# BRight: A Distributed System for Web Information Indexing and Searching

Pedro Falcão Gonçalves
Silvio Lemos Meira
Ana Carolina Salgado

Departamento de Informática, Universidade Federal de Pernambuco
C.P. 7851, Cidade Universitária, 50.732-970 Recife - PE, Brazil
Fax: +55-81-271-8438, email: {pfg,srlm,acs}@di.ufpe.br
web: http://www.di.ufpe.br/~{pfg,srlm,acs}

## Resumo

O tamanho extraordinário e o crescimento exponencial da World-Wide Web demandam novas abordagens aos problemas de indexação e pesquisa de informação em sua estrutura. Neste artigo, discutimos as limitações da abordagem centralizada atualmente em uso, comentamos trabalhos recentes em arquiteturas distribuídas para Recuperação de Informação em Redes, e apresentamos o BRight!, um sistema distribuído para indexar e pesquisar informação na World-Wide Web. Em seguida, focalizamos o BRight! para discutir sua arquitetura e seus conceitos subjacentes de Visões de Web, fazemos comparações com outros sistemas, e mostramos sua escalabilidade em relação ao crescimento da Web. A versão atual do protótipo é apresentada, e trabalhos futuros são delineados.

**Palavras-chave:** Web, indexação, pesquisa, arquiteturas distribuídas, escalabilidade.

## Abstract

The extraordinary size and the exponential growth of the World-Wide Web demand new approaches to the problems of information indexing and searching in its structure. In this paper, we discuss the limitations of the centralized approach in use today, comment recent work on distributed architectures for Networked Information Retrieval, and present BRight!, a distributed system to index and search information in the World-Wide Web. We then focus on BRight! to discuss its architecture and underlying concept of *Web Views*, compare it with other systems, and show how it scales to Web growth. The current version of a prototype is presented, and future work is outlined.

**Keywords:** Web, indexing, searching, distributed architectures, scalability.

## 1. Introduction

The World-Wide Web (Web) is evolving to one of today's most popular communication technologies. With estimates of 100 million pages, the Web size doubles each six months, and the number of servers and users have also grown at similar paces [Admedia 1997, Bell & Semmell 1996, Cooper 1994]. Given this extraordinary size and exponential growth, the Web represents the largest and most widely used electronic information base ever publicly available.

As a consequence of Web growth, the computational cost of locating relevant information in it has become increasingly higher [Berghel 1996]. Simply browsing its distributed hypertext content to look for relevant information has long become too time consuming for the user. Besides this, continued Web growth now challenges the several indexing services available [Campbell 1996, Mitchell 1996].

The problem of selecting Web documents that are relevant to a given user is essentially related to Information Retrieval (IR) [Frakes 1992]. There is a huge distributed hypertext-based document set (the Web), and systems are needed to assist users in locating relevant information in this set. However, successful IR systems to date are only capable of dealing with corpora in the megabyte range or a small number of gigabytes [Crowder & Nicholas 1996], while the Web is in the terabyte range and still growing steadily. To fill this considerable gap, much research activity has been carried out recently [Connolly & Soley 1996, Führ 1996, Schwartz 1996]. In this context, we propose BRight! (BRazilian Internet Guide in HyperText), a *Web View*-based scalable distributed system for information indexing and searching in the Web.

Although our presentation focuses on the problem of information indexing and searching for the Web, the approach taken here applies to distributed hypertexts in general. Furthermore, similar problems are found in the integration of Digital Libraries [Fox 1996, Buckland 1995] and Networked Information Retrieval [Führ 1996], given the large amount of information to be indexed and the distributed nature of the information base.

The rest of the paper is organized as follows: Section 2 presents the centralized approach to Web indexing and searching, and discusses its limitations; Section 3 presents general characteristics of selected distributed systems that have been designed to deal with related problems; Section 4 introduces BRight!, its architecture and the concept of Web Views, and discusses how BRight! scales to Web growth. Section 5 presents the structure of the current version of the prototype, and outlines the next steps of the work on the system. Finally, Section 6 comments on how BRight! is related to other NIR systems, and adds concluding remarks.

## 2. Centralized Web Indexing and Searching

Most popular Web indexing and searching systems are based in one of the following strategies: (1) subject index hierarchies built and maintained by hand (e.g., [Yahoo! 1997, Cadê? 1997]), and (2) index bases automatically updated by Web traversing programs (the so called *Web robots* – e.g., [Altavista 1997, Excite 1997]).

