WebDocBall: A Graphical Visualization Tool for Web Search Results

Jesús Vegas[1], Pablo de la Fuente[1], and Fabio Crestani[2]

[1] Dpto. Informática
Universidad de Valladolid
Valladolid, Spain
{jvegas,pfuente}@infor.uva.es
[2] Dept. Computer and Information Sciences
University of Strathclyde
Glasgow, Scotland, UK
F.Crestani@cis.strath.ac.uk

Abstract In the Web search process people often think that the hardest work is done by the search engines or by the directories which are entrusted with finding the Web pages. While this is partially true, a not less important part of the work is done by the user, who has to decide which page is relevant from the huge set of retrieved pages. In this paper we present a graphical visualisation tool aimed at helping users to determine the relevance of a Web page with respect to its structure. Such tool can help the user in the often tedious task of deciding which page is relevant enough to deserve a visit.

1 Introduction

Searching the Web is one of the most frequent tasks nowadays. Unfortunately, it is often also one of the most frustrating tasks. The user gets little help from a Web search engines or a Web directories. The former lacks precision, while the scope of the latter is often too small. The only tool the user has to initiate the search is usually just a small set of query terms and, after the query process, the user is left with a huge amount of information to process manually. While much work has been devoted to design and develop algorithms to improve the query process, little has been done to help the users to find the needle in this "*document stack*". In the remainder of this paper we will use the terms We page and document interchangeably.

According to Shneiderman, the search process can be divided in four phases [15]:

1. formulation: the information need is expressed in the query.
2. search: the query is submitted to the search engine and is processed.
3. reviewing of the results: the retrieved document set obtained is reviewed by the user.
4. refinement: the query is reformulated in a manual or semi-automatic way to try to improve the retrieval results.

The third phase is where *information visualisation* can help the user improve the effectiveness of the manual task of browsing the results set. In this phase, the user has to identify a relevance relation between the query he formulated and the results obtained. Usually, the user sees a list of URLs and titles of documents with little more information about them, so this task is hard and tedious and, frequently, the user has to wait several seconds to see that a page is not interesting. It is clear that this is the phase where the user needs assistance to identify potentially relevant documents and discard those that are not useful. This phase is very important, since it has a major impact on the user satisfaction with a system.

The areas of interest in this phase are three [12]: set level, site level and document level.

At the set level the objective is to represent the entire set of results. Several techniques has been developed, such as, for example, scatter-plots and star-fields [1], maps or landscapes metaphors [5], wall metaphor [11], or cone trees [13].

The Web site level tries to locate the retrieved documents in the Web site structure, since the user could be interested in Web sites, not just Web pages. It also tries to tackle the "lost in hyperspace" problem [10].

At the document level, the objective is to help the user to decide if one page is relevant to his/her information need or not. Our work is directed at this level. For this task, we propose a graphical tool to assist the user in the review phase. The visualisation tool displays to the user the structure of the HTML document and the estimated relevance of its parts with respect the query. The tool is based on two assumptions: pages have a hierarchical structure, and the result set can be post-processed. Both assumptions can be easily satisfied.

The paper is structured as follows. In Section 2 we review related work, concentrating on visualisation tools used at document level. In Section 3 we introduce a new tool for the visualisation of retrieved documents in Web searches. In addition, we compare our proposal with related work. A description of a Web search engine that incorporated the visualisation tool is reported in Section 4. Finally, Section 5 reports our initial conclusions and provides an outline of future work.

