

PEER-TO-PEER SEARCH ENGINE: THE ARAIGNEE PROJECT

By
Pierre Poignant

Diplôme d'Ingénieur, Ecole Polytechnique, France, 2001

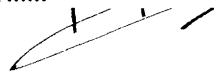
Submitted to the Department of Civil and Environmental Engineering
In Partial Fulfillment of the Requirements for the Degree of

Master of Science in Civil and Environmental Engineering

at the
Massachusetts Institute of Technology
June 2003

© 2003 Massachusetts Institute of Technology
All Rights Reserved

Signature of Author.....



.....
Department of Civil and Environmental Engineering
May 9, 2003

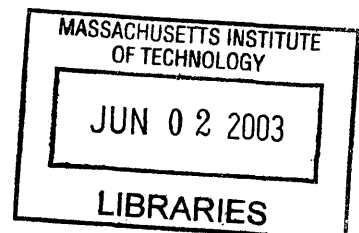
Certified by.....

.....
Steven R. Lerman
Professor of Civil and Environmental Engineering
Thesis Supervisor

Accepted by.....

.....
Oral Buyukozturk
Professor of Civil and Environmental Engineering
Chairman, Departmental Committee on Graduate Studies

BARKER



Abstract

Fast Peer-To-Peer Search Engine The Araignee Project

By
Pierre Poignant

Submitted to the Department of Civil and Environmental Engineering

On May 9, 2003

In Partial Fulfillment of the Requirements for the Degree of
Master of Science in Civil and Environmental Engineering

According to a survey by Yankee Group, 50% of the users search for data every day. As a result, search engines are of utmost importance to Internet users. Currently, conducting an online search is a very frustrating experience as one is often unable to find exactly what one is looking for from the thousands of webpages. This can be attributed to the inadequacies of existing search tools such as Internet directories and search engines. In particular, existing search tools lack a “collective memory” functionality that allow a user who is searching for the same item as a previous user to make use of the latter’s search results.

I propose creating a new tool, “Araignee” for Internet users to simplify their search by developing a new methodology of web archiving that will capitalize on the experience of every user and personalize the search. This proposed tool has the potential to create a new kind of collaboration between web users.

Thesis Supervisor: Steven R. Lerman
Title: Professor

Acknowledgements

First I would like to thank my advisor Professor Steve Lerman for the many enriching discussions we have had and for his invaluable advice in the writing of this thesis. More generally, I would like to thank the MIT Center for Educational Computing Initiatives and the TEAL Project for their financial support that enabled me to further my studies at MIT.

I would also like to thank: My family for their constant support, Helene for always smiling, Sophie for being so lively, Alex for our (in every sense) exciting conversations, Ardoin for being my roommate, Brice for being Brice, the winter for being such a new experience, Lolo for being the master techos, Julien, Yann and Matthieu for the olaloo spirit, My cat, The music for keeping me out of the noisy world and finally adversity for being such a good teacher.

And most importantly I would like to give a special thank to my fabulous girlfriend Eeleen for always being a source of inspiration. She made my life at MIT an unforgettable adventure.

Now, most people do not stop to read the acknowledgements. Since you did, here's a little bonus, come to Paris and you will get a free and nice French dinner from me!

Table of contents

1. The Market	8
1.1. Internet - A huge market.....	8
1.2. Search Engines: A feature the Internet can't do without.....	9
1.3. Peer-To-Peer.....	10
1.4. Conclusion from the Market Research.....	12
2. The Services Provided by Araignee	13
2.1. The core services.....	13
2.2. Company and Community Applications.....	15
2.2.1. Companies.....	15
2.2.2. Communities.....	15
2.3. Single User Applications.....	15
2.3.1. A personal space.....	16
2.3.2. A personalized propositions space.....	16
2.3.3. The best of the web.....	16
2.3.4. A search engine.....	16
2.3.5. A content filter.....	17
2.4. Potential of Araignee.....	17
3. The Competitors	18
3.1. The Search Providers.....	18
3.2. Search Engines.....	19
3.2.1. The Technology.....	19
3.2.2. The Incumbents.....	21
3.3. Web Directories.....	23
3.4. New concepts.....	23
3.4.1. Peer-To-Peer.....	23
3.4.2. Linguistic Analysis.....	23
3.4.3. Graphical User Interfaces.....	24

3.5.	The Newcomers	24
3.5.1.	Exalead.....	24
3.5.2.	Mapstan.....	25
3.5.3.	Kartoo	27
3.5.4.	AltaVista Peer-To-Peer.....	28
3.5.5.	Human Links.....	28
3.5.6.	Others.....	29
3.6.	Conclusion	30
4.	The Technology Behind Araignee	31
4.1.	Overview.....	31
4.2.	The Proxy Server	32
4.2.1.	The Proxy.....	32
4.2.2.	The Web Server	33
4.2.3.	The Search Component.....	33
4.3.	The Proxy Server Architecture	34
4.3.1.	The HTTP proxy server	35
4.3.2.	The request server.....	36
4.3.3.	The filter technology.....	38
4.3.4.	The transducer_filter.....	38
4.4.	The spider_filter.....	38
4.4.1.	Lex - A Lexical Analyzer Generator	38
4.4.2.	The Database.....	39
4.5.	The Peer-To-Peer Server.....	40
4.5.1.	The JXTA Framework	40
4.5.2.	The Peer-To-Peer Structure	41
4.5.3.	The Communication Protocol	43
4.5.4.	The PipeServer.....	44
4.5.5.	The Message Store.....	45
4.5.6.	Message Routing Protocol	45
4.6.	Deployment of the Software	46
5.	Discussion.....	48

5.1.	Scalability	48
5.1.1.	Size of the Database.....	48
5.1.2.	The Routing Protocol.....	48
5.1.3.	Conclusion	49
5.2.	Reliability.....	49
5.2.1.	Crash Resistance	49
5.2.2.	Database Integrity.....	50
5.2.3.	Results Accuracy	50
5.3.	Privacy	51
5.3.1.	Routing Protocol.....	51
5.3.2.	Privacy Statement	51
5.3.3.	Open Source.....	52
5.4.	Profiling / Personalization.....	52
5.4.1.	Collaborative Filtering.....	53
5.5.	Human Computer Interaction	54
6.	Conclusion	55
7.	References.....	56

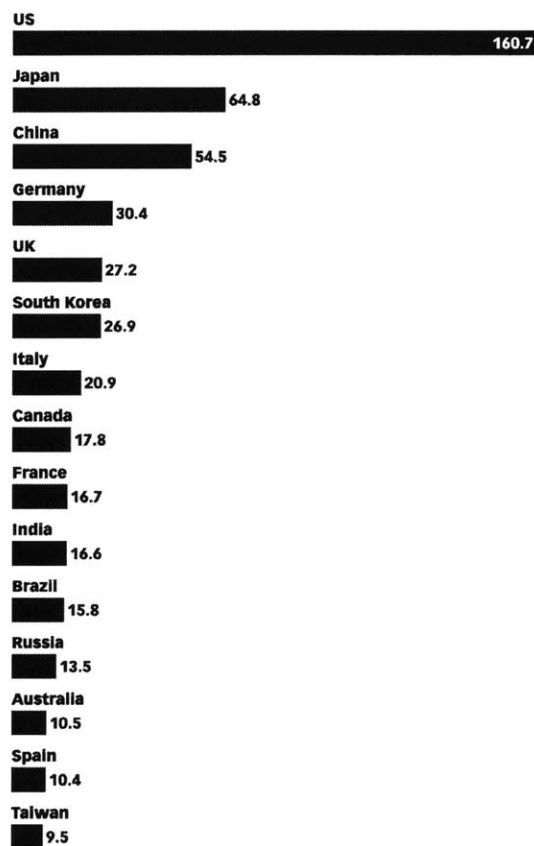
List of figures

Figure 1: Top 15 Countries Ranked by number of Internet Users.....	8
Figure 2: Broadband and dialup subscribers in the US.....	9
Figure 3: Online Activities in Europe (as %of internet users daily).....	10
Figure 4: Architecture Of Peer-To-Peer Technology	11
Figure 5: Use of peer-to-peer in the US.....	11
Figure 6:Graphical illustration of the concept of Araignee	14
Figure 7: Web of Search Providers (Source: Salomon Smith Barney).....	18
Figure 8: Map of the Internet.....	20
Figure 9: Search Engine Audience Home & Work Users January 2003	21
Figure 10: Average Minutes Spent Searching Per Visitor January. 2003	22
Figure 11: Exalead Graphical User Interface.....	25
Figure 12: Mapstan Graphical User Interface.....	26
Figure 13: Kartoo Graphical User Interface	27
Figure 14: Human Links Graphical User Interface.....	29
Figure 15: Araignee Software Architecture.....	31
Figure 16: The proxy server architecture.....	34
Figure 17: Example of an indexed Tree Structure	40
Figure 18: The JXTA architecture (JXTA Programmer Guide).....	41
Figure 19: The GUI Server Architecture	42
Figure 20: Routing Example.....	46
Figure 21: How Collaborative Filtering Works.....	53
Figure 22: Araignee Recommender System	54

1. The Market

1.1. Internet - A huge market

Today, based on the findings in Figure 1, it is estimated that there are over 665 million Internet users worldwide. By 2005, it is expected that this figure will reach 1 billion. Of these, over 80% will be English speaking.



Note: Top 15 total=496.0; Worldwide total=665.9
Source: eTForecasts, December 2002

Figure 1: Top 15 Countries Ranked by number of Internet Users

Currently, most of the Internet users in the US are connected via the traditional phone-line modem. Broadband Internet is however increasing in popularity among the Internet users. With reference to Figure 2, whilst only 20% of the Internet users in the US to-date have broadband connections, this figure is estimated to climb beyond 50% by 2005. In

fact, it is expected that broadband connection will gradually become the standard form of Internet access in many countries.

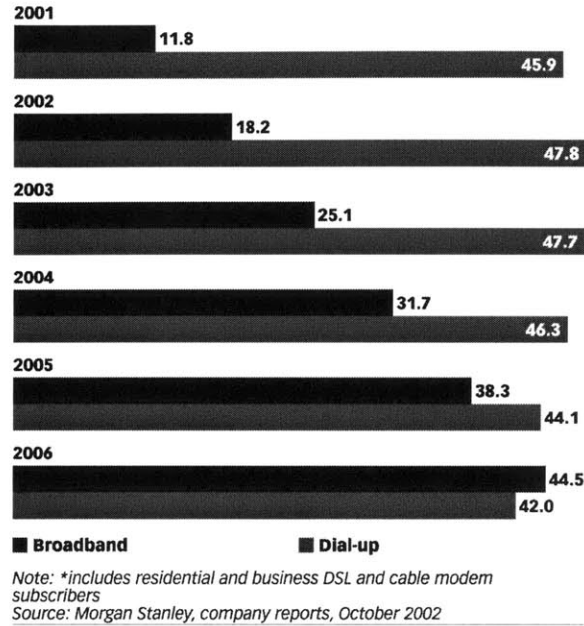
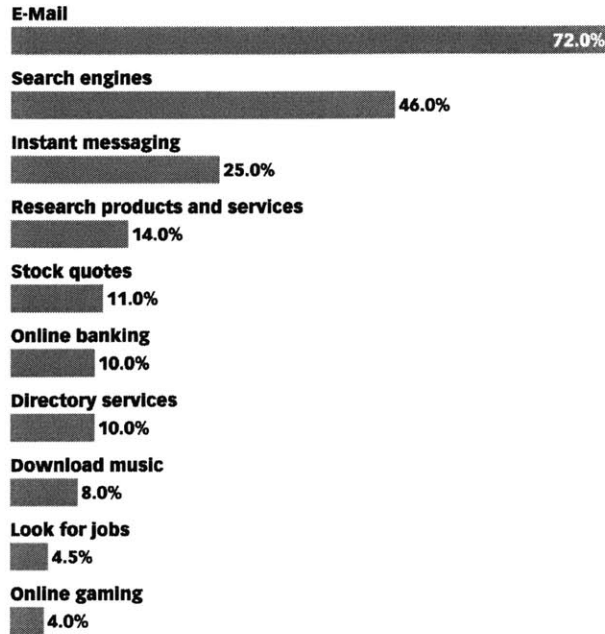


Figure 2: Broadband and dialup subscribers in the US

1.2. Search Engines: A feature the Internet can't do without

Based on the findings in Figure 3, 50% of Internet users use a search engine daily. Given the information overload that currently exists, it is unsurprising that searching for information on the Internet is one of the main activities of Internet users.



Source: Yankee Group, December 2002

Figure 3: Online Activities in Europe (as %of internet users daily)

1.3. Peer-To-Peer

By supporting real-time communications, visibility between end-users, and resource sharing among PCs, peer-to-peer systems has expanded Web functionality significantly. A very good overview of peer-to-peer systems and applications can be found in [Miller 2001].