The first option requires human approval to append items to the index, and human intervention to edit the index hierarchy. Although there are popular such systems, Web size and growth tend to increase the demand for human work to keep the indexes complete and up-to-date. This kind of index is already difficult to keep up-to-date now, and tend to be ever more difficult.

Web robots, the second option, are programs that continually traverse the space of Web pages by following hypertext links, and index the contents of selected pages into an Index Base (IB). The IB is generally implemented on a Database Management System (DBMS), and used by the search engine to provide keyword- and expression-based searching services to users. This situation is illustrated in Figure 1, where a user interacts with the search engine (e.g., [Altavista 1997, Excite 1997]) which accesses its IB to process queries. A robot works to keep the IB up-to-date. Robots continually traverse the Web looking for new and updated pages to index.

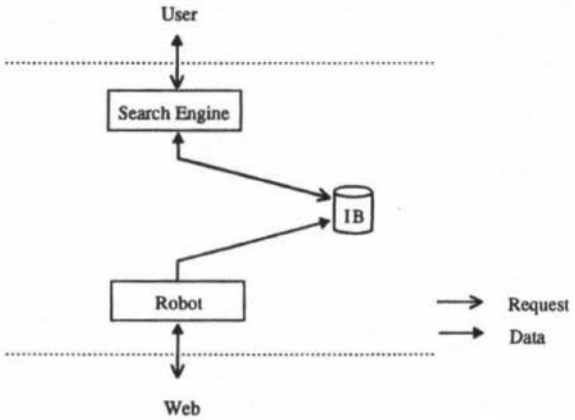


Fig. 1: General architecture of centralized Web indexing and searching systems currently in use.

Robot-based systems do not require any human intervention during the process of selection and indexing of information items, except at configuration time [Koster 1997, Sullivan 1996]. To tackle the scaling problem of Web growth, the automatic nature of this framework is obviously more attractive than the manually updated lists option, but it still has serious limitations to cover the entire Web, as discussed below.

***Building Indexes***

Web robots usually work by retrieving documents and locally processing them to extract indexes. They recursively follow hypertext links found in the processed documents in order to perform exhaustive Web indexing. To index the whole Web with this strategy, a robot needs to retrieve all accessible pages and process them, which demands considerable processing power and network bandwidth. Still, to keep the

index up to date the robot has to revisit all the pages periodically, and the computational costs grow with Web size. Therefore, this solution is not scalable.

***Storing Indexes***

Indexes are usually stored in a DBMS under a single database schema. Although several servers may be used, the schema is generally centralized. Maintaining a centralized IB to index the whole Web requires high processing power and storage space, so Web growth is also a problem in this aspect.

***Searching in IBs***

Centralized indexes are usually accessible to a large number of users. This fact demands reasonable response times, bringing Database and IR technologies to the core of this issue. As the Web grows both in number of pages and users, computational demands tend to increase due to index size and number of user requests. However, the Web is much larger and dynamic than the document sets indexed by real Database and IR systems to date. Extensive research is being carried out in these areas to face the scaling problem in document collections size.

Given these scaling limitations of the centralized approach, and the steady growth of the Web, centralized solutions tend to be ever less feasible to the problem of Web indexing and searching. Actually, very few current search engines claim to be complete and up-to-date in the coverage of the Web.

To go around this problem, specialized partial indexes and *metasearchers* have been proposed and implemented, but these still leave the problem of global indexing unsolved. Specialized partial indexes focus on subsets of the Web, such as *intranets* [Microsoft 1997, Open Text 1997] and sets of servers in given geographic regions [Muscat 1997, NWI 1997]. If the set of pages to be indexed is smaller than the whole Web, and there exists an algorithm to traverse only these pages, then robots may be configured to index only the specified set. What metasearchers do [Falk & Jonsson 1996, Go2Net 1996, Tai 1996] is act as clients of other search services. Instead of running their own robots, they simply take user requests and forward them to several other services, merges their results, and delivers them to the user.

Clearly, global Web indexing based on centralized exhaustive browsing already has excessive computational costs, and the Web is still growing. In addition, partial indexes and metasearchers leave the problem unsolved. In the next section, we will look into recent distributed approaches to this problem.

## 3. Distributed Architectures for Networked IR

In general terms, Networked IR (NIR) deals with distributed heterogeneous collections of documents, typically served by different information providers in a long distance range. The goal, in this context, is to build systems that assist users in the selection and retrieval of relevant documents from the whole collection, according to each user information needs. Therefore, the user should not worry about where and how searches are performed.

