

Test di performance Best-X per 1500 ordini in curando automatico

Versione: 01.00

|  |  |
| --- | --- |
| **Codice Progetto**: | bxmnt-265 |
| **Nome Progetto**: | Test di performance di BestX! per 1500 ordini in curando automatico |
| **Nome Cliente**: | SoftSolutions! |
| **File**: |  |
| **Save date**: |  |
| **Ultima stampa:** |  |

Document updates

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Release** | **Author** | **Description** |
| 8 mar 2013 | 1.0 | Paolo Midali | Document Creation |
| 12 mar 2013 | 1.1 | Davide Rossoni | Aggiunte note conclusive |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Document notes

Il presente documento viene allegato al ticket BXMNT-265.

Viene integrato da “Analisi\_tempi\_Eseguiti\_20130307.xlsx”, allegato allo stesso ticket, che dettaglia i tempi relativi agli ordini effettivamente eseguiti.

Sommario

[1. Modalità di test 4](#_Toc350861995)

[2. Richieste e configurazione effettiva 5](#_Toc350861996)

[3. Analisi dei risultati 6](#_Toc350861997)

[3.1. Utilizzo della memoria (jmx) 6](#_Toc350861998)

[3.2. Utilizzo della CPU (jmx) 8](#_Toc350861999)

[3.3. Threads - numero massimo e medio dei thread usati (jmx) 11](#_Toc350862000)

[3.4. Tempo minimo, medio e massimo di scrittura sul DB (jmx) 13](#_Toc350862001)

[3.4.1. save new order (ms) 13](#_Toc350862002)

[3.4.2. update order (ms) 13](#_Toc350862003)

[3.4.3. save new attempt (ms) 14](#_Toc350862004)

[3.4.4. save new state (ms) 15](#_Toc350862005)

[3.5. Distribuzione dei tempi 16](#_Toc350862006)

[3.6. Ritardo misurato dall'ingresso degli ordini CORP al termine della PD (log) 16](#_Toc350862007)

[3.7. Numero di PD effettuate per ogni intervallo temporale di 15 minuti e 30 minuti (jmx) 17](#_Toc350862008)

[3.8. Numero totale di PD effettuate nel corso del test (jmx) 17](#_Toc350862009)

[4. Note / osservazioni 18](#_Toc350862010)

[4.1. Ordini in Technical Error 18](#_Toc350862011)

[4.2. Salvataggio ultima PriceDiscovery per curando 18](#_Toc350862012)

[4.3. Sizing code e curandoRetry 18](#_Toc350862013)

[4.4. Restore active operation 18](#_Toc350862014)

[4.5. Thread issue 19](#_Toc350862015)

[4.6. Altre issue 20](#_Toc350862016)

[5. TODO 20](#_Toc350862017)

# Modalità di test

Il test è stato effettuato in due giornate.

La prima giornata è servita principalmente per la verifica delle configurazione e per evidenziare eventuali criticità e problemi.

Nella seconda giornata si è svolto il test vero e proprio.

Si è utilizzata una macchina virtuale con configurazione simile a quella del cliente (Credit Suisse) per l’esecuzione di BestX.

L’indirizzo della macchina, il db utilizzato e le modalità di svolgimento sono indicati nel ticket.

Si è utilizzato iSSFixTestJMX per l’invio degli ordini (sono allegati al ticket anche i due files utilizzati: <cs\_curando\_orders\_1500.log> e < cs\_executable\_orders\_500.log>, contenenti rispettivamente i 1500 ordini govies che andranno in Curando Automatico, e i 500 ordini Corporate).

