A Market-oriented Agents-based Model for Information Retrieval

Djamel Eddine MENACER[1], Christophe SIBERTIN-BLANC[2], Habiba DRIAS[3]
[1]*National Computer Science School of Algiers (ESI)*
Algiers, Algeria.
[2]*Université Toulouse 1 - Capitole*
Toulouse, France
[3]*Houari Boumedienne Sciences and Technologies University of Algiers (USTHB)*
Algiers, Algeria
d_menacer@esi.dz; hdrias@usthb.dz; sibertin@irit.fr

ABSTRACT

Information Retrieval in the World Wide Web (Web IR) is essential for a number of activities and it is an active domain of research and development. The main challenges concern the relevance of the results provided to users' queries and the performance regarding respond-time. On the other hand, agent-based market systems prove to be efficient for implementing e-commerce or B2B applications on the internet, thanks to inherent properties such as prominency of interactions, scalability, flexibility, interoperability, etc. Although the use of agents in other application domains is not yet widespread, the integration of mobile agents into market mechanisms bring clear and efficient solutions to Quality of Service issues encountered in most distributed applications and notably in Web IR systems. Mobility allows defining the *seller – buyer model* of interaction, where agents act on behalf of final users or devices providing resources, while the generic *Market Place architecture* provides an organizational setting for the matching of demands and offers. The paper shows how this framework applies to Web IR and provides experimental validation results from a Jade implementation

KEYWORDS

Agents, Mobile Agent, Information Retrieval, Seller–Buyer model, Market Place Architecture, Jade.

1 INTRODUCTION

Most of the current systems for locating information on the World Wide Web, known as Web Information Retrieval systems (Web IR), rely on the use of *search engines* which manage and attempt to keep up-to-date indexing information by a variety of tools based on spiders, web crawlers, *etc* [1]. These engines are then queried by users to locate and find information on particular topics.

The main issue in IR systems is to *quickly* return the *relevant* information to end-users. The relevance and the performance become then the most important requirements in IR systems. In order to optimize the relevance, many approaches have been proposed, such as the personalization of requests [2] and the semantic Web [3]. However, these approaches are not yet feasible because they are hard to implement.

The use of centralized engines in IR systems is a drawback that creates bottlenecks in the search for locating information. The growing size of the information to be indexed and the processing power required to serve search requests jeopardize the suitability of the search engines technology to meet the needs. At any moment, a given search engine is estimated to cover no more than 40% of the web in its database [1]. To perform an exhaustive search, the user must employ several search engines and assume that each one has access to a different 40% part. To avoid the bottleneck problem, indexes need to be distributed.

In [4], the authors suggest, as a possible means for achieving this end, to use mobile agents that wander across the web in a directed fashion for seeking the information on behalf of users. The proposed scheme, called *AgentSeek* system, involves three types of mobile agents:

- *ferrets* which act on behalf of web searcher users, seek for information providers

and advertise the location of information consumers
- *publicists* which act on behalf of web site creators (people providing information), advertise the location of information providers and seek information consumers
- *gurus* which facilitate encounters between ferrets and publicists

However, the proposed scheme uses specific concepts, ferrets, gurus and publicists that cannot be applied to other systems.

NetSA [5] is a multi-agents system for the IR on heterogeneous distributed sources. This system comprises essentially the following agents:
- *User agents* that collect and filter information from and to the clients
- *Broker agents* that associate the requests to agents which are able to respond to them
- *Resources agents*, which are linked to an information resource (internal or external) and are able to update the data

However, the NetSA system is a static multi-agent system.

Calvin [6] is a multi-agents system that provides the following agents:
- *Calvin Web,* an interface agent;
- *Analysis agents (TFIDF, WordSieve* and *DocStats)* that perform analysis of the users' profile and behavior;
- *Research Agents* **(***AltaBot* and *GoogleBot)* that perform profile-based searches for the users.

The Calvin system is also a static multi-agent system.

In [7], an interesting study suggests a components-based approach including mobile agents in order to simplify the development and the deployment of adaptable information retrieval systems in the context of distributed heterogeneous peer-to-peer networks. This promising proposal uses mobile agents as a solution to the deployment problem.

