following pages.

- **Evening Installation Procedures - Server group steps after end of all sessions.**

  2. Verify that the BC table changes have been done as explained above.
  3. Verify that the default routing properties and trading properties have been set correctly.
  4. Shutdown tradeengine on the Master BC (tradengp and tradengh [tradengh is only for W_MAIN] login).
  5. Install new software CBOEDIR_8.4 release, change run_dir links .
  6. Set Context changes :
     i. Change the object pooling 'addCommandCacheArgs' class reference from 'MarketUpdate' to 'MarketUpdateImpl' in setContext.main(2 places) and setContext.main.C2(1 place) files. i.e. basically **replace the below line**

        addCommandCacheArgs 500 500000 MarketUpdate

        with

        addCommandCacheArgs 500 500000 MarketUpdate**Impl**

     ii. For W_ONE session on futures ONE BC(i.e. TradeServer2), add the Current Market bridge process to the ALL_SERVER_NAMES like below.

     ALL_SERVER_NAMES="${ALL_SERVER_NAMES} CurrentMarketBridgeONE"

     iii. For W_CFE session on futures CFE BC(i.e TradeServer3), add the Current Market bridge process to the ALL_SERVER_NAMES like below.

     ALL_SERVER_NAMES="${ALL_SERVER_NAMES} CurrentMarketBridgeCFE"

  7. Start the BC processes (tradengh [tradengh is only for W_MAIN]first and then tradengp logins).
  8. Run the command **`businessExternalServices start`** to make sure that remote connections are established for the adapters on that bc.

- **Evening Installation Procedures - Verification**

9. Check all files (.log, .debug, .out, .err) for errors, exceptions and high system alarms.
10. Make sure all initialization is complete on all processes.
11. If you are installing the *first* slave side box then perform a fail-over so that the upgraded box becomes Master and then continue on with the remainder of the plan.
12. Start the sessions affected by the particular BC install using the SA GUI (NOTE: some BC's can trade multiple sessions in different trade servers).
13. Verify on prdgc01a/b that there are no products in NO_SESSION state.
14. **This instruction only applies if you are installing a W_MAIN BC**.

    Run the following commands to get counts. These commands will be run again after ITG checkouts. The counts should increase.

    i. Run **ar MarketDataReportServer1 showStatistics** to get count
    ii. Run "hsAdmin –c stats –p HybridHistoryServer1" to get count
    iii. Run **"**`hsAdmin –c stats –p HybridTradeServer1`**"** to get count
15. Run ITG Checkouts
16. If installing on a BOB BC (If not installing on a BOB BC, skip this step), enter a manual quote and verify the current market is updated.
17. This instruction only applies if you are installing a W_MAIN BC. Repeat step 11. The counts should increase from the values obtained before running ITG checkouts.
18. Check all files on the affected BC's (.log, .debug, .out, .err) for errors, exceptions and high system alarms
19. This instruction only applies if you are installing a W_MAIN BC. Run the ar command **`hsAdmin –c stats –p HybridTradeServer1`** to verify that counts on the TradeServer are increasing (which means that the TradeServer is sending data to the history server).
20. CoppAdapter Verification:
    i. Make sure that the CoppAdapterFiles.txt includes the following XML files.
        1. MarketBufferConsumer.xml
        2. DIFConfigHome.xml
        3. DIFLogging.xml
    ii. Make sure that the "<listOfHomes>" section in the CoppAdapter.xml includes the following homes.
        1. MarketBufferConsumerHome
        2. MarketDataBufferCollectorReadingHome
        3. DIFLastHome
    iii. Make sure that "MarketBufferBestQuoteProvider" is included in the "<providerList>" definition of following services in the CoppApplFramework.xml.
        1. CoppCurrentMarketService
        2. CoppCboeCurrentMarketService
    iv. Run below AR commands to make sure that the CM consumer is disabled by default.

