netcetera

Apache Solr

An experience report

2013-10-23 - Corsin Decurtins


Apache
Solr

Apache Solr

Notes

Full-Text Search Engine
Apache Lucene Project
based on Apache Lucene

Fast
Proven and Well-Known Technology
Java based
Open APIs
Customizable
Clustering Features

Apache Solr: http://lucene.apache.org/solr/

Setting the **Scene**

Search Terms

Search Tipps

- Everything
- Wiki
- Files
- Emails
- Issues
- People
- Companies
- Projects
- Resources
- Books & Magazines

Include Archive

Project identifier or name

Name or shortname of an author

All Types

Plaza Search

Plaza Search is an integrated search engine for the Netcetera intranet. You can use Plaza Search to look for resources in the Plaza Wiki, our file system, JIRA, Mailstore and in Infostore (people, projects, companies and books and magazines).

Just put your search terms into the corresponding field at the top and use the filters on the left side to further narrow down the results.

If you want to find out what Plaza Search can do for you, make sure to check out the Plaza Search User Documentation.

And if something should not work as you expect it or you have an idea about how to make Plaza Search even better, make sure to let us know - either by creating a JIRA issue or by sending us an email.

Plaza Search

Notes

Full-Text Search Engine for the Intranet of Netcetera

Integrates Various Data Sources

Needs to be fast

Ranking is crucial

Simple Searching

Relevant Filtering Options

Desktop, Tables and Phones

netceteca

Warum Intranet-Suchmaschinen unbrauchbar sind

...und was dagegen getan werden kann

2013-07-03 – Corsin Decurtins

http://www.slideshare.net/netceteragroup/20130703-intranet-searchintranetkonferenz

Search Terms

Search Tipps

- Everything
- Wiki
- Files
- Emails
- Issues
- People
- Companies
- Projects
- Resources
- Books & Magazines

Include Archive

Project identifier or name

Name or shortname of an author

All Types

Plaza Search

Plaza Search is an integrated search engine for the Netcetera intranet. You can use Plaza Search to look for resources in the Plaza Wiki, our file system, JIRA, Mailstore and in Infostore (people, projects, companies and books and magazines).

Just put your search terms into the corresponding field at the top and use the filters on the left side to further narrow down the results.

If you want to find out what Plaza Search can do for you, make sure to check out the Plaza Search User Documentation.

And if something should not work as you expect it or you have an idea about how to make Plaza Search even better, make sure to let us know - either by creating a JIRA issue or by sending us an email.

Some Numbers

Live since
05/2012

Data since
1996

~ **275** Users

~ **500 – 2'000**
Searches per day

~ **3'000'000**
Documents

~ **40** Releases

Index Size
~ **75** GByte

Some Numbers

Notes

Very small load (only a few hundred requests per day)

The indexer agents actually produce a lot more load than the actual end users

Medium size index (at least I think)

Not that many objects, but relatively big documents

Load performance is not a big topic for us

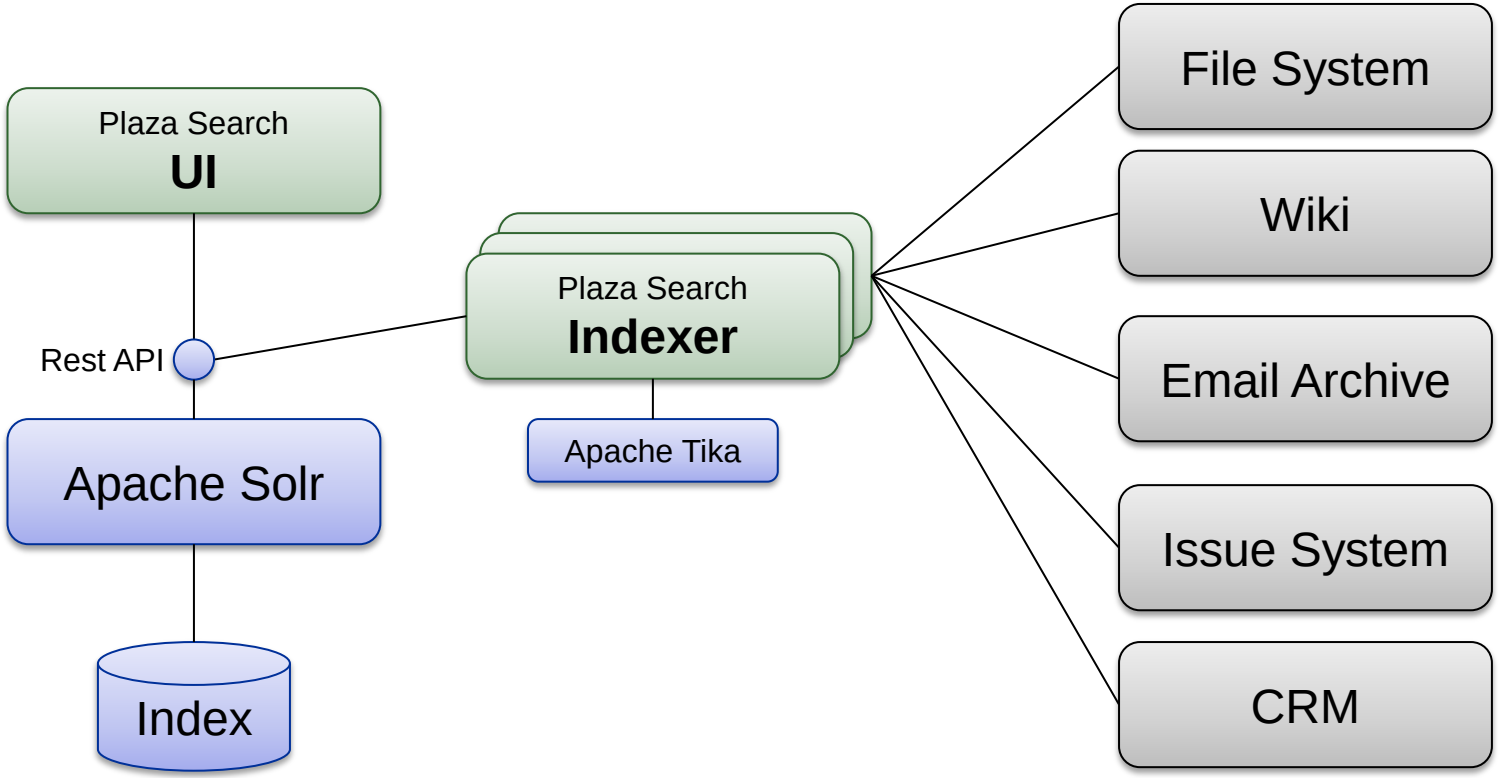
When we talk about performance, we actually usually mean response time

