JBoss Community

Infinispan:
The Path Ahead

Manik Surtani
msurtani@redhat.com
Founder and Project Lead, Infinispan

JBoss Community

Who is Manik?

- R&D Engineer at JBoss by Red Hat
- Founder and Project Lead, Infinispan
- Spec lead, JSR 347
  - Data Grids for Java
- EG representative, JSR 107
  - Temporary Caching for Java

- http://blog.infinispan.org
- http://twitter.com/maniksurtani

Agenda

- Brief summary of Infinispan
- Infinispan to date
- Where we are now
- What the future holds

JBoss Community

What is Infinispan?

- An open source data grid platform
- Written in Java and Scala
  - Not just for the JVM though
- Distributed key/value store
  - Transactional (JTA/XA)
  - Low-latency (in-memory)
  - Optionally persisted to disk
  - Feature-rich

JBoss Community

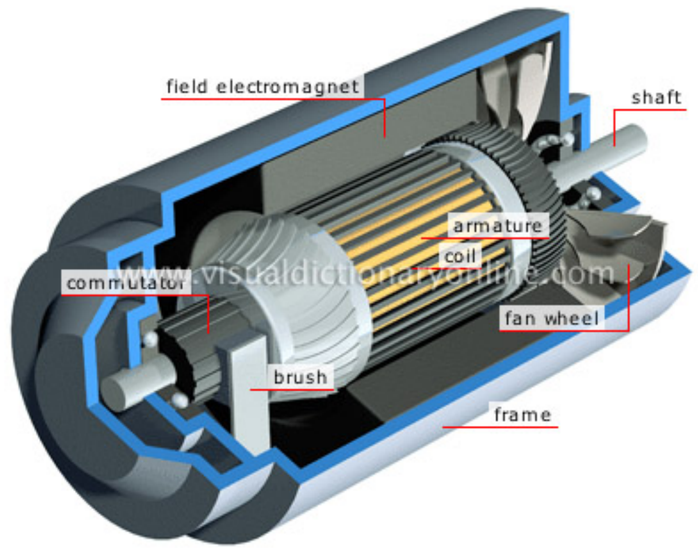
Infinispan

What is Infinispan?

- Open source (LGPL) in-memory Data Grid
- Some concepts from Amazon Dynamo

2 usage modes!

