JUDCon

JBoss Users & Developers Conference

London:2011

Who am I?

Sanne Grinovero

Software Engineer at Red Hat

- Hibernate, especially Search
- Infinispan, focus on Query and Lucene
- Hibernate OGM
- Apache Lucene
- JGroups

Our Index

- Searching in Infinispan
  - Map/Reduce
  - Fulltext indexing
- Infinispan Query engine
- Clustering a Lucene index
- Cloud deployed applications
- Future

Infinispan

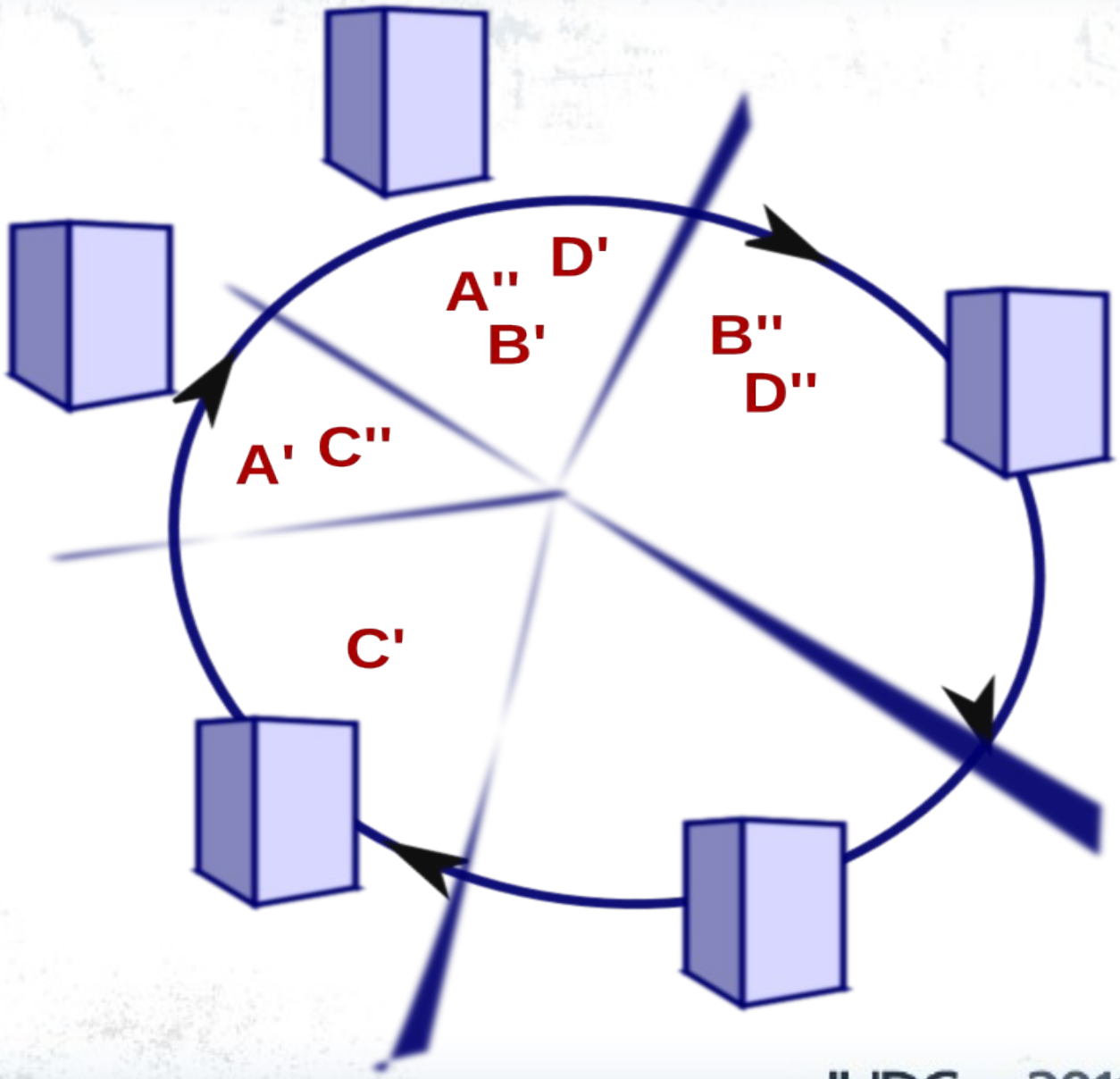
- An advanced multi-node cache
- A transactional scalable datagrid targeting high performance and cloud
- A “NoSQL database”, a *key-value* store
  - How do you query a key value store?

```
SELECT * FROM GRID
```

To Query a Grid

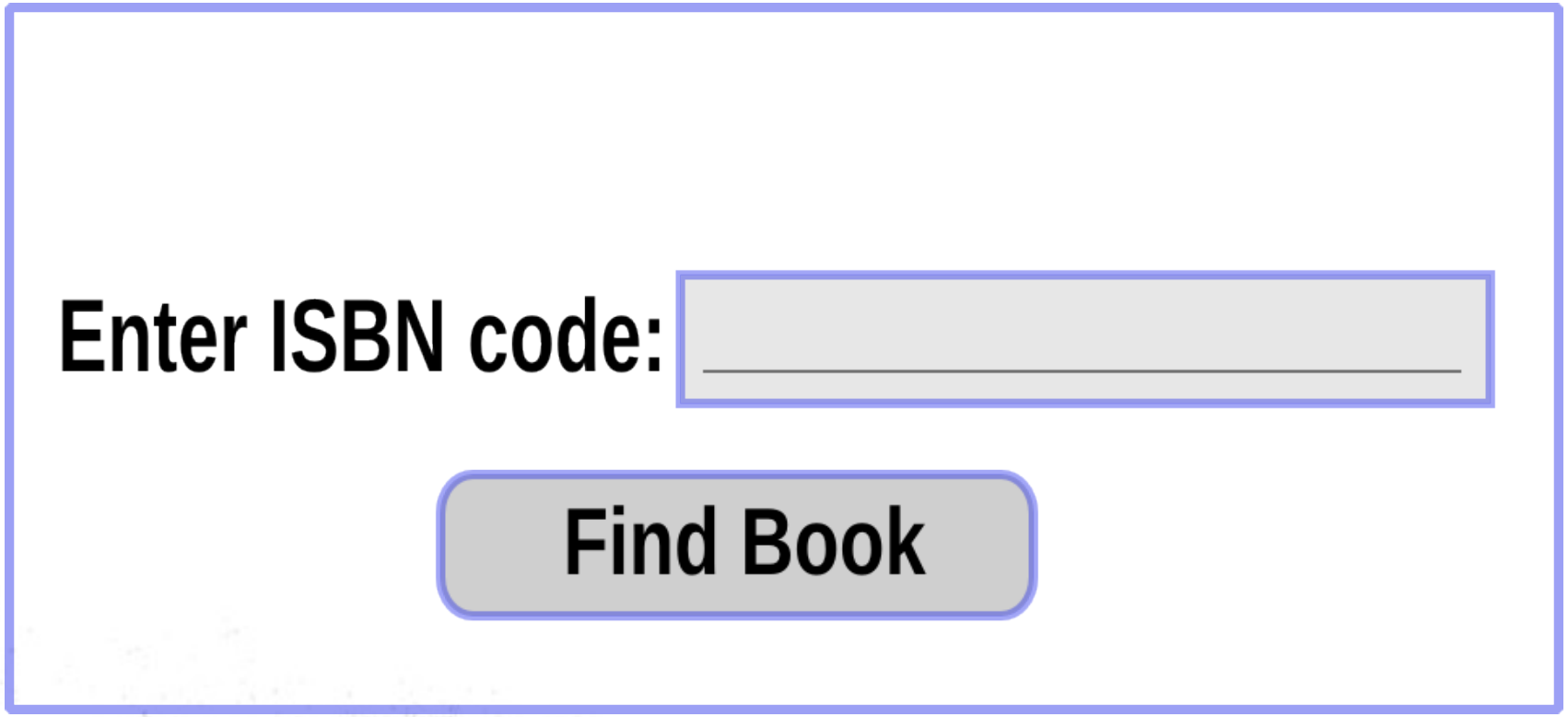
- What's in C7 ?

```
Object v =
 cache.get(“c7”);
```