In [8], the authors discuss the use of mobile agent technology as an enabler for open distributed e-Health applications. More precisely, they describe experiences based on this technology concerning emergency scenarios. The authors show that the use of mobile agents in Medical Information Retrieval in Mass Casualty Scene is very beneficial in terms of performance.

In [9], the authors argue that one of the best means for getting services or finding particular information on a network is the use of Jade mobile agents [10] together with a Web interface that connects users and resources in a transparent, open and scalable way. The authors argue that the deployment of Jade agents eases the development of applications thanks to its open source-code, interoperability with other agents, availability and easiness to use.

In line with these technical proposals, the paper proposes a novel integrated mobile agent-based approach for IR in the WWW.

The remainder of this paper is organized as follows. Section 2 outlines our proposition based upon mobile agents and market-oriented interaction model. Section 3 describes a new mobile agents' model, the *seller – buyer model,* while section 4 describes its implementation through a generic mobile agents-based framework, the *Market - Place architecture*. Section 5 presents how to perform IR tasks by means of market mechanisms and how to apply the MP architecture to IR applications. Section 6 presents the *MP-IR platform*, a jade implementation of the framework and section 7 provides an experimental validation. Finally, section 8 concludes the whole paper.

2 MOBILE AGENTS FOR WEB INFORMATIONN RETRIEVAL

The importance of the quality of service (QoS) in distributed applications (non functional aspect) is becoming critical. The quality of service includes non functional aspects such as [11]: the performance, the security and the safety of functioning (or reliability). The QoS depends also on the distributed application.

In Information Retrieval (IR), the performance should be more important than security and reli-

ability. The performance can be measured by the relevance and the response time.

In order to improve the performance in IR, we should give answer to the two main issues regarding IR systems:

- Using distributed indexes instead of a centralized index
- Improve the relevance

In order to distribute indexes, we can use agents provided with the mobility capability. Each agent may convey one index. Moreover, the users' requests can also be conveyed by *mobile agents* which can act on behalf of users.

In order to improve the relevance, we propose to generalize the market interactions paradigm, which proves its suitability in market applications such as e-commerce, to non-market applications such as IR. In market applications, a set of services or goods is proposed by servers and requested by client users. A *client agent* is delegated by a user to look for the location and availability of a service (so-called a *buyer-agent*) while a *server agent* is delegated by remote services to sell a service (so-called a *seller-agent*). A seller-agent is intended to propose items or services to buyer-agents. Both agents interact according market mechanisms, such as negotiation and competition, in order to achieve the intended service.

Mobile Agents (MA) are software agents [11] with the feature of *mobility*. The MA can be used in distributed applications to reduce the bandwidth consumption in the network and allow disconnected operations. MA represent a good idea to implement IR applications. However, the security problem blocks their development. Indeed, MA is target to two types of security threats:

- Attacks during the migration (on the network), also called exogenous attacks. This kind of attacks can be solved by traditional security means.
- Attacks within the host that receive the MA. This is known as endogenous attacks. This kind of attack lies on the possible existence of malicious hosts that can tamper a received agent. There is no complete security solution to this kind of attacks.

A MA needs a specific execution environment on each of the sites that constitute its itinerary. The execution environment is provided by a *MA platform*. Another issue in using MA is the need of *similar* MA platforms in each node in the network. Hence, many efforts are made to provide interoperability between MA platforms: FIPA [12] and MASIF [13] are the two most known standards.

Therefore, if we use MA in Web IR, we must take into consideration to the following issues:

- Security issue in MA
- Interoperability between MA platforms

The security of MA is an important concern. In IR systems, security is not required as QoS. In our proposition, agents should convey users' only requests and indexes. Therefore, the security of agents is not a critical issue in agents-based IR systems.

Finally, our proposition should be independent of the MA-platform's standard (FIPA or MASIF) used in the nodes of the network (Internet for example).