- Embedded
- Client-server

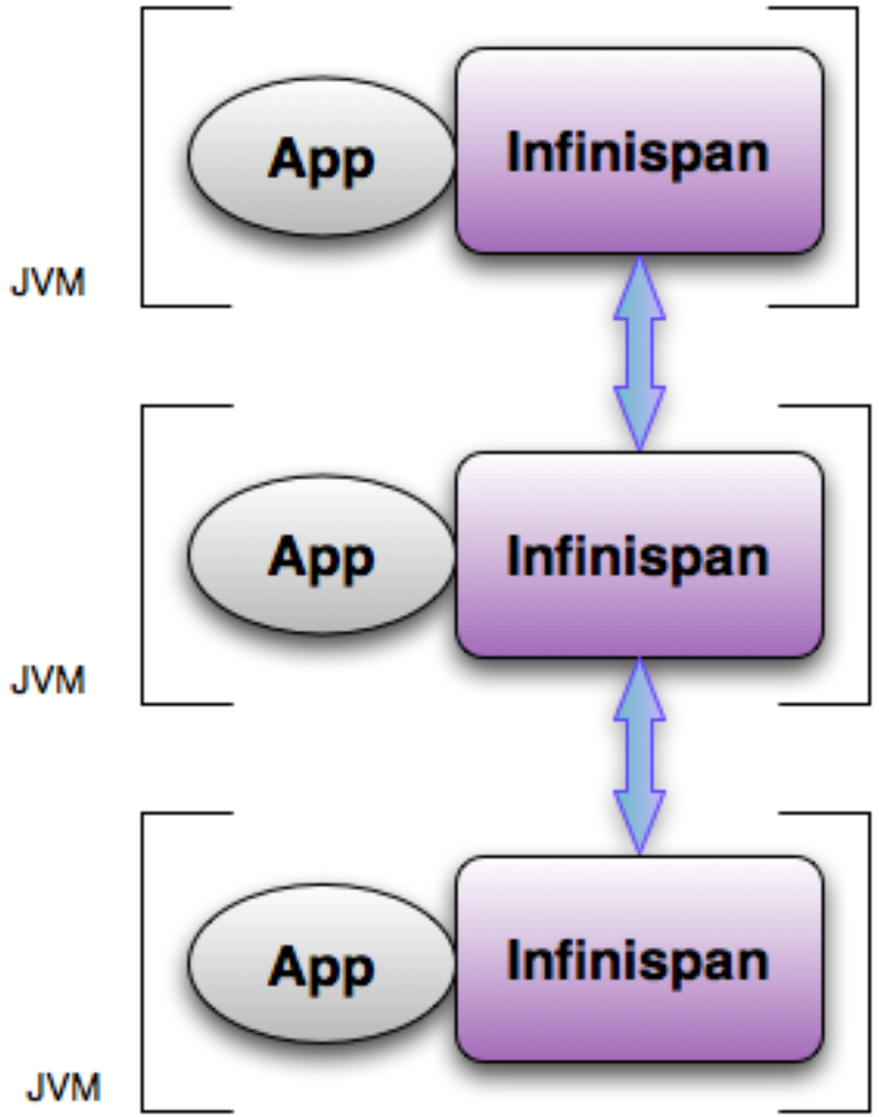


JBoss Community

Embedded Mode

JBoss Community

P2P Embedded Architecture



API

- Map-like key/value store
  - JSR 107 `javax.cache.Cache` interface
  - Asynchronous API

API

- CDI API
- Upcoming JPA-like layer
  - Hibernate OGM
- Other high-level APIs being discussed in the community e.g., ActiveRecord

Transactions

- JTA and XA compliant transactions

Transactions in Infinispan

- Designed with transactions in mind from the start
- Benefits from JBoss Transactions
  - Formerly Arjuna Transaction Manager
  - Supports different levels of XA integration
    - Simple Synchronization
    - Full XA
    - Full XA with recovery
  - Optimistic and pessimistic transactions

Persistent Storage

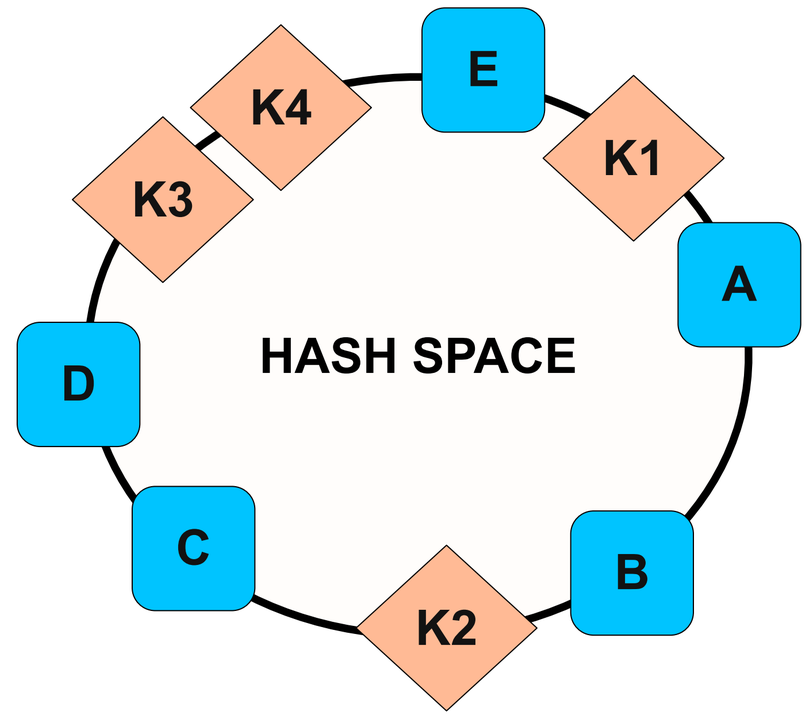
- Not just in memory
  - Support for write though and write behind
  - Support for passivation (spillover to disk)
- Support for preloading/warm starts
- Ships with implementations for:
  - BTree file system based
    - BerkeleyDB, JDB
  - JDBC
  - Cloud storage
    - S3, CloudFiles, Azure, etc via JClouds

