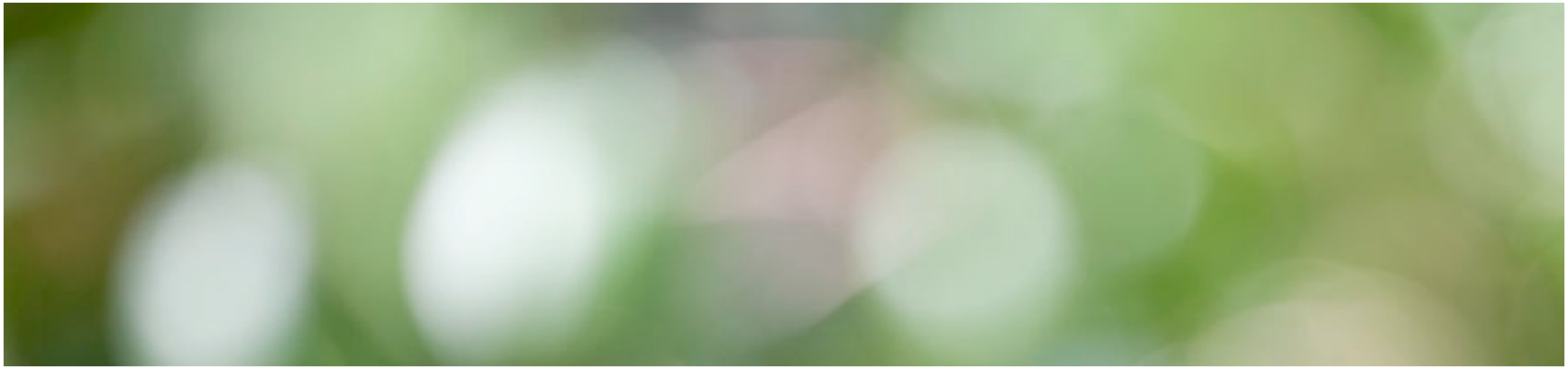
TERRACOTTA



Boost App Performance and Monitor Ehcache

Greg Luck, Founder and CTO Ehcache, Terracotta
http://gregluck.com
http://twitter.com/gregrluck

March 9, 2010

Agenda

- **Intro to Ehcache and Terracotta**
- **Code: Scaling Spring Pet Clinic**
  - With Hibernate
  - With JDBC direct
- **Comparative Performance Testing Results**
  - Database
  - Ehcache EX
  - Memcached
  - Well-known IMDG
- **Ehcache 2.0**
- **Monitoring Tools**

My own Future Predictions

My own Future Predictions

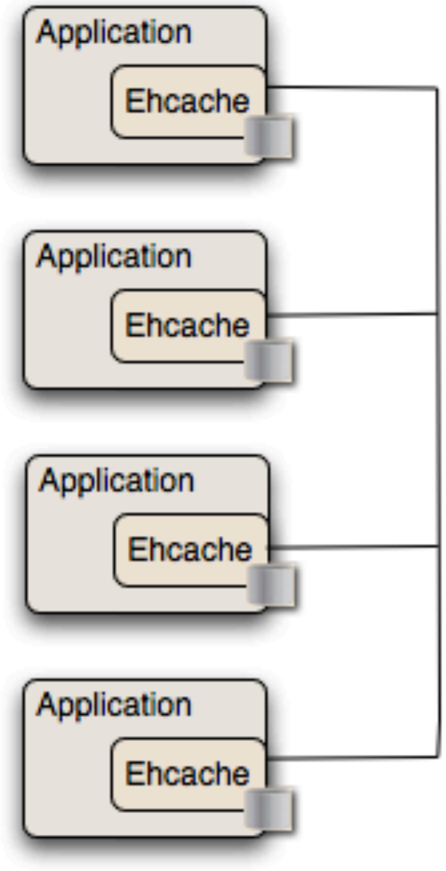
- Developers will not need to deal with the new challenges of highly parallelised CPUs - libraries will and are e.g. JDK, Ehcache
- Maven to become the majority approach
- Few developers are truly comfortable or productive in multiple languages at the same time (besides we already need to know HTML, CSS, JS, Java, XML etc.) so projects done in one will be the norm
- JVM based languages to hold sway with Java Devs rather than
- Groovy/Grails and Scala likely to be the most popular alternate JVM languages
- Sun gave it all away before they were gobbled up by Oracle - the open source community will self heal if harmed
- Virtualisation the norm, possibly abstracted through Cloud tools
- Optimism
- If Global Warming happens people will want to move to NZ

About Ehcache

The world's most widely used Java cache

- Founded in 2003
- Apache 2.0 License
- Integrated by lots of projects, products
- Hibernate Provider implemented 2003
- Web Caching 2004
- Distributed Caching 2006
- Greg Luck becomes co-spec lead of JSR107
- JCACHE (JSR107) implementation 2007
- REST and SOAP APIs 2008
- SourceForge Project of the Month March 2009
- Acquired by Terracotta 2009
- Integration with Terracotta Server Array 2009
- Ehcache 2.0 (with 3rd revision of Terracotta Integration) March 2010