There are typically three different implementations of peer-to-peer technology. These include (i) autonomous networking that is completely decentralized (e.g. Gnutella and Freenet), (ii) distributed computing that is tightly organized around a hub (e.g. SETI) and (iii) coordinated computing that has a hybrid architecture (e.g. Napster and AIM). The proposed new tool, Araignee, will fall under the last category.

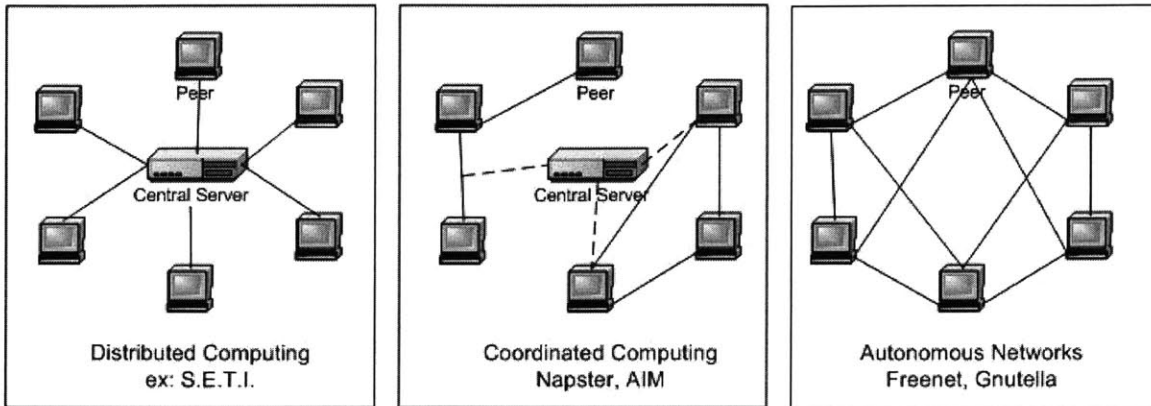


Figure 4: Architecture Of Peer-To-Peer Technology

Currently, enterprises solutions implementing peer-to-peer are not widespread. However, Frost And Sullivan have forecasted a very steep growth in the use of the peer-to-peer technology in the next 5 years as indicated in Figure 5 below:

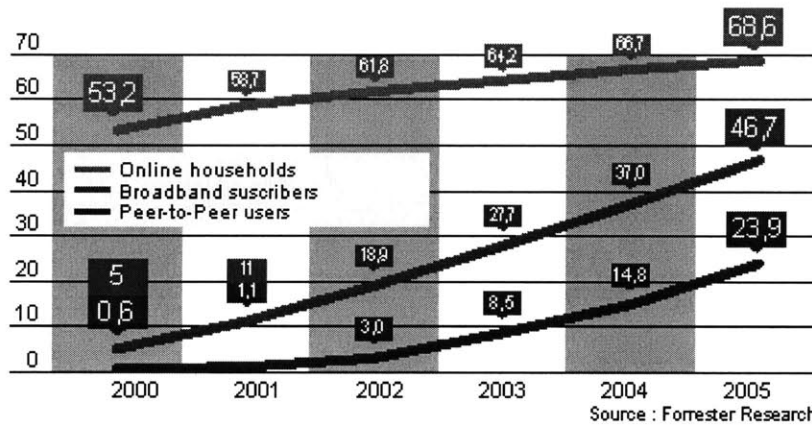


Figure 5: Use of peer-to-peer in the US

From Figure 5, the diffusion of peer-to-peer usage highly correlates to the diffusion of broadband Internet. This is largely due to the fact that peer-to-peer systems thrive on an environment where the user is connected to the Internet 24x7, and broadband connection helps create the conditions for the development of such an environment.

1.4. Conclusion from the Market Research

Both the importance of search tools to Internet users and the predicted widespread adoption of broadband technology create excellent conditions for a peer-to-peer search engine. In addition, with peer-to-peer technology becoming a familiar concept to many (due in part to the proliferation of applications such as Napster), there is clearly a spot in the market for an innovative peer-to-peer search engine.

The subsequent sections detail the functionalities of Araignee as well as its competitors. This review serves to illustrate the originality of Araignee.

2. The Services Provided by Araignee

2.1. The core services

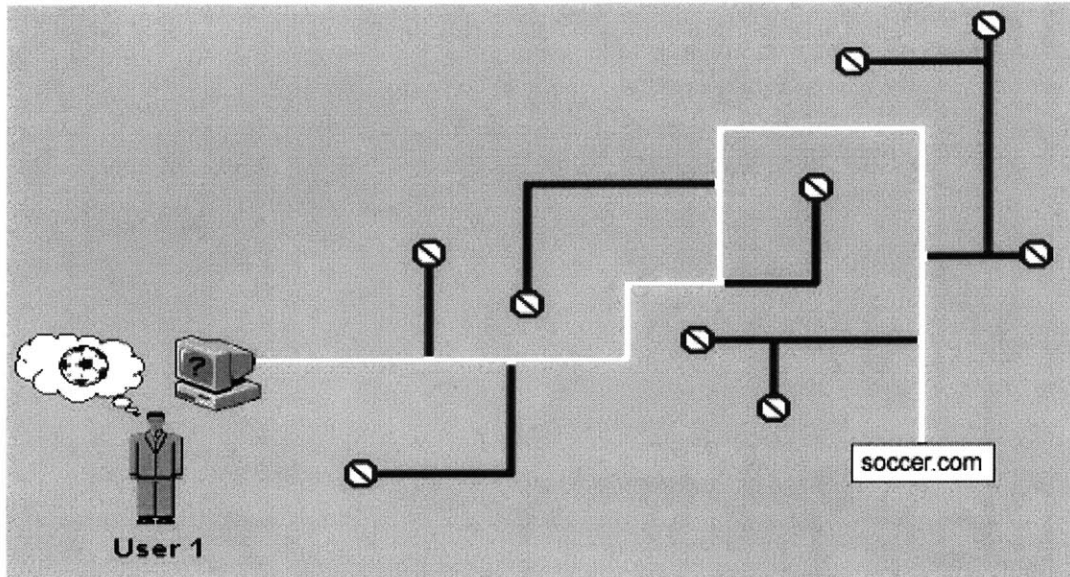
The number of webpages is growing at an exponential pace. However, of these, only a small fraction are what many users view as “quality” webpages. Currently, web directories are unable to go through all the websites, while search engines such as Google do not take into account human assessment of websites. These inadequacies therefore lead to search results that are often less than satisfactory. The proposed search engine on the other hand can overcome these shortcomings and provide an invaluable resource to users.

With Araignee, the user installs a software on his personal computer, which will then be used to monitor the user’s surfing activities. In particular, the software records the pages visited by the user as well as the frequency with which the user visits these pages. Hence, this tool makes use of the frequency with which an individual visits a particular website to indirectly infer the quality of the website. In other words, the more frequently a website is accessed, the more likely it is a quality website. This is a reasonable assumption because the more frequently a website is visited by the user, the more likely this website satisfies the user’s needs and, by extension, is a quality website.

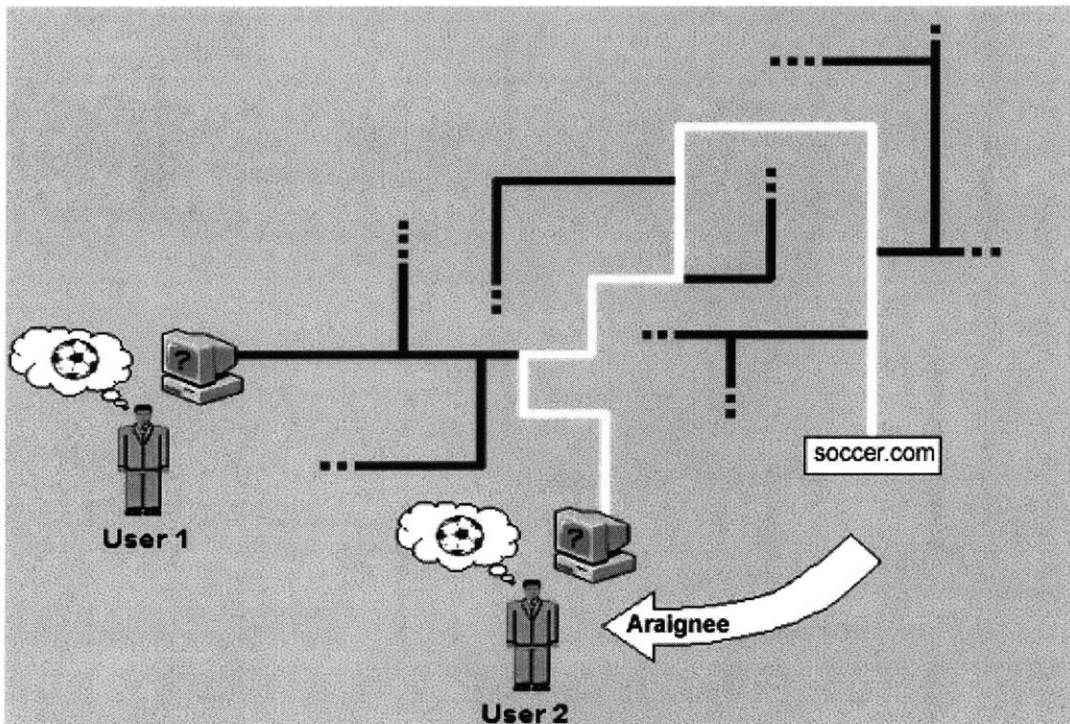
Using the data collected on a user’s surfing activities, Araignee can create profiles for each of its users. Profiles for users can be created based on their interests by studying the web sites they visit as well as the keywords that appear. The more a user surfs, the more precise will be his profile. Eventually, clusters of users with similar profiles can be formed.

Araignee also offer users the opportunity to benefit from a group experience by allowing users to search in group-databases. This means that someone who wishes to know more about C# will be able to know which websites are most frequently visited by others interested in the topic. Similarly, users will be able to join groups to share their search results. For example companies can group their employees such that when someone from a particular group needs information on topic X, he will be able to quickly know which websites other members in his group frequently visit sites pertaining to topic

X. Not only can this help to reduce the search time significantly, it can also help to enhance knowledge sharing in the company.



A first user searches about a theme. After a few useless websites, he finally finds the useful information.



A second user searches about the same theme. Araignee capitalizes on the experience of the first user and is able to immediately provide the relevant websites to the second user.

Figure 6: Graphical illustration of the concept of Araignee

2.2. Company and Community Applications

2.2.1. Companies

Araignee can be very useful to companies. Nowadays companies need to put in more effort to capitalize on the knowledge of employees, and Araignee can serve as an invaluable resource in this regard. Indeed, Araignee enables the employees of a company to benefit from the web searches of all the other employees. Araignee also enables employees to participate in the capitalization of knowledge without adding to their workload. In this respect, Araignee can be considered to be a very useful knowledge management tool. Furthermore, Araignee can provide information on the searches carried out by employees on the company's Intranet. By using these statistics to make improvements to its Intranet, a company can then optimize its information system.

2.2.2. Communities

Araignee services are also very well designed for communities sharing the same interests. Indeed the search engine will be very efficient if the users have the same kind of profiles. By profile, we refer to the interests and surfing habits of the user. In the current state of the Araignee project, the user is assumed to choose his own interests and hence create his own profile. However, as we will examine in section 5.4, Araignee will also have the capability to create a user's profile from scratch by comparing his surf habits with other users' surf habits.

2.3. Single User Applications

Araignee is based on a revolutionary principle, i.e. to enable a web user to benefit from the experience and the searches of other web users.

Through analyzing the surfing habits of a user, Araignee can pinpoint a user's interests and favorite websites. Personalized webpages corresponding to the user's expectations could then be created automatically for every user by tapping on the

experiences / searches of other Araignee users who share similar interests or profiles as the user.

The following sections detail the various applications that will prove useful to an Araignee user.

2.3.1. A personal space

Links to the websites most frequently visited by a user would be displayed in this space. These links can be classified according to the user's interest or some other manner depending on the user's preference. This would allow a user to access his favorite websites much more quickly without having to tediously create bookmarks on his browser.

2.3.2. A personalized propositions space

The links of the websites most frequently used by other web users for the subjects listed in the user's personal space would be displayed in this space. Thanks to Araignee's ability to generate a user's profile based on his surfing activities, users would be able to see the favorite websites of other web users with similar profiles as themselves, a la "knowledge sharing" in the true sense of its word. Forum propositions and newsletters of possible interest to the user can also be displayed in this section.

2.3.3. The best of the web

This section would display websites that are most often used by Internet users and hence deemed to be successful or popular websites. Websites with the greatest traffic growth could also be displayed in this section, so as to inform users the "in" thing on the web.

2.3.4. A search engine

By enabling users to share their search result without any action from their part, Araignee creates a new kind of passive collaboration. Armed with the ability to analyze the surfing data of users, Araignee can display the pages that are most visited and hence deemed to be most relevant to someone conducting a search on a particular subject. As more and more users get on the system, the results will be correspondingly more refined,

leading to the creation of a dynamic search engine that continuously gets better using the cumulative data about the surfing choices of its users.

