

# JUDCon

JBoss Users & Developers Conference

## 2012:India



# **Infinispan in 50 minutes**

**Galder Zamarréño**  
Senior Software Engineer  
Red Hat, Inc

25th January 2012, Bangalore

# Galder Zamarréño

- R&D Engineer, Red Hat Inc.
- Infinispan developer
- 5+ years exp. with distributed data systems
- Twitter: **@galderz**
- Blog: [zamarreno.com](http://zamarreno.com)



# Agenda

- What is Infinispan?
  - Principal use cases and key features
- Extras: Querying, Hibernate OGM...

# Introducing

# Infinispan



# What is Infinispan?

An in-memory, highly available, elastic, and open source (LGPL) data grid platform

# Infinispan can be used as...



# Local in-memory cache

**Boost performance** caching data which is hard to calculate or expensive to retrieve



# ConcurrentHashMap ?

**Highly concurrent thanks to MVCC, has built-in eviction, pluggable persistence, JTA support...etc**

# Expiration

Associate a lifespan or maximum idle time to the data in the cache

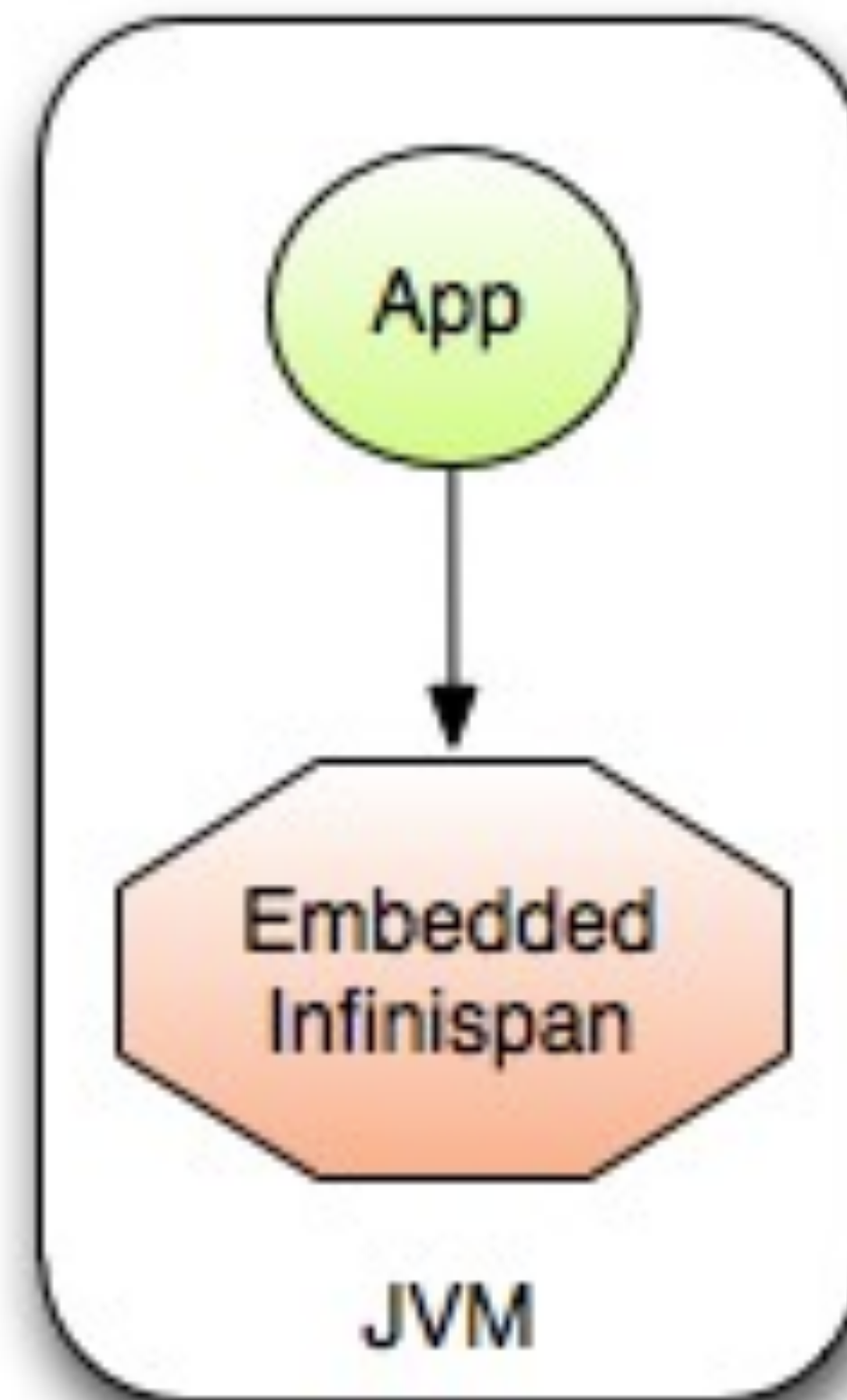


# Eviction

Selective local cache removal based on an algorithm (LRU, **LIRS...**) when cache size exceeded

# Embedded access

- Client application and Infinispan live in same JVM
- **Zero Latency Access!**





# Infinispan meets CDI

```
...
import javax.ejb.Stateless;
import javax.inject.Inject;
import org.infinispan.Cache;

@Stateless
class FooEJB {
    @Inject @MyCacheQualifier
    Cache<String, String> myCache;
}
```

```
...
import javax.enterprise.inject.Produces;
import org.infinispan.cdi.ConfigureCache;
import org.infinispan.configuration.cache.Configuration;
import org.infinispan.configuration.cache.ConfigurationBuilder;

class Config {
    @ConfigureCache("my-cache-name")
    @MyCacheQualifier
    @Produces
    Configuration myCacheConfiguration() {
        return new ConfigurationBuilder()
            .eviction()
            .strategy(FIFO)
            .maxEntries(2048)
            .build();
    }
}
```



# ... and JSR-107

```
...
import javax.cache.interceptor.CacheResult;
import javax.cache.interceptor.CachePut;
import javax.cache.interceptor.CacheValue;
import javax.cache.interceptor.CacheRemoveEntry;
import javax.cache.interceptor.CacheRemoveAll;

class MyDAO {
    @CacheResult(cacheName = "user-cache")
    User getUser(long id) {...};

    @CachePut(cacheName = "user-cache")
    void storeUser(long id, @CacheValue User user) {...};

    @CacheRemoveEntry(cacheName = "user-cache")
    void removeUser(long id) {...};

    @CacheRemoveAll(cacheName = "user-cache")
    void removeAllUser() {...};
}
```



