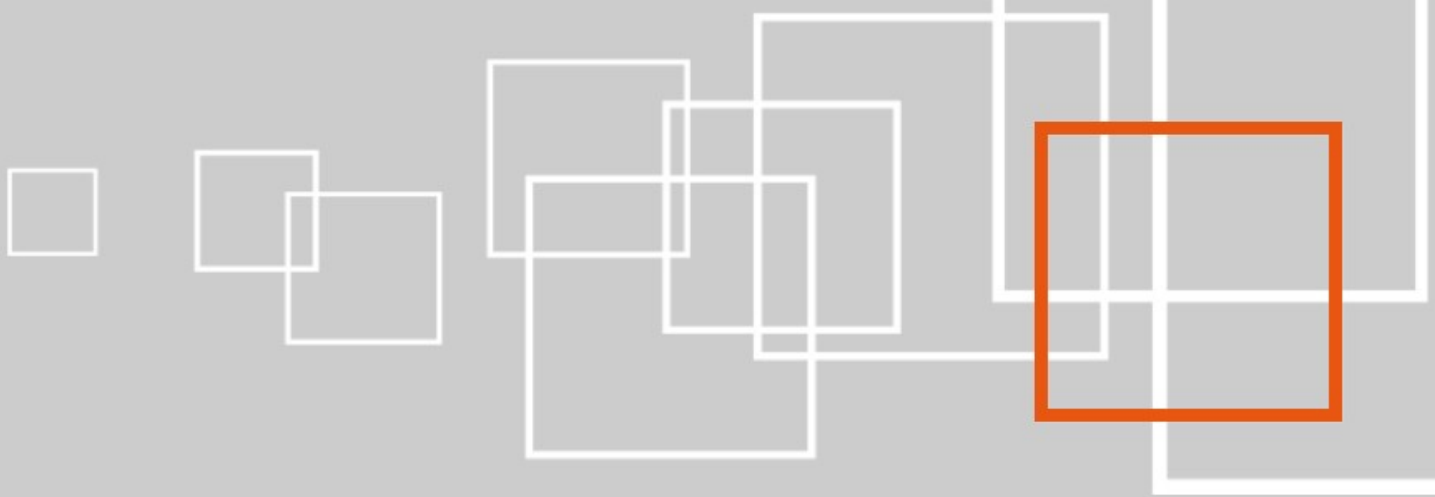


On eZ Find

Paul Borgermans

eZ®
THE CONTENT MANAGEMENT ECOSYSTEM

Outline

- A short status update of what is coming in Solr & eZ Find
- Use eZ Find beyond search: an alternate content engine & platform
- Archiving
- Some tips & tricks

