Multi-Tier Agent-Based Architecture For Web Information Retrieval

N. A. EL-RAMLY¹, H. M. HARBO², M. A. ABD-ELWAHED¹, A. M. TOLBA¹
¹ Computer Science and Math Dept.,
Faculty of Science,
Menoufyia University,
EGYPT

² Systems Engineering and computers,
Dept. Faculty of Engineering,
Azhar University,
EGYPT

Abstract- The rapid growth of the network-centered (Internet and Intranet) computing environments requires new architectures for information retrieval systems. Typically, in these environments, the information resources are dynamic, heterogeneous and distributed. In addition, these computing environments are open, where information resources may be connected or disconnected at any time. This paper presents a Multi-Tier Agent-Based Architecture For Web Information Retrieval. The architecture includes five types of agents: Interface, Broker, Domain, Mobile and Information agents.

Keywords: Software Agents, Information Retrieval, Search Engine, Mobile Agent.

1 Introduction
The problems of information retrieval on a large and distributed scale have become highly apparent during the last six years. The information explosion that has taken place, particularly with the exponential growth of the World Wide Web, has shown that users treat information, especially hyperlinked information, as a valuable resource. However, the electronic information community is already showing signs of what has been commonly termed “information overload” [4].

Currently, internet search engines are one of the commonly used tools for information retrieval. Although this engines are of valuable services, they are attached to several limitations. For example, it only provide the user with the address of the information (location) rather the information itself. The search for information is limited and typically biased towards indexing more “popular” information, and each search engine covers a small portion of information resources on the internet [7]. The task of information retrieving and integration is the user responsibility.

Recently, there is a growing interest in using intelligent agents’ approach for designing systems that assist users on the WWW. Because of the flexible and dynamic characters of intelligent agents, they are being used widely as an interface system between the user and the WWW for different applications. For example, an agent that finds web pages for the users, helping users to browse the WWW or assisting the user on scientific literature search [8].

The main objective of this work is to build an information retrieval system that is capable of helping users to locate and retrieve information from distributed resources in an open environment. The main design principles of the system are that the system should: 1) provide the users with an integrated view of information, 2) provide relevant information to the users interest, 3) pro-actively search for information, 4) monitor any possible changes to the information resources, 5) provide information to the users within a bounded-time, 6) be aware of the new information sources that may enter and old information sources
that may exist the environment. This paper presents a multi-agent system architecture that is build to accommodate the design principles.

2 Web Information Retrieval

With the fast growth of the internet, more and more information is available on the web and as a result, web information retrieval has become a fact of life for most internet users. However, compared with classic information retrieval, web information retrieval systems are faced totally different data sets. The uniqueness of web information retrieval is listed as the following [7,8].

- **Bulk**: The bulk size of the internet is approximately 900 million documents as measured on July, 2000 which is growing at the speed of 20 million documents per month.
- **Dynamic Internet**: The Internet is changing everyday while most classic Information Retrieval systems are design for mostly static text databases.
- **Heterogeneity**: The internet contain a wide variety of document types pictures, audio files, text, and scripts etc.
- **Variety of languages**: The types of languages used in the internet are more than 100.
- **Duplication**: Copying is another important characteristic of the Web, as claimed that nearly 30% of the web pages are duplicates.
- **High linkage**: Each document averagely has more than 10 links to other pages.
- **Wide Variance in Users**: Each web user varies widely in their needs, expectation and knowledge.

We can summarize the problems of existing solutions as follows:

1. There are many different systems providing information, and the user need to have knowledge about the differences to access properly,
2. In case that the user need to search for some information in several sites, connecting and accessing one by one to each of them is boring, time consuming and may imply high cost of communication,
3. Some database system do not allow easy user access to the information they record. They have complex and clumsy interface that insists the user to produce unambiguous request for information,
4. The set of data that represents the best response to a query may be the aggregation of data acquired from distributed, heterogeneous sources.

3 Existing Approaches For Locating Information On Internet

We decided to work on the internet, as it is one of the largest publicity available source of information and it constitutes a good testing ground. The existing solutions for locating information on internet are based on two paradigms: **Browsing** and **Searching** [10]. We classified them from the agent point of view into non-agent and partially-agent based approaches. We review the characteristic features bellow.