# Richieste e configurazione effettiva

1. Configurare 2 code di PD, una per i GOVIES a 35 ordini al minuto e l'altra per i CORP a 30 ordini al minuto, entrambe con RTFI e BBG come mercati
   1. coda GOVIES 35/min, coda CORP 30/min
2. Usare complessivamente 20 Market Makers e almeno 20 ISIN diversi
   1. 10 MM bloomberg + 3 attivi RTFI (2 disattivi)
   2. 54 isin sui GOVIES, 54 isins sui CORP
3. Configurare il timer di sveglia del curando automatico a 60 min (in BESTX.properties impostare Order.Curando.Timeout = 3600000)
   1. Configurato a 30min per "controbilanciare" il minor numero di market makers (punto 2)
   2. Nell’ultimo test è stato configurato a 60min.
4. Mandare 1500 ordini fuori prezzo su titoli GOVIES (es, tutti in buy a 50 e/o in sell a 200)
   1. Nella prima giornata sono stati inviati a blocchi di 100, con una pausa di qualche minuto al termine di ogni blocco.
   2. Nella seconda giornata sono stati inviati senza interruzioni, con un intervallo di 2 secondi.
5. Mandare altri 500 ordini nell'arco di 1 ora su CORP, tali che almeno la metà sia eseguibile su RTFI.
   1. Sono stati inviati come richiesto in un’ora, con un intervallo di 7 secondi.

Il primo giorno ne sono stati eseguiti solo 85 a causa del Bloomberg Feed che, variando molto i prezzi, spesso è diventato best (e il sistema non chiude se la best è bloomberg)

Circa 190 sono andati in stato "non eseguito" per questo motivo.

I rimanenti (circa la metà) sono andati in Curando automatico, aggiungendosi ai GOVIES

Il secondo giorno è stato modificato il simulatore RTFI; in questo modo il prezzo RTFI è risultato sempre best, e 245 ordini sono stati eseguiti.

*TODO: verificare i 5 ordini non eseguiti.*

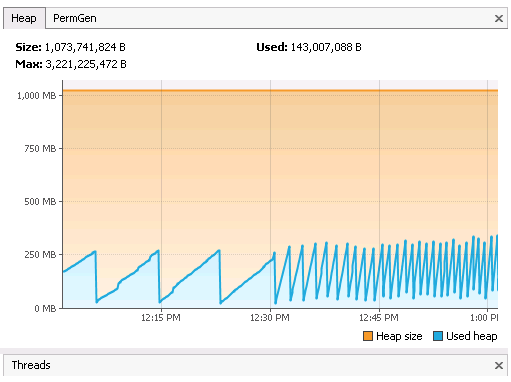
# Analisi dei risultati

## Utilizzo della memoria (jmx)

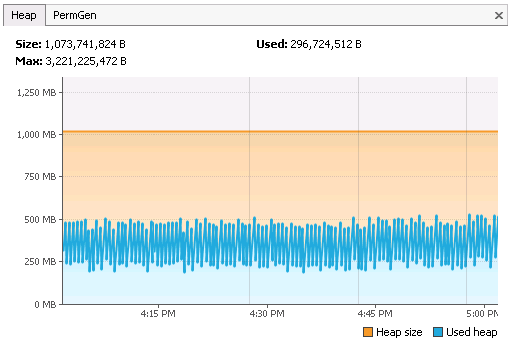
Utilizzo della memoria (jmx), sia per i thread che per gli oggetti

*[ I grafici sono un “mix” delle due giornate, essendo stato l’andamento simile ]*

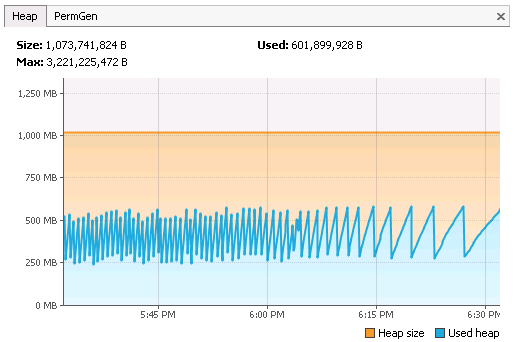
Fase di carico degli ordini:



Funzionamento a regime, con gli ordini in curando automatico:



Fase di scarico, con chiusura EndOfDay degli ordini:



La memoria aumenta in modo regolare, e si stabilizza una volta che tutti gli ordini sono stati ricevuti, oscillando in un intervallo compreso tra i 240 e i 560MB.