BARCAMP

- Q & A unlimited

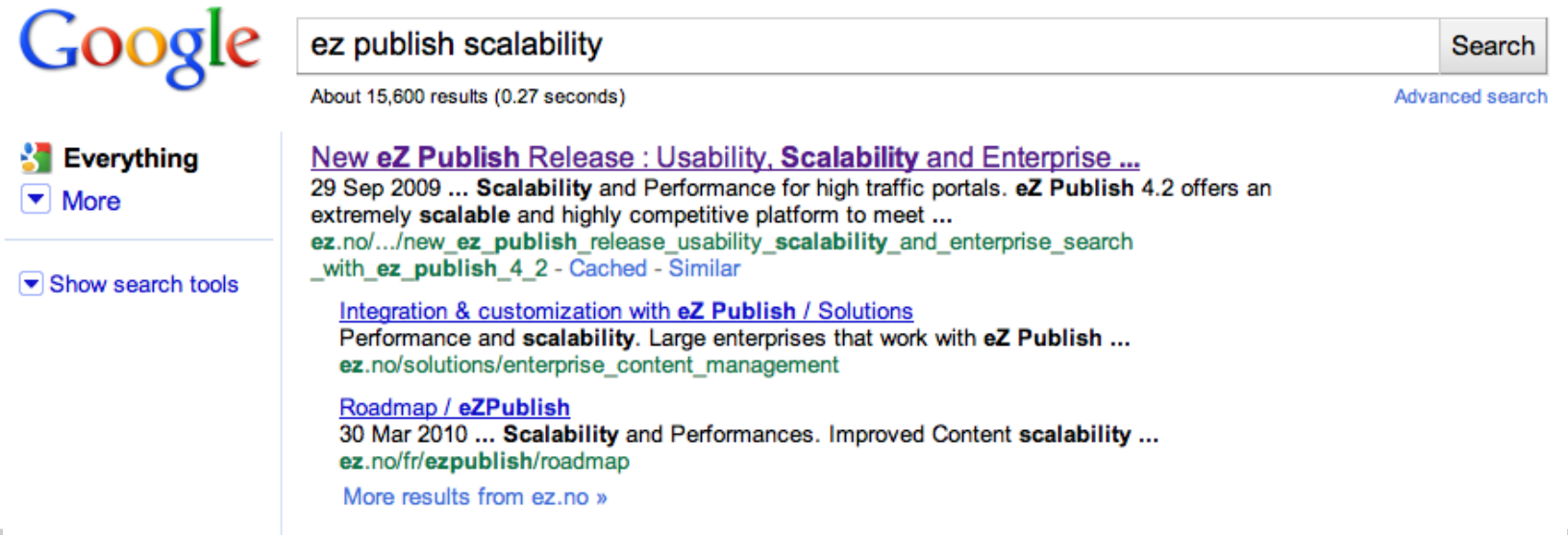
Solr & eZ Find outlook

End User related

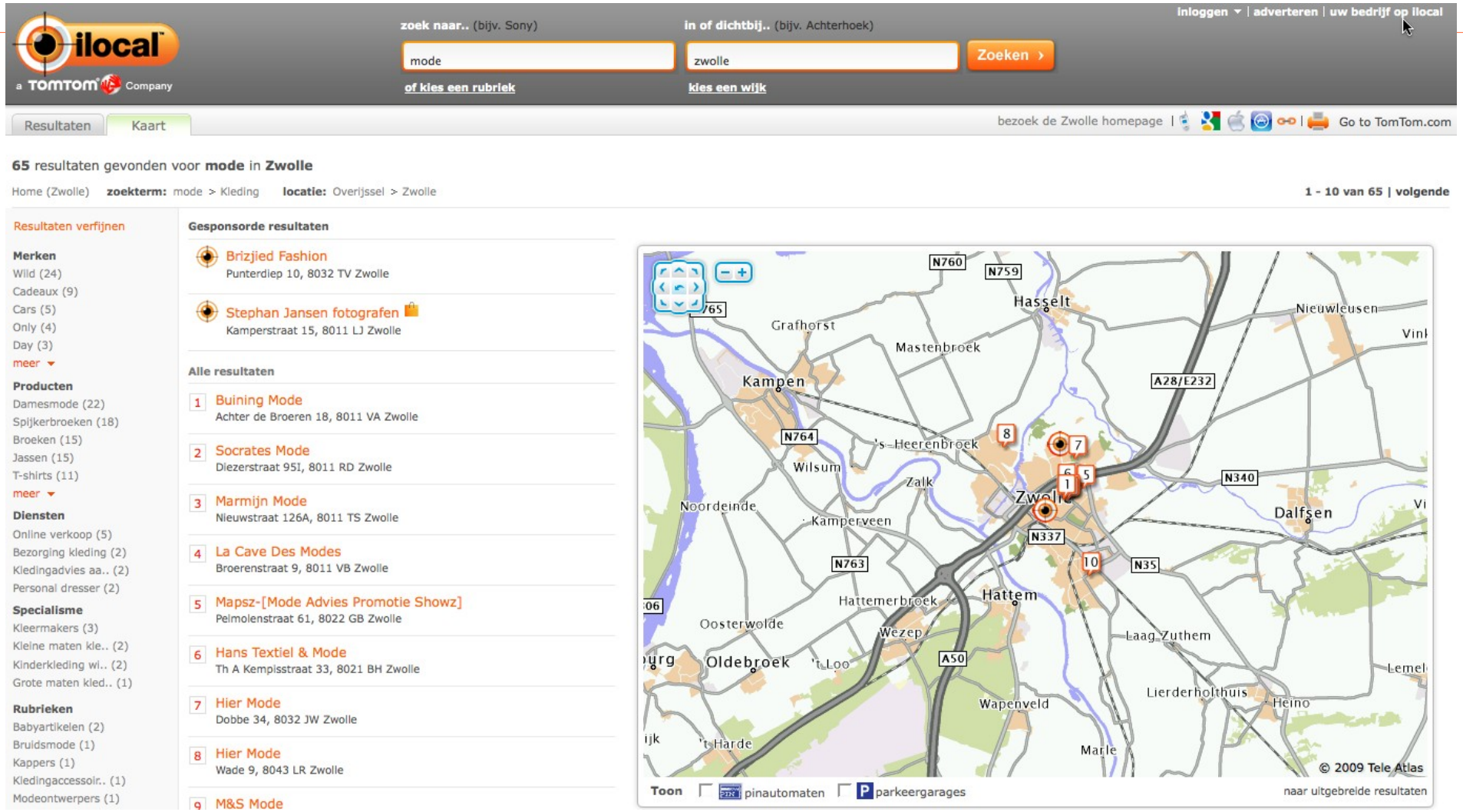
- More power in queries / searching
  - Better heuristics
  - Pure negative queries
  - Enhanced stopword handling