1. showCFELegacyCMConsumerStatus or ShowONELegacyCMConsumerStatus based on the configured BC/session
2. showBridgeLegacyCMConsumerStatus

21. CoppBridge Verification**:**
    i. Make sure that the CoppBridgeFiles.txt includes the following XML file.
        1. MarketBufferConsumer.xml
    ii. Make sure that the "<listOfHomes>" section in the CoppBridge.xml includes the following homes.
        1. MarketBufferConsumerHome
        2. MarketDataBufferCollectorReadingHome
    iii. Run below AR command to make sure that the CM consumer is disabled by default.
        1. showBridgeLegacyCMConsumerStatus
22. Trade Server Verification**:**
    i. Run below AR command to make sure that the CM publisher is disabled by default.
        1. showLegacyCMPublisherStatus e.g. ar HybridTradeServer1 ShowLegacyCMPublisherStatus
23. CurrentMarketBridgeCFE Verification:
    i. Process:
        1. **command:** chk | grep -i CurrentMarketBridgeCFE
            a. **passing result:** user <owner> pid <pid> <env_prefix>CurrentMarketBridgeCFE<host_name>
        2. Process Parameters:
            a. **command:** pargs <pid> | grep prefixSessionList
                i. **passing result:**... -DprefixSessionList=CFE_MAIN
            b. **command:** pargs <pid> | grep config_CMBProcessSuffix
                i. **passing result:**... -Dconfig_CMBProcessSuffix=CFE
    ii. XML:
        1. **command:** ls -la ./properties/xml/CurrentMarketBridge.xml
            a. **passing result:** -rw-r--r-- 1 <owner> <group> <size ~11559> <timestamp> ./properties/xml/CurrentMarketBridge.xml
        2. **command:** ls -la ./properties/xml/CurrentMarketBridgeHome.xml
            a. **passing result:** -rw-r--r-- 1 <owner> <group> <size ~ 930> <timestamp> ./properties/xml/CurrentMarketBridgeHome.xml
    iii. Log files:
        1. **command:** grep "Adding consumer com.cboe.internalBusinessServices.currentMarketBridge.CurrentMarketBridgeHomeImpl" CurrentMarketBridgeCFE.log
            a. **passing result:**

i. ..."MarketBufferConsumerHome>>> Adding consumer com.cboe.internalBusinessServices.currentMarketBridge.CurrentMarketBridgeHomeImpl$MarketBufferChannelReader@... to channel index 0 ..." 0

ii. ..."MarketBufferConsumerHome>>> Adding consumer com.cboe.internalBusinessServices.currentMarketBridge.CurrentMarketBridgeHomeImpl$MarketBufferChannelReader@... to channel index 4 ..." 0

24. CurrentMarketBridgeONE Verification:
    i. Process:
        1. **command:** chk | grep -i CurrentMarketBridgeONE
            a. **passing result:** user <owner> pid <pid> <env_prefix>CurrentMarketBridgeONE<host_name>
        2. Process Parameters:
            a. **command:** pargs <pid> | grep prefixSessionList
                i. **passing result:** ... -DprefixSessionList=ONE_MAIN
            b. **command:** pargs <pid> | grep config_CMBProcessSuffix
                i. **passing result:** ... -Dconfig_CMBProcessSuffix=ONE
        3. XML:
            a. **command:** ls -la ./properties/xml/CurrentMarketBridge.xml
                i. **passing result:** -rw-r--r-- 1 <owner> <group> <size ~11559> <timestamp> ./properties/xml/CurrentMarketBridge.xml
            b. **command:** ls -la ./properties/xml/CurrentMarketBridgeHome.xml
                i. **passing result:** -rw-r--r-- 1 <owner> <group> <size ~ 930> <timestamp> ./properties/xml/CurrentMarketBridgeHome.xml
        4. Log files:
            a. **command:** grep "Adding consumer" CurrentMarketBridgeONE.log
            b. **passing result:**
                i. ..."MarketBufferConsumerHome>>> Adding consumer com.cboe.internalBusinessServices.currentMarketBridge.CurrentMarketBridgeHomeImpl$MarketBufferChannelReader@... to channel index 0 ..." 0
                ii. ..."MarketBufferConsumerHome>>> Adding consumer com.cboe.internalBusinessServices.currentMarketBridge.CurrentMarketBridgeHomeImpl$MarketBufferChannelReader@... to channel index 4 ..." 0
25. StockCFNAdapter Verification:
    i. XML:

1. **command:** grep com.cboe.server.events.MarketBufferConsumerHomeImpl ./properties/xml/StockCFNAdapter.xml
    a. **passing result:** <homeImpl>com.cboe.server.events.MarketBufferConsumerHomeImpl</homeImpl>
2. **command:** grep useMarketBufferChannels ./properties/xml/StockCFNQuoteConsumer.xml
    a. **passing result:** <useMarketBufferChannels booleanValue="true"/>
3. Log files:
    a. command: grep "Adding consumer" ./log/StockCFNAdapter.log
        i. **passing result:**..."MarketBufferConsumerHome>>> Adding consumer com.cboe.externalIntegrationServices.stock.cfn.adapter.CFNQuoteConsumerHomeImpl$MarketBufferChannelReader@... to channel index
        ii. ."MarketBufferConsumerHome>>> Adding consumer com.cboe.externalIntegrationServices.stock.cfn.adapter.CFNQuoteConsumerHomeImpl$MarketBufferChannelReader@... to channel index 4 ..." 0

Note Some times after end of session the remote application will not allow us to connect to them. So the only way to verify is by looking at the log file to make sure that the program's are making an attempt to connect to the remote system on correct ip address and correct port nbrs.