3 THE SELLER – BUYER MODEL

MA are a good way to implement distributed applications. However, a direct use of MA is not recommended due to their security and interoperability issues. To *avoid* these issues, we have built an extended Mobile Agent (MA) model towards a more suitable model, the Seller-Buyer model (SB) [14].

In SB, there are two kinds of MA: buyer-agents and seller-agents. Both agents have just to meet on dedicated sites called *market places* (MP). In a MP, a buyer agent can meet several seller agents that offer similar service.

In this scheme, a buyer agent should visit multiple MP (in order to optimize its satisfaction), thus it must be mobile. Selling a service does not require mobility but a seller agent must move in the two following situations:

- At the creation of the agent, it must migrate on an appropriate place.
- When the current place becomes less profitable, the seller agent should migrate to a more profitable place.

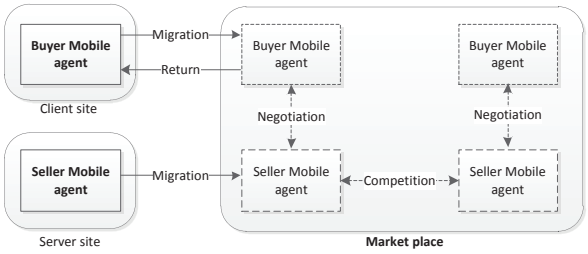

Fig. 1. The seller-buyer model

In SB, both client and server processes are mobile. This is called *service mobility*. Figure 1 shows an overview of the SB model.
Finally, seller agents should only carry the **minimum resources** from their providers. In order to complete the service rendering, seller agents can run remote invocations with their providers. This is the key of service mobility in SB.

3.1 Security in the SB model

In the SB model, the security model is based upon *trust*. Mobile agents should move only to trusted nodes. This is done by preventing an agent from migrating *directly* to a host and a host from receiving mobile agents. This is possible if we dissociate the rendering of services and the hosting of visiting agents; in fact, a mobile agent (buyer agent) representing the *client* meets, on *trusted sites* (market places), a service *provider* representative (the seller agent). Mobile agents (buyer and seller) may migrate *only* on *market places.*

Although our proposition does not require security, the SB model allows reducing the security issues in the MA model, by removing the endogenous attacks as shown in the figure 2. A complete survey of the security in SB model is presented in [15].

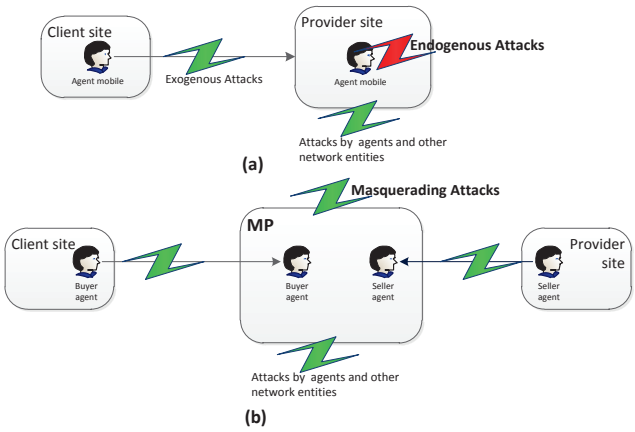

Fig. 2. Improvement of the MA (a) security by the SB model (b)

3.2 The SB model for multi-agents systems

The SB interaction model extends market mechanisms to distributed systems. Buyer agents perform *negotiation* with seller agents in the MP so that the seller agents are in *competition*. The negotiation is based upon a price p that can be the QoS of the requested service. Facilitator agents are used to ease the migration of MA. Figure 3 shows the different interactions within a *SB multi-agents system*.