2 Information Visualisation Tools at the Document Level

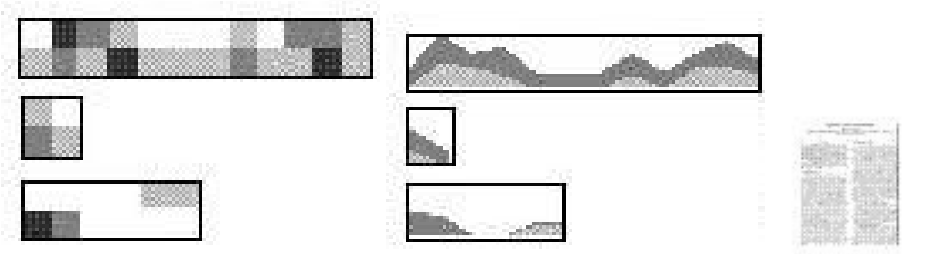
The document level is the one in which the user analyse the answers obtained from a Web search engine. Here, a large number of pages (the actual number can vary from one to thousands) are presented to the user, who has the hard task of identifying those relevant to his/her information need. Usually, the user is provided with little help for this task. The most important tools designed for this tasks are Tilebars [9], relevance curves [14] and the thumbnail view [4].

Before we present and compare about these three tools, we need a framework for comparison. In order to build it, we need to answer the following question: what does the user need when he is browsing at the document level a result set obtained from a Web query? An established answer, to which we fully subscribe,

is that the user not only needs to know about the relevance of a document, but also "where" the document is relevant.

The vast majority of documents that are available on the Web have some hierarchical structure [2], which can either be explicit, like in XML documents, or implicit, like in HTML documents. We argue that the user could get very quickly an idea of the actual relevance of a document by looking at the way the parts estimated to be relevant are distributed in it. In addition, the user can be interested not only about the concentration of relevant parts, but in what are the hierarchical relations among them. Another important information about the retrieved document is its type. This can be inferred from the structure of the document. The user could be more interested in a document with a very flat structure, than in one with a more deep structured. Another important aspect of the retrieved documents is its size. A short letter and an extend technical report are clearly not the same.

So, there are three aspects that can help the user analyse the results of a Web search at document level: the relevance the document and how this is distributed on the document parts, the structure of the document, and the size of the document.



a) Tilebars b) Relevance Curves c) Thumbnails

Figure 1. Tilebars, Relevance Curves and Thumbnails.

Figure 1 shows document representations using Tilebars, relevance curves and thumbnail. The first two representations show the relevance of each passage or segment of the text, using a row for each query term. The size of the documents is represented by the Tilebars or the relevance curve size. The passages can correspond to paragraphs or sections, or units of fixed arbitrary length. Both Tilebars and relevance curves show whether and where the document is relevant. However, the use of the structure is only done in a linear way, being it impossible to appreciate the hierarchical relations between the structure elements found in the document. The third document representation is the thumbnail. In this case no information about the size or the structure of the document is provided. The only information represented is related to the appearance of the document. Therefore, a thumbnail representation can complement the two first representa-

tions and it is useful when the user works frequently with the same document set and can recognise a document by its thumbnail view. The only information about the structure that the user can obtain from this representation is the information that can be inferred from the appearance of the different elements in the document, and this is not enough, since the user also needs information about the relevance of the different parts of the document.

3 The Webdocball

In this paper we present a visual metaphor that can explain the user why a Web document was retrieved by a Web search engine, by showing where in the structure of the document estimated relevance lies. We call this metaphor *Webdocball*, in analogy to the Docball metaphor presented in [16], of which it is an extension in the Web domain. This visualisation element is included in a prototype system that works on top of a Web search engine by post-processing the results obtained from the search engine to allow the user to browse the result set using different representations of the documents. Figure 2 shows the Webdocball representation of a Web document.

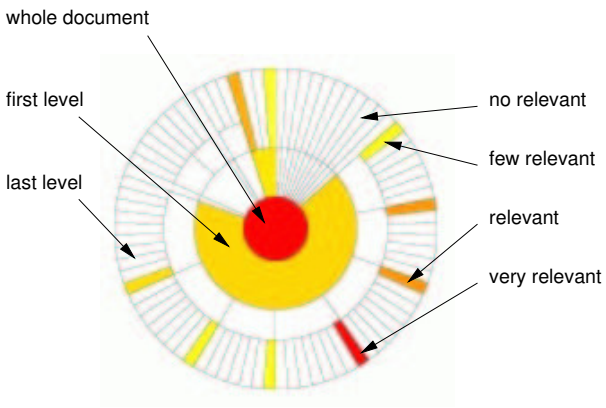


Figure 2. The Webdocball metaphor.