Ehcache before Terracotta



RMI
JGroups
JMS

Up to 8GB in-process

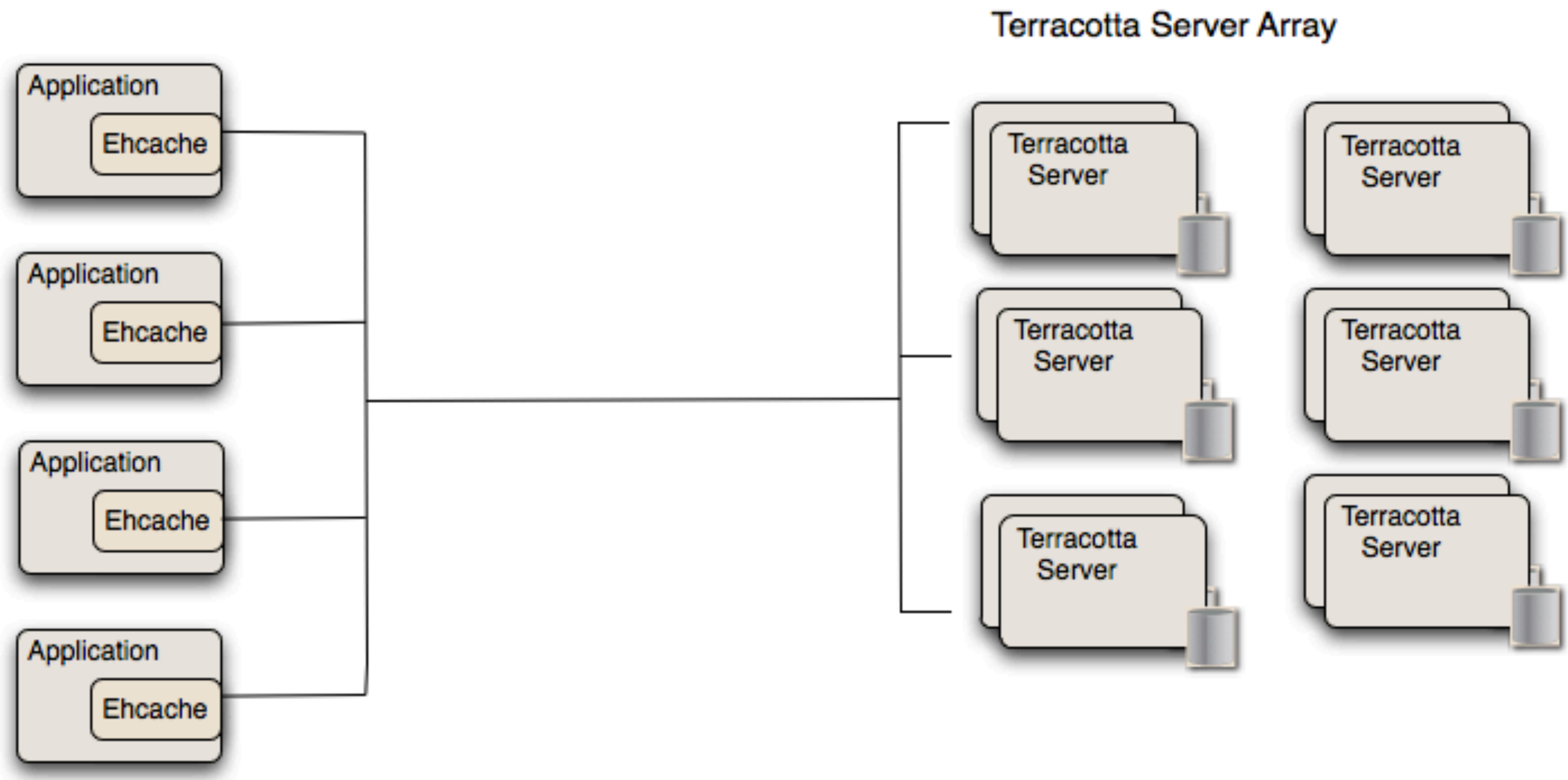
Up to 20GB on disk

Replicated Distribution up to 20 nodes

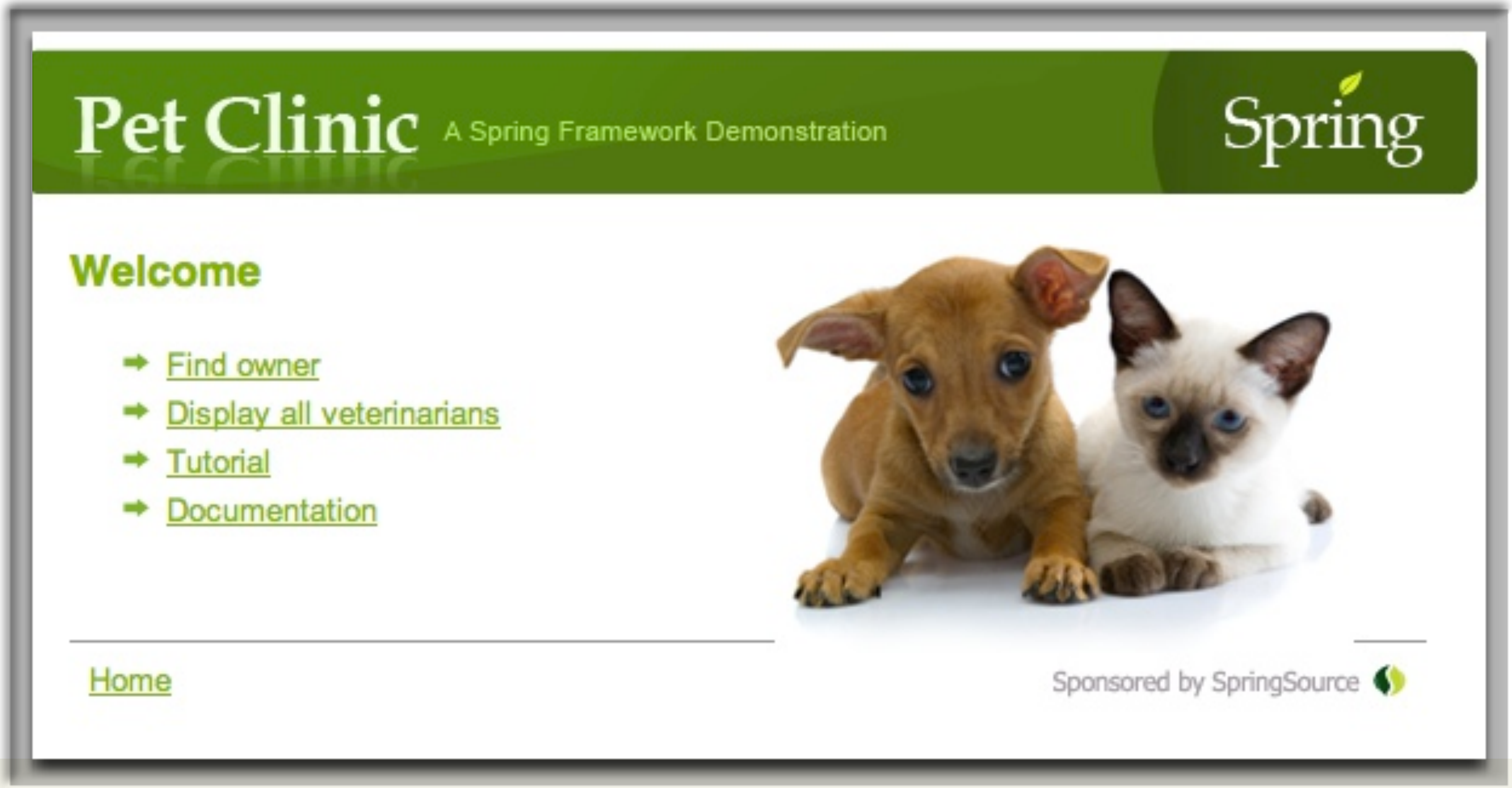
Not coherent/ transactional/HA

Ehcache after Terracotta

Simple + Performant + Coherent + HA + Scaleable



Enabling Hibernate Caching with Spring Pet Clinic



Pet Clinic A Spring Framework Demonstration

Spring

Welcome

- Find owner
- Display all veterinarians
- Tutorial
- Documentation

Home

Sponsored by SpringSource

Caching Hibernate

Steps:

- Configure PetClinic for Hibernate
- Configure hibernate for second-level cache
- Configure hbm file for caching
- Update query code to add caching

Optional but recommended:

- add ehcache.xml to WEB-INF/classes
- specify cache regions and config
- turbo charge with Terracotta