2.3.5. A content filter

As Araignee functions as a proxy and analyzes the content of a webpage, it can therefore act as a content filter and block out undesirable webpages contained in a list. This list can be stored on a central server and be updated continuously by users themselves. As such, Araignee can even double up as a parental control filter or even an advertisement blocker.

2.4. Potential of Araignee

The greatest benefit Araignee yields is that of the community search engine. As indicated in Section 2.1, Araignee is a statistics-driven tool which makes use of the frequency with which an individual visits a particular website to infer the quality of the website. This information is then used to improve the surf experience of an individual or of other individuals through a variety of ways. Future collaborative content management systems are expected to be modeled in this manner.

3. The Competitors

Araignee is a community tool centered on web searches. Its use and design are different from a typical search engine. Nevertheless, some of the functionalities offered by Araignee do overlap with those offered by search engines and web directories. It is therefore necessary to understand the history and development of these other tools.

3.1. The Search Providers

A user makes use of a search provider to search for information or websites on the Internet. The search provider in turn extracts the search results from paid listings (where companies pay to be listed), from editorial listings (where the editors of the search provider index the various websites) or from traditional search engine (where robots crawl the Internet to look for the required information).

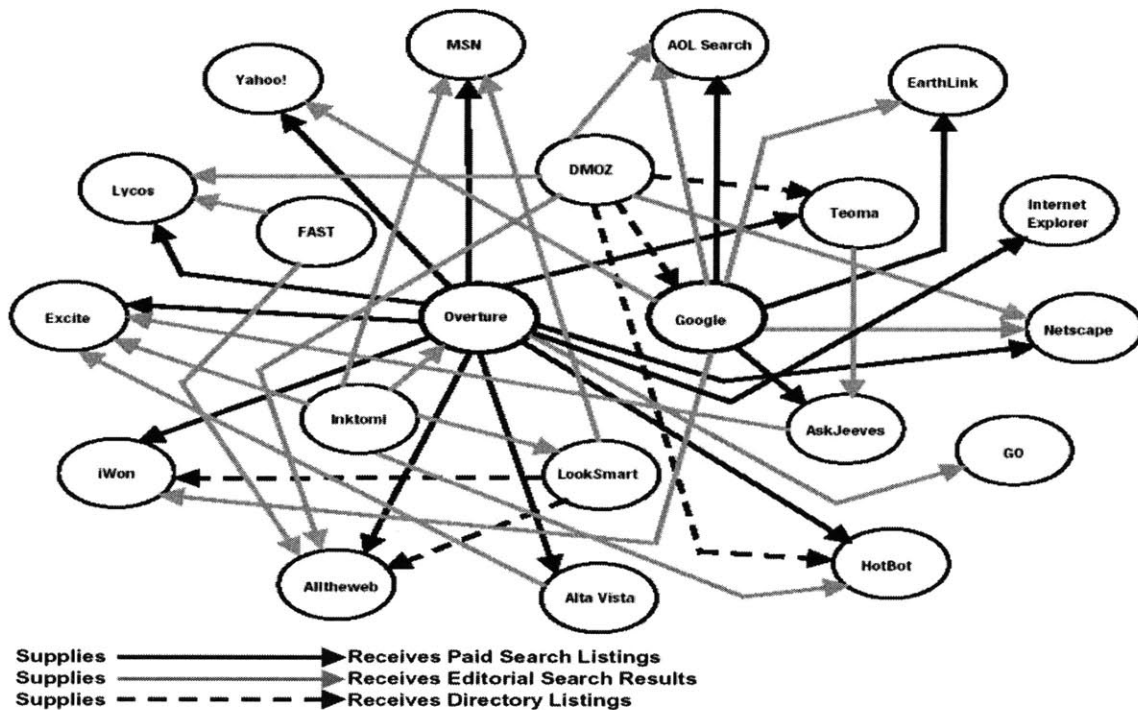


Figure 7: Web of Search Providers (Source: Salomon Smith Barney)

The various search providers exchange listings using their respective search methods. Among them, Google, Teoma, Inktomi are the traditional search engines; Overture

provides paid listings whilst DMOZ is a traditional web directory. In spite of the complex market relations amongst the various search providers, the search results yielded can nonetheless be divided into two main categories, namely search engine results and directory results. In the former case, the results are automatically generated; as for the latter case, the results are indexed manually. Araignee goes one step further by combining both of these approaches, i.e. automatic indexation and human judgment. The following sections examine the current search engine and directories market and show where Araignee fits into the current scheme of things.

3.2. Search Engines

3.2.1. The Technology

A search engine functions with two main components, a crawler and an indexer. The crawler surfs the web from website to website by following the links between the websites. The pages detected are then passed to the indexer. Using statistical analysis, the indexer analyses the content of the page, such as the language used. The words in the webpage are then classified, with greater importance given to certain words such as those contained in the title or those repeated many times in the webpage.

With this technique, a lot of problems can arise, for instance spam indexing by webmasters. In such situations, “rogue” webmasters who are able to decipher the algorithms used to classify and rank webpages, will, in their attempts to push up the ranking of their webpage, make modifications to the content of the webpage so that it ends up highly ranked by the search engine. In actual fact however, such webpages are totally irrelevant to the user. More details about the algorithms used by search engines to classify and rank webpages can be found in [Marckini 2001]. Although search engines have developed numerous methods to overcome problems such as spam indexing, ultimately, search engines can only do a statistical analysis of words. The search engines cannot for instance analyze or grade the content of the webpage. Hence, as the number of webpages on the Internet increases, the accuracy of the results generated by search

engines will almost certainly decrease. Araignee overcomes this problem by making use of the human factor in the indexing of the webpages.

Another problem that can arise is related to the inherent repartition of the Internet according to the links between the webpages. This results in search engines not being able to automatically go through all the available webpages.

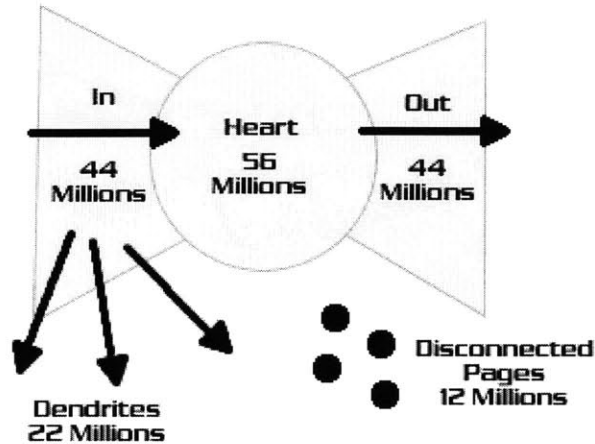


Figure 8: Map of the Internet

The 200-plus million webpages on the Internet can be divided among five categories [S&V 2000]. The 56 million heart pages are interconnected and linked to one another. There are 44 million pages that point to the heart, and 44 million pages that are linked from the heart. Another 12 million are disconnected pages that belong mostly to Intranets.

Whilst crawlers can easily index the heart, the rest of the Internet is difficult to reach. Hence, human intervention is needed to index most of these other pages. Araignee can provide a seamless solution to this problem.

3.2.2. The Incumbents

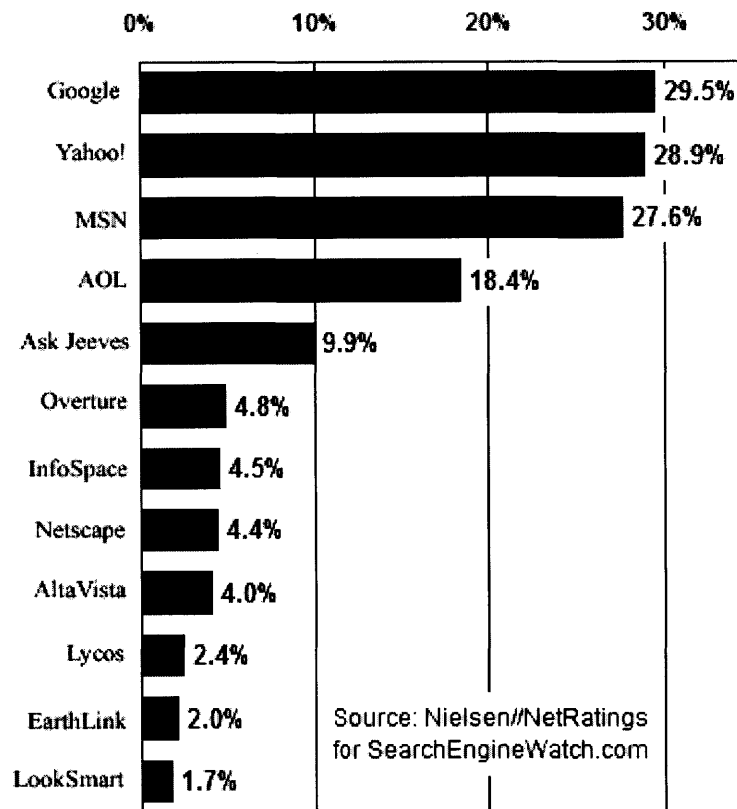


Figure 9: Search Engine Audience Home & Work Users January 2003

AltaVista was the first famous search engine of the web. Unfortunately, its leadership position has since been eroded, and the likes of Google looks set to be the next “big thing” in the field of search engines. MSN on the other hand is the search engine of Microsoft, the most commonly used operating system today. By building this search engine as part of the Internet platform, Microsoft has managed to maintain MSN’s foothold in the world of search engines.

Google was launched in 1998 amid little fanfare and with very little funding. But Internet users have been rapidly seduced by its speed and simplicity. The principle of Google is to detect the number of links pointing to another page. This allows Google to compute the popularity of a site and to classify it in its list of results without any human intervention. In fact, such is the attraction of Google’s technology that they have been able to sell their technology to web directories like Yahoo!.

Today, Google can be said to be the market leader, with their technology being used by 60% of the users (if we include the use of Google on Yahoo!). However, one problem associated with the Google technology is fast becoming evident – with the ever increasing number of webpages that need to be indexed, the indexation process is becoming very complex. This has slowed down the performance of the Google search engine considerably as illustrated in Figure 8.

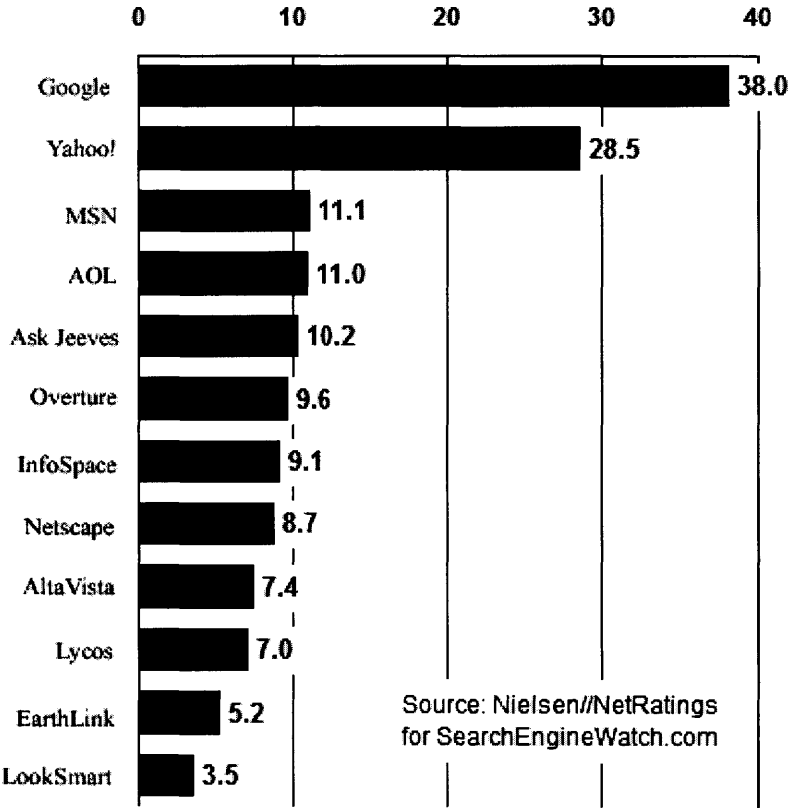


Figure 10: Average Minutes Spent Searching Per Visitor January, 2003

A further weakness associated with Google is that given its current leadership position, it would be less receptive to making changes in its technology than a newcomer to the scene, as the associated risks would be far too high. This could potentially stifle any technological evolution.

3.3. Web Directories

Web directories can be said to be the indirect competitors of Araignee. Users can either search using a traditional search engine or the directory resources. However whilst web directories are very popular, they are being used as search tools less and less often. This is because the web site indexation is done by humans, and this cannot keep up with the exponential growth of the Internet content. In fact, companies offering web directories such as Yahoo understand this limitation and are instead adopting a strategy of diversifying their operations to offer a huge array of services so as to attract the users to stay on the directory pages.

