ORACLE®

XTP, Scalability and Data Grids
An Introduction to Coherence

Tom Stenström
Principal Sales Consultant
Oracle

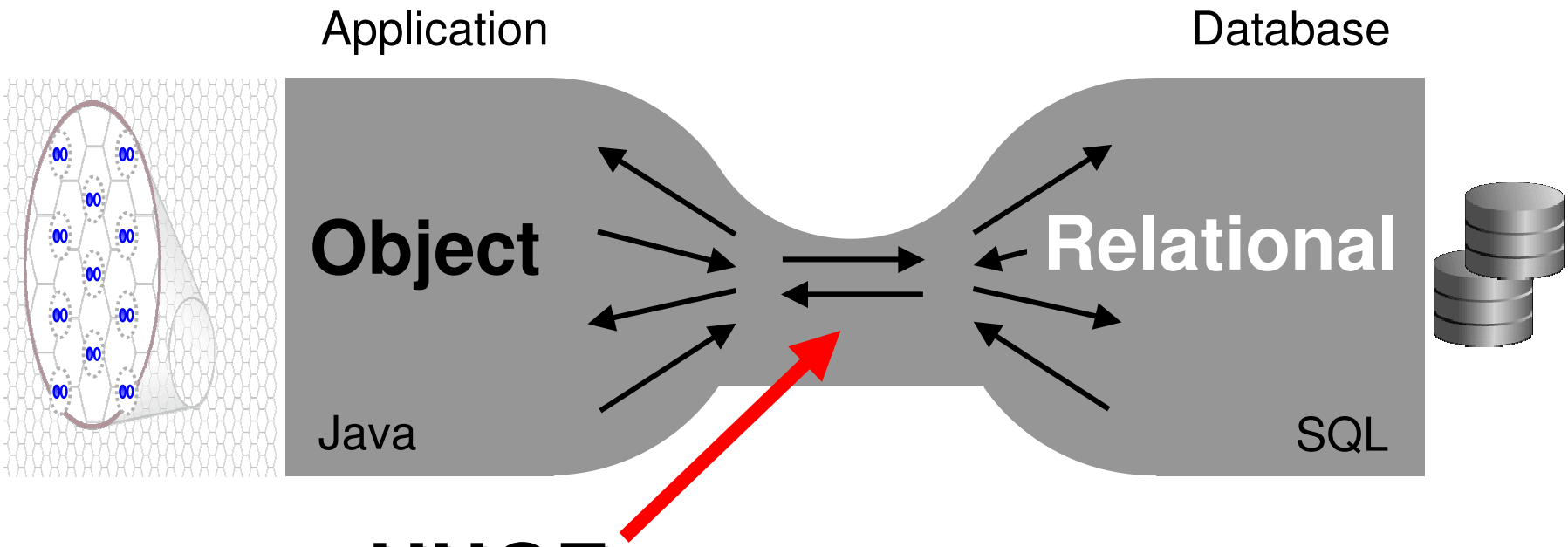
Presentation Overview

- The challenge of scalability
- The Data Grid
- What is Coherence
- How does Coherence work
- Using Coherence
- Topologies
- Examples of users and usage

ORACLE

What is the challenge ?

Why Go Outside the Database to Scale Java Applications?



A HUGE performance bottleneck:

Volume / Complexity / Frequency of Data Access

Impact of XTP (Extreme Transaction Processing)

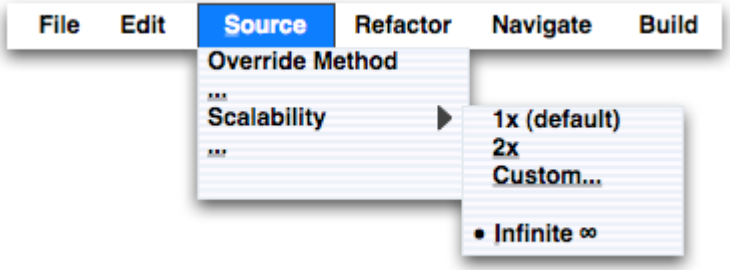
- Handling 10 users running 10 tps is simple ☺
- But what happens when you have a winner on your hands? The Killer App!
- Is it designed for...
  - Scaling!
  - Performance!
  - Availability!
- 1000 Users running 1000 tps?
- 10000 Users running 10000 tps?
- What about 500.000 tps?

ORACLE

What is Coherence?

Application Scalability

- Scaling the Application-Tier is difficult
- If it was easy it would be an IDE option



Not possible!

- Scalability is a design option
  - Developers have the "option" to consider building it in!
  - It's not an IDE option
- Coherence is scalability infrastructure for the application-tier

In the Industry

- JCP JCache (JSR 107 spec lead)
  - JSR 236/237 implementations
- Tangosol productified it as “Coherence”
- Tangosol acquired by Oracle 2007
- JSE and/or JEE – Pure Java
  - Application servers: Oracle, BEA, IBM, Sun…
- .Net client
  - Pure, no embedded JVMs!
- Proprietary Network Stack (Peer-To-Peer model)
  - TCMP

Coherence - For Applications

- Oracle Coherence doesn't require a container / server
- A single library
  - Database and File System Integration
  - Top Link and Hibernate
  - Http Session Management
  - Spring
- No external / open source dependencies
- Can be embedded or run standalone
- Runs where Java SE / EE, .NET runs
- Won't impose architectural patterns

ORACLE

Coherence - Data

- Data in Oracle Coherence…
- Any serializable* Object
- Fully native Java & .NET interoperability
- No byte-code instruction or multi-layer facades
- Not forced to use Relational Models, Object-Relational-Mapping, SQL etc
- Just real POJOs and PONOs

***serialization = writing to binary form**

Coherence - Data

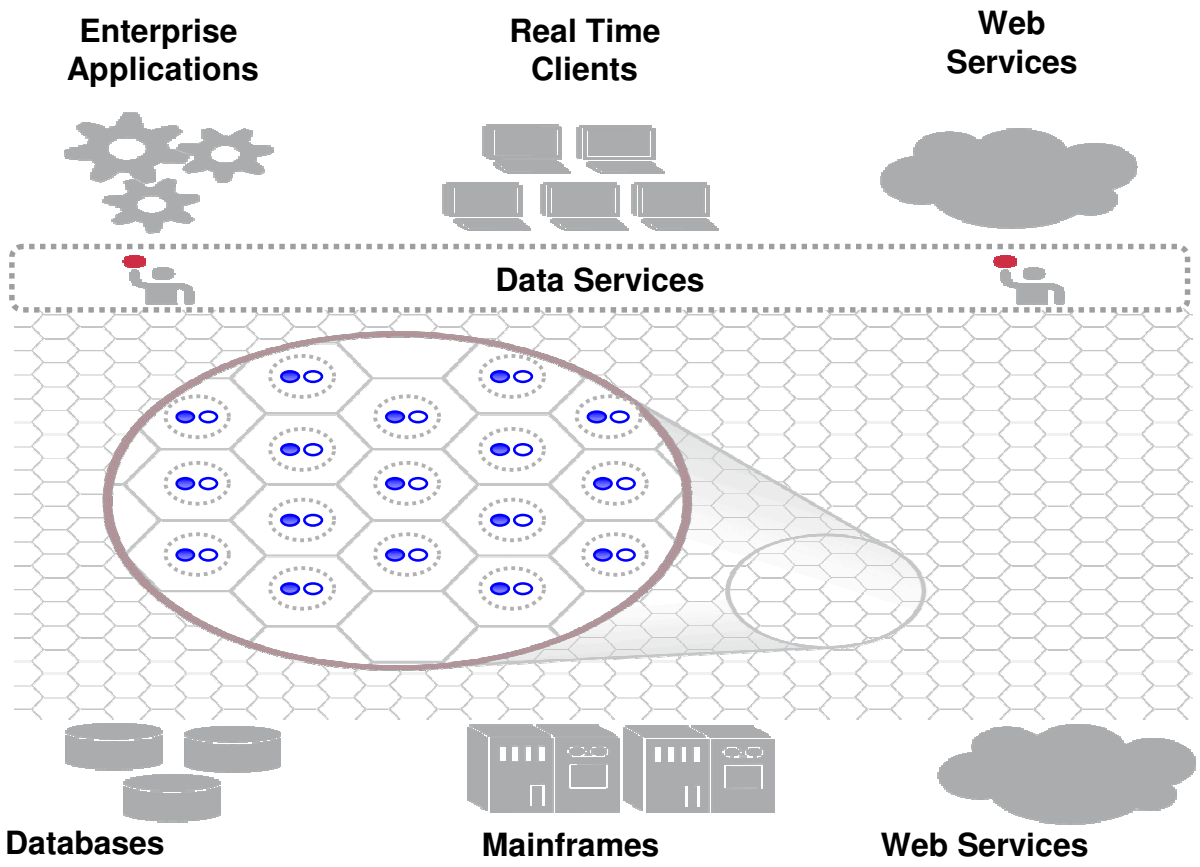
- Different topologies for your Data
- Simple API for all Data, regardless of Topology
- Things you can do…
  - Distributed Objects, Maps & Caching
  - Real-Time Events, Listeners
  - Parallel Queries & Indexing
  - Data Processing and Service Agents (Grid features)
  - Continuous Views
  - Aggregation
  - Persistence, Sessions…

Coherence - Management Solution

- Responsible for Clustering, Data and Service management, including partitioning
- Ideally engineers should not have to…
  - design, specify and code how partitioning occurs in a solution
  - manage the Cluster, either manually or in code
  - shutdown the system to add new resources or repartition
  - use “consoles” to recover or scale a system.
- These are impediments to scaling cost effectively
- Clustering technology should be invisible in your solution!

ORACLE

Positioning Coherence



How does Coherence work?

Membership Consensus

- Membership Consensus:

  "A common agreement between a set of processes as to the membership of the group at a point in time"

Clustering is about Consensus!

Oracle Coherence Clustering Goal:

- Maintain Cluster Membership Consensus all times
- Do it as fast as physically possible
- Do it without a single point of failure or registry of members
- Ensure all members have the same responsibility and work together to maintain consensus
- Ensure that no voting occurs to determine membership

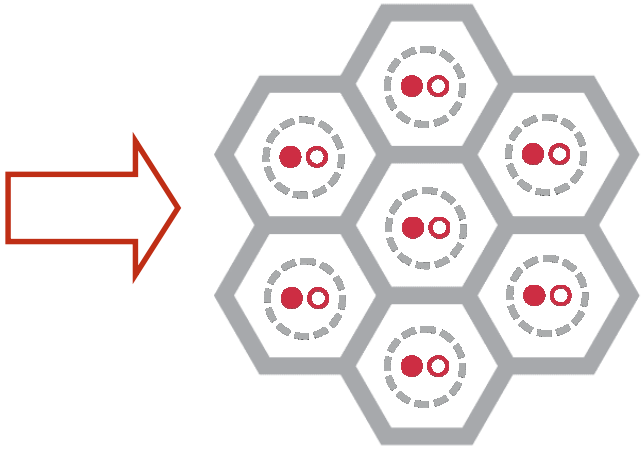
Clustering is about Consensus!

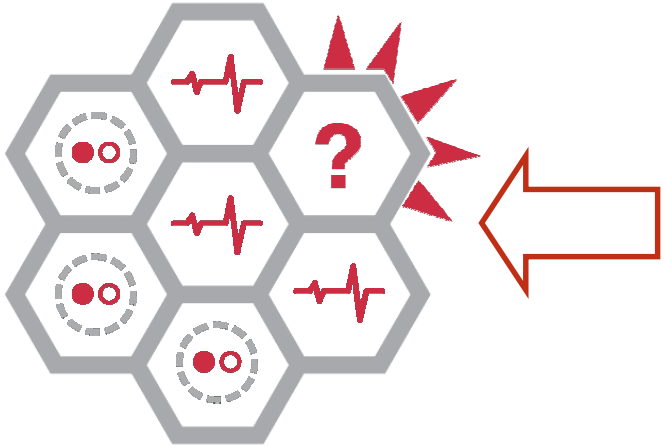
Why: If all members are always known…

- We can partition / load balance Data & Services
- We don't need to hold TCP/IP connections open (resource intensive)
- Any member can "talk" directly with any other member (peer-to-peer)
- The cluster can dynamically (while running) scale to any size

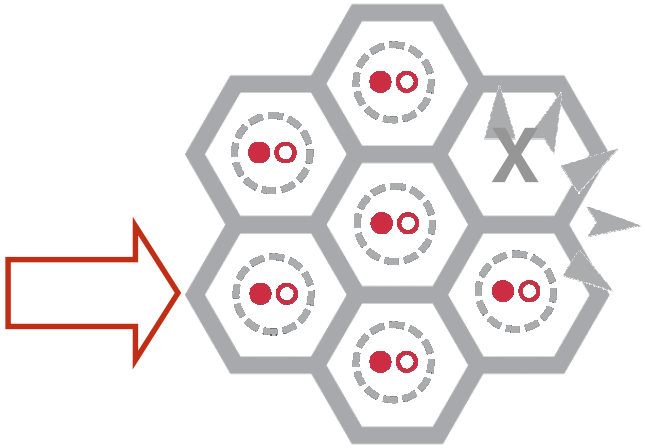
How Does Coherence™ Data Grid Work?