3.1 Non-Agent Based Approach On Internet

With the internet having seen an explosive growth in recent years, a number of services have arisen to help users locate and retrieve documents from servers around the world. Examples of this applications (known as Network Navigation Systems) are WAIS, Gopher and WWW. The problem with this systems is that although they allow user to search through a large number of information sources, they provide very limited capabilities for locating, combining and processing information. The load of finding information is still on the user [10].

3.2 Partially Agent Based Approach

According to the two paradigms mentioned before, we have

3.2.1 Agent Helping Browsing:

People feel often “lost or disoriented” when navigating through the WWW; browsing agent like Web Watcher [2], may alleviate this feeling by interactively advising Web users –in the browsing process- about which hyperlink to follow next; it “learns” by observing the user’s reaction to its advice.

The problem with them is that badly designed agents can misguide, constantly making annoying suggestions.

3.2.2 Agent Helping Searching:

Searching is an automated process, where the user only gives his/her requirements and the system will try to find the best matches. Several searching tools (known as Network based Information Retrieval Systems) have been developed. Basically they are of two types:

(A) A client-based search tool that does automated navigation. They work more or less like
a browsing user, but much faster and following an optimized strategy. The disadvantages of this approach are: (1) retrieving information through the internet can be time-consuming, (2) the use of network resources is sometimes considered unacceptable high, (3) waste of resources by transferring redundant information. If two users from the same site perform individual searches there is a possibility of overlapping the search spaces and the same information being returned to each user [3].

(B) Indexing agents: these are the most popular type of agents on the web (e.g Google, Lycos, WebCrawler, InfoSeek) [10]. Indexing agents carry out a massive, autonomous search of the web (scanning millions of documents) and store an index of words from document titles and document bodies. The user can query the agent by asking for documents containing certain keywords [5]. The disadvantages of this approach are: (1) with various indexes available, users must choose the correct one to use. (2) while a centralized index allows users to perform flat searches (i.e. without regard to how the indexed information is organized), it can suffer consistency problems as the amount of data increases. This inconsistency may be acceptable for data changing relatively slowly. For quickly changing data, a centralized index is difficult to manage [3].

in order to overcome the mentioned drawbacks, we propose a completely agent-based solution for information retrieving from the World Wide Web.

4 Agent Communication Language

Agents should be able to communicate with other agents in a common language, which conveys meaning and represents information in a standard format. An example of such standards are the Knowledge Query and Manipulation Language (KQML) and the Knowledge Interchange Format (KIF). KQML is a high-level protocol and language for agent to service and agent to agent interaction and communication, commonly termed Agent Communication Languages (ACL). KIF is a rich language that provides a standard representation of information. It can express beliefs, rules, facts and partial descriptions of functions amongst other things. However, KIF is used for more than just a information representation; it also provides a central language for communication between other ACLs [1]. The advantages of using ACLs and standard information representations are clear; if all agents within the system conform to an interchange standard, then the amount of data translation is reduced and the introduction of translation errors into the data are lessened.

5 System Architecture

Our model presented in this paper focuses on viewing information retrieval as problem independent of its structure, for which a single or a group of agents can participate in an open environment. Our system architecture proposed in this work is a multi-agent multi-tier system architecture, in which each agent is autonomous, goal-driven, cooperative, coordinated, rational and able to communicate with other agents to fulfill the users’ needs. The system architecture includes five types of agents: Interface, Broker, Domain, Mobile and Information agents as shown in Figure 1. At the front end of the system Interface agents provide flexible mediation services between users and the information environment. They interact directly with the users, accept queries and provide results. At the back end of the system, Information agents act as information providers for the interface agents, keep track of the distribution of the information resources and the dynamic nature of the information resources in terms of their contents. At the middle-tier of the system, Broker agents keep track of the existing agents of all tiers and their capabilities to facilitate the appropriate cooperation among them, Domain agent supervises the activities that occur within a domain, Mobile agents are the components within the architecture which can migrate between network nodes. The user interacts with the system through a Graphical User Interface (GUI) to submit queries and get results. The agents communicate with each other by means of high-level agent query language, using Knowledge Query Manipulation Language (KQML) [1]. This approach provides a high system performance when agents retrieve similar information or achieve more than one goal simultaneously, through the interface agents. Through the broker agent, the copes well with new information sources that may join or disjoint the environment. In addition, it is responsible for the scalability of the system as the number of agents increases. This approach also facilitates monitoring any possible changes to the information resources and proactively searches for information, through the information agents.
6 Agents Functionalities

This section describes the functionalities of each type of the agents in the proposed system architecture.

