

Le Fonds Européen de Développement Régional et la Région wallonne investissent dans votre avenir



[ Rencontres Mondiales du Logiciel Libre 2010 - Thursday, July 8 ]

# Introduction to libre « fulltext » technology

Author : Ir Robert Viseur



[www.cetic.be](http://www.cetic.be)

Your connection to ICT research

- My name : Robert Viseur
  - Civil Engineer, Master in Management of Innovation.
  - Specific expertise in the economics of free software and practices of co-creation.
  - Manager of *logiciellibre.com* (directory of free software companies).
  - Assistant in the Department of Economics and Management of Innovation ([mi.fpms.ac.be](http://mi.fpms.ac.be)) of the Faculty of Engineering, University of Mons ([www.umons.ac.be](http://www.umons.ac.be)).
  - Technology advisor at CETIC ([www.cetic.be](http://www.cetic.be)).
    - Belgian ICT Research and Technology Transfer Centre.
    - Initiator of Cellavi project (Center of Expertise for Open Source use in Industrial Applications).



# What do we talk about?

- Limits of conventional DBMS for text searching.
- Technologies for fulltext in DBMS (MySQL fulltext, PostgreSQL tsearch, Sphinx Search for MySQL).
- Free indexers (Lucene family, Xapian).
- We will not speak about NoSQL databases (Cassandra, CouchDB, HBase,...).

# Why is this useful?

- Look for articles in a CMS (*Content Management System*), for posts in a forum or for items in an online shop.
- Research in news, podcasts or RSS / Atom feeds.
- Research in the content of books or PDF papers.
- ...

25 résultat(s) pour 'sarkozy'.

---

[0.969] | [url](#) - [flux](#) (rss) | [lire](#) | **Libération - Désintox**

#### Desintox

... truqueur» de la statistique | Aide au développement : des promesses en l'air | Chômage des jeunes, Duflot ressert un quart | Partage des profits : Sarkozy radote, Wauquiez fayotte | Renault : Sarkozy balance des faux chiffres et du vent | Woerth et la baisse des dépenses de ...

Pays: fr - Langue: fr - Fraicheur: :- ) - Taille: 33138 caractères environ

Agréger dans: [Google](#) | [Yahoo!](#) | [Netvibes](#) | [Wikio](#) | [Webwag](#)

---

# Four steps

# What are the important steps?

- Four important steps:
  - the extraction,
  - the indexing,
  - the research,
  - the presentation of results.

# Step 1 : the extraction (1/2)

- Conversion of the file to index in plain text.
- Simple cases :
  - structured text files
    - Examples:
      - XML (with PHP::SimpleXML),
      - RSS (with PHP::SimplePie),
      - HTML (with PHP::strip\_tags or HTML analyzer).
  - documented complex formats
    - Examples: ODF = XML compressed file (ZIP).

# Step 1 : the extraction (2/2)

- Complex cases: undocumented binary formats.
  - Example: Office formats (97, 2000, XP ,...)
  - Use of Open Source projects:
    - Apache Jakarta POI (MS Office), Apache Tika (various documents), xls2csv (Microsoft Excel), catdoc (Microsoft Word), pdfinfo (PDF ),...
    - Extraction often imperfect (~ 20% error with POI).
  - Use IFilters (MS Windows).
    - Extensions proposed by the publishers themselves to extract the contents of files (Microsoft Office, Autocad, etc.).
- Scanned documents: OCR Open Source solutions.
  - See Tesseract, OCRAD and GOCR (still emerging).

## Steps 2 & 3: indexing and search

- What everybody knows: SELECT ... WHERE ... LIKE ...
- What is less known: the regular expressions.
- What you may be tempted to do: « *do it yourself* ».
- What it is instead recommended to do: use standard technologies.

## Step 4: presentation

- Export results to XML (API: OpenSearch standard format)
- XSL transformations to RSS, Atom,...
- Spellchecking (eg with Aspell or with algorithms like Soundex or Levenshtein distance)
- Classification of results (eg with Reverend).

0 résultat(s) jugé(s) pertinent(s) sur 0 pour la requête : sarkozi.

Il n'y a pas de résultat.

Voulez-vous dire ?

sarkozy • sahraoui

10067 news dans la base de données.

# « Do it yourself » approaches

# What happens if there are no fulltext solutions ? (1/2)

- This is particularly the case in SQLite.
- The most famous: LIKE
- Example: `SELECT news.title, news.url FROM news WHERE news.title LIKE '%linux%'`
- Possible improvements: decomposition of research in tokens, regular expression filtering, ...
- Disadvantage: problem of relevance, not suitable for large volumes of data.

# What happens if there are no fulltext solutions ? (2/2)

- The attached functions or regular expressions in SQL:
  - Example (PHP, SQLite):
    - `sqlite_create_function ($db, 'sqlite_fulltext', 'sqlite_fulltext', 2);`
    - `$sql = "SELECT * FROM torrents WHERE sqlite_fulltext (Search, ".  
Sqlite_escape_string ($ q ).") == 1 ORDER BY Title ";`
    - In "sqlite\_fulltext", using: `preg_match ("/\b($word)\b/i", $text).`
    - Disadvantage: not suitable for large volumes of data.
  - Regular expressions also supported by MySQL, PostgreSQL and Firebird (since 2.5).

# Is it possible to do by yourself?

## (1/2)

- Yes, but difficult development.
  - Create the dictionary:
    - Filtering of text by removing non-alphanumeric characters.
    - Decomposition of text filtered "terms" (tokens).
    - Removal of black words (the, the, the, my, your, their, our, your, their,...).
    - Establishment of a correspondence table (identifier of the document, term).
      - Each term is associated with a list of documents containing that term.
  - Write the good SQL requests...

# Is it possible to do by yourself?

## (2/2)

- Possible improvements:
  - Lemmatization terms.
    - Each term is replaced by its canonical form.
      - Example: studying, studying, student, students, ... => "study".
    - Several open source implementations of the Porter algorithm (eg Snowball).
  - Associations of the terms or lemmas with a phonetic form (soundex, Metaphone, etc.).
  - Warning: stemming and phonetic forms depend on the language.
    - Automatic language detection (eg in PHP : `Text_LanguageDetect` in PEAR).



# Fulltext standard solutions for DBMS

- SQLite: not included as standard (?) but SQLite FTS3 extension (untested).
- Firebird (untested): no standard module but extensions (including Sphinx Search).
- MySQL: MySQL FULLTEXT standard module and extensions (including Sphinx Search).
- PostgreSQL: PostgreSQL standard module tsearch (standard since v8.3).
- Other: Senna (untested).
  - Triton = MySQL + Senna | Ludia = PostgreSQL + Senna.
  - Cfr. Kazuhiko Shiozaki (Solutions Linux 2008).

- MySQL provides an automatic fulltext mode.
  - Creation:
    - `CREATE TABLE news ( id INT UNSIGNED AUTO_INCREMENT NOT NULL PRIMARY KEY, title VARCHAR(256), body TEXT, FULLTEXT (title, body) )`
  - Selection:
    - `SELECT id, title, body, MATCH (title,body) AGAINST ('linux') AS score FROM news WHERE MATCH (title,body) AGAINST ('linux') ORDER BY score`

- Strengths:
  - Supported on most shared web hosting services;
  - Support the creation of the dictionary and the analysis of the request;
  - Availability of search operators,
  - Evaluation of a score of relevance,
  - Mechanism for query expansion, ...
- Weaknesses:
  - No control over the analysis of the text (tokenisation but not stemming)
  - Minimum size of tokens (terms) set by default to 4 characters (not editable on a shared web hosting service).

# MySQL with Sphinx Search

- Sphinx Search extension must be compiled for MySQL.
- Support for PostgreSQL.
- Used by craigslist.org, mininova.org, ...
- Strengths: support very large volumes of data (> 100 GB of text), storage always provided by MySQL, portable.
- Weaknesses: stemming limited to English and Russian.

# PostgreSQL tsearch (1/2)

- PostgreSQL offers an automatic fulltext mode.
  - Creation:
    - ALTER TABLE tpages ADD COLUMN vecteur tsvector;  
UPDATE tpages SET vecteur=to\_tsvector(contenu);
  - Selection:
    - SELECT \* FROM docs WHERE vecteur @@  
to\_tsquery('tsearch2');
- Advanced Features:
  - Score of relevance with ts\_rank.
  - Creation of "snippet" with ts\_headline.

# PostgreSQL tsearch (2/2)

- Strengths: query parsing, stemming function of language.
- Weakness: not supported on shared web hosting services.



# What exists?

- Lucene (and its multiple ports),
- Other: Xapian,...

- Fork of Open Muscat (BrightStation PLC); C ++, GPL.
- Strengths: many bindings, import filters (extraction), stemming (many languages supported), synonymy (extensions of requests), correction of requests, support for indexing SQL databases (MySQL, PostgreSQL, SQLite, Oracle, DB2, MS SQL, LDAP and ODBC).
- Weaknesses: less popular.

- Supported by the Apache Foundation.
- Wide ecosystem:
  - Used in Alfresco, Jahia ...
  - Multiple integrations (eg CouchDB-lucene).
  - Many third-party tools: Luke (read index), Solr (search server; without crawler), Nutch (search engine with crawler), Carrot<sup>2</sup> (search interface compatible with OpenSearch and Solr),...
- Lucene index format becomes a kind of standard.

- Many ports (Perl, Python, .Net,...).
  - Lucene.Net (. Net) PyLucene (Python), CLucene (C++) Plucene (Perl), Zend Search (PHP).
  - Warning: functional coverage, release of supported index!
- Three types of port:
  - by literal translation (API compatible),
  - translation adapted for the target language (best performance),
  - by binding (for Python).

- Ability to change the text analyzers.
- Access to the dictionary of terms.
- Multiple search operators (AND, OR, NOT, +, -, ?, \*, ...).
- Exact or fuzzy search, management of synonyms, ...
- Ability to search by fields (eg title: linux).
- Ability to sort by field.



# What was tested?

- Taking into account:
  - the speed of index creation,
  - the speed of insertion
  - the speed of removal,
  - the speed of search.
- No systematic consideration of relevance.
- Two sets of data:
  - 20,000 textual data from 1kB to 900kB,
  - 200,000 textual data from 2kB to 5kB.

# MySQL, PostgreSQL, Sphinx Search (2008) (1/2)

- MySQL:
  - smaller index,
  - slower search compared to PostgreSQL or Sphinx Search,
  - deletion is slow
  - Insertion is very slow with bigger data.
- PostgreSQL:
  - index creation is slow,
  - insertion is very slow with bigger data.

# MySQL, PostgreSQL, Sphinx Search (2008) (2/2)

- Sphinx Search (with MySQL) :
  - manual (re)indexation (but very fast),
  - fast searches,
  - relatively insensitive to data size.

# Xapian, Lucene, PyLucene, Lucene.Net (2008)

- Xapian:
  - Slow when creating or updating the index, large index (compared to Lucene);
  - Installation more difficult.
- Lucene:
  - Performance fairly homogeneous (Lucene, PyLucene and Lucene.Net);
  - PyLucene significantly slower in creating and updating of the index (why?).

# Zend Search (2010)

- PHP technology built into the Zend framework.
- Easily hostable.
- Very useful for small volumes of data.
- Fragility of index (corruption under heavy solicitation in insertion).
- Example: [www.retronimo.com](http://www.retronimo.com) (search RSS).

# Discussion and conclusion

# Which technology to choose? (1 / 2)

- Database or index?
  - Indexer whether purely textual data.
  - Database if:
    - Structured data,
    - Need of relational model,
    - Need of SQL language.

# Which technology to choose? (2/2)

- Databases:
  - MySQL: well suited for basic solutions (relevance average, good performance on small data sizes), easily hostable, integrated platform LAMP / MAMP / WAMP.
  - PostgreSQL: well suited for professional solutions (but avoid with bigger data).
  - Sphinx Search: suitable for large volumes of data of any size.
- Indexers:
  - Lucene confirms its reputation as a reference.
  - Zend Search only useful for small volumes of data.

Thank you for your attention.

Questions?

# Some resources

- SQLite ([www.sqlite.org](http://www.sqlite.org)).
- MySQL ([www.mysql.com](http://www.mysql.com)).
- WampServer ([www.wampserver.com](http://www.wampserver.com)).
- Sphinx Search ([www.sphinxsearch.com](http://www.sphinxsearch.com)).
- PostgreSQL ([www.postgresql.org](http://www.postgresql.org)).
- Tritonn ([qwik.jp/tritonn/](http://qwik.jp/tritonn/)).
- Lucene ([lucene.apache.org](http://lucene.apache.org)).
- Zend framework ([framework.zend.com](http://framework.zend.com)).
- Xapian ([xapian.org](http://xapian.org)).

- SolR ([lucene.apache.org/solr/](http://lucene.apache.org/solr/)).
- Carrot<sup>2</sup> ([project.carrot2.org](http://project.carrot2.org)).
- Luke ([www.getopt.org/luke/](http://www.getopt.org/luke/)).
- Nutch ([nutch.apache.org](http://nutch.apache.org)).
- Tesseract ([tesseract-ocr.googlecode.com](http://tesseract-ocr.googlecode.com)).
- Apache POI ([poi.apache.org](http://poi.apache.org)).
- Snowball ([snowball.tartarus.org](http://snowball.tartarus.org)).
- Reverend  
([divmod.org/trac/wiki/DivmodReverend](http://divmod.org/trac/wiki/DivmodReverend)).

# Resources and useful links (1 / 3)

- Justine Demeyer (stagiaire), Robert Viseur (maître de stage) et Tom Mens (directeur de stage) (2008). Comparaison de technologies d'indexation fulltext. U Mons / CETIC, 2008.
- Robert Viseur (2008). "Solutions Linux: session sur l'indexation fulltext dans les SGBD". URL:  
<http://www.robertviseur.be/news-20080222.php> .
- Robert Viseur (2008). "Atelier de présentation du mode FULLTEXT de PostgreSQL 8.3 aux RMLL 2008". URL:  
<http://www.robertviseur.be/news-20080728.php> .
- Robert Viseur (2009). "Première comparaison de Tesseract, OCRAD, GOCR et... PhpOCR". URL:  
<http://www.robertviseur.be/news-20080726.php> .

## Resources and useful links (2/3)

- Erik Hatcher et Otis Gospodnetić (2004). "Lucene in Action". Manning Publications Co.
- "Annexe F. Expressions régulières MySQL". URL: <http://dev.mysql.com/doc/refman/5.0/fr/regexp.html> .
- "9.7. Pattern Matching". URL: <http://www.regular-expressions.info/postgresql.html> .
- Philippe Makowski (2009). "Firebird 2.5, les principales nouveautés". Code way 3. 16-20 novembre 2009. URL: <http://www.firebirdsql.org/download/rabbits/pmakowski/firebird-25.pdf> .
- "Does Firebird support full-text search?". URL: <http://www.firebirdfaq.org/faq328/> .

# Resources and useful links (3/3)

- Björn Reimer & Dirk Baumeister (2006). "Full text search in Firebird without a full text search engine". Firebird Conference Prague 2006. URL: <http://www.ibphoenix.com/downloads/FirebirdConf2006/TECH-TPZ303-R/TECH-TPZ303-R.zip> .
- "SQLite FTS3 Extension". URL: <http://www.sqlite.org/fts3.html> .
- "Full-Text Search on SQLite". URL: <http://michaeltrier.com/2008/7/13/full-text-search-on-sqlite> .
- "Tritonn - MySQL with Senna". Sumisho Computer Systems Corporation Brazil, Inc. URL: [http://qwik.jp/tritonn/about\\_en.files/tritonn-eng.pdf](http://qwik.jp/tritonn/about_en.files/tritonn-eng.pdf) .
- Kazuhiko Shiozaki (2008). "Moteurs plein texte sous MySQL et PostgreSQL pour la gestion de connaissances". Solutions Linux, 2008. URL: <http://www.robertviseur.be/news-20080222.php> .

- Ir. Robert Viseur.
- Email : robert.viseur@cetic.be
- Phone : 0032 (0) 479 66 08 76