- Cluster of nodes holding % of primary data locally
- Back-up of primary data is distributed across all other nodes
- Logical view of all data from any node



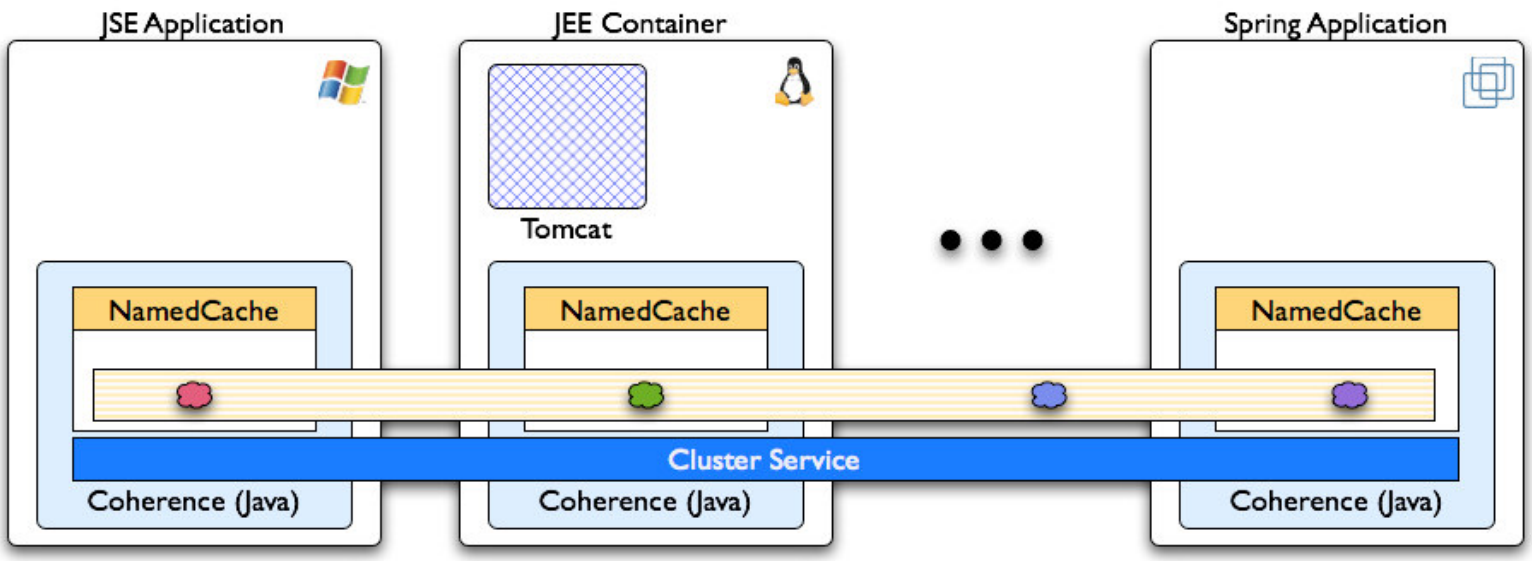


- All nodes verify health of each other
- In the event a node is unhealthy, other nodes diagnose state



- Unhealthy node isolated from cluster
- Remaining nodes redistribute primary and back-up responsibilities to healthy nodes

TCMP Provides the Foundation



Coherence “demo”…

Starting a Cache Server (Data Management Process)

Terminal — bash — 107x30

```
host94:~/Desktop/coherence-swing-demo-2.0 boliver$ ./cache-server.sh
```

Starting a Cache Server to store data in memory

Starting a Cache Server (Data Management Process)

The ID of the first cluster member

Terminal — java — 107x30

```
Group{Address=224.3.3.0, Port=33387, TTL=4}

MasterMemberSet
  (
  ThisMember=Member(Id=1, Timestamp=2007-10-31 11:05:47.923, Address=172.16.154.1:8088, MachineId=9729, Loc
ation=process:400@host94.msm.che.vodafone)
  OldestMember=Member(Id=1, Timestamp=2007-10-31 11:05:47.923, Address=172.16.154.1:8088, MachineId=9729, L
ocation=process:400@host94.msm.che.vodafone)
  ActualMemberSet=MemberSet(Size=1, BitSetCount=2
    Member(Id=1, Timestamp=2007-10-31 11:05:47.923, Address=172.16.154.1:8088, MachineId=9729, Location=pro
cess:400@host94.msm.che.vodafone)
    )
  RecycleMillis=120000
  RecycleSet=MemberSet(Size=0, BitSetCount=0
    )
  )

Services
  (
  TcpRing{TcpSocketAccepter{State=STATE_OPEN, ServerSocket=172.16.154.1:8088}, Connections=[]}
  ClusterService{Name=Cluster, State=(SERVICE_STARTED, STATE_JOINED), Id=0, Version=3.3, OldestMemberId=1}
  DistributedCache{Name=DistributedCache, State=(SERVICE_STARTED), Id=1, Version=3.2, OldestMemberId=1, Loc
alStorage=enabled, PartitionCount=257, BackupCount=1, AssignedPartitions=257, BackupPartitions=0}
  ReplicatedCache{Name=ReplicatedCache, State=(SERVICE_STARTED), Id=2, Version=3.0, OldestMemberId=1}
  Optimistic{Name=OptimisticCache, State=(SERVICE_STARTED), Id=3, Version=3.0, OldestMemberId=1}
  InvocationService{Name=InvocationService, State=(SERVICE_STARTED), Id=4, Version=3.1, OldestMemberId=1}
  )
```

Loading Some Data into Grid (creating financial positions)

```
Terminal — bash — 107x30
host94:~/Desktop/coherence-swing-demo-2.0 boliver$ ./create-positions.sh
2007-10-31 11:07:20.021 Oracle Coherence 3.3/387p3 <Info> (thread=main, member=n/a): Loaded operational con
figuration from resource "jar:file:/Users/boliver/Desktop/coherence-swing-demo-2.0/lib/coherence-3.3.0.jar!
/tangosol-coherence.xml"
2007-10-31 11:07:20.034 Oracle Coherence 3.3/387p3 <Info> (thread=main, member=n/a): Loaded operational ove
rrides from resource "jar:file:/Users/boliver/Desktop/coherence-swing-demo-2.0/lib/coherence-3.3.0.jar!/tan
gosol-coherence-override-dev.xml"

Oracle Coherence Version 3.3/387p3
 Grid Edition: Development mode
Copyright (c) 2000-2007 Oracle. All rights reserved.

2007-10-31 11:07:20.333 Oracle Coherence GE 3.3/387p3 <Info> (thread=main, member=n/a): Loaded cache config
uration from resource "jar:file:/Users/boliver/Desktop/coherence-swing-demo-2.0/lib/coherence-3.3.0.jar!/co
herence-cache-config.xml"
2007-10-31 11:07:20.836 Oracle Coherence GE 3.3/387p3 <Warning> (thread=main, member=n/a): UnicastUdpSocket
 failed to set receive buffer size to 1428 packets (2096304 bytes); actual size is 44 packets (65507 bytes)
. Consult your OS documentation regarding increasing the maximum socket buffer size. Proceeding with the ac
tual value may cause sub-optimal performance.
2007-10-31 11:07:21.263 Oracle Coherence GE 3.3/387p3 <Info> (thread=Cluster, member=n/a): This Member(Id=3
, Timestamp=2007-10-31 11:07:21.064, Address=172.16.154.1:8089, MachineId=9729, Location=process:409@host94
.msm.che.vodafone, Edition=Grid Edition, Mode=Development, CpuCount=2, SocketCount=2) joined cluster with s
enior Member(Id=1, Timestamp=2007-10-31 11:05:47.923, Address=172.16.154.1:8088, MachineId=9729, Location=p
rocess:400@host94.msm.che.vodafone, Edition=Grid Edition, Mode=Development, CpuCount=2, SocketCount=2)
Adding Indexes...
Creating 100000 Positions...
Creation Complete! (Cache contains 100000 positions)
host94:~/Desktop/coherence-swing-demo-2.0 boliver$
```

We index the positions to make queries event faster in memory

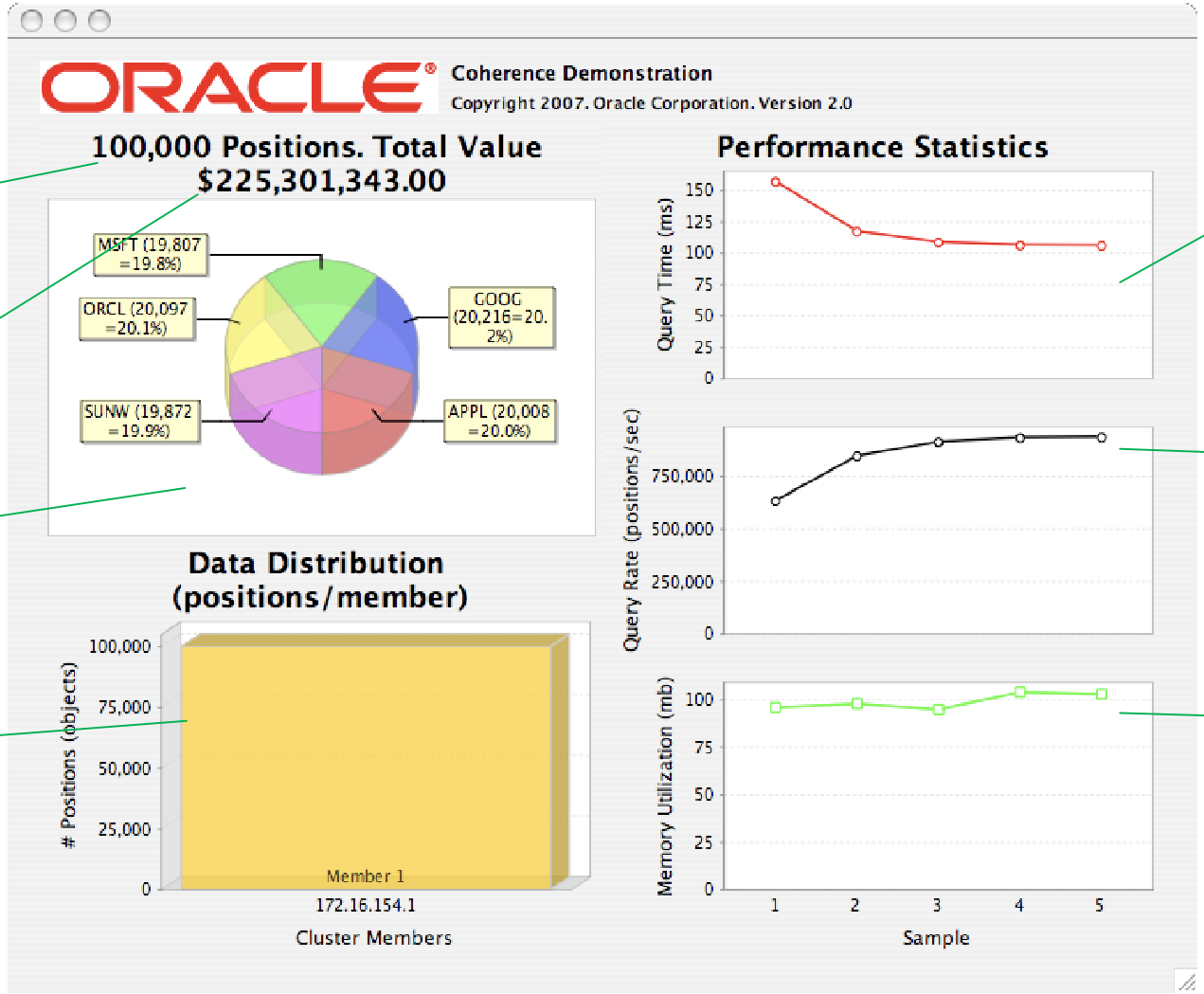
Starting GUI (client) Application

Number of Positions in the Cluster

Total portfolio value

Position Break Down (6 separate queries)

Positions managed per Cluster member (server)



Average Query Response Time (lower is better)

Average number of positions we can query per second (higher is better)

Memory Consumption per Cluster member (server)

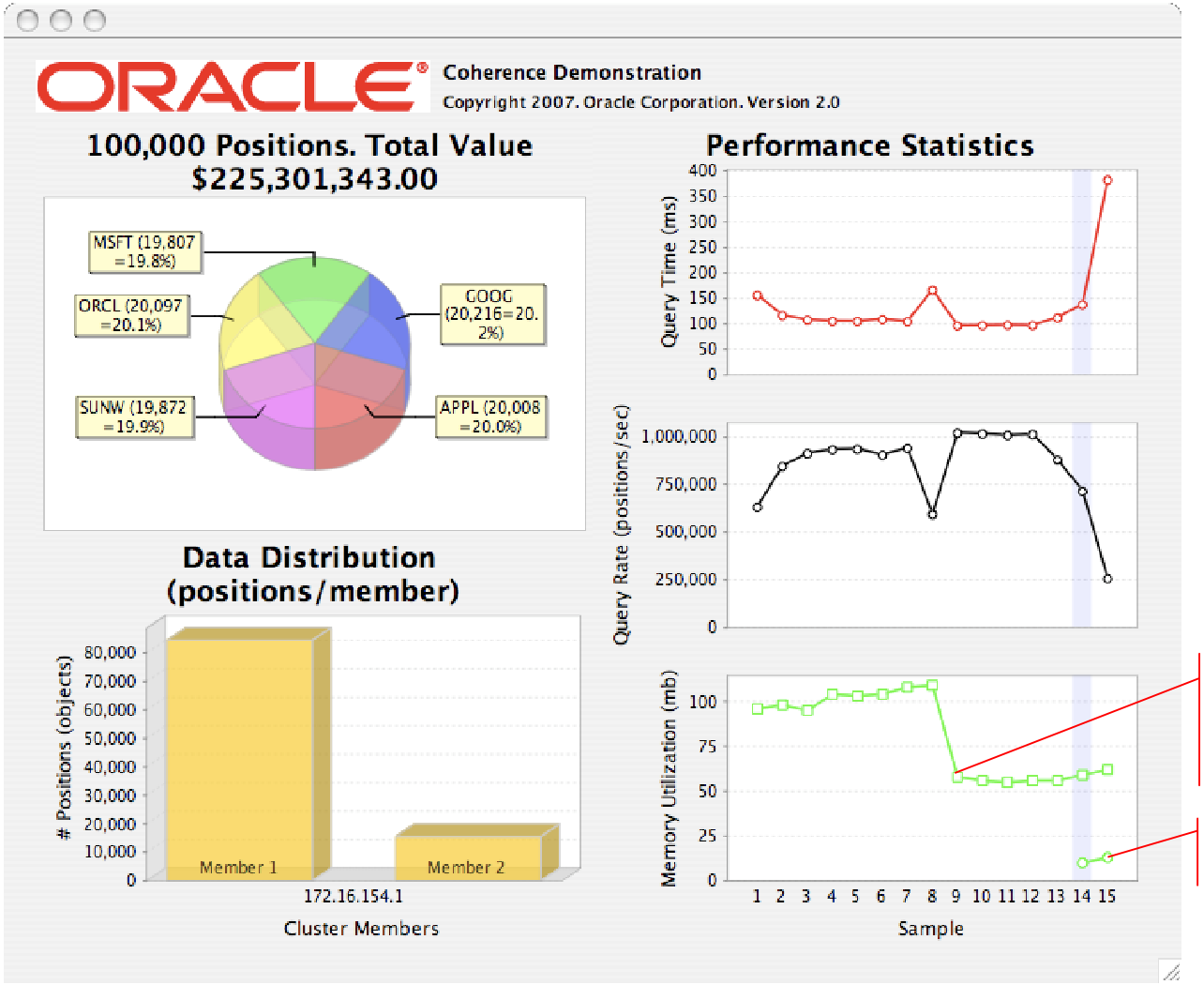
Starting another Cache Server (Data Management Process)

