CBOEDirect Release Notes

CBOEDIR_8.3.5

April 27, 2010

Table of Contents

Overview of changes in this release

Work requests/Interims included in this release

6410 – SPREAD PROCESSING OPTIMIZATION

6674 – (interim)PERSIST STOCK ISO ORDER WITH AWAY EXCHANGE DATA

6673 **-** (interim) CBSX MARKET ORDER PROHIBITION DURING EXTENDED TRADING HOURS

6675 – DARK MATTER – Phase 1

WR6410 – Spread Processing Optimization

Goal

The goal of the of Spread Processing Optimization project is to remove all spread processing from the "regular" spread quote and order execution paths. In high-volatility classes where there are a large number of spread products and/or a large number of resting spread orders, the response time for a simple order or a quote is significantly impacted by the cost of processing all spreads legged into the series being updated.

Summary of changes

To eliminate all latency issues derived from spread processing, in this project we are migrating all spread processing on a BC to a separate process, the SpreadTradeServer. All spread order traffic will be directed to this process. It will also listen passively to market data ("current market") from the hybrid trade servers, and calculate the derived spread market based on this data. When a spread order is determined to be marketable, it is sent to the hybrid trade server for execution against legs, and then traded against any COB orders that it crosses.

All existing spread trading rules and features, including AIM, COA, *et cetera*, will be supported. The only change to execution will be that some trading opportunities may be missed if the market "flickers" such that the spread trade server cannot respond in time to a leg market quote change.

Additionally, as it was never a supported feature of spreads, quoting will not be supported on the spread products in the SpreadTradeServer.

Interim 6673 – CBSX Market Order Prohibition During Extended Trading Hours

Goal

Currently CBSX accepts Market Orders during the extended trading hours. This is a rule violation and   OES compliance department already notified CBSX in this regard. This release will fix this issue. With this release, any market orders that CBSX receives outside of the regular market hours will be rejected unless it has a cross_type contingency.

Summary of changes

a) The following two trading properties were added as part of the change:

i) RegularMarketHours

This property defines the regular market hours using two fields: openTime and closeTime. These two fields can be specified as HH24:MI:SS format or as TimeInMillis since midnight.

When openTime is less than closeTime (e.g. 8:30:00-15:00:00), the regular market hours is greater than or equal to the openTime <u>AND</u> less than or equal to the closeTime. When openTime is greater than closeTime (an ATG scenario; e.g. 20:00:00-18:00:00), the regular market hours is greater than or equal to the the openTime <u>OR</u> less than or equal to the closeTime. If openTime and closeTime are the same, it is considered as a 24 hour market day. Default value will be set to 0-0 (12am-12am).

ii) EnableRegularMarketTimeValidation

This boolean property can be used to turn on or off the regular market hour validation. This will be set to true only for the W_STOCK session. By default, this will be false.

b) Order Validation for the Stock session will do the following additional validation:

i) If the order is a Market Order and if it is not one of the cross types (MIDPOINT_CROSS, BID_PEG_CROSS and OFFER_PEG_CROSS), it goes to the validation for the regular market hours in the next step.

ii) If the EnableRegularMarketTimeValidation is set to true and if the current time of the day falls <u>outside</u> of the regular market hours defined through the properties mentioned above, the order will be rejected with a DataValidationException. The message of the exception will be: "*Invalid order. Market orders (other than valid cross types) are not accepted outside regular market hours.*" and the error code will be *INVALID_STATE*.

c) Support was added to get and update the above two properties through the AR commands. When setting the time range for the RegularMarketHours through ar command, specify that either as time(e.g. "8:30:00-15:00:00") or as timeInMillis since midnight ("30600000-54000000").

Interim 6674 – Persist Stock ISO Order With Away Exchange Data

Goal

Persist Montage and order relationship for Stock linkage orders.

Summary of changes

a) Persist montage data for each exchange with 315 event type for underlying orders that needs linking. So if there are 5 exchanges in montage, there will be 5 entries with type 315.

b) Persist montage data for underlying order with LINKAGE ORDER RELATIONSHIP (311)

c) Persist montage data for the linkage order generated.