A''
D'
B'
B''
D''
A'
C''
C'

If you don't know the key, no way to find the value

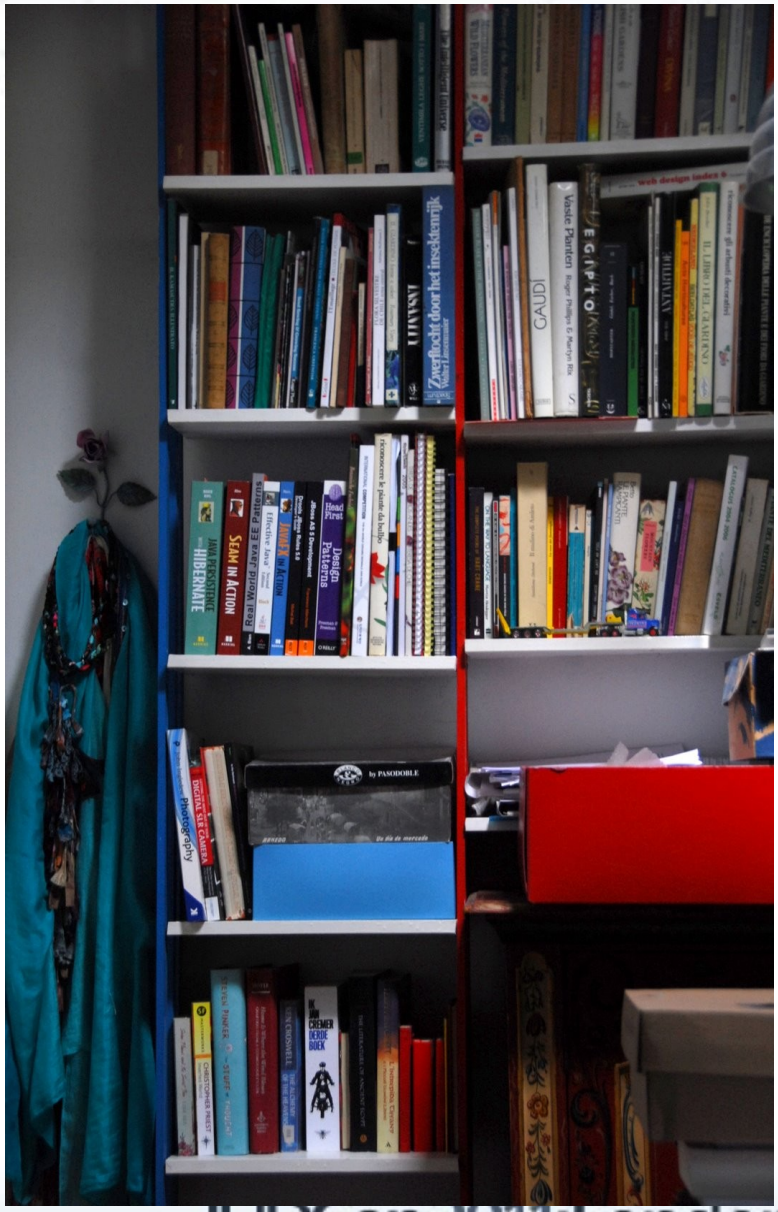
Some services have no chance:



Let's test my bookshelf

- Where's Hibernate Search in Action?
- Could you hand me

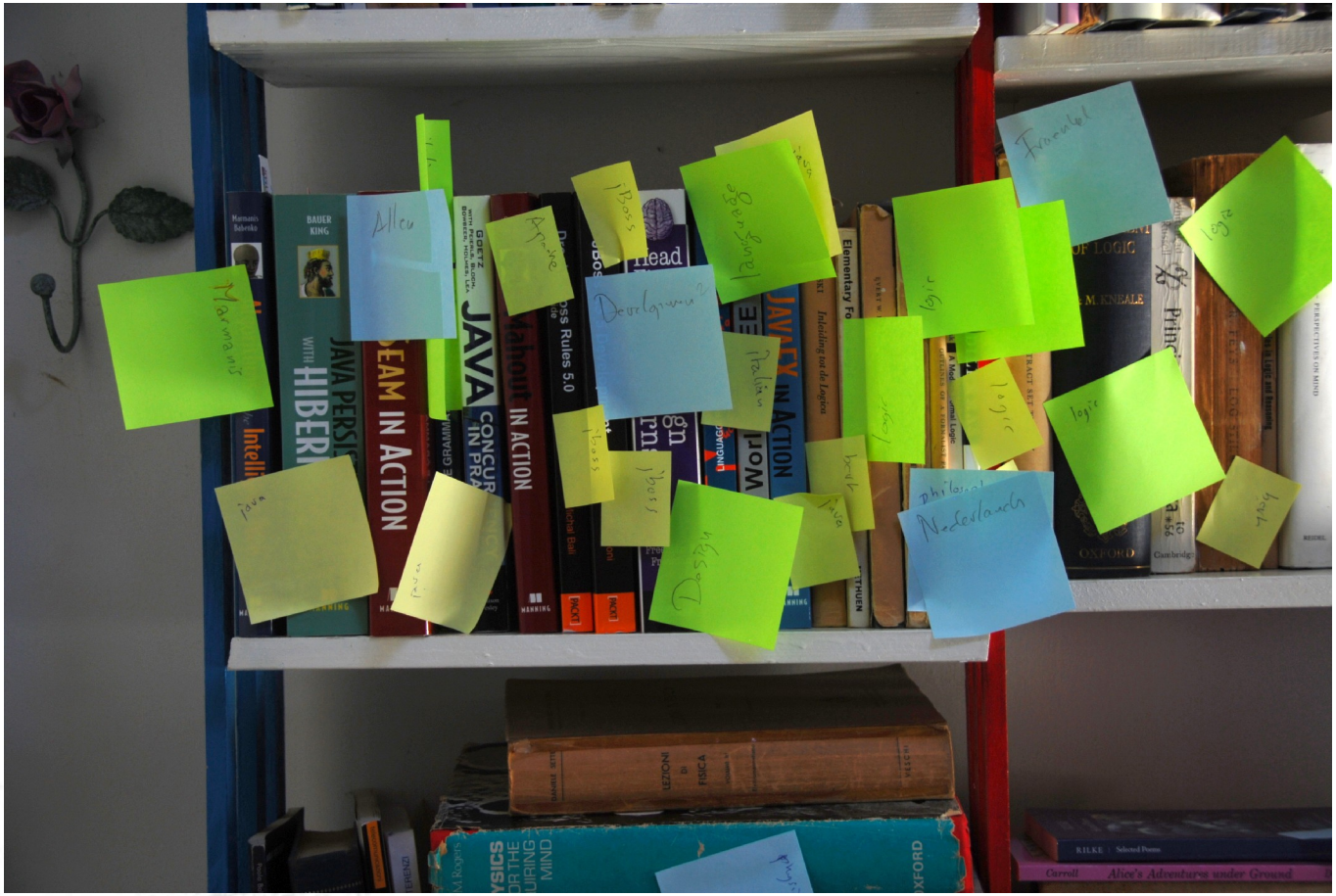
  ISBN 978-1-933988-17-7 ?
- How many books about Gaudí ?