Adding a specific ehcache.xml

ehcache.xml:

```
<ehcache>
    <defaultCache
            maxElementsInMemory="10000"
            eternal="false"
            timeToLiveSeconds="120"
            />

    <cache name="org.hibernate.cache.UpdateTimestampsCache"
           maxElementsInMemory="10000"
           timeToIdleSeconds="300"
            />

    <cache name="org.hibernate.cache.StandardQueryCache"
           maxElementsInMemory="10000"
           timeToIdleSeconds="300"
            />
</ehcache>
```

Adding Terracotta

ehcache.xml

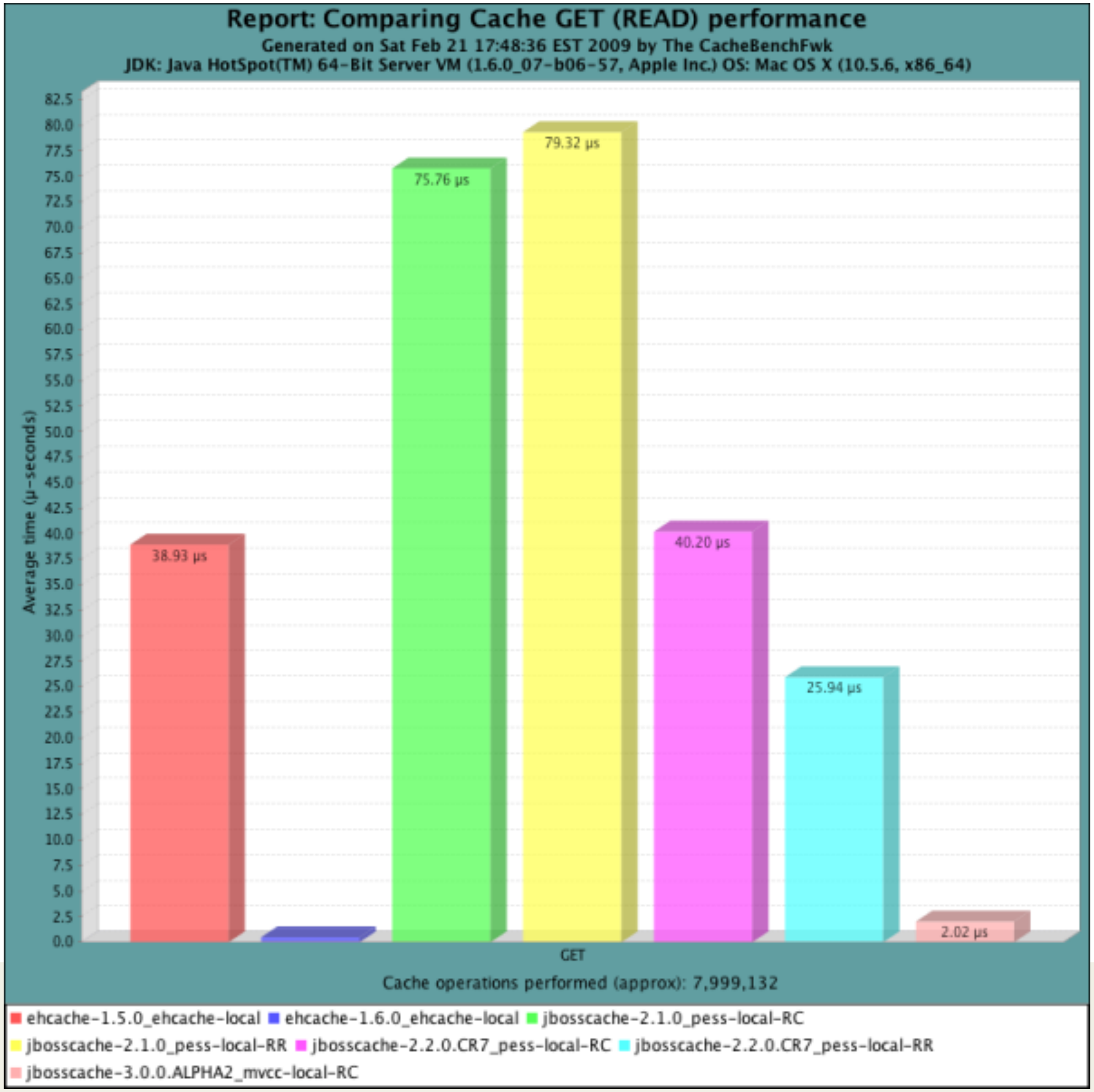
```
<ehcache>
    <terracottaConfig url="someserver:9510"/>
    <defaultCache
            maxElementsInMemory="10000"
            eternal="false"
            timeToLiveSeconds="120"
            />

    <cache name="com.company.domain.Pets"
           maxElementsInMemory="10000"
           eternal=”true”>
           <terracotta clustered="true" coherent="false"/>
           </cache>

    <cache name="com.company.domain.Pets"
           maxElementsInMemory="10000"
           timeToLiveSeconds="3000">
           <terracotta clustered="true" coherent="true"/>
           </cache>
```

Ehcache 2.0 Performance

- Standalone
  - vs Older Ehcache/JBoss
  - vs Memcache
- Distributed
  - vs MySQL
  - vs Memcache
  - vs IMDG
  - vs older Ehcache