WR 6675 – Dark Matter (phase 1)

Goal

The goal of the Dark Matter project is to reduce the frequency of object allocations in both the trade engines and the OHS processes. This in turn is expected reduce the frequency of garbage collections, and thereby reduce the number of user transactions impacted by the pauses incurred by these activities. By altering implementations, introducing object pools and flyweight implementations, and eliminating redundant memory usage, we expect in phase 1 of this project to have an appreciable difference in the object allocation patterns.

Summary of changes

The following summarizes the changes made in this project. They are listed here for reference, as they mostly describe very low-level changes in the code base.

Trade Server

BrokerCommand.execute(), MarketMakerQuoteServiceLocalImpl.createNewQuote(),TradingClassBlockCommand.execute()

- Eliminated call to Class.getSimpleName(), which is non-trivial and allocates a String per occurrence.

BrokerImpl.acceptCommands()

Converted enhanced for(:) syntax to for (;;) syntax to avoid creating list iterators.

BrokerProcessorHybridImpl.getAuctionProcessor(Integer)

Altered method implementation to accept a native int param to avoid the creation of Integer objects.

CurrentMarketPublisher.acceptCurrentMarketsForClass()

Eliminated the creation of a new NBBOStruct[0] on every publish.

QuoteHistoryCodecUtil.getDayOfWeek(...)

Replaced calls to this method with calls to the more efficient getCurrentDayOfWeek().

QuoteHomeImpl

Removed call to makePersistent(), since it is not needed for quotes. Changed to use a constant (-1) object id forQuoteImpl. Removed String.valueOf(int) in create() method (passed toQuoteImpl.setName()). Replaced Calendar, StringBuffer, buffer appends, toString with currentTimeMillis % (24*3600*1000), saving the allocation of many objects per quote on first quote.

QuoteImpl.transactionSequenceNumber

Made this field non-reflective, non-transactional.

QuoteStruct, QuoteStructV3

Restored -D to enable caching.

TradingClassBlockCommand

Eliminated cloning the ArrayList of Command objects being cloned before being executed.

TradingSessionNameHelper.getOpenConfiguredSessionName()

Eliminated call to getTradingSessions(): replaced with call to getSessionList().values()

OHS

BaseOHSContext.getOrderHistoryCommonInfo(int)

Introduced a pre-allocated QuoteQueryV3Struct for ‘null’ history entries.

DateStructDefaultFactory.create()

Added -D's to OHServer to enable use of DateStruct flyweight factory.

ExchangeFirmStructDefaultFactory.create()

Added -D's to OHServer to enable use of ExchangeFirmStruct flyweight factory.

HistoryQueueReader.run(), OrdersQueueReader.run(), TSBQueueReader.run()

Allocated ArrayList outside of the “read loop” and re-use it, rather than allocate per read operation.

OrderId.getOrderIdsForLocking()

Replaced this method with two calls, one to return a long (primitive) and another with the order id string. - The order id string should now be stored in theOrderId, based on the OrderLockManagerImpl changes recommended above.

PriceSqlType

Implemented a PriceSqlType factory to compliment the PriceFactory. Replaced all in-path calls to newPriceSqlType(...) with calls to the PriceSqlTypeFactory.

PriceStructDefaultFactory.create()

Added -D's to OHServer to enable use of PriceStruct flyweight factory.

SQLOrderInsert.bindValues(), SQLOrderHistoryInsert.bindValues()

Created the StringBuffer with larger initial capacity (columnStrings.length()+typicalStringLength). Eliminated all calls to SQLPrepStatementUtility.bindAsStringValue().

ValuedPrice

Added ValuedPrice string value to the PriceFactory, and invoke the initialization of these strings at factory initialization time.