# Event Listeners

**Code hooks associated with  
cache or cache manager  
operations or lifecycle  
events**

# Transactions

Cache operations can  
participate in on-going  
transactions



# Optimistic

Assumes **low lock contention** and so acquires locks on transaction prepare

# Pessimistic

Assumes high lock contention and so acquires locks on each cache write



# XA or Synchronization

If Infinispan used as **data store**, use **XA**

If Infinispan used as **cache**, use **Synchronization**

# Plugs into frameworks

**Hibernate/JPA,  
Torquebox, Grails,  
Spring...etc**



**A local cache might  
not be enough...**

# Clustered caches

**Scale-up your application and  
add failover capability**



# Clustering toolkit

**JBoss Application Server,  
Mobicents, Lucene directory,  
Modeshape...etc**

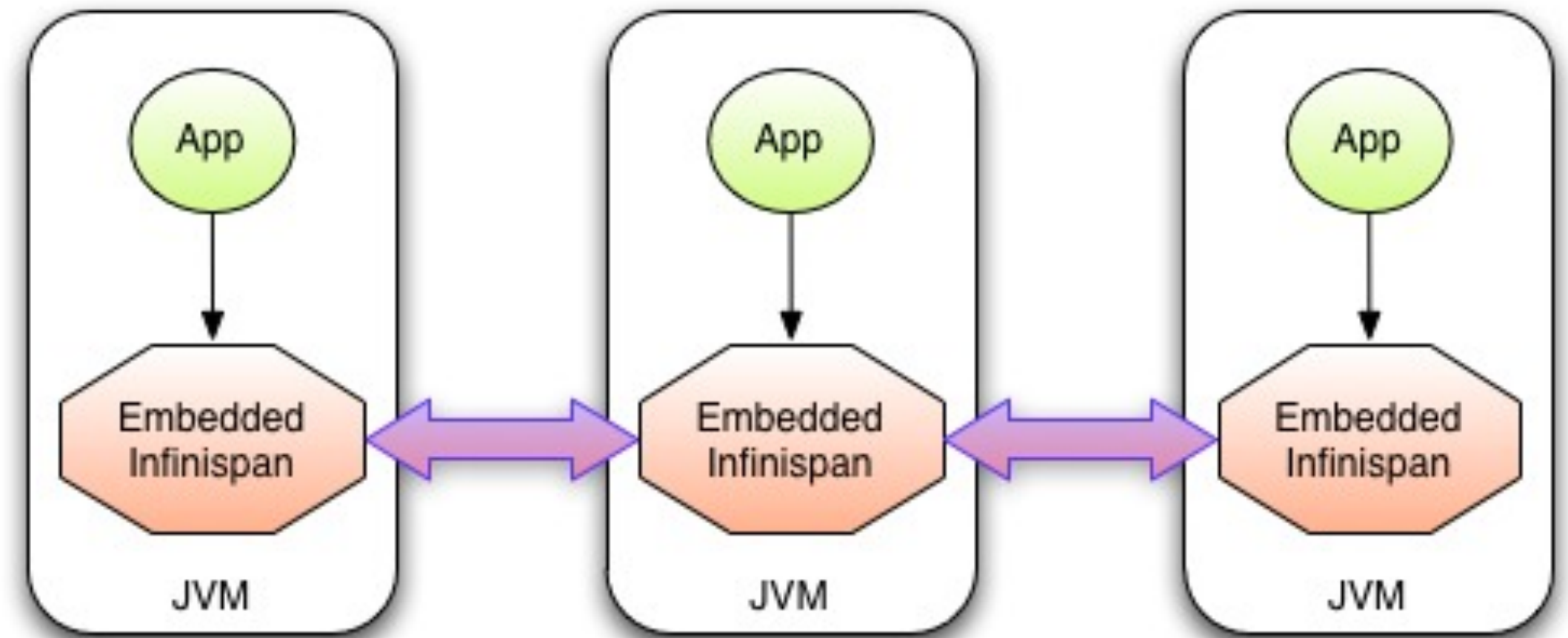
# JGroups

**Reliable multipoint  
communication library at the  
core of Infinispan's cluster  
functionality**



# Total Replication

- Also known as **Replication**
- Data replicated to **all nodes** in the cluster