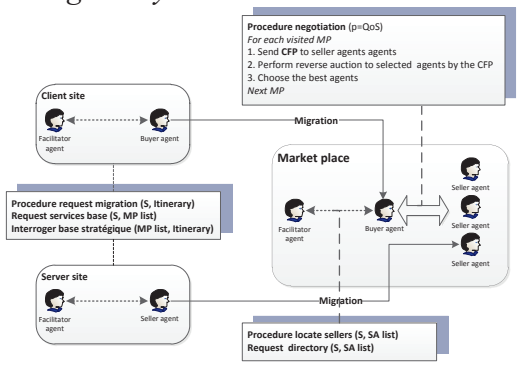

Fig. 3. The SB multi-agents system

4 THE SB MODEL FOR DISTRIBUTED APPLICATIONS: THE MP ARCHITECTURE

We now propose a general framework based upon the SB model for the development of mobile agents-based distributed applications. We will call this framework the *MP (Market - Place) architecture.* This architecture addresses the following objectives:

– to provide a general multi-agents framework based upon the SB model, regardless of the agent platforms used to manage agents

– to provide mobile agents with a protection based upon *trust*
– to distribute the provided services by means of seller agents
– to provide acceptable level of QoS by market interaction between agents

4.1 MP Components

There are two *external* actors considered by the MP architecture: *clients* that send requests by means of buyer mobile agents and *providers* that offer services at MP by means of seller mobile agents. The basic idea is that each service *S* in a MP system belongs to a *class of services SC* and each class of services SC belongs to an *application domain D* (see figure 4).

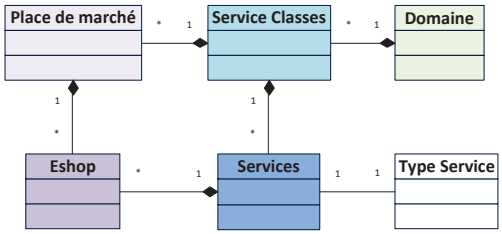

Fig. 4. Class diagram of the service organization in MP architecture

The MP architecture consists of several components as shown in the figure 5:

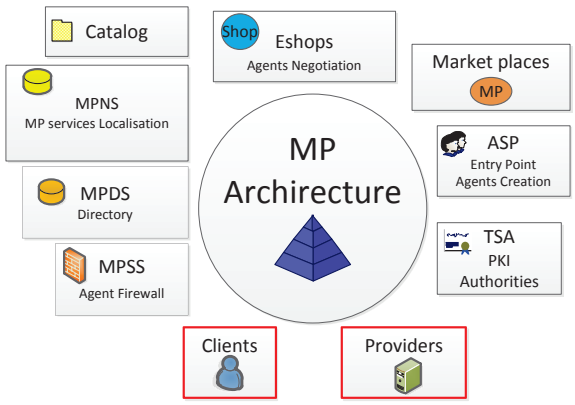

Fig. 5. Overview of the MP architecture

- **Market places (MP).** In order to give mobile agents a *directed way* to request a service, the MPs are organized according to the *class* of the offered services. One MP hosts one service class. The services are located, within the MP, in **e-shops** [16]. Each e-shop hosts one service. Therefore, the dialogue between buyer agents and seller agents takes place in e-shops. The information on e-shops and services are stored in directory services provided by the MPDS (MP Directory Services) managed by the MPSM (MP Service Manager). Each MP is protected by a firewall called MPSS (MP Security Service).
- **Agent Service Providers (ASP).** This site is responsible for the creation of mobile buyers or seller agents, according to the type of user (client or provider). We refer this site to as Agent Service Provider (*ASP)* [16].
- **MP Name Servers (MPNS).** When a mobile agent requests or offers a service *S* that belongs to the class *SC*, it searches MP providing the class *SC*. To do this, the agent sends a request to MP Name Servers (MPNS). The answer is a list of MP that offers the class of service *SC*, constituting the *itinerary* of the agent.
- **Trust and Security Authorities (TSA).** A mobile agent must be *certified* before it visits MPs. We propose to add *PKI* (Public Key Infrastructure) components [17] to the architecture. The ASP provides a pair of keys (private, public) to mobile agents by means of a cryptography service. The PKI *certification authorities* are hosted in sites called Trust and Security Authority (*TSA)*.