Bug fixes included in this release

SEDL	PITS	Test Plan	Assigned	Date Merged	Description
8684	23384	Yes	Mike H	6/17/2010	Returning linkage orders not expiring auctions
8709	22049	Yes	Jay	6/3/2010	Tied cross orders not displaying in HDOMT unless queried for.
8734	23170	n/a	Francisco	5/10/2010	Market Data Buffer excessive logging to be removed
8751	23272	n/a	Anil	5/10/2010	JVM setting on MDBAdapter change for long remarks
8791	18046	n/a	Anil	5/10/2010	JVM settings on Status Server
8803	26711	n/a	Jay	5/10/2010	Remove Production Linkage Default Ping Old router setting
8709	22049	Yes	GUI	6/3/2010	CBSX Tied Cross Within & Cross Within orders

					are not displaying in the SA GUI or Trader GUI, despite being accepted and processed correctly in server
8790	26268	n/a	Anil	6/1/2010	User Cache Proxy Hashmap unsafe
8900	32751	Yes	Hari	6/21/2010	SAL’d order traded through limit
8909	34174	n/a	Dalji	7/15/2010	Inconsistent return code to user when quote in block fails to update market
8921	35112	Yes	Sri	6/7/2010	publishUackedTradeReport NPE exception
8950	37029	Yes	Connie	6/14/2010	F and B origins are able to start SAL **(Note:** already minor’d to production BC97 during week ok 6/14**)**
	23524	Yes	Esquivel	3/19/2010	Sweep and Sweep Book trades reported to STOPP including F in sale condition field

Outage related enhancements

Outlier related enhancements

Performance related enhancements

Configuration and development/performance testing related enhancements

Other enhancements included in this release

Prerequisite

Make sure QPE_FLAG (known as OHS Indicator) is off for all classes.

In this release, we are using prod_class QPE_FLAG field for class by class rollout to Spread Trade Server processing. This requires to set all the classes to be false instead of true prior to rollout of 8.3.5 global if they are not already in production.

Run following command to turn off QPE_FLAG after server is up with 8.3.5:

ar GlobalServer rolloutDisable -all

ITG checkout scripts

We will be retiring the ar command to set OHS indicators as the field is no longer used for OHS rollout. ITG checkout scripts that has the changes to no longer calling these OHS indicator commands will need to be installed to production prior to the rollout.

Implementation Plan 8.3.5 for C1

GC Database changes

none

BC Database changes

none

Installation procedures GC02A/B

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_8.3.5/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.

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

4. Install new software CBOEDIR_8.3.5 release,
5. Update setContext in home directory with changes if any.
6. Change run_dir links and log back in.
7. Do any database conversions if needed.
8. Start the GC process.
9. Master side only. Create Spread Trade Server routing group

   configureClusterGroupsForSTS Prod
10. Add new trading properties and event channel if any:
    - TradingPropertyServiceClient setPropertyDefinitions ${RUN_DIR}/properties/TradingPropertyDefinitions.csv
11. Master side only. Run the script ***addRegularMarketTimeValidationProperties*** to set the two new properties: *RegularMarketHours* and *EnableRegularMarketTimeValidation* for different sessions.
12. **Run following command to turn off QPE_FLAG after server is up with 8.3.5:**

ar GlobalServer rolloutDisable -all

To verify: loginDb GLOBAL

Select count(*), qpe_flag from prod_class where prod_type_code in (7, 11) group by qpe_flag;

You should see something like this indicating qpe_flag is OFF or null for all 7 and 11 product classes. You should not see any count >0 where QPE_FLAG is 1

```
SQL> select count(*), qpe_flag from prod_class where prod_type_code in (7, 11)
group by qpe_flag;

COUNT(*) Q
---------- -
2
6178 0
```

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. The following verification are for CBSX Market Order Prohibition:
    i. Using getTradingProperty (*getTradingProperty <session> default <propertyName>*) verify the following:
        1. ***RegularMarketHours*** is set to **0:0:0 – 0:0:0** for all sessions except W_STOCK. For W_STOCK, it should be **8:30:0-15:0:0**.
        2. ***EnableRegularMarketTimeValidation***is set to **false** for all sessions except W_STOCK. For W_STOCK, it should be **true**.
18. 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)
19. 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).
20. Do test trades using quotes/orders on at least one BC for all sessions.
21. For Futures TradesServers (TradeServer3) – You can use the same user to enter Orders/Quotes.
22. For Hybrid TradeServers (you will need to use quotes to trade with each other) + Enter a couple of orders.
23. 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.
24. 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.
25. Verify that Order Status messages (New Order, Order filled are showing up on the Status window screen)
26. Check all files on prdgc01 (.log, .debug, .out, .err) for errors, exception’s and high system alarms.

