2000

Searching the WWW with XML

Matthew Montebello
University of Malta, mmont@cs.um.edu.mt

Robert Caippara
University of Malta

Searching the WWW with XML

Matthew Montebello, Robert Caippara
Department of Computer Science and Artificial Intelligence, University of Malta, Malta.
mmont@cs.um.edu.mt

Abstract

Searching and retrieving the right information from the World-Wide Web (WWW) has always been considered of foremost importance and of considerable A.I. intensivity. Internet search technologies have been evolving over the years and will continue to do so as the WWW will continue to expand in size and increase in popularity. In a desperate attempt to restore order to the WWW after the chaos that has developed due to its heterogeneous, unstructured and uncensored nature, the eXtended Markup Language (XML) is being heralded as the successor to HTML. In this paper we investigate the evolution of Internet search technologies and present a possible and viable solution in a functional system we developed and which makes use of XML at its very core. We discuss the design issues involved as well as practical issues such as tendencies and tactics employed by some of the major players in this well-sought area.

Introduction

In recent years there has been a well-publicised dissatisfaction with the methodologies employed in identifying relevant information available on the WWW (Berners-Lee et al., 1994). This problem is due to the increase in the WWW sheer size and information overload (Montebello, 1999), as well as to the inability of traditional search engines to efficiently and effectively index the information over the same web. In an attempt to precisely describe WWW documents and eventually index them appropriately, metadata tags were initially suggested and used. Metadata tags contain information such as author, publication dates, keywords, and so forth, and is commonly defined as data about data. Search engines could take advantage of metadata, but the failure of HTML authors to abide with the metatags' criteria, partly due to the weakly typed nature of HTML, and its misuse or abuse, make plain metadata inpractical. This gave rise to the development of the Extensible Markup Language (XML) (W3C, 2000), which strictly expreses the structure of data within web documents, thereby giving rise to a possible and sensible solution to optimally index WWW information. The rest of the paper is organised as follows. In the next section we discuss the problem tackled in some detail, while XML itself is discussed immediately afterwards. This prepares the path for a system we designed and developed to take advantage of XML and optimally index WWW documents. Our conclusions and future work will follow in the final section.

The WWW and the search engines

The Internet is the greatest repository of information man has ever created. It contains information on almost any subject conceivable. According to a survey carried out in the beginning of 1999 (Forecast Worldscape Strategies, 1999), the publicly indexable World Wide Web contained about 800 million pages encompassing around 15 Terabytes of data. Since then a year and a half has passed and the Internet has continued growing at a phenomenal rate.

There is all this information out there, and yet many people are not happy with the service they receive when using search engines. The problem, however, is not the search engines themselves. The main search engines like Altavista (Altavista Search, 2000), Hotbot (Hotbot, 2000), Yahoo (Yahoo!, 2000), and others, are in fact very sophisticated machines and carry out their tasks very well.

There are two main problems that normally arise when issuing a query. The first is that you get fewer results than you might expect, considering there are all those millions of pages of information. The second is that usually you have to browse through the first 20 or 30 results to find exactly what you are looking for. These problems have very different causes and have to be tackled differently.

The first problem is merely a question of resources. No company has enough resources to spider the whole Internet. In fact, few search engines spider more than 10% of the net, with the very best of them not exceeding 15% (Lawrence et al., 1999). What's more is that the Internet is growing at a faster rate than the search engines, and so these estimates are always decreasing. One innovative way to improve on this situation is to employ meta-search engines that combine the results of multiple search engines (Montebello, 1998a). It is estimated (Search Engine Watch, 2000) that the overlap between the engines is relatively low at around 42%, however it is still much better than the results obtained from any single engine (Montebello, 1998b).

The second problem is much more serious. Not finding enough information is one thing; not finding any information is another. The problem here lies not with the search engines' abilities to rank the pages correctly according to your query, but more with the way that the information is currently presented on the Internet. Most of the information that is found on the Internet is in the form of HTML text. The problem with HTML however is that all the effort in the document goes to the way in which the information should look on the screen. All the machine-readable information in an HTML document is related directly to the presentation of that document by the browsers. There is no meta-data about the information content of the document itself.