3.4. New concepts

Overall, the search engine market has been static for the past few years, with the greatest contribution made by Google. However, more and more offerings are emerging in this market that could change everything in the next few years. In particular, the latest technologies developed in other fields and which simplify and speed up searches significantly are being used for new search engines. These are detailed below.

3.4.1. Peer-To-Peer

This technique enables users to share files stored on different computers and is therefore well-designed for information, documents or files search. A handful of new search engines currently use or partially use this technique to share files or web addresses. In fact, this technique has been made extremely popular by the prolific musical files exchange via Napster.

3.4.2. Linguistic Analysis

Search engines use key word matching to find the pages corresponding to a user's query. But this approach is sometimes inefficient. Indeed words can have different meanings; they can be used in different contexts, and only with a precise analysis of the

context can we determine if the page will satisfy the user. Some search engines therefore make use of linguistic analysis for greater efficiency. For instance, if one sends out a query for web pages containing the words “recycling cars”, he would likely be interested in pages containing the words “recycle car” as well. Linguistics search engines analyze the etymology of the words in the query and display pages containing all the relevant combination of the results instead of just what the user typed in. This can thus help to save a lot of time for the user.

3.4.3. Graphical User Interfaces

Search engines and web directories present their results in the same way: the user query is entered in a form field, and the answer is given as a list of web sites classified by accuracy one after another. Some recent search engines organize their results in an altogether different format: the web sites are presented graphically on a map. The search is therefore more intuitive and can be refined easily since the links between the different websites are displayed. This translates to a more convivial, less discouraging and faster search for the user.

3.5. The Newcomers

3.5.1. Exalead

Website: <http://www.exalead.com/>

Exalead is a new search engine developed by Francois Bourdoncle and a team from Ecole Polytechnique. This search engine integrates a linguistic module that enables a greater flexibility during the indexation of a web page. Exalead recognizes words or verbs with different forms, regardless of the orthography or the conjugation. Indeed the search engine detects the root of the keyword and is therefore able to find pertinent pages even if they do not include the matching word.

The strategy of Exalead is to develop partnerships with existing portals or search engines. The incorporation of Exalead technology on AOL.fr led to a growth from 1% to 6% of the market for the portal search engine.

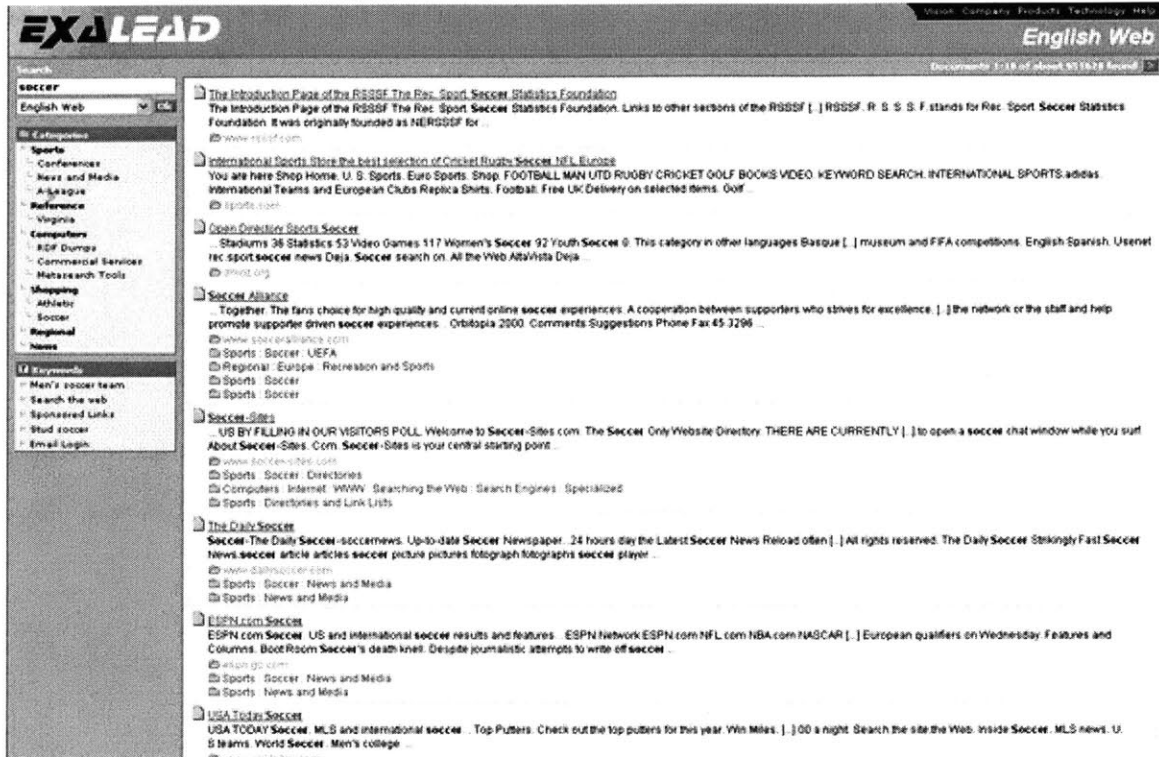


Figure 11: Exalead Graphical User Interface

3.5.2. Mapstan

Website: <http://www.mapstan.com/>

Mapstan is a meta search engine that displays the results of its searches as a map as indicated in Figure 12. The best result is numbered 1. The other results are either presented in the same circle if they are very close, or in other circles linked to the first one according to the traffic between those two websites. As for the blue circles, the farthest away propose websites that would fit the user profile.

There are two ways of using Mapstan:

- 1) Using it directly as a traditional search engine. In this case however, the maps do not evolve.
- 2) Downloading a client software plug-in that will make the personal map evolve according to the websites the user visits. Although this functionality is very close to the Araignee project, Araignee provides a lot more functionalities.

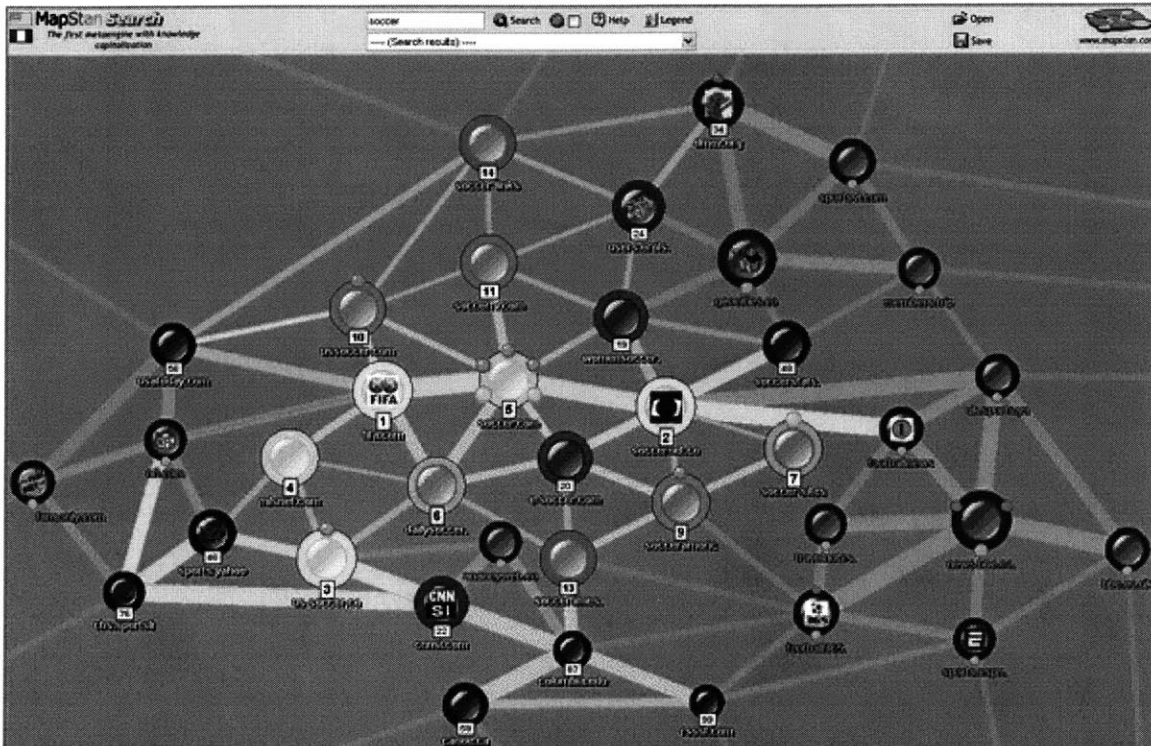


Figure 12: Mapstan Graphical User Interface

Strengths of Mapstan:

- The search results appear to be as good as those of a traditional search engine. However, in Mapstan’s case, there is an added advantage in that the presentation of the results clearly shows the relationship between websites, hence enabling a user to quickly see what are the relevant results. Furthermore Mapstan’s client software enables the user to discover new websites according to his interests.
- This is a novel tool, completely different from Google, the current search engine leader.

Weakness of Mapstan:

- For first-time users, the use of the search engine can be difficult, even without downloading the software.
- The software is a bar that gets displayed on the user’s screen. This can sometimes be disturbing.
- The search engine is slow.

- The graphical user interface is excessive. Whilst it helps one to understand the concept behind the search engine, it takes far too long to load.
- Both the use of the search engine and the software is free. However, Mapstan keeps track of users' data and eventually sells it for analytical purposes.

3.5.3. Kartoo

Website: <http://www.kartoo.com/>

Kartoo is a meta search engine that displays its result on a map. To refine a search the user just has to click on some part of the map.

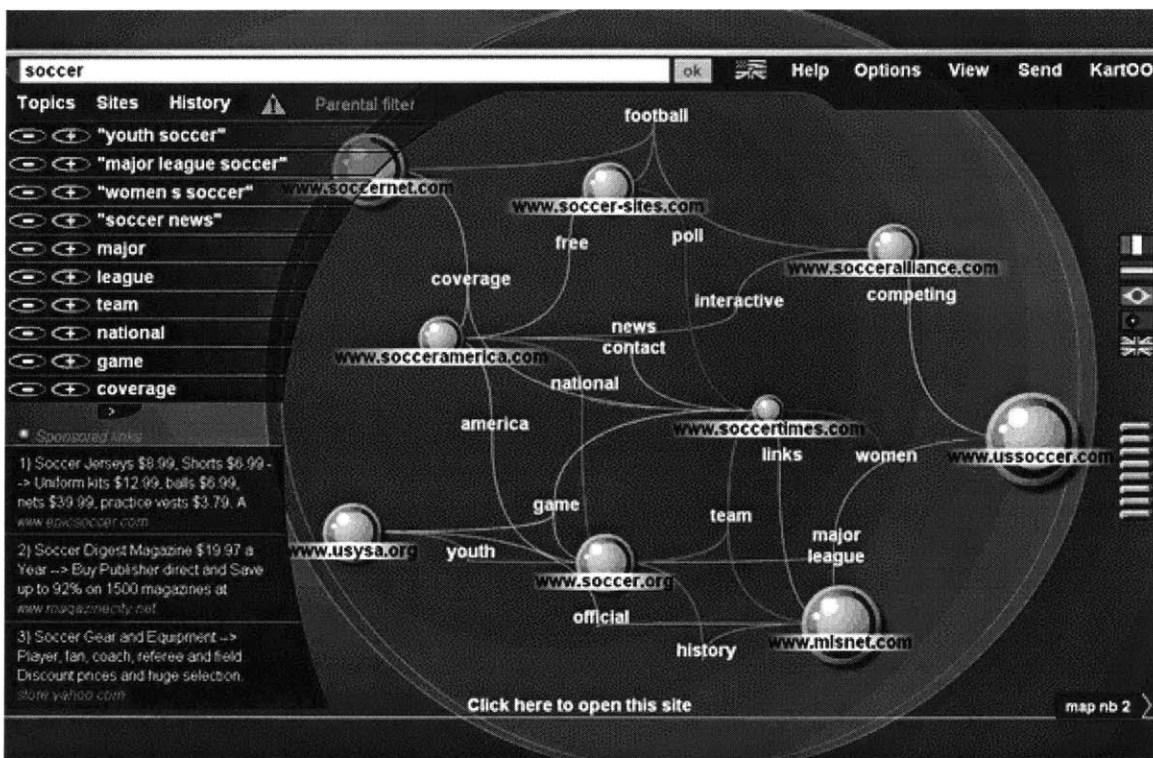


Figure 13: Kartoo Graphical User Interface

Strengths of Kartoo

- The search engine is fast.
- It is very easy and intuitive to modify a search and to refine it. It is more guided than traditional search engines.

Weaknesses of Kartoo

- The results are not better than those of a traditional search engine, only the presentation differs. The user profile is not used; neither is the history of the other user searches.
- The user's learning curve of the search engine is very steep.

3.5.4. AltaVista Peer-To-Peer

Website: <http://solutions.altavista.com/>

AltaVista has developed a technology that searches shared files, notably Office documents. This enables a company to quickly find information that is available on its Intranet. This is a very interesting technology, but it is different than Araignee because it does not allow the user to search on the Internet.