Terminal — bash — 107x30

```
host94:~/Desktop/coherence-swing-demo-2.0 boliver$ ./cache-server.sh
```

Starting another Cache Server to store data in memory

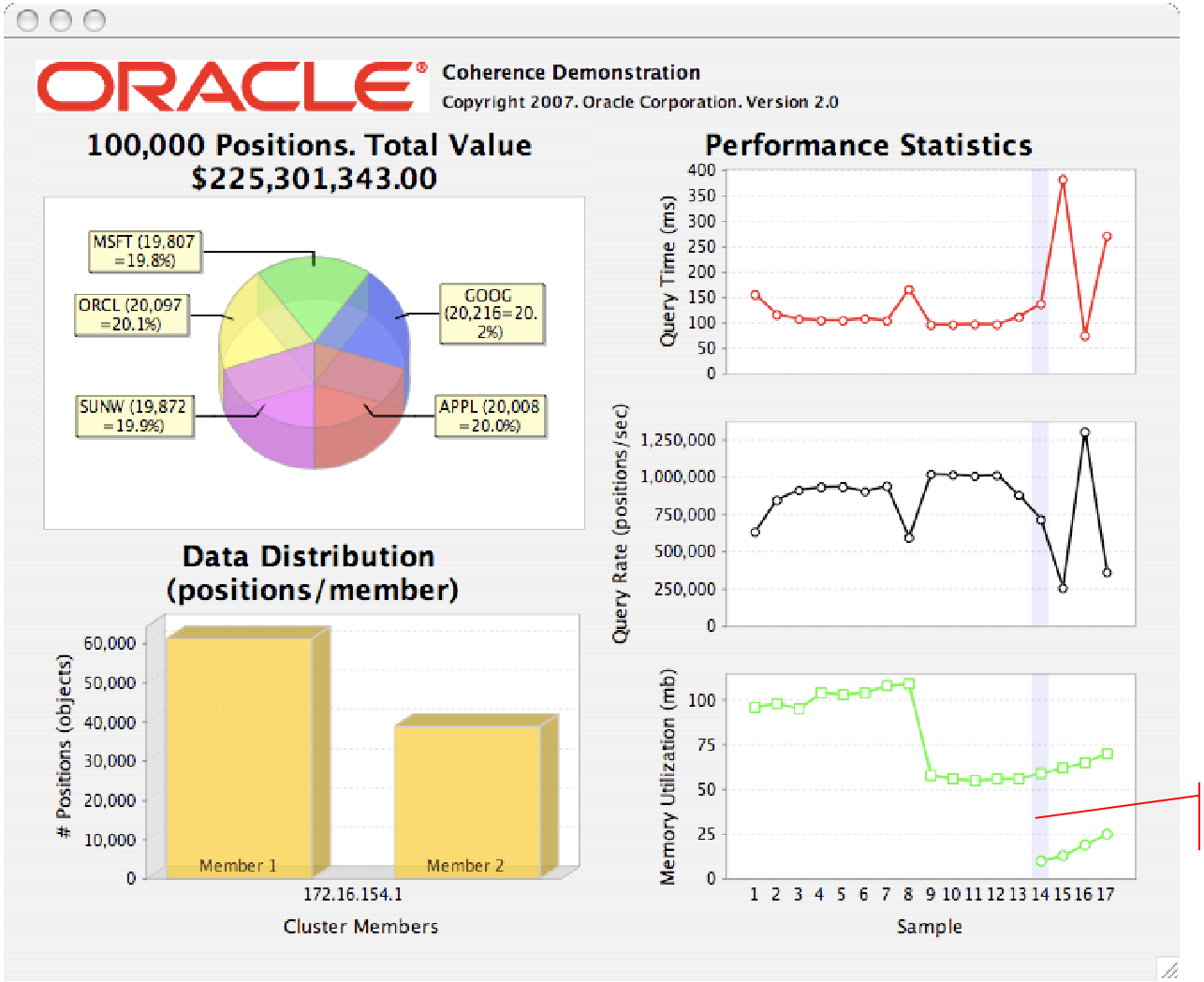
Automatic Load Balancing started…



Garbage Collection (memory management)

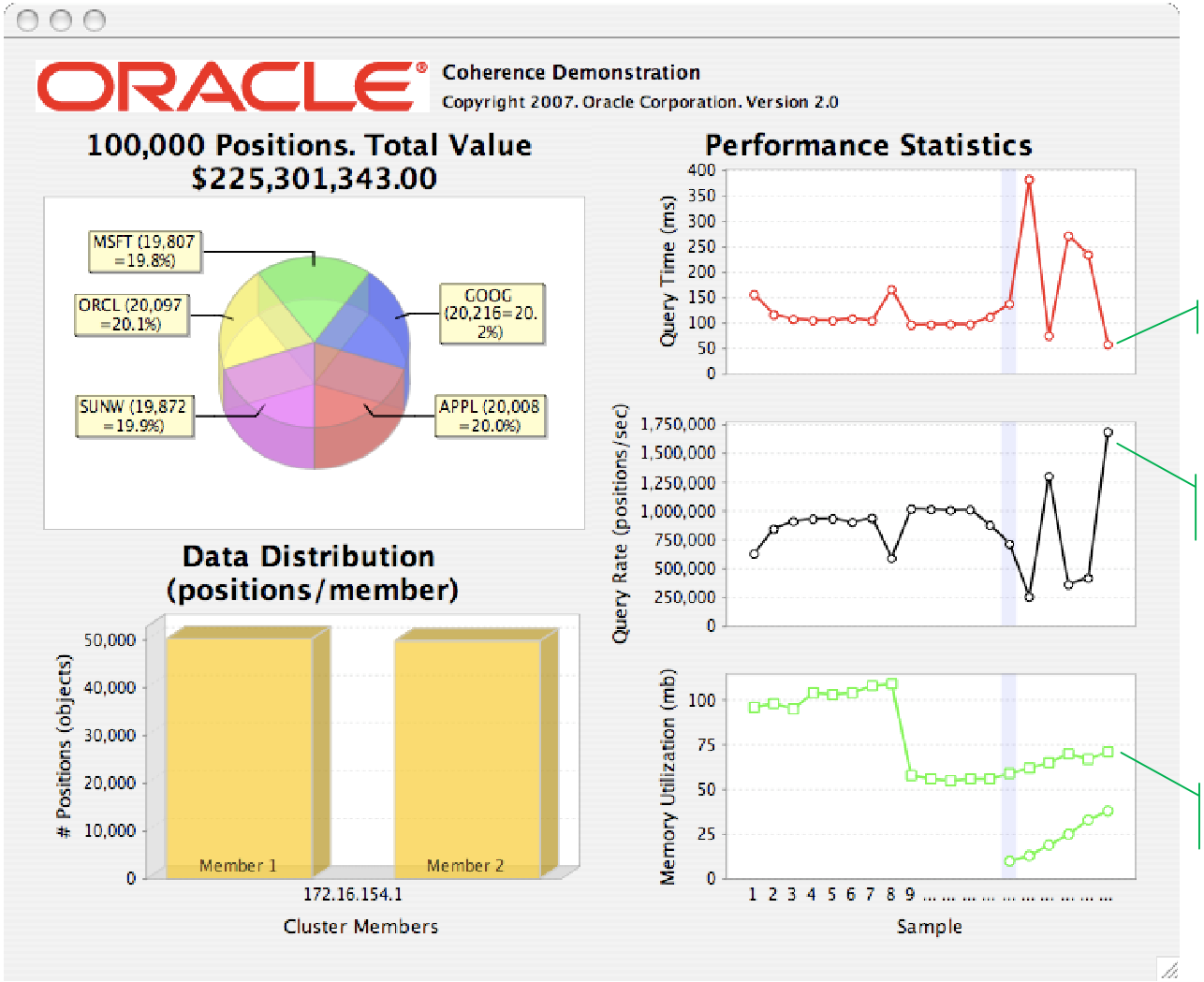
New Member Joined Cluster

Automatic Load Balancing in progress..



Cluster Changed

Data and Processing Scaled-Out!



Latency Halved

Throughput Doubled

GC pauses reduced

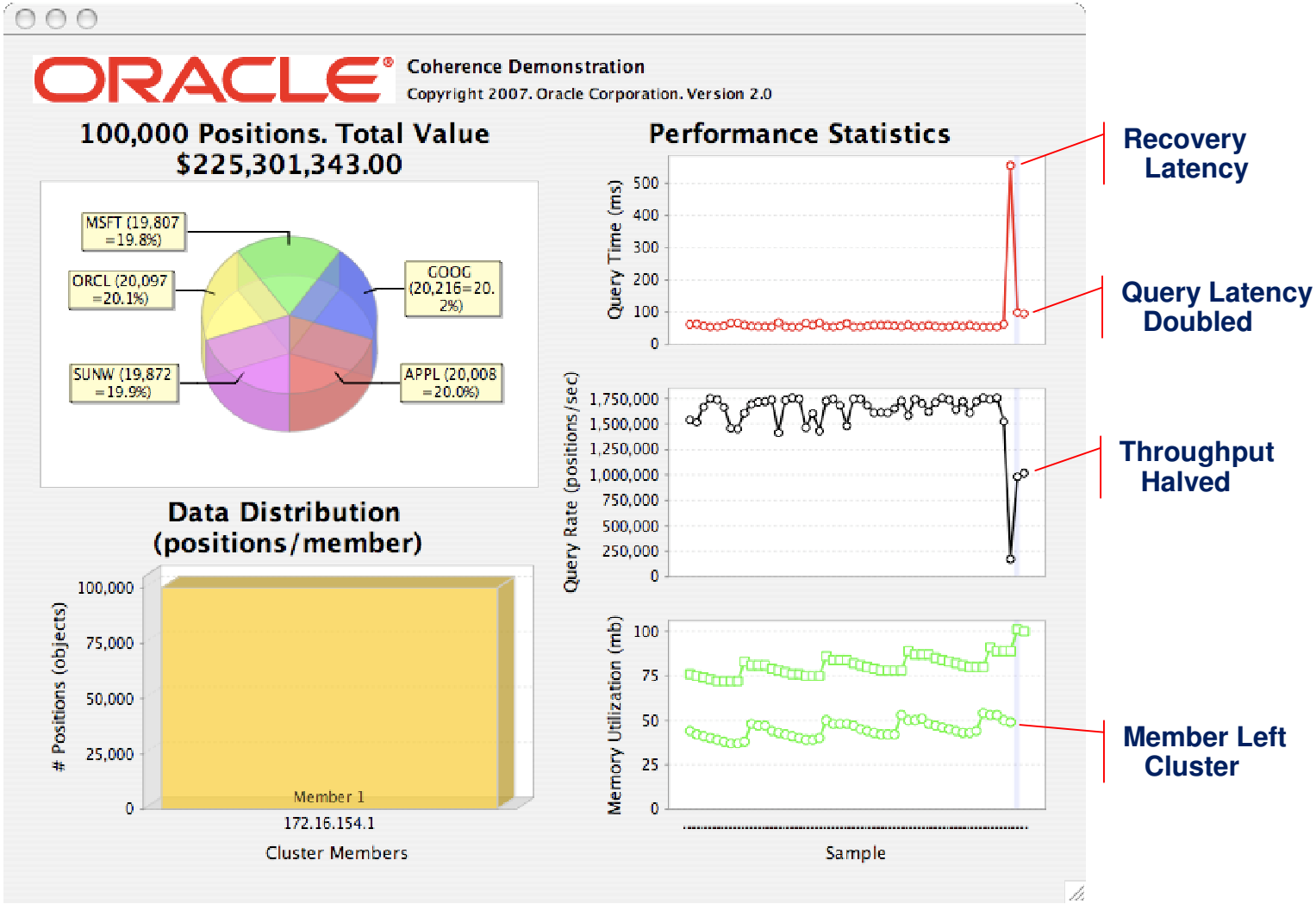
Killing a Cache Server (force data recovery)