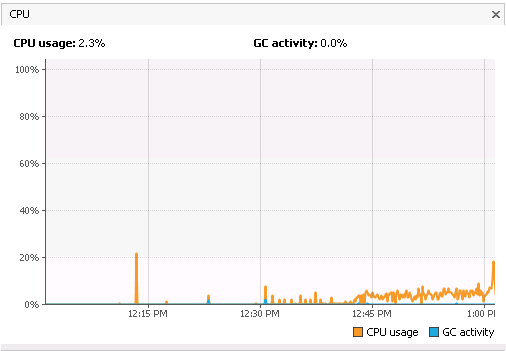
Con il crescere del numero di ordini la frequenza delle oscillazioni (dovuta all’intervento del Garbage Collector) aumenta.

A fine giornata, con la chiusura “EndOfDay” degli ordini, la frequenza diminuisce.

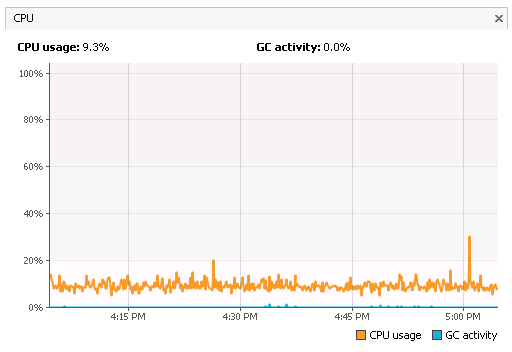
## Utilizzo della CPU (jmx)

[ Anche in questo caso i grafici provengono da entrambe le giornate ]

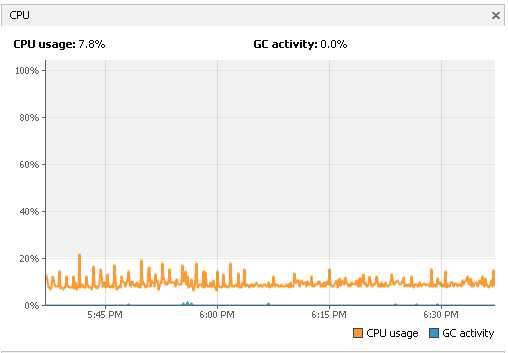
Fase di carico degli ordini:



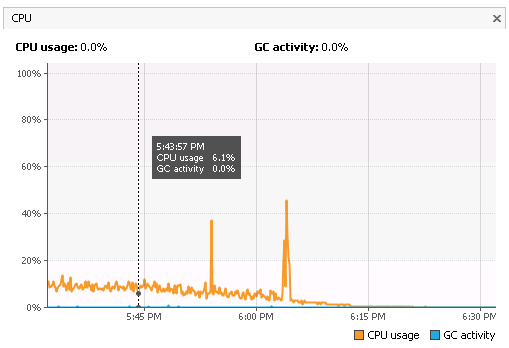
Funzionamento a regime, con gli ordini in curando automatico:



Fase di esecuzione degli ordini (parte sinistra del grafico):



Fase di scarico, con chiusura EndOfDay degli ordini:



La cpu si stabilizza su valori non elevati (7 / 8% ).

Ha oscillazioni maggiori durante la fase di esecuzione degli ordini, ma sempre mantenendosi al di sotto del 10%.

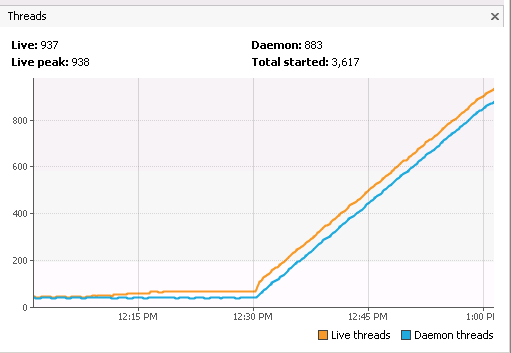
A fine giornata, con la chiusura “EndOfDay” degli ordini, ritorna verso lo 0.