4.2 The dynamics of the MP architecture

Each request of a client user is associated to a specific service S that belongs to a service class SC. This request is linked to a buyer mobile agent created in the ASP site. Before migrating, the buyer agent asks the MPNS for an itinerary for the SC class of service. The buyer agent obtains its private and public keys from a local cryptographic service. The buyer agent registers to the TSA by its public key and obtains a certificate. Almost the same process is applied to the seller agents that represent providers. We can represent the dynamics of the MP architecture through the diagram shown in figure 6.

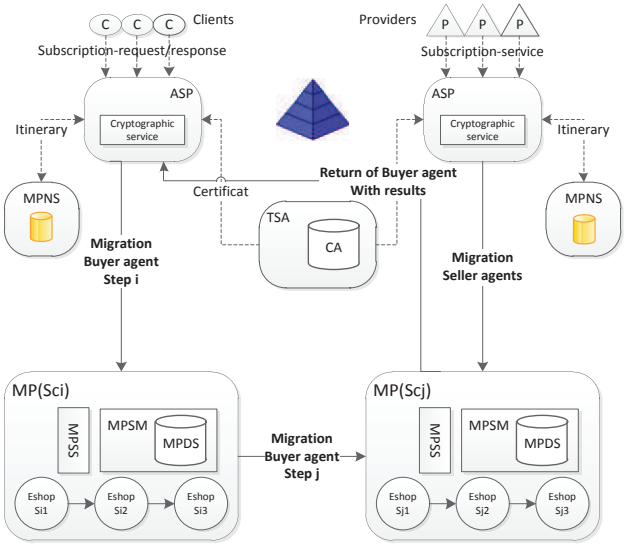

Fig. 6. The MP architecture dynamic

4.2.1 Agents in the MP architecture

There are two types of agents: mobile agents that comprise buyer agents and seller agents, and static agents that comprise manager agents and facilitator agents. Manager agents manage different sites in the system: ASP, MP. For MP, the manager agent acts as the MPSM. Facilitator agents are used to ease the migration of mobile agents [18]. There is one facilitator agent at: ASP, MP and e-shops. The figure 7 shows the hierarchy of agents' classes in MP.

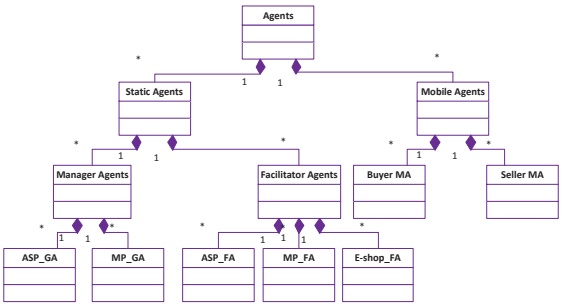

Fig.7 . Agents' classes' hierarchy in MP

4.2.2 The MP migration model

The *ASP facilitators* help the buyer and seller agents to migrate to the MP sites by providing them an itinerary obtained by querying the MPNS servers. The address of the facilitators is included in the itinerary provided by the MPNS server to the mobile agent. This kind of migration is called *external (or global) migration*. The protocol of the global migration is shown by figure 8.

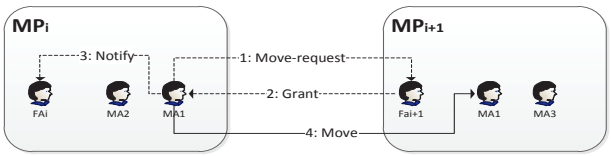

Fig. 8. The global migration of agents

The *MP facilitators* help buyer and seller agents to migrate on the e-shops within the same MP by using the MPDS service. This second kind of migration is called *internal (or local) migration* and is similar to the migration process provided in FIPA platforms as Jade [10]. The protocol of the local migration is similar to global migration. Finally, the *e-shop facilitators* help buyer agents to locate seller agents within the e-shop.