To achieve this goal, several distributed architectures have been recently proposed and tested. For instance, the MeDoc Information Brokering System [Boles *et al.* 1996] aims at helping users find relevant information on Computer Science. It offers transparent access to several bibliographic and full-text databases. CAFE [Crowder & Nicholas 1996] is a multiagent system, in which specialized agents manage local corpora using IR engines as back-ends. CAFE assumes that a single entity has control over the entire corpus, while MeDoc deals also with third party databases. Another example is GlOSS [Gravano & García-Molina 1995], which implements a "Glossary-of-Servers Server" to support decisions on what databases should be used to process a given query. As well as MeDoc, it deals with third party databases but gives a different treatment to brokers by organizing them in a hierarchical structure.

The architectures of most NIR systems, such as the ones mentioned above, have some basic components in common. Figure 2 shows a layered architecture outline that depicts the three central components described below: the User Agents layer, the Brokers layer, and the Provider Agents layer.

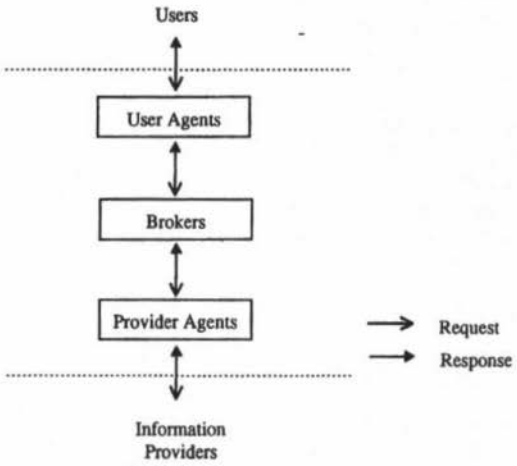


Fig. 2: General layered architecture of Networked Information Retrieval Systems.

The *User Agents* layer consists of a collection of user interface agents. Each user agent typically offers a different interface to the same set of system functionalities, and is session-oriented. The *Brokers* layer contains information brokers that are able to communicate with agents from the other layers in order to receive information requests (from user agents) and have them solved (by using one or more provider agents). Each *Provider Agent* is specialized in accessing a specific class of information providers. Given a broker request, a provider agent decides how to perform the necessary operations on the appropriate information providers in order to fulfill the request.

Besides the common aspects among their architectures, NIR systems also share some design goals. Below, we comment on two of these goals: scalability and modularity.

***Scalability***

The number documents managed by the information providers may grow, and so may the number of information providers to be accessed. The NIR system should be ready to deal with this scaling problem.

***Modularity***

Information providers have different interfaces, offer different functionalities, and manage different types of objects. Modularity is desirable in the *Provider Agents* layer to hide these details from the higher level functions of the system. The same applies to the *User Agents* layer, where particular user interaction styles are irrelevant to more central operations of the system.

Other research directions have also been recently explored in the area of NIR, especially in the Web. Given the proliferation of Web search engines, the STARTS protocol [Gravano *et al.* 1997], for example, has been proposed to facilitate the task of querying multiple document sources in the Internet. It is a group effort coordinated by the Stanford Digital Library project, and involving over 11 companies and organizations.

Another example is the integration of the Web with CORBA [Edwards 1996, Merle *et al.* 1995], which provides a means for object oriented programs to perform dynamic object loading and linking over the Web. This means that client applications do not need to know, at compilation time, how to handle all the object types they will be using. This integration has interesting applications, for instance, in resource discovery. If objects of any unknown types are available in such repository, an indexing robot would still be able to manipulate them and extract information from them.

## 4. BRight! - A Web View-Based Architecture

The architecture of BRight! is based on a set of autonomous modules for indexing the Web: the *Index Servers* (ISs). Each IS has its own IB and focuses on a particular scope within the Web. Each IS also runs a *Web Robot* that continuously traverses the Web looking for pages that fall within its scope, and therefore should be indexed. ISs cooperate by exchanging indexes.

A higher level of abstraction is built on top of the Index Servers: the *Index Brokers* (Brokers). They form another set of autonomous modules that select the appropriate ISs to respond to user requests, and then manage search sessions, possibly involving multiple ISs. This structure is depicted in Figure 3.