Field collapsing

- Combine results based on a common value
  - Category
  - Subtree
  - ...



Geospatial search

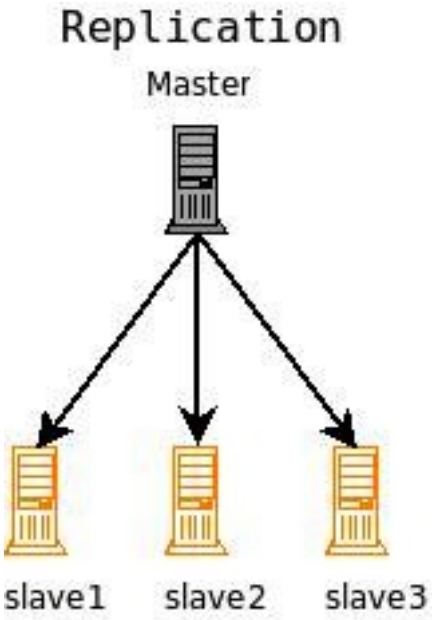


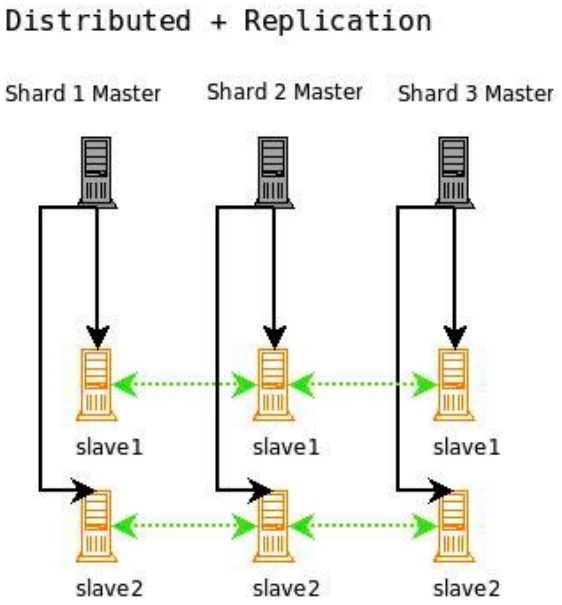
Geospatial search

- Half-way support in eZ Find 2.2 (with some schema manipulation)
  - See talks by Gilles and Ivo later today
- eZ Find 2.3
  - Leverage stable spatial features of Solr trunk
  - Dedicated fetch parameters for filtering, boosting, sorting

Almost in the cloud …

- 1 Solr node = 5-20M objects
- Clustering possibilities (also HA) => 5 x 10^9





Schema's from the presentation Solr in the Cloud, Mark Miller, Apache Lucene Eurocon, Prague, 2010

Solr in the cloud

- SolrCloud main features
  - Shared/Central Config
  - High Availability and Fault Tolerance
  - Cluster Resizing and Rebalancing
  - Cluster Status Support

- Based on Apache Zookeeper

Solr in the cloud (...)