- ## Final verification

26. Close all products in all sessions using the SA GUI (Pick the tab to close ALL the products).
27. Verify memory usage for **all processes** on the BC and Garbage Collection activity by comparing with the OLD and NEW .out files. **(STILL NEED TO DETERMINE WHAT PASS/FAIL IS)**
28. End all the sessions
29. If you started this upgrade on the Slave side perform a fail-over so that the upgraded box becomes Slave now.
30. Test Complete, Notify operations.

- ## Fallback

1. See operator procedures on how to failover BC's or list instructions specific to your release here.

- **Saturday BC verification after upgrade**

#	Description	PASS/FAIL
	Hybrid BC - Verification Test	
	You will need 2 trader gui's and 1 sa gui - login thru all these guis before you start the test. XtpReplay data should be captured on a production day between 8:29-8:45. This applies only if installing on a W_MAIN or W_STOCK BC.	
1	Pre-requisites – Ops has brought up system and successfully ran checkouts	
2	Verify on prdgc01a/b that there are no products in NO_SESSION state.	
3	Ask Ops to verify the "checkout results" e-mail that is sent to the CCS e-mail group when checkouts complete. There should be 3 e-mails for CMI and 3 e-mails for FIX.	
4	Verify XTP Data is being received by the trade servers, Underlying recap (W_MAIN), BOTR exchange indicators (W_MAIN and W_STOCK using a production class) - Verify using ar commands	
5	Using a Trader GUI, do MDH Queries and verify NBBO and exchange indicators.	
6	Run the following commands to get counts. These commands will be run again after ITG checkouts. The counts should increase. Run **ar MarketDataReportServer1 showStatistics** and note the number of messages received. Run "hsAdmin –c stats –p HybridHistoryServer1" to get count Run **"**`hsAdmin –c stats –p HybridTradeServer1`**"** to get count	
7	Run ITG checkouts for Trade Server 1 on the BC being installed.	
8	WHILE ITG CHECKOUTS ARE RUNNING - Verify current market, last sale, trade reports, fill reports over all external connections. Enter a quote, order, and trade for a production class for the W_MAIN, W_STOCK, W_ONE, W_CFE sessions. **CTMr** – verify there are no un-acked trade reports. This can be verified via the following command: (**TO DO: Insert command here)** **COPP** – This is for W_MAIN. verify the COPP GUI shows a queue being build for the OPRA lines. This will be true if COPP was brought up *without* OPRA being live and *without* the OPRA simulator being up. These conditions result in no place for COPP to send the data and it will queue inside COPP. **FOPP-ONE** – This is for W_ONE. verify the FOPP-ONE GUI shows an increase in the number of messages that were sent out the outbound lines. **FOPP-CFE** – This is for W_CFE. verify the FOPP-CFE GUI shows an increase in the number of messages that were sent out the outbound lines. **STOPP** – This is for W_STOCK. verify the STOPP GUI shows a queue being build for the outbound lines. This will be true if STOPP was brought up *without* STIC/NASDAQ being live and *without* the STOPP simulator being up. These conditions result in no place for STOPP to send the data and it will queue inside STOPP. **PDS** - This instruction only applies if you are installing a W_MAIN BC. Verify the PDS Overhead is updating with values from the class being used. Quote and last sale information should change. This verifies the MDB broadcast. Depending on the type of display OPS has configured on their test PDS Overhead, the last sale might not show up. Have them switch screen types until they find the one that shows the last sale. Make sure XTP replay is NOT running during the ITG checkouts. XTP Replay can cause spooling and there would be a possibility of dropped quotes/trades which could cause the test to appear to fail.	