I picchi sembrano più problemi del grafico che valori effettivi.

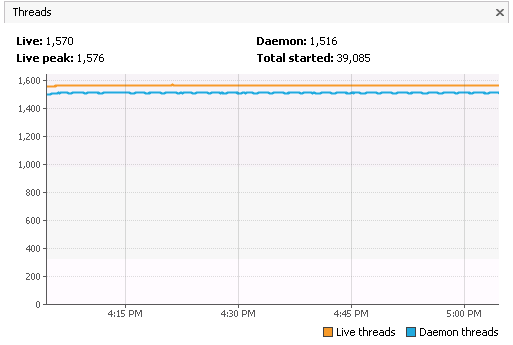
## Threads - numero massimo e medio dei thread usati (jmx)

[ Grafici da entrambe le giornate ]

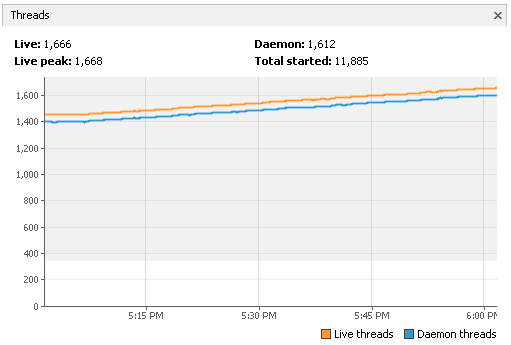
Fase di carico degli ordini:



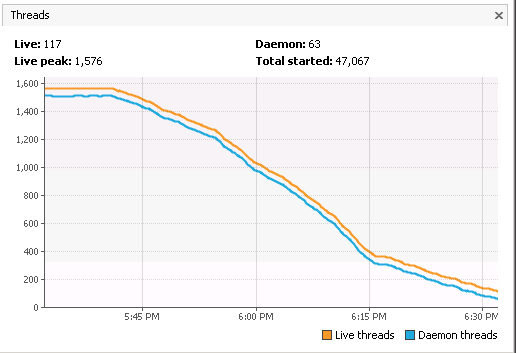
Funzionamento a regime, con gli ordini in curando automatico:



Fase di esecuzione degli ordini:



Fase di scarico, con chiusura EndOfDay degli ordini:



L’andamento è regolare, con un nuovo thread per ogni ordine.

Il rilascio a fine giornata ha un andamento corretto.

## Tempo minimo, medio e massimo di scrittura sul DB (jmx)

I tempi sono stati ricavati dai files di log del secondo giorno.

Sono suddivisi per categoria.

### save new order (ms)

n. tempi: 2000

min: 0

max: 1297

avg: 5.7

Distribuzione:



### update order (ms)

n. tempi: 34856

min: 0

max: 1953

avg: 7.45

Distribuzione:



### save new attempt (ms)

n. tempi: 2245

min: 0

max: 1719

avg: 4.69

Distribuzione:



### save new state (ms)

n. tempi: 20838

min: 0

max: 1922

avg: 3.1

Distribuzione:



## Distribuzione dei tempi

La distribuzione dei vari tempi è molto simile.

I picchi sono distribuiti in modo abbastanza uniforme, con una lieve prevalenza tra le 18:30 e le 20:00.

SI tratta di una fase in cui tutti gli ordini sono già stati ricevuti e sono a regime, per cui non sembra esserci un motivo specifico.

La quasi totalità (99%) delle operazioni su database si conclude comunque entro 25-50 ms, e l’85% sono praticamente istantanee.



## Ritardo misurato dall'ingresso degli ordini CORP al termine della PD (log)

Misure dettagliate delle tempistiche delle varie fasi degli ordini CORP (quelli per cui è stata possibile l’esecuzione) sono evidenziate nel foglio excel allegato al ticket.

In linea di massima, comunque, il tempo medio risulta di 358 msec. Il tempo minimo è di 125 msec, il massimo di 2454 msec.

2 ordini superano i 2 secondi, e 11 sono compresi tra 1 e 2 secondi.