- No need for external load balancers
  - Transparent for clients (eZ Find)
- Near real-time search in a highly scalable environment (milliseconds scale)
- High Availability for Writes
- Elastic (with vertical clustering)
  - Rebalance automatically
- Solr will handle index/document partitioning itself

Using eZ Find/Solr beyond search:

an alternate content engine

eZ Find with two main roles

- eZ Find/Solr as IR engine/layer
  - Without clustering, it can scale easily to several 10's of millions objects with current hardware
  - With clustering/sharding supports billions of objects
- Solr as a content engine
  - Document oriented storage system (hello NoSQL)
  - Archive use-case is also proof of concept

Provisions in eZ Find

- Multi-core setup
- Attribute storage
- Distributed search in `fetch(ezfind, search)`
  - Query parameters
  - Filter parameters
  - Fields to return (for rendering)

Use cases to consider

- Use eZ Find as a “caching” storage layer
  - Remove the burden on your DB
  - Significant speedups also for regular content
- Archiving
  - Offloading yet still serving massive amounts of seldom used objects for performance
  - Typically content that won't be edited anymore

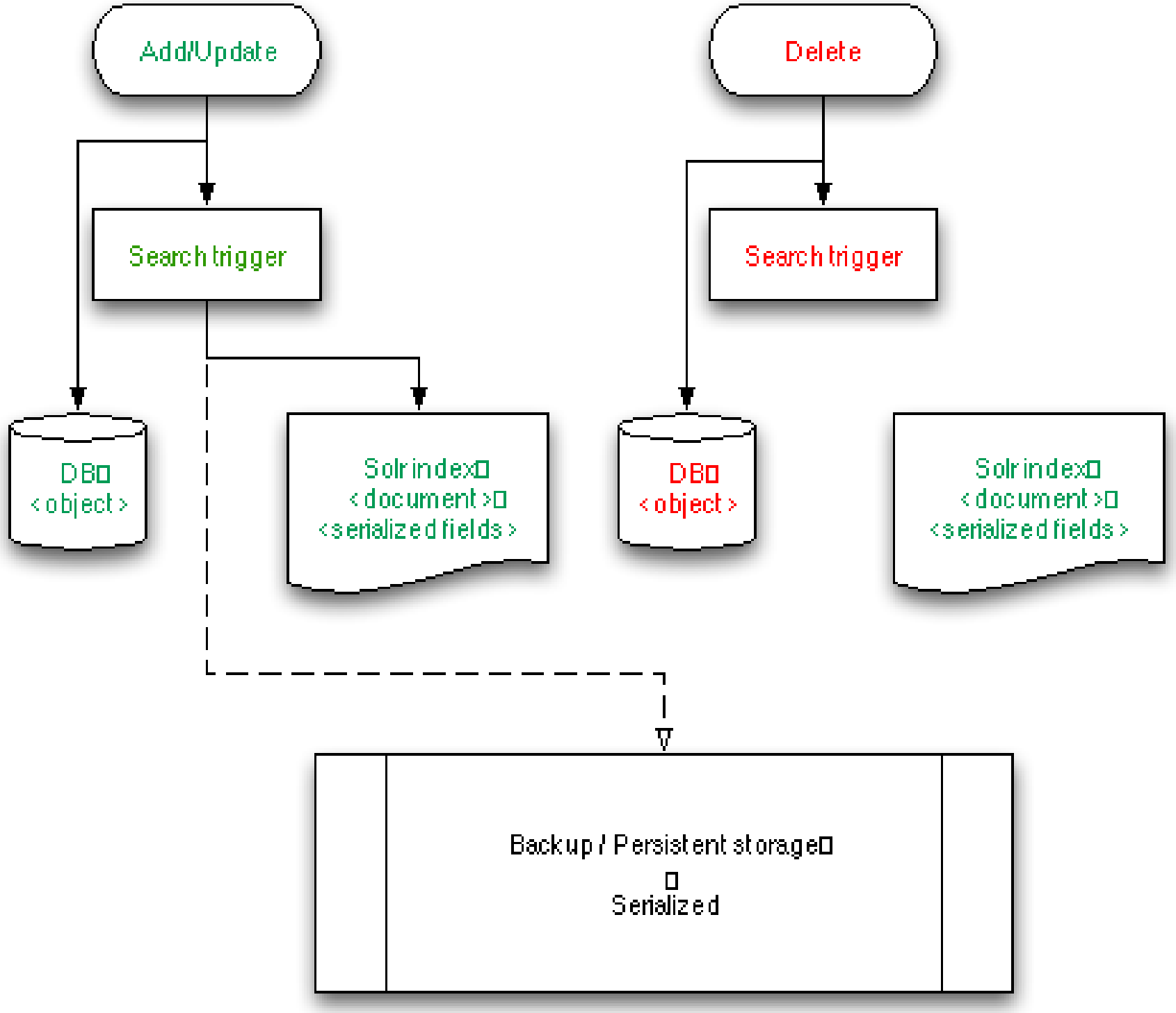
Use cases to consider (...)