A *real-world* example

A *real-world* example

A *real-world* example



A *real-world* example

Bookshelves don't scale

How to implement the bookshelf features on a k/v?

- Where's “Hibernate Search in Action”?
- Can you hand me

  “ISBN 978-1-933988-17-7” ?
- How many books about Gaudí ?

Most document based NoSQLs support Map/Reduce

- Infinispan does not focus on documents
  - That won't stop you from using any format JSON, XML, YAML, Java:

```
public class Book implements Serializable {

   final String title;
   final String author;
   final String editor;

   public Book(String title, String author, String editor) {
      this.title = title;
      this.author = author;
      this.editor = editor;
   }

}
```

Iterate & collect

```
class TitleBookSearcher implements
                          Mapper<String, Book, String, Book> {
  final String title;
  public TitleBookSearcher(String t) { title = t; }
  public void map(String key, Book value, Collector collector){
     if ( title.equals( value.title ) )
        collector.emit( key, value );
  }

class BookReducer implements
                          Reducer<String, Book> {
  public Book reduce(String reducedKey, Iterator<Book> iter) {
         return iter.next();
  }
}
```

How to implement the bookshelf features on a k/v?

- ✔ Where's “Hibernate Search in Action”?
- ✔ Can you hand me “ISBN 978-1-933988-17-7” ?
- ✗ How many books ***about*** “Shakespeare” ?
  - To properly score fulltext results we need to consider relative term frequencies on the whole corpus
  - Pre-tagging is a poor choice

Apache Lucene

- Open source Apache™ top level project
- Countless products and sites use it
- Integrates in Hibernate via Hibernate Search
- Clusterable via Infinispan

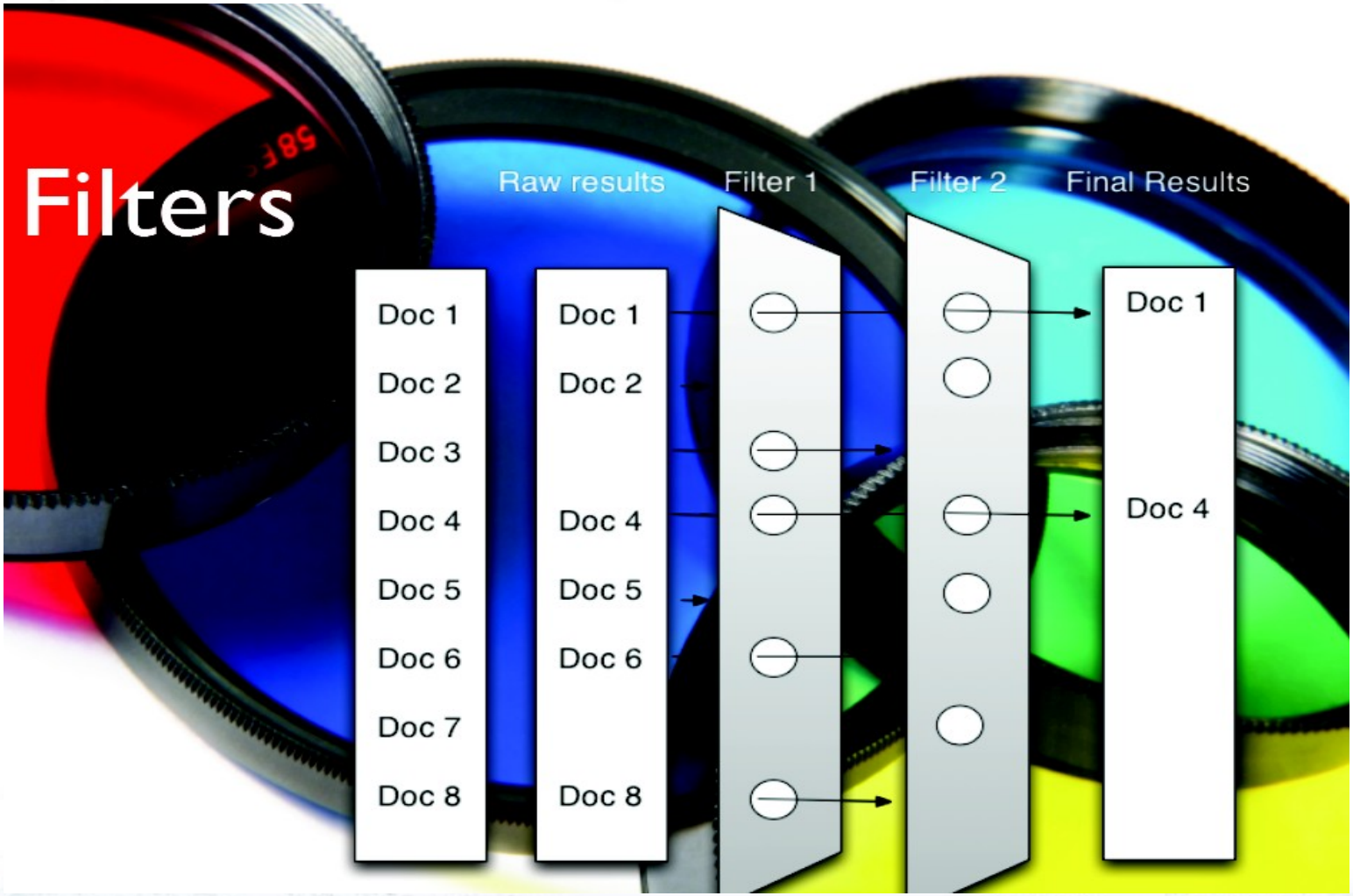
What does Lucene get us?

- Similarity scoring searches
- Advanced text analysis
  - Sinonyms, Stopwords, Stemming, ...
- Reusable declarative Filters
- TermVectors
- MoreLikeThis
- Faceted Search
- Speed!