For us

Performance

means

Response Time


Plaza Search
UI
Rest API
Apache Solr
Index
Plaza Search
Indexer
Apache Tika
File System
Wiki
Email Archive
Issue System
CRM

Architecture

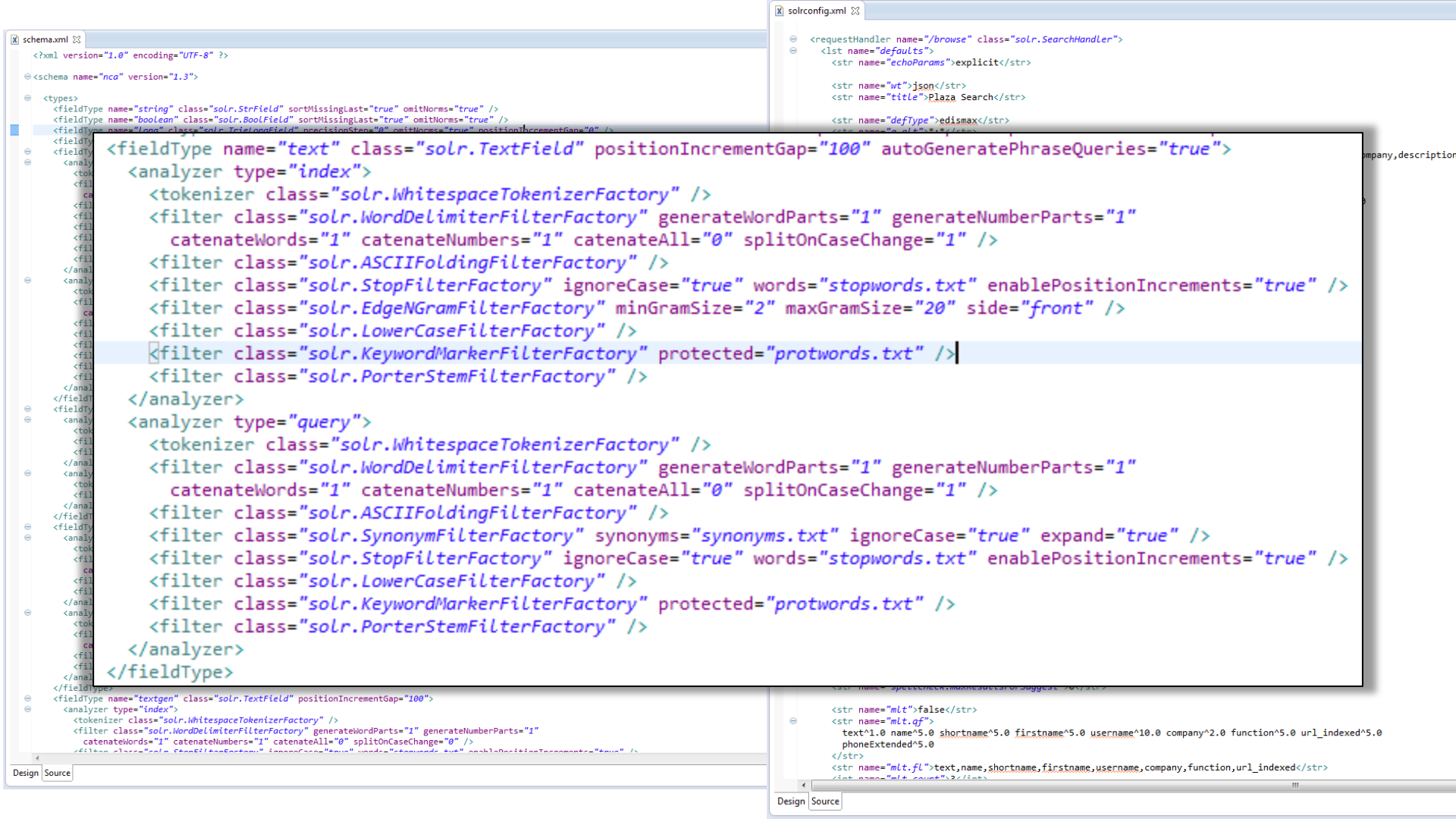
Notes

Based on Apache Solr (and other components)
Apache Solr takes care of the text-search aspect
We certainly do not want to build this ourselves

Apache Tika is used for analyzing (file) contents
Also here, we certainly do not want to build this ourselves