4.2.3 The MP interaction model

The negotiation between a buyer agent and seller agents takes place in e-shops that comprises one or more seller agents. The FIPA *Contract-Net* [19] interaction protocol is used to implement negotiation between agents by using CFP (call for proposal). The *initiator* of CFP is the buyer agent, and the seller agents are the *participants*. The CFP allows getting a list of interested seller agents. Then, the buyer agent has to choose the appropriate seller(s). To this end, it starts a new negotiation through a *re*verse auction mechanism. The e-shop becomes then an auction room. The algorithm 1 describes how a buyer agent interacts with the seller agents of an e-shop:

/* CFP process */
1. The buyer agent issues a *call for proposal* by sending a *CFP* message to all seller agents;
2. The seller agents interested by the CFP answer the buyer agent by sending a service offer;
3. The buyer agent selects one (or several) of the sellers having sent an answer;
4. The buyer agent sends its request to the selected seller agents;
5. The selected seller agents answer the buyer agent with a *proposition*;

/* reverse auction process */

6. The buyer agent defines the wished (and hidden) price *wp*
   /*price between a maximum and a minimum prices*/
7. For each *round* in (1..MaxRnd) do
   /* MaxRnd is the maximum number of rounds allowed */
   a. While *number of iterations* ≤ MinIt do
      /* MinIt is the minimum number of iterations */
      Selected seller agents send public propositions to the buyer agent;
      End while;
   b. if *wp is lesser than all the propositions* then
      Seller agents are invited to decrease their propositions;
      Else

The buyer agent selects the first lesser proposition
Exit; /* End of auction */
End if;
Next round;
8. End of auction /* The auction ends if the maximum round number is reached without results with seller agents */

Algorithm 1: The MP interaction model

The figure 9 shows the different interactions within the MP multi-agents system.

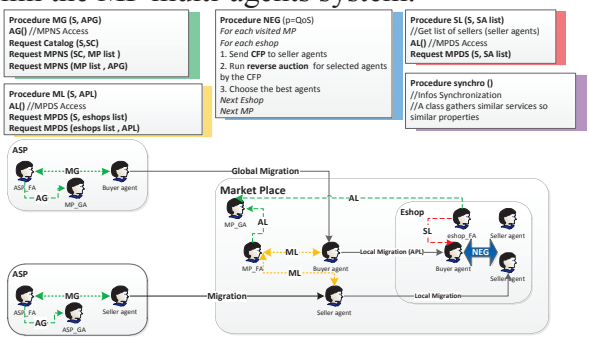

Fig. 9. The MP multi-agents system

5 APPLICATION TO THE INFORMATION RETRIEVAL

The MP architecture is aimed to support any distributed application. We will now show how it operates for IR in the WWW. As other IR systems our approach should also use distributed indexes, user's agents, provider's agents and facilitator's agents. However, it relies upon the generalization of the MP architecture market mechanisms to IR systems. To do this, we define the application domain "IR" as including the class of service "Search Category", and the service as "Search theme". For example: D=IR, SC=general, S=general, for general purpose search and D=IR, CS=IT, S=software, for specific search.

5.1 Adaptation of the MP architecture to the IR

Each MP becomes a **search place** and corresponds to a search category. The e-shops become **negotiation rooms** and host a search theme.

A buyer mobile agent is a *search agent* or *metasearch* and acts on behalf of a user. A seller agent owns an index and a search code, and can be considered as an *index agent* which acts on behalf of a provider. It is reasonable that an index agent carries only an index of a search theme S. To facilitate the users' searches, we propose a *distributed index* through index agents.

According to the search category *SC* of the request, the metasearch agent asks the MPNS server for an itinerary that comprises a list of places belonging to SC. After migration, the metasearch agent meets the index agents in the e-shops corresponding to the search theme *S* and located in a place that belongs to the search category *SC;* the metasearch agent can then ask several index agents, merge and filter the different results and return the best result to the user.

A metasearch agent holds a code (called SEARCH code) used to express the request according to a representation model. The SEARCH code is matched with the index. If the seller agents do not use the same representation model, the SEARCH code must be included in the seller agent that becomes a search engine agent; in this case, the metasearch agent just carries the request as a set of keywords and becomes a light metasearch agent (figure 10).

