
ORACLE®

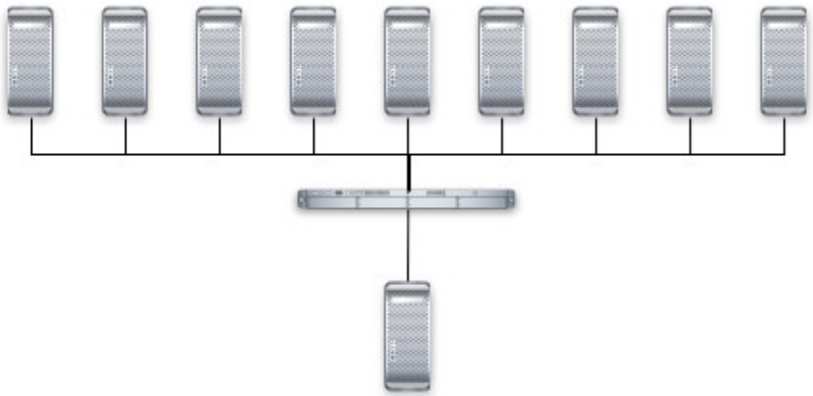
ORACLE®

eXtreme Transaction Processing:
Oracle Coherence Data Grid

Cameron Purdy
Vice President of Development
Oracle

What's so extreme about it?

Middleware-Based Transaction Processing



- Java EE provides a set of widely adopted standards for transaction processing
  - EJB, JTA, JMS+MDB
- C#/.NET + MTS, Tuxedo, WS*, ..
- PL/SQL
- *"RAC in the Back"*

Time-tested development models …

Why not Java EE?

Oracle OC4J Leadership in Transaction Processing Rates

SPECjAppServer2004	JOPS	Date	Result
HP RX2660, Single Node, HP-UX	219/Core	May 2007	World Record, JOPS/Core
Proliant BL685, Single Node, Linux	125/Core	May 2007	World Record, JOPS/Core x86-64 AMD8220
HP RX3600 11 Nodes, HP-UX	6812	Dec 2006	World Record

SPECjAppServer2002	TOPS	Date	Result
Fujitsu PrimePower 450/2500, Solaris	5,991	Mar 2005	World Record, Multiple Node

SPECjAppServer2001	BOPS	Date	Result
HP RP8400 Cluster, HP-UX	2,529	Apr 2003	World Record, Multiple Node
Sun SunFire V1280m, Solaris	521	Oct 2002	World Record, Dual Node

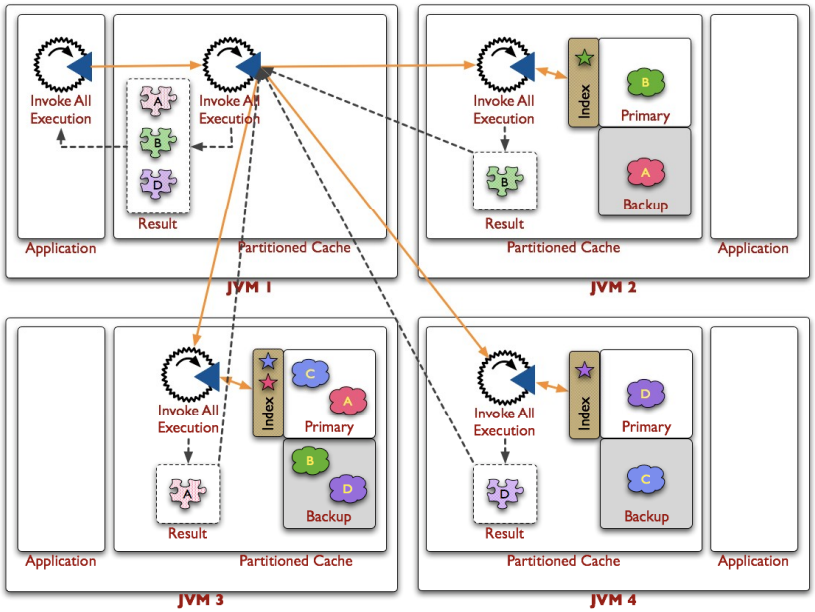
ECPerf	BBops/Min	Date	Result
Sun SunFire 3800, Solaris	61,682	Jul 2002	World Record, Dual Node

ORACLE

Reference: http://www.oracle.com/solutions/performance_scalability/appserver-1206.html

What's so extreme about it?

Grid-Based Transaction Processing

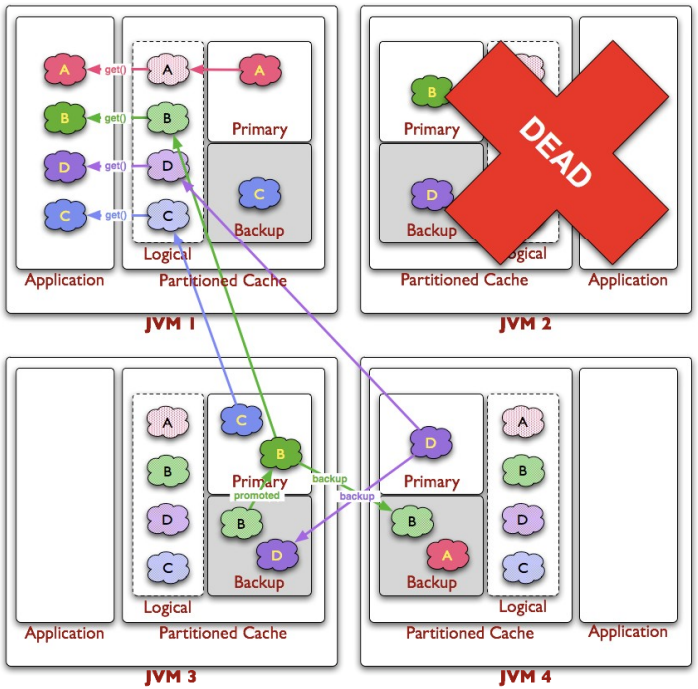


- There is a *Scalability Chasm*
  - Not an incremental solution
- Extreme Transaction Volumes
  - Sustained rates of over one million TPS on commodity blade servers
- Stock exchanges, utilities, banks, and the world's busiest websites such as FedEx.com

Rethinking high scale architectures …

What's so extreme about it?

… without sacrificing Quality of Service



- Which of these should be optional for your transactional infrastructure?
  - Continuous Availability
  - Information Reliability
  - Incremental Scalability
  - Predictable Performance
- "It's your data"

XTP Requires a Bullet-Proof Infrastructure

Gartner: eXtreme Transaction Processing

A Rapidly Growing Computing Paradigm

"Transaction processing has been well understood for decades. Yet, advanced service-oriented architecture, multi-channel,

Internet-enabled business models will push transactional requirements to the extreme. Extreme TP will dramatically affect technologies, vendor strategies and user architectures"

Gartner ***August 2006*** *(1)*

"Distinctive of Coherence distributed caching platform is that it can be used to support multiple scenarios, including extreme transaction processing, event driven architectures (EDAs) and analytical compute-intensive applications"

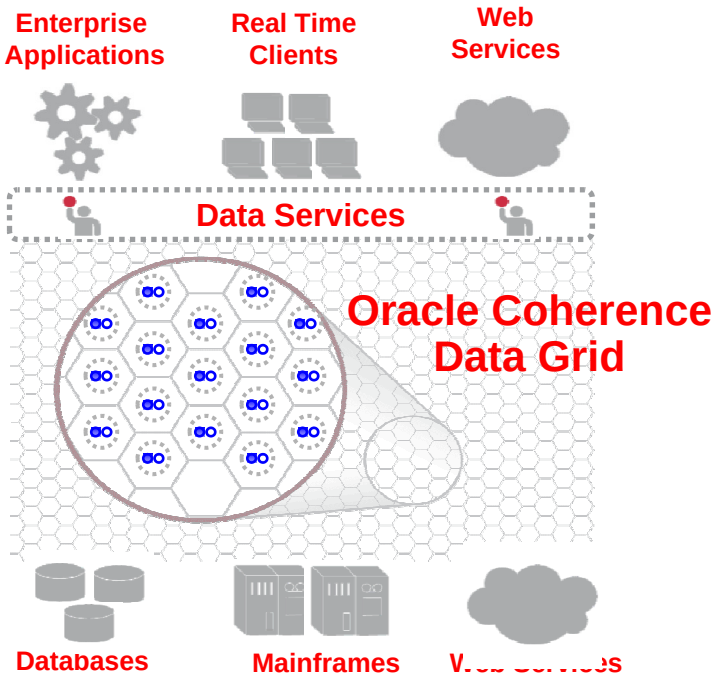
Gartner ***March 2007*** *(2)*

ORACLE

1. Gartner, The Challenges of Extreme Transaction Processing in a World of Services and Events, August 31 2006
2. Gartner, Cool Vendors in Integration and Application Platforms 2007

Oracle Coherence Data Grid

Distributed in Memory Data Management

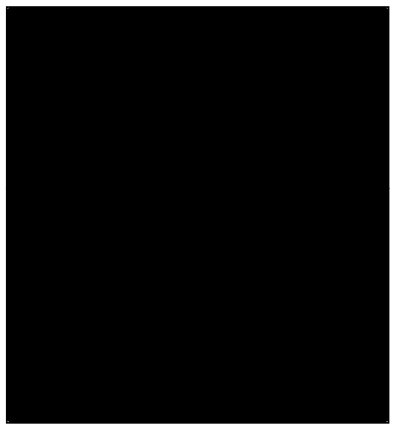


- Provides a **reliable data tier** with a single, consistent view of data
- Enables dynamic data capacity including **fault tolerance** and load balancing
- Ensures that **data capacity scales** with processing capacity