**Interface Agent:** The interface agent is the user’s intelligent interface to the system and allows the user to interact with the information systems environment. The interface agent provides graphical interfaces for the user to submit both queries and the desired constraints, to provide feedback, and to display results. The interface agent accepts queries from the user or from other interface agents. The query is described by a set of words that include the topic of interest representing the goal to be achieved and a set of constraints. The user’s interest includes information ‘quality’ and the desired response time. A goal might be decomposable into subgoals; each might be locally achievable or require interaction with the user and/or other agents. The interface agent generates the solutions to each goal and chooses a solution that best fits the user’s needs. The interface agent learns about the user’s topic of interest and preferences. It also learns about other agents’ capabilities in relation to the user’s topic of interest, the desired information quality and the expected response time, each time they involve in interaction. The interface agent is capable of accessing a local database, storing information for future queries, and maintaining models of the other agents (user and software). Therefore, the interface agent may act as an information provider to other interface agents. The interface agent is able to interact directly with the Information agents or indirectly through the broker agents, for which the efficiency of the system can be enhanced when retrieving similar queries, and preventing the system from collapsing if the broker agent in the middle tier is malfunctioning or disappears.

**Broker Agent:** The broker agent acts as a service provider that pairs interface agents seeking information about specific topics with interface and/or Information agents that are able to provide that information. As the interface and the information agents come online, they advertise their capabilities to the broker agent, using KQML. The broker agent accepts advertisements from other agents to confirm their existence and capabilities, and organizes them into groups based on their topic of interest; The broker agent knows the unavailable agents either by receiving notification messages or by setting an expiration time for acknowledging their existence. The broker agent responsibility has a very advantageous impact on the system’s functionality and performance. Firstly, users don’t need to interact with the broker agents. This supports the transparency aspect of the system. Secondly, the agents are grouped based on their topics of interest. This allows the interface agents to direct requests and messages to the interested agents only; therefore, the interface agent’s knowledge about the network structure and the network traffic required to achieve its goals are reduced. Thirdly, the architecture allows new agents of different types to join the system and existing ones to disjoin. This makes the system operates in an open environment and achieves the scalability of the system by a seamless integrations as the number of agents grows.
**Domain Agent:** A domain agent is a static agent that supervises the activities that occur within a domain; a domain is a logical boundary used to delimit nodes, agents and resources into manageable and distinct entities. The domain agent has a number of responsibilities to information resources, users and agents within a given domain:

* Providing a migration service to mobile agents wishing to leave the domain.
* Providing an authentication and validation check on mobile agents wishing to enter the domain.
* Mediating global access to information resources; before a mobile agent can access an information resource it must obtain permission from the domain agent.
* Ensuring that the domain does not become swamped by mobile agents.

The domain agent, then, is the key force within a domain and is ultimately responsible for ensuring that security is enforced within the domain and that agents can communicate with information resources and other agents.

**Mobile Agent:** Mobile agent is the component within the model which can migrate between network nodes. It is the mechanism by which the user exercises control over their own distributed information resources and gain access to other, shared information resources. Mobile agents are equipped with a set of user-defined goals which describe the nature and limits of their functionality, for example, a resource discovery agent is a mobile agent with a different goal set than, say, a navigation assistant agent. It is probable that, in addition to the limits on the functionality that users place on their mobile agents, the mobile agents themselves will encounter limits (in the form of security, authentication and validation) that exist within domains [5]. In some cases, these limits will compromise the goals that have been given to the mobile agent.

**Information Agent:** The information agent acts as an information provider that has a direct access to the information resources. The responsibility of the information agent is to accept queries submitted by the interface agents and returns the results to them. Due to the dynamic nature of the environment, information resources are constantly changing by, adding new information, deleting old information or modifying existing information. The information agent is able to handle event notifications on the information resource updates. In addition, it monitors any changes in the information resources based on a predefined timeframe. This capability allows the information agent to provide the interface agent with an appropriate set of information resources that carry up-to-date information. The Information agent receives queries during interaction from other agents expressed in a common query language (KQML). The information agent is able to construct the received query from a common query language into a language of the information source. Once the query is constructed, the information agent sends the query to the information source for execution, and constructs the answers back into the common query language. The Information agent accepts a query similar to that of the interface agent, which consists of a set of words describing the interface agent’s topic of interest and the associated constraints. The constraints might include the quality of information and the response time at which the information should be available.