- **GC - Saturday verification after upgrade**

CBOEDirect verification after Global Cluster upgrade

#	Description	PASS/FAIL
	Perf truncate should not be run on BC97	
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.	
8.5	*** Special Test *** Verify in BC97 database there are resting SPX GTC orders. Failover BC97. **NOTE** do not suspend or halt classes before failing over. After failover put the OHS in debug: ar OHServerHybrid setLogLevel ALL DEBUG Page FIX support (Arun: 847-530-3493). Support will login a user (with GTC's) on FIX 50 engine 4, verify orders are received and canceled.	
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	Repear 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	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)	
28	Start XTP Replay (1.5 rate(. Open the *Market Display For Underlying* window and verify ticker and recap. *Special Test* Note if broker on mdgc01 is discarding – if so then skip test 38	
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.	

37.5	***Special Test*** Failback mdgc01. Reset to “old config”. Run xtp replay @1.5 rate and verify discarding happens.	**Skip**
38	Failback GC01 to CD 8.3 codebase.	
***	Set GC flag for OSI testing on. Bounce GC01 process. Begin OSI testing. Page/notify Client Support (FIX Support) to run OSI FIX tests on fix 50.01. Begin OSI Backoffice testing After OSI testing: Set GC flag for OSI testing OFF (Server Support). Set FIX OSI flag OFF (FIX Support).	
39	Close all products and run End Of Sale and updateClose after all other tests have completed.	
40	Run and verify any **new** end of day procedures as listed in the operator procedures	

Fallback

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

Installation procedure for the 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. There are no BC table changes for this load.
3. Shutdown tradeengine on the Master BC (tradengp and tradengh [tradengh is only for W_MAIN] login).
4. Install new software CBOEDIR_8.3.5 release, change run_dir links .
5. update setContext as follows **for C1 HYBRID BC only. (Don't do this for C2 as there is no SpreadTradeServer for C2 environments)**
   - Add line **only if is SLAVE BC. DO NOT ADD this if is MASTER BC.**

     export STS_MODE=DISABLED

   Do the following ONLY on BOTH MASTER and SLAVE SIDE when both side has upgraded with new code base so we will have STS started both side. Otherwise will cause failover issues.

   - Add ***SpreadTradeServer1*** to ALL_SERVER_NAMES.

     Please note, there will be only 1 Spread Trade Server on each Hybrid BC.

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

- **Evening Installation Procedures – Verification during rollout**

8. Check all files (.log, .debug, .out, .err) for errors, exceptions and high system alarms.
9. 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.
10. 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).
11. Verify on prdgc01a/b that there are no products in NO_SESSION state.

12. **This step applies only if you are installing a Stock BC.**

    (*These steps may be skipped if those are included in the checkout)*

    Bring up a Trader GUI and open a test stock class on the Market Display Window and perform the following steps. Set the properties *RegularMarketHours* and *EnableRegularMarketTimeValidation* (using SAGUI or using *setTradingProperty*) to appropriate values as required in the steps:

    i. Set RegularMarketHours to outside of current time and set EnableRegularMarketTimeValidation to true.
        1. Enter a regular Market Order.
            a. Verify the order gets rejected with the message: "*Invalid order. Market orders (other than valid cross types) are not accepted outside regular market hours.*"
        2. Enter Market Orders with contingencies: MIDPOINT_CROSS, BID_PEG_CROSS and OFFER_PEG_CROSS
            a. Verify those orders accepted rejected
        3. Enter a Limit Order
            a. Verify the order is NOT rejected

    ii. Set EnableRegularMarketTimeValidation to false, keeping the RegularMarketHours still outside of current time.
        1. Enter a regular Market Order.
            a. Verify the order is NOT rejected.

    iii. Set EnableRegularMarketTimeValidation to true and change the RegularMarketHours window to includes the current time.
        1. Enter a regular Market Order.
            a. Verify the order is NOT rejected.

13. **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

14. Run ITG Checkouts.
15. 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.
16. This instruction only applies if you are installing a W_MAIN BC. Repeat step 11.
17. Check all files on the affected BC's (.log, .debug, .out, .err) for errors, exceptions and high system alarms.