La distribuzione degli ordini più lenti è abbastanza uniforme.

## Numero di PD effettuate per ogni intervallo temporale di 15 minuti e 30 minuti (jmx)

Sono indicati il numero di price discoveries/min.

La media risulta essere di 34.9 / sec.

*TODO: perché ci sono oscillazioni?*



## Numero totale di PD effettuate nel corso del test (jmx)

Il numero totale è di 11073

# Note / osservazioni

## Ordini in Technical Error

Alcuni ordini vanno in technical error per timeout su waiting price;

* il primo giorno sono 11. La differenza è probabilmente dovuta al fatto che il primo giorno gli ordini sono stati inseriti a blocchi di 100,
* il secondo giorno tutti i 1500 ordini sono stati inseriti sequenzialmente, a 2 secondi di distanza, e gli ordini che vanno in tecnica issue sono 256; si decide di riavviare il sistema, portando il timeout di retry per il curando automatico da 30 a 60 minuti, per sincronizzarlo meglio con la velocità delle code. In questo secondo tentativo, con la **velocità delle code** adeguata al **tempo di curando automatico**, nessun ordine va in tecnical issue

## Salvataggio ultima PriceDiscovery per curando

Per gli ordini in curando automatico non ci sono indicazioni dei nuovi tentativi, quindi non c’è nessuna indicazione visiva del fatto che l’ordine è attivo. Si potrebbe (ipotesi) salvare l’ultima price discovery per ognuno degli ordini, in modo da averne una visualizzazione nella gui, e poi sovrascriverla con la successiva, o rimuoverla se viene eseguito.

## Sizing code e curandoRetry

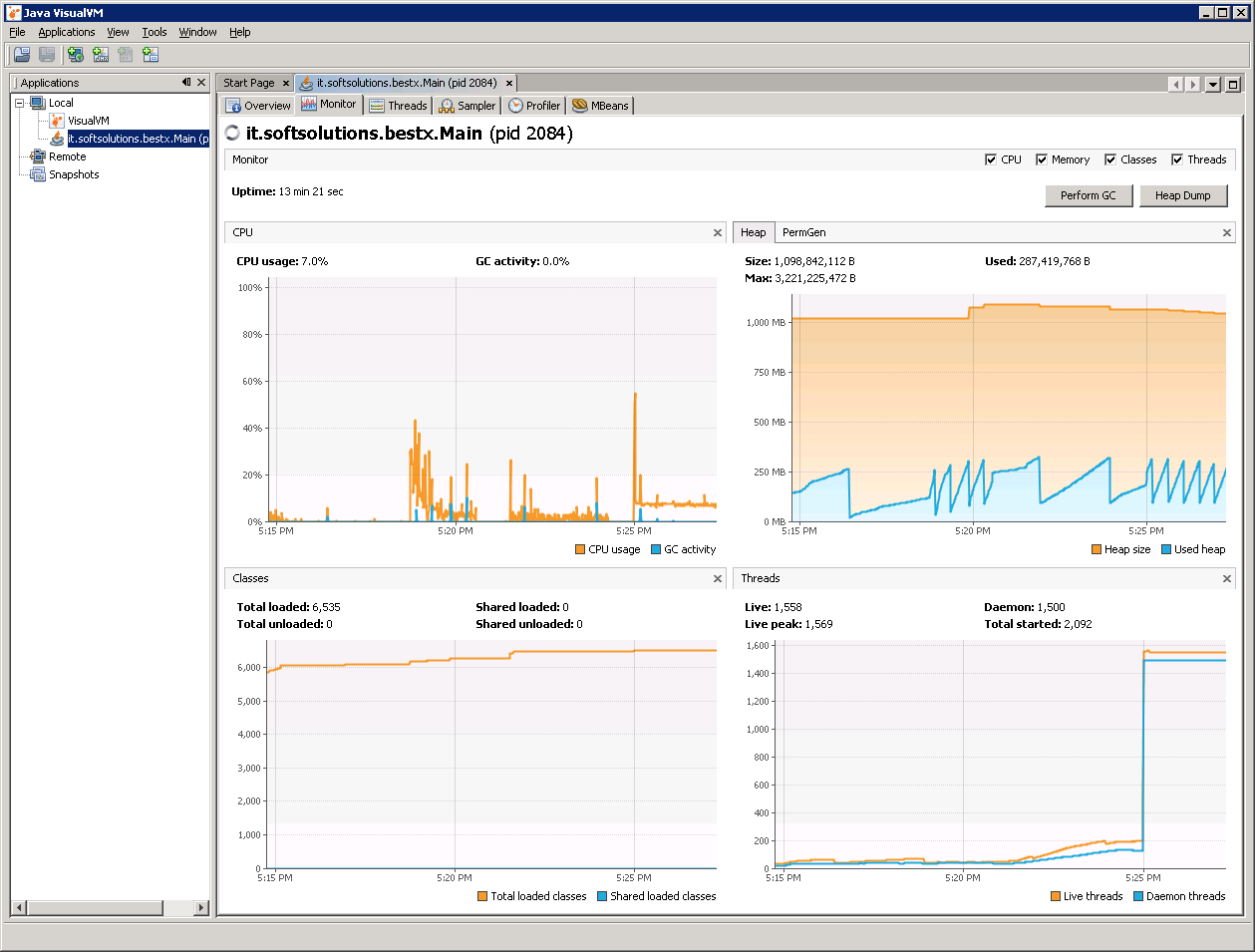
Occorre prestare molta attenzione al dimensionamento delle code di PD e del timeout sul curandoRetry: definendo una velocità delle code troppo bassa avremo ordini in TechnicalIssue, la stessa cosa accadrebbe impostando un valore basso per il curandoRetry.