Terminal — bash — 107x30

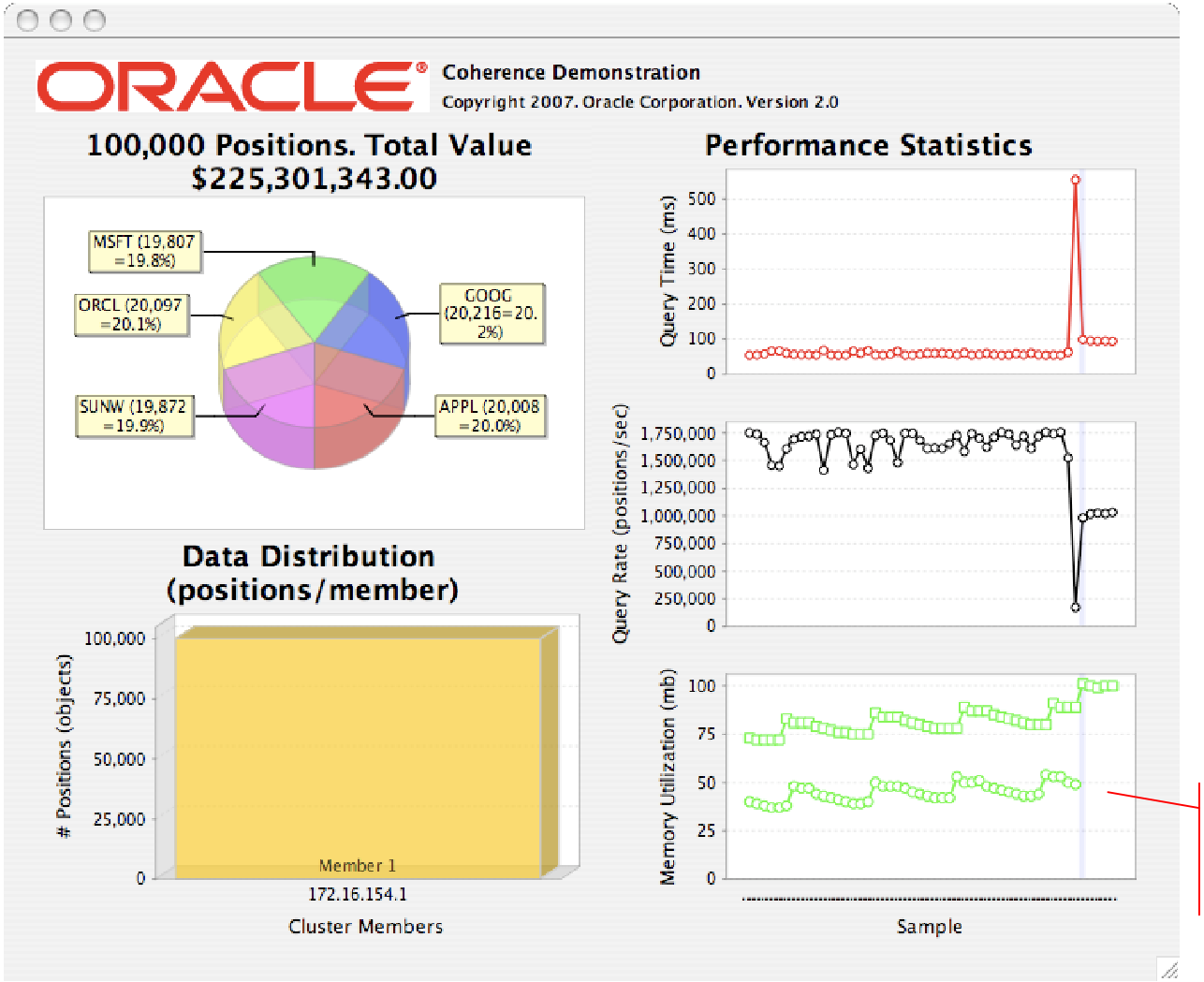
```
  (
  ThisMember=Member(Id=2, Timestamp=2007-10-31 11:08:54.836, Address=172.16.154.1:8090, MachineId=9729, Loc
ation=process:416@host94.msm.che.vodafone)
  OldestMember=Member(Id=1, Timestamp=2007-10-31 11:05:47.923, Address=172.16.154.1:8088, MachineId=9729, L
ocation=process:400@host94.msm.che.vodafone)
  ActualMemberSet=MemberSet(Size=3, BitSetCount=2
    Member(Id=1, Timestamp=2007-10-31 11:05:47.923, Address=172.16.154.1:8088, MachineId=9729, Location=pro
cess:400@host94.msm.che.vodafone)
    Member(Id=2, Timestamp=2007-10-31 11:08:54.836, Address=172.16.154.1:8090, MachineId=9729, Location=pro
cess:416@host94.msm.che.vodafone)
    Member(Id=4, Timestamp=2007-10-31 11:08:04.635, Address=172.16.154.1:8089, MachineId=9729, Location=pro
cess:411@host94.msm.che.vodafone)
    )
  RecycleMillis=120000
  RecycleSet=MemberSet(Size=0, BitSetCount=0
    )
  )

Services
  (
  TcpRing{TcpSocketAccepter{State=STATE_OPEN, ServerSocket=172.16.154.1:8090}, Connections=[1]}
  ClusterService{Name=Cluster, State=(SERVICE_STARTED, STATE_JOINED), Id=0, Version=3.3, OldestMemberId=1}
  DistributedCache{Name=DistributedCache, State=(SERVICE_STARTED), Id=1, Version=3.2, OldestMemberId=1, Loc
alStorage=enabled, PartitionCount=257, BackupCount=1, AssignedPartitions=0, BackupPartitions=0}
  ReplicatedCache{Name=ReplicatedCache, State=(SERVICE_STARTED), Id=2, Version=3.0, OldestMemberId=1}
  Optimistic{Name=OptimisticCache, State=(SERVICE_STARTED), Id=3, Version=3.0, OldestMemberId=1}
  InvocationService{Name=InvocationService, State=(SERVICE_STARTED), Id=4, Version=3.1, OldestMemberId=1}
  )

^Chost94:~/Desktop/coherence-swing-demo-2.0 boliver$
```

Brutal ^C to kill Cache Server

Data and Processing Scaled-Back



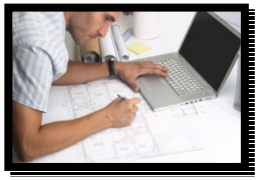
Continuous Availability



System still operational after server crash / death

Using Coherence

Clustering Java Processes



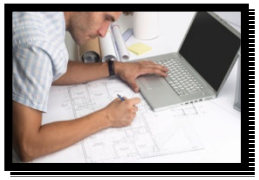
- Joins an existing cluster or forms a new cluster
  - Time "to join" configurable

```
Cluster cluster = CacheFactory.ensureCluster();
```

- `cluster` contains information about the Cluster
  - Cluster Name
  - Members
  - Locations
  - Processes

- No "master" servers
- No "server registries"

Using a Cache
get, put, size & remove

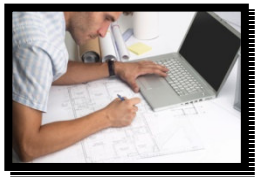


- `CacheFactory` resolves cache names (i.e.: "`mine`") to configured `NamedCache`s
- `NamedCache` provides data topology agnostic access to information
- `NamedCache` interfaces implement several interfaces;
  - java.util.Map, Jcache, ObservableMap*, ConcurrentMap*, QueryMap*, InvocableMap

Coherence* Extensions

```
NamedCache nc = CacheFactory.getCache("mine");

Object previous = nc.put("key", "hello world");

Object current = nc.get("key");

int size = nc.size();

Object value = nc.remove("key");
```

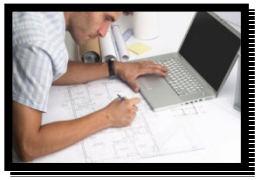
Using a Cache
keySet, entrySet, containsKey



- Using a `NamedCache` is like using a java.util.Map
- What is the difference between a Map and a Cache data-structure?
  - Both use (key,value) pairs for entries
  - Map entries don't expire
  - Cache entries may expire
  - Maps are typically limited by heap space
  - Caches are typically size limited (by number of entries or memory)
  - Map content is typically in-process (on heap)

```
NamedCache nc = CacheFactory.getCache("mine");

Set keys = nc.keySet();

Set entries = nc.entrySet();

boolean exists = nc.containsKey("key");
```

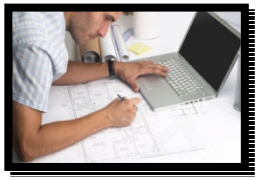
Querying Caches
QueryMap



- Query `NamedCache` keys and entries across a cluster (Data Grid) in parallel* using Filters
- Results may be ordered using natural ordering or custom comparators
- Filters provide support almost all SQL constructs
- Create your own Filters

```
NamedCache nc = CacheFactory.getCache("people");

Set keys = nc.keySet(
                 new LikeFilter("getLastName",
                                "%Stone%"));
Set entries = nc.entrySet(
                 new EqualsFilter("getAge",
                                  35));
```

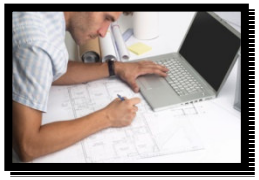
Aggregating Information
InvocableMap



- Aggregate values in a `NamedCache` across a cluster (Data Grid) in parallel* using Filters
- Aggregation constructs include; Distinct, Sum, Min, Max, Average, Having, Group By
- Create your own aggregators

```
NamedCache nc = CacheFactory.getCache("stocks");


Double total = (Double)nc.aggregate(
    AlwaysFilter.INSTANCE,
    new DoubleSum("getQuantity"));

Set symbols = (Set)nc.aggregate(
    new EqualsFilter("getOwner", "Larry"),
    new DistinctValue("getSymbol"));
```

Mutating Information
InvocableMap

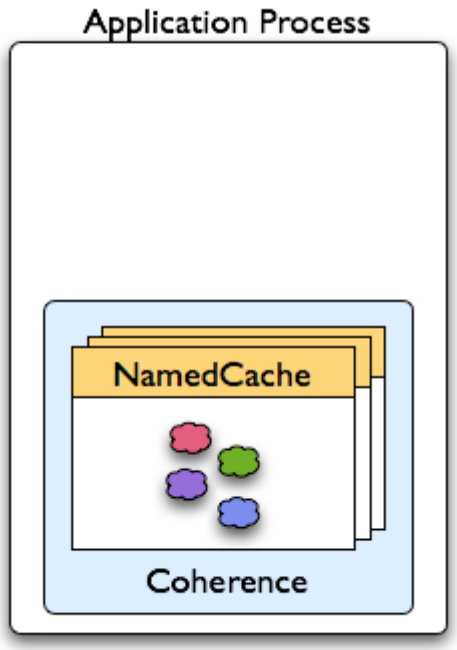


- Invoke `EntryProcessors` on zero or more entries in a `NamedCache` across a cluster (Data Grid) in parallel* (using Filters) to perform operations
- Execution occurs where the entries are managed in the cluster, not in the thread calling `invoke`
- This permits **Data + Processing Affinity**

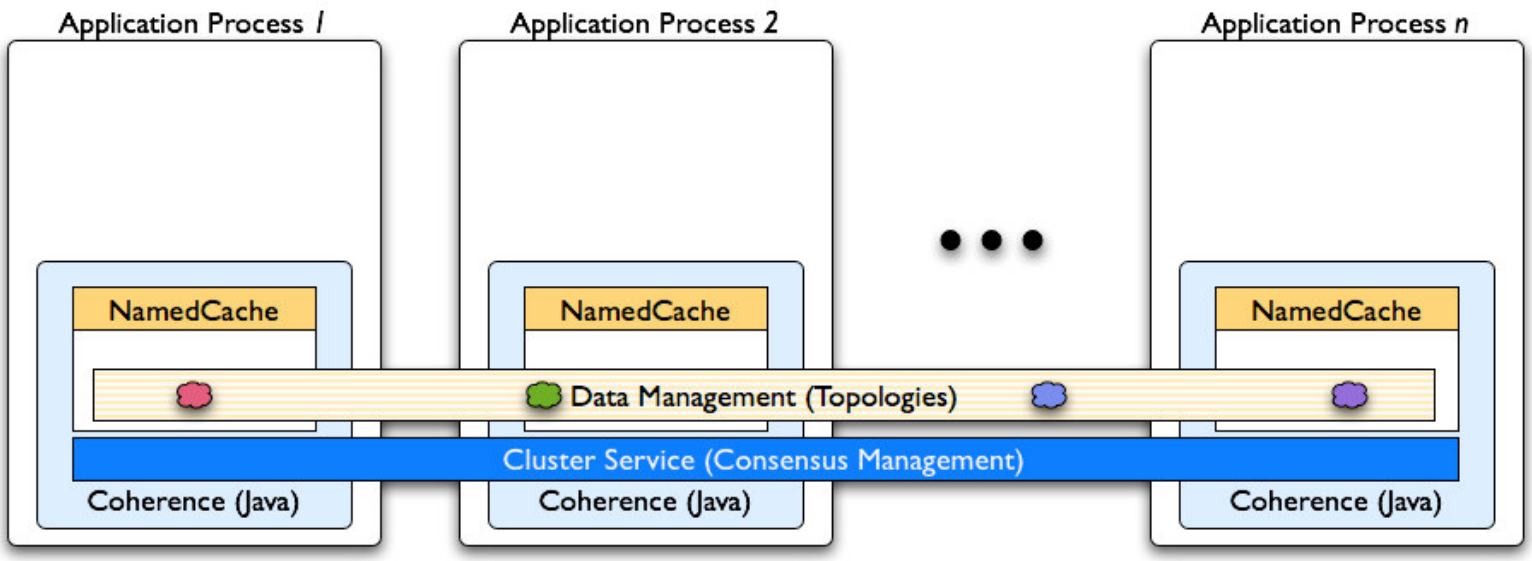
```
NamedCache nc = CacheFactory.getCache("stocks");


nc.invokeAll(
    new EqualsFilter("getSymbol", "ORCL"),
    new StockSplitProcessor());


 ...



class StockSplitProcessor extends
      AbstractProcessor {

    Object process(Entry entry) {
        Stock stock = (Stock)entry.getValue();
        stock.quantity *= 2;
        entry.setValue(stock);
        return null;
    }

}
```