Standalone Performance

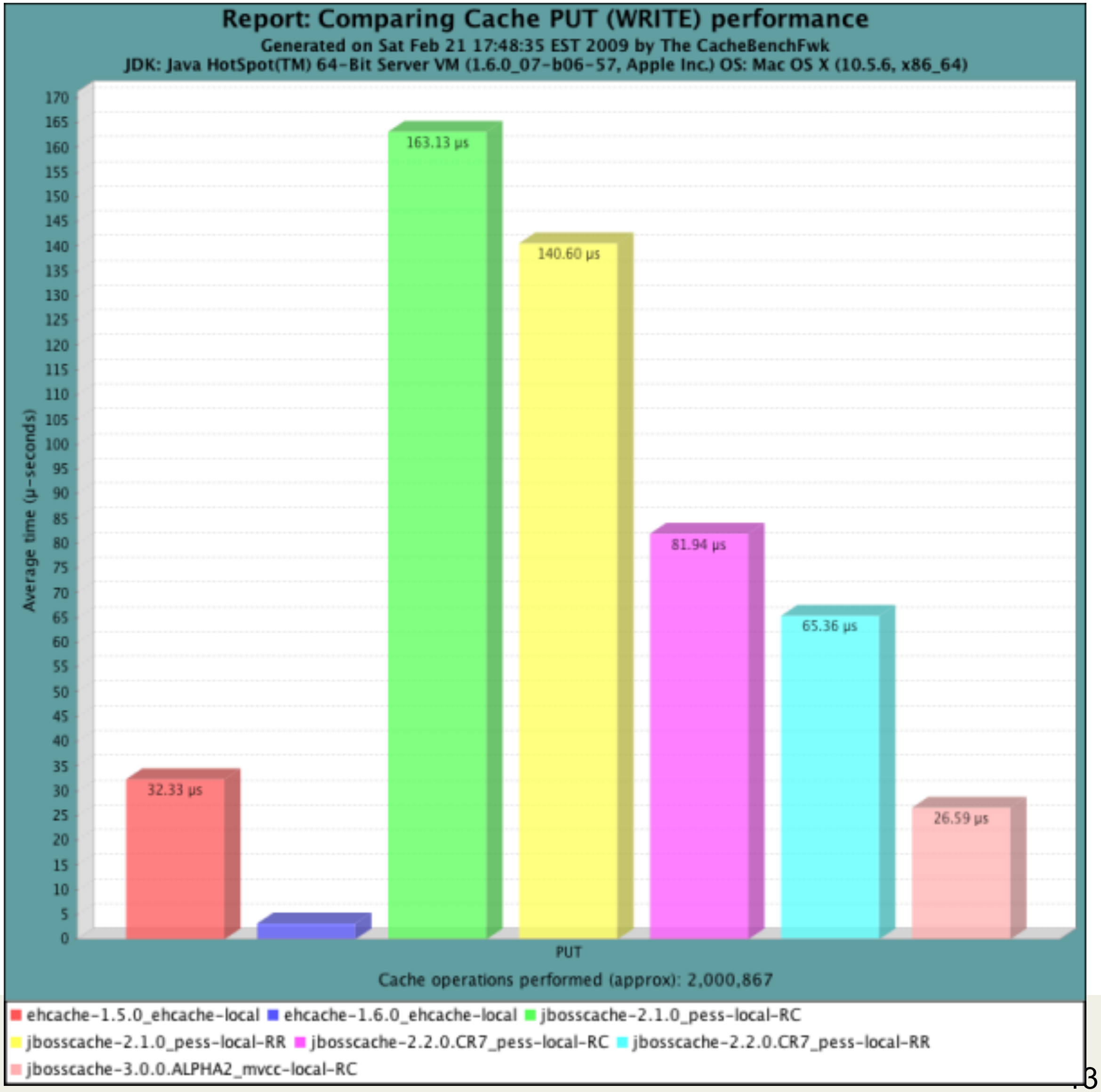
- Older Ehcache
- JBoss Cache
- Read Performance



Source:
JBoss Cache Benchmark Tool

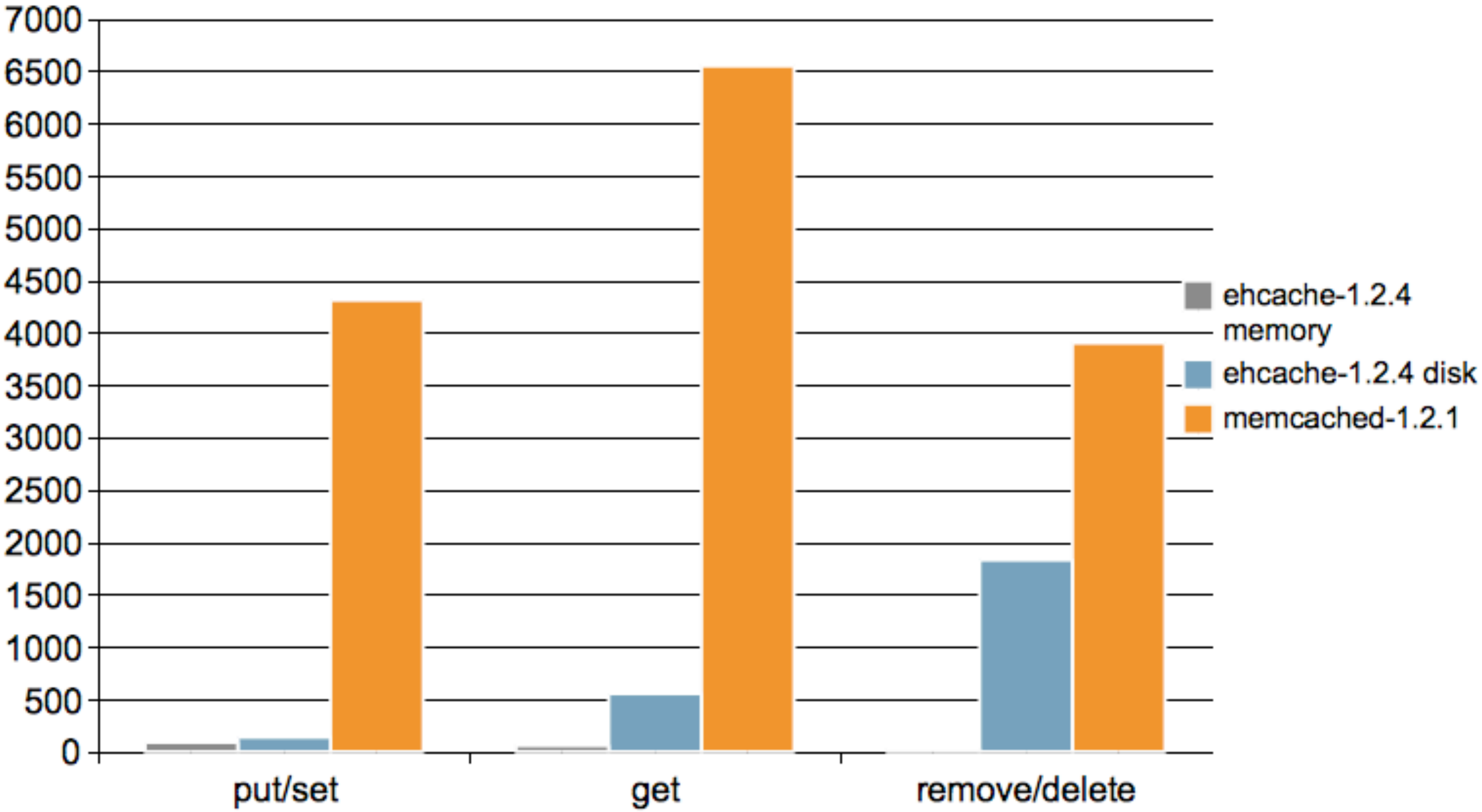
Standalone Performance

- Older Ehcache
- JBoss Cache
- Put Performance



Source:
JBoss Cache Benchmark Tool

Ehcache in-process vs Memcached



(Source: MemCacheBench benchmark)