As shown in Figure 3, each Web Server $WS_i$ relies on a database (DB) where its Web pages are stored; each Index Server $IS_i$ has an Index Base (IB); and each Index Broker $B_i$ has a Scope Base (SB), where scope information about known ISs and pointers to other Brokers are kept. Index Servers request pages from Web Servers for indexing, and Index Brokers request indexes from Index Servers to respond to the users' search requests. Index Servers cooperate by exchanging indexes (e.g., $IS_1$ and $IS_3$ in Figure 3), and Index Brokers cooperate by exchanging metainformation about Index Servers (e.g., $B_1$ and $B_2$ in Figure 3).

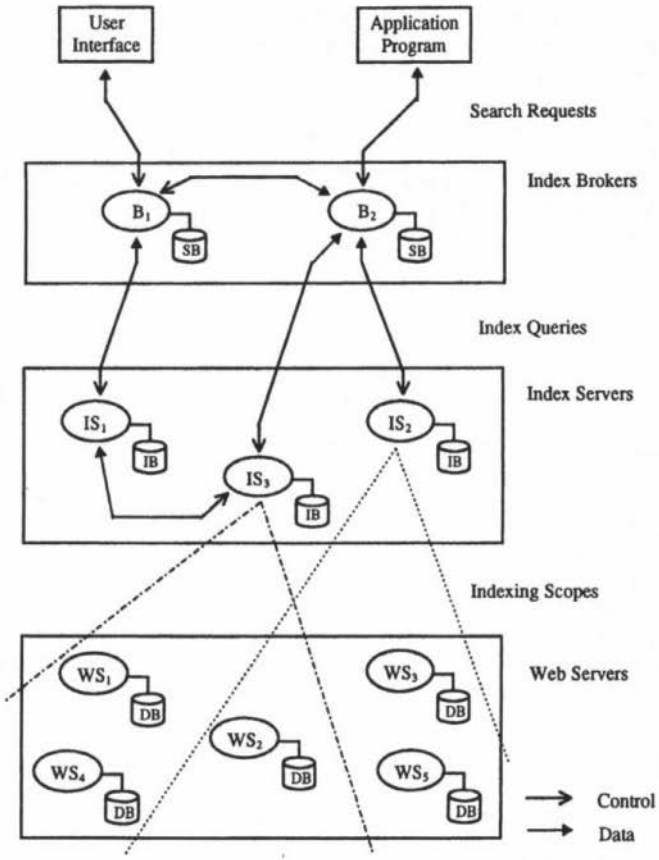


Fig. 3: The proposed distributed architecture for Web indexing and searching

### 4.1. Web Views for Defining Scopes

Two basic goals motivated the development of the concept of Web Views (Definition 6) to the problem of indexing and searching: (1) to achieve computational efficiency, through an architecture that allows for a feasible solution to the problem at hand, and (2) to achieve effectiveness by providing a means to locate documents that are relevant to users' queries. In order to pursue these goals, the definition of Web Views is done in terms of four concepts of Digital Distance, as stated in Definitions 1 through 4.

***Definition 1: DD by Geographic Location (DDGL).***

The DDGL between two documents is the physical distance, in kilometers (Km), between the two Web servers where the documents are stored[1].

***Definition 2: DD by Network Location (DDNL).***

Given two documents, the DDNL between them is calculated according to the IP (Internet Protocol [Tanenbaum, 1992]) addresses[2] of the Web servers where they are stored. Table 1 gives the association of values.

| WEB SERVERS | - | DDNL |
|---|---|---|
| same IP address | - | 0 |
| same class C network | - | 1 |
| same class B network | - | 2 |
| same class A network | - | 3 |
| different class A networks | - | 4 |

**Table 1: Calculating DDNL's.**

***Definition 3: DD by Hypertext Location (DDHTL).***

The DDHTL between two documents is the length of the shortest known hypertext path between the two documents in the Web. The DDLHT from any document to itself is zero.

***Definition 4: DD by content (DDC).***

The DDC between two documents is calculated according to a function $f_{DDC}$: $\mathbf{Doc}^2 \rightarrow \mathbf{N}$, where **Doc** is the set of all Web documents, and **N** is the set of all natural numbers. The DDC between two documents with identical contents is zero, and there may be documents with different contents whose DDC is also zero, depending on the $f_{DDC}$ function chosen.

Given the definitions of Digital Distance, a simple way to characterize a scope can be defined: the Digital Neighborhood.

***Definition 5: Digital Neighbourhood (DN) of type X.***

The DN of type X and radius R around a reference Web document is a space in which are included all the Web documents whose DD of type X to the reference document is less than or equal to R, where X denotes a single type of DD (*i.e.*, DDGL, DDNL, DDHTL or DDC).