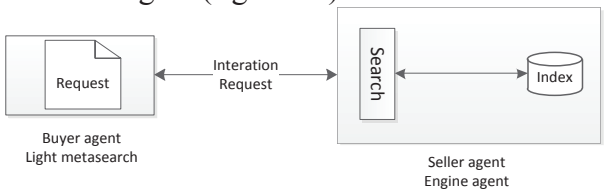

Fig. 10. Light metasearch and search engine agents

If the representation model of the seller agents is the same, the SEARCH code must be implemented on the metasearch agent that becomes a heavy metasearch agent; in this case, the seller agents become a just index agent (figure 11).

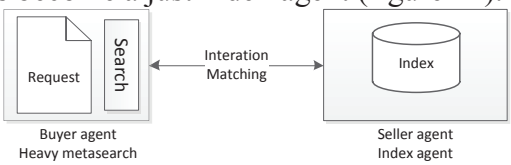

Fig. 11. Heavy metasearch and index agents

5.2 The IR Negotiation protocol

In negotiation rooms, the negotiation process must decide how much the client's request may be satisfied by the index agents. Several index agents can offer the same theme S but with different *qualities of service (QoS)*. The QoS in IR applications is mostly measured by performance. The performance can be evaluated according to

the *relevance* and the *size* of the returned results. We assume that each index agent is able to return, in addition to results, the average *relevance* and the *size* of these results. The negotiation process, based upon the relevance R and the size of the results S, can be summarized in the algorithm 2:

For each MP in the metasearch agent's itinerary
  For each e-shop in the MP visited by the metasearch agent
  /* CFP process */
1. The metasearch agent makes a call for proposal by sending a CFP message (SC, S) to all index agents present in the e-shop;
2. Each index agent interested by the CFP is added to the *selected* index agents
3. The selected index agents answer the metasearch agent by sending a service offer; /* Ns is the number of selected agents */
  /* Reverse auction process */
4. The initiator of the auction, the metasearch agent, defines the wished (and hidden) price that reflects the Rmin relevance and the Szmax size of the results corresponding to the search request.
5. The metasearch agent sends its request (SC, S, (k1, k2,..., kn)) to the selected index agents (matching) ; /* ki are the keywords */
6. For each round in (1.. Rdmax) /* Rdmax is the maximum number of rounds allowed */
   a. While number of iterations j < Jmin, (1<Jmin≤Ns) /* Jmin is the minimum number of iterations */
      Each selected index j (0≤j≤Jmin-1) sends public proposition (SC,S,Rj,Szj) to the metasearch agent;
      End while;
   b. if (Rj<R or Sj>Sz ∀j) then
      Index agents are invited to decrease their propositions for another round (decrease Sz and/or increase R);
      Else
      the metasearch agent selects the three most suitable propositions (the answers that feature the maximum relevance R, the minimum size Sz and the auction ends.
      Exit; /* End of auction */
      End if;
  Next round;
7. The metasearch stores the results (SC,S,(url1,url2,...,urlm),Sz,R) in its memory; /* urli are the URL of the relevant documents */
  Move to next e-shop;
Move to next place;

Algorithm 2. The interaction protocol between metasearch and index agents within IR e-shops.

5.3 The MP-IR framework

According the considerations above, we are able to outline a MP-based architecture for Web Information Retrieval systems. We refer this architecture to as *MP-IR*. Figure 12 shows an overview of MP-IR, where NEG represents the IR interactions protocol defined by the algorithm 2.