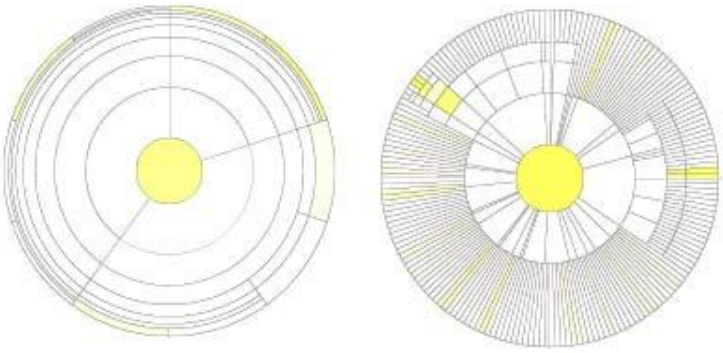
In the Webdocball, each ring represents a level of the hierarchical structure of the document, where the centre represents the entire document. In a ring, the regions represent the structural elements in that level. Inside a ring elements are presented in a clockwise order.

Each region in the Webdocball is coloured with regard to its estimated relevance: a hotter colour indicates more relevance, a colder colour represents less relevance. White regions have been estimated to be not relevant.

Regions are "clickable", so that the Webdocball can be used as a navigation tool of the Web document.

The Webdocball representation can be considered as the representation of a tree structure, which depends on two parameters: the number of elements in the deeper structural level (the number of leaves) and the number of structural levels in the document (the height of the tree).

A Web document with a large number of elements will have very narrow regions in the corresponding rings. The regions corresponding to leaves are represented from their level ring to the exterior ring. In addition, in order to enable a clear representation of Web documents with many levels, the height of the rings decreases the deeper the structural level it represents. This feature was not present in the Docball representation. Such feature can be seen in Figure 3, where different Webdocball are shown with respect to the number of leaves and structural levels of the document represented. The Webdocball (a) represents a Web document with a deep structural hierarchy, whereas the document represented by the Webdocball (b) is more flat with respect the its structure, but has a larger number of leaves.



a) b)

Figure 3. Different Webdocballs representing different structures of Web documents.

3.1 Comparing Webdocball and Tilebars Representations

In this section, we compare the Webdocball and the Tilebars representations. We do not consider relevance curves because they can be considered just a version of Tilebars. Also, we do not consider Thumbnails because they do not represent structural information.

A Webdocball exploits the structure of the documents to show the estimated relevance of the documents, in a similar way to Tilebars, but it goes one step beyond Tilebars. In Tilebars the structure is represented in a linear way, and it is impossible to extract the hierarchical relations between the structural elements as it can be done in the Webdocball representation. So, a Webdocball allows

to visualise the structural complexity of the retrieved documents. However, it should be recognised that Tilebars enable the representation of the length of the document, something that is not possible in the Webdocball representation.

Another difference is that in Tilebars, the relevance of each single query term is displayed, whereas a Webdocball represents the relevance of all the query terms. In the case of few query terms Tilebars can be better that Webdocballs. However, in the case of many query terms, Webdocballs are clearly better. In fact, Tilebars cannot manage query expansion by relevance feedback process, where many query terms are often added to the query. In such case, Webdocballs are more useful, since the relevance representation is independent of the number of query terms.