The servers locations, geographical and topological, are used in the definitions of DDGLs and DDNLs, respectively. This allows for the identification of groups of Web Servers that are either physically located within a given geographic region, or connected to a given IP subnetwork. Indexing modules that focus on their geographical area or network neighborhood avoid the need to use long distance backbones for indexing, and have a smaller number of pages to index.

---

[1] This concept of physical distance may be extended to deal with more complex shapes than a circle [Star and Estes, 1990] in order to model more meaningful regions, such as states and countries.

[2] The concept of Network Distance may be adapted to deal with other definitions of network locality (e.g., IPng [Stevens, 1996]).

The documents content is used in the definitions of DNHTLs and DDCs. This accounts for indexing effectiveness by introducing measures of similarity between documents based on their content and also on the hypertext links connecting them. Indexing modules can then be specialized in given subject areas. Techniques from the area of Information Retrieval may be used to choose the $f_{DDC}$ function of Definition 4. For instance, a simple function to compute the distance between two documents can be based on the comparison of lists of indexing terms from these documents [Smeaton & Quigley, 1996]. Moreover, the relevance of hypertext distance (DNHTL) comes from the use of the structure given by authors to Web documents. This structure constitutes an important way to identify correlated information [Buckland *et al.*, 1993; Brown, 1988].

Web documents can be characterized by: (a) server location; (b) document content; (c) the object where the document is stored (*e.g.*, an ordinary file, a database entry, or a hypertext document); and (d) metainformation extracted from the document's content (e.g., title and language). Attributes (c) and (d) above compose the documents' *Representation Scheme*.

Now, given the definitions of DN and Representation Scheme, a broader way to characterize scopes can be defined: Web Views.

***Definition 6: Web View.***

A Web View is a composition of DNs optionally filtered by a set of restrictions on Web pages. These restrictions apply to the attributes used in the documents' Representation Scheme.

An example of search scope that could be modeled by Web Views is:

> *Consider all documents that are similar in content to the document in "http://somehost/x.html", and that are stored in servers at most 200 km away from Fortaleza, CE, Brazil.*

Restrictions such as "only documents updated later than 25.04.97" and "only documents written in Portuguese" can also be added, as stated in Definition 6.

### 4.2. A Distributed Architecture Based on Web Views

In Figure 3, each IS module performs indexing and/or searching. Indexing is done by a software robot that collects information from the Web within a given Web View, and stores it in an IB. Searching is performed on IBs, which may be local or remote. Modules cooperate to share indexing efforts and searching services, as described below.

When a new IS module is set up, an indexing scope is defined by the user, and an initial list of known modules is automatically bootstrapped. The indexing scope is given by a Web View, and the list of known modules also contains their respective scopes in the form of Web Views. At this point, the system tries to expand the list of known modules by activating a Broker module. If scope intersections involving the new module are detected, the user is questioned about the extent of shared resources the new module should use from others. Two basic forms of resource sharing are described below.

Let $M_1$ be the new IS module with scope $S_1$, $M_2$ an existing IS module with scope $S_2$, and $I_{12}$ the non-empty intersection between $M_1$ and $M_2$, that is, $I_{12} = (M_1 \cap M_2) \neq \varphi$.

***Sharing Indexing Efforts***

The new module ($M_1$) may rely on the existing module's ($M_2$) indexing effort instead of independently performing the indexing of their scope intersection ($I_{12}$). Then, $M_1$ periodically requests updates from $M_2$ about the indexing of $I_{12}$. This saves $M_1$'s computational resources, and also saves resources from the Web servers in $I_{12}$.

***Sharing Searching Services***

$M_1$ may not want to keep duplicated index data about $I_{12}$. Instead, it may share $M_2$'s searching services, that is, it does not keep any indexes about $I_{12}$, and queries $M_2$ every time it needs to perform searches involving $I_{12}$. This saves $M_1$ both computational resources and storage space, but makes it dependent upon another module to perform queries, which implies in less fault tolerance and possibly higher response times.

Assuming that the typical user interaction style with Web search engines is based on series of requests, our system adopts a session oriented interaction style. At the beginning of a searching session, the user contacts a broker module and is asked to define a searching scope within which the session should proceed. The interactively defined scope is then translated into a Web View representation which can be handled by the system.

At this point, the broker module in use checks whether the IS modules it knows are enough to cover the requested scope (*i.e.*, whether there exists a combination of known ISs that suffices). If this is the case, the broker is ready to process the session's requests. However, if a solution is not found to cover the requested scope with the ISs it knows, it then enters client mode and requests help from other known brokers in other to find alternative ISs.