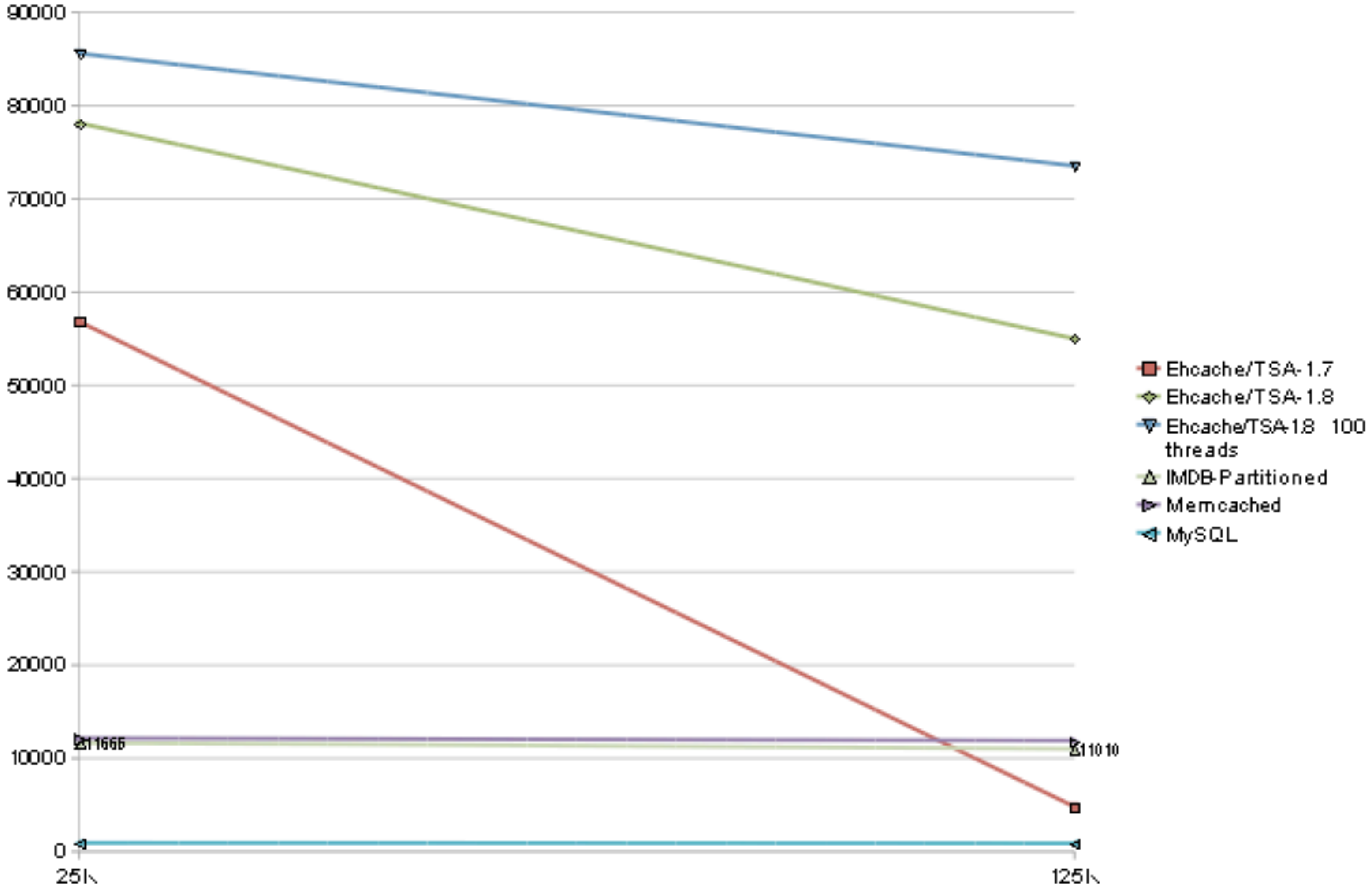
Ehcache with Terracotta vs the Rest

- **Application**
  - Tests done with Owners = 25K and 125K which translates to total objects of 0.3 M and 1.5 M
  - Minimal tuning.
- **Cluster Configuration:**
  - 8 Client JVMs (1.75G Heap)
  - 1 (+0) Terracotta Servers (6G Heap)
  - MySql: sales18.