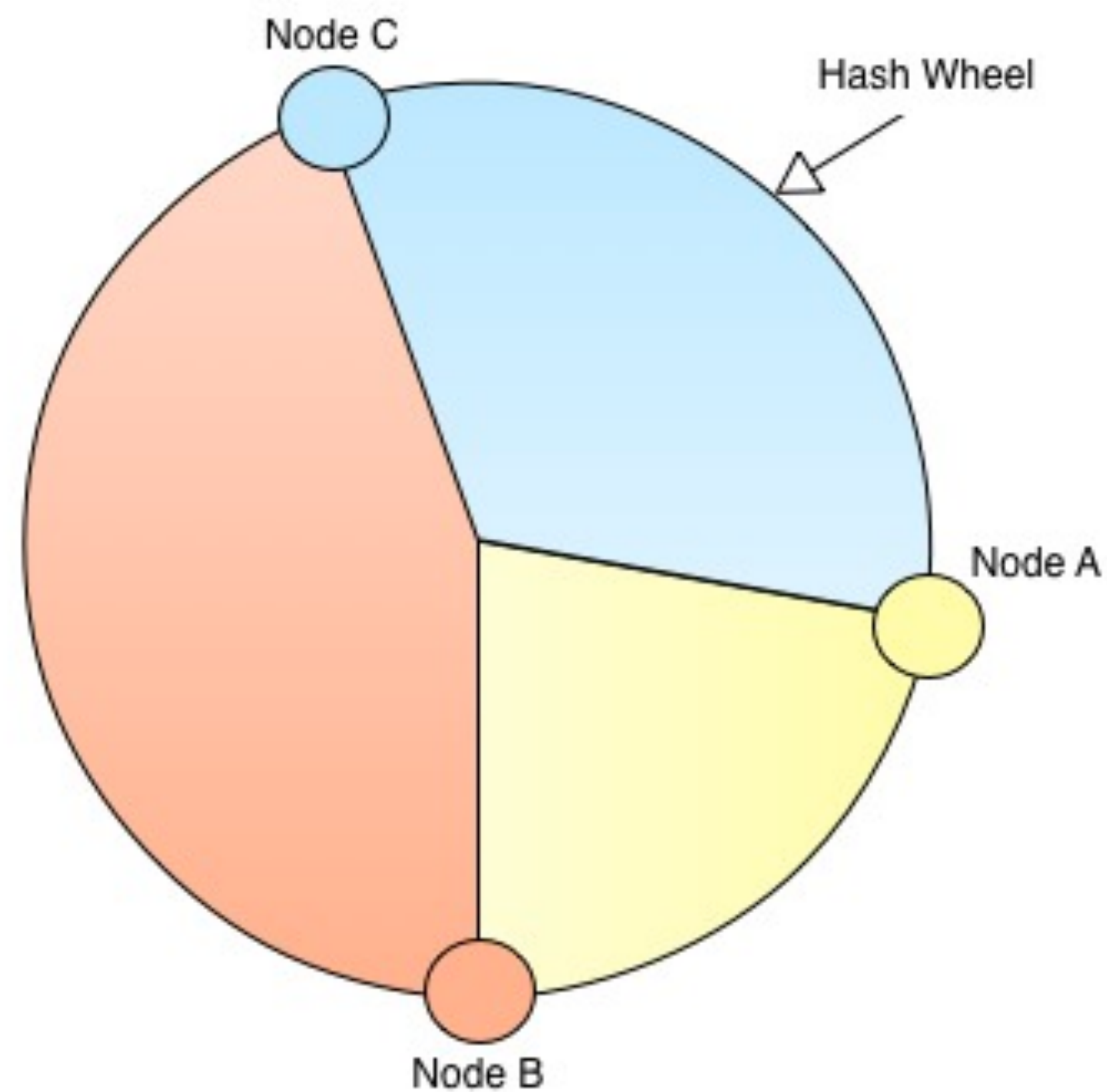
# Partial Replication

Also known as **Distribution**, replicates data to a subset of the cluster...