3.5.5. Human Links

Website: <http://www.human-links.com/>

Human Links is a collaborative oriented search engine. Here, users share their experience of web surfing. Searches are therefore more accurate thanks to the knowledge of what other people have searched. The user's Internet favorites are shared, as are the interesting pages a user finds and indexes in the database.

Once again the user-interface is designed to more accurately and simply represent the complexity of the natural world by rendering concepts, documents and contacts in relation to each other on a visual map.

The problem is that if there are not enough users, the software will not be useful since people do not usually bookmark a lot of web sites. More generally the company has assumed that users would be very willing to collaborate in order to build a very accurate database. However, users need to download a large application and to personally index the websites they visit. Such requirements may discourage users from using this search engine.

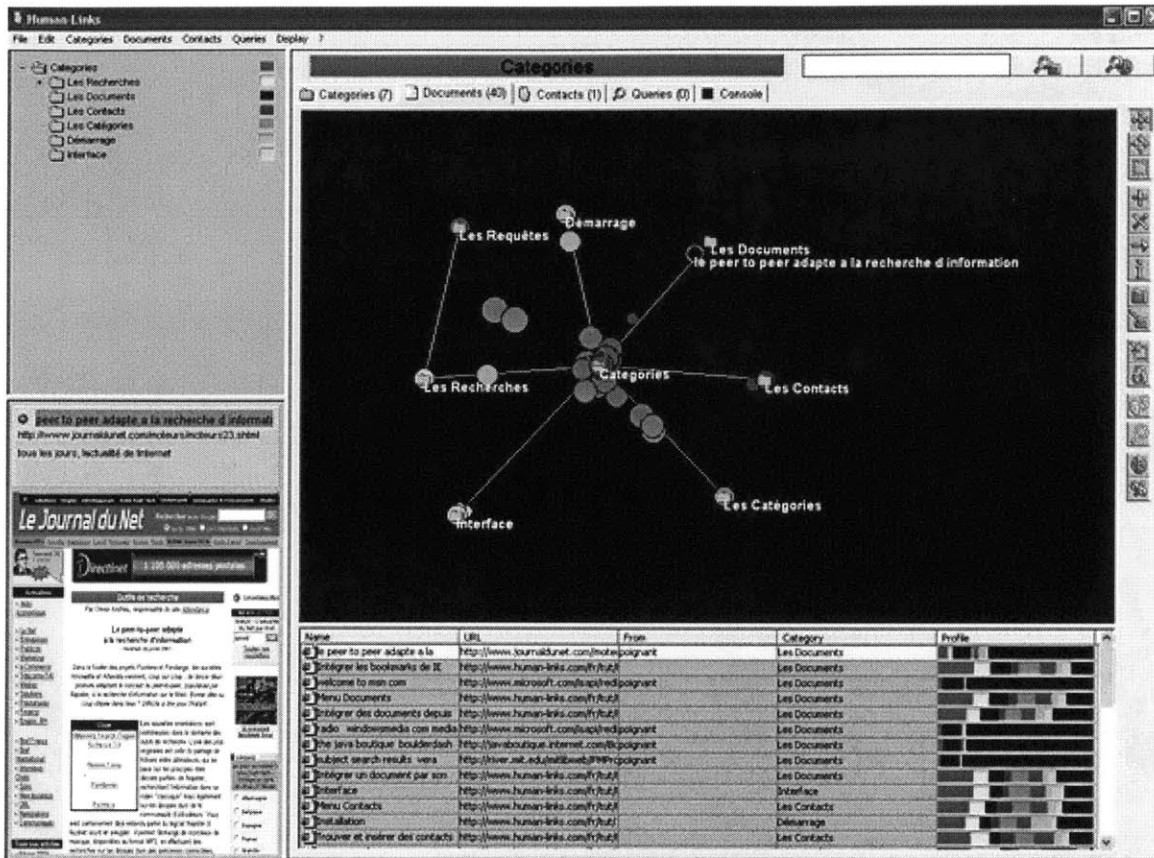


Figure 14: Human Links Graphical User Interface

3.5.6. Others

Many other search engines are being developed or are currently on the market. These include solutions by companies like Pointera, Vivissimo, Lasoo and Pandango. However, the functionalities of these search engines are similar to that mentioned in the above section.

3.6. Conclusion

Although the search engine market has been static for the past few years, the new generation of search engines looks set to create an impact in the coming months. This will undoubtedly challenge the dominant position of Google.

New search engines separately make use of a variety of web techniques like languages analysis, peer-to-peer as well as graphical presentations of the results. However none of the products currently on the market bring all of these techniques together under one roof to provide an encompassing search engine which Araignee aims to do. Not only will Araignee put peer-to-peer technology at the center of the searching process, it will also allow users to choose between a traditional view or a graphical view and to incorporate a linguistic module if the user so wishes.

4. The Technology Behind Araignee

4.1. Overview

In this chapter, we will examine in detail how we have chosen to implement Araignee. Figure 15 shows the general overview of the architecture of Araignee, whose software will be run locally by users on their machines.

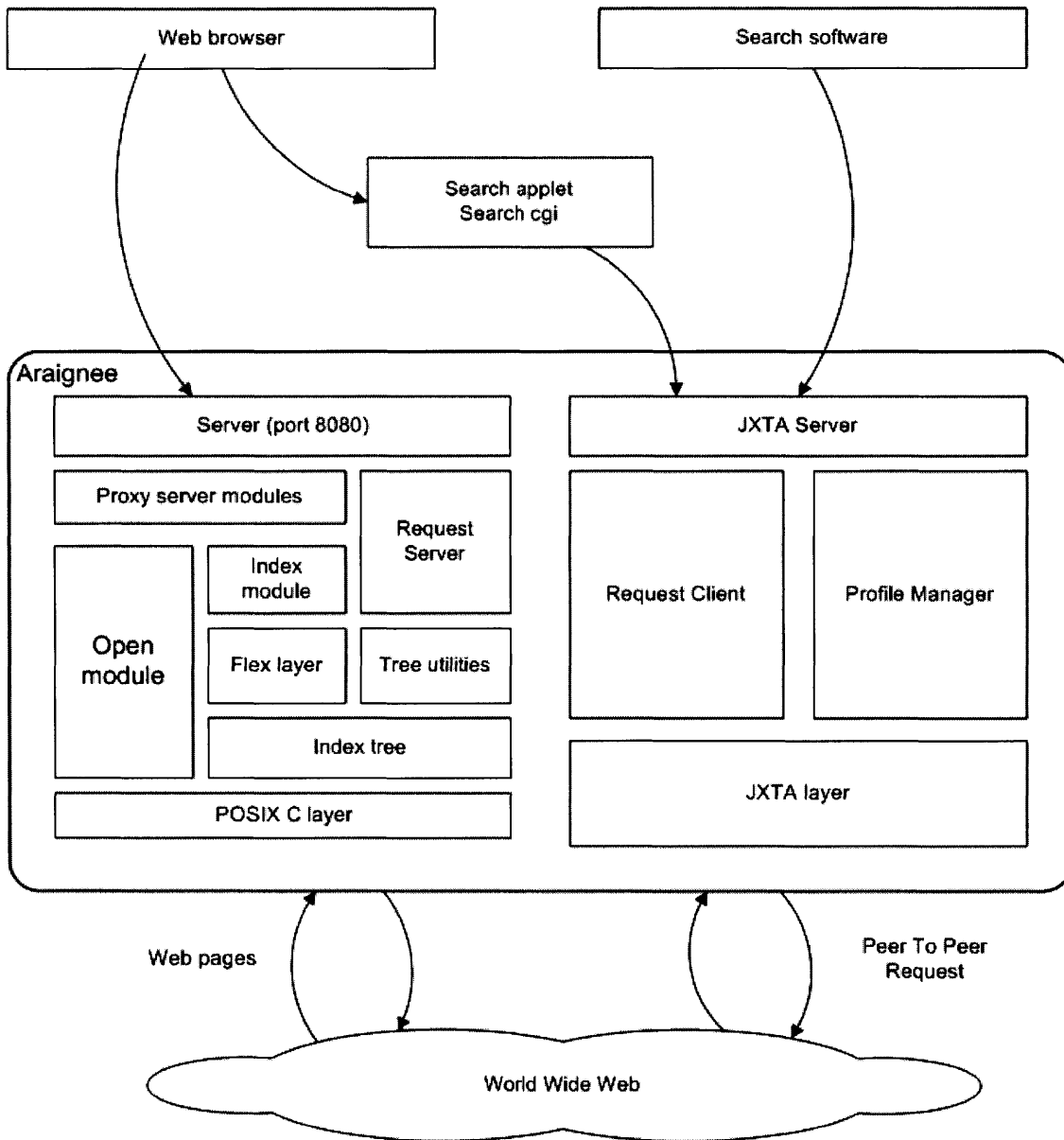


Figure 15: Araignee Software Architecture

Araignee's architecture can be divided into 2 main components, the local server (See Sections 4.2, 4.3 and 4.4) and the JXTA¹ server (See Section 4.5). The local server acts as the proxy, indexes the webpages and offers basic web server functions. On the other hand the JXTA server connects the user to the global peer-to-peer network to facilitate the exchange of data that has been indexed. The former component has been implemented in C++ using the POSIX system, while the latter component has been implemented in Java so that integration of the software within the JXTA framework can take place.

4.2. The Proxy Server

The proxy server is depicted in the left half of the schematic diagram in Figure 15: Araignee Software Architecture. The proxy server runs as a local service on the machine of every client. The services provided by this server can be classified into three different categories.

4.2.1. The Proxy

This component records the Internet surfing activities of Araignee users. The client uses his navigator to surf the Internet. However, instead of connecting directly to the Internet, the browser connects to the server on localhost: 8080. The server then relays the query on the Internet and returns the result to the client. This architecture is seamless for the user – the proxy monitors the web crawling without interfering with the user's habits.

¹ JXTA was a project initiated by Sun in 2000. This project came under the Apache License in April 2001 and has gained a huge community of developers to date. JXTA is a framework for peer-to-peer software based on use of XML format for messages. The goal of JXTA is to enable the implementation of distributed network computing using peer-to-peer topology, and to develop basic building blocks and services that would enable innovative applications for peer groups.

4.2.2. The Web Server

Some local pages are also available to the user through the server. These pages can be used to document how one should use the Araignee software, to help guide users who are unfamiliar with the Araignee software. Additionally, a user can make use of these pages to publish and explain the kind of data available on his computer in order to facilitate the use of such data by other users.

4.2.3. The Search Component

This component is very important as it parses and indexes the different pages opened by the user on the Internet. Using all this data, an accurate profile of the user is then built.

4.3. The Proxy Server Architecture

Figure 16 shows the UML structure of the proxy server, with the arrows depicting the links between the different components.

