Create a Specialized Search Engine
The Case of an RSS Search Engine

Robert Viseur
Université de Mons, Faculté Polytechnique, 20, Place du Parc, 7000 Mons, Belgium

Keywords: API, Atom, Crawler, Indexer, Mashup, Open Source, RSS, Search Engine, Syndication.

Abstract: Several approaches are possible for creating specialized search engines. For example, you can use the API of existing commercial search engines or create engine from scratch with reusable components such as open source indexer. RSS format is used for spreading information from websites, creating new applications (mashups), or collecting information for competitive or technical watch. In this paper, we focus on the study case of an RSS search engine development. We identify issues and propose ways to address them.

1 INTRODUCTION

RSS and Atom syndication formats are widely spreading in companies for technological or competitive watch. Unfortunately dedicated RSS and Atom search engines are not common, and famous Feedster worldwide search engine stopped some years ago. In this paper we focus on RSS and Atom search engine development. We identify issues et ways to address them.

2 BACKGROUND

2.1 Custom Search Engine

A search engine roughly consists of a crawler, an indexer and a user interface (Chakrabarti, 2002).

The crawler may seem simple to develop. However, it must be able to avoid bot traps, optimize bandwidth consumption, target wished web contents and respect the robots.txt protocol (Chakrabarti, 2002; Thelwall and al., 2005). Indexer must implement inverted index and support boolean operators for offering fast, accurate and customizable search (Chakrabarti, 2002; Viseur, 2010).

Fortunately some components (for example open source libraries and softwares) can be reused: large scale full search engine (e.g.: Nutch, Yacy), crawler (e.g.: wget), fulltext databases (e.g.: MySQL, PostgreSQL) or indexers (e.g.: Xapian, Whoosh, Lucene and other Lucene ports such as Zend Search, PyLucene or Lucene.Net) (Christen, 2011; McCandless, 2010; Viseur, 2010; Viseur, 2011).

Moreover it is possible to design custom search engines using reusable components and commercial search engines API inputs. That is the principle of open architectures. It is however needed to respect contracts terms of the open source and API licences (Alspaugh and al., 2009; Thelwall and Sud, 2012).

2.2 Search Engines API

Commercial search engines such as Google, Bing (and before MSN Search) or Yahoo! offer API to access their services (Foster, 2007; McCown and Nelson, 2007). The search engine API must be used to build new applications consuming search engine data such as meta-search engines (e.g.: merging results from several engines, offering graph representation of results, etc.).

Moreover, some search engines offer specific syntax for finding RSS feeds. For example Bing allows to search for RSS or Atom feeds by using "*feed:*" and "*hasfeed:"* operators (Bing, 2012). The first one gives list of RSS or Atom feeds. The second one offers list of web pages linking to RSS or Atom feeds. Those operators can be mixed with keywords, and "*loc:"* (or "*locations:"*) geographic or "*language:"* language operators for more accurate queries.

The API have some advantages. The developer using API do not have to maintain the search engine infrastructure (crawler, indexer, etc.), and can focus

Viseur R..
Create a Specialized Search Engine - The Case of an RSS Search Engine.
DOI: 10.5220/0004051302450248
In *Proceedings of the International Conference on Data Technologies and Applications* (DATA-2012), pages 245-248
ISBN: 978-989-8565-18-1

on new and innovative features. Commercial search engines index huge number of pages. Google thus indexes more than 8 billions pages (Boughanem and al., 2008). Moreover the search engine geographic coverage became very wide, and is still wider by combining several search engines results sets.

However the API have some disadvantages. Differences (e.g.: hit counts, ordering, etc.) are observed between Web User Interfaces and Application Programming Interfaces (Mayr and Tosques, 2005; Kilgarriff, 2007; McCown and Nelson, 2007). The access to the API can be restricted. The number of requests can be limited by user, by day or by IP address (Foster, 2007; Kilgarriff, 2007). The search engine companies can also charge a fee. It is the case for Google and Yahoo! (code.google.com; developer.yahoo.com). The technologies and the results formats can change over time. For example Google migrated its API from SOAP to JSON, and stopped its SOAP API and the first JSON API.

More generally the use of commercial search engines have some disadvantages. Operators can change over time, and even disappear. The "*linkdomain*" operator offered by Yahoo! or the "*near*" operator offered by Altavista are examples (Romero-Frias, 2009; Taboada and al., 2006; Turney, 2002). Problems of liability can occur with some operators. For example, Google recognized that "*link*" operator is not accurate (McCown and Nelson, 2007b). The results can slighly differ from an engine to another one (Véronis, 2006). Results can also be influenced by commercial partnerships, and geographic bias are observed (Thelwall and al., 2005; Véronis, 2006).

The limitations of the search engines and their API can be overcome by using results from several search engines. Meta-search engines can thus be feed by API or by components grabbing results from search engines Web User Interfaces (Srinivas and al., 2011). Grabbing results sets is not always allowed by search engines. Those ones are often able to detect and block automatic querying tools (Thelwall and Sud, 2012).