Topologies and examples of Coherence architectures

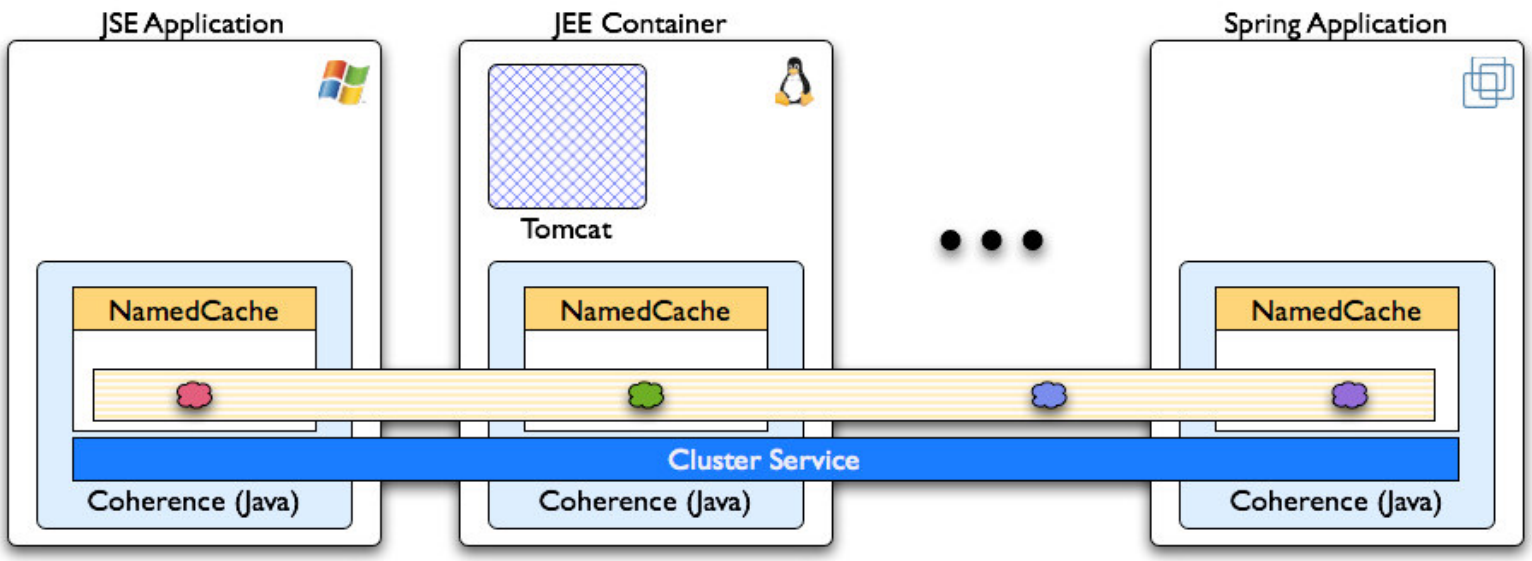
Single Application Process



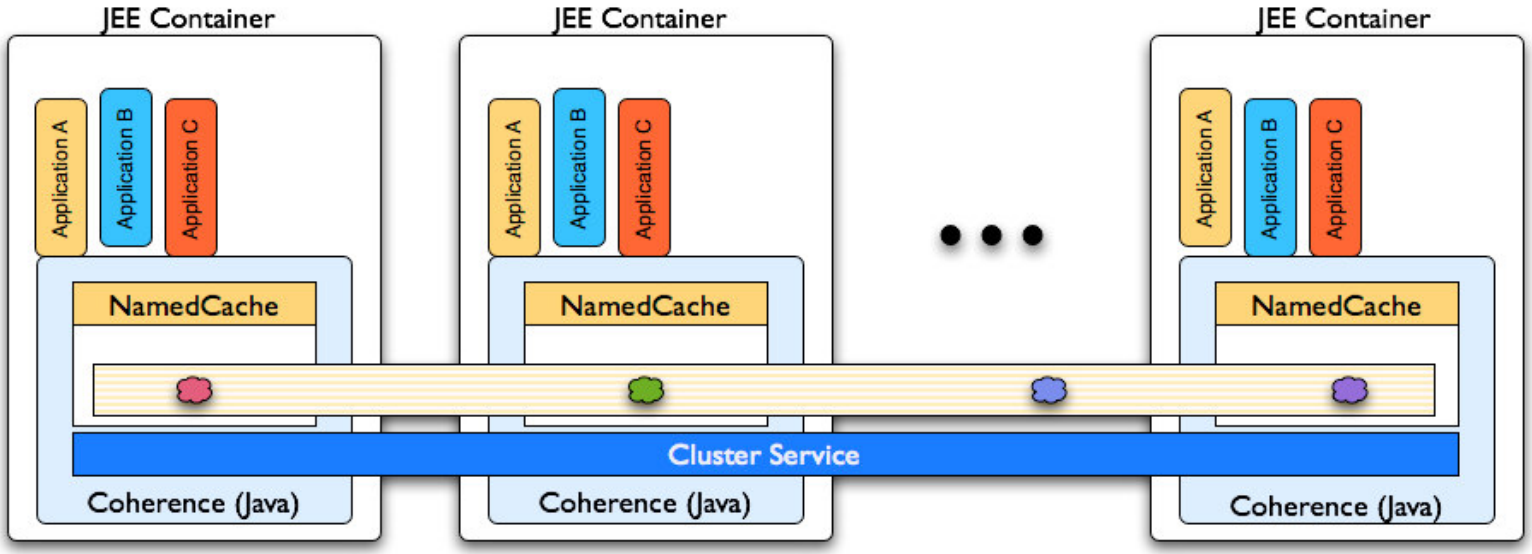
Clustered Processes



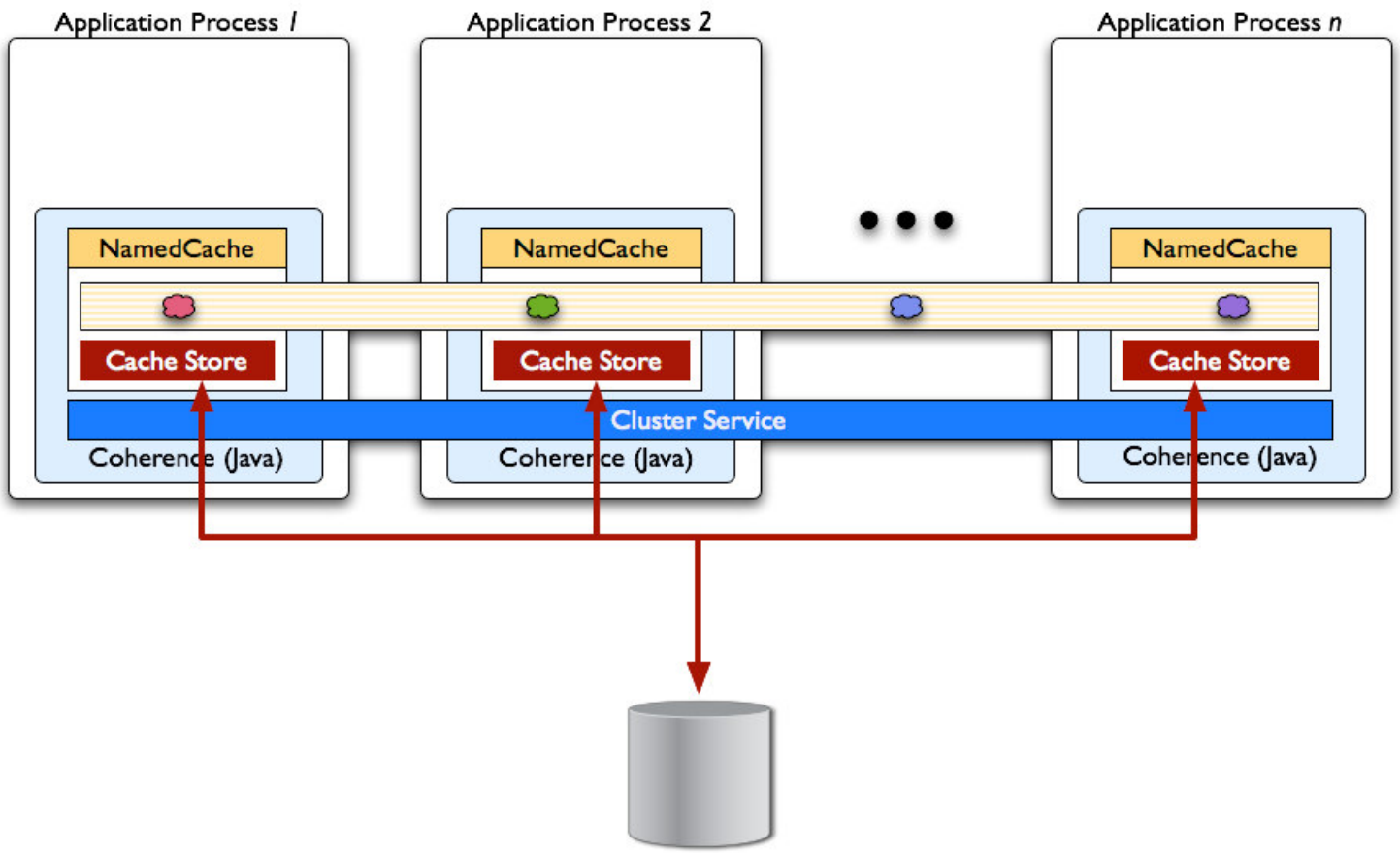
Multi Platform Cluster



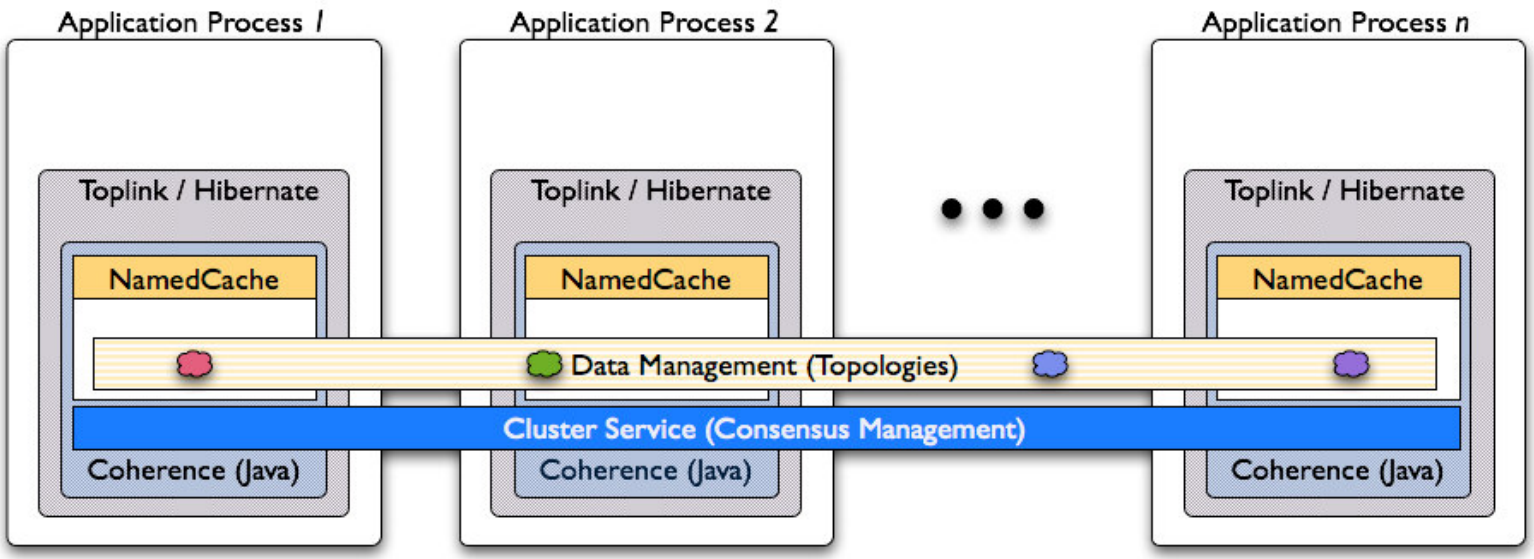
Clustered Application Servers



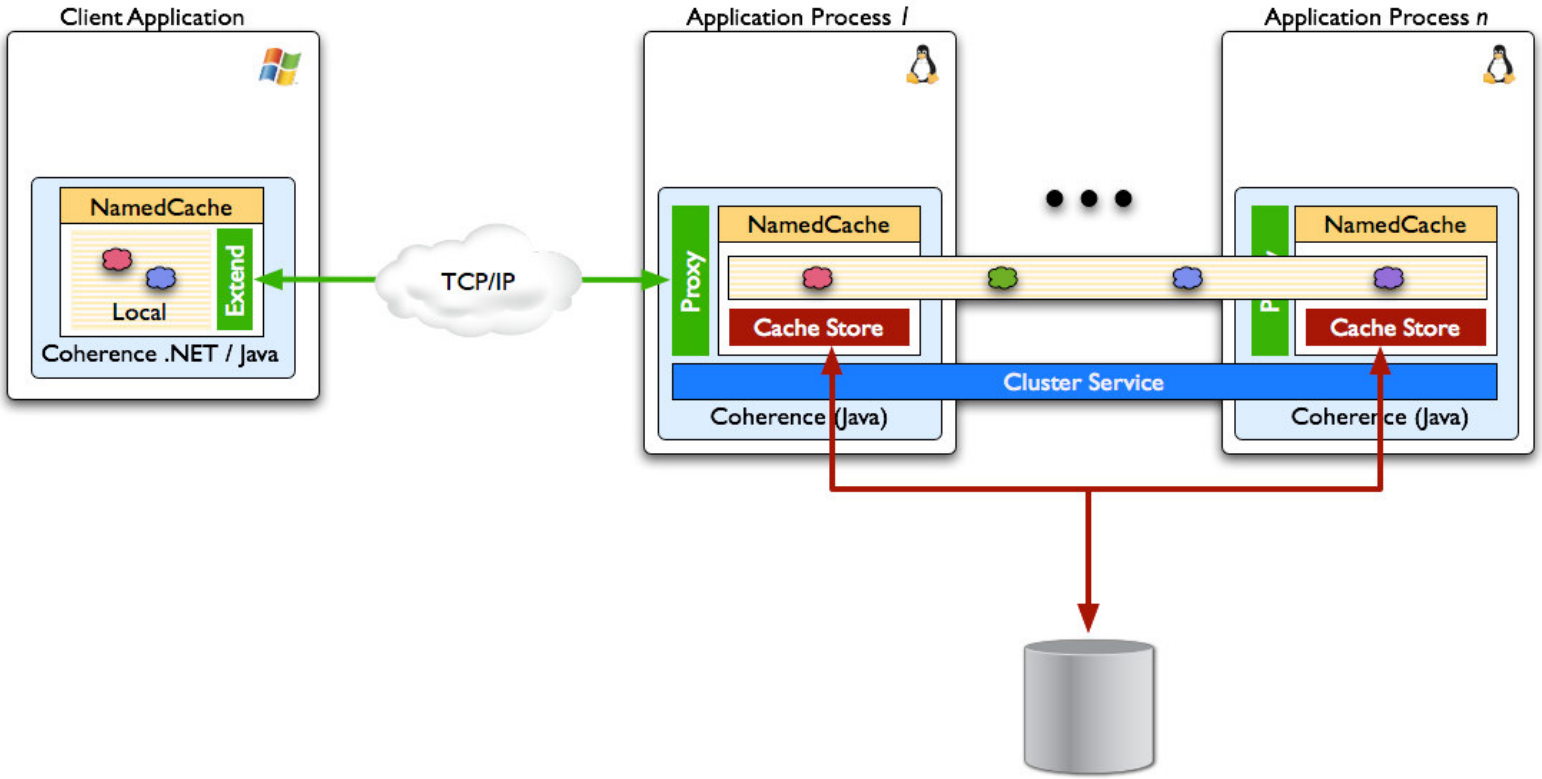
With Data Source Integration
(Cache Stores)



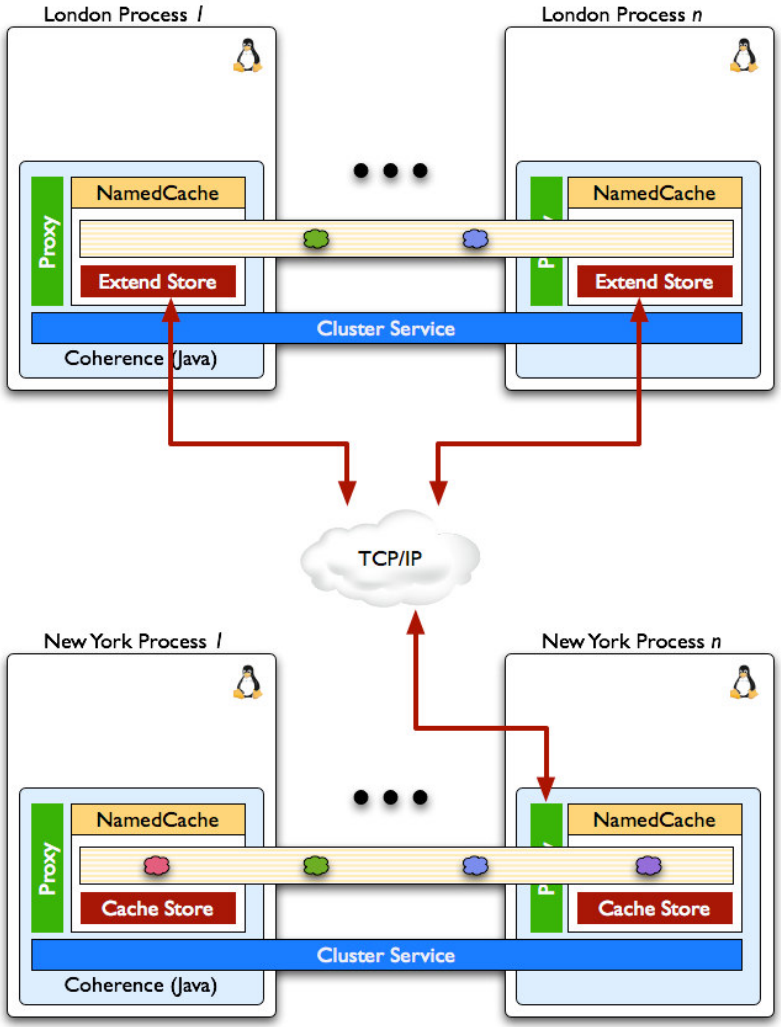
Clustered Second Level Cache
(for Hibernate)



Remote Clients connected to Coherence Cluster



Interconnected WAN Clusters



Distributed Data Management