Apache
Solr

Magic


schema.xml
<?xml version="1.0" encoding="UTF-8" ?>
<schema name="nca" version="1.3">
<types>
<fieldType name="string" class="solr.StrField" sortMissingLast="true" omitNorms="true" />
<fieldType name="boolean" class="solr.BoolField" sortMissingLast="true" omitNorms="true" />
<fieldType name="text" class="solr.TextField" positionIncrementGap="100" autoGeneratePhraseQueries="true">
<analyzer type="index">
<tokenizer class="solr.WhitespaceTokenizerFactory" />
<filter class="solr.WordDelimiterFilterFactory" generateWordParts="1" generateNumberParts="1"
catenateWords="1" catenateNumbers="1" catenateAll="0" splitOnCaseChange="1" />
<filter class="solr.ASCIIFoldingFilterFactory" />
<filter class="solr.StopFilterFactory" ignoreCase="true" words="stopwords.txt" enablePositionIncrements="true" />
<filter class="solr.EdgeNGramFilterFactory" minGramSize="2" maxGramSize="20" side="front" />
<filter class="solr.LowerCaseFilterFactory" />
<filter class="solr.KeywordMarkerFilterFactory" protected="protwords.txt" />
<filter class="solr.PorterStemFilterFactory" />
</analyzer>
<analyzer type="query">
<tokenizer class="solr.WhitespaceTokenizerFactory" />
<filter class="solr.WordDelimiterFilterFactory" generateWordParts="1" generateNumberParts="1"
catenateWords="1" catenateNumbers="1" catenateAll="0" splitOnCaseChange="1" />
<filter class="solr.ASCIIFoldingFilterFactory" />
<filter class="solr.SynonymFilterFactory" synonyms="synonyms.txt" ignoreCase="true" expand="true" />
<filter class="solr.StopFilterFactory" ignoreCase="true" words="stopwords.txt" enablePositionIncrements="true" />
<filter class="solr.LowerCaseFilterFactory" />
<filter class="solr.KeywordMarkerFilterFactory" protected="protwords.txt" />
<filter class="solr.PorterStemFilterFactory" />
</analyzer>
</fieldType>
<fieldType name="textgen" class="solr.TextField" positionIncrementGap="100">
<analyzer type="index">
<tokenizer class="solr.WhitespaceTokenizerFactory" />
<filter class="solr.WordDelimiterFilterFactory" generateWordParts="1" generateNumberParts="1"
catenateWords="1" catenateNumbers="1" catenateAll="0" splitOnCaseChange="0" />
Design
Source
solrconfig.xml
<requestHandler name="/browse" class="solr.SearchHandler">
<lst name="defaults">
<str name="echoParams">explicit</str>
<str name="wt">json</str>
<str name="title">Plaza Search</str>
<str name="defType">edismax</str>
mpany,description
<str name="mlt">false</str>
<str name="mlt.qf">
text^1.0 name^5.0 shortname^5.0 firstname^5.0 username^10.0 company^2.0 function^5.0 url_indexed^5.0
phoneExtended^5.0
</str>
<str name="mlt.fl">text,name,shortname,firstname,username,company,function,url_indexed</str>
Design
Source

Apache Solr

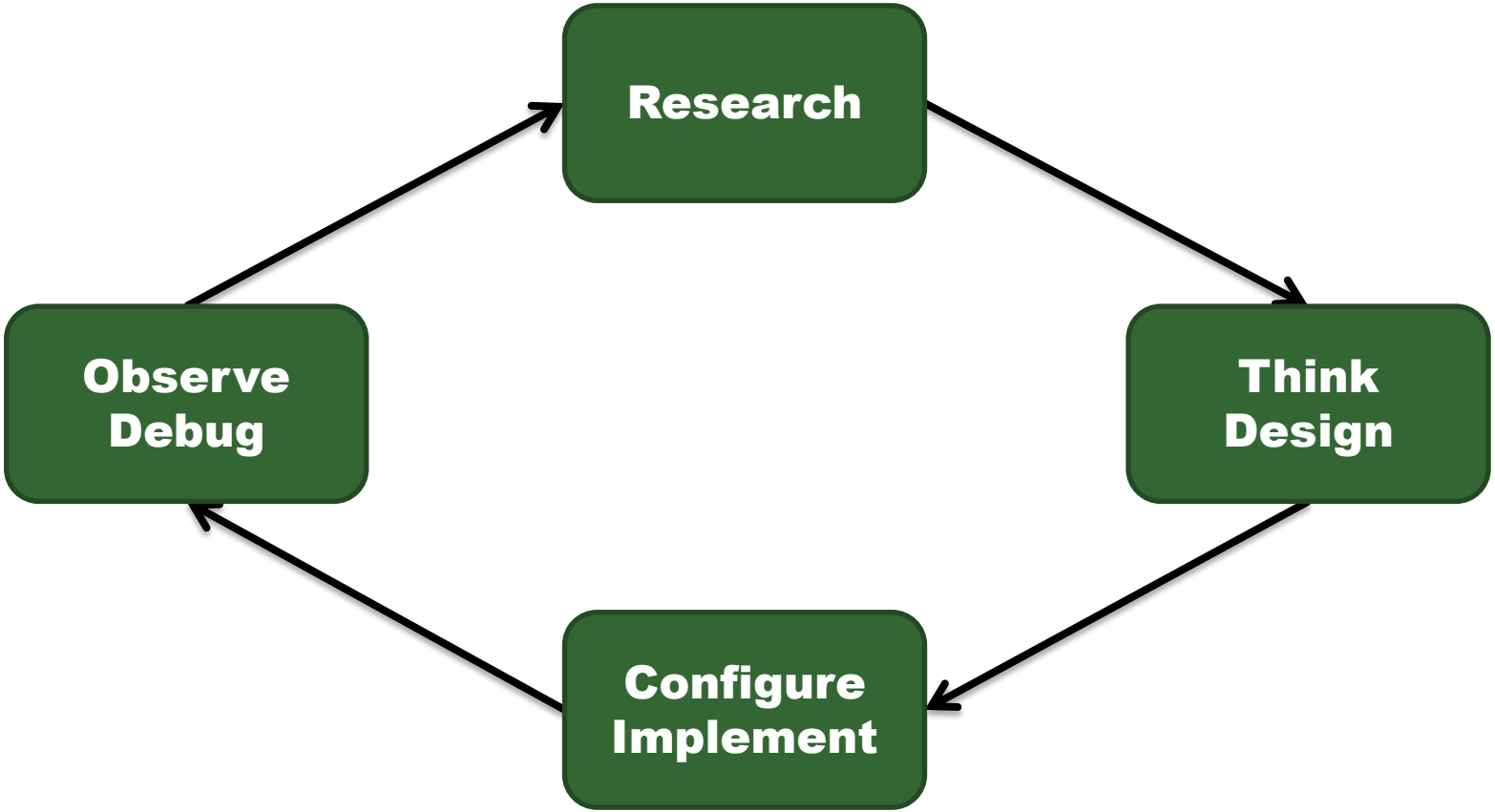
Notes

Apache Solr is a very complex system with a lot of knobs and dials
Most things just seem like magic at the beginning … or they just do not work