4 A Description of the Webdocball System

The Webdocball visualisation tool is part of a prototype Web search engine we have developed on top of an existing search engine. The prototype is written in Java and the Webdocball visualisation tool has been developed using the SVG (*Scalable Vector Graphics*) standard [6]. The Web search engine is built on top of Google using the Google Web APIs[1]. Google uses the SOAP (*Simple Object Access Protocol*) [8] and the WSDL (*Web Services Description Language)* [7] standards, enabling us to use Java for the prototype development.

Since a Webdocball represents the estimated relevance of a Web document at each structural level, we need to be able to determine the structure of the Web document, and to estimate the relevance of each structural element with respect to the query. To achieve this, we need to post-process the results set obtained from Google to obtain the information needed to draw the Webdocball. All of this can be seen in the diagrammatic system architecture shown in Figure 4.

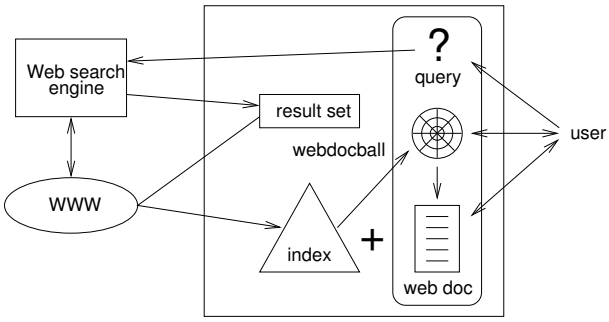


Figure 4. Architecture of the Webdocball system.

[1] Google Web APIs are a free beta service and are available for non-commercial use only at `http://www.google.com/apis`

The query process is the usual one: the system enables the user to submit a natural language query, which is processed by Google. The search results are presented to the user as a list of Web documents with associated Webdocball representations. An example of result list is presented in Figure 5.

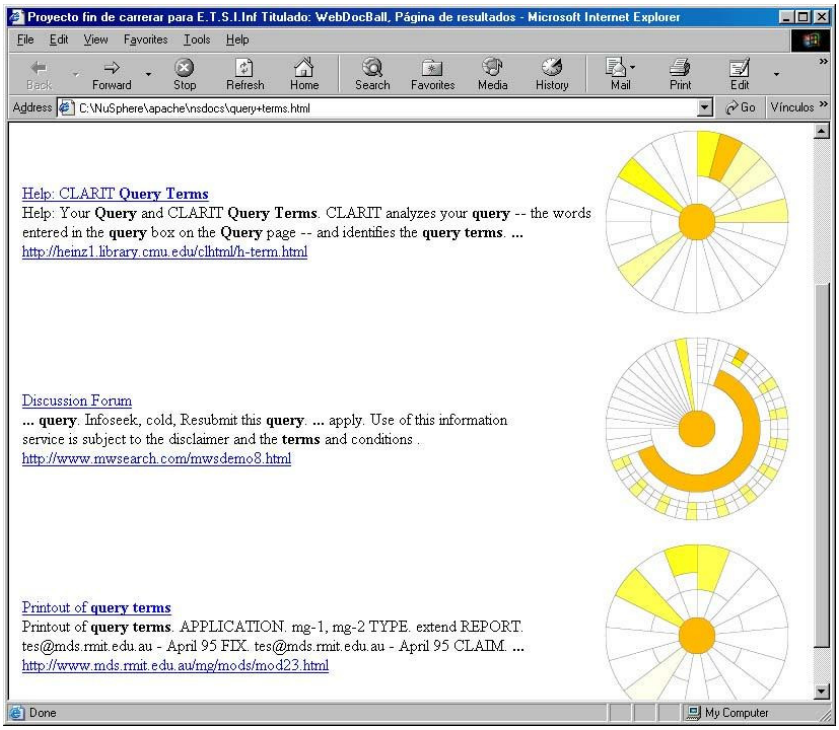


Figure 5. Results in the Webdocball system.