Coherence Quotes That We Didn't Pay For

Coherence … ensures data is "closer" to the applications issuing transactions against one or more databases/data stores … The result is almost linear scalability from 2 million to more than 60 million "aggregations" per second, according to a joint investment-bank benchmark

February 2007



Top 10 Product in Network World Next Generation Data Center Product Review

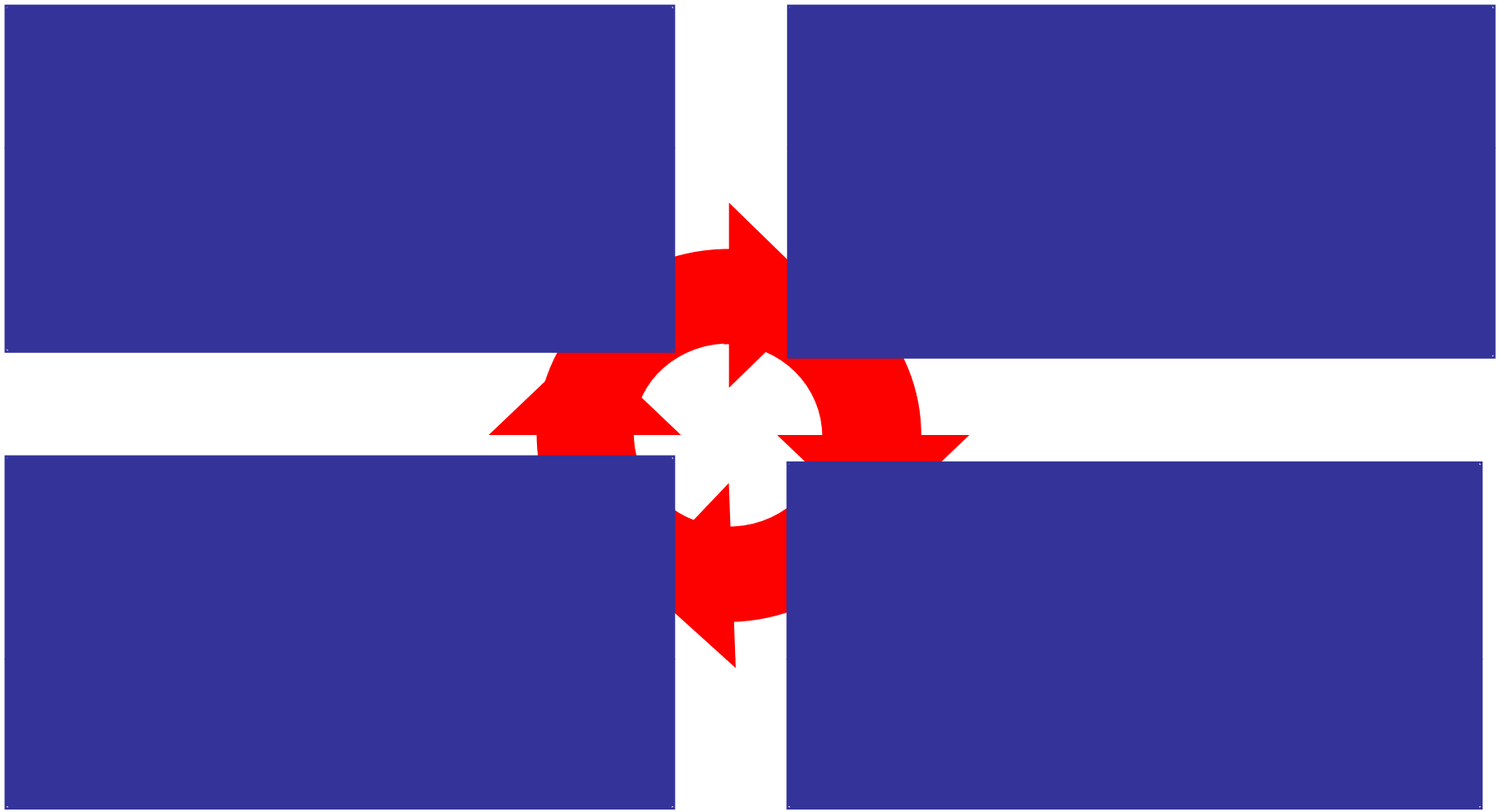
'With Coherence, performance has improved by as much as 100 times …'

Network World, March 2007

Oracle Coherence Select Customers

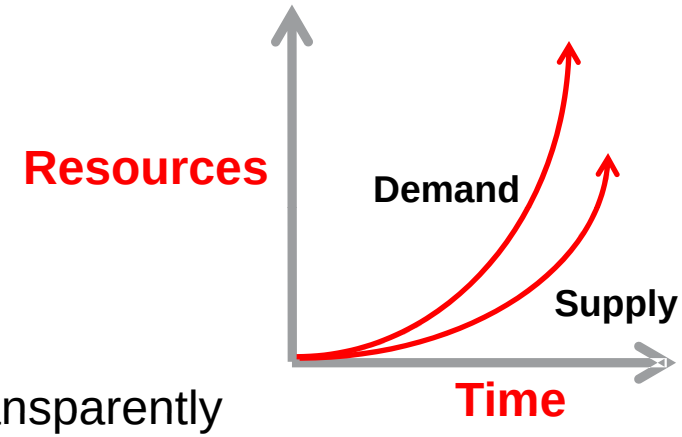
100s of Direct Customers, 1000s of Production Installs

Crossing the Architectural Chasm



Extreme? Whatever. Why should I care?

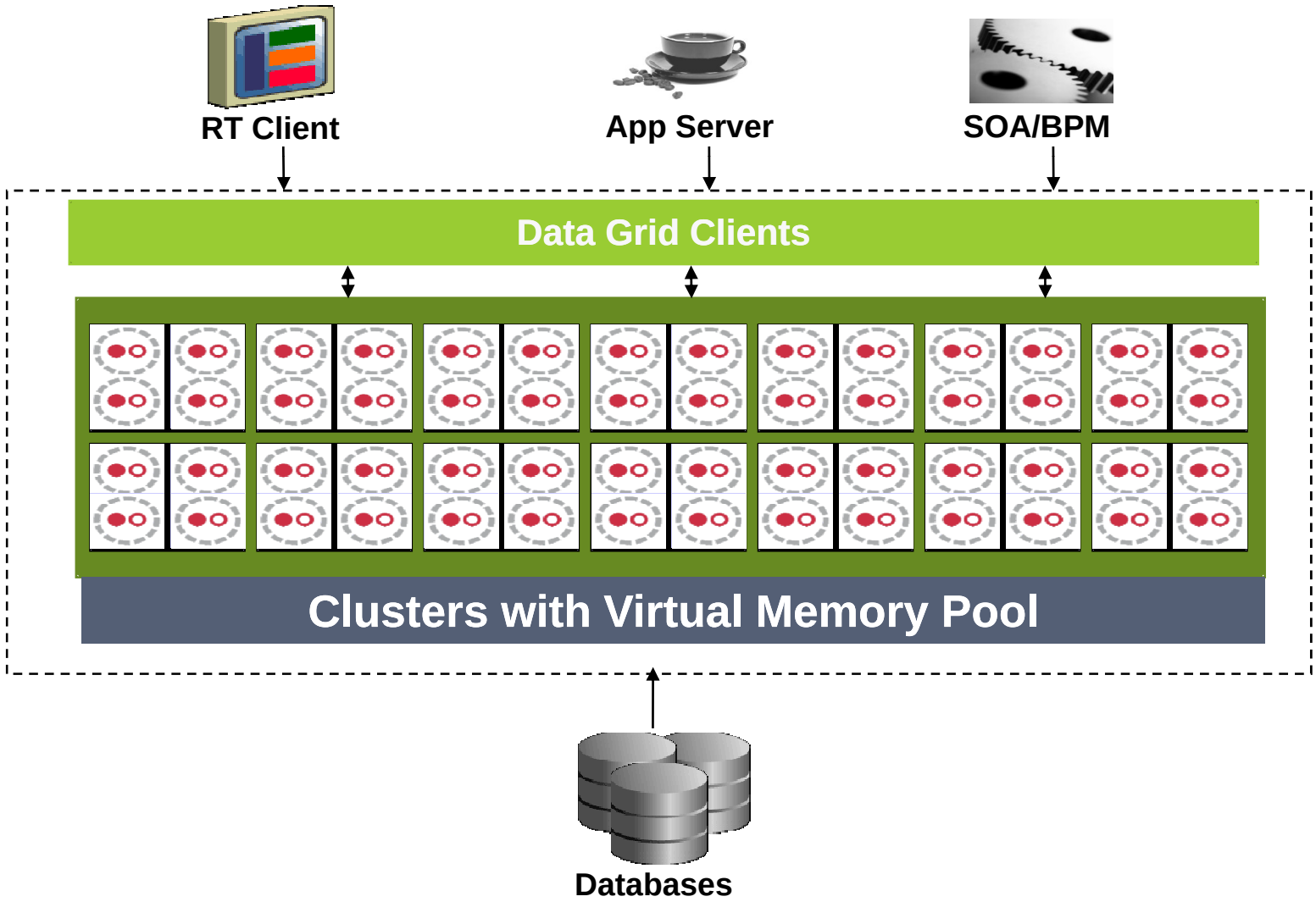
- **Architecture**
  - What applications don't want those QoS?
  - Two servers or two thousand servers
- **Virtualization**
  - Increased demand on Data Sources
  - Application re-provisioning must occur transparently without interruption of data access



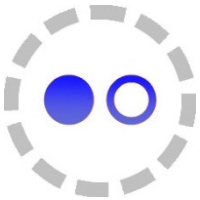
- **SOA**
  - Increasing common access to resources
  - Weakest Link: Continuous availability and absolute reliability
- **XTP**
  - Highest volume, Low Latency, Absolute Transactional Integrity
- **EDA**
  - Event driving transactions causing massive increase in load

Oracle Coherence

Reliable, Coherent, In-Memory Data Grid



Data Grid Uses



Caching

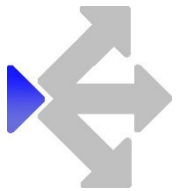
Applications request data from the Data Grid rather than backend data sources

Analytics

Applications ask the Data Grid questions from simple queries to advanced scenario modeling

Transactions

Data Grid acts as a transactional System of Record, hosting data and business logic



Events

Automated processing based on event

Insurance Company

Problem

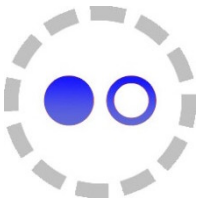
- Managing user-entered policy information on public web site.
- Persisting profiles to database required upwards of one second – multiplied by thousands of concurrent users

Challenge

Needed to offload rapidly expanding middleware processing from core backend database processing

Solution

Caching to manage all data operations in-memory



Benefits

- 90% reduction of database load = increase in capacity
- Application survived an extended database outage with no impact

Financial Institution

Problem

Query-intensive Portfolio Management application required 30+ seconds to generate pages via database queries

Challenge

Portfolio managers require rapid access to accurate information

Solution

Execute all queries against data directly in memory across Data Grid.

Benefits

- No changes to database schema: operational cost savings
- All access to database during off-peak hours: lowered operational impact

Hospitality Chain

Problem

Throughput challenges for rule-based price-optimizing reservation engine due to volume of transactions exceeding database server capacity

Challenge

Enable thousands of customer service representatives to maximize per-stay hotel revenue

Solution:

Use Data Grid for system of record for all transactions

Benefits

- Dramatically increased system scalability
- Increased capacity of existing infrastructure

Gaming Company

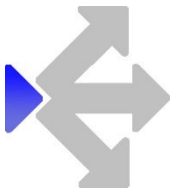
Problem

Matching engine supporting several thousand matches per second, with intense “hot spots” on specific instruments

Challenge

Revenue tied directly to customer activity. Need for high-throughput, low-latency solution for financial transactions

Solution: Use event-driven architecture, treating bids as incoming events, modifying the state of bidding markets, and dispatching matched bids

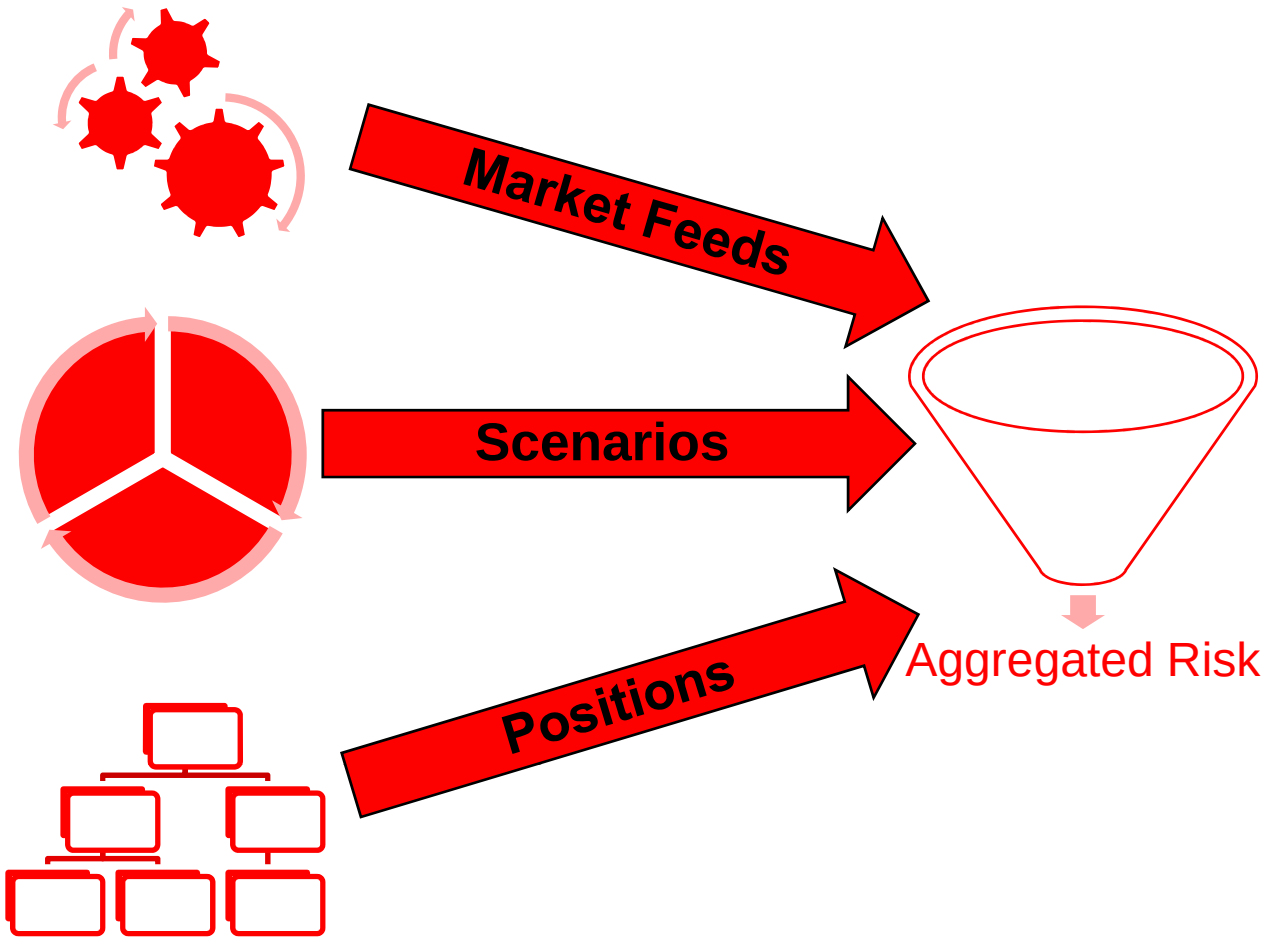


Benefits

- Moving event processing into application tier increased capacity to handle peak loads
- Enabled application developers to modify logic without impacting the database; operational cost savings & increased flexibility

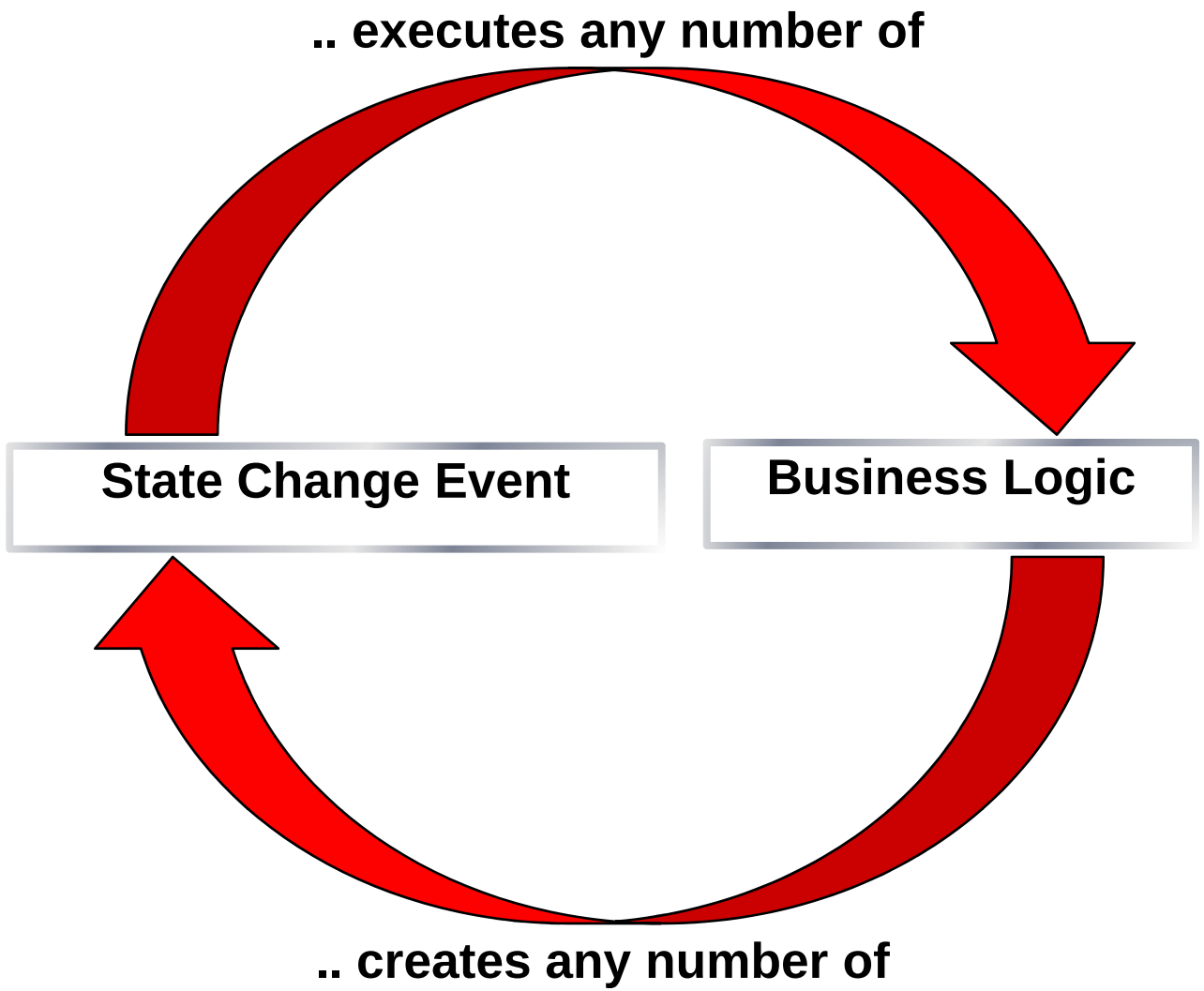
Time-Bound Risk & Real-Time Risk

- Large In-Memory Data Sets
- Parallel Aggregation
- Capable of Absorbing Real Time Feeds
- Recalc on Changes / Calc on Demand
- 100% Data Locality for computations
- Latency inversely proportional to hardware resources
  - Linear Scale
- Real World Results:
  - 50 Days -> 1 Hour



Event Driven Architectures

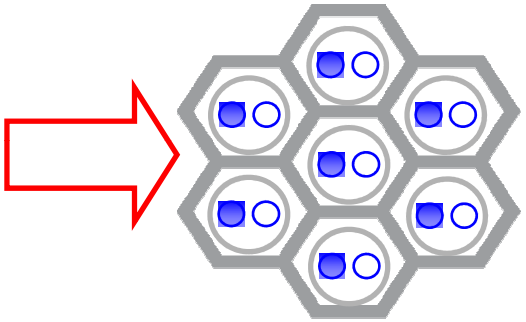
- Business Logic driven by State Changes
- Once-and-Only-Once Guarantees
- Encapsulated Transactions
  - No “Global-TX”
- 100% Data Locality for Business Logic execution
  - Events spread across the Data Grid
  - Parallel Execution
- Real World Results:
  - #1 FX Market

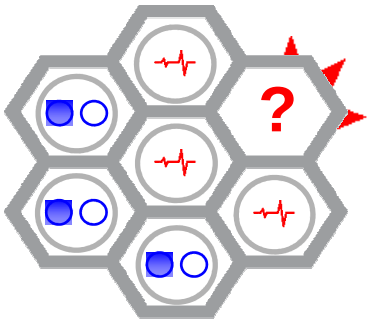


Oracle Coherence: Overview

How Does Oracle Coherence Data Grid Work?

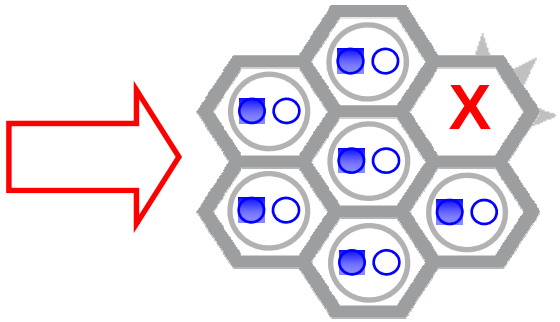
- Data load-balanced in-memory across a cluster of servers
- Data automatically and synchronously replicated to at least one other server for continuous availability
- Single System Image: Logical view of all data on all servers





- Servers monitor the health of each other
- In the event a server fails or is unhealthy, other servers cooperatively diagnose the state

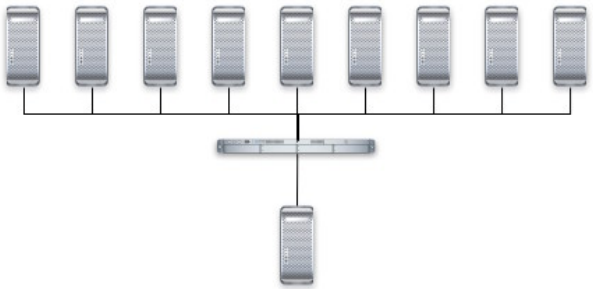
- The healthy servers immediately assume the responsibilities of the failed server
- Continuous Operation: No interruption of service or loss of data due when a server fails



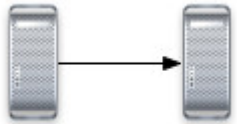
Traditional Scale-Out Approaches…

#1. Avoid the challenge of maintaining consensus

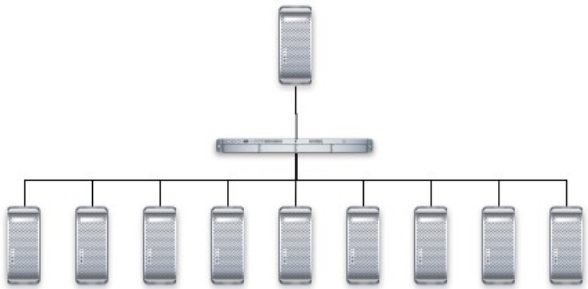
- Opt for the “single point of knowledge”



Client + Server Model
(Hub + Spoke)



Active + Passive
(High Availability)



Master + Worker Model
(Grid Agents)

#2. Have crude consensus mechanisms, that typically fail and result in data integrity issues (including loss)

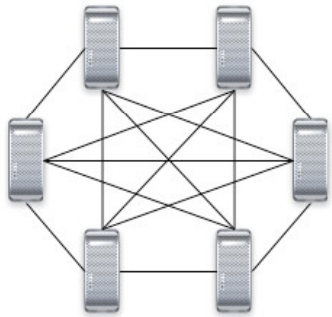
Traditional Scale-Out Approaches…

Real-World Feedback

- Have unbalanced / unfair load and task management
  - Some servers have greater system responsibility than others
- Have Single Points of Bottleneck (SPoB)
- Have Single Points of Failure (SPoF)
  - “Micro outages” are magnified as you scale-out
- Exhibit Strong Coupling to Physical Resources
  - Software completely dependent on individual physical servers
- Require specialized deployment and operation for individual Resources
  - Some servers require “special attention” to operate

Coherence: A Unique Approach

- In Coherence…
  - Members **share** responsibilities (health, services, data…)
  - Completely Peer-to-Peer
  - No Single Points of Bottleneck (SPOBs)
  - No Single Points of Failure (SPOFs)
  - Linearly scalable to thousands of servers by design
- Servers form a full “mesh”
  - No Masters / Slaves etc.
  - Data Grid members work together as a team
  - Communication is almost always point-to-point
    - Designed for commodity switched infrastructures
    - Scalable throughput up to the limit of the backplane



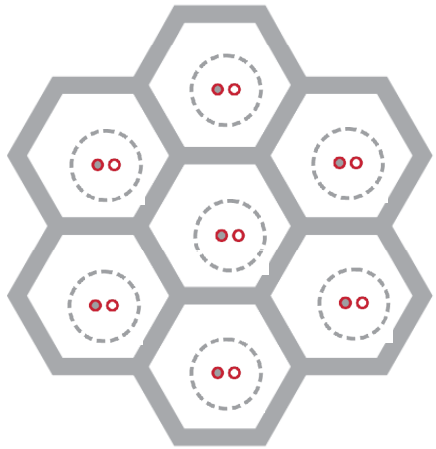
Oracle Coherence Data Grid

- Continuous Availability for application data and processing
- Scales out linearly, whether 2 or 2,000 servers
- Power to perform massive Data Grid based analytics, transaction and event processing
- Provides instant data access while reducing load on back-end data sources

Oracle Coherence simultaneously addresses Availability, Reliability, Scalability and Performance

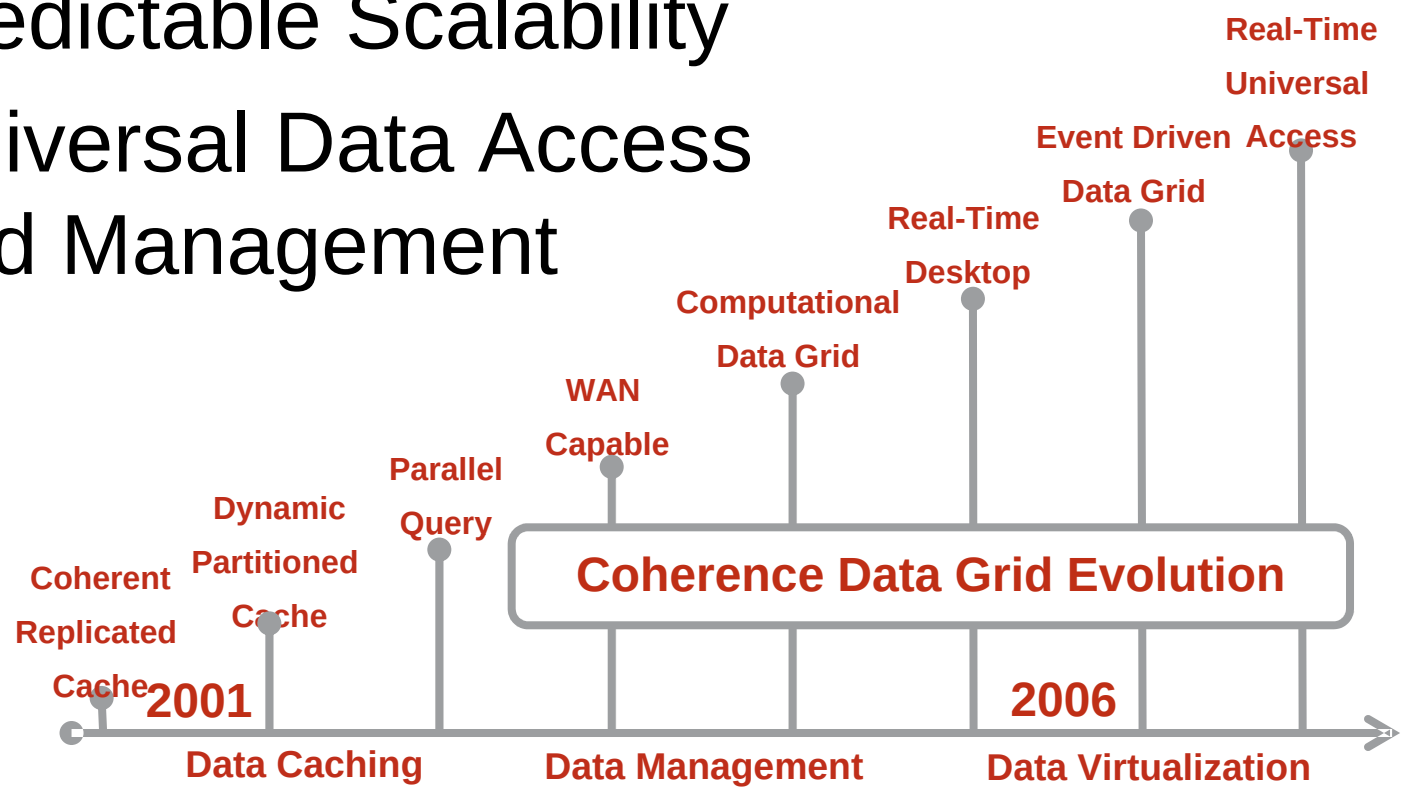
Universal Access & Management

- All data in the Data Grid is accessible from any single node
  - Single System Image = Simple programming paradigm
- Optimizes data locality in Grid based on usage or access
  - Move state or behavior
- Parallelizes data loading, data queries, processing of data managed in grid
- Database integration
  - Blocking write-through (Synchronous)
  - Reliable write-behind (Asynchronous)



Oracle Coherence Data Grid

- Reliable by Design
- Predictable Scalability
- Universal Data Access and Management



For More Information

http://search.oracle.com

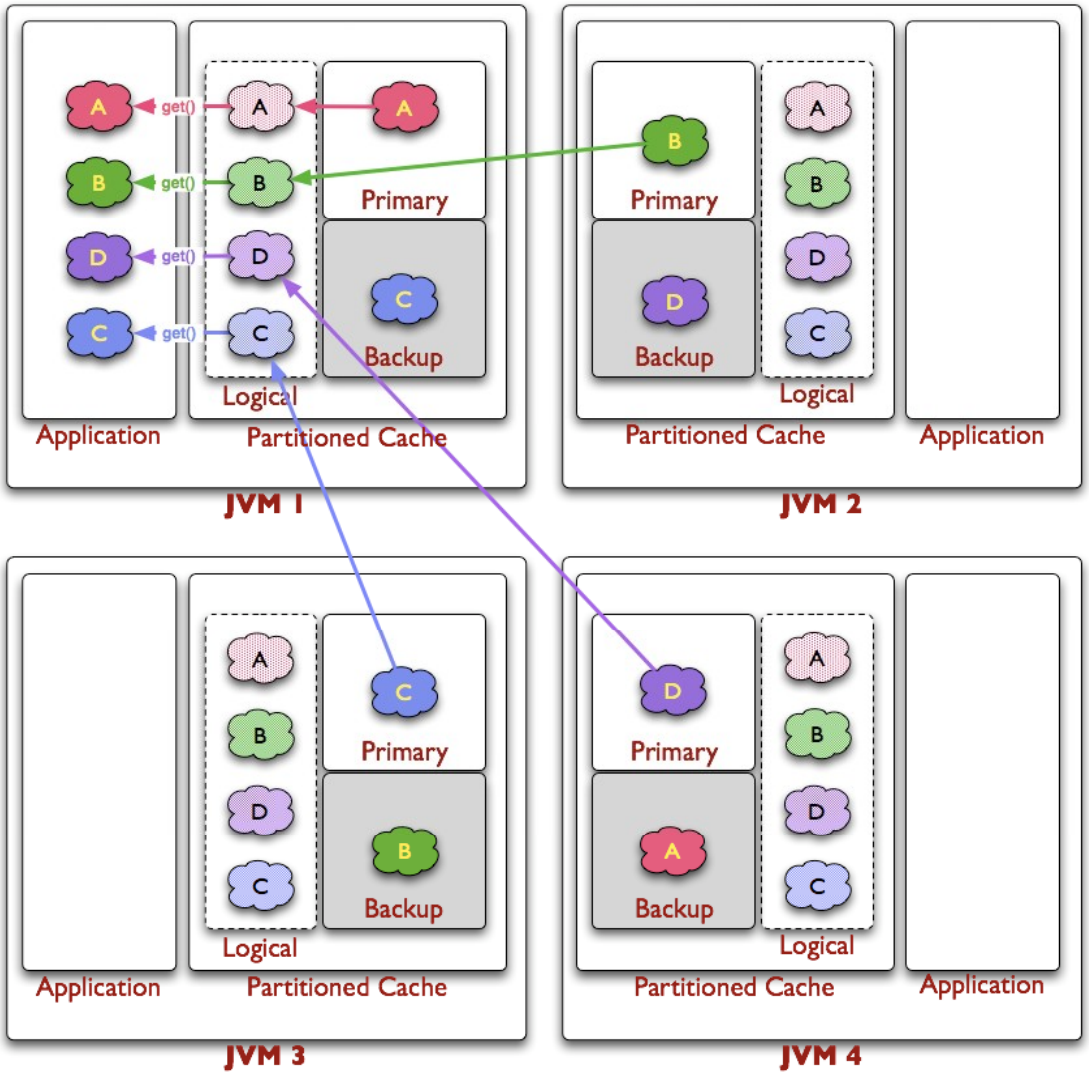
or

http://www.oracle.com/products/middleware/coherence/index.html

Oracle Coherence: Data Grid

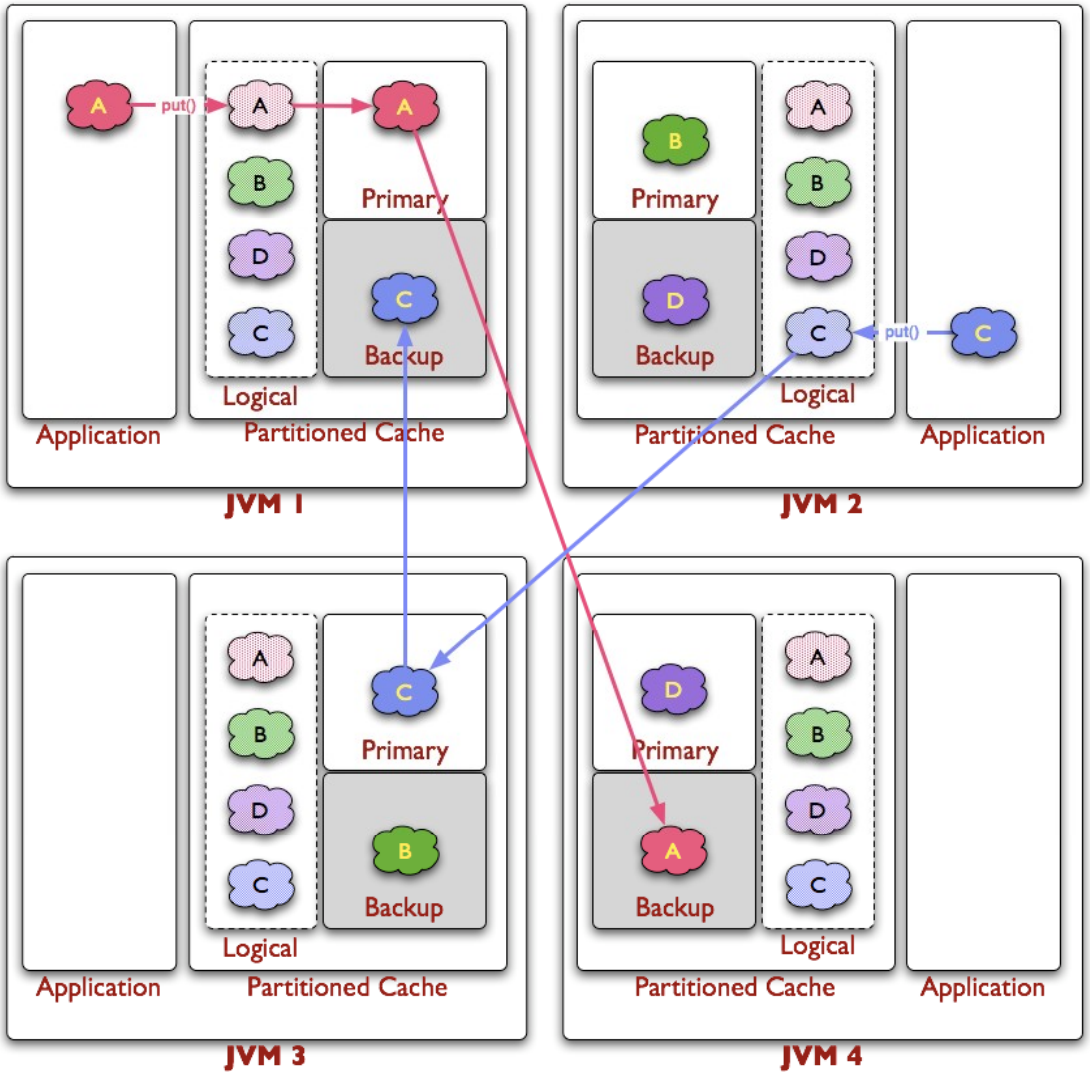
Partitioned Topology : Data Access

- Data spread and backed up across Members
- Transparent to developer
- Members have access to all Data
- All Data locations are known – no lookup & no registry!



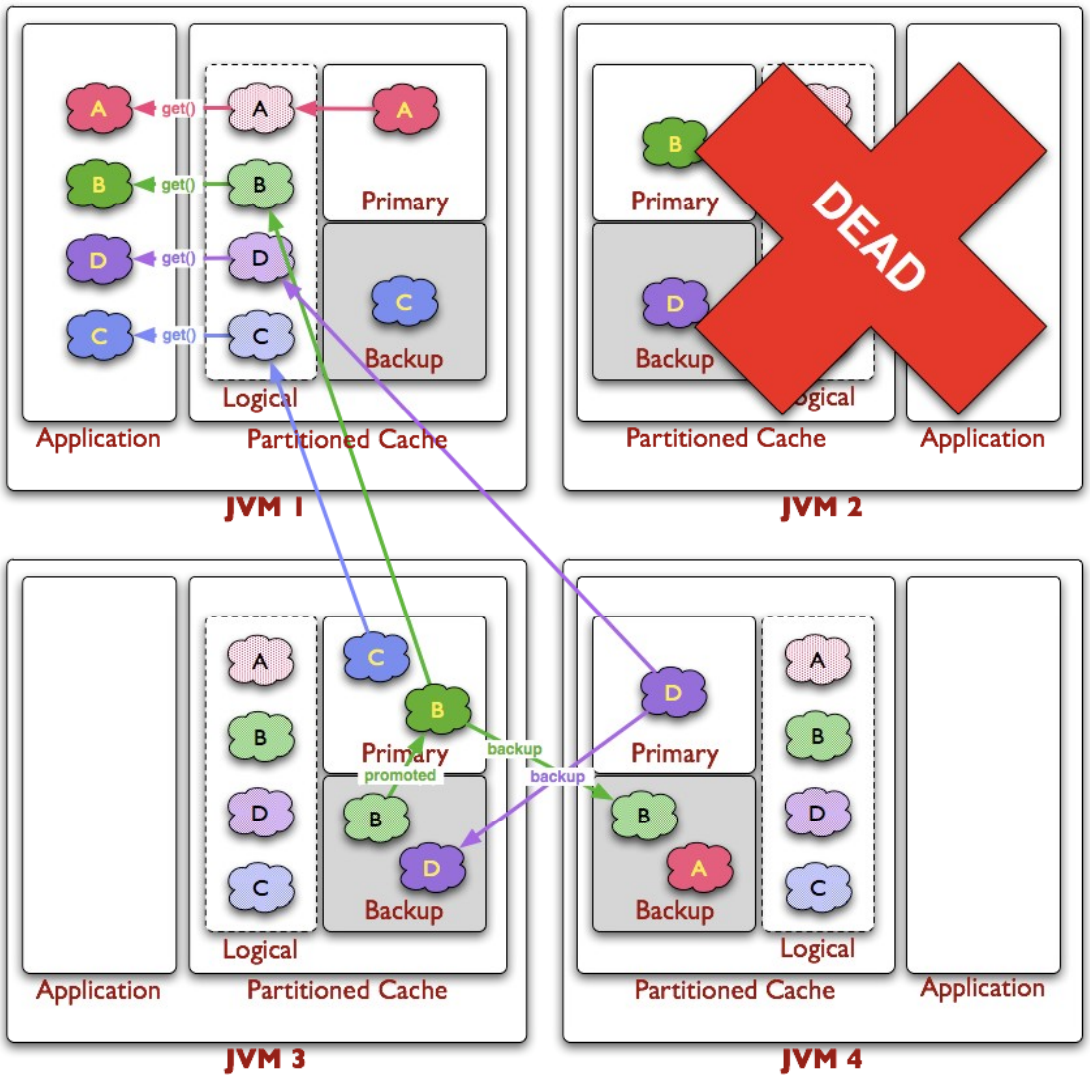
Partitioned Topology : Data Update

- Synchronous Update
- Avoids potential Data Loss & Corruption
- Predictable Performance
- Backup Partitions are partitioned away from Primaries for resilience
- No engineering requirement to setup Primaries or Backups
- Automatically and Dynamically Managed



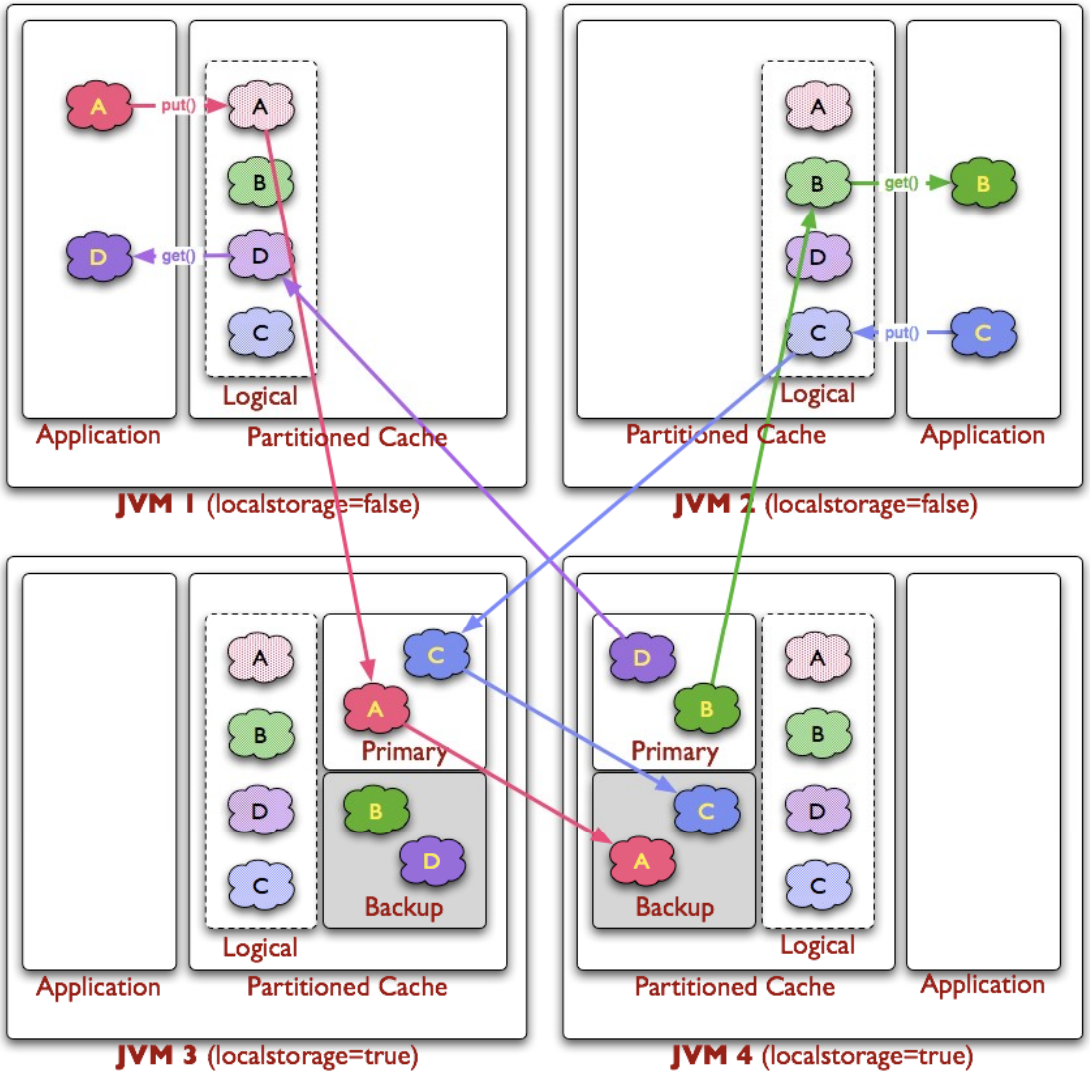
Partitioned Topology : Recovery

- Membership changes (new members added or members leaving)
- Other members, using consensus, recover and repartition automatically
- No in-flight operations lost, no availability gap!
- Some latencies (due to higher priority of asynchronous recovery)
- Information Reliability & Continuous Availability are the priorities



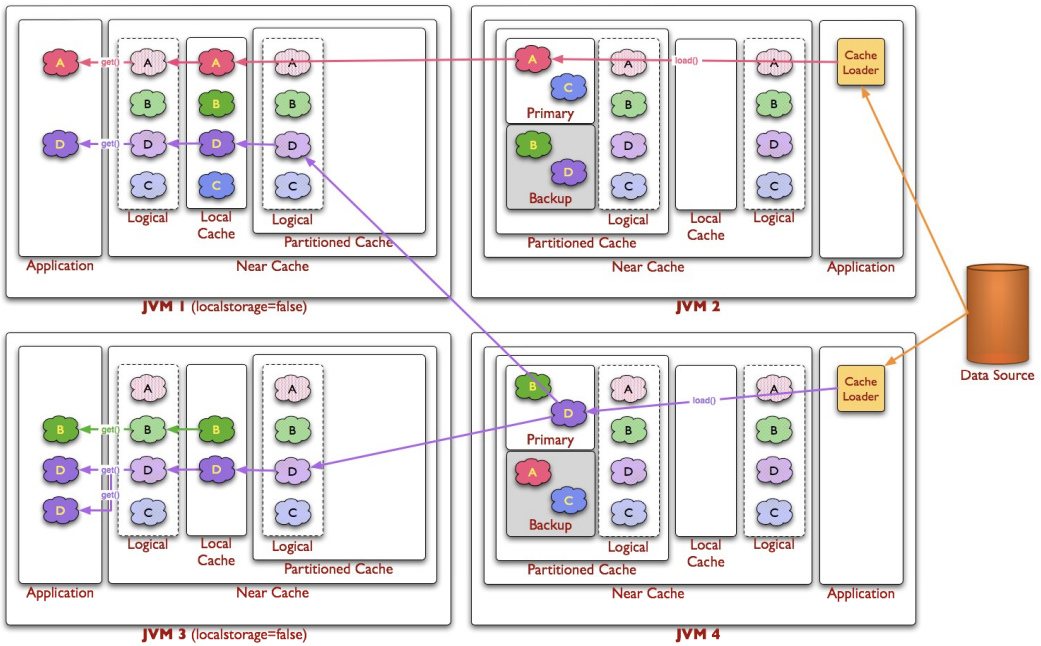
Partitioned Topology : Local Storage

- Some members are used to manage data
- Other members are temporary in a cluster, or do not have memory to spare for managing data
  - They should not cause repartitioning
- Specialization of roles within a Data Grid: Clients and Servers



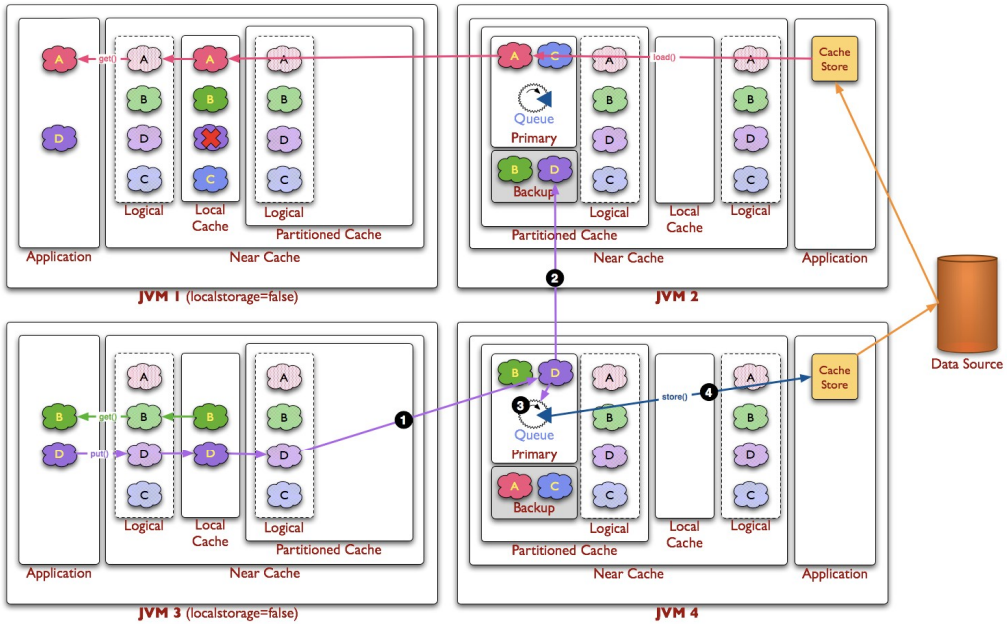
Read-Through & Write-Through

- Access to the data sources go through the Data Grid.
- Read and write operations are always managed by the node that owns the data within the Data Grid.
- Concurrent accesses are combined, greatly reducing database load.
- Write-Through keeps the in-memory data and the database in sync.

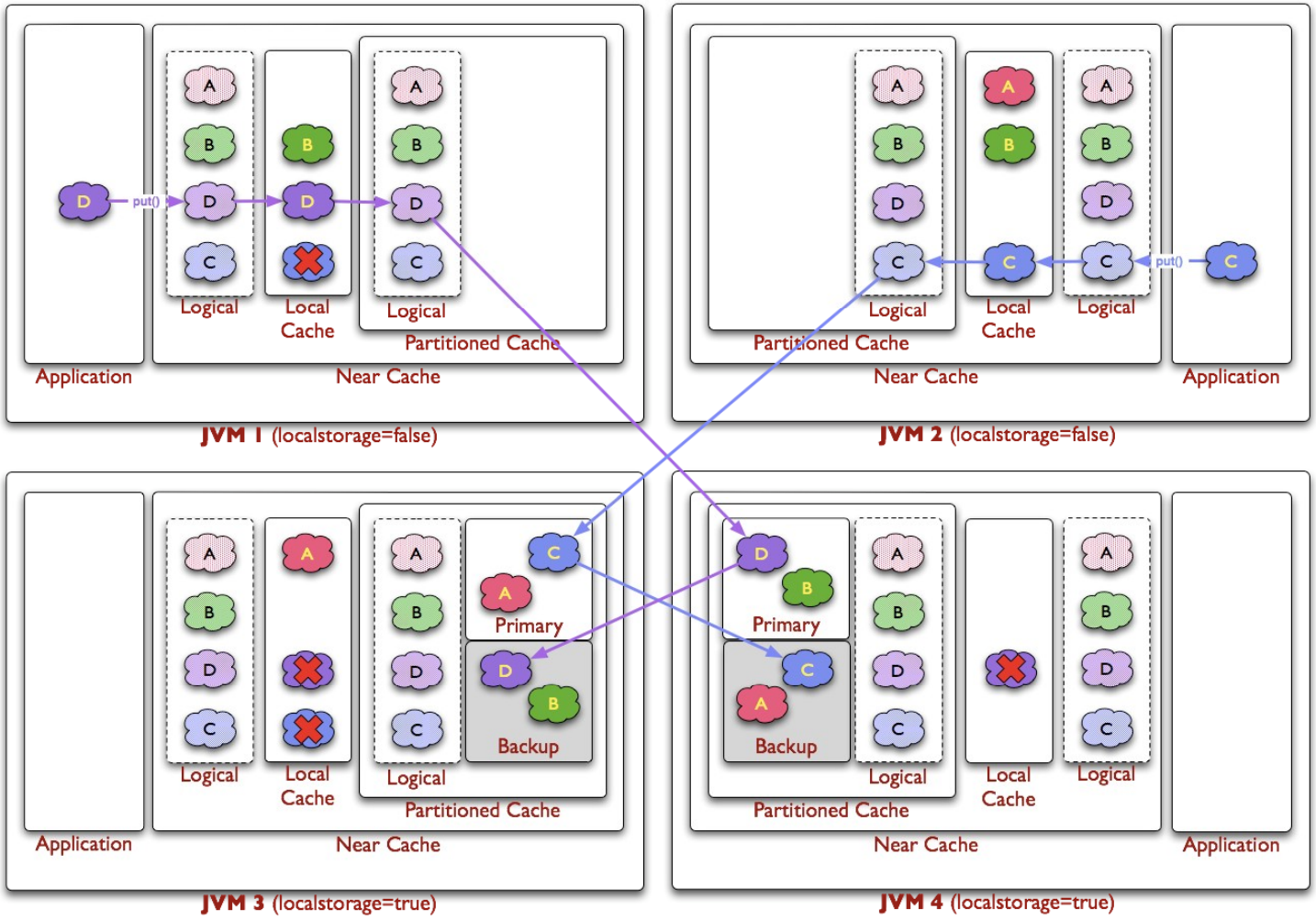


Write-Behind

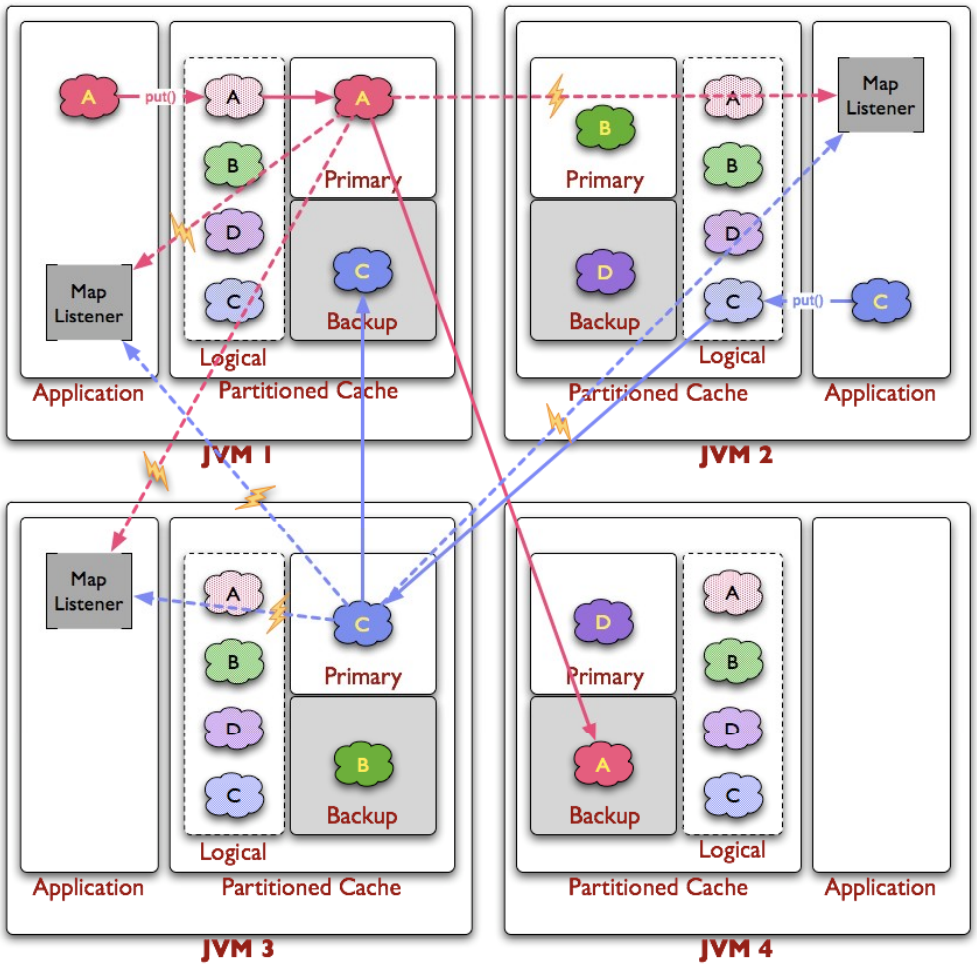
- Write-Behind accepts data modifications directly into the Data Grid
- The modifications are then asynchronously written back to the data source, optionally after a specified delay
- All write-behind data is synchronously and redundantly managed, making it resilient to server failure



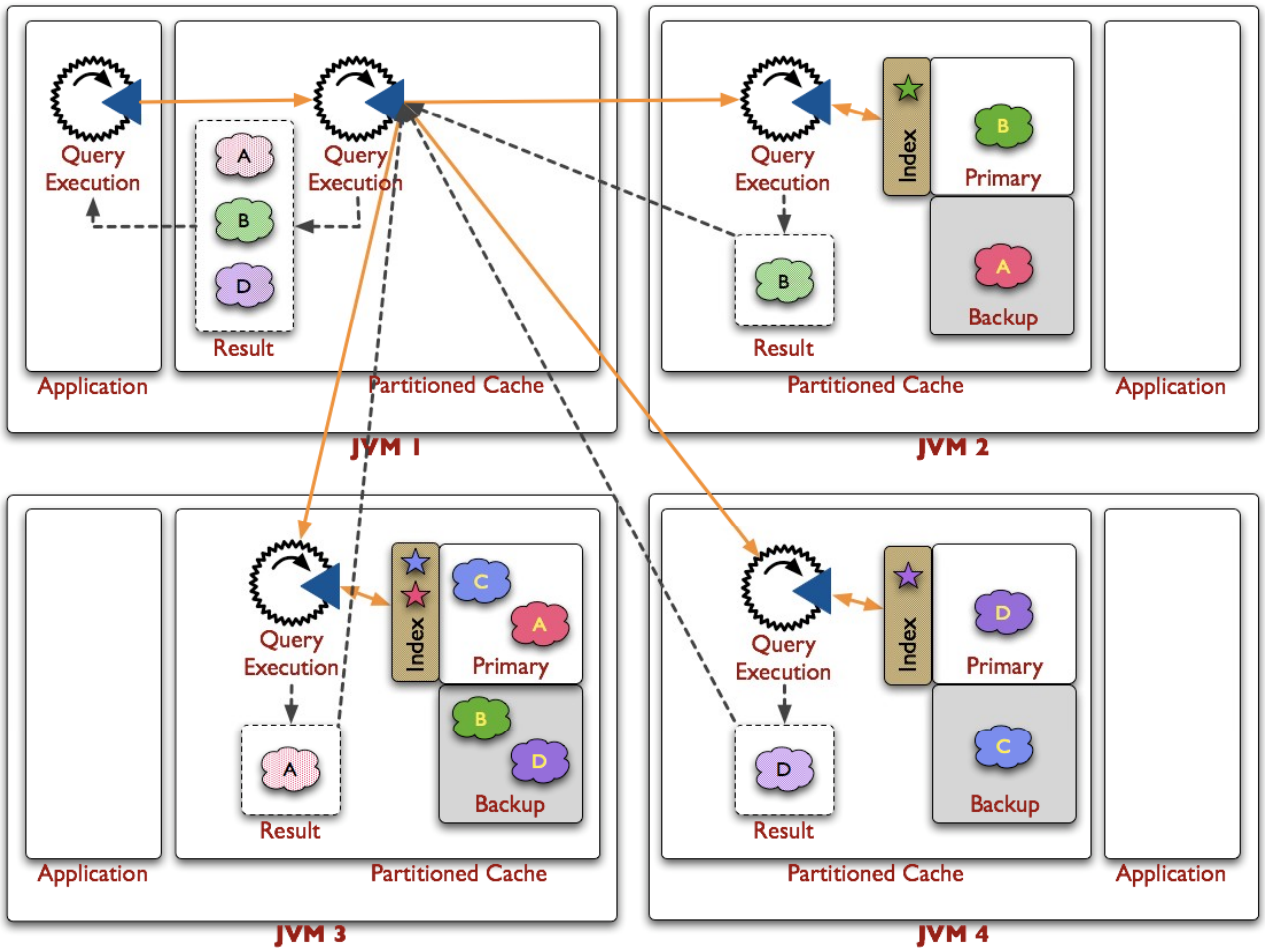
Topology Composition : Near Topology



Features : Observable Interface

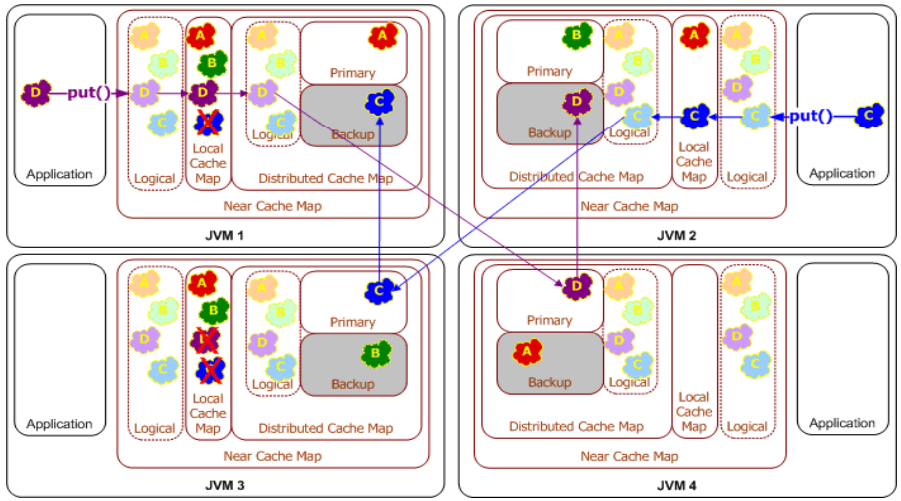


Features : QueryMap Interface

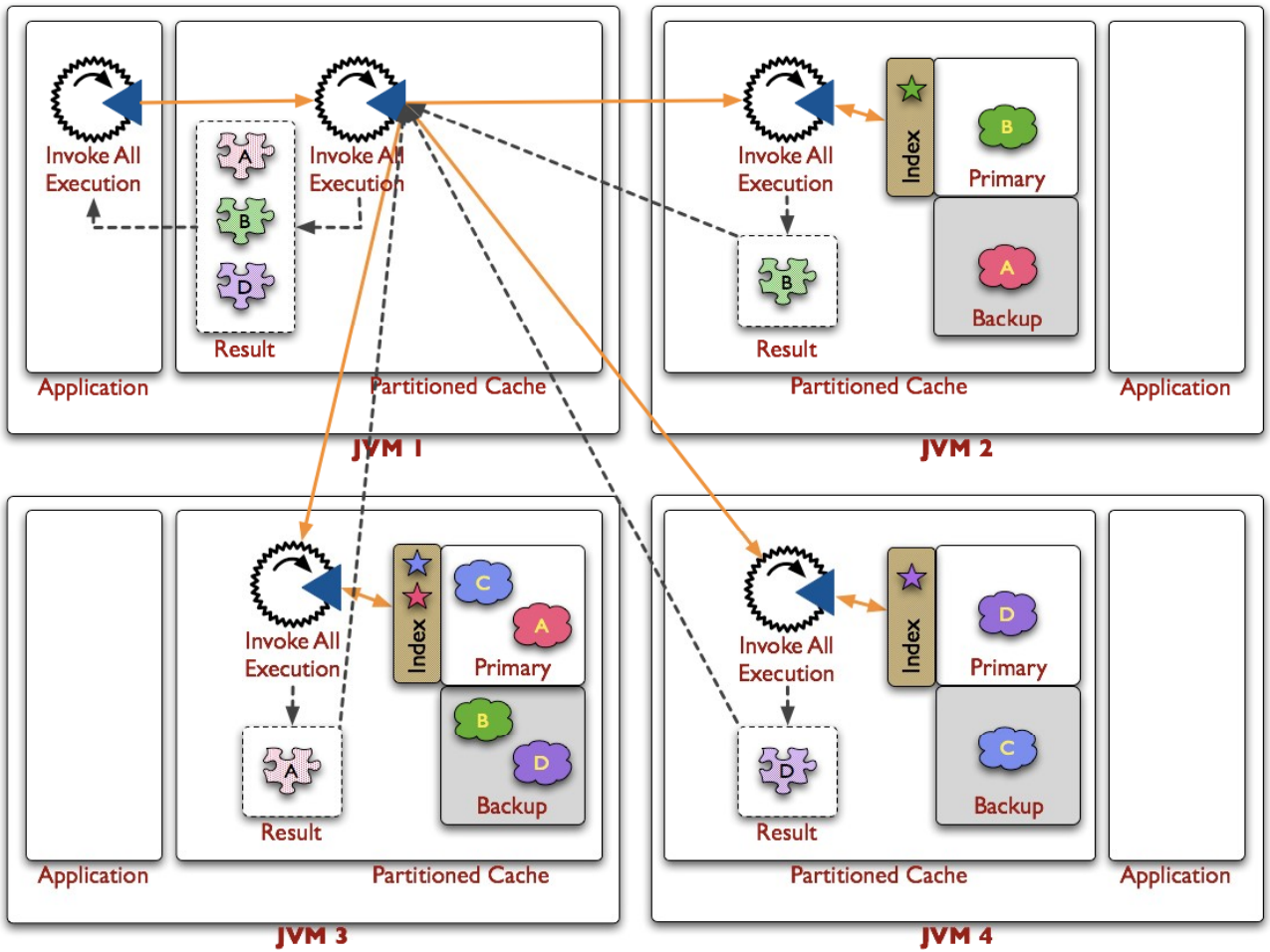


Concurrency

- Implicit: Queueing of operations
  - *Virtual* queue & thread per entry
- Explicit: Pessimistic locking
  - Grid-Wide Mutex
- Transactions: Unit of work management
  - Both optimistic and pessimistic transactions
  - Isolation levels from read-committed through serializable
  - Integrated with JTA



Features : InvocableMap Interface



For More Information

http://search.oracle.com

or

http://www.oracle.com/products/middleware/coherence/index.html