# Agent Architecture for Score-based Web Local Search [1]

M. Angelaccio, B. Buttarazzi

*Dipartimento di Informatica, Sistemi e Produzione - Università degli Studi di Roma "Tor Vergata"*

*Via di Tor Vergata, 110 - 00133 - ROMA (Italy)*

*angelaccio@info.uniroma2.it buttarazzi@info.uniroma2.it*

## Abstract

*Big changes are taking place in the area of information supply and demand on Internet. The first big change, which took place quite a while ago, is related to the type of information is available and the amount that it is available in, the number of sources and the ease with which it can be obtained.*

*What's more, information is playing an increasingly important role in our lives, as we are moving towards an information society. Information has become an instrument, a tool that can be used to solve many problems. Recent research trends show that these developments will carry on into the future through the agent technology. Software agents are a rapid developing area of research and they are largely used for web search tools.*

*In this paper is presented an agent-based framework for local search tools and it has been applied to the case of VSEARCH a local search tool implemented in Java. In particular, it has shown how agent properties are strictly related to VSEARCH document filtering.*

## 1. Introduction

The amounts of information available in Internet today is too vast: information that is being sought is probably available somewhere, but often only parts of it can be retrieved, or sometimes nothing can be found at all. The number of needles that can be found has increased, but so has the size of the haystack they are hidden in.

The sheer endlessness of the information available through the Internet, which at first glance looks like its major strength, is at the same time one of its major weakness.

Typically, one would expect that satisfying information demand has become easier, but this in not completely true, the reasons for this being:

- The dynamic nature of the Internet itself: there is no central supervision on the growth and development of Internet. Anybody, who wants to use it and/or offer information or services on it, is free to do so. This has created a situation where it has become very hard to get a clear picture of the size of the Internet, let alone to make an estimation of the amount of information that is available on or through it;
- The dynamic nature of the information on Internet: information that cannot be found today, may become available tomorrow. And the reverse happens too: information that was available, may suddenly disappear without further notice, for instance because an Internet service has stopped its activities, or because information has been moved to a different, unknown location;
- The information and information services on the Internet are very heterogeneous: information on the Internet is being offered in many different kinds of formats and in many different ways. This makes it very difficult to search for information automatically, because every information format and every type of information service requires a different approach.

Most of the current solutions are based on programs that roam the Internet (with names like spider, worm or searchbot). The gathered information, characterized by a number of keywords (references) and perhaps some supplementary information, is then put into a large database. Anyone who is searching for some kind of information on the Internet can then try to localize relevant information by giving one or more query terms (keywords) to such a search engine.

[1] The work described in this paper has been carried out under the financial support of the Italian Ministero dell'Università e della Ricerca Scientifica e Tecnologica (MURST) in the framework of " ALGORITHMS for LARGE DATA SETS: SCIENCE and ENGINEERING" Project

Although search engines are a valuable service at this moment, they also have several disadvantages ([3], [7]).

A different solution for the problem as described in 2, is the use of tools based on Software Agents that autonomously explore a neighbor of a starting node (local search).

There are many different kinds of software agents, ranging from Interface agents to Retrieval agents. This paper will be mainly about agents that are used for discovery information tasks, such as finding any kind of information.

## 1.1 Agent Properties

In this section we outline the main issues of software agents that will be used in the rest of the paper (see [5] and [6]). An agent is usually described as a software program that supports a user with the accomplishment of some task or activity.

Perhaps the most general way, that distinguishes agents from ordinary programs, in which the term software agent is used to denote a software system that enjoys a long list of properties, which for information retrieval and integration purposes can be limited to the following:

- Autonomy: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;
- Social ability: agents interact with other agents via communication language;
- Reactivity: agents perceive their environment (which may be the physical world, a user via a graphical user interface, a collection of other agents, the Internet, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it. This may entail that an agent spends most of its time in a kind of sleep state from which it will awake if certain changes in its environment (like the arrival of new e-mail) give rise to it;
- Proactivity: agents do not simply act in response to their environment, they are able to exhibit goal-directed behavior by taking the initiative;
- Temporal continuity: agents are continuously running processes (either running active in the foreground or sleeping/passive in the background), not once-only computations or scripts that map a single input to a single output and then terminate;
- Mobility: the ability of an agent to move around a network.

Notice that no single agent possesses all these abilities and at this moment no consensus has yet been reached about the relative importance of each of these characteristics in the agent as a whole.

Thus, a simple way of conceptualizing an agent is as a kind of UNIX-like software process that exhibits the properties listed above.

