WHITEPAPER

Hazelcast for Oracle Coherence Users

*For Oracle Coherence users new to Hazelcast, this whitepaper will highlight some similarities and differences.**

ARCHITECTURE

Hazelcast 과 Oracle Coherence 솔루션은 진정한 의미의 in-memory data grids 시스템입니다. Application server 와 동일한 하드웨어안에 embedded 하거나 , 독립적인 하드웨어에 설치하여 사용할 수 있습니다. 독립적인 데이터 운용과는 달리 한 시스템 안에서 서로 연결되며, 데이터 반복적으로 물리적인 서버들 사이에서 복제되고 생산됩니다. Hazelcast 과 Oracle Coherence 동시에 distributed and replicated Maps, Query and Execution 을 지원합니다. Hazelcast 는 특별한 세팅이나 추가 작업없이 설치와 동시에 supports Topics, Sets and Queues 기능을 지원합니다.

Grid communication in Oracle Coherence is provided by the closed, proprietary TCMP (Tangosol Cluster Management Protocol). Hazelcast's communication protocol is open source. 이러한 특징은 최종사용자가 소스코드를 보유함과 동시에 필요한 기능들을 추가적으로 개발할 수 있습니다.

Hazelcast has built its distributed collections on top of an open source Service Provider Interface (SPI). The SPI can be used to build new distributed data structures.

SCALE (확장성)

Hazelcast and Oracle Coherence both achieve scale out by elastically adding nodes. In both systems, nodes may be added or removed dynamically at runtime and data will rebalance across the new grid topology.

For both Hazelcast and Oracle Coherence, scale up is achieved with off-heap memory by storing keys and metadata on-heap while storing values off-heap for a 10x storage capacity over heap alone for the same garbage collection performance. In Hazelcast, this is done via the High-Density Memory Store.

Hazelcast supports Entry Processors within off-heap memory, while Oracle Coherence does not.

PERSISTENCE (연속성)

Hazelcast supports write-behind (asynchronous) and write-through (synchronous) persistence interfaces. Oracle Coherence offers persistence via Backing Maps. – 시스템 운용 속도에서 현저한 성능 차이를 보여줍니다. Both allow cache write-through, write-behind and read-through onto back end data stores such as an RDBMS or NoSQL services.

LICENSING

Hazelcast is dual license, offering both open source and commercial licenses. Oracle Coherence is commercial only

COMMERCIAL OFFERINGS

Hazelcast offers a professional 24/7 support subscription for open source users, and a license for the enterprise version with additional value-add features such as security, High-Density Memory Store, WAN Replication, C# and C++ Clients, and an enterprise cluster management application.

Oracle Coherence is a commercial product with three editions: Standard, Enterprise and Grid. Features such as off-heap memory, WAN support and Real Time Clients are only available in the Grid Edition.

USE CASES

Hazelcast is an in-memory data grid which can be used for:

- Distributed Collections
- Distributed Locking
- Distributed Compute

Both Hazelcast and Oracle Coherence support distributed caching and web sessions.

GRID CLIENTS

Hazelcast provides an open source Java client and also supports the Memcache interface so that any platform with a memcache client driver can communicate directly with the grid. C# and C++ versions are available within the Enterprise version. The clients provide notifications, execution, near cache and continuous query. Hazelcast clients may connect to any configured member of the grid.

Oracle Coherence provides two different types of client licences. Data clients are available in Standard and Enterprise editions that may get, put, and query data in the grid. Available only in the grid edition are Real Time clients that may receive notifications from the grid, provide near cache and continuous query. Both types of client are available in Java, C# and C++.

Oracle Coherence clients must connect to the grid via specially configured “Extend Proxies.”

	HAZELCAST 3.5	COHERENCE 12.1.2
License	Open Source and Commercial Versions	Commercial
Product Category	In-Memory Data Grid	In-Memory Data Grid
Architectural Topology	Grid/Client-Server	Grid/Client-Server
Elasticity	✓	✓
Data Redundancy	Up to 6 replicas	Recommended 0-2
Memory Density in JVM	Hundreds of GBs	2-3GB/node. RAMJournal and SSDJournal available but not all features work
Persistence		Yes, via Backing Maps
USE CASES		
Cache	✓	✓
Web Sessions	Yes, filter based for Servlet 2.4 and Tomcat native session	✓
DISTRIBUTED COLLECTIONS		
Replicated Map	✓	✓
Distributed Map	✓	✓
Multimap	✓	
DISTRIBUTED CONCURRENCY		
Distributed Lock	✓	✓
Distributed Atomic Long	✓	✓
Distributed Atomic Ref	✓	
Distributed Semaphore	✓	
CLIENTS		
Java Thick	✓	✓
Java Thin	✓	✓
C++	✓	✓
C#	✓	✓
REST	✓	✓
WAN	Yes, highly extensible and supports custom WAN replication implementation	Yes, Push based, not part of main product

	HAZELCAST 3.5	COHERENCE 12.1.2
SEARCH		
Predicate API	✓	✓
SQL API	✓	✓
Continuous Query	✓	✓
Map/Reduce	✓	
Aggregations	Yes, built on Map/Reduce	✓
CACHE		
Memcache API	✓	✓
JCache API	✓	✓
Near Cache	✓	✓
High-Density Caching	Yes, with High Density Memory Store	
High-Density Near Cache	✓	
Bootstrap Near Cache		
Write-Behind	✓	✓
Refresh-Ahead		✓
Scheduled Refresh		
DISTRIBUTED COMPUTE		
Entry Processor/ Invocable Map	✓	✓
Executor Service	✓	✓
Map/Reduce	✓	✓
DISTRIBUTED MESSAGING		
Distributed Queue	✓	Yes, Coherence Incubator
Distributed Events	✓	Yes, Coherence Incubator
TRANSACTIONS		
Local Transactions	✓	✓
XA Transactions	✓	✓
Isolation Levels	READ_COMMITED, REPEATABLE_READ	READ_COMMITED, REPEATABLE_READ

	HAZELCAST 3.5	COHERENCE 12.1.2
TRANSACTIONS (CONTINUED)		
Transaction Manager Support	Yes, full support for global as well as local Transaction Managers of both types – built-in and third party	✓
ENTERPRISE MANAGEMENT		
Monitor	✓	
Management API	Yes, via JMX	✓
Backup		
Recovery		
Upgradeability with Data	Yes, via Portable Serialization	Yes, via POF Serialization
Rolling Upgrades		✓
Integration with Enterprise Management	Yes, via REST or JMX	✓
SECURITY		
Authentication	✓	✓
Authorisation	✓	✓
Confidentiality - wire	✓	✓
Confidentiality - storage	✓	
GRID FEATURES		
Key Affinity	✓	✓
Delta Updates	Yes, via EntryProcessor	✓
Partial Deserialize Objects	✓	✓
Embeddable Nodes	✓	✓

* Comparison based on our best knowledge at the time of document creation.