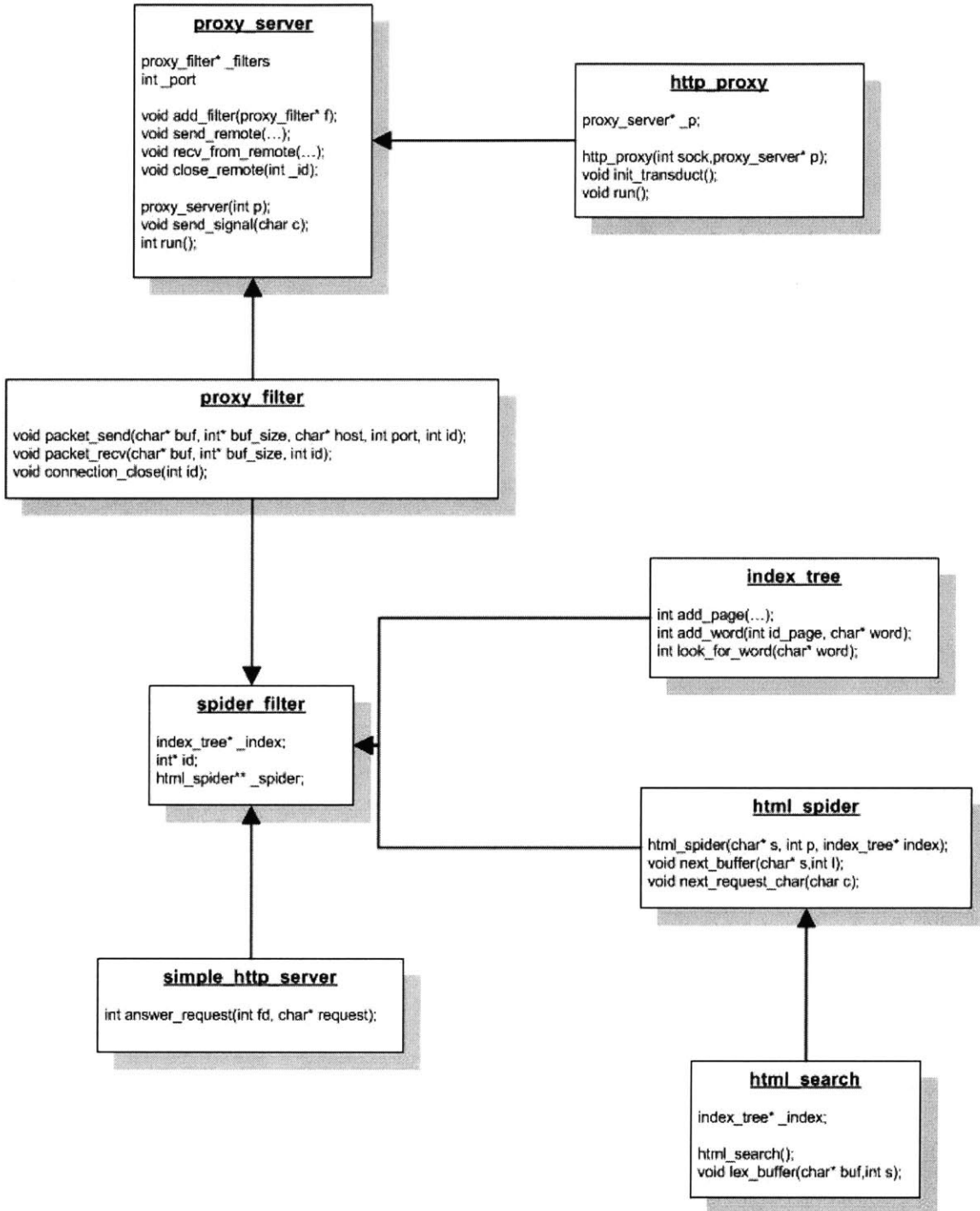


Figure 16: The proxy server architecture

4.3.1. The HTTP proxy server

The HTTP proxy functionalities are as described in the RFC 1945 [Fielding 1996]. In the Araignee architecture, the `proxy_server` class is in charge of this functionality. The browser sends to the proxy the following request:

```
GET http://www.google.com/index.html HTTP/1.0
...
```

The transducer part of the class recognizes the host name (in this case the Google search engine) and a connection is established to `www.google.com` on port 80. The following request is then sent to the web server :

```
GET http://index.html HTTP/1.0
...
```

The server will respond as follow:

```
HTTP/1.1 200 OK
...
```

and this response is transmitted to the browser.

The use of transducers to write proxy servers has been examined in [Brooks 1995]. A transducer is essentially a processing entity that is inserted into the stream Browser– ProxyServer–Server. A transducer processes messages. Its functionality can be divided into four categories, (i) filtering individual HTTP requests and responses, (ii) characterizing sets of messages, (iii) transforming message contents, (iv) additional processing indicated by the messages.

One may ask, why is a transducer needed? This is because there is no guarantee that the request will arrive in a single packet even if this is true in most cases. To

overcome this issue, either a buffer can be created to save the first line of the request or a simple transducer can be included. In this project, the latter course of action has been adopted.

4.3.2. The request server

The database is referenced through the `spider_filter`. This class is hence in charge of responding to the traditional HTTP request. When a request is received, the transducer analyses the request and checks that it is correct and legitimate, i.e. that the user is not trying to perform an illegal operation such as accessing a banned website.

There are a few possibilities that the transducer will detect:

- **A GET/POST method** – the client wants to surf the Internet: The server has to act as a proxy and return the corresponding webpage. The transducer connects to the remote host and the `http_proxy` returns the corresponding webpage (See Section 4.3.1 for greater details).
- **An Administration request:** The transducer executes the command entered by the administrator after ascertaining that the command comes from the local host.
- **A research command** – when a Araignee client wants to reference the database of another user. The Araignee server will then act as a web server. There are two kinds of queries that the server will have to answer:

(i) *A profile query:* What kind of profile does the user hold?

The server will respond based on the words that appear most frequently in the pages accessed by the user. However, the user can choose to specify words he does not want to appear in his profile. These would typically include trivial words like “the”, “and” and “a”. Additionally, users who are concerned that being associated with words with negative connotations such as “terrorism” or “pedophiles” may result in them being viewed in a negative light can also choose to exclude these words from their profiles.

- (ii) *A keyword query:* What are the webpages visited by the user that correspond to the keywords entered?

The query is passed to the server in the following manner:

```
GET /?urlencodedrequest HTTP/1.0
```

The answer is then returned with a traditional webpage such as:

```
HTTP/1.1 200 OK
Content-type: text/html
<HTML><BODY>
<br>www.google.com 80 / Google 17 21
...
</BODY></HTML>
```

This page can therefore be accessed through every web browser. A possible next step would be to implement this with XML.

- **A Web server query:** The server can answer some page queries. The file `net_help.h` provides basic support for this functionality. The main problem here is security since access to critical documents like that below should not be granted.

```
GET //etc/passwd HTTP/1.0
```

The solution here is simply to disallow files with paths that start with “/” or “./”, or which contain the string “./”.

4.3.3. The filter technology

The proxy is scalable, with the virtual class `proxy_filter` allowing filters to be added to the service. Every request is transmitted to the filters and every filter treats each request independently.

4.3.4. The transducer_filter

The transducer analyses the request and checks that it is correct and legitimate i.e. the user is not trying to perform an illegal operation such as accessing a banned website. The transducer then uses the different tools of the proxy to return the answer corresponding to the request.

4.4. The spider_filter

This filter is in charge of indexing all the content going through the proxy (at the moment only HTML or text). The class `index_tree` and the subclasses are used for this. The data extracted from the various pages are stored in a database.

4.4.1. Lex - A Lexical Analyzer Generator

Background

Lex is a program generator designed for lexical processing of character input streams. The user specifies regular expressions in the code given to Lex. Lex recognizes these expressions in an input stream and partitions the input stream into strings matching the expressions. Lex associates the regular expressions and the program fragments of the code. As each expression appears in the input to the program written by Lex, the corresponding fragment is executed.

The user supplies additional code beyond expression matching needed for completing his tasks. The program which recognizes the expressions is generated in C. Thus, a high-level expression language is provided to write the string expressions to be matched while the user's freedom to write actions is unimpaired. This avoids forcing the

user who wishes to use a string manipulation language for input analysis to write processing programs in the same and often-inappropriate string handling language.

Lex Source Code

The general format of Lex source is:

```
{definitions}
%%
{rules}
%%
{user subroutines}
```

The following example generates a histogram of the length of the word of an input file:

```
int lengs[100];
%%
[a-z]+  lengs[yyval]++;
. |
\n ;
%%
yywrap(){
    int i;
    printf("Length No. words\n");
    for(i=0; i<100; i++)
        if (lengs[i] > 0) printf("%5d%10d\n",i,lengs[i]);
    return(1);
}
```

In the above example, the method `yywrap()` is a C method called by Lex. Using this tool we generate a code that parses HTML. We can therefore detect different HTML tags and differentiate the content of the page from its layout. Every time a word is encountered, this word is added to the indexed tree.

4.4.2. The Database

The database in Araignee is composed of two structures: a list of WebPages and a tree of words. Every time a new page is looked up, it is stored in the WebPages list and assigned a new id. The software then parses the page using the code created by Lex. Every time Lex encounters a new word, it adds the word to the tree. The tree has an indexed structure, optimized for look up.

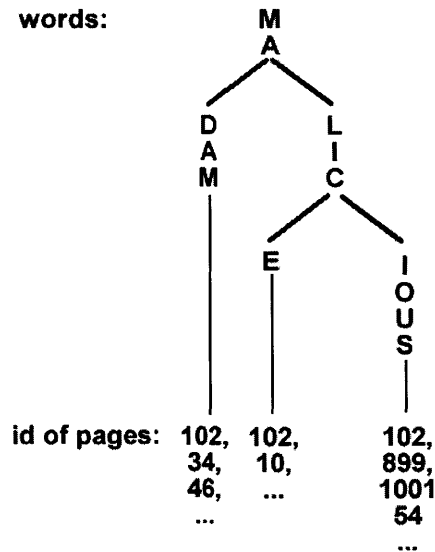


Figure 17: Example of an indexed Tree Structure

A Web Page is indexed only once and only the most recent visit is indexed. For each page, the following information is stored: the host, the port, the page, the title (extracted from the HTML) and the number of times the user has visited the page.

4.5. The Peer-To-Peer Server

4.5.1. The JXTA Framework

JXTA was a project initiated by Sun in 2000. This project came under the Apache License in April 2001 and has gained a huge community of developers to date. JXTA is a framework for peer-to-peer software based on the use of XML format for messages. The goal of JXTA is to enable the implementation of distributed network computing using peer-to-peer topology, and to develop basic building blocks and services that would enable innovative applications for peer groups.

JXTA was chosen as the foundation for the first implementation of Araignee for the following reasons:

- Apache License authorizes the use of the code in commercial products.

- The peer-to-peer network created can be used across firewalls (because it uses HTTP)
- Code for many platforms are available, including Pocket PC.
- The Java implementation is cross platform
- Many implementations are planned for other languages (C, python etc.)
- JTXA implements multiple services very useful to peer-to-peer networks. For example the Discovery Protocol locates advertisements from other peers. The Resolver Protocol locates peers, groups, pipes etc. The Information Protocol queries the status of other peers, controls peer nodes etc.
- A lot of open-source packages are available e.g. JXTA search (data exchange) or jnushare (file exchange protocol)

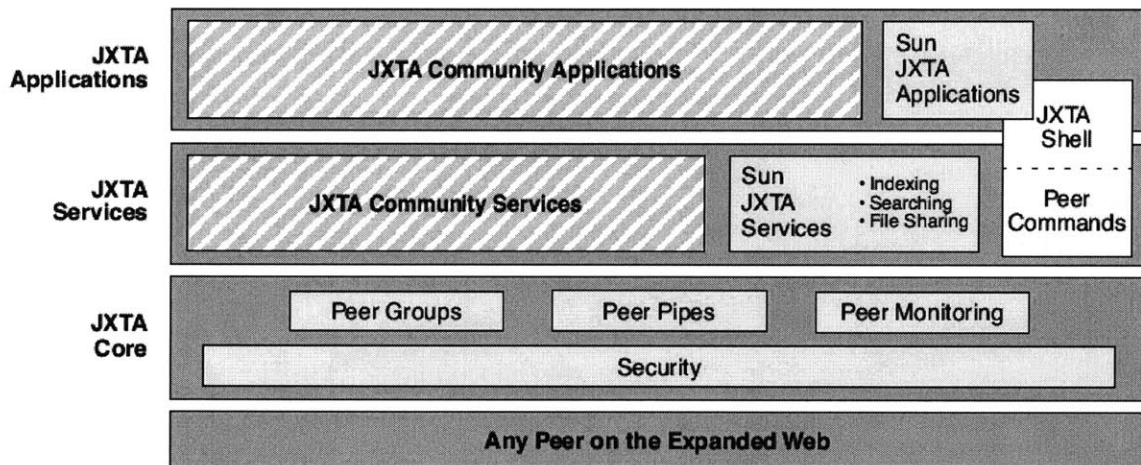


Figure 18: The JXTA architecture (JXTA Programmer Guide)

All the JXTA projects (services and applications) are based on the JXTA core code. Araignee’s peer-to-peer system is also based on the JXTA core but the inspiration for many the concepts and code implementation comes from open source projects provided by Sun.

4.5.2. The Peer-To-Peer Structure

Figure 19 shows the different components of the JXTA server.