## 2. Agents for Web Local Search

We propose to exploit the class of Web local searching tools that aim to explore a neighbor of a starting web node, in terms of agent concepts.

The purpose is to show a practical example of local search tool named VSEARCH (see [1] for a first description) whose architecture details are given in terms of software agents.

Before entering into architecture details for VSEARCH, we analyze in this section the agent architecture of a local searching tool that provides an interface (virtual level) for the searching/browsing phases.

### 2.1 Agent Description for Local Search

Traditionally the so-called user-web server access for information retrieval can be decomposed in two phases: searching phase and browsing phase.

The first searching phase consists in:

- Submission of a query to a search engine

The second one: searching & browsing for each interesting link returned from search engine:

- Browsing of selected document.
- Seek for the correlated information.

Normally the browsing process is not limited to the only pages returned by the search engine, but also to those correlated.

Fig. 1 shows the two phases of the typical searching-browsing access schema for the client-remote server architecture.

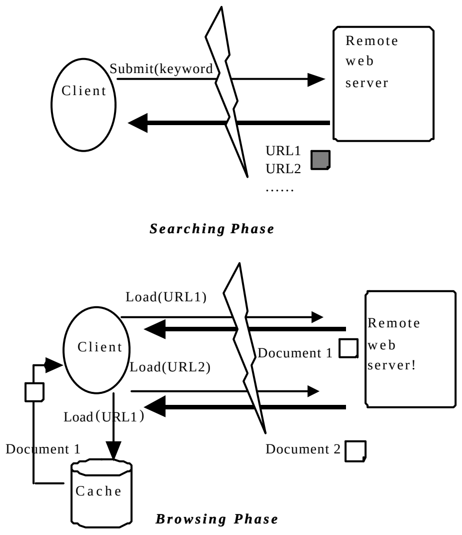


**Fig. 1 User-remote server access**

Bolded arrows evidence slow communications due to network latency and server overloading.

In the searching phase the client submits a keyword to the remote server. Next a list of URLs (URL1, URL2,…) will be received from the server as a result of the search executed on the remote server.

In the browsing phase a sequence of loads whose address has been taken from the answers obtained in the searching phase have been submitted to remote server. Hence it happens that only cached URLs offer fast access, thus motivating the need to have a suitable interface between the user client and the remote server in order to reduce latency time for other documents.

As alternative to the user-agents-webserver access a suitable interface has been introduced in many local search tools. In this way, the agent-based approach to local search can be viewed as a virtual access to the web. In other terms software agents are used

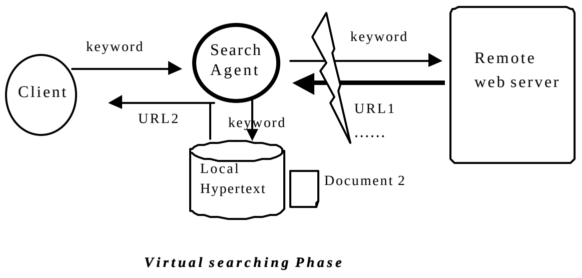


**Fig. 2 Virtual searching in the virtual server model**

as interface between the user client and web server in a way like a "virtual server".

Fig. 2 and Fig. 3 describe how virtual server is organized. It evidences the agents used to support a virtual access for the searching-browsing schema. These agents are templates for true agents that are used in local searching tools. In particular we evidence search and prefetch template agents.

The searching phase (Fig. 2) may be viewed as virtual search in the sense that the search is explicitly executed on the local copy freeing the user to have remote access. The completion of the search is done remotely by the search agent that independently communicates with the remote server. In such a way the system offers the capability to overlap the incoming answers. For instance we have assumed that URL2 is a local answer that will be sent directly from the local mirroring agent and concurrently from the other answers that will be sent by the remote server via the search agent.

The browsing phase (Fig. 3) is implemented as virtual browsing in the sense that all load actions are directed to the local hypertext by consulting the prefetch agent. The user avoids remote loads, which are delegated to the prefetch agent.

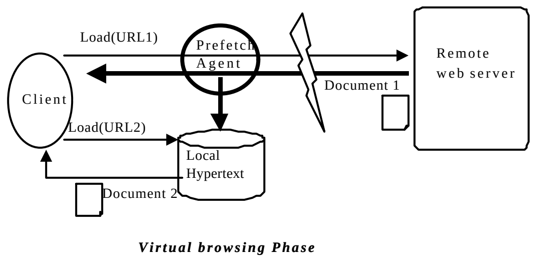


**Fig. 3 Virtual browsing in the virtual server model**

The behavior of the virtual server requires the definition of the actions taken from the prefetch in corresponding of both virtual searching phase and virtual browsing phase.