# How is data distributed??

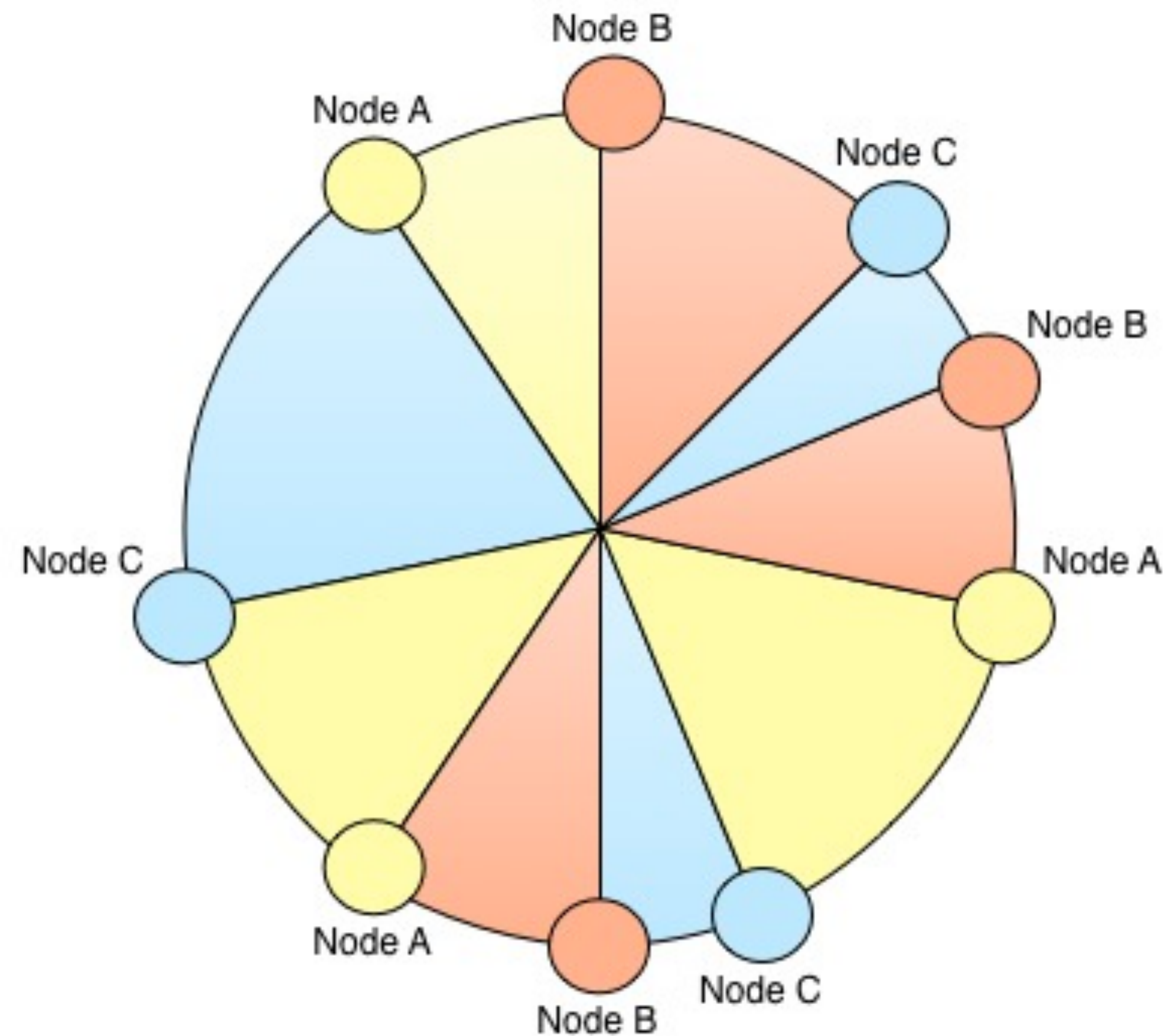
# Consistent Hashing





# Solving unequal distribution

# Virtual Nodes





# Infinispan is not just a cache!