- Integrating various information repositories
  - Lucene connector framework (Sharepoint, Livelink, Documentum, …) → Solr
  - ERP systems
  - Other eZ Publish sites
  - …
- “Importing” large amounts of data
  - External news article feeds
  - Directly into Solr

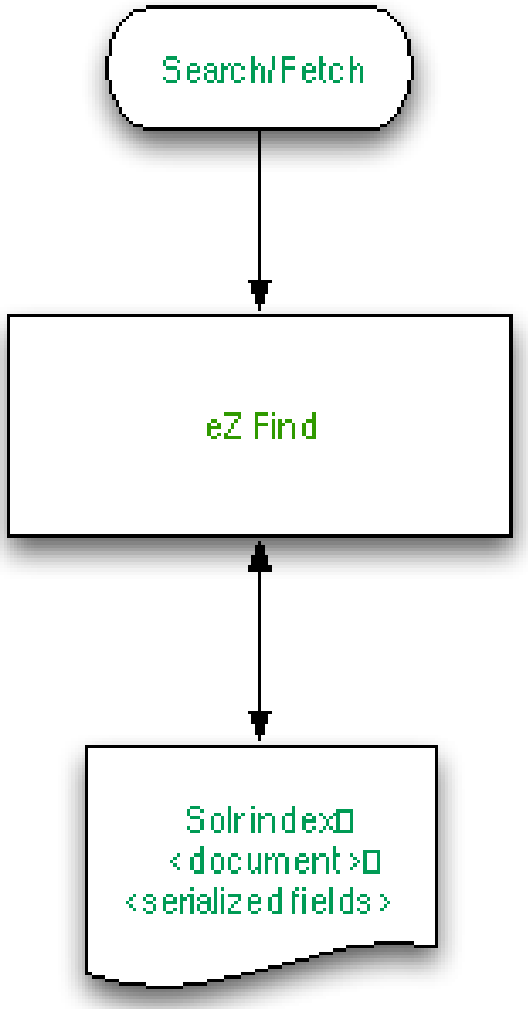
Archiving

eZ “Archive” system: initial version

- Inserts/updates into the archive system are done through the search engine plugin hooks present in the kernel
  - normal deletes won't be honored
  - manipulation of the archive system in a dedicated module
- The data are fetched from Solr and delivered in a similar structure as eZ Publish objects


Add/Update
Delete
Search trigger
Search trigger
DB
<object>
Solr index
<document>
<serialized fields>
DB
<object>
Solr index
<document>
<serialized fields>
Backup / Persistent storage
Serialized

Searching/fetching returns data directly from Solr



More archiving plans

- Dedicated module
  - Administration of archive itself
  - Statistics
- Mark objects for archiving based on several criteria
  - Subtrees
  - Age
  - ...

eZ “Archive” system: more storage options

- Persistent store options (“third line”)
  - Document oriented stores: MongoDB/CouchDB
  - Key-value/bigtable stores: Cassandra, …
- In an extended ezfind layer, a hook will be introduced for syncing with the 3rd line storage

Tips and tricks

Use the multi-core features!

- Multilingual sites
  - Better relevance
  - Language specific tuning
- A core per data-source
- Performance! (it is partitioning)

Optimize facets/filtering/sorting

- [SolrFieldMapSettings]
  - Searching (default)
  - Facets
  - Filtering
  - Sorting
- Solr has many analyzers/filters to use at index time!
  - Key to get the results you want

Replace extended attribute filters ...

- … with eZ Find filter constructs
- If needed, adapt the Solr schema
  - Copy fields
  - You can do lots of manipulation using Solr analyzers

Enable attribute storage

- eZ Find trunk/2.3
- Decrease the DB queries with 2 orders of magnitude (about 5x faster rendering)
- Can do lazy loading as well (specify only the attributes you need)
  - Memory
  - Bandwidth

Thank you!