In order to perform Web View operations on the searching scopes of various modules, such as finding an indexing scope composition that matches (or approximates) the selected searching scope, an algebra of Web Views is needed. A simplified version of such algebra is presented in [Gonçalves *et al.* 1997a], which allows for the verification of the inclusion relation between Web Views. Further comments on the structure of IS modules are given in [Gonçalves *et al.* 1997b], and a broader motivation to the problem can be found in [Gonçalves 1996].

### 4.3. Facing Web Size

Drawing a parallel with the centralized approach discussed in Section 2, the following topics discuss the benefits of the distributed approach in terms of computational efficiency.

***Building Distributed Indexes***

Instead of a single indexing module, the distributed approach allows for the installation of several cooperating modules, each of which with its own indexing scope, to run in different points of the network. Given that each module does not need to index the entire Web, processing power and network bandwidth requirements will be lower to each module.

By sharing indexing efforts as described in Section 4.2, redundant indexing by various modules (and the consequent waste of computational resources) can be avoided. This represents savings both to the indexing modules and to the Web servers involved.

***Storing Distributed Indexes***

Not all the indexes of a module need to be stored locally. To save disk space, one module may behave as a client of another module when a given subscope is concerned. Conceptually, it would still be covering its entire scope, although part of the indexes for that scope is physically stored in another module's IB. We call this practice "IB sharing", and point out that one way to achieve this is by sharing searching services, as described in Section 4.2.

For performance and fault tolerance reasons, one or more modules may keep copies of the indexes to a common subscope. In this case, sharing indexing efforts would still be an advantage (*i.e.*, the two concepts are orthogonal).

***Searching in Distributed IBs***

In a search session, one or more ISs may be used to cover the search scope specified by the user. If one IS suffices, there is conceptually no difference to the centralized approach. If, however, various ISs are used, then parallelism is introduced, and the broker that is interacting with the user manages the execution of the various remote subqueries submitted to other modules.

The broker determines the final ordering of the response items before delivering them to the user. Assuming that the results of the subqueries submitted to other modules are part of the final result to be presented to the user, response time may be prioritized if the total ordering in the responses is relaxed. This issue, however, is beyond the scope of this paper, and is analogous to the collection fusion/merging problem [Voorhees 1995, Voorhees *et al.* 1994].

The additional cost of finding a set of modules to cover a given searching scope is revealed only at the beginning of a searching session. Once the module selection is made, the set of search requests in the same session are handled with the same modules, exploiting the parallelism between them.

## 5. The Prototype

A prototype implementation is currently in operation, running under the Solaris Operating System using a Relational DBMS (RDBMS) to store the IB. Two other versions are under development, one for Windows NT with an RDBMS, and another for Solaris with an Object-Oriented DBMS. The aim of running versions of the prototype under different platforms is to test the openness of the distributed system from the conceptual level down to the implementation level, including module's communication mechanisms and heterogeneous IB sharing.

A sample running module using the Brazilian Internet as its indexing scope is publicly accessible at http://www.di.ufpe.br/~pfg/bright.html[3]. The indexing robot and the mobile code based interface are written in Java [Flanagan, 1996], for portability and object-orientation reasons. Web hypertext hierarchies are visible to the user, who can choose to expand the results of a search by recursively looking into links to other pages without actually retrieving the full documents.

The running version of the prototype constitutes an initial step of the implementation process, in which a Web indexing robot was implemented, and the Web-Database communication and Java-based mobile code interfaces were explored and tested. The second step, under development, includes the investigation, implementation and testing of cooperating modules running in heterogeneous platforms in an open distributed architecture.

The platform chosen for communication between modules is Java RMI [Sun, 1996]. To provide for independence from a particular DBMS, ODBC [Microsoft, 1994] was the framework chosen for interaction between the application and the DBMS. To account for interoperability, ROJ [Sommers, 1996] is under consideration, as it provides integration between RMI and CORBA [Mowbray & Zahavi, 1995].

## 6. Relation to other Systems and Conclusion

BRight! and its Web Views concept were designed to deal with the specific problem of information indexing and retrieval in the Web. It shares structural characteristics and operational goals with the other NIR systems outlined in Section 2. However, the Web Views concept allows for the definition of a consistent framework for modules cooperation and resource sharing, while using meaninful parameters for the specific problem of the Web. Another explicit concern in BRight! regards interoperability with other systems, as described in Section 5.