9	Run the following commands to get counts. The counts should have increased from the values obtained before the ITG Checkout Run MDRS Admin Request to show "last sale" count Run "hsAdmin –c stats –p HybridHistoryServer1" to get count Run **"**`hsAdmin -c stats -p HybridTradeServer1`**"** to get count	
10	If this is a BOB BC installation Operations or on site support staff need to enter a manual price report in for one of the products on the Trader GUI. The last sale column should be updated. Check, MDH, there should be an entry for this last sale.	
11	**BC Failover test**	
12	For the BC being installed, use the procedures as listed in the operator procedures and have ops execute the BC failover over procedures.	
13	Time the fail over, it should take roughly 3 minutes	
14	After the fail over run the ITG checkout script on all BC's.	
15	Enter Manual Quote (only if installing on a BOB BC, otherwise, skip this step) – Verify Current Market Update	
16	Enter Manual Price Report – Verify Last Sale Update – Verify MDH entry If installing on a BOB BC: (otherwise, skip this step) Enter Manual Quote – Verify Current Market Update Enter Manual Price Report – Verify Last Sale Update – Verify MDH entry	
17	Open DSP (Display Price Series) screen and check information displayed for product. The DSP screen is opened on the Trade GUI using a login that has a *price reporting* roll.	
18	Fallback to the new BC and run ITG checkouts on all 4 trade servers for the new BC.	
19	Close the products, End the sessions and verfy that all the sessions have ended successfully.	

Appendix E – Rollout Plans

???

IGNORE THE REST OF THIS DOCUMENT FOR RIGHT NOW. NOT SURE WHAT SHOULD BE HERE OR NOT….

Installation procedures – SACAS hosts

QA steps

On installation day after end of trading, deliver planned INFRA and SBT releases and scripts to designated SACAS and SAGUI boxes.

Client group steps

Set/update these variables in config/bin/setContext.v2sacas01

```
export CAS_REMOTE=false
export GLOBAL_IDENTIFIER=Global
export FRONTEND_IDENTIFIER=PRDFE03 (for sacas1)
export FRONTEND_IDENTIFIER=PRDFE05 (for sacas2)
```

Refer to ***CASProductionInstallation.doc*** and follow the implementation procedures.

SACAS rollout will start on one host; and leave it run for 3 days. The fourth day, the other SACAS box will install new software.

GUI group steps

Install SAGUI on one box (possibly 2 or 3 boxes) so that SACAS can be verified.

Verification

Verify that each SACAS rolled out is visible in Patrol.

Use new SAGUI to log on and ensure that it receives data from SACAS. In particular, check the User Management Window, the Product Class Groups window, and the Product Definition window.

On the first day of SACAS rollout, check all files (.log, .debug) for errors, exceptions and high system alarms.

Fallback

Follow the standard backout procedure.

Installation procedures – CAS hosts

QA steps

On installation day after end of trading, deliver planned INFRA and SBT releases and scripts to designated CAS boxes.

Infrastructure group steps

Update the ExtentMaps for the designated CAS boxes and refresh the ChannelAdmin.

Client group steps

Configure CAS to use JDK 1.6.0_18.

Refer to ***CASProductionInstallation.doc*** and follow the implementation procedures.

Verification

Verify that each CAS rolled out is visible in Patrol.

If today's rollout included a floor cas (cas0014, cas0015, cas0016, cas3005), use a Trader GUI to connect to it and make a trade using a test symbol. Ensure that the MarketDisplay window uses MDX. If possible, do two tests, one with a product in session W_MAIN and one with a product not in W_MAIN.

On the first day of CAS rollout, Infrastructure group verifies that CAS is getting status via the ExternalQuoteStatus and ExternalOrderStatus channels.

On the first day of CAS rollout, check all files (.log, .debug) for errors, exceptions and high system alarms on the CAS host and on the appropriate FE pair.

Fallback

Stop CAS and Infra processes.

Client Group will change the run_dir and v2cas* directories to previous release.

Operations will start Infra and CAS processes via Patrol.

Installation procedures – FIXCAS hosts

QA steps

On installation day after end of trading, deliver planned INFRA and SBT releases and scripts to designated FIXCAS boxes.

Infrastructure group steps

Update the ExtentMaps for the designated FIXCAS boxes and refresh the ChannelAdmin.

Client group steps

(FIXCAS should already be configured to use JDK 1.6.0_18).

Refer to ***CASProductionInstallation.doc*** and follow the implementation procedures.

Verification

Verify that each FIXCAS rolled out is visible in Patrol.

On the first day of FIXCAS rollout, run a FIX script to connect to the FIXCAS and make a trade using a test symbol. If possible, do two tests, one with a product in session W_MAIN and one with a product not in W_MAIN.

On the first day of FIXCAS rollout, check all files (.log, .debug) for errors, exceptions and high system alarms.