Replication of data

- Uses 2 clustered modes:
  - Replication (Full Replication)
  - Distribution (Partial Replication)
- Replication
  - very fast for reads
  - more expensive for writes
  - however has scalability limitations
- Distribution
  - Linear and far more scalable
  - Access to very large heaps

Distributed Data

- Consistent hash based distribution
- Fast, deterministic lookup of keys
  - Node position = `abs(hash(address)) % HASH_SPACE`
- Configurable num_owners
- Support for topologies



Distributed Data – L1 cache

- Remote GETs can get expensive!
- Hence the need for L1
  - “Near Cache” in other vendors’ parlance
- L1 entries have limited lifespan
- L1 entries are also invalidated
  - Smart invalidating algorithms

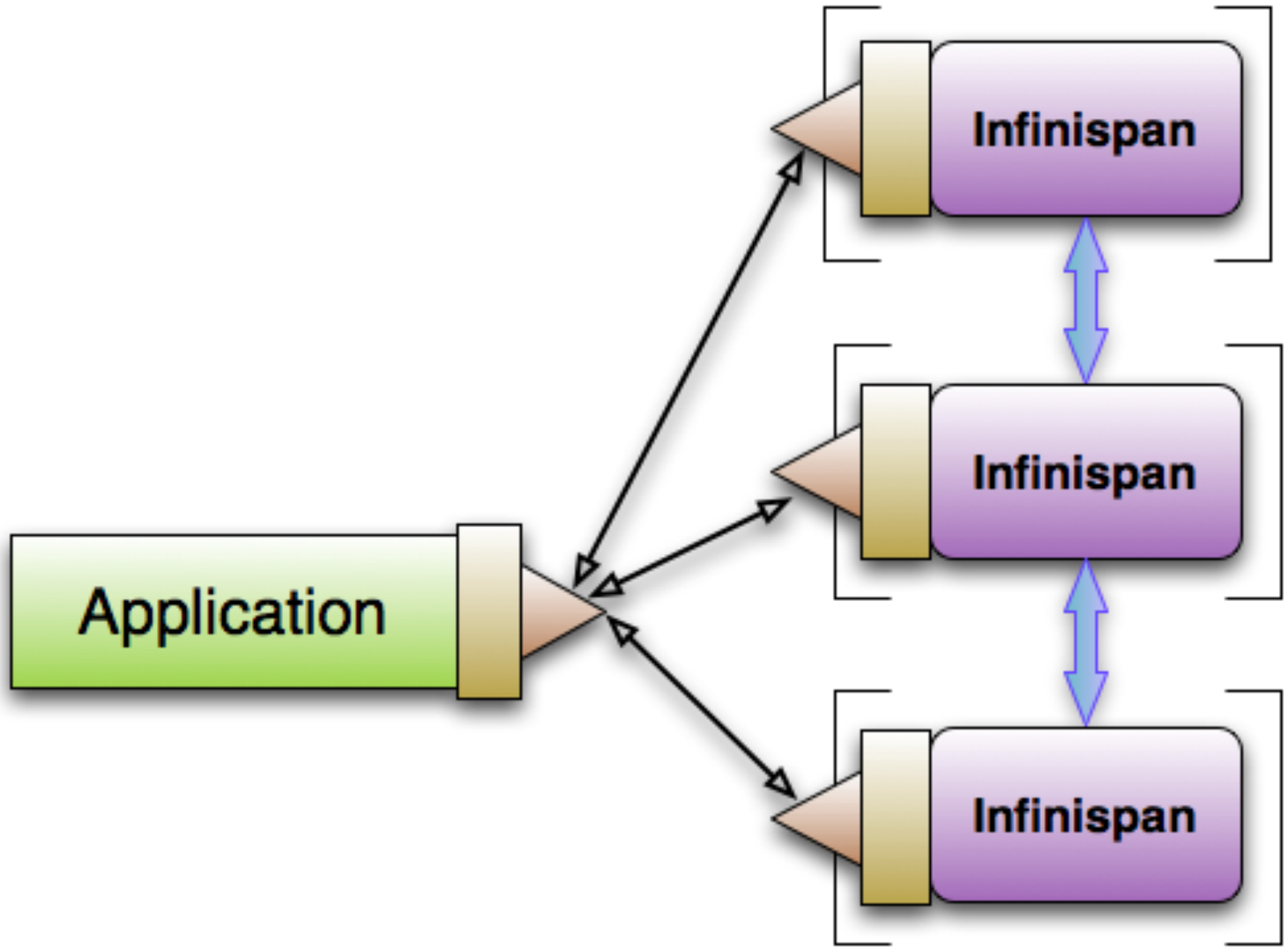
Code Execution

- Move logic to data, not vice versa
- Implemented using familiar Java Executors and Callable interfaces and APIs
- Map/Reduce
  - Compresses calculation results

Client/Server Mode

JBoss Community

Client/Server Architecture



WTF is Hot Rod?

- Wire protocol for client server communications
- Open
- Language independent
- Built-in failover and load balancing
- Smart routing

Server Endpoint Comparison

	Protocol	Client Libraries	Clustered?	Smart Routing	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, Ruby	**Yes**	**Yes**	Dynamic

Why use this stuff?

JBoss Community

Why use distributed caches?

- Cache data that is expensive to retrieve/calculate
  - E.g., from a database
- The need for fast, low-latency data access
  - Performance or time-sensitive applications
- Very commonly used in:
  - Financial Services industry
  - Telcos
  - Highly scalable e-commerce