The architecture presented in this paper and its underlying concept of Web Views provide a means to deal with the scale problem of Web size in the tasks of indexing and searching for information. In this paper, we have shown how cooperating modules are able to index limited portions of the Web more efficiently, and also how they exchange indexes when needed. As a consequence, users, servers and indexing services can save local computational resources, and network bandwidth (a more global/regional resource) is also saved.

By working cooperatively, a community of specialized modules introduces a higher level of abstraction to the problem of locating information in the Web. On top of a network of information (the Web), a network of indexes working cooperatively and consistently, in an open distributed framework, constitutes an efficient way to deal with the problem of information indexing and searching in the Web.

[3] This is a provisional url. By the time of the conference, the definitive site should be available at http://www.bright.org.br .

## References

Admedia, "Size of the Internet (Worldwide)", January 1997 (http://www.admedia.aust.com/ws-4.htm).

Altavista (http://www.altavista.digital.com).

Bell,G.; Semmell,J. "On-Ramp Prospects for the Information Superhighway Dream". *Communic. of the ACM*, 39(7):55-61, 1996.

Berghel,H. "The Client's Side of the World-Wide Web". *Communic. of the ACM*, 39(1):30-40, 1996.

Boles,D.; Dreger,M.; Grossjohann,K. "MeDoc Information Broker - Harnessing the Information in Literature and Full Text Databases". In *Proc. of the Workshop on Networked Information Retrieval*, Zurich, Aug. 22, 1996.

Brown,P.J. "Linking and Searching within Hypertext". *Electronic Publishing*, 1(1):45-53, 1988.

Buckland,M. "Searching Multiple Digital Libraries: A Design Analysis". Digital Library Project, School of Information Management and Systems, Univ. of California, Berkeley, CA, 1995 (http://www.sims.berkeley.edu/research/oasis/multisrch.html).

Buckland,M.; Butler,M.H.; Norgard,B.; Plaunt,C. "OASIS: Prototyping Graphical Interfaces to Networked Information". *Proc. 56th Annual Meeting of the American Society for Information Science*, Medford, NJ, pp. 204-210, 1993.

Callan,J.P.; Lu,Z.; Croft,W.B. "Searching Distributed Collections with Inference Networks". In *Proc. 18th Annual International ACM SIGIR Conf. on Research and Development in Information Retrieval*, pp. 21-28, Seattle, WA, 1995 (http://www.di.ufpe.br/~bright/artigos/FTSIRetrieval/Ca llanEtAl95.ps.gz).

Campbell,K. "Understanding and Comparing Web Search Tools". 1996 (http://www.hamline.edu/library/links/comparisons.html).

Cooper,I. "Indexing the World", July 1994. Computing Laboratory, University of Kent, Canterbury, Kent, ENGLAND (http://stork.ukc.ac.uk/computer_science/Html/Pubs/IHC10-94).

Connolly,D.; Soley,R.M. (Eds.) **Joint W3C/OMG Workshop on Distributed Objects and Mobile Code**. Sponsored by the World Wide Web Consortium and the Object Management Group. Boston, MA, 1996 (http://www.w3.org/pub/WWW/OOP/9606_Workshop).

Crowder,G.; Nicholas,C. "Resource Selection in CAFE: an Architecture for Networked Information Retrieval". In *Proc. of the Workshop on Networked Information Retrieval*, Zurich, Aug. 22, 1996.

Edwards,N. "CORBAweb - A CORBA Gateway for the Web". In Connolly,D.; Soley,R.M. (Eds.) *Joint W3C/OMG Workshop on Distributed Objects and Mobile Code.* Boston, MA, 1996 (http://www.w3.org/pub/WWW/OOP/9606_Workshop/submissions/16-wrkshp.html).

Excite (http://www.excite.com).

Flanagan,D. **Java in a Nutshell - a Desktop Quick Reference for Java Programmers. Cambridge**, O'Reilly & Associates, Inc., 1996.

Frakes,W.B. "Introduction to Information Storage and Retrieval Systems". In Frakes,W.B.; Baeza-Yates,R. (Eds.) **Information Retrieval - Data Structures and Algorithms.** Englewood-Cliffs, NJ, Prentice-Hall, pp. 1-12, 1992.

Fox,E.A. "Rethinking Libraries in the Information Age: Lessons Learned with Five Digital Library Projects". School of Information & Library Science, UNC Chapel Hill, 1996 (http://fox.cs.vt.edu:80/talks/UNC96).

Führ,N. (Ed.) **Networked Information Retrieval Workshop**, SIGIR96 - ACM SIGIR Conf. on Research and Development in Information Retrieval. Zurich, Switzerland, August 18-22, 1996.