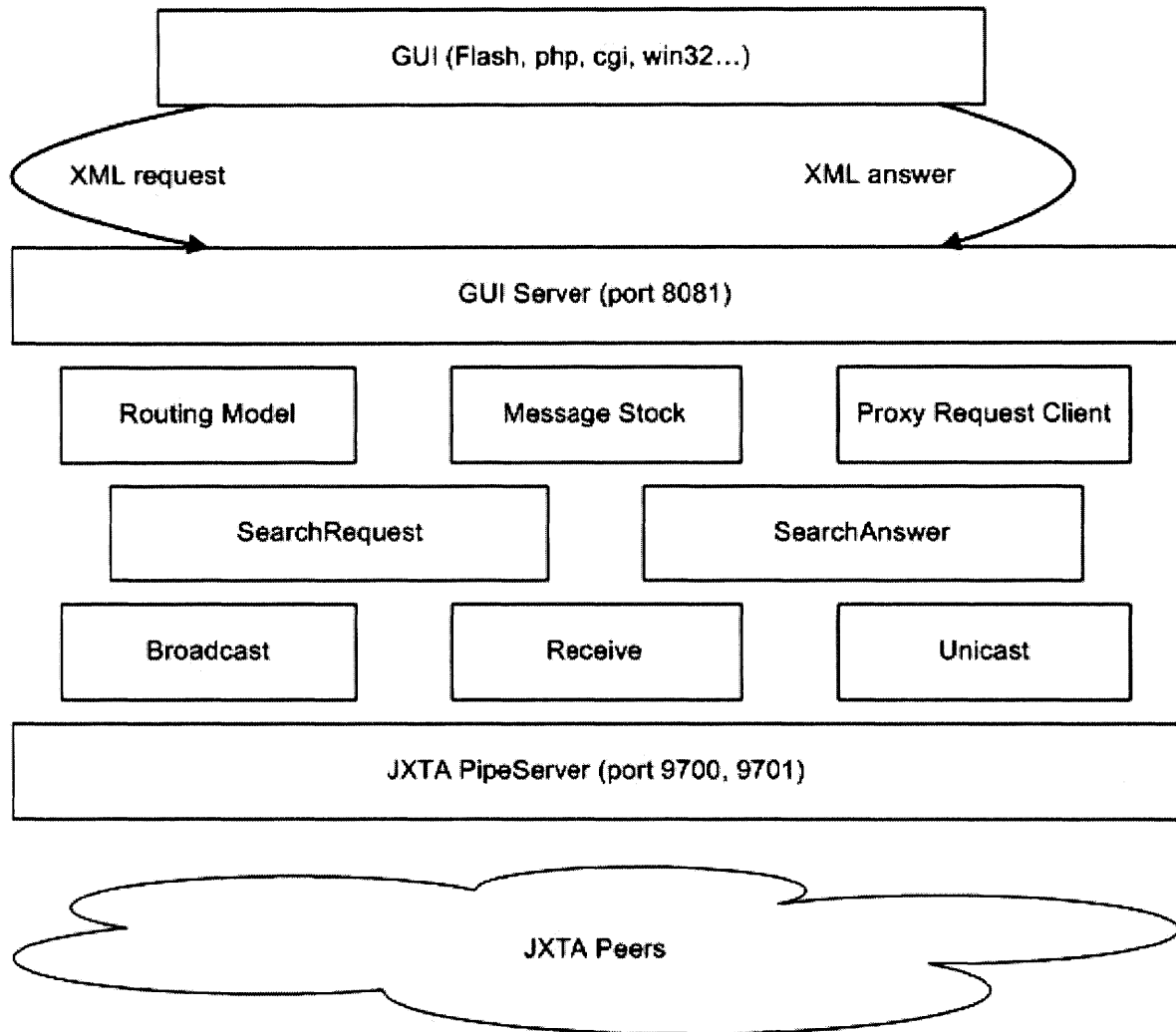


Figure 19: The GUI Server Architecture

The user uses a GUI to query the local server (i.e. the GUI server). The GUI server then uses the JXTA PipeServer to query the peer-to-peer network.

The query sent to the GUI has the following format:

```
<req request="urlencodedkeywords" profile="" />\0
```

This is a very straightforward use of XML. The request is a XML tag with two parameters, (i) the keywords stored in the request field and (ii) a profile field that will

eventually be used to hold the profile of the user so that queries can subsequently be restricted to the databases of users with similar profiles.

The results arrive with the following format:

```
<ans status="OK" />\0
<item    host=www.google.com    port="80"    page="/"
title="Google" ...
<ans status="END-OK" />\0
```

The results are in XML format. A result starts with an “ans” tag, followed by an “item” tag. Every “item” tag received gives a different website that corresponds to the query. (The fields of the item tag are used to connect the page.) At the end of the list of items, it is followed by another “ans” to indicate the end of the result.

4.5.3. The Communication Protocol

The communication protocol used is a simplified version of the Cristian Protocol [Cristian 1996].

MultiHop

The peer can parameterize the number of hops of each request before launching the request. As the routing protocol is stored on every peer, this parameter can therefore depend on the speed of the Internet connection.

MultiHub

Hubs centralize the request. The results obtained from their neighbors are then grouped before an answer is sent to the peer which launched the request.

Request Structure

The requests have a very simple format :

```
structure : ARAIGNEE_REQ_1.0 request request_origin nb_hops id_req
request : req:urlencodedrequest[,profil:urlencodedprofil]
id_req : id_machine-id_message or md5 hash(id_req)
```

Every request starts with the tag ARAIGNEE_REQ_1.0. This is followed by the origin of the request (which can be set to anything for privacy reasons), followed by the number of hops the request has to do on the network (this number will decreased everytime the request reaches a hub) and finally an unique identifier (id_req) that is used to tag the message.

```
response : ARAIGNEE_ANS_1.0 request id_req
list of results
```

The answers have a similar format. We only add the list of results behind the description of the message.

The main advantage of this format lies in the simplicity of the parser.

4.5.4. The PipeServer

The PipeServer is the entry point of all the communications related to the JXTA framework and therefore receives all the requests. It saves the id_req (unique identification of the request) for a limited time so that if a message comes back within that span of time, it will not be processed again. The PipeServer then acts as a hub for the message, forwarding the message to neighbors. As this continues, the number of hops is consequently decreased. When the PipeServer receives an answer, it adds the answer to those previously received so that a base of answers gets built up gradually. An answer is bounced back to its emitter when answers have arrived from all the emitters that were queried or when the message's time out is reached. This mechanism would therefore greatly benefit if a cache were implemented so that the answers to a query previously posed can be easily extracted from the cache instead of having to repeat the entire process.

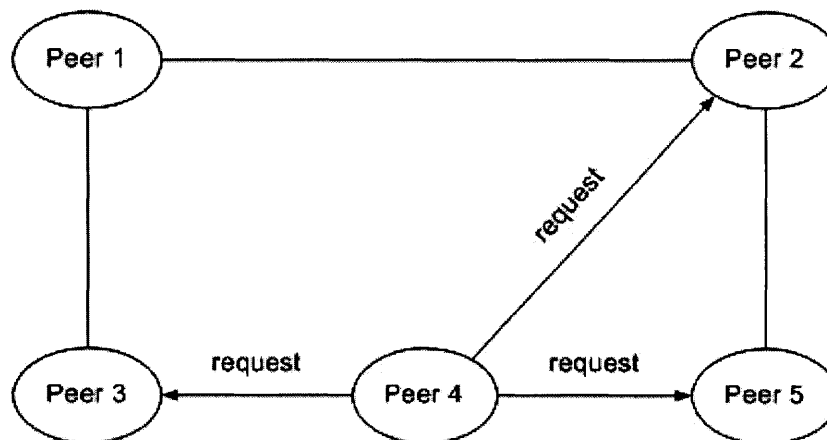
4.5.5. The Message Store

The message store remembers the previous message, gathers the answers and launches a cleaning thread every 10 ms to erase the timed out messages.

4.5.6. Message Routing Protocol

The routing protocol is very simple. Every peer sends the message to n neighbors (where n is a parameter). If it does not know enough neighbors, it will issue a request for new neighbors; this will be done asynchronously via the JXTA API. The id of a message is stored for a few seconds in the message store. If a message with the same id arrives again during this span of time, it is ignored. However, even though the duplicate message is ignored, an empty answer will be sent because the peers which sent out this “repeat” message are awaiting a reply to the message. If no answer is sent, then a timed out is used. The interesting point here is that all the routing rules are stored locally. While this allows for a very flexible implementation, unfortunately, it also creates a lot of redundancy. This issue will be addressed in Section 5.1.2.

The protocol is illustrated in Figure 20. Peer 4 first sends a request to all his neighbors (i.e. Peers 2, 3 and 5). Peers 2, 3 and 5 then forward the request to their neighbors and wait for their answers. Peer 1 answers Peer 2, and having answered Peer 2, sends an empty answer to Peer 3. Similarly, having received the query already previously from Peer 4, Peer 2 returns an empty answer to Peer 5. Finally, Peers 2, 3 and 5 answer Peer 4.



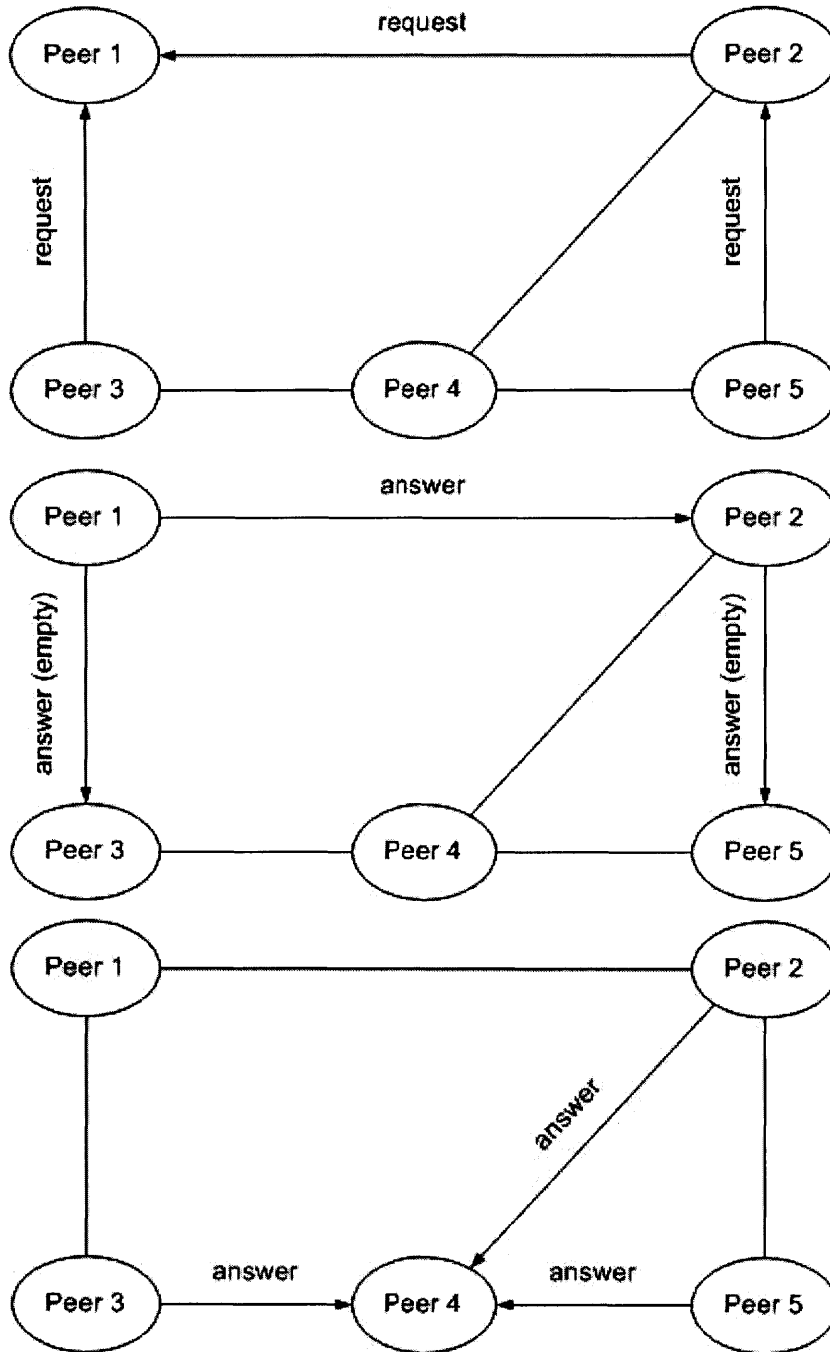


Figure 20: Routing Example

4.6. Deployment of the Software

The proxy server has been implemented and tested on a Solaris machine. The peer-to-peer application on the other hand, having been written in Java, can be executed from any operating system supporting Java.

Several rounds of testing using various machines have been conducted, and the test results show the software to run as intended.

5. Discussion

In this section, we will examine the critical success factors of the Araignee project. For this thesis, only the initial development of the Araignee software has been addressed. However, before Araignee can be deployed commercially, a number of other issues will need to be followed up on. Future work pertaining to the Araignee project should address the following issues.

5.1. Scalability

Thus far, Araignee has only been deployed using a handful of computers. The scalability of the software has thus not yet been tested. We can nonetheless identify a number of issues that are likely to arise when the software is deployed with a wider network.

5.1.1. Size of the Database

The design of the current Araignee database is very simple. If the number of sites exceeds say 1000 per peer, it should be recoded. This upper limit of 1000 pages will not be easily reached especially if the user only goes through the proxy for particular searches. However, this might not be the case in a network where there is only one proxy server for many peers, for instance a company network. The filter technology in which the proxy server has been written can however enable this change to take place easily.

5.1.2. The Routing Protocol

The routing rules have been stored locally. This means that a peer does not have to connect to a central server in order to know how to route the messages. Further, for reasons of privacy, the path of the message is not stored inside the message. Whilst this could create quite a number of redundant messages on the network, the messages are in

general very small (typically a few bytes), hence there is little risk that the network would be flooded.

Nonetheless, grouping peers according to regions and allowing only one communication per message between regions can resolve this issue, and the JXTA framework can easily handle this.

5.1.3. Conclusion

In general, we expect the peer-to-peer networks created in Araignee will be relatively small. In fact, Araignee is designed to help Internet users in very specific searches. The database on each terminal will therefore store only a limited number of webpages so a group of peers possessing the same interests will unlikely be huge. We foresee that Araignee will mainly be used for small communities or enterprise networks, so the issue of scalability should not pose a significant problem.