Data Grids as Clustering Toolkits

- To introduce high availability and failover to frameworks
  - Commercial and open source frameworks
  - In-house frameworks and reusable architectures
- Delegate all state management to the data grid
  - Framework becomes stateless and hence elastic
- Very important for cloud

JBoss Community

Data Grids used as Cloud Storage (NoSQL key/value store)

JBoss Community

Cloud Storage

- Traditional mechanisms (RDBMSs) are hard to deal with
- Clouds are ephemeral
- All cloud components are expected to be:
  - elastic
  - highly available

Past, Present and Future

JBoss Community

Some background

- JBoss Cache
  - A clustering toolkit for JBoss AS 4.x ~ 6.x
  - Used to replicate HTTP and EJB sessions for high availability
  - Often used as a data grid and key/value store
  - But **not** designed for that purpose!
- Infinispan designed both as a data grid as well as a clustering toolkit
  - JBoss Cache in maintenance mode for several years now
  - All focus on Infinispan

Where have we come?

Infinispan

4.0 “Starobrno”

- Basic distribution and replication
- Asynchronous API
- Write through, write behind and passivation
- FIFO and LRU Eviction
- Listeners
- Transactions
- Querying

RELEASED

Feb 2010

Where have we come?

Infinispan

4.1 “Radegast”

- Server modules
  - Memcached
  - Hot Rod
  - REST
- Hot Rod client for Java
- Adaptive eviction based on LIRS

RELEASED

Sept 2010

Where have we come?

Infinispan

5.0 “Pagoa”

- Distributed Executors
- Map/Reduce
- Grouping API
- XA Recovery
- Virtual Nodes
- CDI API
- Smart L1 invalidation

RELEASED

Aug 2011

JBoss Community

Where have we come?

