

Hibernate Search: search engine for the masses



2007

GRAILS EXCHANGE

<http://www.grails-exchange.com> | <http://www.grails.org> | <http://skillsmatter.com>



Who the hell are you?

- § Emmanuel Bernard
- § Member of the Hibernate Team
- § Core developer at JBoss, a Division of Red Hat
- § e-commerce background



Search: left over of today's apps

Add a search dimension to your persistent domain model

“Frankly, search sucks on this project”

-- Anonymous' boss

Why? How to fix that? For cheap?



Integrate full-text search and persistent domain

SQL Search vs Full-text Search

Object model / Full-text Search mismatches

Demo

Hibernate Search architecture

Configuration and Mapping

Full-text based object queries

Some more features



SQL search limits

- § Wildcard / word search
 - § '%hibernate%'
- § Approximation (or synonym)
 - § 'hybernate'
- § Proximity
 - § 'Java' close to 'Persistence'
- § Relevance or (result scoring)
- § multi-"column" search



Full Text Search

§ Search information

- § by word

- § inverted indices (word frequency, position)

§ In RDBMS engines

- § portability (proprietary add-on on top of SQL)

- § flexibility

§ Standalone engine

- § Apache Lucene(tm) <http://lucene.apache.org>



Mismatches with a domain model

- § Structural mismatch
 - § full text index are text only
 - § no reference/association between document
- § Synchronization mismatch
 - § keeping index and database up to date
- § Retrieval mismatch
 - § the index does not store objects
 - § certainly not managed objects

DEMO



2007

GRAILS

EXCHANGE

<http://www.grails-exchange.com> | <http://www.grails.org> | <http://skillsmatter.com>



Hibernate Search

- § Under the Hibernate platform
 - § LGPL
- § Built on top of Hibernate Core
- § Use Apache Lucene(tm) under the hood
 - § In top 10 downloaded at Apache
 - § Very powerful
 - § Somewhat low level
 - § easy to use it the “wrong” way
- § Solve the mismatches



Architecture

- § Transparent indexing through event system (JPA)
 - § PERSIST / UPDATE / DELETE
- § Convert the object structure into Index structure
- § Operation batching per transaction
 - § better Lucene performance
 - § "ACID"-ity
 - § (pluggable scope)
- § Backend
 - § synchronous / asynchronous mode
 - § Lucene, JMS



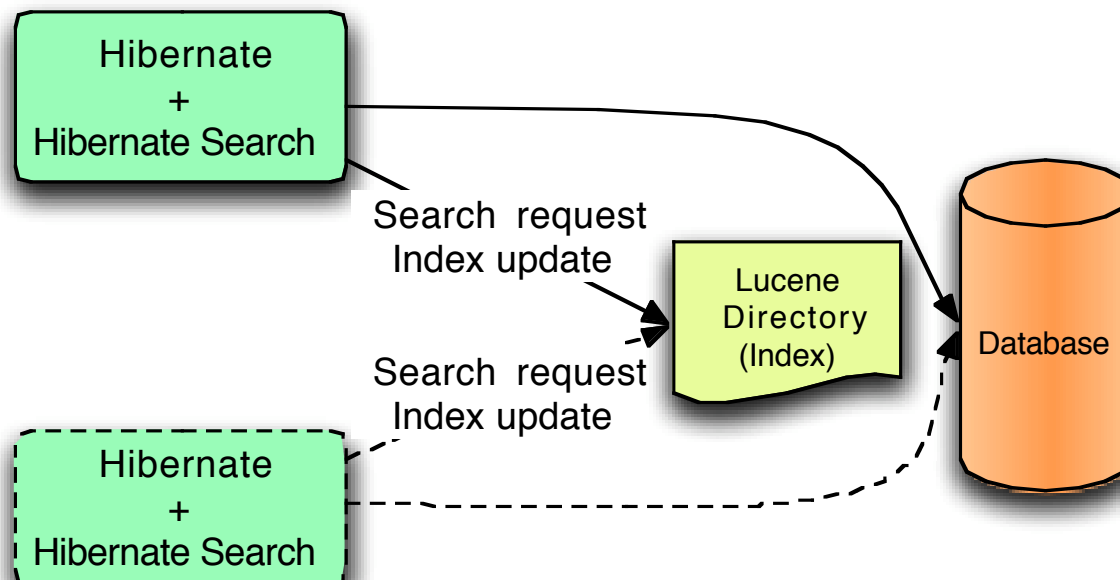
Architecture (Backend)