Lucene: Stopwords

a, able, about, across, after, all, almost, also, am, among, an, and, any, are, as, at, be, because, been, but, by, can, cannot, could, dear, did, do, does, either, else, ever, every, for, from, get, got, had, has, have, he, her, hers, him, his, how, however, i, if, in, into, is, it, its, just, least, let, like, likely, may, me, might, most, must, my, neither, no, nor, not, of, off, often, on, only, or, other, our, own, rather, said, say, says, she, should, since, so, some, than, that, the, their, them, then, there, these, they, this, tis, to, too, twas, us, wants, was, we, were, what, when, where, which, while, who, whom, why, will, with, would, yet, you, your

Filters


Filters
Raw results
Filter 1
Filter 2
Final Results
Doc 1
Doc 2
Doc 3
Doc 4
Doc 5
Doc 6
Doc 7
Doc 8
Doc 1
Doc 2
Doc 4
Doc 5
Doc 6
Doc 8
Doc 1
Doc 4

Faceted Search

Shop All Departments | Search | Computers & Internet | Hibernate Search

Books | Advanced Search | Browse Subjects | New Releases | Bestsellers | Th

Department
- ‹ Any Department
  - ‹ Books
    - **Computers & Internet**
      - Programming (14)
      - Computer Science (4)
      - Databases (2)
      - Software (2)
      - Web Development (2)
      - Networking (1)
      - Home Computing (1)

Format
- ☐ Paperback (15)

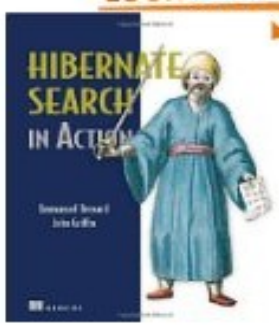
Author
- **Any Author**
- Joe Vitale (1)

Shipping Option (What's this?)
- **Any Shipping Option**
- Free Super Saver Shipping

Books › Computers & Internet › "Hibernate Search"

Showing 1 - 12 of 15 Results

1. LOOK INSIDE!



Hibernate Search in Action

★★★★★ (3 customer reviews)

Formats

Paperback

Order in the next **2 hours** to get it by **Monday, Apr 18.**

Only 1 left in stock - order soon.

~~$49.~~

Eligible for **FREE** Super Saver Shipping.

Excerpt - Page 1: "... breaking the susp

Surprise me! See a random page in thi

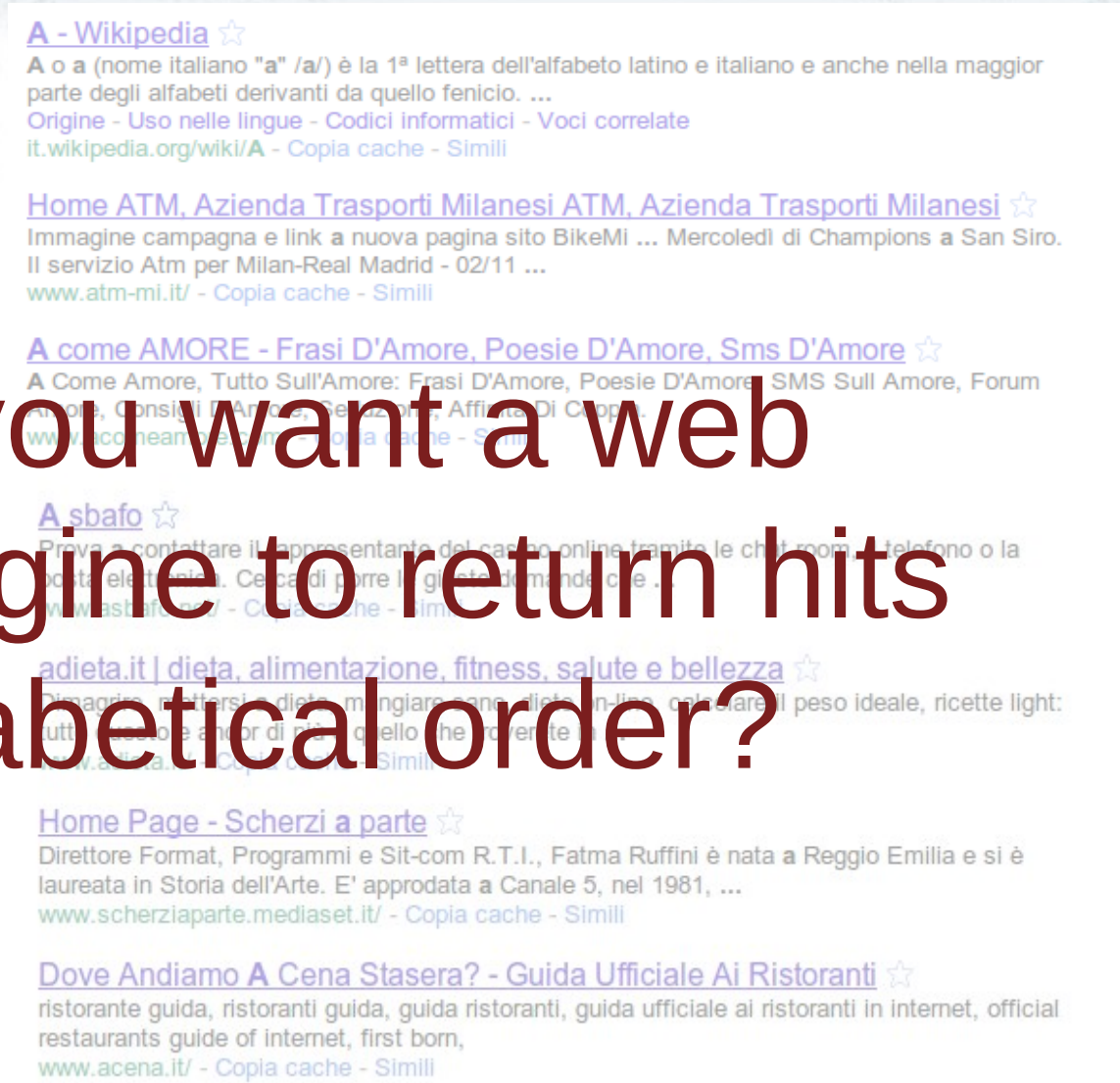
2. LOOK INSIDE!

Spring Persistence with Hib
(Nov 2, 2010)

★★★★☆ (5 customer reviews)

Formats

Paperback



Would you want a web Search engine to return hits in alphabetical order?

The downsides

- Requires an Index
  - in memory
  - on filesystem
  - in Infinispan
- Made of immutable segments
  - Optimized for search speed, not for updates
- A world of strings and frequencies

Infinispan Query quickstart

- Enable it in configuration
- Have infinispan-query.jar in your classpath
- Annotate your POJO values to specify what to index

```
<dependency>
    <groupId>org.infinispan</groupId>
    <artifactId>infinispan-query</artifactId>
    <version>5.1.0.BETA3</version>
</dependency>
```

Enable Infinispan Query, programmatically

```
Configuration c = new Configuration()
    .fluent()
        .indexing()
        .addProperty(
"hibernate.search.default.directory_provider",
"ram")
        .build();

CacheManager manager = new DefaultCacheManager(c);
```

Enable Query in Infinispan XML configurations

```
<?xml version="1.0" encoding="UTF-8"?>
<infinispan
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:schemaLocation="urn:infinispan:config:5.0
http://www.infinispan.org/schemas/infinispan-config-5.0.xsd"
            xmlns="urn:infinispan:config:5.0">
<default>
    <indexing enabled="true" indexLocalOnly="true">
        <properties>
            <property name="hibernate.search.option1" value="..." />
            <property name="hibernate.search.option2" value="..." />
         </properties>
    </indexing>
</default>
```

Annotate your model

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

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

   public Book(String title, String author, String editor) {
      this.title = title;
      this.author = author;
      this.editor = editor;
   }