In the next section we show an example and its implementation of such agent architecture named VSEARCH that explore a neighbor of a starting node by using a score-based prefetch model. In this case the agent characterization introduced in 1.1 will match VSEARCH features like score-based prefetch model.

## 3. VSEARCH Software Agents Architecture

In this section we describe how agents introduced in the virtual server model are implemented by showing a concrete example named VSEARCH. This local searching tool performs a neighbor search, by using a score-based prefetch model to select pages that must be explored and then pushes the results back to the user's browser in a tree map. Herein we focus on its agent architecture that is described in Fig. 4.

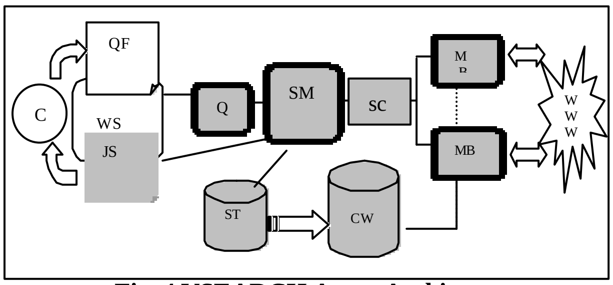


**Fig. 4 VSEARCH Agent Architecture**

Colored boxes in Fig. 4 evidence VSEARCH components. The two agents SearchManager (SM) and MicroBrowser (MB) respectively implement Search and Prefetch agents introduced in virtual server model. Moreover to obtain a dynamic html interface, a third agent named QueryParser (QP) has been added.

The others components of Fig. 4 are standard software components that interface with VSEARCH. In particular, WS is the web server that receives query from the browser client C via the html CGI program called by the query form QF, and JS is the html document that displays the results using a dynamic JavaScript visualization program. It has been evidenced the score computation function (sc) called by the SM agent and applied to each page fetched by the MBs.

## 3.1 VSEARCH Agent Properties

In the following we give for each agent a description of its behavior by showing its agent properties.

The table shown in Fig. 5 lists the main agent properties introduced in section 1.1.

Mobility has not been used for all components because VSEARCH is a local tool running on a site and is not able to migrate autonomously on other nodes (this is a feature for future developments).

QueryParser evidences a little of agent properties because it is only an HTML/Jscript interface component and it is launched only at the beginning of query session (no autonomy and no temporal continuity).

**Table 1 Agent properties for VSEARCH**

| | Search Manager | Micro Browser | Query Parser |
|---|---|---|---|
| *Autonomy* | × | × | |
| *social ability* | × | × | × |
| *reactivity* | × | × | × |
| *proactivity* | × | | |
| *temporal continuity* | × | × | |

It is interesting to note that the main difference between SearchManager and MicroBrowser is the fact that only the former has the proactivity property. This corresponds to the score computation made for each loaded document and used for taking a decision regarding how many links must be further explored (score-based prefetch model).

## 4. Example

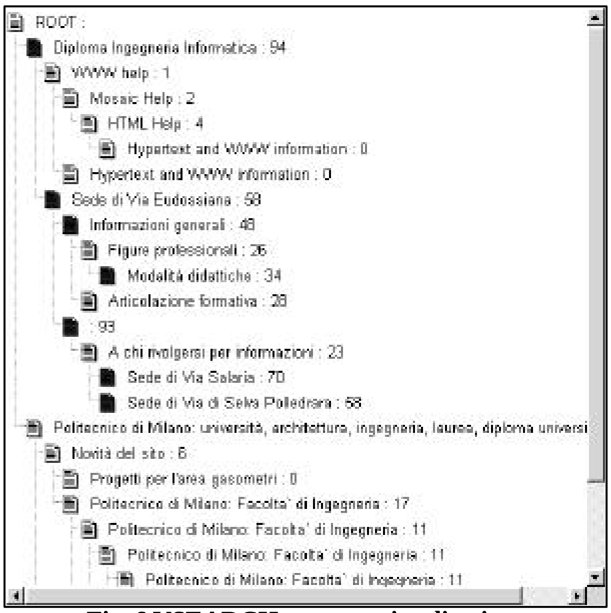
To illustrate the behavior of VSEARCH, we show an example of a query given in a session.

**query :**
Find all documents in the **neighbor** of "***Local Yahoo***"
With the keywords "***Diploma Ingegneria***"
and **`depth`** = 8 **`width`** = 5

**Fig. 5 Query example**

Fig. 5 gives an example of query expressed in natural language with neighbor size parameters depth and width. After submission of this query by inserting keywords and parameters in the html interface, the QueryParser awakes SearchManager and a Searching/Browsing process starts.