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**

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

- **Fallback**

1. See operator procedures on how to failover BC's or list instructions specific to your release here.
2. For Stock BC, if we need to turn off the Market Order Prohibition during extended hours, set the property *EnableRegularMarketTimeValidation* to false (and/or set *RegularMarketHours* to 0-0).

- **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. 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. **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. OPS should verify: **CTMr** – verify there are no un-acked trade reports. See Operational procedures. **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.
8	Repeat step 6 to verify counts increase.
9	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.
10	**BC Failover test (BC 28 & BC 9)**
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.
	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
15	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 (if it is stayin in production service) and run ITG checkouts on all 4 trade servers for the new BC. Backout any BC that is not staying in for Monday.
19	Close the products, End the sessions and verfy that all the sessions have ended successfully.

Installation procedure for the FE

- **QA steps**

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

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

Note: FE is installed using script setContext.main(vobs\server\properties\). In 8.3 this script generates setenv(for GC, BC and FE) with jdk1.6 update 18. For 8.3.5 FE Rollout FE must use jdk1.6 update 14. Manullay rollback this change from setenv file to use jdk1.6 update 14.

```
export JAVA_HOME=/usr/local/jdk1.6.0_14
```

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

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

(SACAS should already be configured to use JDK 1.6.0_14).

GUI group steps

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

Verification

Use new SAGUI to log on and ensure that it receives data from SACAS. In particular, perform the following:

1. Bring up a new SAGUI from this release. Go to Trading Properties window and verify we see the following two properties:
    a. Regular Market Hour Validation Enable
    b. Regular Market Hours
2. Try setting the above properties and verify they are getting set correctly. Reset the values back to the original settings.

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.

On installation day after end of trading, install new configureCAS.ksh onto each CAS box being upgraded.

Client group steps

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.

On installation day after end of trading, install new configureCAS.ksh onto each FIXCAS box being upgraded.

Client group steps

???

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.

On installation day after end of trading, install new configureCAS.ksh onto each MDCAS box being upgraded.

Client group steps

???

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.

On installation day after end of trading, install new configureCAS.ksh onto each CFIX box being upgraded.

Client group steps

???

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 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_8.3.5 /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_8.3.5
>
> 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.

Strategy Class Rollout to Spread Trade Server and Fallback Procedures

Overview

Here is the summary for different stages of rollout in RoutingGroupAssigment properties file. Assuming IBM(11, 7) has rollout flag (QPE indicator) on. Minors will need to be done on Global to change the setting after rollout is completed to turn off the checking of rollout flags.

	checkRolloutFlag	include ProdTypes	AutoAssignStrategies	Example	Rollout Status
HTS	yes	Options	true	IBM(11, 7), GE(11, 7)	Initial Rollout
STS	yes	Strategy	false	<empty>	Initial Rollout
HTS	yes	Options	true	GE(11, 7), IBM(7)	Rolling out
STS	yes	Strategy	false	IBM(11)	Rolling out
HTS	no	Options	false	IBM(7), GE(7)	Roll Complete
STS	no	Strategy	false	IBM(11),GE(11)	Roll Complete

Day 1

Spread server functionality not available.

Both Master/Slave side will rollout and Spread Trade Server can be started on both side. However, Slave side STS will be running in DISABLED mode (setContext setting) and routing group assignment will not assign any classes to STS. No strategy classes will have rollout indicator on.

Failover: follows the same procedure that we have currently in production.

Day 1: Spread server functionality not available					
Day	QPE Flag	Spread Server Rollout Flag	Server	Auto Assign Classes	Example: Migration of class IBM.

1	Null or 0	0	STS	No strategies assigned to STS.	STS is not in path of strategy processing as all service offers from STS have been removed (at the end of day 1 as a result of STS turn off) or are not present as STS has not been rolled out
			HTS	Strategies and options are assigned to HTS	Current production scenario

Day 2

Rolling Out Classes to STS.

The day before rolling out, we need to have IPD to change the rollout indicator for the strategy class and its options class to be enabled. During this phase of rollout out, only the rollout classes (QPE rollout indicator = true) will be routed to STS in the routing group table.