}
```

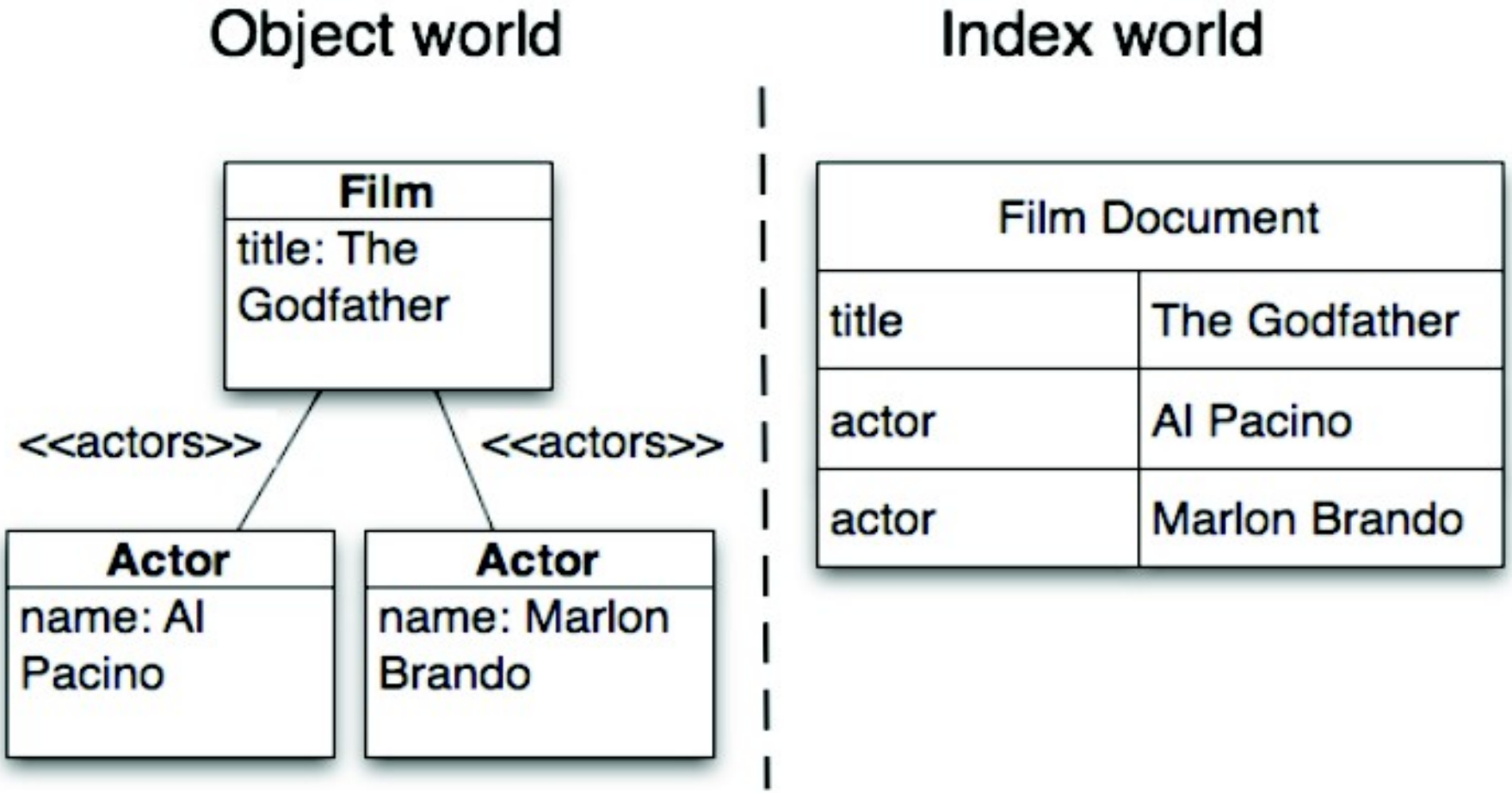
Run a Query

```
SearchManager qf = Search.getSearchManager(cache);

Query query = qf.buildQueryBuilderForClass(Book.class)
    .get()
        .phrase()
            .onField("title")
            .sentence("in action")
    .createQuery();

List<Object> list = qf.getQuery(query).list();
```

The code

- Integrates Hibernate Search
  - Listen to Hibernate events & transactions
    - Infinispan events & transactions
  - Maps Java types and model graphs to Lucene Documents
  - Thin-layer design

Index mapping



declarative analyzers

```
@Entity @Indexed
@AnalyzerDef(name = "frenchAnalyzer", tokenizer =
  @TokenizerDef(factory=StandardTokenizerFactory.class),filters = {
     @TokenFilterDef(factory = LowerCaseFilterFactory.class),
     @TokenFilterDef(factory = SnowballPorterFilterFactory.class,
         params = {@Parameter(name = "language", value = "French")})
})
public class Book {

   @Field(index=Index.TOKENIZED, store=Store.NO)
   @Analyzer(definition = "frenchAnalyzer")
```

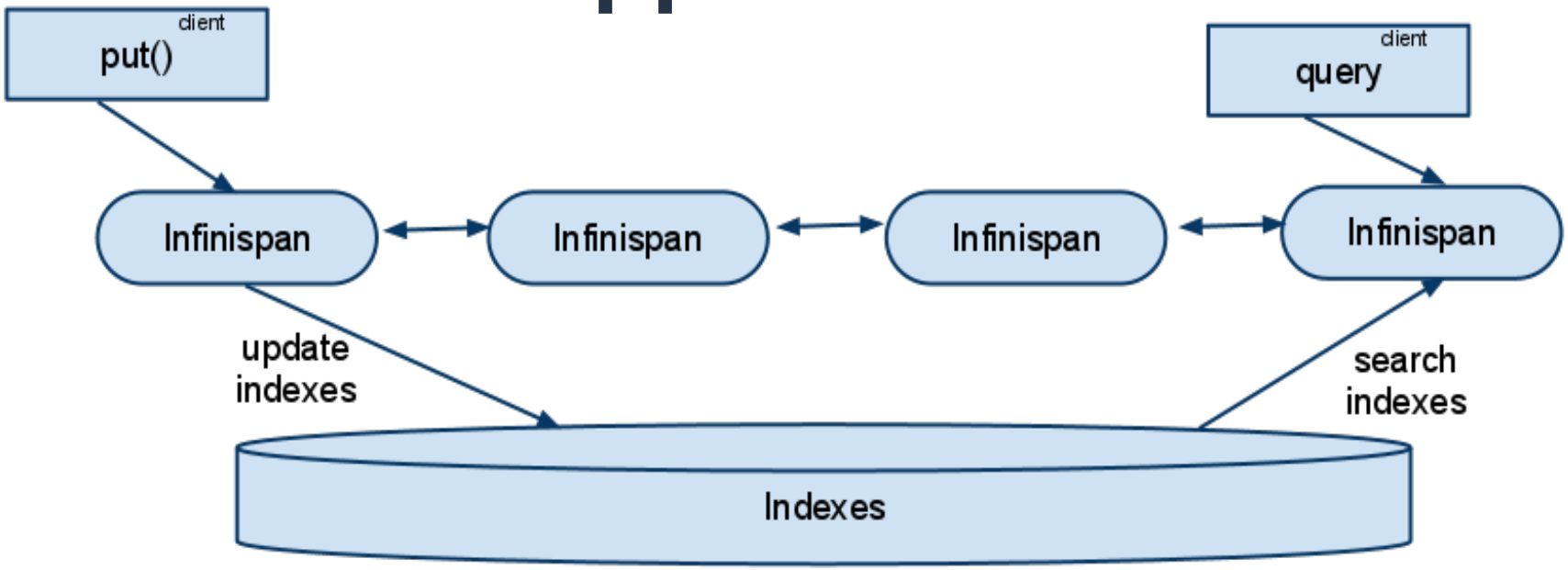
Query test

https://github.com/infinispan/infinispan

I'm not going in details..