Apache Solr is extremely powerful with a lot of features
You have to know how to configure the features
Most features need a bit more configuration than just a check box for activating it

Configuration options seem very confusing at the beginning
You do not need to understand everything from the start
Defaults are relatively sensible and the example applications are good starting point

Development Process



Development Process

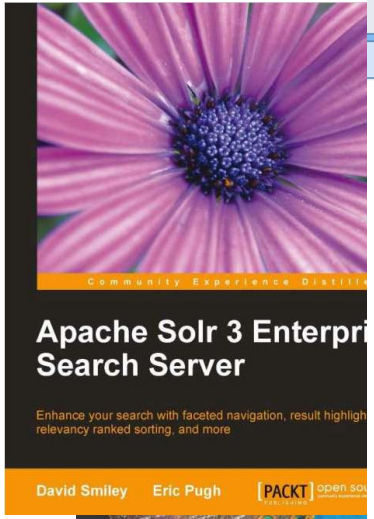
Notes

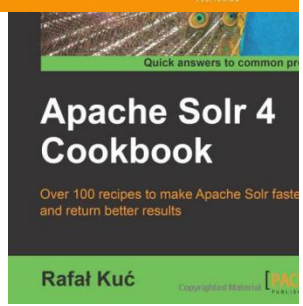
In our experience, Apache Solr works best with a very iterative process
Definition of Done is very difficult to specify for search use cases

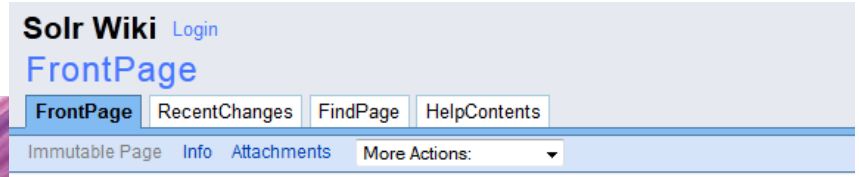
Iterate through:

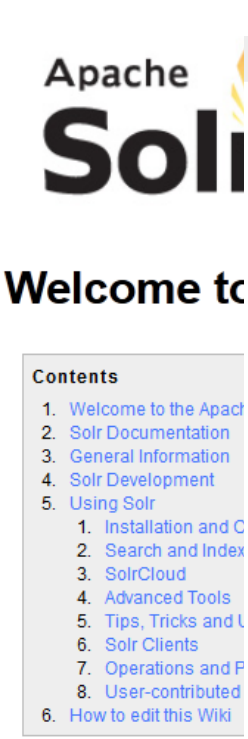
- Researching
- Thinking / Designing
- Implementation / Configuration / Testing
- Observing / Analyzing / Debugging

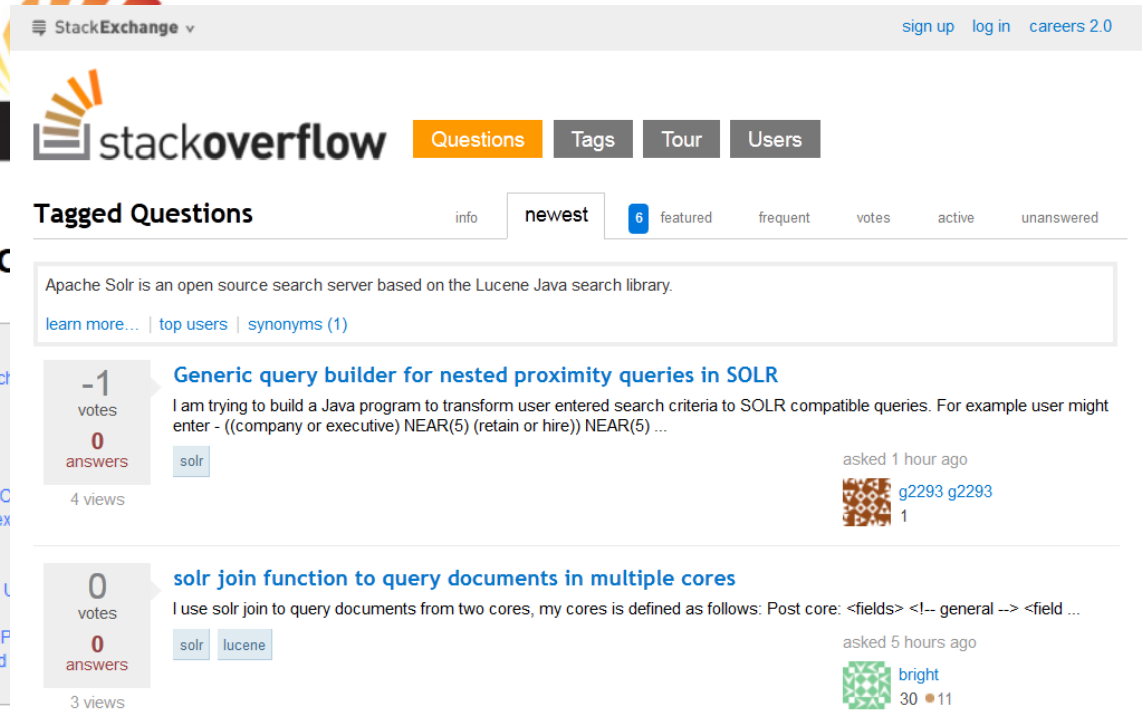
Research


Apache Solr 3 Enterpri
Search Server
Enhance your search with faceted navigation, result highligh
relevancy ranked sorting, and more
David Smiley
Eric Pugh
PACKT