- Members have logical access to all Entries
  - At most 2 network operations for Access
  - At most 4 network operations for Update
  - Regardless of Cluster Size
  - Deterministic access and update behaviour (performance can be improved with local caching)

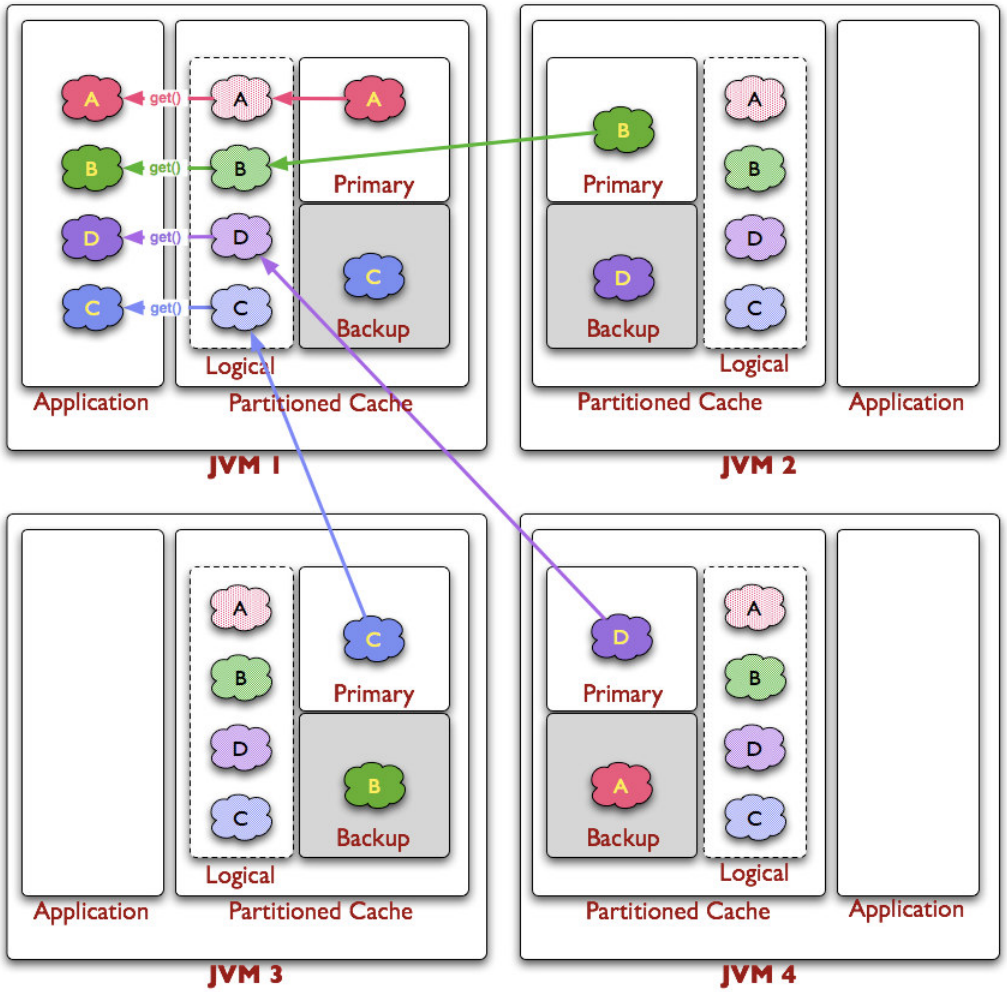
- Predictable Scalability
  - Cache Capacity Increases with Cluster Size
  - Coherence Load-Balances Partitions across Cluster
  - Point-to-Point Communication (peer to peer)
  - No multicast required (sometimes not allowed)

Distributed Data Management (access)

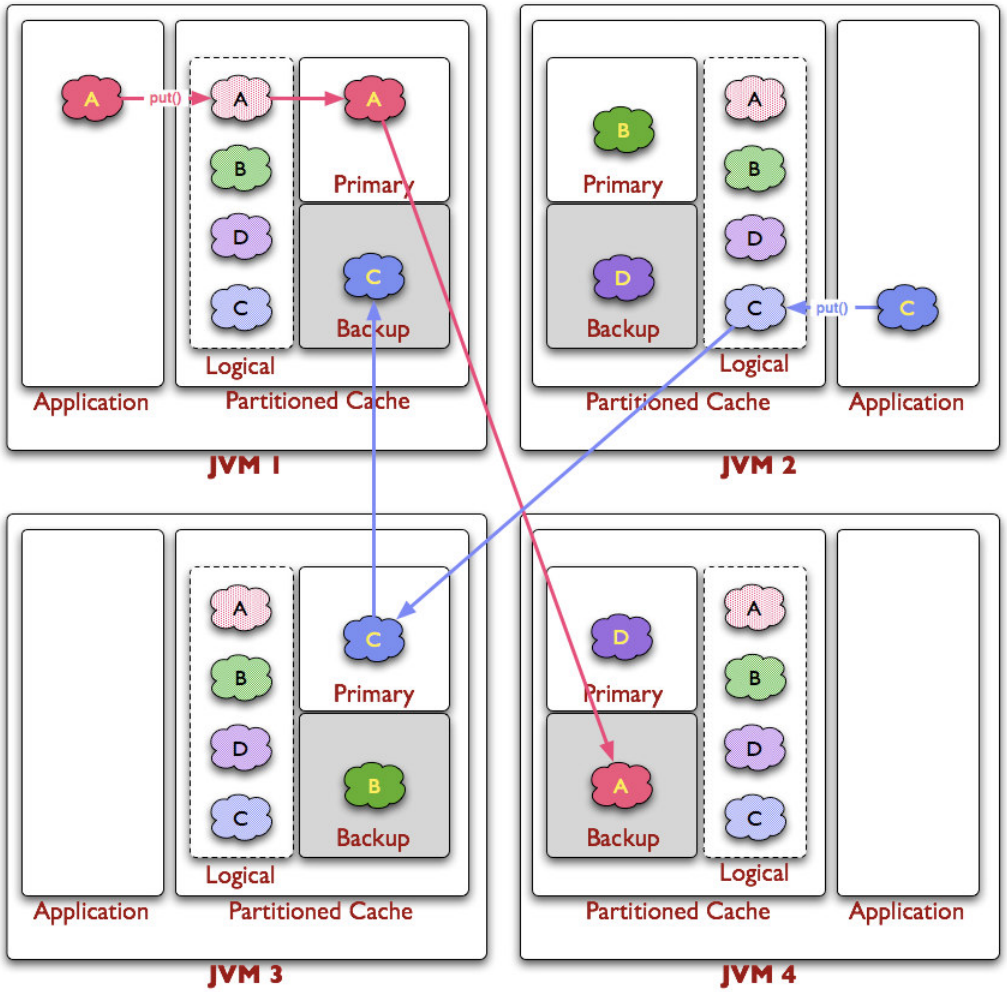
The Partitioned Topology

(one of many)