```
org.apache.lucene.search.Query luceneQuery =
        queryBuilder.phrase()
                .onField( "description" )
                .andField( "title" )
                .sentence( "a book on highly scalable query engines" )
                .enableFullTextFilter( “ready-for-shipping” )
                .createQuery();

CacheQuery cacheQuery =
                    searchManager.getQuery( luceneQuery, Book.class);
List<Book> objectList = cacheQuery.list();
```

Architecture: simplest approach

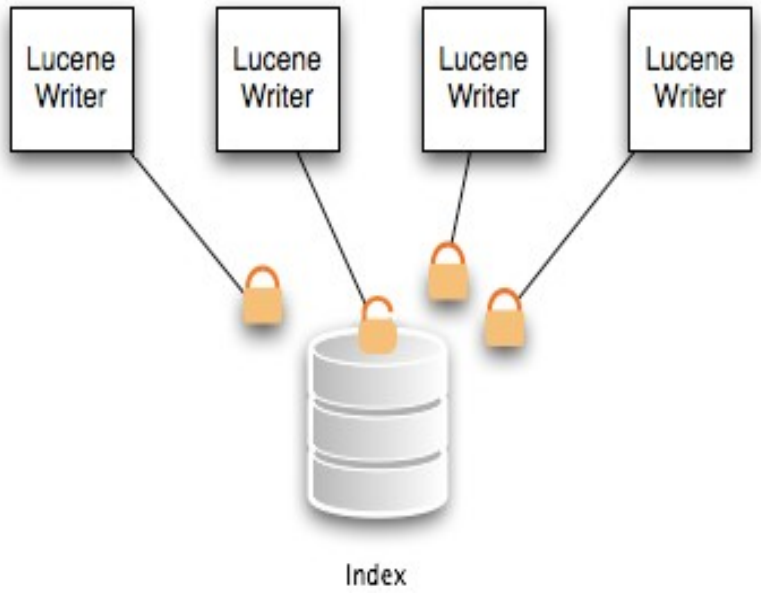


Works for DIST, INVAL and REPL

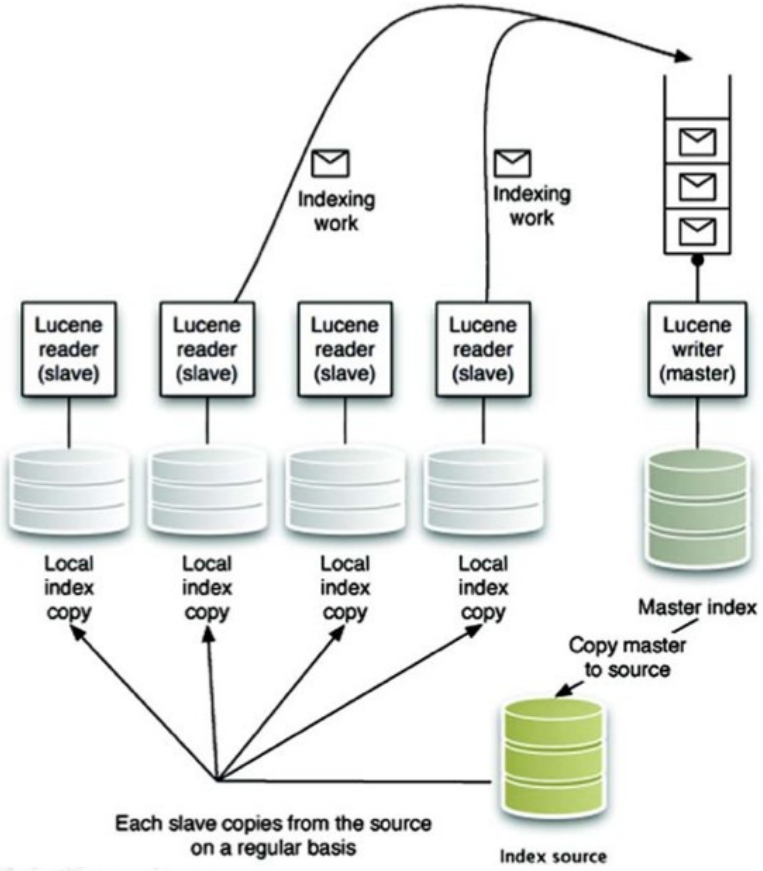
sharedIndexes = true
indexLocalOnly = true

Scalability issues

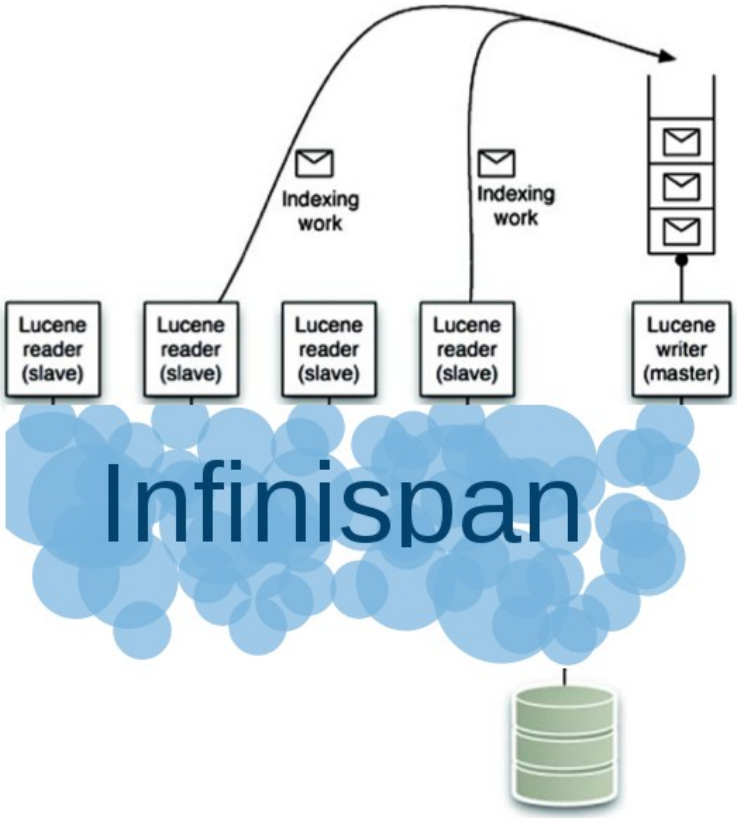
- Global writer locks
- NFS based index sharing *very* tricky



Queue-based clustering (via filesystem)



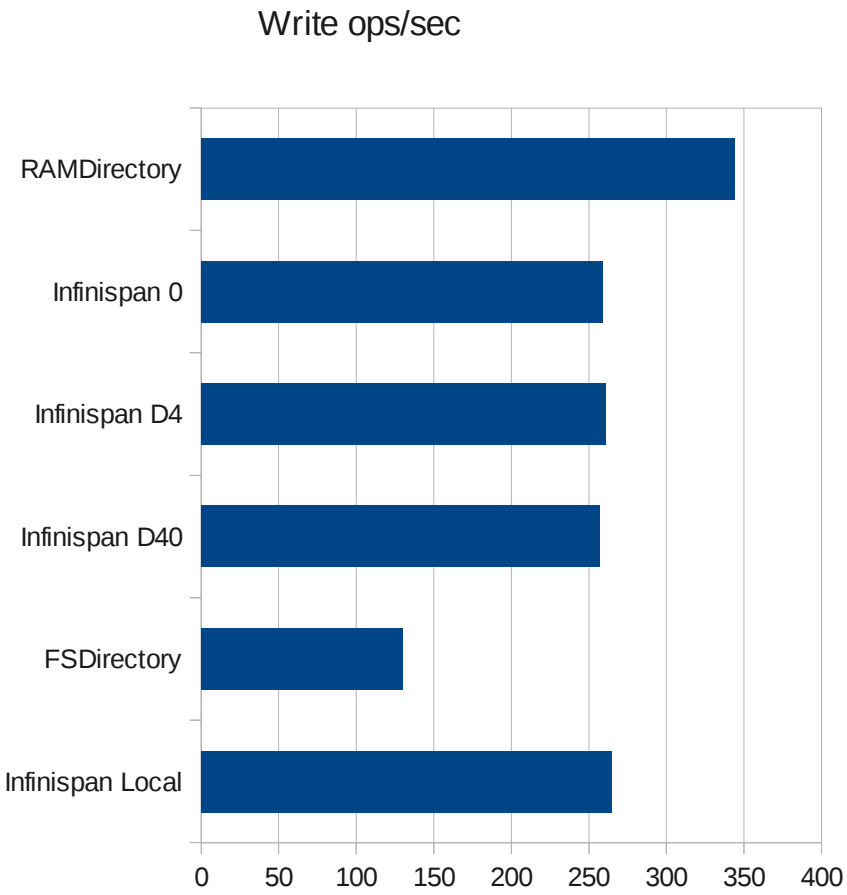