Imagine having a large spreadsheet on your computer that is stored as a graphical image. It may look very good, but that's where the good things end. You, as the user, may be able to look at the image and read the data, but that data is not machine-readable. This means that many important functions cannot be carried out on that data, for example searching for a particular value. This is synonymous to the situation with HTML and it makes searching in HTML documents a hard business.

Coming back to our spreadsheet image, the trick is to keep the data in a real spreadsheet where the computer can carry out all the necessary functions such as searching, editing and so on. The data is then output as an image before presenting it to the user. This way the output is as good-looking as before but the data is still machine-readable and –editable. The concept therefore is to separate the presentation and the content of a document. This allows you to mark up the content so that a computer can understand the contents as well as the user.

XML

This problem on the Internet is being tackled by the introduction of a new alternative to HTML. XML uses a set of rules for defining semantic tags that break a document into parts and identify the different parts of the document. XML is not just another mark-up language like HTML. HTML defines a fixed set of tags that describe a fixed number of elements. If the mark-up language you use does not contain a tag you need – then you're out of luck.

XML, on the other hand, is a meta-mark-up language. It is a language in which you make up the tags you need as you go along. These tags must be organised according to certain general principles but they're quite flexible in their meaning. XML defines a meta-syntax that domain-specific mark-up languages like MusicML, MathML and CML must follow in areas of Music, Mathematics and Chemistry respectively. If an application understands this meta-syntax, it automatically understands all the languages built from this meta-language.

A browser does not need to know in advance each and every tag that might be used by thousands of different mark-up languages. Instead it discovers the tags used by any given document as it reads the document or its Document Type Definition (DTD). The detailed instructions about how to display the content of these tags are provided in a separate style sheet that is attached to the document. XML mark-up describes a document's structure and meaning. It does not describe the formatting of the elements on the page. The document itself only contains tags that say what is in the document, not what the document looks like.

As far as search engines are involved, this separation of content and presentation and the marking up of the document content provides the following useful advantages:

- The information presented in the document is presented in a very structured manner that makes it much easier to parse (and hence, understand) by a computer.
- Much more information is implicitly defined in the document. This makes it much easier for the correct documents to be picked by a search engine in response to a query.

An XML search engine

The figure overleaf shows the major components of the system we designed and developed, namely, the *Initiator*, *Spider Farm* and *Indexer Farm* at the centre of the functional underlying application. Other components that contribute to the functionality of these three major components are the *Index* itself, the *Link Checker*, and the *Validator*. Finally, two other components which concern the user interface itself are the *Administrator Interface* and the actual *Front-End* which a web user accesses to make use of the search engine's services. A more detailed description of the various components and the way they interact with each other follows:

The Initiator: This component maintains a list of web page addresses (or URLs) which the system will download and parse. These URLs will eventually be passed to the Spiders through the Spider Farm.

Spider Farm: Spider Farm mainly manages the Spider threads that will be crawling the WWW. When a new spider is to be created, a request from Spider Farm to the Initiator is made and a URL is passed onto the new spider to access the actual document.