Infinispan

5.1 “Brahma”

- Optimistic and Pessimistic transactions
- Chunked state transfer

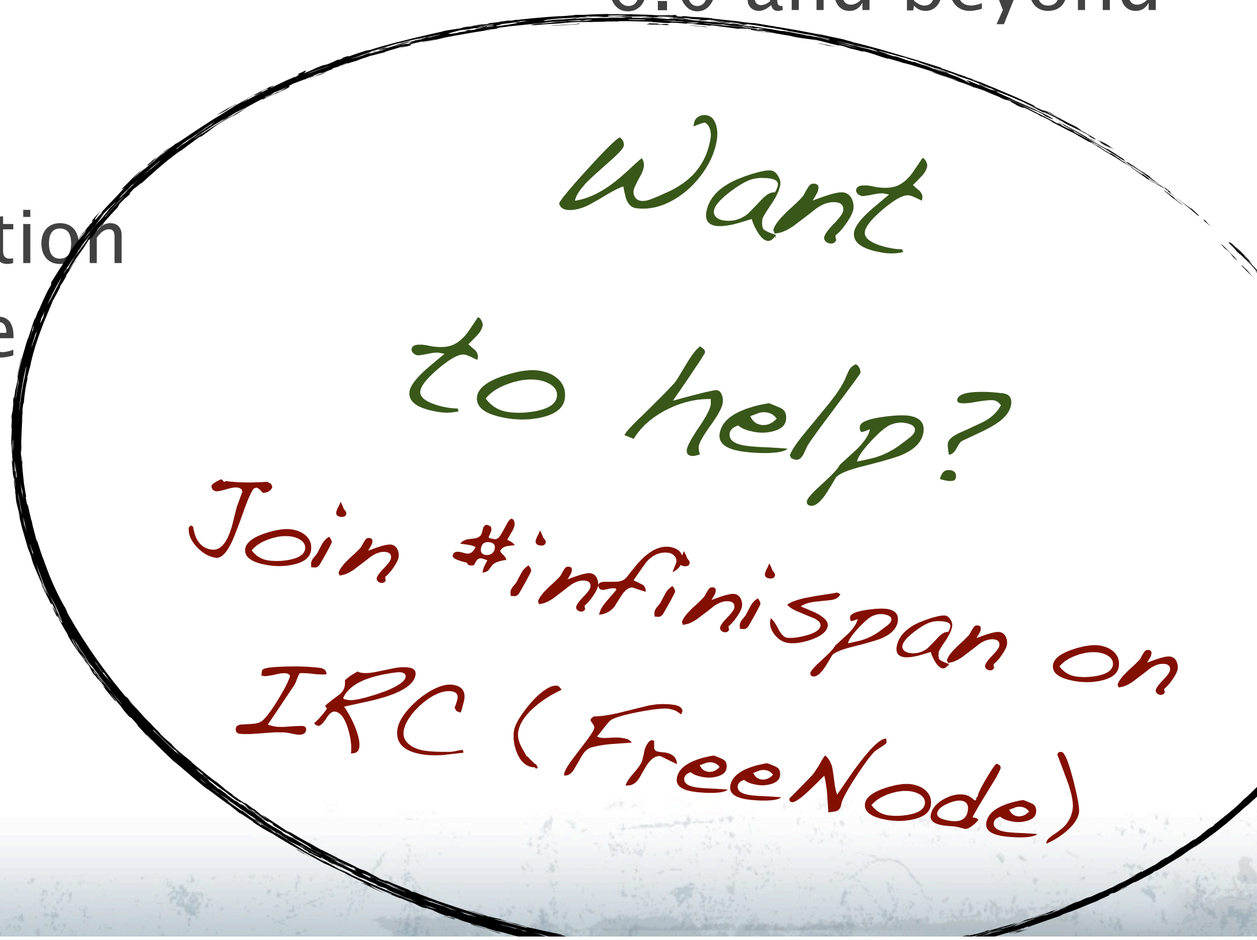
JBoss Community

Where have we come?

- Cross-datacentre/WAN replication
- Eventual consistency
- Distributed queries
- Non-blocking state transfer
- Schematic entries and validation
- Support for more persistence engines
- ... ?

Infinispan

6.0 and beyond



JBoss Community

To Summarize

- A brief overview of Infinispan
- Where it has been
- Where is today
- What the future holds

Questions & More Info

- http://www.infinispan.org
- http://twitter.com/infinispan

JBoss Community