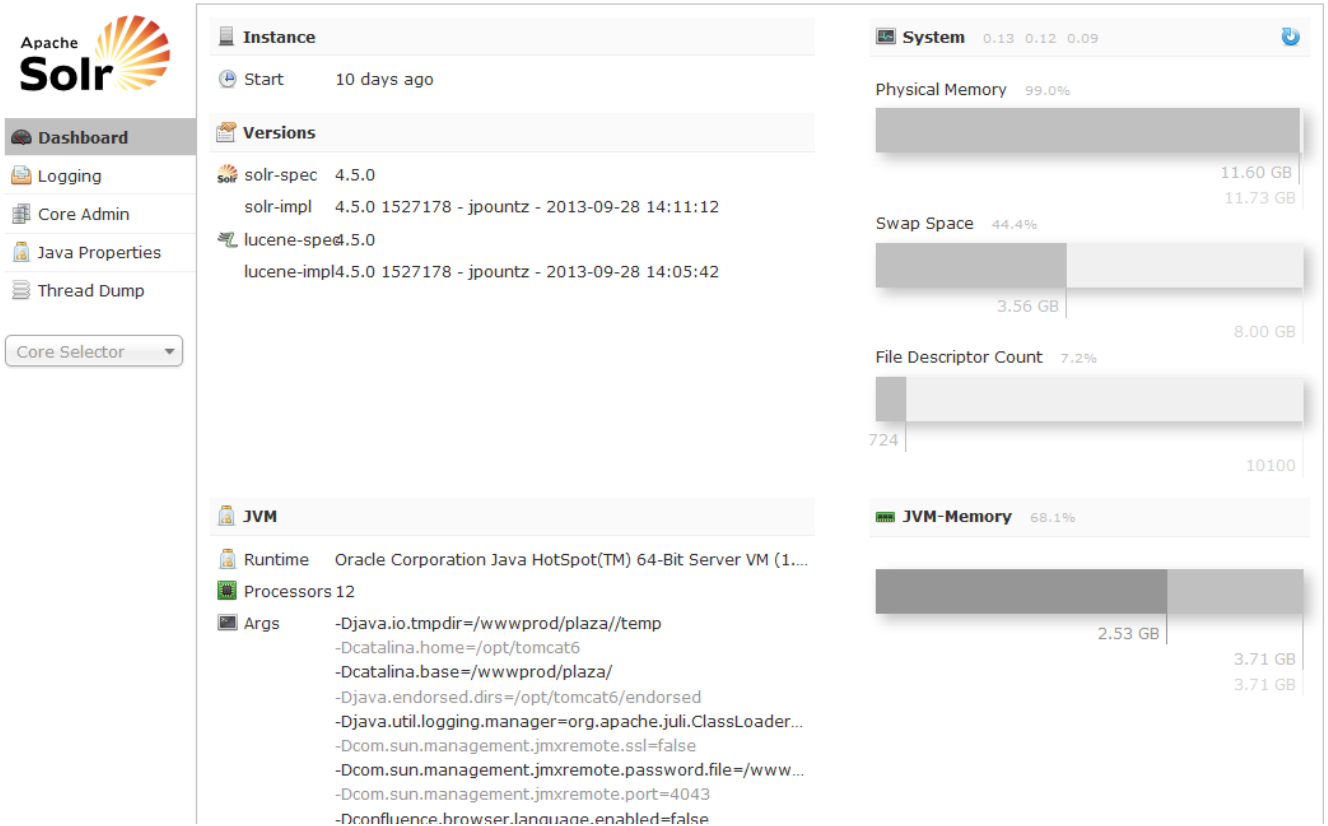

Apache Solr 4
Cookbook
Over 100 recipes to make Apache Solr faste
and return better results
Rafał Kuć


Solr Wiki
Login
FrontPage
FrontPage
RecentChanges
FindPage
HelpContents
Immutable Page
Info
Attachments
More Actions:


Apache
Soli
Welcome to
Contents
1. Welcome to the Apach
2. Solr Documentation
3. General Information
4. Solr Development
5. Using Solr
1. Installation and C
2. Search and Index
3. SolrCloud
4. Advanced Tools
5. Tips, Tricks and U
6. Solr Clients
7. Operations and P
8. User-contributed
6. How to edit this Wiki


StackExchange
sign up
log in
careers 2.0
stackoverflow
Questions
Tags
Tour
Users
Tagged Questions
info
newest
6
featured
frequent
votes
active
unanswered
Apache Solr is an open source search server based on the Lucene Java search library.
learn more... | top users | synonyms (1)
-1
votes
0
answers
4 views
Generic query builder for nested proximity queries in SOLR
I am trying to build a Java program to transform user entered search criteria to SOLR compatible queries. For example user might enter - ((company or executive) NEAR(5) (retain or hire)) NEAR(5) ...
solr
asked 1 hour ago
g2293 g2293
1
0
votes
0
answers
3 views
solr join function to query documents in multiple cores
I use solr join to query documents from two cores, my cores is defined as follows: Post core: <fields> <!-- general --> <field ...
solr
lucene
asked 5 hours ago
bright
30 11

Observe
Debug



Solr Admin Interface

Notes

Apache Solr has a pretty good admin interface

Very helpful for analysis and (manual) monitoring

If you are not familiar with the Solr Admin interface, you should be

Other tools like profilers, memory analyzers, monitoring tools etc. are also useful

Our Requirements

Correctness

Results that match query

Relevance

Results that matter

Speed

"Instant" results

Intelligence

Do you know what I mean?

Intelligence

Do you know what I mean?

synonyms.txt

stopwords.txt

protwords.txt

Solr Configuration Files

Notes

Solr has pretty much out-of-the-box support for stop words, protected works and synonyms

These features look very simple, but they are very powerful

Unless you have a very general search use case, the defaults are **not enough**

Definitely worth **developing a configuration specific to your domain**

Iterate; consider these features for **ranking optimizations** as well

Relevance

Results that matter

term frequency
inverse document frequency
field weights

boosting function
index time boosting
elevation

Ranking in Solr (simplified)

Notes

Solr determines a score for the results of a query

Score can be used for sorting the results

Score is the product of different factors:

A **query-specific part**, let's call it the **match** value that is computed using

- term frequency (tf)
- inverse document frequency (idf)

There are also other parameters that can influence it (term weights, field weights, …)

The match basically says how well a document matches the query

Ranking in Solr (simplified)