**7 System Characteristics**

The basic characteristics of the agent-based information retrieval system are as follows [9].

- **Autonomous and Intelligent:** An autonomous agent is within and part of an environment and acts on it, over time, in pursuit of its own agenda and so as to affect what it senses in the future [9]. We easily see that the broker agent satisfies these requirements. We can also say that it is intelligent since it has the ability to decide, can think on behalf of the user. Its actions can change over time as agents in the system change their behavior.
- **Learning:** We shortly discussed the questions broker agent can respond. This could be possible with learning. When the broker agent first gets on-line, it knows nothing but with control messages it sends to other agents, it begins to retrieve information and in a short time it becomes fully informed about the status of the entire system. Its information base is up to date since these control messages are sent periodically [2].
- **Communicative/collaborative:** The system consists of multiple agents that try to respond to user requests by communicating and working together [4].
Flexible: Besides the flexibilities provided to the user, the system can easily be expanded with the integration of new information agents.

8 Advantages Over Classical Information Retrieval Systems

The system has many advantages over classical information retrieval systems. Traditional system rely on a central database while this system has a distributed database architecture. Utilization of category information minimizes network usage and search time. Even if a user line is broken before he gets the results, he does not need to make the search again because system holds the result for him. When he is online again he will receive the results. The user can also register the request so that at a later time, he can retrieve the results. When he issues a request, he does not need to wait for the results and can start a new search before his prior requests are finalized. This is not possible in classical systems. Because of this system architecture arranged as a multi-tier model, it takes all advantages of multi-tier application such as scalability, load balance and if any agent in any tier fail the system remains intact.

9 Agent Architecture

Agent-oriented technology provides the next step in the evolution of computational modeling, programming methodologies and software engineering paradigms. The first principle of agenthood is that an agent should be able to operate as a part of a community of cooperative distributed systems environment, including human users. In this paper, we used a Generic Intelligent Software Agent (GISA) model, as shown in Figure 2. In our view, Every agent has a State given by the values of the internal variables, which can be modified according to some Rules based on some specific Knowledge. The Agent "sees," "acts on" and "talks to" the world via an Interface, part which handles the medium and low levels of communication. All this is linked together by some Glue Code. It is important to notice that there is no clear cut between the different components, one can easily "hide" some tasks of the others.[8]

9.1 The GISA State

The State of an agent is what makes it act differently for similar inputs. It represents somehow the "history" or "memory" of an agent. Going down to the implementation level the state is nothing more than the set of the values of all the variables at a certain moment. every agent has a Mental State composed from beliefs, commitments, intentions, and capabilities. This Mental State can be modified according to different rules and inputs. Beliefs, intentions, and commitments can be added or deleted. Capabilities are usually non-modifiable, unless the agent can learn.

9.2 The GISA Rules and Knowledge

The behavior of an agent can be described in a set of rules, which also encapsulate some knowledge about the outside world, maybe simple models of other agents or of the world (i.e. ontologies). These rules can be formulated as simple IF..THEN..ELSE constructions in an imperative language, or as complex sets of declarative rules handled by an inference machine. The more elaborated the set of rules is, the more intelligent our agent is [6]. An agent able to learn would be able to modify it's behavior, the rule base.

9.3 The GISA World Interface

The Interface should handle every interaction between agent and environment. This includes message (knowledge) translation from internal representation to a common language understood by the other agents in the world, or in other words, the interface has to know what to send. Another important feature of this component
is to know how to find other agents, thus to know not only what to send but also where. Finding other agents is done in practice by accessing a known yellow pages server. The conversation can be either routed through the server, or performed peer to peer after finding the needed address. Different architectures handle this differently [6]. In the case of mobile agents, the Interface is also responsible for agent replication or migration.

9.4 The Glue Code

This part links all the components together. In a good framework the developer shouldn't be concerned about writing this part, this being automatically generated. Sometimes it is hard to make a clear distinction between Glue Code and all the other components.

10 Conclusion

In this paper, an agent-based system architecture is presented as a solution to the problem of information retrieval in WWW. The system architecture is layered into five types of agents: Interface, Broker, Domain, Mobile and Information agents. This paper described the functionalities and design of the agents in detail. It has been shown that the proposed system has some fundamental advantages over similar classical systems, providing the user various flexibilities. The system is also expandable and tolerant to failures.

11 References: