Database or Datagrid?

Mircea Markus
Redhat

JBoss Community

Who’s this guy?

- Sr. SE at JBoss by Redhat
- Projects: Infinispan, JBossCache, PojoCache, jGroups, JBossAS..
- Founder http://radargun.sourceforge.net
- Twitter: @mirceamarkus

Agenda

- Where do data grids fit?
- In-memory data grid - state of play
- Data grids + RDBMS
- Data grids without a RDBMS
- My take

RDBMS

- “Traditional” way of storing data
- Proven, well known
- Limitations
  - built for vertical scale
  - not cloud friendly - hard to partition data
  - rigid
    - what if you don’t need durability?
- One size does not fit all

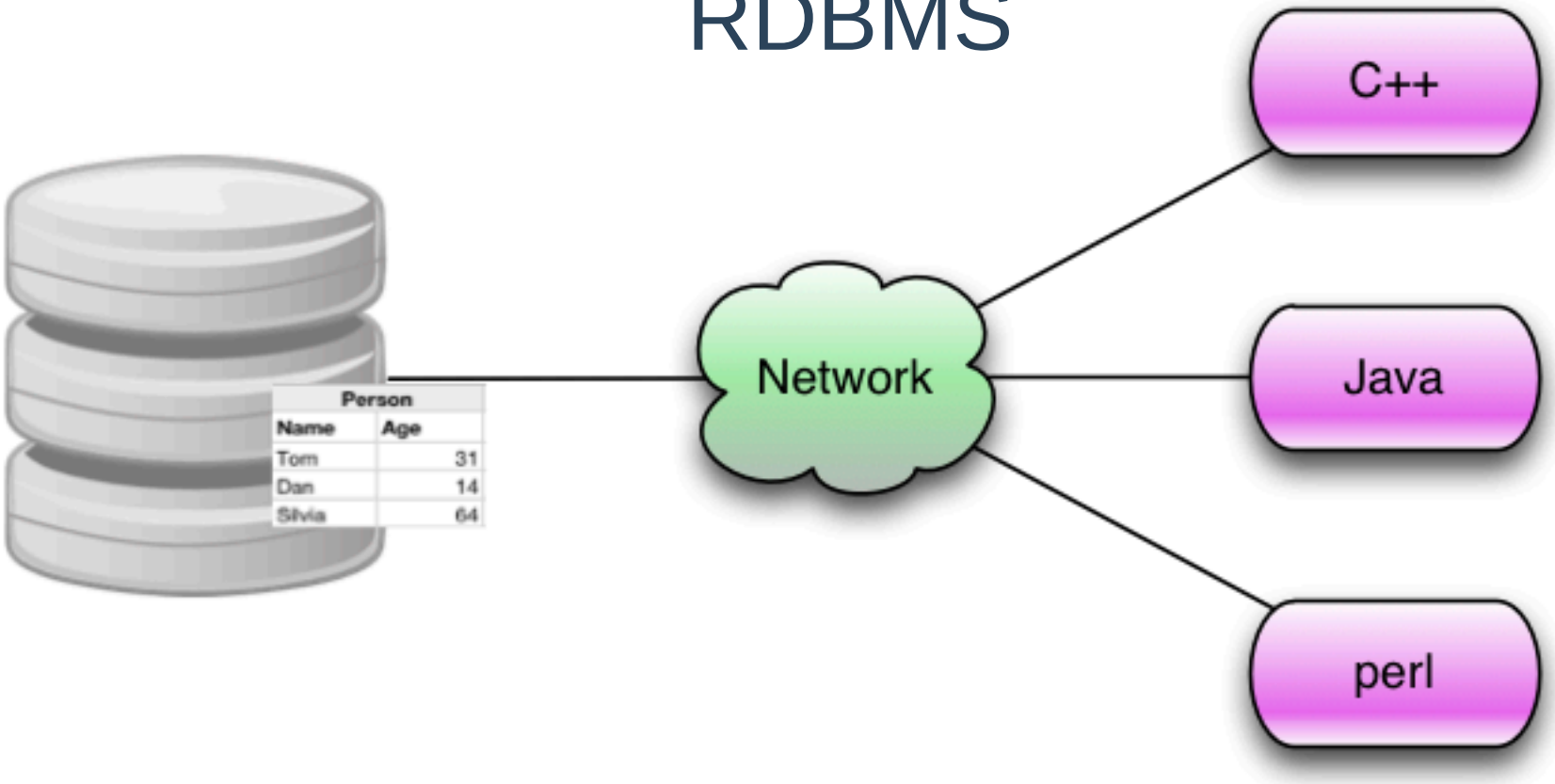
Data grid evolution

- Have been around for a while
- Moving into mainstream
  - vertical scaling is harder
  - memory is cheap
  - network is powerful
  - cloud is here
    - elasticity is important
- More and more interest

Available on the market

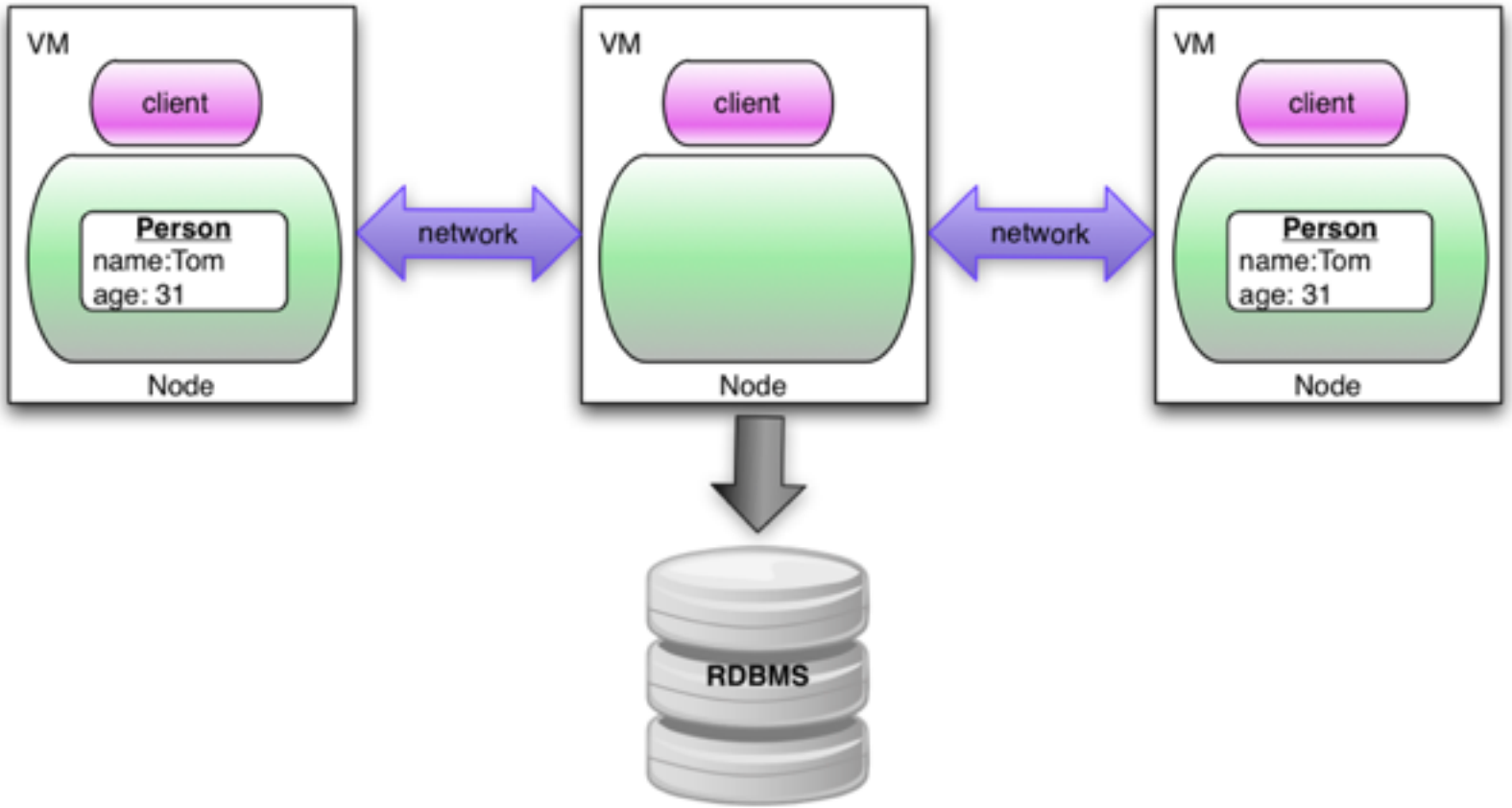
- Open source
  - Infinispan
  - Ehcache
  - Hazelcast
- Commercial
  - Oracle Coherence
  - Gigaspaces XAP
  - Gemfire

RDBMS



JBoss Community

And a data grid...



JBoss Community

Agenda

- Where do data grids fit?
- In-memory data grid - state of play
- Data grids + RDBMS
- Data grids without a RDBMS
- My take

In-memory data grid characteristics

- No standard (yet)
  - but products share similar characteristics
- API
  - Map based
  - no/less strict schema
- Build with horizontal scaling in mind
  - consistent hashing commonly used
  - suitable for commodity, heterogeneous

Storage

- In memory
  - fast access
- OO model
  - (potentially) language independent
  - object shared across multiple servers
- Durability
  - redundancy
  - flushing state to disk

Access pattern

- Embedded
  - client and node in same VM
  - fast communication
    - less object serialisation
  - supports transactions
- Remote
  - similar to DB
    - client/server

Transactions

- Important aspect
  - data store!
- XA or not
  - fancy distributed transactions?
  - tendency to support XA
- Transactional access
  - embedded
    - most products

Agenda

- Where do data grids fit?
- In-memory data grid - state of play
- Data grids + RDBMS
- Data grids without a RDBMS
- My take

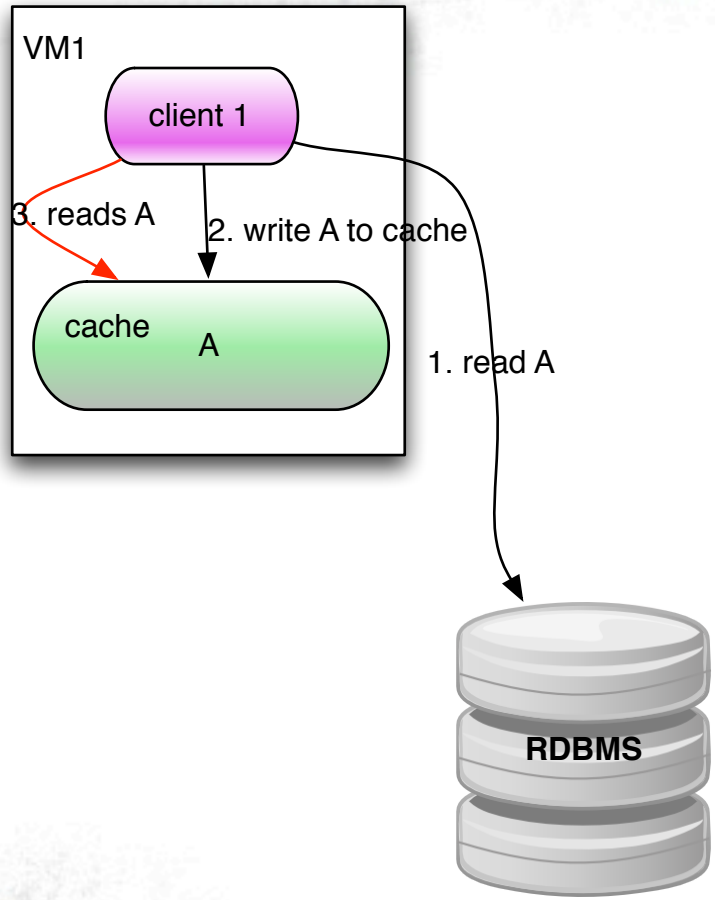
RDBMS + data grid use cases

- Cache on top of a database
- Different levels of caching
  - Clustered cache
    - more caching capacity
  - Caching server
    - dedicated cache cluster
    - remote access

At the beginning...

- Local cache
  - java.util.Map
- Challenges
  - eviction
  - expiry
  - write through, write behind
  - preloading
  - notifications

Local caching

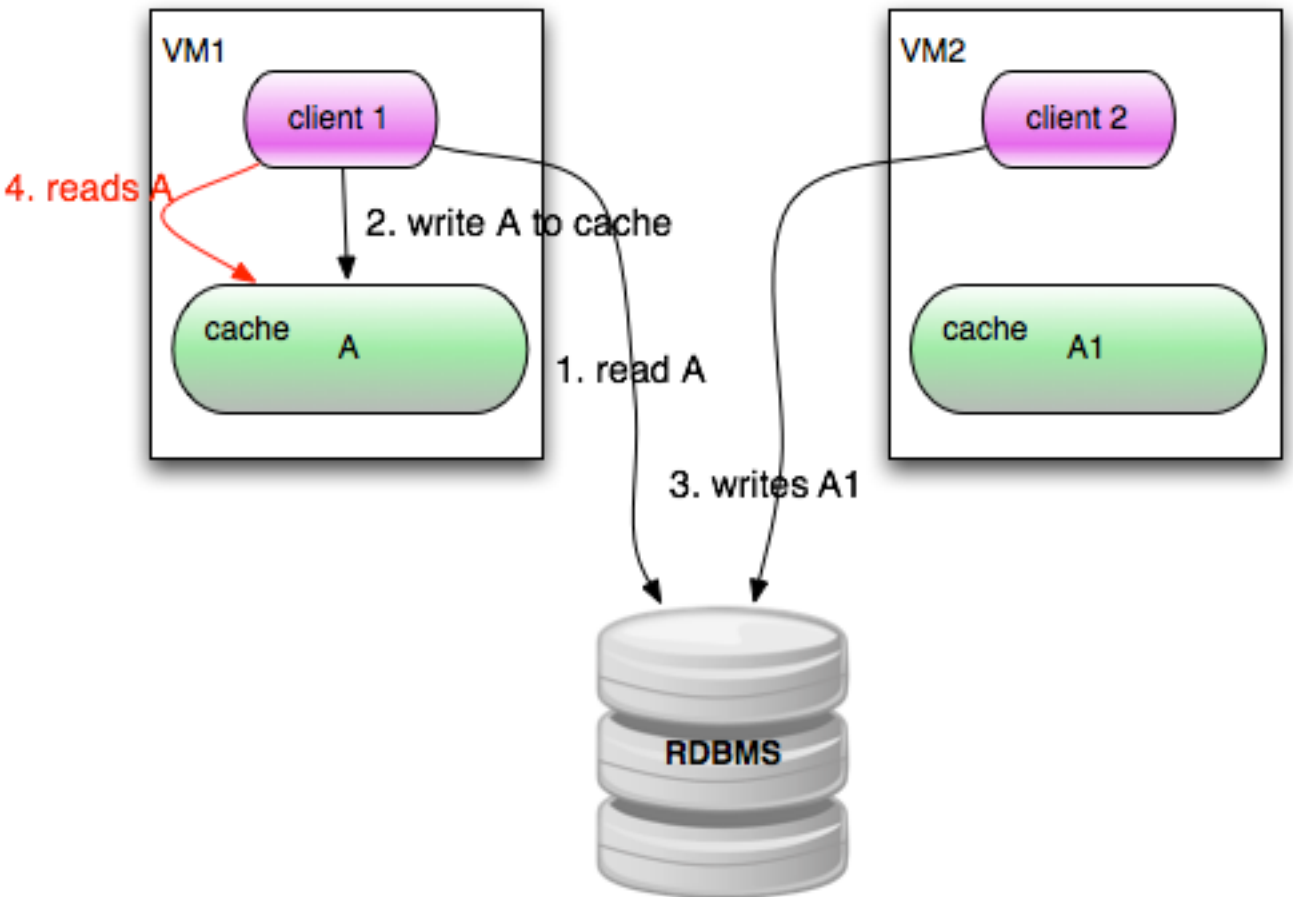


JBoss Community

But local caching has limitations..

- Doesn’t scale
  - not enough memory
  - not HA
    - cannot write-behind
- Dirty reads
  - multiple nodes write
  - difficult to invalidate the local cache

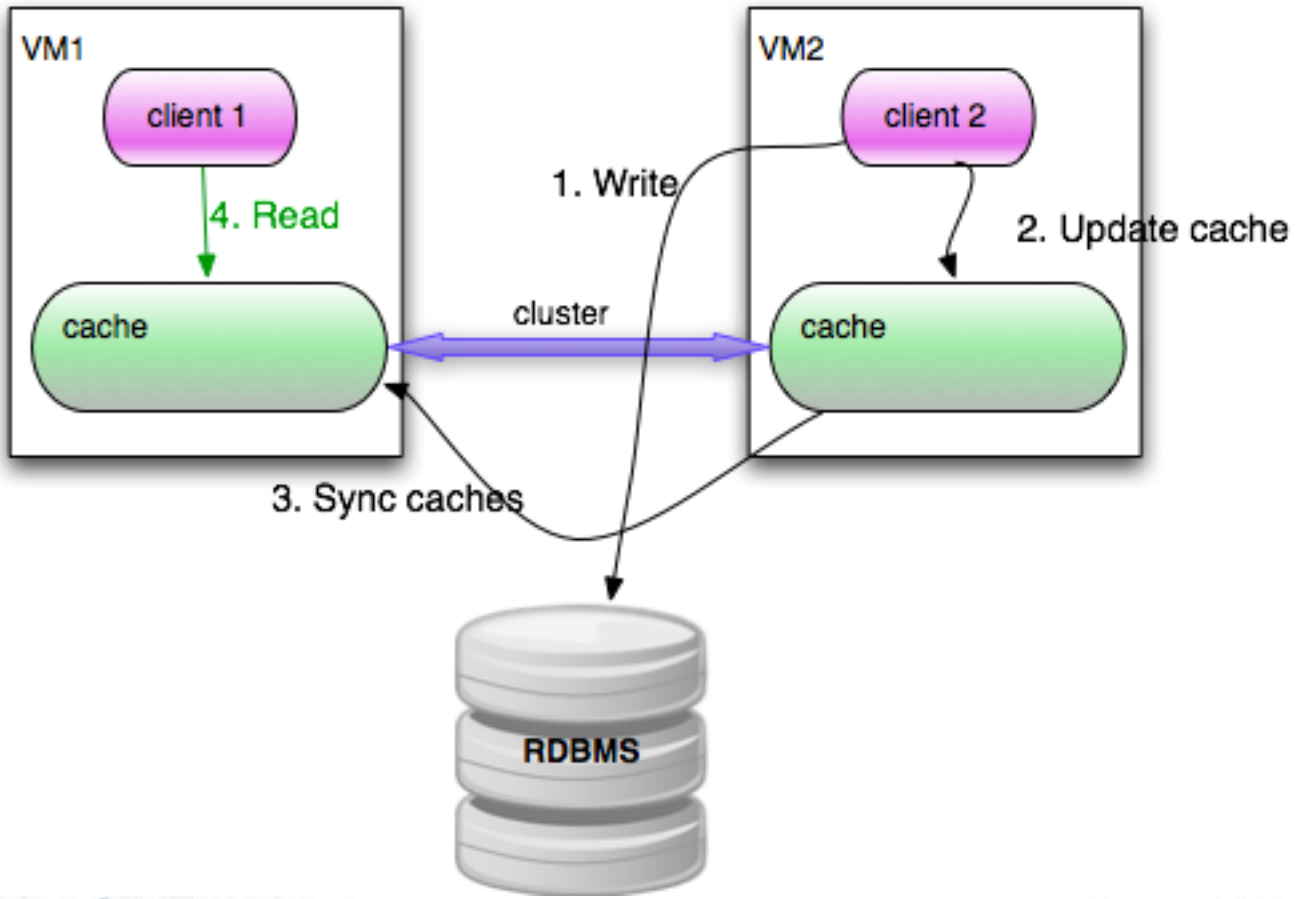
Dirty reads with local cache



Why clustered in memory caches?

- Solve the problems of local caches
- Performance booster
- Similar to local cache
  - but more shared memory to use
- Cluster aware
  - no risk of dirty reads
    - replication
    - invalidation

Cluster of caches

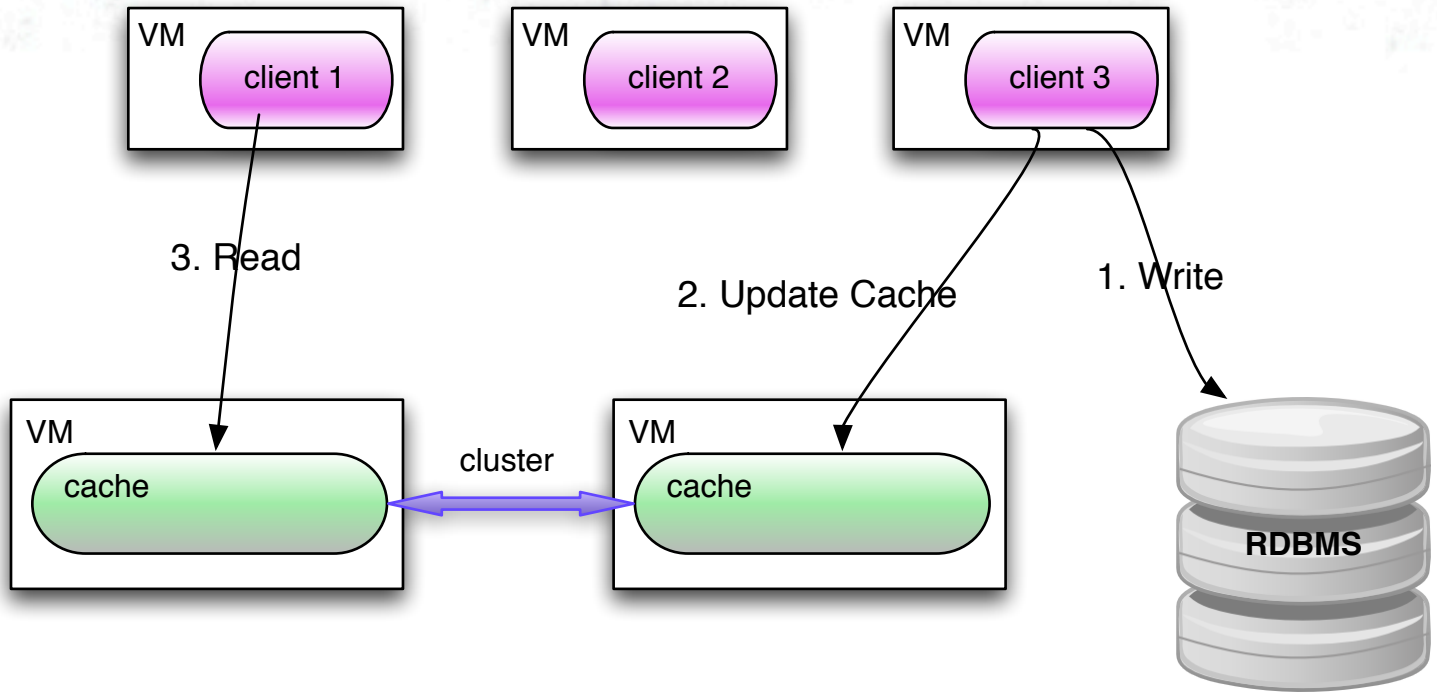


JBoss Community

But that brings other concerns..

- Client is affected by cache topology changes
- Increased client startup time
  - start the cache
  - transfer state
- Tier management
  - incompatible JVM tuning
  - security
  - garbage collection

Cache servers



Client/Server - vendor support

- Most vendors
- Proprietary protocols
  - Coherence - *Extend
- Open protocols
  - Memcached
  - REST
  - Hotrod
    - smart routing

Open Client/Server protocols

	Protocol	Client Libraries	Clustered?	Smart Routin	Load Balancing/ Failover
REST	Text	N/A	Yes	No	Any HTTP load balancer
Memcached	Text	Plenty	Yes	No	Only with predefined server list
Hot Rod	Binary	Java, Python	Yes	Yes	Dynamic

Agenda

- Where do data grids fit?
- In-memory data grid - state of play
- Data grids + RDBMS
- Data grids without a RDBMS
- My take

Data grid use cases

- Why not RDBMS?
- Challenges
- Types of applications

Mainly to overcome DB

- Speed
  - disk access is slow
  - keep data closer to the computation unit
    - memory!
- Scalability
  - DB is hard and expensive to scale
  - consistent-hash for scaling
    - virtually infinite horizontal scalability

Data grid challenges

- Different/new access pattern
  - key based/Map API
  - hierarchical
  - JPA API
    - smooth transition
    - Hibernate OGM on top on Infinispan
- Different skill set
  - OO programmer vs SQL

E.g. the DB would do this..

```
-- insert some data
insert into PERSON(pname, age) values('Tom', 31);
insert into PERSON(pname, age) values('Dan', 14);
insert into PERSON(pname, age) values('Silvia', 64);


-- how many persons can have a beer?
select count(*) from PERSON where ager >= 18;
```

Same thing in Infinispan..

```
Cache<Integer, Person> c = getCache();

c.put(1, new Person("Tom", (short) 31));
c.put(2, new Person("Dan", (short) 14));
c.put(3, new Person("Silvia", (short) 64));
```

```
//now the query...
MapReduceTask task = new MapReduceTask(c);
Map<String, Integer> count = task.mappedWith(new PersonAgeMapper()).
      reducedWith(new PersonAgeReducer()).execute();
System.out.println("How many can have a beer? " + count.get("count"));
```

JBoss Community

Based on Infinispan's Map/Reduce

```
public static class PersonAgeMapper implements Mapper... {
   @Override
   public void map(Integer key, Person value, Collector... c) {
      if (value.isOver(18)) {
         c.emit("count", 1);
      }
   }
}

private static class PersonAgeReducer implements Reducer... {
   @Override
   public Integer reduce(String rK, Iterator<Integer> i) {
      int sum = 0;
      while (i.hasNext()) sum += i.next();
      return sum;
   }
}
```

Frequent use cases

- Performance! Performance! Performance!
- Analytics
  - financial/trading applications
- Many transactional
  - XTP
- Event driven architecture
  - CEP
- Clustering toolkit

Agenda

- Where do data grids fit?
- In-memory data grid - state of play
- Data grids + RDBMS
- Data grids without a RDBMS
- My take

Is data grid a RDBMS replacement?

- No!
- DB
  - proven
  - mature
  - known and understood
  - many deployments
- DB is here to stay
  - but not a universal solution for storing data

But definitely to be considered

- DB + data grid
  - take some of DB responsibility
  - caching
  - gain performance
- Fully-fledged datastore
  - durability
    - backup (async) to a database
  - transactional

Still a way to go!

- Data grid not new
  - only started to take off in the recent years
- More
  - products
  - community interest
    - use cases
- Standardisation effort
  - JSR-107 - caching API

- Mircea Markus
- Redhat mmarkus@redhat.com
- Twitter: @mirceamarkus