Fallback

Stop FIXCAS and Infra processes.

Client Group will change the run_dir and v2fixcas* directories to previous release.

Operations will start Infra and FIXCAS processes via Patrol.

Installation procedures MDCAS hosts

QA steps

On installation day after end of trading, deliver planned INFRA and SBT releases and scripts to designated MDCAS boxes.

Infra group steps

Update the ExtentMaps for the designated MDCAS boxes and refresh the ChannelAdmin.

Configure MDCAS to use JDK 1.6.0_18.

Configure INFRA installation.

Verify as Operations runs startInfraSystem.

Client group steps

Refer to ***CASProductionInstallation.doc*** and follow the implementation procedures.

Verification

Ensure that Infra System and MDCAS processes start successfully when Operations starts them.

On the first day of MDCAS rollout, check all files (.log, .debug) for errors, exceptions and high system alarms.

Fallback

Follow the standard backout procedure.

Installation procedures – CFIX hosts

QA steps

On installation day after end of trading, deliver planned INFRA and SBT releases and scripts to designated CFIX boxes.

Infra group steps

Update the ExtentMaps for the designated CFIX boxes and refresh the ChannelAdmin.

Set up CFIX to use JDK 1.6.0_18.

Configure INFRA installation.

Verify as Operations runs startInfraSystem.

Client group steps

Refer to ***CASProductionInstallation.doc*** and follow the implementation procedures.

Verification

On the first day of CFIX rollout, check all files (.log, .debug) for errors, exceptions and high system alarms.

Fallback

Follow the standard backout procedure.

Installation procedures – MDX hosts

QA steps

On installation day after end of trading, deliver planned INFRA and SBT releases and scripts to designated MDX boxes.

Infrastructure group steps

Configure the INFRA installation.

Verify the installation.

Client group steps

Modify .setenv to use JDK 1.6.0_18 build 7.

Wait till Infra group is done with the new Infra installation.

Refer to ***CASProductionInstallation.doc*** and follow the implementation procedures.

Verification

Verify startup and shutdown through command line and patrol

On the first day of MDX rollout, check all files (.log, .debug) for errors, exceptions and high system alarms.

Verify the subscriptions work. If server is up, verify MDX is delivering market data as well.

Fallback

Follow the standard backout procedure.

Installation procedures LC's

QA steps

At 3:15 have qa load the new software using the QA setup steps as documented above.

Server group steps

Master or Slave side can be upgraded after 3:15. There is no need to wait for End Of Session.

If installing the master side box. Then shutdown tradeengine processes on the Slave Side first then shutdown tradeengine processes on the master side.

If installing the slave side then Shutdown tradeengine processes on the Slave side.

Verify QA setup steps from above (as listed in this document, use the check list for verification).

Only if needed fix the setContext file in /sbt/prod/tradeeng directory. After you login, if the setContext version has changed then the setContext is not run and you will not be able to start any process. In this case just copy the /sbt/prod/tradeeng/CBOEDIR_8.0 /setContext.template file into /sbt/prod/tradeeng as setContext and then correct all the variables in it. You can use the old setContext file as an example to update the new file.

Change run_dir links

NOTE : Delete orun_dir and move run_dir to orun_dir. Helpdesk needs "orun_dir" to look at old log files.

Change the run_dir link in tradeeng to point to the new release CBOEDIR_8.0

Logout and log back in as tradengp.

Do any database conversions if needed..

Have operations bring up tradeengine processes using PATROL

If this is the slave box then failover and run thru the verification steps listed below.

If you are installing the master side then run **`linkageExternalServices start`** to verify if the connections are established correctly.

Also use the OLA Fixometer to connect to OCC to verify if all connectivity is OK.

Note	Some times because TPF is down and OCC is down you may be unable to connect to the external systems, in this case just verify in the log files that we have made an attempt and that the other systems are down at the time.

Verification

Check all files (.log, .debug, .out, .err) for errors, exceptions and high system alarms.

Make sure all initialization is complete on all processes.

If you are installing the slave side box then installation is complete. (Just make sure operations runs the slave box in master mode the next day).

When installing the Master side run through the procedure on "How to check Linkage processes using Test Orders". Follow the procedures as documented in the "Linkage Operator procedures". Basically a script "**`checkLinkageTestOrder`**" needs to be run that will test out the flow between all processes.

Verify memory usage for **all processes** on the LC's and Garbage Collection activity by comparing with the OLD and NEW .out files.