5.2. Reliability

5.2.1. Crash Resistance

As the Araignee software is a server, the issue of reliability is of critical importance. The proxy needs to be running all the time; otherwise the user cannot browse the Internet. In fact, the restarting of the server will have to be made easy so that in the event that the software crashes, the server can be easily restarted.

As the JXTA server is based on Sun's code, there is therefore a huge community of developers which can be relied upon to report bugs and crashes. In fact, the JXTA framework has been tested extensively by this community, which has also developed many applications using the JXTA framework. As such, the JXTA framework appears to be relatively reliable.

5.2.2. Database Integrity

Issues of integrity and reliability will undoubtedly arise when it comes to databases. The former issue can be easily resolved if the database is local, by simply locking the entire database when someone is writing into it. Such a move will affect the performance only minimally. If, however, a company or a group decides to use the proxy for more than one computer, Araignee should be rewritten with a better database.

The trickier issue is what happens when the software crashes. Currently, the database is stored on the disk every time 50 pages have been visited. A related issue that needs to be further considered is what happens should a peer decide to leave the network. In such a scenario, will all the data then be lost? To better cater to these two issues, a central server should eventually be implemented to store some of the data.

5.2.3. Results Accuracy

Through searching the database of users belonging to specific interest groups, Araignee generates more accurate search results for a user. However, any user is almost certainly not going to surf the Internet for just one purpose (and the same one at that) over an extended period of time. The software should therefore provide a user with the flexibility to change his group setting while surfing, for instance via an application accessed in the system tray. This will require some collaboration from the user.

For greater accuracy of the search results, the time spent by a user on a page can also be included as a parameter. Due to the nature of the HTTP protocol however, it might prove difficult to assess accurately the time spent by a user on a specific page.

A further method to improve the accuracy of the search results is to let users judge for themselves the various pages either by book-marking it through Araignee or by grading the pages. The more highly graded webpages would correspondingly get higher weight in the display of the results. Once again this would require co-operation from users. Araignee however has been designed to be seamless for its users. Such collaboration required from users might hence compromise on this ideology.

A key factor in judging the relevance of a particular webpage to a particular topic is the familiarity of the person with the subject at hand. A history professor's evaluation

of a webpage is in general deemed more valuable than a student's evaluation of the same webpage. By this reasoning, the webpages accessed by the professor should have a heavier weight in the display of the results. Though straightforward, this fact is quite difficult to exploit in the Araignee software. In companies or small communities, individuals whose evaluation ought to carry a heavier weight can be identified relatively easily. In larger groups of people where the people do not know one another, this is much more difficult. One solution is to give higher weight to the pages evaluated or accessed by the more active participants. The flip side of this solution however is that it might lead to more spam.

5.3. Privacy

Privacy is a crucial issue in Araignee especially since the software monitors everything the user does on the Internet. The recent public outcries against spyware software have shown the public's increasing reluctance to use such software. The unique thing about Araignee is that there is a function which allows the user to enable or disable the software according to his preference, in order to protect his privacy. Nevertheless, we need to be more specific about what else can be done to protect an individual's privacy. This is detailed in the following sections.

5.3.1. Routing Protocol

The protocol ensures privacy of requests. When a user wants to query Araignee, he queries the local JXTA PipeServer through the GUI server. The PipeServer then acts as a hub for this request. It is therefore impossible to track who issued a particular request on the network.

5.3.2. Privacy Statement

Whilst a high level of integrity is required to ensure that data extracted from users' surfing activities will be treated with the highest confidence and not be divulged, it should also be supplemented by a software agreement with a clear privacy statement, to inform and raise awareness amongst users of the potential risks of privacy infringement arising from use of the Araignee software. Together, these can help bring about responsible monitoring and enable users to benefit from a peer-to-peer search engine rather than having to take chances or being overly worried that using Araignee will infringe upon their privacy.

5.3.3. Open Source

For the moment, Araignee is not an open source project. However, once the first version of Araignee is launched, it is suggested that the code be under the GPL license. Not only would this enable the project to grow with a growing community of developers, the increased transparency would also help create trust with the public by assuring them that Araignee is not a spyware.

5.4. Profiling / Personalization

Araignee is a very useful tool for gathering statistics on a user. However these statistics will not be of much use if they are not analyzed. As indicated in Section 5.2.3, whilst we want to give a user the flexibility of choosing his own profile, at the same time, we want the use of Araignee to be as seamless as possible for a user. It would therefore be ideal if a user's profile can be detected through his surf activities. A lot of work has been done in the field of collaborative filtering. Some of the ideas pertaining to collaborative filtering could perhaps be applicable in Araignee, to analyze the statistics obtained from monitoring a user's surf activities. This is discussed in detail in the following sections.

5.4.1. Collaborative Filtering

Collaborative Filtering is a technique for providing recommendations and/or critiques based on statistical match of peoples' taste. The Collaborative Filtering System maintains a database of users as well as the items rated by the users, for instance webpages. The system then compares the vectors of ratings of different users and groups people with similar opinions together ([Basu 1998], [Breese, 1998], [Chaptini 2003]). Should a user from a particular group seek advice on a particular issue, advice will then be provided to him based on the experience of others in his group. The rationale behind this is that since the different users in a particular group have more or less similar preferences / opinions, it is likely that the advice will correlate with the opinion of the user.

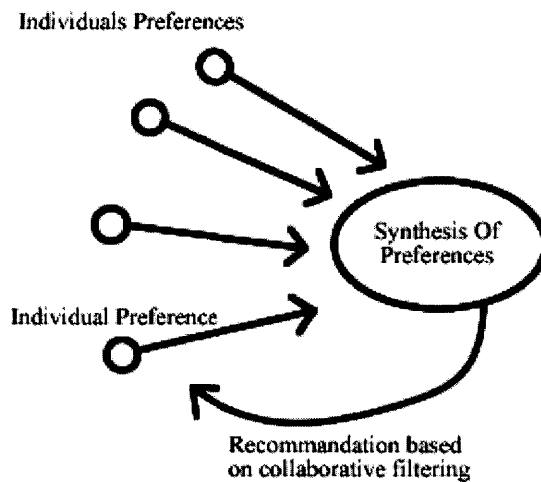


Figure 21: How Collaborative Filtering Works

Araignee can make great use of this technique. The search engine can query a central service that dishes out advice according to the surf habits of the user. The central service then reroutes the search engine to the community that would match the user's profile the most. This allows Araignee to offer to a user webpages which others with similar profiles have found useful.

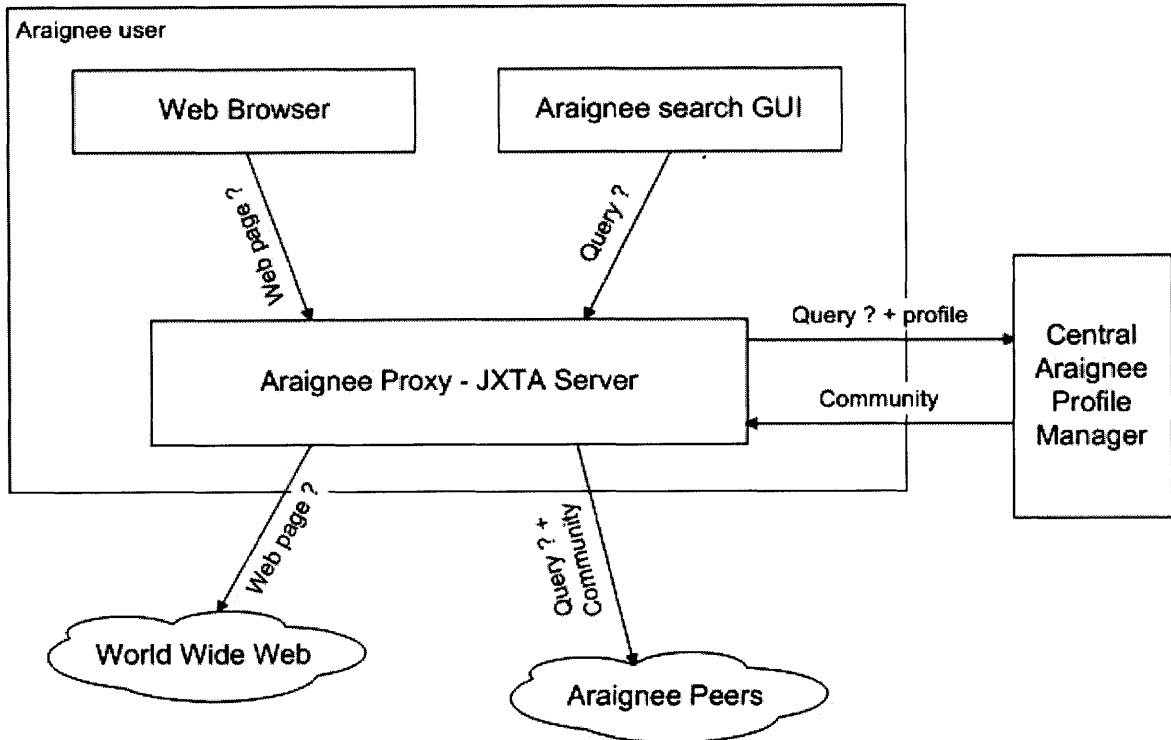


Figure 22: Araignee Recommender System

5.5. Human Computer Interaction

For a community search engine, the more users that subscribe to it, the more effective it will be. The interface of the Araignee will hence need to be very friendly in order to enhance the usability of the product to encourage uptake of the application. In particular, having a good interface in place will allow users to pick up the application easily as opposed to leaving users frustrated if the user-interface is poorly designed.

6. Conclusion

Given the information overload nowadays, Internet search tools are becoming an indispensable tool to users. The Internet search tool market used to be dominated by Google. However, as Internet users increasingly demand for more accurate search results, Google's position looks set to be toppled by the new generation of search engines such as Araignee which adds a human factor to the classification of results. Such new search engines offer the best of both worlds by combining the accuracy of directories with the breadth of results of automatic search engines. Furthermore, the predicted widespread adoption of broadband technology in the near future (with its 24x7 connection) will help create a fertile environment for the deployment of such peer-to-peer search engines.

In this thesis, we have taken a first crack at laying the groundwork for the development of the Araignee software. Whilst we have deployed different technologies in the process, much work still needs to be done to reach a stage before the Araignee software can be commercialized. In particular, it should be noted that this thesis is not intended to be an ending piece that aims to give the last word on peer-to-peer search engines. Rather, we hope it will be a seminal piece that can stir researchers into launching further investigations into this area. Peer-to-peer search engines can then become the killer application which many have predicted they would be.

7. References

[Basu 1998] Basu, Hirsh, and Cohen "Recommendation as Classification: Using Social and Content-based Information in Recommendation", *Recommender System Workshop '98* pp. 11-15.1998

[Breese, 1998] Breese, J. S., Heckerman D., and Kadie C. "Empirical Analysis of Predictive Algorithms for Collaborative Filtering". *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pp. 43-52. 1998

[Brooks 1995] Charles Brooks, Murray S. Mazer, Scott Meeks, and Jim Miller, "Application-Specific Proxy Servers as HTTP Stream Transducers", *MIT/OSF Research Institute*, December 1995 <http://www.w3.org/Conferences/WWW4/Papers/56/>

[Chaptini 2003] Chaptini Bassam, « Reccomender Systems and Collaborative Filtering : Litterature Review", *Unpublished* February 2003.

[Cristian 1996] F. Cristian, "Synchronous and Asynchronous Group Communication", *Communications of the ACM*, vol. 39, 1996

[Fielding 1996] R. Fielding, UC Irvine, H. Frystyk, "Hypertext Transfer Protocol -- HTTP/1.0", *MIT Laboratory for Computer Science*, May 1996
<http://www.ietf.org/rfc/rfc1945.txt?number=1945>

[Gradecki 2002] Joe Gradecki, "Building Java peer-to-peer applications", Wiley Pub., 2002.

[JXTA 2001] Sun Microsystems, Inc., "Project JXTA Programmer's Guide", 2001,
http://www.jxta.org/docs/jxtaproguide_final.pdf

[Marckini 2001] Fredrick Marckini, "Search engine positioning" Wordware Pub., 2001
<http://www.books24x7.com/marc.asp?isbn=155622804X>

[Miller 2001] Michael Miller, "Discovering P2P" San Francisco Calif SYBEX, 2001,
<http://www.books24x7.com/marc.asp?isbn=0782140181>

[Paxson 1995] Vern Paxson, "Flex, A fast scanner generator", March 1995,
http://www.gnu.org/manual/flex-2.5.4/html_mono/flex.html

[S&V 2000] Science et Vie, "Les secrets des moteurs de recherche", *Science et Vie* NO 998, November 2000

[Searchenginewatch 2003] Internet.com, "Search Engine Watch Reports", January 2003
<http://www.searchenginewatch.com>

[STL 1994] Silicon Graphics, "Standard Template Library Programmer's Guide", 1994
<http://www.sgi.com/tech/stl/>