## Restore active operation

Dal grafico della restore si evidenzia che tutti gli ordini vengono caricati e riattivati nello stesso momento, causando un picco di attività; questo probabilmente porterà anche a far scattare nuovamente in contemporanea i timer di tutti gli ordini in curando automatico.

Sarebbe meglio temporizzare la riattivazione degli ordini, in modo da distribuire la ripartenza e da non interferire troppo con l’operatività di BestX.

Altra ipotesi è quella di fare prima la restore degli ordini attivi (non curando) e solo in una seconda fase far partire un processo disaccoppiato per la restore dei curando.



Alcuni ordini (CORP, eseguibili) inviati 5 minuti dopo la restore vengono comunque eseguiti senza problemi.

## Thread issue

Se durante la fase di price discovery c’è qualche eccezione, ad esempio nel caso di mercato non connesso, per ogni ordine ogni tentativo di PD apre un nuovo thread ed il precedente non muore. Avremo così 1500 nuovi threads ogni 60 min, collassando il sistema.

È il modello di BestX ad essere sbagliato. Non si deve creare un nuovo thread ad ogni tentativo di PD, occorre definire un nuovo modello che non tenga attivi tutti questi thread.

Un paio di ipotesi:

* Definire un pool di thread, un **ExecutorService** a cui demandare il processing dell’ordine, invocando l’ExecutionService solo per il tempo strettamente necessario
* Utilizzare **Quartz** convertendo i thread sugli ordini come task da schedulare

## Altre issue

* **Ordini scompaiono** - alcuni ordini (es. A1002669448582 del 6 marzo) scompaiono dalla gui in determinati stati, oppure non hanno la descrizione dello stato in cui si trovano
* **Log files** - i files di log diventano difficilmente leggibili con 1500 ordini attivi contemporaneamente

# TODO

* statistica su tempo price discovery dei govies, vedere se e quanto aumenta con l’aumentare degli ordini in curando automatico – è più difficile da ottenere, perché nel file delle statistiche non sono presenti i dati necessari (si fa riferimento all’inizio dell’esecuzione, disponibile solamente per gli ordini che vengono eseguiti).
* analisi con profiler su macchina di sviluppo
* verificare il motivo per cui sono stati eseguiti 245 anziché 250 ordini corporate
* capire e stimare interventi per rendere “1500 curando” un prodotto certificato

End of Document