Notes

A **generic part** (not query specific), let's call this a **boosting** value

Basically represents the general importance that you assign to a document

Includes a ranking function, e.g. based on the age of the document

Includes a boosting value, that is determined at index time (index-time boosting)

We calculate the boost value based on different attributes of the document, such as

- type of resource (people are more important than files)
- status of the project that a document is associated with (closed projects are less important than running projects)
- archive flag (archived resources are less important)
- …

Ranking Function

recip(ms(NOW,datestamp),3.16e-11,1,1)

Index-Time Boosting

Regression
Ranking Testing

```
assertRank("jira", "url",
        "https://extranet.netcetera.biz/jira/", 1);
assertRank("jira", "url",
        "https://plaza.netcetera.com/.../themas/JIRA", 2);
```

Regression Testing for the Ranking

Notes

Ranking is influenced by various factors

We have continuously executed tests for the ranking

Find ranking regressions as soon as possible

Tests are executed every night, not just with code changes

War Stories

War Story #1:

Disk Space

Situation

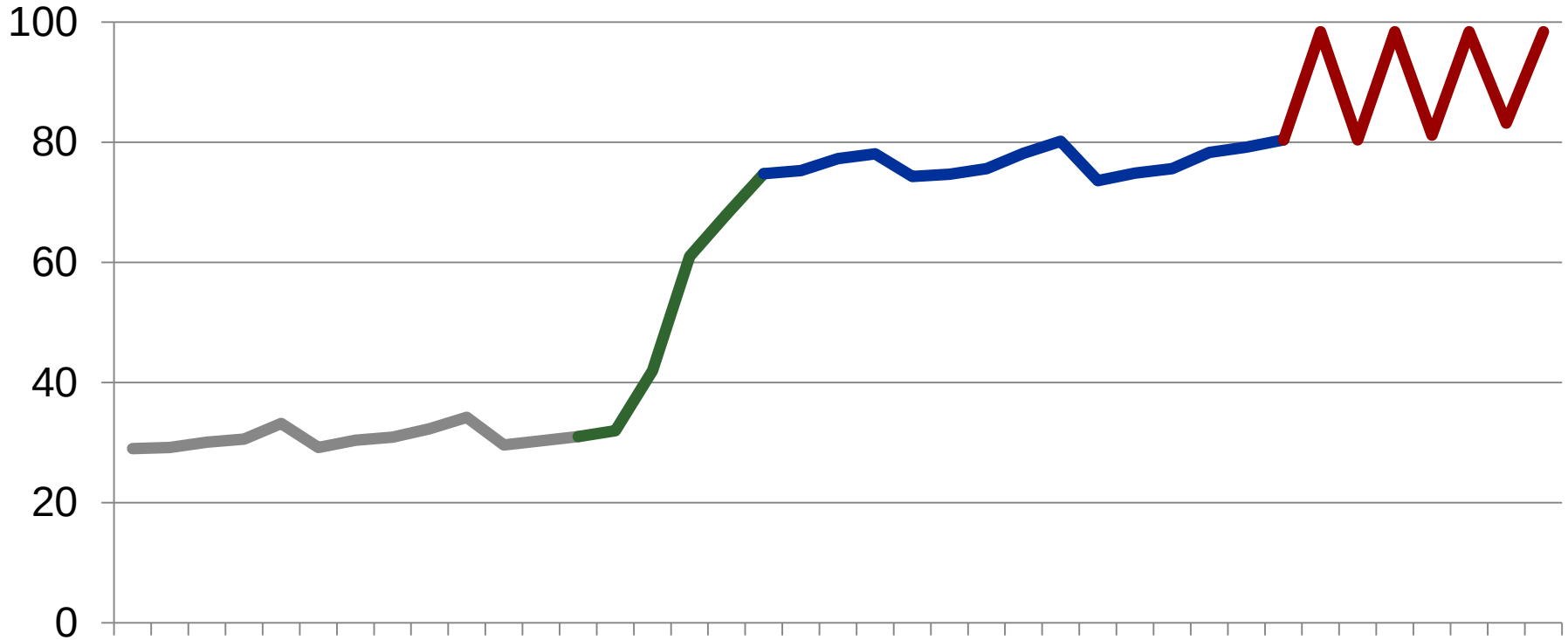
Notes

Search is often extremely slow, response times of 20-30s
Situation improves without any intervention
Problem shows up again very soon
Other applications in the same Tomcat server are brought to a grinding halt

No releases within the last 7 days
No significant data changes in the last 7 days

2-3 weeks earlier a new data sources have been added
Index had grown by a factor of 2, but everything worked fine since then

Disk Usage (fake diagram)



Lucene Index – Disk Usage

Notes

Index needs optimzation from time to time when you update it continuously

Index optimzation uses a lot of resources, i.e. CPU, memory and disk space

Optimzation requires twice the disk space than the optimal index

When your normal index uses > 50% of the available disk space, it's already too late

It's difficult to get out of this situation (without adding disk space)

Deleting stuff from the index does not help, as you need an optimization

Lessons Learned

We need least **2-3 times** as much space as the "ideal" index needs
If your index has grown **beyond 50%**, it's already **too late**.
Disk Usage Monitoring has to be improved

Some problems take a **long time to show themselves**
Testing long-term effects and **continuous delivery** clash to some extent

War Story #2:

Free Memory

Situation

Notes

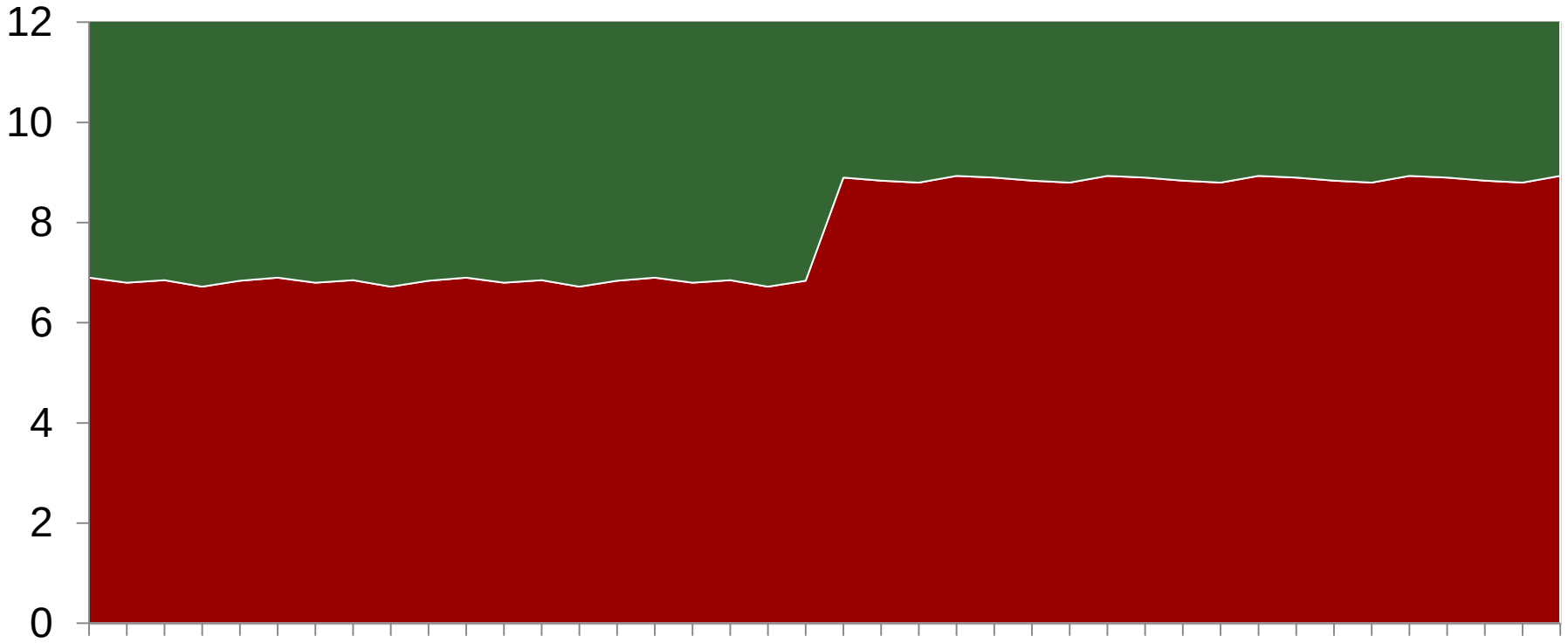
Search is always extremely slow, response times of 20-30s

Other applications in the same Tomcat server show normal performance

No releases within the last few days

No significant data changes in the last few days

Memory Usage (fake diagram)



I/O Caching

Notes

OS uses "free" memory for caching

I/O caching has a HUGE impact on I/O heavy applications

Solr (actually Lucene) is a I/O heavy application

Lessons Learned

Free memory != **unused** memory

Increasing the **heap size** can **slow down Solr**

OS does a better job at caching Solr data than Solr

War Story #3:

Know Your Maths

Situation

Notes

Search starts up very fine and is reasonably fast

Out Of Memory Errors after a couple of hours

Restart brings everything back to normal

Out Of Memory Errors come back after a certain time (no obvious pattern)

Analysis

Notes

Analysis of the memory usage using heap dumps
Solr Caches use up a lot of memory (not surprisingly)
Document cache with up to 2048 entries
Entries are dominated by the content field
Content is limited to 50 KByte by the indexers (or so I thought)
Content abbreviation had a bug
Instead of the 50KByte limit of the indexer, the 2 MByte limit of Solr was used

2048 * 2 MByte = 4GByte for the document cache
Heap size at that time = 4GByte

Lessons Learned

Heap dumps are your friends

Study your heap from time to time, even if you do not have a problem (yet)

Test your **limiters**

War Story #4:

Expensive Features

Situation

Notes

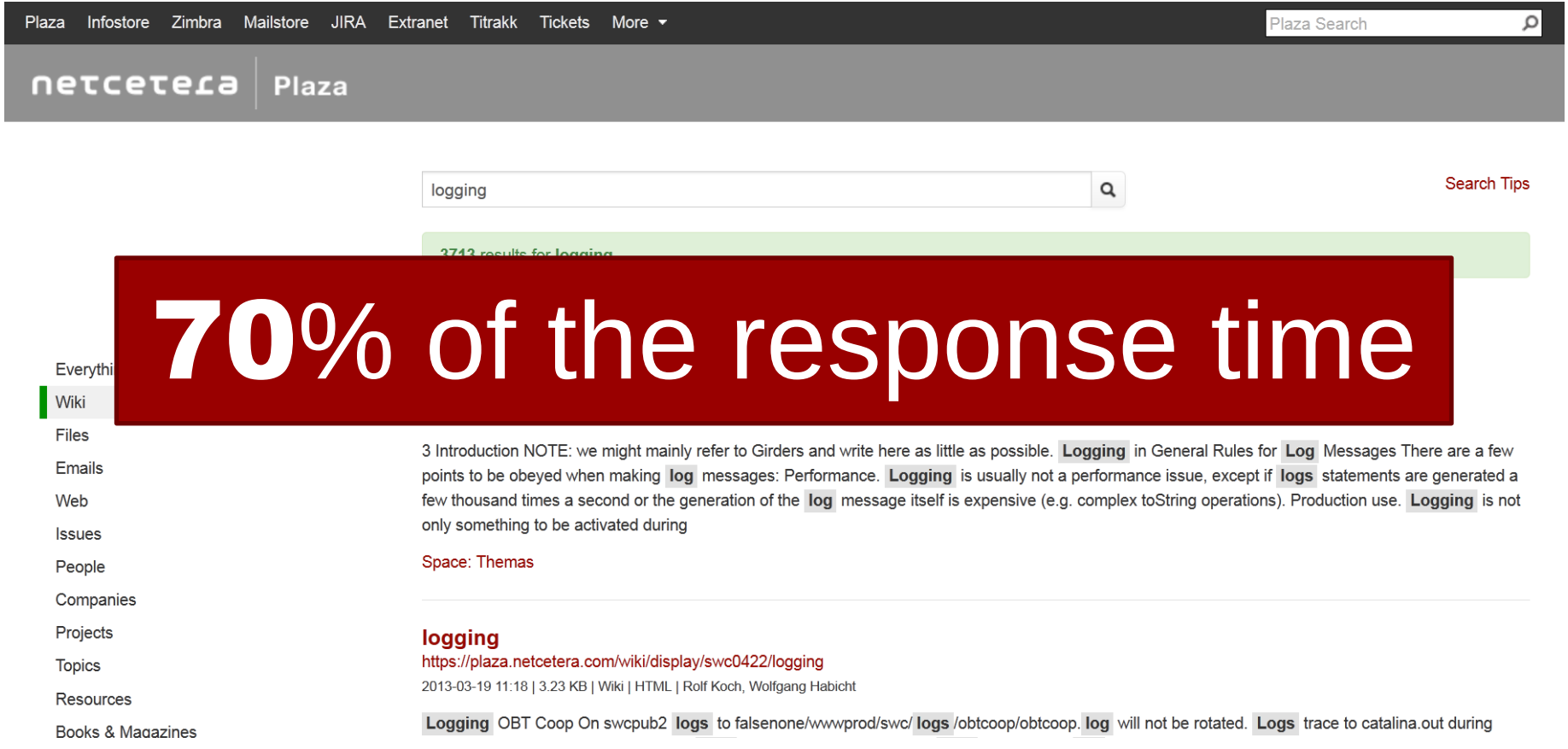
Search has become slower and slower

We added a lot of data, so that's not really surprising

Analysis into different tuning parameters

Analysis into the cost of different features

Highlighting



Lessons Learned

Some features are cool, but also **very expensive**

Think about what you need to **index** and what you need to **store**

Consider loading stuff **"offline"** and **asynchronously**

Consider loading stuff from **other data sources**

A few words on

Scaling

Solr Cloud – Horizontal and Vertical Scaling

Support for Replication and Sharding

Added with Apache Solr 4

Based on Apache Zookeeper

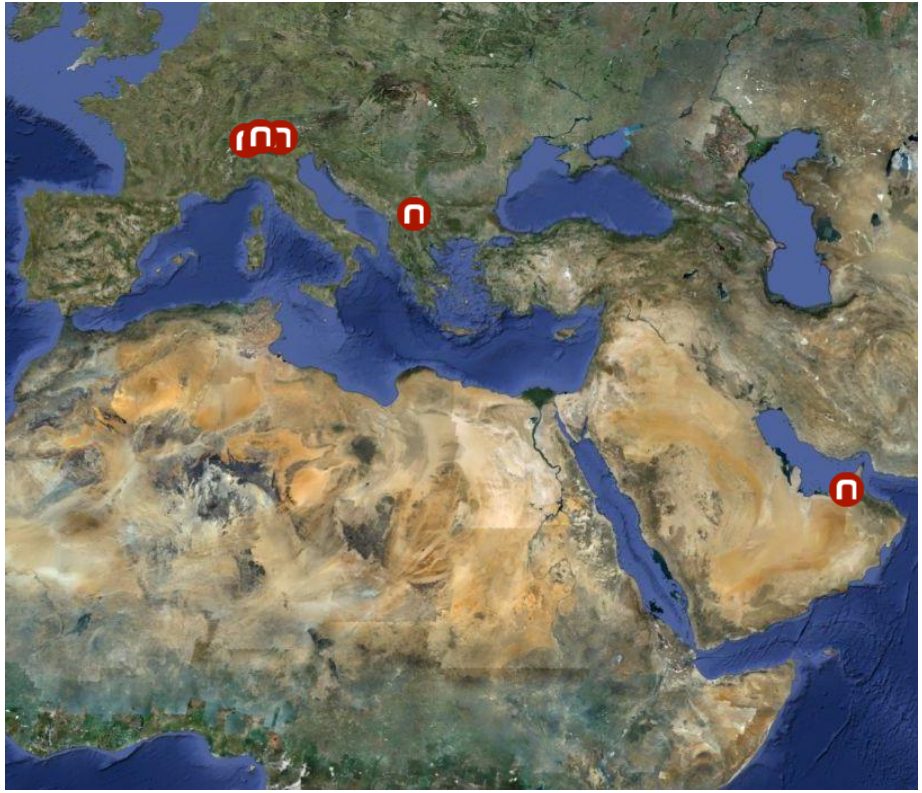
Replication

Fault tolerance, failover

Handling huge amounts of traffic

Sharding

Dealing with huge amounts of data



Geographical
Replication

Geographical Replication

Notes

Load is not an issue for us, but response time is

We have multiple geographically distribute sites

Network latency is a big factor of the response time if you are at a "far away" location

We have been thinking of setting up replicas of the search engine at the different locations

Relevance-Aware
Sharding

Relevance-Aware Sharding

Notes

Normal sharding distributes data on different, but equal nodes

We have been thinking about creating deliberately different nodes for the distribution of the data:

Node 1

- extremely fast
- small index, i.e. small amount of data
- lots of memory, CPU, really fast disks

Node 2

- lots more data
- big, but slower disks
- less memory and CPU

Frontend would send queries to both nodes and show results as they come in.

Distribution of the data would be based on the (query independent) boosting value

Wrapping Up

Search rocks

Apache Solr rocks

Learning Curve

Definition of **Done**

Continuous **Inspection**
Continuous **Improvement**

Get your hands dirty

Ranking Optimizations

Continuous **Testing**
and **Monitoring**
for **Ranking** and
Performance Issues

Verification
of features can take
a **long time**

Cool **side projects** rock

Contact

Corsin Decurtins
corsin.decurtins@netcetera.com
@corsin

References

Notes

Apache Solr
http://lucene.apache.org/solr/

Apache Solr Wiki
http://wiki.apache.org/solr/

Apache Solr on Stack Overflow
http://stackoverflow.com/questions/tagged/solr