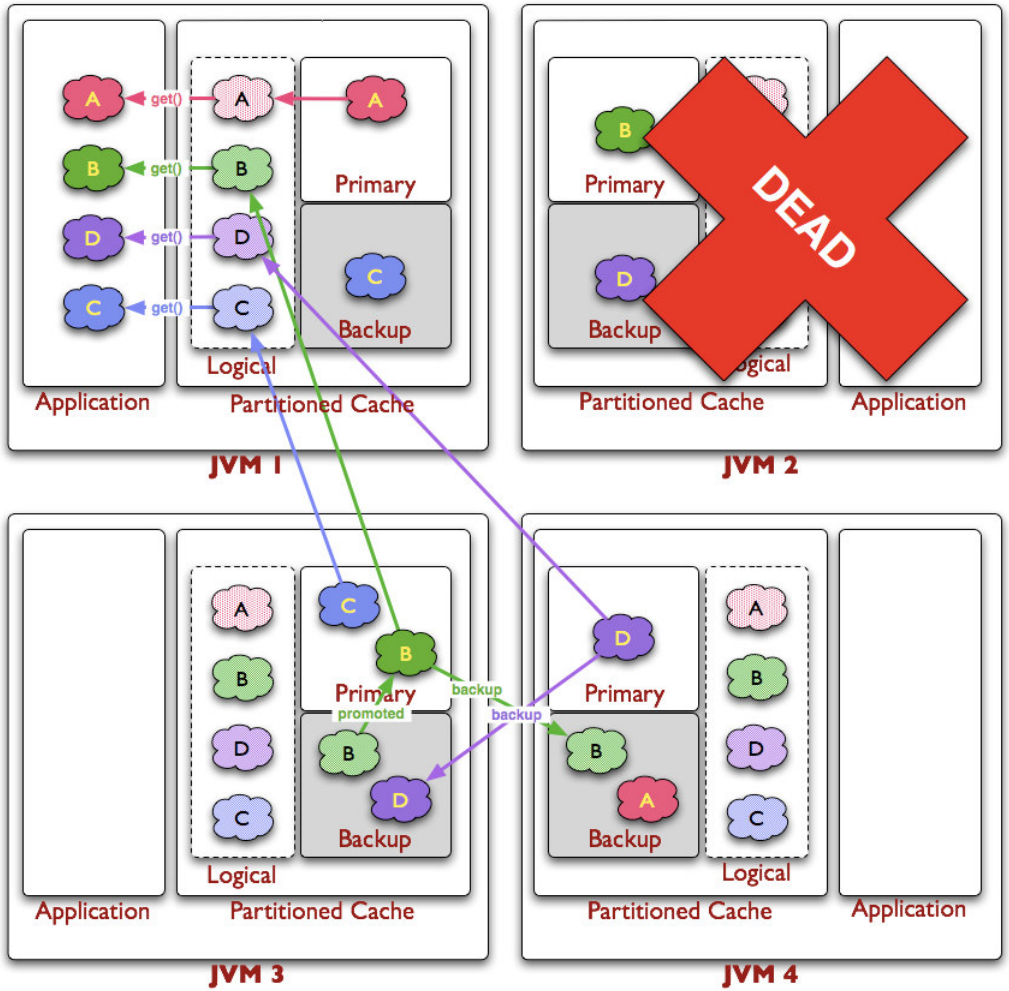
In-Process Data Management



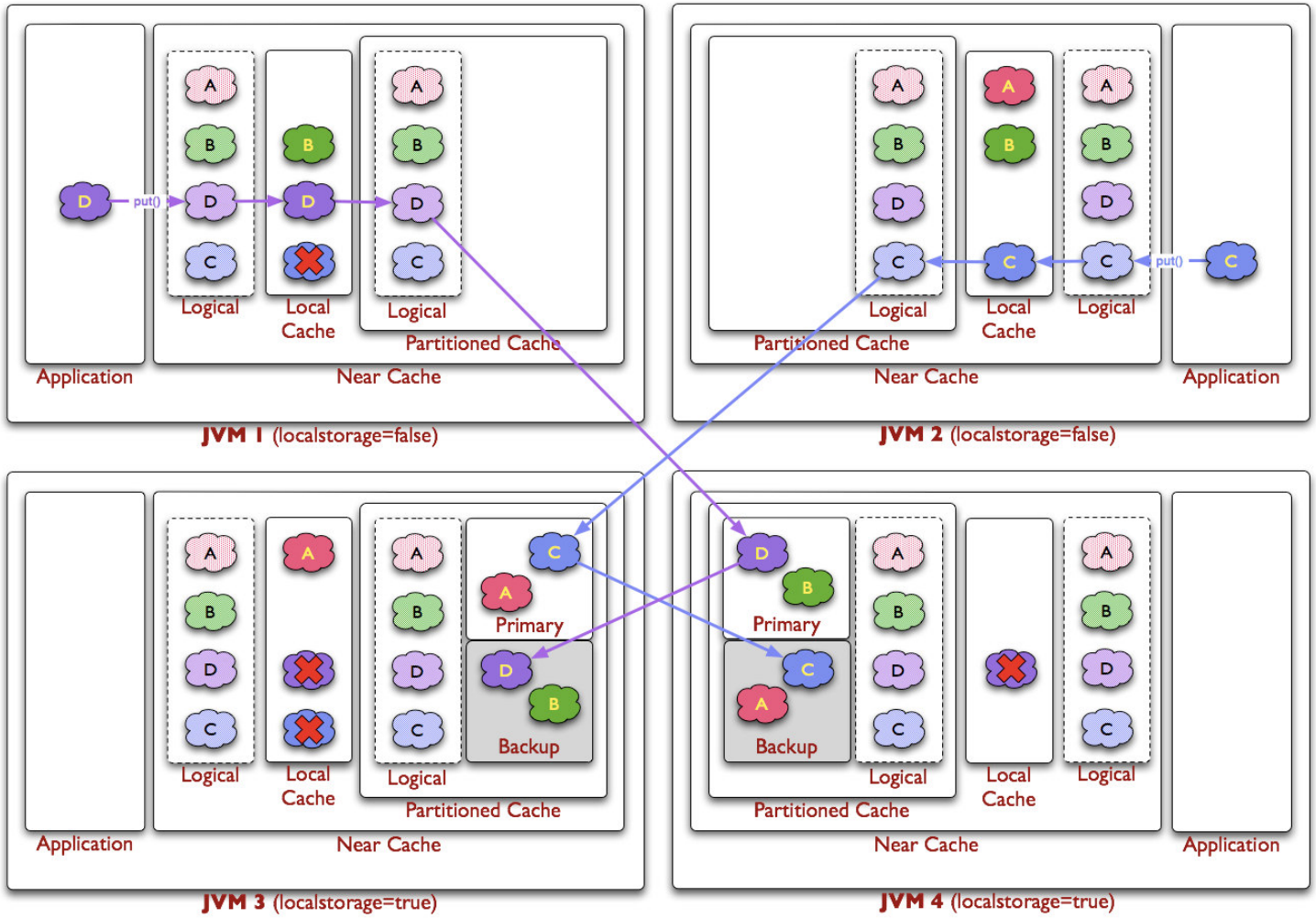
Distributed Data Management (update)



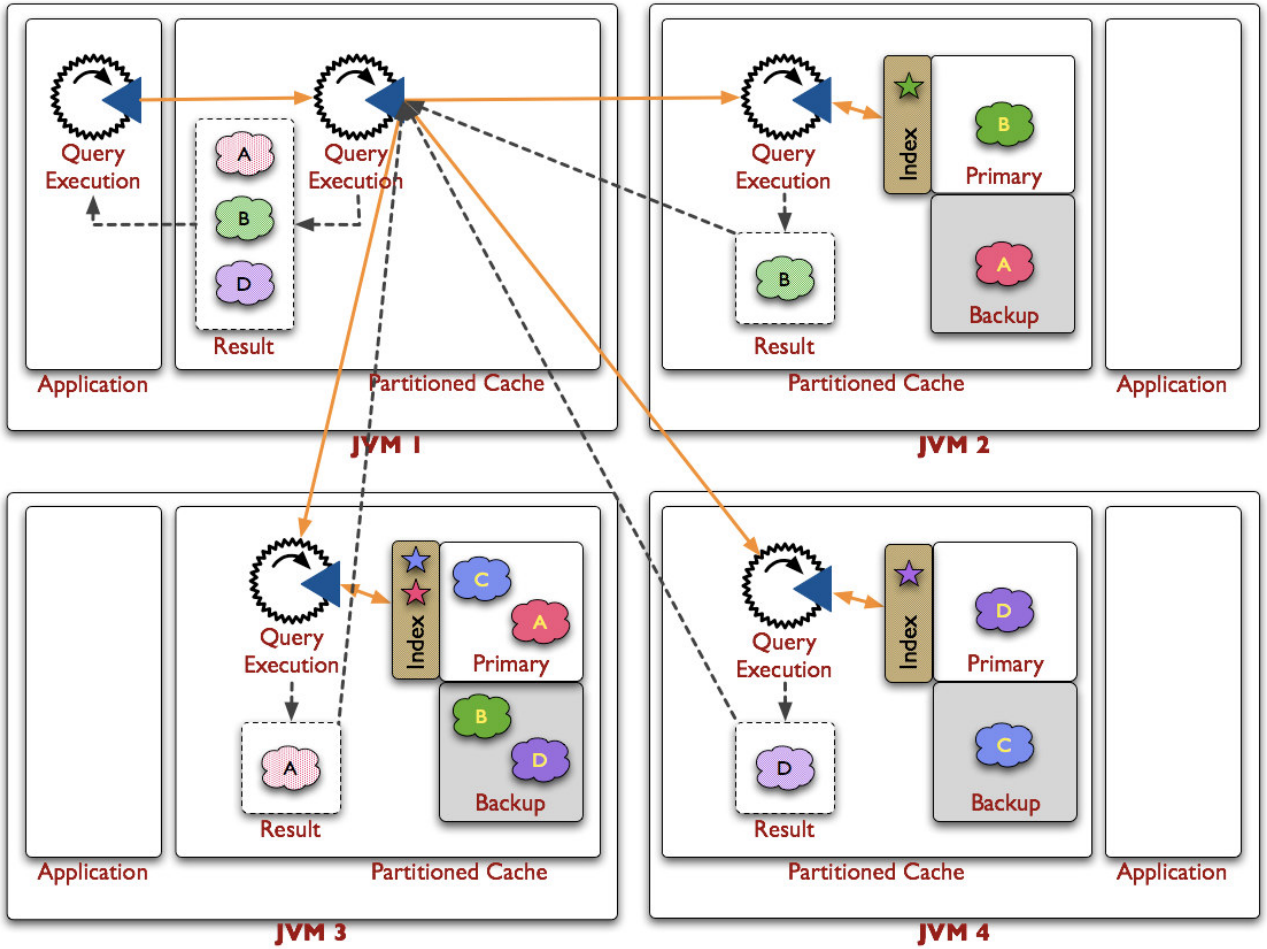
Distributed Data Management (failover)