# In-memory data grid

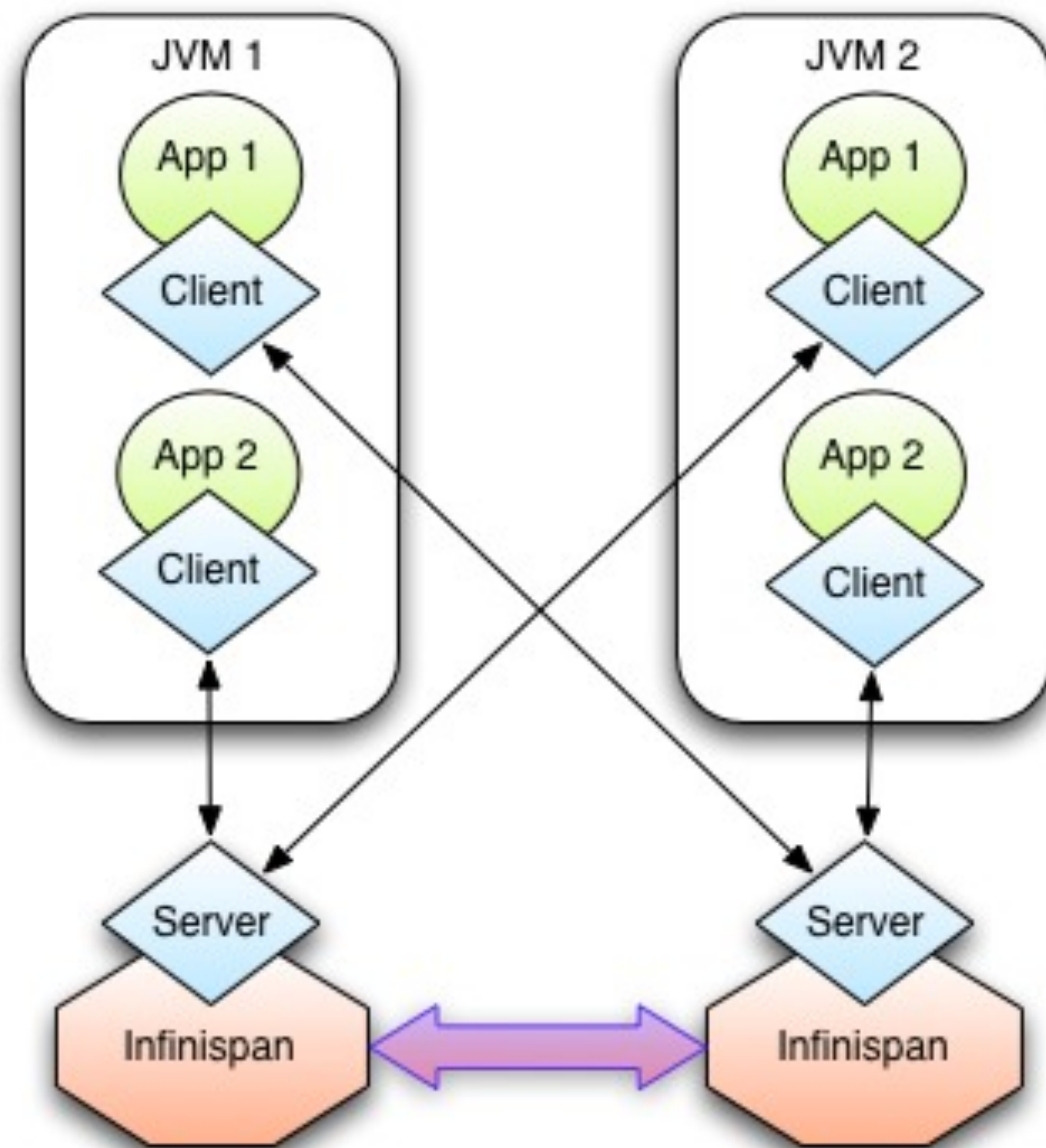
It's a **Fast, Available,  
Distributed, Elastic** data store,  
not just a cache!