Given these limitations developing a custom search engine could be interesting (Kilgarriff, 2007; Thelwall and al., 2005). But you should overpass some difficulties such as the crawler development or the indexer configuration.

2.3 RSS Specifications

The RSS is a specification based on XML written in 1999 by Netscape (Gill, 2005; Lapauze and Niveau, 2009). This format allows to easily publish information such as news or classified ads. With a dedicated reader the users receive new contents in real time and no longer need to manually scan websites. The RSS files are called "*feeds"*. A RSS feed typically includes a set of items with properties such as "*title*", "*description*" and "*link*".

The RSS is used by newspapers or portal editors for widely spreading their contents. Companies use it for staying informed about competitors or technological changes. Developers can create new applications called mashups by aggregating data from API and RSS feeds, or index feeds collections for creating news search engines (Gulli, 2005; Jhingran, 2006; Samier and Sandoval, 2004).

RSS feeds are useful, and providing tools for effective search is important.

The use of RSS feeds nevertheless raises several difficulties. The RSS feeds are sometimes not well-formed (i.e it is not formed as defined in its specification). Moreover some properties are not normalized or sometimes users do not respect specification (e.g.: date and time specification of RFC 822). There are several RSS specifications (Gill, 2005). The first version from Netscape was numbered 0.90. The 0.91 specification quickly followed. In June 2001, Userland company released another 0.91 specification incompatible with that one from Netscape. The RSS-DEV Working Group published 1.0 version based on Netscape 0.90 and RDF format in late 2002. Between 2001 and 2003 Userland offered several specifications. The last one was numbered 2.0.1.

In 2004 Google supported new specification called Atom and later proposed it as a standard. The Atom Publishing Protocol became an IETF standard in October 2007.

3 DESIGN

We developed RSS search engine prototype based on following principles.

Bing search engine API is a useful tool for gathering RSS feeds due to the operators allowing accurate requests. However we saw that the features of search engine API were not stable over time. Furthermore the analyse of RSS feeds by commercial search engines is often poor. For example freshness or contents (e.g.: podcast) are not taken into account. It seems better to use feeds lists from Bing as complements to the URL discovered by the crawler. The link with search engine is thus weak and the tool can more easily evolve. A deep

analyse can then be processed: feed freshness (updates), podcast inclusion, etc.

Crawler is a tricky tool. Fortunately we do not have to develop a large scale crawler but a specific crawler (Prasanna Kumar and Govindarajulu, 2009). That one does not need to process deep exploration of websites. We propose to design the robot so that it passes as quickly as possible from one homepage to another ("*jumping spider*"). The aim is to explore the web as fast as possible (for finding RSS and Atom feeds), and save bandwidth. We know feeds are generally linked from homepage with RSS button or LINK HTML tag (Lapauze and Niveau, 2009; W3C, 2006). As suggested before feeds list may be fill by crawler and search engine API.

Feed reader used from gathering feed content must be able to understand several open and sometimes standard formats, and to implement "*liberal parsing*" (in order to understand feeds with invalid tokens). The analyse of feed content will also include freshness measure (based on date), use as podcast, language detection (by using 3-gram frequencies of known languages) and geolocalisation (by using domain name extension or locating IP server address). IP address mapping can be processed with HostIP tool (Gao and al., 2006). The indexation will be made with an easily hostable Lucene port. Lucene sorts results by extending well-known TF-IDF method (McCandless, 2010).

4 PROTOTYPE

Our prototype allows to quickly collect RSS and Atom URL from some URL seeds. A first indexation test worked fine. It is able to process searches by keyword, by language and by country. Queries can also be targeted for podcasts and for recently updated feeds only.

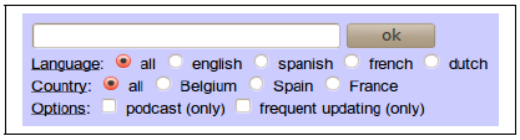

Figure 1: Query WUI.

We used standard RSS and Atom reader supporting liberal parsing and able to read a part of not-well formed feeds. New feeds can be found by fast crawl or by using search engine API ("*weak link*").

5 FUTURE WORKS

The crawler could be improved in order to allow to focus more precisely the links that it grabbed. For example, the jumping spider could automatically detect the URL geographic location or the topic of a page, and filter collected feeds (see Gao and al., 2006). Such an automatic detection of news sources would simplify the supply of a news search engine with relevant feeds.

The RSS feeds are used for various goals. Search for news feeds or for classified ads (e.g.: car or house sales) are distinct use cases. Training classification tools for detecting type of feed could allow to offer new search criterion. Other metadata could be used and grabbed from commercial search engine API. One example is the weight of a website that can be estimated by number of pages ("*site:*" operator in most of search engines).

Important expressions (entities) could be extracted from text contained in the feeds. The feasibility should be studied because some entities extractors use part-of-speech tagging tools which could be affected by the shortness of feeds contents. Entities extraction could allow to highlight popular tags and trends, or design query expansion feature based on entities associated to a results set (see Lohmann and al., 2009; Xu and Croft, 1996).

Newspapers often use to propose several feeds for different topics (Gill, 2005). It would be useful to group RSS by domain name in the results set. However some contents such as press releases can be published on several feeds and websites. Detection of duplicate contents and measures of similarities could be useful to suggest similar feeds, or on the contrary a short list of very different but relevant feeds (see Prasanna Kumar and Govindarajulu, 2009).

REFERENCES

Alspaugh, T. A., Asuncion, H. U., Scacchi W., 2009. Intellectual property rights requirements for heterogeneously-licensed systems. In *17th IEEE International Requirements Engineering Conference (RE'09)*, pp. 24–33, Augustus 31 - September 4, 2009.

Bing, 2012. Advanced Operator Reference, *MSDN* (msdn.microsoft.com). Read: February 3, 2012.

Boughanem, M., Tamine-Lechani, L., Martinez, J., Calabretto, S., Chevallet, J.-P., 2006, Un nouveau passage à l'échelle en recherche d'information. In *Ingénierie des Systèmes d'Information (ISI)*, 11 (4), pp. 9-35.

Chakrabarti, S., 2002. *Mining the Web*, Morgan-

Kaufmann Publishers.
Christen, M., 2011. Web Search by the people, for the people. *RMLL 2011*, Strasbourg (France).
Foster, J. C., 2007. Automating Google searching. In Long, J., *Google Hacking for Penetration Testers.* Syngress.
Gao, W., Lee, H. C., Miao, Y, 2006. Geographically focused collaborative crawling. In *Proceedings of the 15th International Conference on World Wide Web*, Edinburgh, Scotland, May 23-26, 2006), ACM Press, New York pp. 287-296.
Gill, K. E., 2005, Blogging, RSS and the information landscape: a look at online news. In *WWW 2005 Workshop on the Weblogging Ecosystem*, May 10-14, 2005, Chiba (Japan).
Gulli, A., 2005. The Anatomy of a News Search Engine, In *WWW 2005*, May 10–14, 2005, Chiba (Japan).
Jhingran, A., 2006. Enterprise Information Mashups: Integrating Information, Simply. In *VLDB 2006*, September 12-15, 2006, Seoul (Korea).
Kilgarriff, A., 2007. Googleology is Bad Science. In *Computational Linguistics*, 33(1), pp. 147-151.
Lapauze, J., Niveau, S., 2009. Agrégation de flux RSS. In *RICM5*, 6 novembre 2009.
Lohmann, S., Ziegler, J., Tetzlaff, L., 2009. Comparison of Tag Cloud Layouts: Task-Related Performance and Visual Exploration. In *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part I.*
McCandless, M., Hatcher, E., Gospodnetic, O, 2010. *Lucene in Action*, Manning Publications; 2 edition 2010.
Mayr, P., Tosques, F., 2005. Google Web APIs - an instrument for Webometric analyses?. In *Proceedings of the ISSI 2005 conference.*
McCown, F., Nelson M. L., 2007a. Search engines and their public interfaces: which apis are the most synchronized?. In *WWW '07 Proceedings of the 16th international conference on World Wide Web*,
McCown, F., Nelson M. L., 2007b. Agreeing to disagree: search engine and their public interface. In *JCDL'07*, June 18-23.
Prasanna Kumar J., Govindarajulu P., 2009. Duplicate and Near Duplicate Documents Detection: A Review. In *European Journal of Scientific Research*, Vol. 32, Issue 4, pp. 514-527.
Romero-Frias, E., 2009, Googling companies - a webometric approach to business studies. In *The electronic journal of business research methods*, Vol. 7(1), pp. 93-106.
Samier, H., Sandoval, V., 2004. La veille sur les weblogs. In *Actes du colloque VSST*, Toulouse, October 2004.
Srinivas, K., Srinivas, P. V.S., Govardhan, A., 2011. Web service architecture for meta search engine. In *International Journal of advanced computer science and applications*, Vol. 2 n°10, pp.31-36.
Taboada, M., Anthony, C., Voll, K., 2006. Methods for creating semantic orientation dictionaries. In *Proceedings of Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, Genoa, Italy, pp. 427-432.
Thelwall, M., Vaughan, L., Björneborn, L., 2005. Webometrics. In: *Annual Review of Information Science and Technology*, 39, pp. 81-135.
Thelwall, M., Sud, P., 2012. Webometric research with the Bing Search API 2.0. In *Journal of Informetrics*, 6(1), pp44-52.
Turney, P. D., 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, Philadelphia, Pennsylvania, 2002, p. 417-424.
Véronis, J., 2006. *Etude comparative de six moteurs de recherche*, Université de Provence, 23 février 2006.
Viseur, R., 2010. Introduction to libre fulltext technology. In *RMLL 2010*, Bordeaux (France), July 6-11, 2010.
Viseur, R., 2011. Développement d'un moteur de recherche avec Zend Search. In *RMLL 2011*, Strasbourg (France), 11-14 juillet 2011.
W3C, 2012. Use <link>s in your document. In *W3C* (www.w3.org), November 24, 2006 (read: March 13, 2012).
Xu, J., Croft, B. C., 1998. Query expansion using local and global document analysis. In *SIGIR'96*, Zurich.