In the result list each document is presented with its KWIC (*Key Word in Context*), the link to the site where is the document, and the correspondent Webdocball. The Webdocball is connected with a post-processed form of the original Web document which is stored in a local cache. The Webdocball is then a simplified version of the Web document that shows to the user information related to the document structure, which is coloured in relation to the estimated relevance degree of each element. From this post-processed document, the user can visit the original one, either by accessing the system cache or by going to the real Web page following the link.

In the next sections, we are going to explain the two most interesting phases of the process of construction of a Webdocball representation: the document structure extraction and estimation of relevance of each structural element of the document.

4.1 Extracting the Structure of a Web Document

Documents on the Web are of many different types, but mostly they are HTML documents [2], although XML, PDF and other types of documents can also be found. As explained in the previous section, in order to construct a Webdocball we need to determine the structure of Web documents. To obtain this from XML documents is very easy, since the document structure is expressed explicitly. Unfortunately, the number of XML documents available on the Web is very limited, and most documents are in HTML, so we have to extract structural information from HTML labels. We found several problems in this process, the main two being that HTML labels are used with no discipline by the Web documents authors, and that the HTML labels express structural information in an implicit way. Therefore we need to answer these two questions: which are the most common HTML labels in the Web and what are their structural significance.

To answer these questions we build a random URL generator using the Google search engine. We retrieved via Google, using a random set of words, 30,000 distinct HTML pages, comprising 13,124,339 HTML (labels). From an analysis of the results we found that the most often used labels were, in decreasing order: `content`, `p-implied`, `td`, `br`, `tr`, `img`, `p`, `comment`, `table`, `span`, `li`, `script`, `div`. `meta`, `input`, `title`, `center`, `hr`, `ul`, `body`, `html`, `head`, `form`, `link`, `select`, `frame`, `object`, and `ol`.

In addition, we obtained some important information about the structure of HTML documents:

- The average number of structural levels in HTML documents is 10.66.
- Tables are widely used, with an average of 10 tables per page.
- The paragraph label (`p`) is used 13 times per page on average.
- The number of frames, forms and links per page is very low, less than 2, on average.

From the previous label set, we have chosen to consider those labels that have structural meaning, that is, labels used in Web documents to express hierarchical relations. These were the following:

- tables: `table`, `tr`, and `td`.
- lists: `ul`, `ol`, and `li`
- paragraphs: `p`, and `br`.

We do not use the main structural level labels `html`, `body` and `head` because there is only one of each per page and they do not add structural information that could be used to discriminate between the retrieved documents.

To understand how a Web document is represented using a Webdocball metaphor, let us consider the Web document shown in Figure 6, as it can be viewed using the Netscape We page editor (the boxed labels represent the HTML labels). This is a page of an introduction course in IR. The document is composed by four tables (title, navigational tool, content, and foot). The tables are used to organise the content in the Web page. The most interesting table is the