Failover with STS Disabled to have HTS process strategies assigned to STS

During this phase of rolling out, only the rollout strategy classes (QPE indicator = true) will be routed to STS in the routing group table. If failover is due to critical errors in STS, we need to go back using HTS to process classes enabled for STS.

- Make sure STS is running in DISABLED mode. Check setContext in home directory for STS_MODE. If not in this mode, change the setting to be DISABLED, then restart STS in SLAVE before next step.
- Run command *prepareForSTSDisabledModeFailover.* This command has to be done prior to the initiation of go master on all processes on this BC.
- Start BC failover to go master command.

Day	QPE Flag	Spread Server Rollout Flag	Server	Auto Assign Classes	Example: Migration of class IBM.
2	1	1	STS	Yes – Strategies only.	IBM (11) will be assigned to STS
				Option classes specified in buddy group	IBM (7) will be loaded as proxies
			HTS	Yes – Remove strategies from assignment.	IBM (11) will be removed from HTS group during class assignment.

Day	QPE Flag	Spread Server Rollout Flag	Server	Auto Assign Classes	Example: Migration of class IBM.
				Assign option classes only.	IBM (11) will be loaded as proxies.
				Strategy classes specified in buddy group	IBM (7) will be created as normal objects.

Scenario: Spread server functionality rolled back for some reason during the day

Day	QPE Flag	Spread Server Rollout Flag	Server	Auto Assign Classes	Example: Migration of class IBM.
Fail-over	1	0	STS	Strategies already assigned at start of the day	STS will route strategy calls to HTS.
			HTS	Strategies not assigned at start of day.	HTS will accept calls for strategy products for the rest of the day.
				Option classes specified in buddy group.	During fail over HTS will create *normal* objects for strategy classes.

Day 3

Both STS enabled on slave and master side.

Minor to set slave side STS_MODE in setContext to be true.

Failover with STS Enabled: For failover situations when there is hardware problems, or JVM errors on the master side, just make sure STS is running in ENABLED mode on slave side before goMaster command is issued. Follow normal failover procedures.

Failover if SpreadTradeServer has problems processing

Follow steps for day 3: Failover with STS Disabled to have HTS process strategies assigned to STS

Day 4

All strategy classes enabled for Spread Trade Server processing.

GlobalServer minor on RoutingGroupAssignment.properties for production to update checkRolloutFlag and autoAssignStrategyClasses to false for HTS and STS.

Subsequence minor/project can be done on the BC in setContext.main to change spreadTradeServerRolloutCompleted to true so code will no longer checking QPE rollout indicator can hence can be used for other projects for other purposes.

Failover with STS Enabled: For failover situations when there is hardware problems, or JVM errors on the master side, just make sure STS is running in ENABLED mode on slave side before goMaster command is issued. Follow normal failover procedures.

Scenario: Spread server functionality rolled out completely

Day	QPE Flag	Spread Server Rollout Flag	Server	Auto Assign Classes	Example: Migration of class IBM.
4	Null or 0	1	STS	Strategies assigned to STS.	IBM (11) will be assigned to STS.
				Options specified in buddy group.	IBM (7) will be loaded as proxy objects.
			HTS	Options assigned to HTS.	IBM (11) will be removed from HTS group during class assignment and will be loaded as proxy objects.
				Strategies specified in buddy group.	
					IBM (7) option classes will be loaded as normal.

Spread Trade Server Mode

We introduced 2 modes for Spread Trade Server in order to handle failover situations during rollout. setContext (export STS_MODE=)in the $HOME directory will have the setting to indicate the default mode for startup.

During rollout, the MASTER side will have the value in setContext set to be ENABLED While the Slave side boxes will have the mode set to DISABLED. Only when we are done with rolling out phase, this will be changed to ENABLED on the slave side.

1. ENABLED. STS in this mode will behave like a trade engine. STS will be responsible for all the processing of strategy related functionalities.

2. DISABLED. STS will act as a routing proxy server that can route requests to HTS to handle classes assigned to STS group that have exported services to the CAS during failover situation.

SpreadTradeServer.xml will load corresponding implementations (routing proxy if DISABLED, or real impl if ENABLED) based on the mode that is set for STS server.