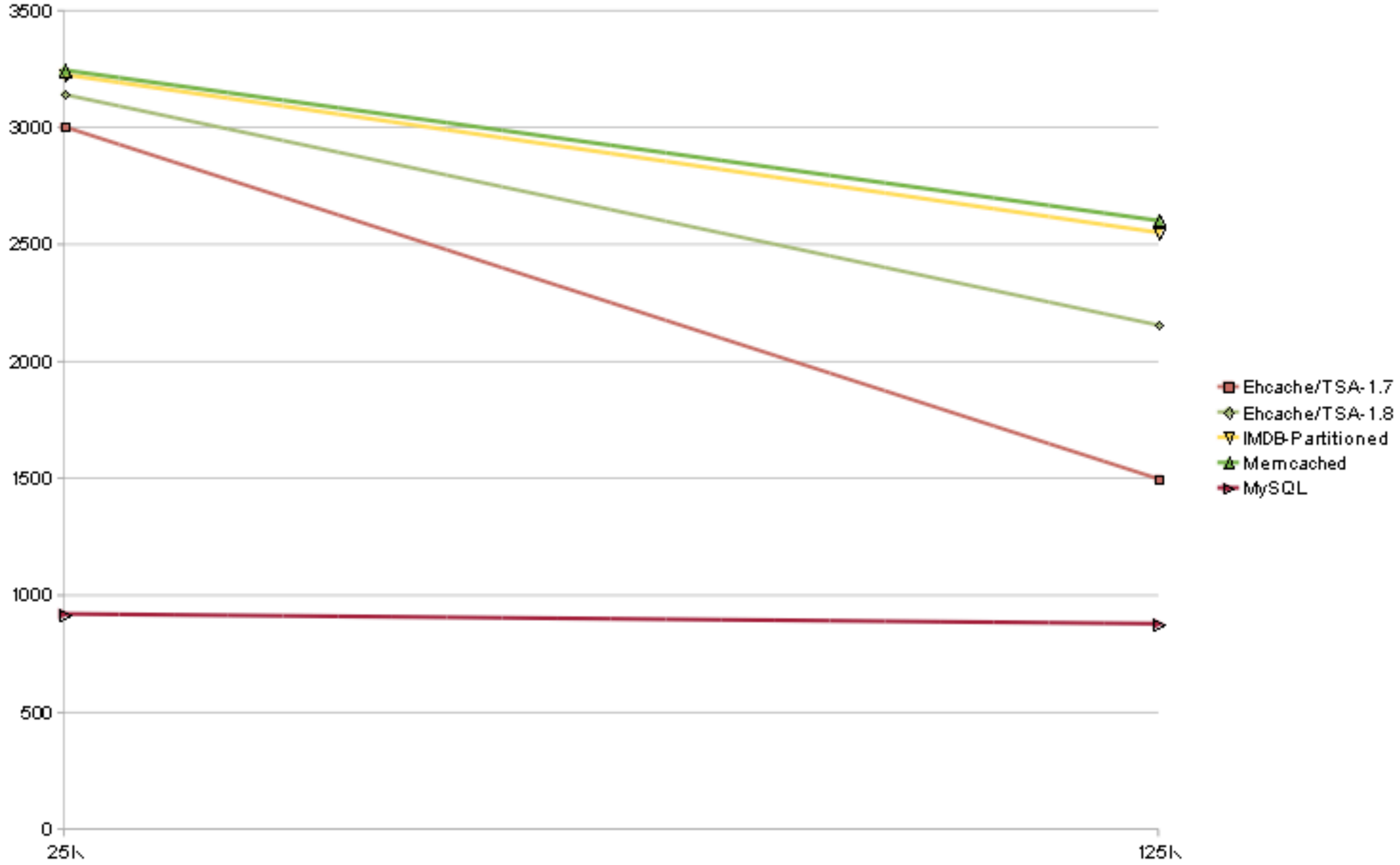
Ehcache with Terracotta vs the Rest

- **Ehcache**
  - Replicated with RMI not included because not coherent
  - Single TSA Server
  - 15 threads and some with 100 threads
- **IMDG**
  - 15 threads
  - Cache deployed in Partitioned Mode
  - Tests were also done with Replicated – which did well for small cache sizes but failed to complete with larger cache sizes. So, it is not included.
- **memcached**
  - 15 threads
  - 1 server

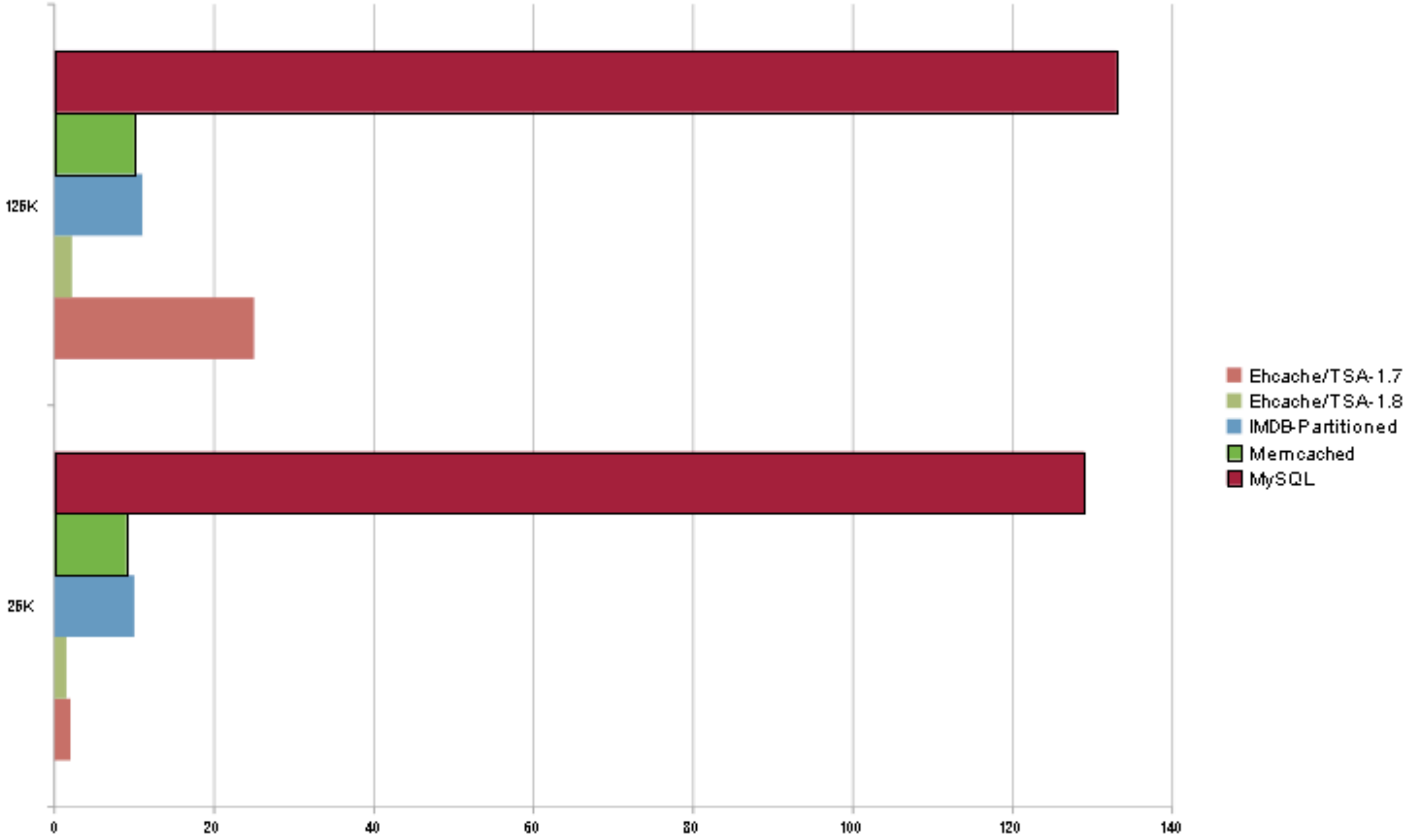
Hibernate - Read Only TPS



Hibernate - Read Write TPS

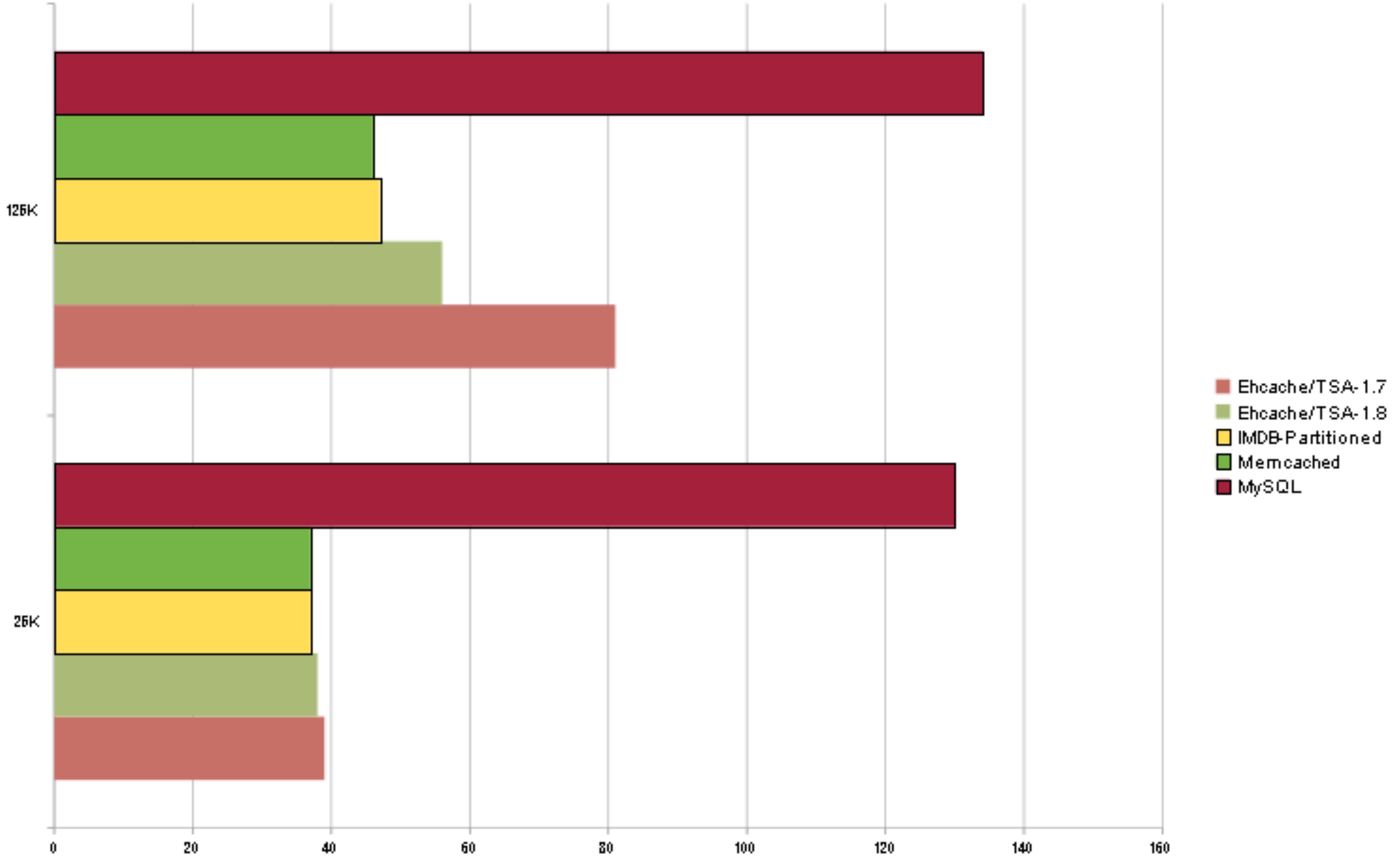


Hibernate - Read Only Latency



TERRACOTTA

Hibernate - Read Write Latency



Test Source

- The code behind the benchmarks is in the Terracotta Community SVN repository.

- Download https://svn.terracotta.org/repo/forge/projects/ehcacheperf/
(Terracotta Community Login Required)

Performance Conclusions

- **With Hibernate, Using Spring Pet Clinic**
  - After app servers and DBs tuned by independent 3rd parties
  - 30-95% database load reduction
  - 80 times read-only performance of MySQL
  - Notably lower latency
- **1.5 ms versus 120 ms for database (25k)**

Ehcache 2.0

- **Hibernate 3.3+ Caching SPI**
  - Old SPI was heavily synchronized and not well suited to clusters
  - New SPI uses CacheRegionFactory
  - Fully cluster safe with Terracotta Server Array
  - Unification of the Ehcache and Terracotta 3.2 providers
- **JTA**
  - Cache as an XAResource
  - Detects most common Transaction Managers
  - Others configurable
  - Works with Spring, EJB and manual transactions

Ehcache 2.0 cont.

- **Write-behind**
  - Offloads Databases with high write workloads
  - CacheStorer Interface to implement
  - cache.putWithWriter(...) and cache.removeWithWriter(...)
  - Write-through and Write-behind modes
  - Batching, coalescing and very configurable
  - Standalone with in-memory write-behind queue.
  - TSA with HA, durability and distributed workload balancing
- **Bulk Loading**
  - incoherent mode for startup or periodic cache loading
  - 10 x faster
  - No change to the API (put, load etc).
  - SetCoherent(), isCoherent(), waitForCoherent()

Product Roadmap ...cont.

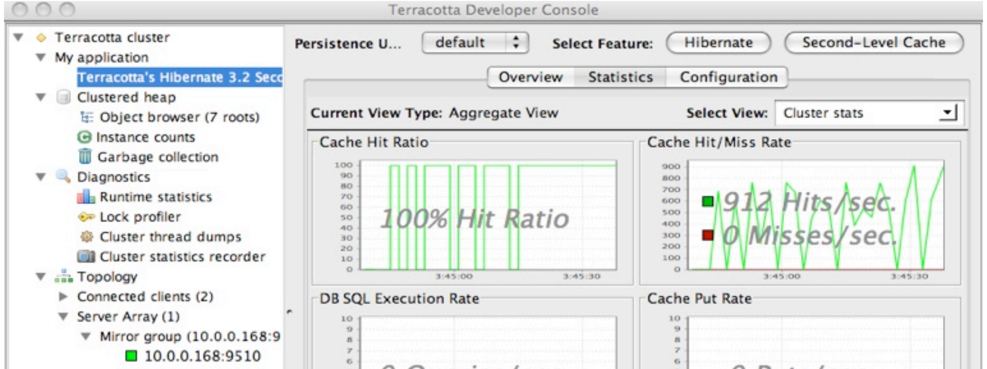
- **New CAP configurability – per cache basis**
  - coherent – run coherent or incoherent (faster)
  - synchronousWrites – true for ha, false is faster
  - copyOnRead – true to stop interactions between threads outside of the cache
  - cluster events – notification of partition and reconnection
- **Management**
  - Dynamic Configuration of common cache configs from JMX and DevConsole
  - New web-based Monitoring with UI and API

Monitoring Options

- Terracotta Dev Console (if using Terracotta)
- JMX is built in to Ehcache
- Ehcache Console