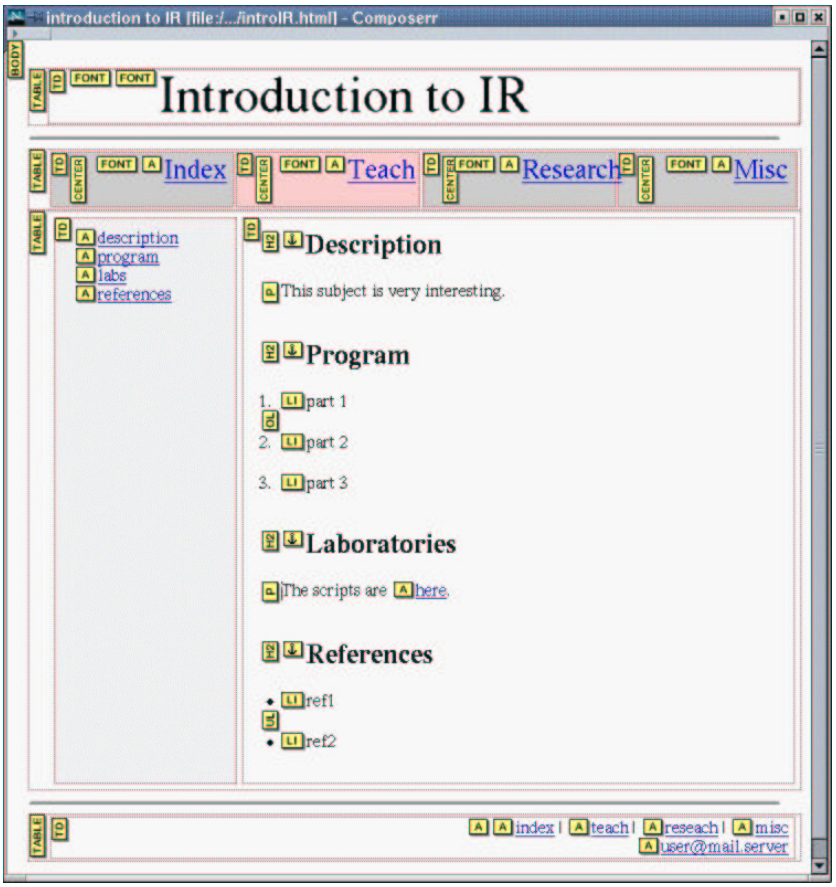


Figure 6. Example of Web document.

content one, because its left column is used to index the content in the right column. In this right column, there are several HTML elements (paragraphs and lists, mainly) with structural meaning. This structural information can be seen in the Webdocball depicted in Figure 7.

4.2 Determining the Relevance of a Structural Element

In order to colour the Webdocball we need to estimate the relevance of each structural element of the Web document with respect to the query. To do this we use a modified version of the Vector Space Model (VSM). The original VSM is not suitable to index Web documents, because it does not consider important aspects of the Web like the hyperlinks and other information useful to index the Web documents with success (see [3] for more information about indexing the Web). However, it can still be very useful for our very specific purpose.

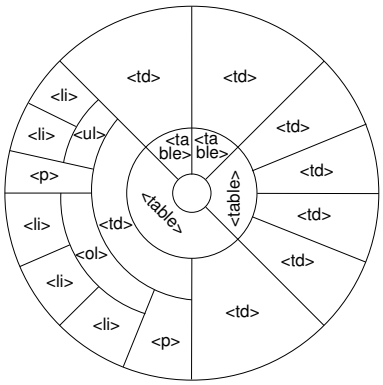


Figure 7. The Webdocball of the Web document in the Figure 6.

The model we propose considers a Web document as the collection of items to index, where the elements of the document are the items to index. In a Web document there are two types of elements, content (text) and structural elements (labels). We can transform a Web document in a collection of items to index in a two phase process: first each content element is indexed using the VSM, then the indexing weight of the content elements are spread to the top of the structural hierarchy (i.e. towards the centre of the Webdocball) as indicated by the structural elements. This process is more formally explained in the following.

Let us assume that a Web documents d is composed of a hierarchy S of m structural elements s, and can be viewed as d=$\{s_1, s_2, \ldots, s_m\}$. Two structural elements are related by an inclusion relation, $s_i \subset s_j$ when s_i is included in s_j according to the hierarchy S. The inclusion relation is called directed, denoted by $\subset'$, if $\nexists s_k \ / \ s_i \subset s_k \subset s_j$. A structural element will be a leaf element when there is no element included in it.

A structural element s_j in a document will have a weight $w_{i,j} \geq 0$ with respect to the index term k_i calculated as follows:

$$w_{i,j} = \begin{cases} \frac{freq_{i,j}}{max_l freq_{l,j}} \times \log \frac{m}{m_i} & \text{if } s_j \text{ is a leaf} \\ \\ \sum_{\forall s_l \subset' s_j} w_{i,l} & \text{if } s_j \text{ is not a leaf} \end{cases} \quad (1)$$

Therefore, as in the VSM, the weight vector of a structural element s_j in the document d is represented by $s_j = \{w_{1,j}, w_{2,j}, \ldots, w_{t,j}\}$.