The result is shown in Fig. 6 by using a dynamic visualization. This output has the novel feature to show both the search tree structure obtained by the agents and the computed scores via a coloring technique. This has been made possible thanks to the agent properties of reactivity and proactivity.



```
ROOT :
  Diploma Ingegneria Informatica : 94
    WWW help : 1
      Mosaic Help : 2
        HTML Help : 4
          Hypertext and WWW information : 0
      Hypertext and WWW information : 0
    Sede di Via Eudossiana : 58
      Informazioni generali : 48
        Figure professionali : 26
          Modalità didattiche : 34
        Articolazione formativa : 28
      : 93
        A chi rivolgersi per informazioni : 23
          Sede di Via Salaria : 70
          Sede di Via di Selva Poliedrara : 58
  Politecnico di Milano: università, architettura, ingegneria, laurea, diploma universi
    Novità del sito : 6
      Progetti per l'area gasometri : 0
      Politecnico di Milano: Facolta` di Ingegneria : 17
        Politecnico di Milano: Facolta' di Ingegneria : 11
          Politecnico di Milano: Facolta' di Ingegneria : 11
            Politecnico di Milano: Facolta` di Ingegneria : 11
```

**Fig. 6 VSEARCH answers visualization**

## 5. Conclusion and Related works

This paper has presented a high-level description of local search tools in terms of software agents that it has been applied to the VSEARCH tool in order to characterize the behavior of its main Java modules. Hence main differences among components are evidenced without entering into architecture details. Other examples of local search tool are:

- Personal agents that build user profiles ([4], [8]). These systems exploit the knowledge of user behavior to assist user browsing. This require other different technologies to learn from user actions.
- WebSite search programs that build a searchable data structure of a website. WebGlimpse [9] is one example that has been the first to include neighbors. However the approach is static. In fact the dimension of neighbors cannot be defined at user level as in VSEARCH. We think that since autonomy and reactivity seems not satisfied they can be better classified as programs instead of agents.

Therefore VSEARCH and in general agent-based local search tools provide a starting point for developing more powerful local search tools. As a future work, since WebGlimpse has been described as example of intelligent search that can be expressed in the relational web query language WebSQL ([10], [11]), it is interesting to give an analogous description for general local search tools like VSEARCH.

## 6. References

[1] M. Angelaccio, L. Zamburru, D. Genovese "BOTH:Cooperative Automatic Web Navigation with Hierarchical Filtering" , *AusWeb96 conference*, July 1996 http://elmo.scu.edu.au/sponsored/ausweb/ausweb96/tech

[2] G. Arocena, A. Mendelzon, G. Mihaila. "Application of a web query language", *Proc. of 6th WWW Conference*, Santa Clara, California, April 1997.

[3] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. http://huron.stanford.edu/ backrub/google.html, 1998

[4] Chen, L. and Sycara, K. "WebMate: A Personal Agent for Browsing and Searching." In *Proceedings of the 2nd International Conference on Autonomous Agents and Multi Agent Systems*, Minneapolis, MN, May 10-13, 1998

[5] Haverkamp, D. S. & Gauch, S. (1998), "Intelligent Information Agents: Review and Challenges for Distributed Information Sources," in Journal of the American Society for Information Science, 49(4):304-311.

[6] Jennings, N. R. & Wooldridge, M. J. (Eds.) (1998), *Agent Technology*. Springer Verlag

[7] J. M. Kleinberg. "Authoritative Sources in a Hyperlinked Environment", *Proc. of ACM-SIAM Symposium on Discrete Algorithms*, 1998

[8] Lieberman, H. "Letizia: An Agent That Assists WebBrowsing", *Proceedings of AAAI 95 - AI Applications in Knowledge Navigation and Retrieval* - MIT Cambridge MA, USA, 1995, pp. 97 –103

[9] U. Manber, M. Smith, B. Gopal,. "WebGlimpse-Combining Browsing and Searching". *Proc. Of the Sixteenth ACM Symposium on Principles of Database Systems*, Tucson, Arizona, May 1997.

[10] A. Mendelzon, T. Milo. "Formal Model of web queries". *Proc. Of the Sixteenth ACM Symposium on Principles of Database Systems*, Tucson, Arizona, May 1997.

[11] Alberto O. Mendelzon, George A. Mihaila, and Tova Milo. "Querying the world wide web". *Journal of Digital Libraries*, 1 1997 (1), 54-67.