Go2Net (http://www.metacrawler.com).

Gonçalves,P.F. "Uma Arquitetura Distribuída Baseada em Código Móvel para Pesquisa e Recuperação de Informações em World-Wide Web", PhD working plan, October, 1996 (http://www.di.ufpe.br/~bright/publications/plano96.html).

Gonçalves,P.F.; Meira,S.L.; Salgado,A.C. "A Distributed Mobile Code-Based Architecture for Information Indexing, Searching and Retrieval in the World-Wide Web. *Proc. 7th Annual Conf. of the Internet Society* (INET'97). Kuala Lumpur, Malaysia, June 1997a (http://www.di.ufpe.br/~bright/publications/inet97.html).

Gonçalves,P.F.; Salgado,A.C.; Meira,S.L. "Digital Neighbourhoods: Partitioning the Web for Information Indexing and Searching". In Olivé,A., Pastor,J.A. (Eds.) *Advanced Information Systems Engineering, 9th International Conference* (CAiSE'97), Barcelona, Catalonia, Spain. Springer Verlag, Lecture Notes in Computer Science 1250, pp. 289-302, June 1997b (http://www.di.ufpe.br/~bright/publications/caise97.html).

Koster,M. "Database of Web Robots, Overview", March 1997 (http://info.webcrawler.com/mak/projects/robots/active/html/index.html).

Sullivan,D. "How Search Engines Work", 1996 (http://calafia.com/webmasters/work.htm).

Merle,P.; Gransat,C.; Geib,J.-M. "CorbaScript and CorbaWeb: A Generic Object-Oriented Dinamic Environment upon CORBA". Technical Report URA CNRS 369, Laboratoire d'Informatique Fondamentale de Lille, 1995.

Microsoft, "Microsoft Index Server Guide", 1997
(http://www.microsoft.com/ntserver/search/docs).

Microsoft Corporation, **Microsoft ODBC 2.0 Programmer's reference and SDK Guide**. Microsoft Press, 1994.

Mitchell,S. "General Internet Resource Finding Tools: a Review and List of those used to Build Infomine". Bio-Agricultural Library, Univ. of California, Riverside, CA, 1996.

Mowbray,T.J.; Zahavi,R. **The Essencial CORBA - Systems Integration using Distributed Objects**. New York, John Wiley & Sons, Inc., 1995.

Muscat, "Search Engines - Euroferret", 1997 (http://www.muscat.co.uk/euroferret).

NWI, "Nordic Web Index", 1997 (http://nwi.ub2.lu.se/?lang=uk).

Open Text, "Livelink Search Overview", 1997
(http://www.opentext.com/livelink/ll_search.html).

Schwartz,M. (Ed.) **Report of the Distributed Indexing/Searching Workshop, Sponsored by the World Wide Web Consortium**, Cambridge, Massachusetts, May 28-19, 1996 (http://www.w3.org/pub/WWW/Search/9605-Indexing-Workshop).

Smeaton,A.F.; Quigley,I. "Experiments on Using Semantic Distances Between Words in Image Caption Retrieval". In *Proc. of the 19th Annual International ACM SIGIR Conf. on Res. and Dev. in Information Retrieval*. pp. 174-179, Zurich, Switzerland, August 18-22, 1996.

Sommers,B. "Distributing Java: Remote Objects for Java". 1996
(http://www.javaworld.com/javaworld/jw-06-1996/jw-06-remote.objects.html).

Sun Microsystems, "Java(tm) Remote Method Invocation Specification". 1996
(http://phanouri.bevc.blacksburg.va.us/ROJ/doc/rmi-spec/rmiTOC.doc.html).

Tai,A. "MAUV - Metasearcher at the University of Virginia", April 1996
(http://www.cs.virginia.edu/~act9m/mauv/help.html).

Tanenbaum,A.S. **Computer Networks**. McGraw-Hill, 1992.

Voorhees,E. "Siemens TREC-4 Report: Further Experiments with Database Merging".
In *Proc. Text REtrieval Conference (TREC-4)*, 1995.

Voorhees,E.; Gupta,N.K.; Johnson-Laird,B. "The Collection Fusion Problem". In *Proc. Text REtrieval Conference (TREC-3)*, Gaithersburg, MD, 1994
(http://www.di.ufpe.br/~pfg/artigos/FTSIRetrieval/VoorheesEtAl95.ps.gz).

Yahoo! (http://www.yahoo.com).