The similarity between the query vector and the structural elements vector can be calculated using cosine correlation, as it is done in the standard VSM.

5 Conclusions and Future Work

We have presented a graphical tool for the visualisation of Web documents retrieved by a Web search. The proposed tool uses the structure of the Web docu-

ments to show the user where estimated relevance is located inside the document. In addition, we have developed a prototype system, based on Google, that retrieves documents from the Web and uses a modified version of the Vector Space Model to estimate the relevance of each structural element of Web documents to a user query.

Currently, we are carrying out an evaluation of the visualisation tool. Preliminary tests have been satisfactory with respect the appearance and utility of the visual metaphor, as it was already found for the Docball metaphor [16]. The most serious drawback, at this stage, is the answer time of the system, which it is very high due to the amount of post-processing work needed to calculate and to draw the Webdocball representation. We are currently working to reduce the time necessary to do that.

The work presented in this paper is part of a wider work aimed at the design, implementation, and evaluation of a complete system for Web retrieval. The next step in this project will be to answer the question of where retrieved documents are located in a Web site. We plan to use the same visual metaphor to represent the search results at the set, site and document level, to help the user in the Web searching process.

Acknowledgements

Large part of the implementation of the GUI has been carried out by Javier López González, student at the University of Valladolid.

This work was partially supported by the Spanish CICYT program (project TEL99-0335-C04).

References

1. C. Ahlberg and B. Shneiderman. Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. In *Human Factors in Computing Systems. Conference Proceedings CHI'94*, pages 313–317, 1994.
2. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.
3. S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.
4. S.K. Card, G.G. Robertson, and W. York. The Webbook and the Web Forager: an Information Workspace for the World Wide Web. In *Proceedings of the Conference on Human Factors in Computing Systems CHI'96*, 1996.
5. M. Chalmers. Using a Landscape to Represent a Corpus of Documents. In *COSIT'93*, 1993.
6. SVG Working Group. *Scalable Vector Graphics (SVG) 1.1 Specification. Candidate Recommendation*, 2002. http://www.w3.org/TR/2002/CR-SVG11-20020430/.
7. Web Services Description Working Group. *Web Services Description Language (WSDL) Version 1.2. Working Draft*. http://www.w3.org/TR/2002/WD-wsdl12-20020709/.
8. XML Protocol Working Group. *SOAP Version 1.2 Part 0: Primer. Working Draft*. http://www.w3.org/TR/2002/WD-soap12-part0-20020626/.

9. M.A. Hearst. Visualization of Term Distribution Information in Full Text Information Access. In *ACM SIGCHI Conference on Human Factors in Computing Systems (CHI)*, 1995.
10. J. Lamping and R. Rao. Laying out and Visualizing Large Trees using a Hyperbolic Space. In *ACM Symposium on User Interface Software and Technology*, pages 13–14, 1994.
11. J.D. Mackinlay, G.G. Robertson, and S.K. Card. The Perspective Wall: Detail and Context Smoothly Integrated. In *CHI'91*, pages 173–179, 1991.
12. T.M. Mann. Visualization of WWW Search Results. In *DEXA Workshop*, pages 264–268, 1999.
13. G. Robertson, J. Mackinlay, and S. Card. Cone trees: Animated 3d Visualizations of Hierarchical Information. In *Conference on Human Factors in Computing Systems CHI'91*, pages 189–194, 1991.
14. Arisem S.A. http://www.arisem.com.
15. B. Shneiderman. *Designing the User Interface. Strategies for Effective Human-Computer Interaction*. Addison-Wesley, 1992.
16. J. Vegas, P. de la Fuente, and F. Crestani. A Graphical User Interface for Structured Document Retrieval. In *ECIR 02, BCS-IRSG European Colloquium in Information Retrieval Research*, pages 268–283, March 2002.