Index *stored* in Infinispan

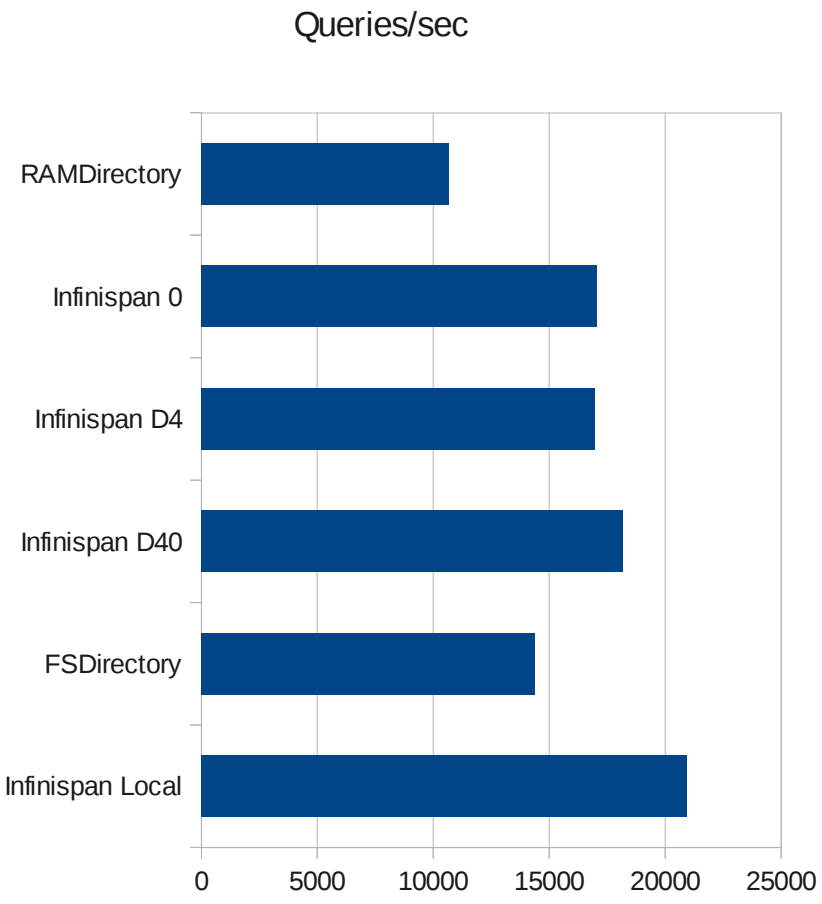


Clustering “native” Lucene access

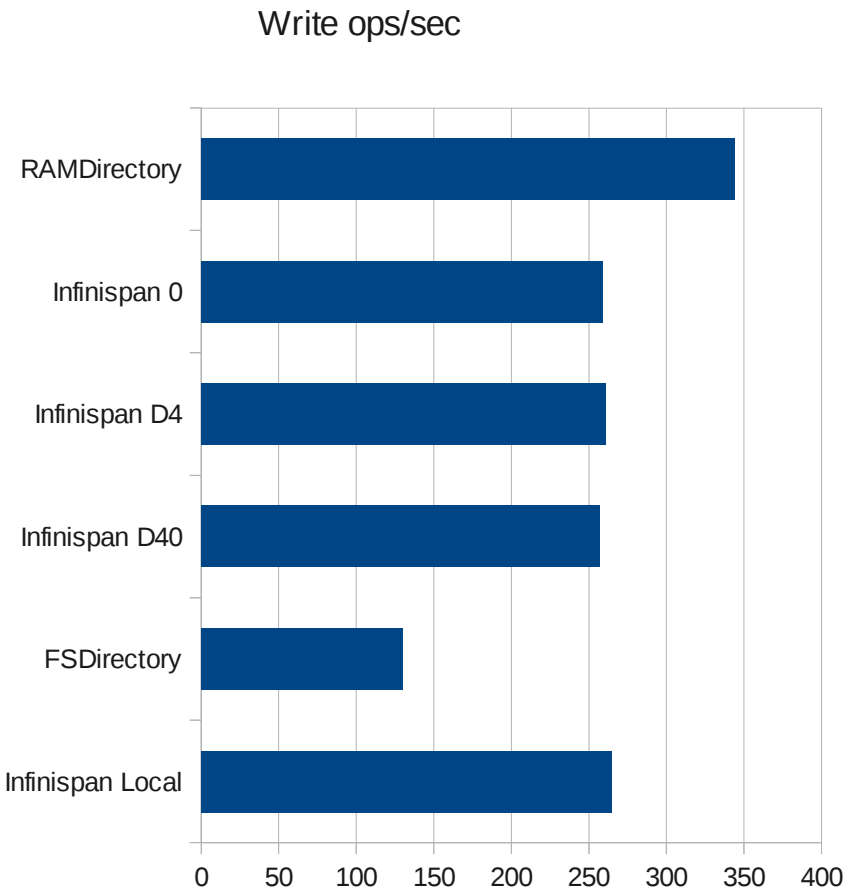
- Using org.apache.lucene directly
  - Distributed on multiple nodes
  - On any cloud

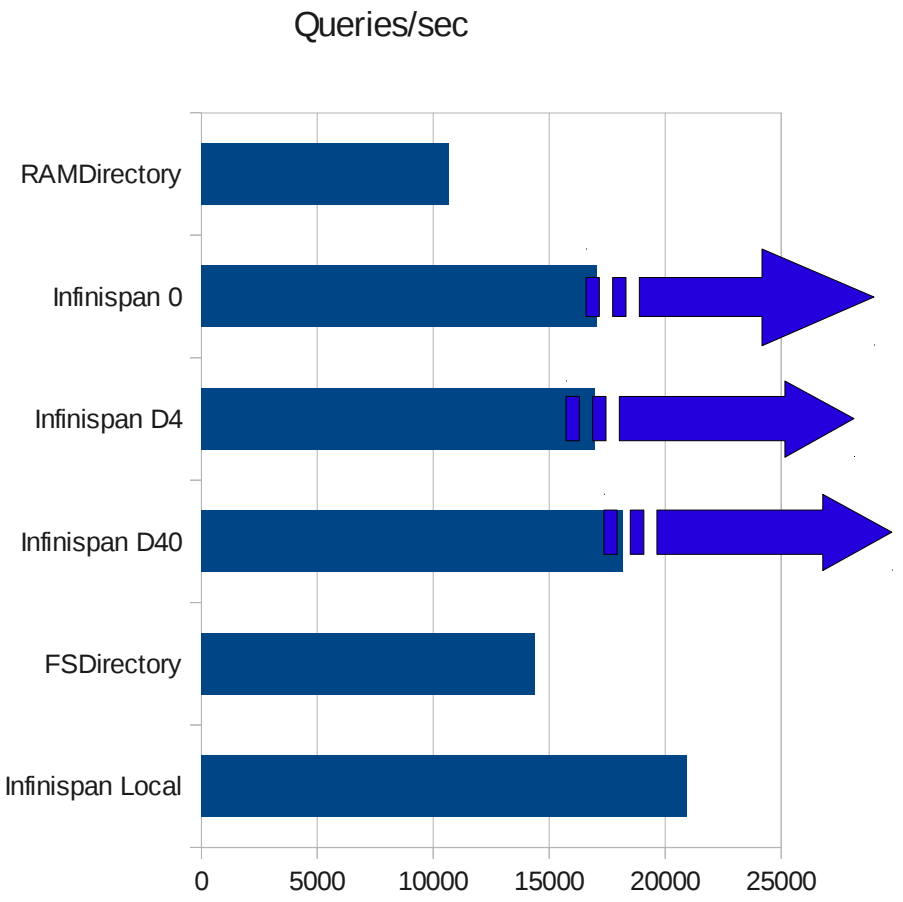
Single node *performance idea*



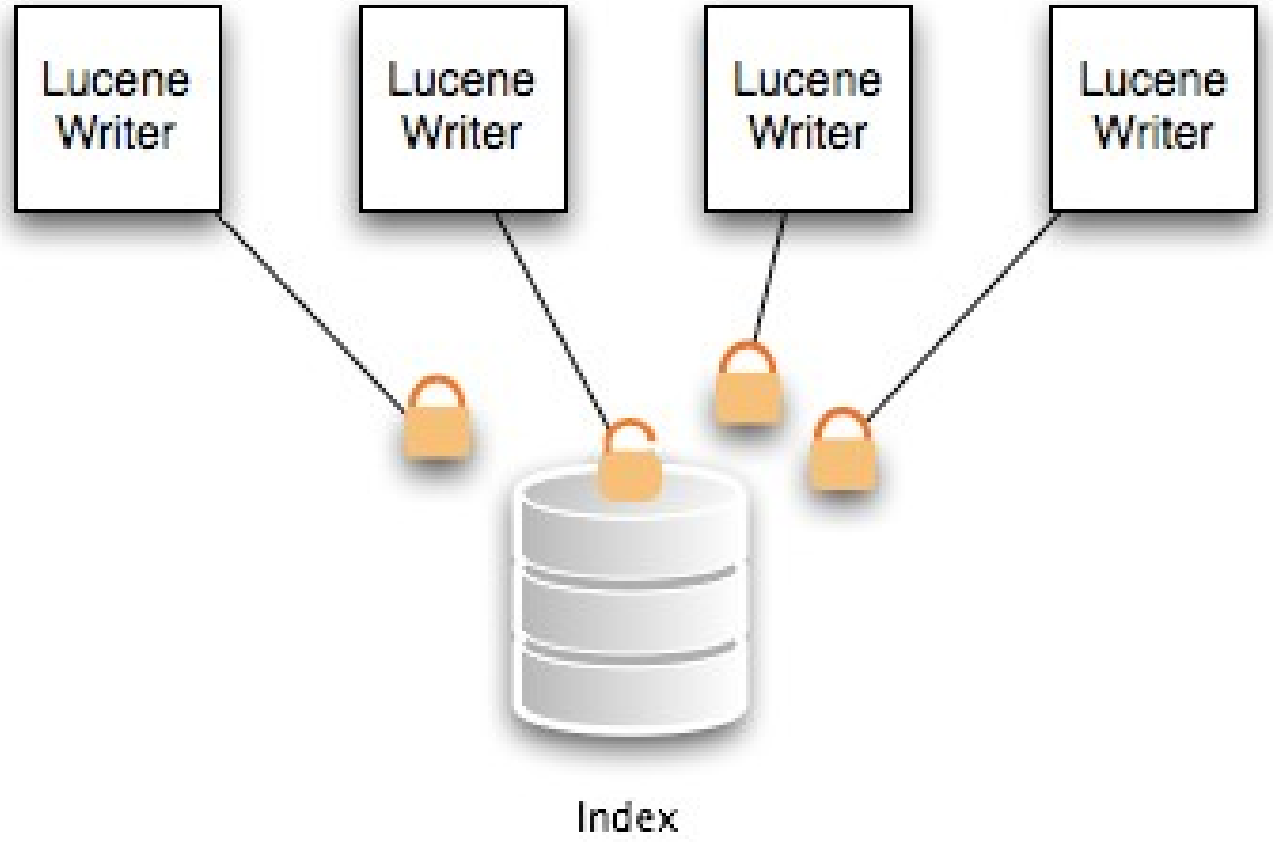


multi-node *performance idea*





Why does writing not scale?



Performance hints

- Setting Lucene's maximum segment size to fit in LuceneDirectory *chunk_size* will avoid readlocks
- Verify blob sizes fit in JGroups network packets, tune JGroups
- Check for CacheStores “sweet spot” size

Memory requirements

- RAMDirectory: all must fit in a single VM's memory
- FSDirectory: OS does a great caching job – but if it doesn't fit in memory
- Infinispan: comparable to FSDirectory
  - Flexible
  - Fast
  - Network vs. disk

Ingredients for a cloud

One Infinispan to rule them all

- Store Lucene indexes
- Hibernate second level cache
- Application managed cache
- Datagrid
- EJB, session replication in AS7
- As a JPA “store” via Hibernate OGM

Ingredients for a cloud

- JGroups discovery protocol
  - MPING
  - TCP_PING
  - JDBC_PING
  - S3_PING
- Choose a CacheLoader
  - Database based
  - Jclouds (S3, ...)
  - Cassandra

jclouds™

What's next

- Facilitate writing scalability
- Ease configuration aspects for clustering – ergonomics!
- Parallel searching
- A component of
  - http://www.cloudtm.eu

Related talks at JUDCon

15:15 – JPA applications in the era of NoSQL and Clouds: **Introducing OGM**

Q&A

http://infinispan.org **@Infinispan**
http://in.relation.to **@Hibernate**
http://jboss.org **@SanneGrinovero**