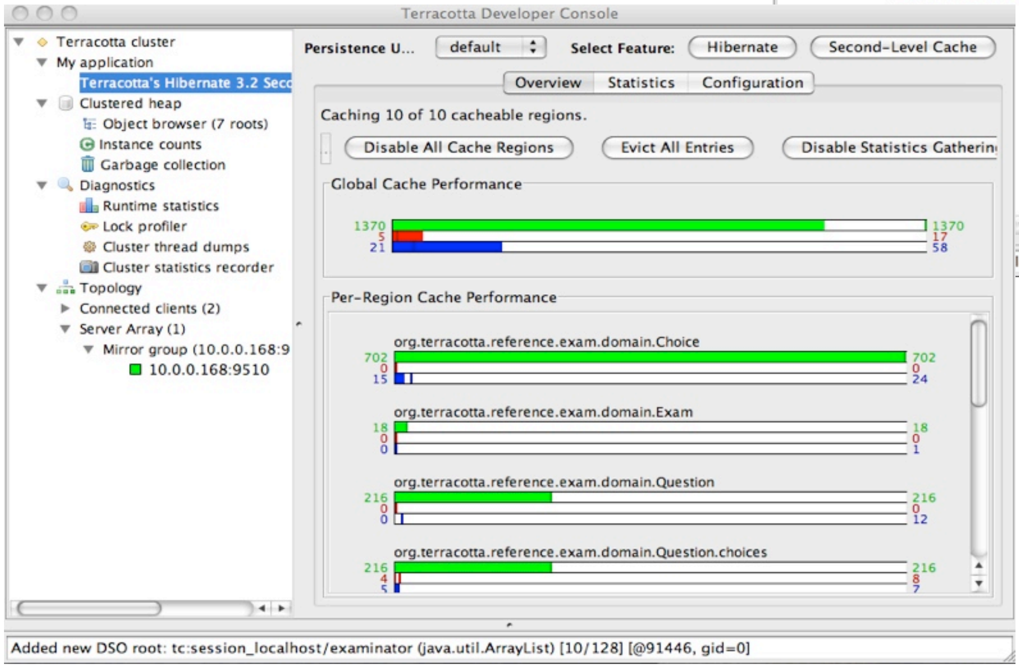
Visual Cache Tuning - Dev Console

- Cache hit ratios
- Hit/miss rates
- Hits on the database
- Cache puts
- Detailed efficiency of cache regions



Region	Hit Ratio	Hits	Misses	Puts	In-Memo...	Total Co...	Hit Latency	Load Lat...
t.r.e.d.U...	100%	16	0	2	---	0	0	0
t.r.e.d....	100%	4,524	0	36	---	0	0	1
t.r.e.d.C...	100%	14,703	0	117	---	5	0	0
t.r.e.d.E...	100%	380	0	3	---	0	0	7

localhost/examinator (java.util.ArrayList) [10/128] [@91446, gid=0]



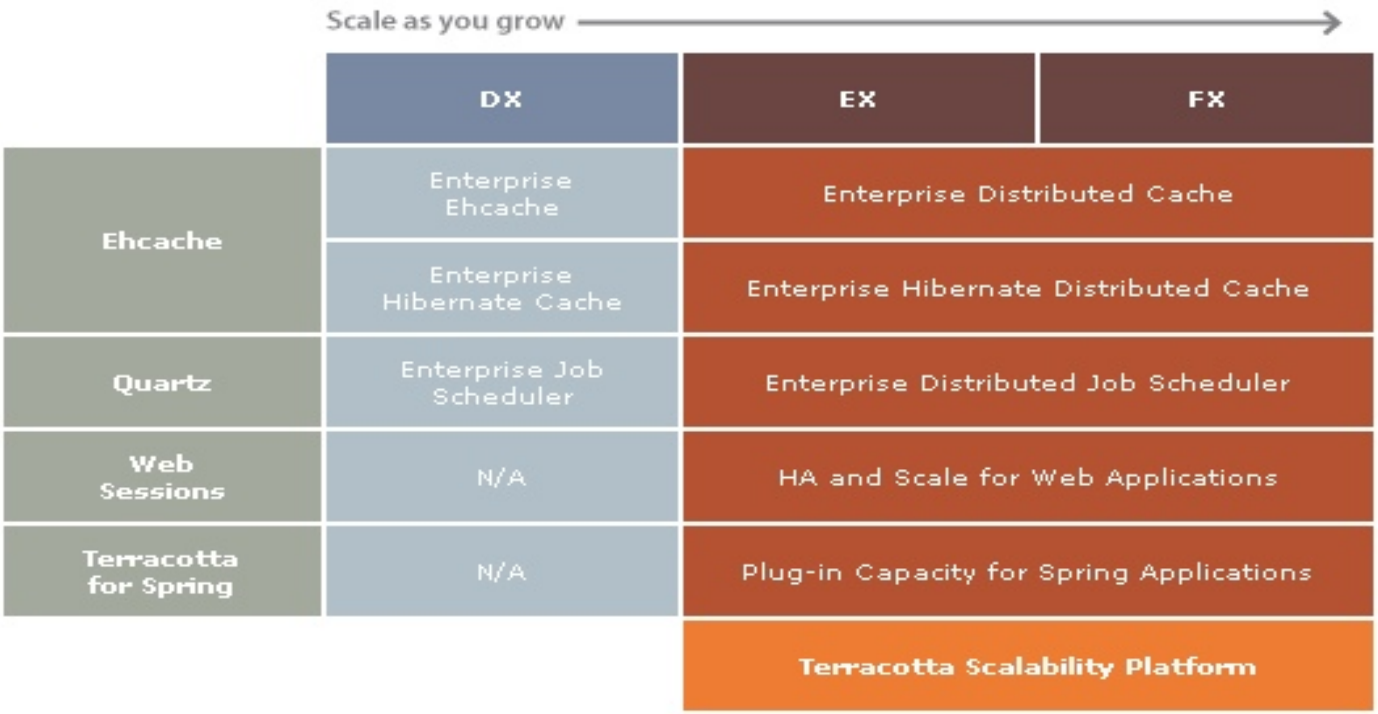
Caching 10 of 10 cacheable regions.

Disable All Cache Regions
Evict All Entries
Disable Statistics Gatherin
Global Cache Performance
1370
5
21
1370
17
58
Per-Region Cache Performance
org.terracotta.reference.exam.domain.Choice
702
0
15
702
0
24
org.terracotta.reference.exam.domain.Exam
18
0
0
18
0
1
org.terracotta.reference.exam.domain.Question
216
0
0
216
0
12
org.terracotta.reference.exam.domain.Question.choices
216
4
5
216
8
7

Added new DSO root: tc:session_localhost/examinator (java.util.ArrayList) [10/128] [@91446, gid=0]

Dramatically simplifies tuning and operations, and shows the database offload.

Terracotta Commercial Products



Scale as you grow →

	DX	EX	FX
Ehcache	Enterprise Ehcache	Enterprise Distributed Cache	
	Enterprise Hibernate Cache	Enterprise Hibernate Distributed Cache	
Quartz	Enterprise Job Scheduler	Enterprise Distributed Job Scheduler	
Web Sessions	N/A	HA and Scale for Web Applications	
Terracotta for Spring	N/A	Plug-in Capacity for Spring Applications	
		Terracotta Scalability Platform	

Enterprise Support Included in Commercial Offerings:

- 24x7 support for mission critical business functions
- Guaranteed time-to-respond service level agreement (SLA)
- Thoroughly tested patches

Additional Ehcache Information

- Website: www.ehcache.org
- Documention: www.ehcache.org/documentation
- Hibernate: www.ehcache.org/documentation/hibernate.html
- Commercial Products: www.terracotta.org/ehcache/
- Twitter: www.twitter.com/Ehcache

Q&A

- Please ask any questions you have in the Q&A window.

Terracotta Contact Information

- Website: www.terracottatech.com
- Telephone: +1 415-738-4000
- Email: info@terracottatech.com
- Facebook: www.facebook.com/Terracotta
- Twitter: www.twitter.com/TerracottaTech