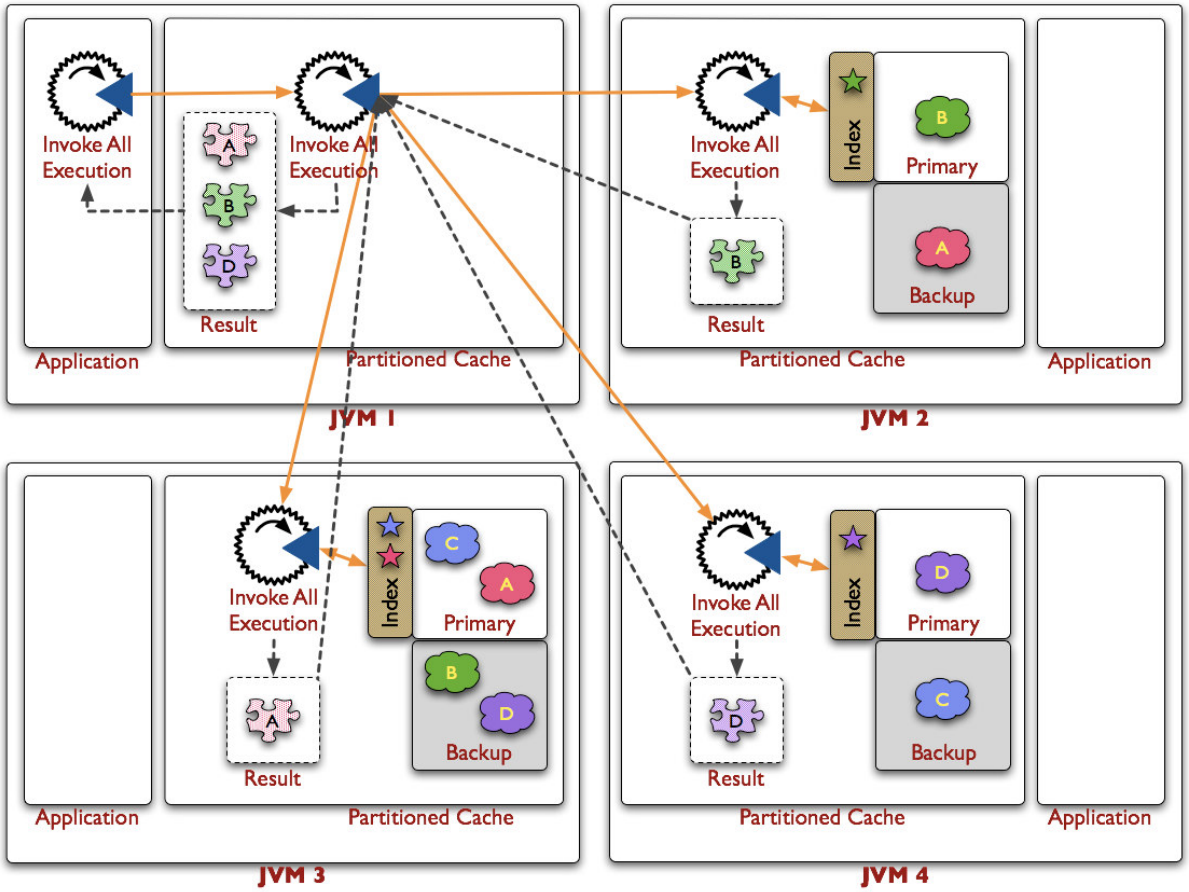
Near Caching (L1 + L2) Topology



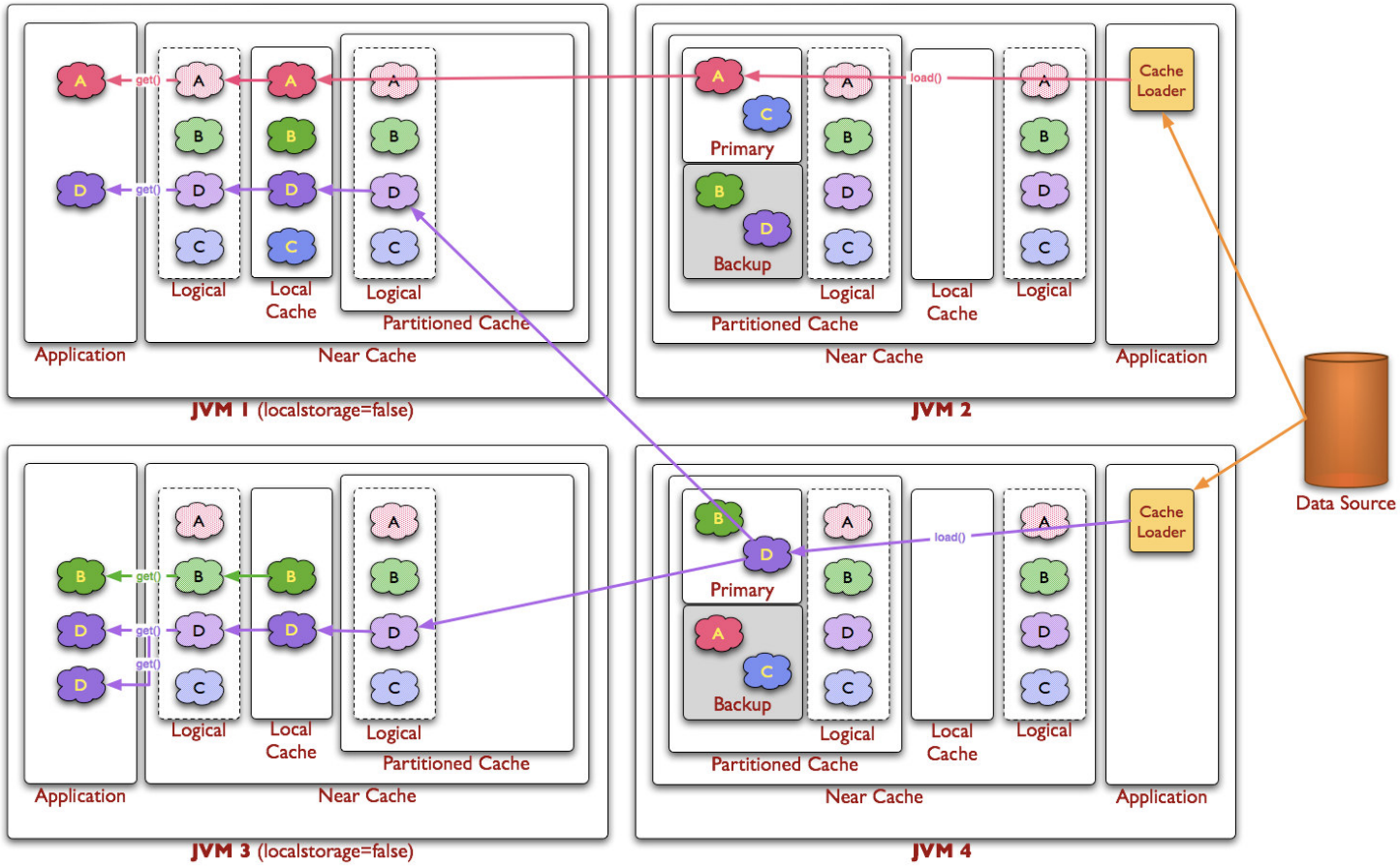
Parallel Queries



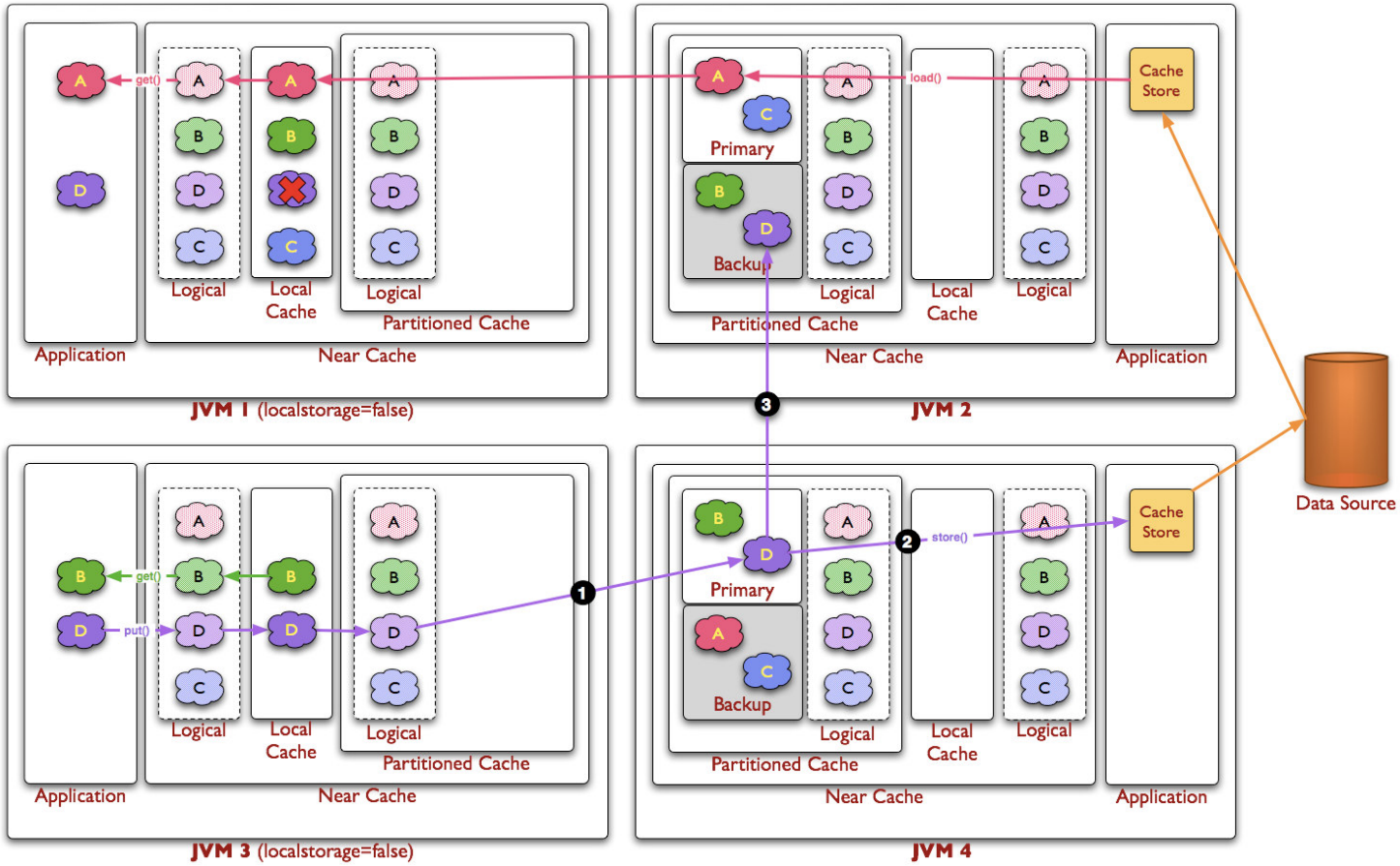
Parallel Processing and Aggregation



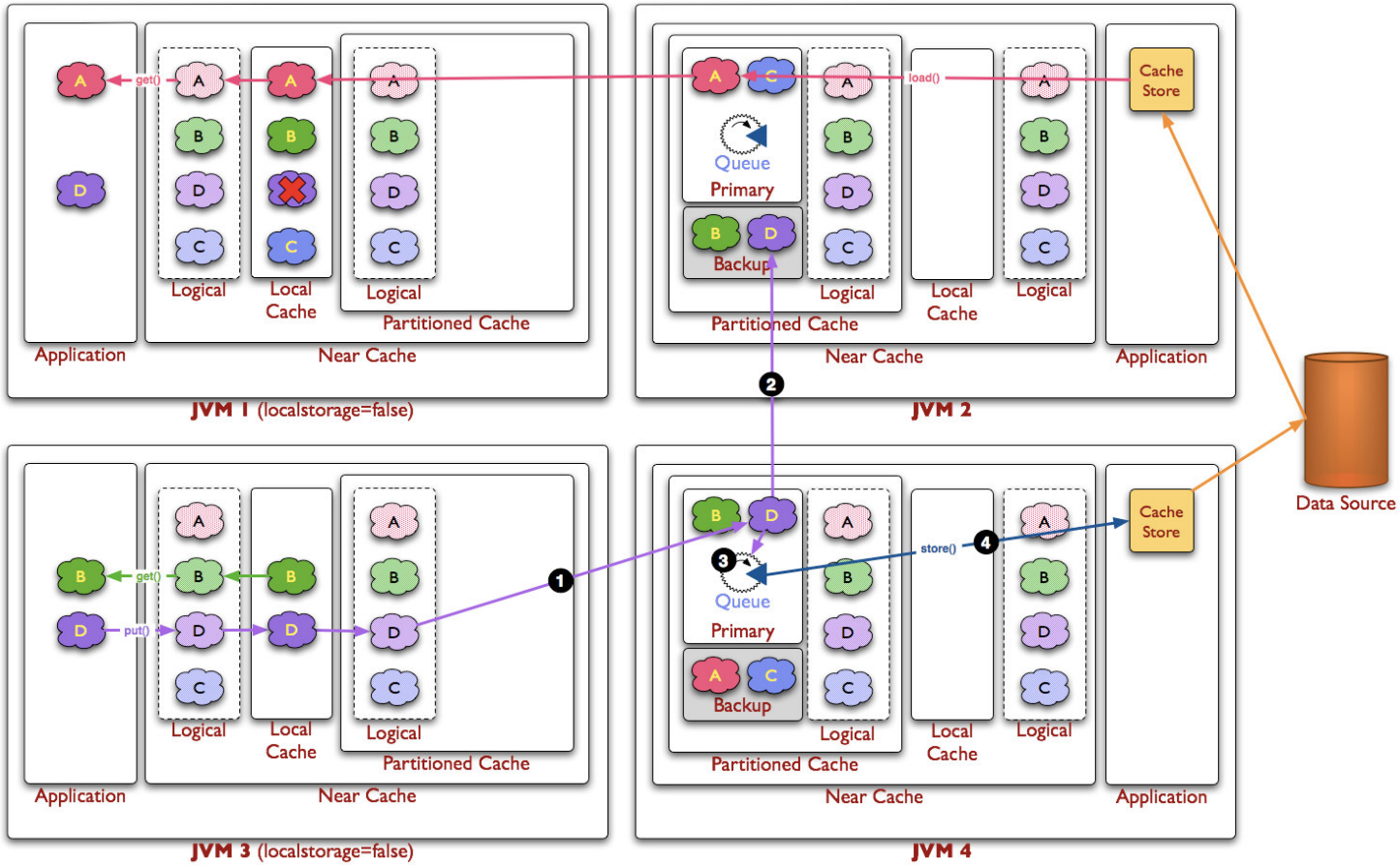
Data Source Integration (read-through)



Data Source Integration (write-through)



Data Source Integration (write-behind)



Where is Coherence used?

Some scenarios

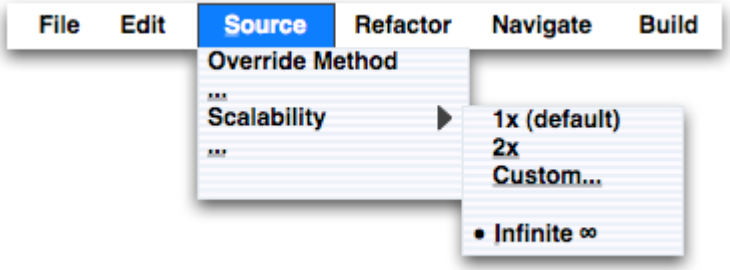
- Together with frameworks like...
  - Hibernate
  - TopLink
  - Spring
- Used by ISVs as integrated technology
- As part of applications (old, new)
- ?

Some real-life examples

- Financial systems
  - Trading
  - Insurance applications
- On line gambling / betting
- Manufacturing / planning / production
- CAD/CAM systems
- On-Line travel booking

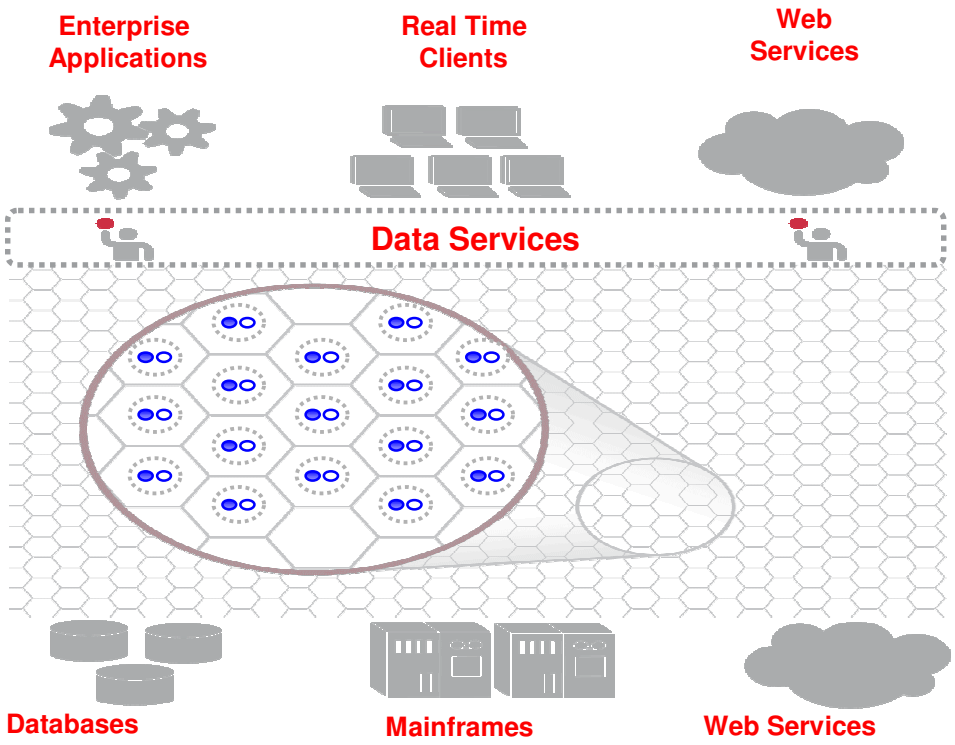
In Summary

- Scaling the Application-Tier is difficult
- If it was easy it would be an IDE option



- Scalability is a design option
  - Requires knowledge, care and experience
  - Developers have the “option” to consider building it in!
  - It’s not an IDE option
- Coherence is scalability infrastructure for the application-tier

Thank You!


Enterprise Applications
Real Time Clients
Web Services
Data Services
Databases
Mainframes
Web Services