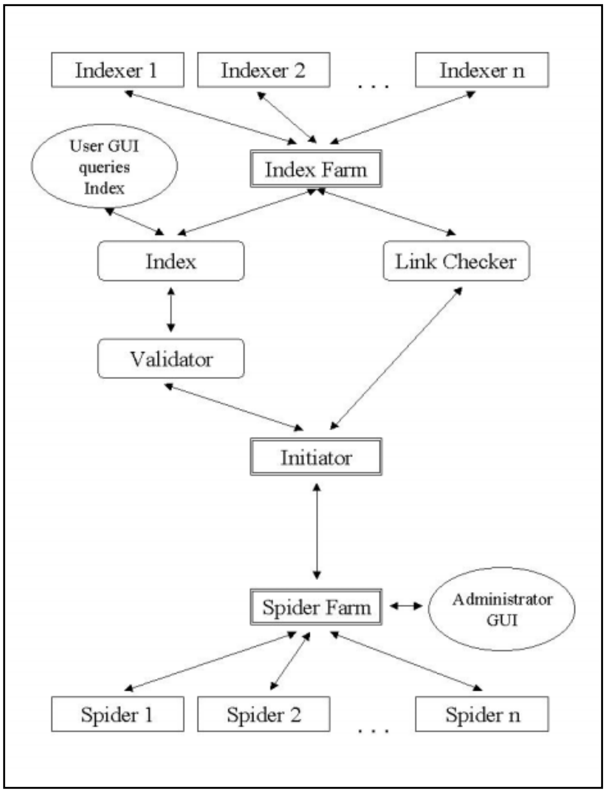
Indexer Farm: This component has a similar function as the Spider Farm as it manages a group of threads, in this case Indexers. This Farm receives downloaded document from Spider Farm and forwards them to one of the Indexers just created.

The Index: All indexed documents will be recorder within the Index. The URL, date last indexed, indexed contents, etc... will all be stored within the systems database to be available when a user query is to be satisfied.

Link Checker: A record of all the links that have been spidered by the system together with the date when they were last spidered is held and employed by the Link Checker to ensure that all URLs coming from the Indexer Farm need to be spidered or not.

Validator: The Validator is a periodic component which goes through the whole index looking for documents which have been residing in the index over a set period of time. Those documents which pass this time threshold are resent to the Initiator for the system to update the index while ensuring that all indexed URLs are still live.

GUIs: There are two kinds of user interfaces which have been developed to accompany our system. The first one is an administrator GUI whose main purpose is to echo to the outside world what the system is doing. At the same time this interface is used to update any system settings like maximum number of spiders and indexers, time threshold employed by the Validator, etc... The second GUI is the one that a web user will access in order to query the XML search engine. This has the task of accepting a query from a user, parses the query, find any indexes that match the query and return the results to the user.



Conclusions

Internet search technologies have been, are, and will continue to be a vital part of the WWW itself. Users depend upon them when utilising the web for any of their needs. The evolution of these technologies has been analysed in this paper, leaving an interesting question of how will the trend for future generations of search facilities will be. We argued in favour of employing the XML framework as a basis to develop a search engine which spawns spiders in search of XML documents. In this paper we presented the basic architecture of such a functional system and discussed its major components. In future we will be performing in depth evaluation tests to analyse the effectiveness of our system and attempt to improve and optimise the information retrieved and indexed in order to make good use of the available WWW information resources.

References

Altavista Search, www.altavista.com, (Current May, 1, 2000).

Berners-Lee, T., Caillian, R., Luotonen, A., Nielsen, H.F., and Secret, A. "The World-Wide Web," *Communications of the ACM,* 37(8):76-82, 1994.

Forecast Worldscape Strategies, "*The Internet Market Review,"* www.imr.com, (Current Dec, 1,1999).

Hotbot, www.hotbot.lycos.com, (Current May, 1,2000).

Lawrence, S., and Giles, C. L., "Accessibility of Information on the web," *Nature*, Vol. 400, pg. 107-109, July, 1999.

Montebello, M., "Metasearch and Machine Learning to optimise WWW searching," in proceedings of the *ninth international conference on Computing and Information (ICCI'98) pg. 245-252,* Winnipeg, Manitoba, Canada, 1998a.

Montebello, M., "Optimising Recall/Precision scores in Information Retrieval over the WWW," in proceedings of the *21st. Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98) pg. 361-362,* Melbourne, Australia, 1998b.

Montebello, M., "*Personalised Information Retrieval over the WWW,"* Ph.D. thesis, Computer Science Department, Cardiff University, Cardiff, U.K., 1999.

Search Engine Watch, "Tips About Internet Search Engines & Search Engine Submission", http://searchenginewatch.internet.com/.

W3C, The World-Wide Web Consortium, *Extended Markup Language*, www.w3.org/XML/, (Current May, 1, 2000).

Yahoo!, www.yahoo.com, (Current May, 1, 2000).