# Accessing Infinispan data grid



# Remote Access



- Via protocols :
  - REST
  - Hot Rod



# Why separate data tier?

Manage/tune data tier  
independently, helps build  
stateless application tiers, and  
is easily scalable!

# Querying



# Querying a data grid

- What's in C7?

```
Object p = cache.get("c7");
```

- Where is the white king?





# How to query the grid

Key access, statistics (JMX),  
Map/Reduce or **indexing**  
**stored objects**



# Query Module

Uses Hibernate Search to  
index the stored (annotated)  
objects

# Annotate your objects

```
@ProvidedId @Indexed
public class Book implements Serializable {

    @Field String title;
    @Field String author;
    @Field String editor;
    ...

}
```



# Introducing



# Object/Grid mapper

**JPA for NoSQL engines with  
Infinispan as first supported  
engine**



# OGM objectives

Encourage new data usage patterns within a familiar environment - **JPA** :)



# OGM Example

```
@Entity
public class Dog {
    @Id @GeneratedValue(generator = "uuid")
    public Long getId() { return id; }
    public void setId(Long id) { this.id = id; }
    private Long id;

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
    private String name;

    @ManyToOne
    public Breed getBreed() { return breed; }
    public void setBreed(Breed breed) { this.breed = breed; }
    private Breed breed;
}
```

```
<dependency>
    <groupId>org.hibernate.ogm</groupId>
    <artifactId>hibernate-ogm-core</artifactId>
    <version>3.0.0.Alpha2</version>
</dependency>
```

```
<?xml version="1.0"?>
<persistence version="2.0">
    <persistence-unit name="org.hibernate.ogm.tutorial.jpa" transaction-type="JTA">
        <!-- Use Hibernate OGM provider: configuration will be transparent -->
        <provider>org.hibernate.ogm.jpa.HibernateOgmPersistence</provider>
        <properties>
            <property name="hibernate.transaction.manager_lookup_class"
                value="org.hibernate.transaction.JBossTSSandaloneTransactionManagerLookup" />
        </properties>
    </persistence-unit>
</persistence>
```



# Summary

Infinispan as fast powerful  
local cache that can be  
clustered!

# Summary

Comes with memory control mechanisms, can be plugged with listeners, participate in JTA transactions...



# Summary

Integrates with CDI and JSR-107, and supports total or partial cluster-wide replication

# Summary

Can also be F.A.D.E. data grid, accessible in embedded or remote fashion



# Summary

**Querying** supported via Map/Reduce or indexing, and provides the data grid for **Hibernate OGM**

# Questions

[infinispan.org](http://infinispan.org)

[blog.infinispan.org](http://blog.infinispan.org)

[twitter.com/infinispan](https://twitter.com/infinispan)

[hibernate.org/subprojects/ogm.html](http://hibernate.org/subprojects/ogm.html)

Rate this talk!

<http://speakerate.com/galder>