§ Lucene Directory

§ standalone or symmetric cluster

§ immediate visibility

§ Can affect front end runtime

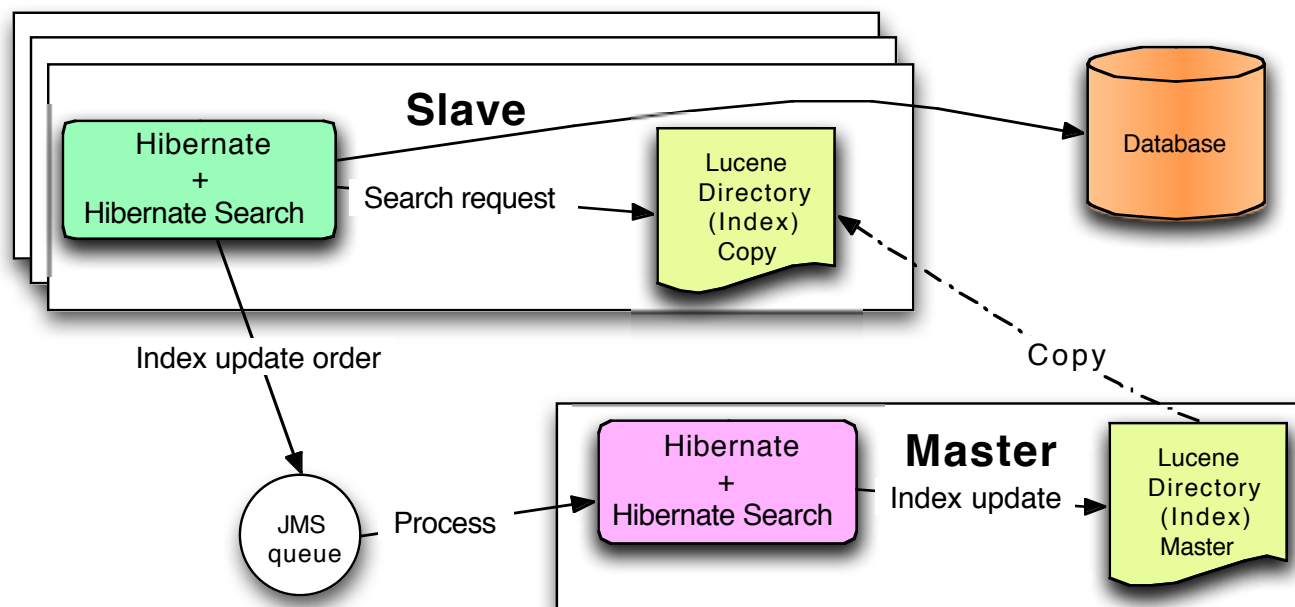




Architecture (Backend)

§ JMS (Cluster)

- § Search processed locally
- § Changes sent to a master node (JMS)
- § asynchronous indexing (delay)
- § No front end extra cost





Configuration and Mapping

§ Configuration

§ event listener wiring

- transparent in Hibernate Annotations

§ Backend configuration

§ Mapping: annotation based

§ @Indexed

§ @Field(store, index)

§ @IndexedEmbedded

§ @FieldBridge

§ @Boost / @Analyzer



Mapping example

```
@Entity @Indexed(index="indexes/essays")
public class Essay {
    ...

    @Id @DocumentId
    public Long getId() { return id; }

    @Field(name="Abstract", index=Index.TOKENIZED,
           store=Store.YES)
    public String getSummary() { return summary; }

    @Lob @Field(index=Index.TOKENIZED)
    public String getText() { return text; }

    @ManyToOne @IndexedEmbedded
    public Author getAuthor() { return author; }
}
```



Query

- § Retrieve objects, not documents
 - § no boilerplate conversion code!
- § Objects from the Persistence Context
 - § same semantic as a JPA-QL or Criteria query
- § Use `org.hibernate.Query/javax.persistence.Query`
 - § common API for all your queries
 - § pagination support
 - § list / scroll / iterate support
- § Query on correlated objects
 - § "JOIN"-like query



Query example

```
org.apache.lucene.search.Query luceneQuery;  
String queryString = "summary:Festina Or brand:Seiko"  
luceneQuery = parser.parse( queryString );
```

```
org.hibernate.Query fullTextQuery =  
fullTextSession.createFullTextQuery( luceneQuery );
```

```
fullTextQuery.setMaxResult(200);
```

```
List result = fullTextQuery.list();  
//return a list of managed objects
```

```
queryString = "title:hybernate~" //Approximate search  
queryString = "\"Hibernate JBoss\"~10" //Proximity search  
queryString = "author.address.city:Atlanta"  
//correlated search
```



More query features

- § Fine grained fetching strategy
- § Sort by property rather than relevance
- § Projection (both field and metadata)
 - § metadata: SCORE, ID, BOOST etc
 - § no DB / Persistence Context access
- § Filters
 - § security / temporal data / category
- § Total number of results



Some more features

- § Automatic index optimization
- § Index sharding
- § Manual indexing and purging
 - § non event-based system
- § Shared Lucene resources
- § Native Lucene access



Full-Text search without the hassle

- § Transparent index synchronization
- § Automatic Structural conversion through Mapping
- § No paradigm shift when retrieving data
- § Clustering capability out of the box
- § Easier / transparent optimized Lucene use

Q&A



2007

GRAILS EXCHANGE

<http://www.grails-exchange.com> | <http://www.grails.org> | <http://skillsmatter.com>



For More Information

§ Hibernate Search

§ <http://search.hibernate.org>

§ Apache Lucene

§ <http://lucene.apache.org>

§ Lucene In Action - Manning

§ Java Persistence with Hibernate - Manning

§ <http://in.relation.to>

§ <http://blog.emmanuelbernard.com>