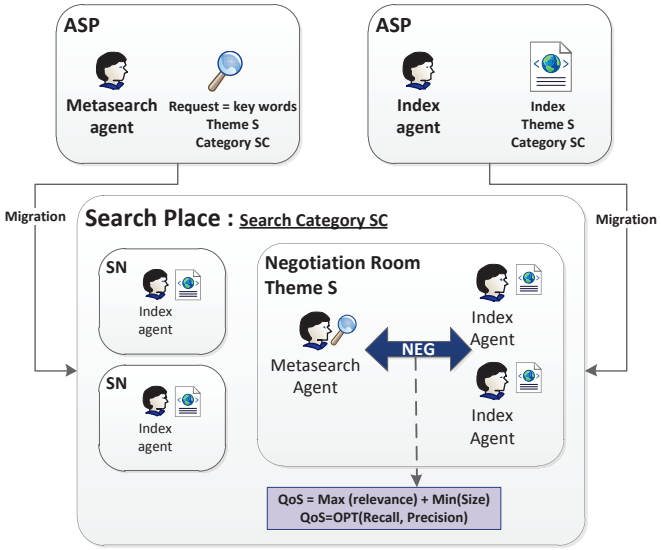

Fig. 12. Overview of MP-IR architecture

6 A JADE IMPLEMENTATION OF MP-IR

6.1 Implementation of the MP architecture using jade and Java

Jade [10][20][21] is a free and open source platform for the development of FIPA agents-based systems.

The MP-IR architecture can be implemented as a set of Jade platforms distributed over several computers in a network. A market place is a set of computers including a jade main platform server and one or several jade platforms without main container (known as containers) servers that implement e-shops. The ASP is a main Jade platform and the other MP components (MPNS and TSA) may be java services. As a result, such an implementation of the MP architecture is called **MP-Jade** framework. This framework uses Jade for agents' management and java as programming language (figure 13).

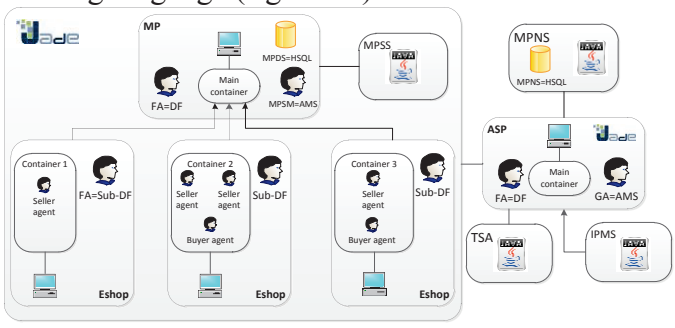

Fig. 13. The MP-Jade framework

The MP-IR architecture can then be implemented by the MP-Jade framework.

6.2 Agents in MP-IR

Every agent inherits the *Agent* class of the package *jade.core.agent*. The tasks of each Jade agent are called *behaviours*. Jade allocates one thread for each agent. Each jade platform is controlled by the AMS (Agent Management System) agent. Information about agents which are available on the platform is provided by the DF (Directory Facilitator) agent.

6.2.1 Static agents

The static agents implement a *cyclicBehaviour* (a repetitive behaviour issued from the class *CyclicBehaviour* of the package *jade.core.behaviours*) since they run repetitive tasks.
The DF agent of the main platform (for example a market place) can act as MP facilitator agent. The AMS agent of the main platform manages the places and can act as the MPSM agent.

6.2.2 Mobile agents

- The *metasearch agents* may have an advanced decision autonomy model. We think that the Jade finite state machine (FSM) model is suitable for this type of agent. FSM are instances of the class *FSMBehaviour* of the package *jade.core.behaviours.FSMBehaviour* and can implement behaviours. The *request* of the client is included in the behaviour of the agent.

- The *index agents* may also implement a finite state machine but lighter than those of the metasearch agents because it do not performs multiple migrations as metasearch agents do. The *service (code+index)* of the provider is included in the behaviour. At its arrival in a place, an index agent registers its services to the sub-DF agent of the appropriate e-shop. Finally, an index agent can interact remotely with its provider site by sockets.

7 EXPERIMENTAL VALIDATION

In this section, we will show the experimental tests we did in order to validate our proposition. The tests address the relevance of the results it provides to requests. For this purpose, we have developed two IR systems. The first one is a classical IR system based upon Terrier [22]. Terrier is a highly flexible and efficient open source search engine, used in large collections of documents. Terrier is a complete and transparent java platform for research and experimentation in the text retrieval. The second IR system is MP-based IR system using market interactions according to the algorithm 2 (see figure 14). We have also used two benchmarks: the first one is a huge benchmark taken from a collection of XML documents called INEX 2005 [23] that contains 17 000 items, a set of requests and a list of relevant documents for each request. The second one is a personalized benchmark called *corpus* that contains a set of documents, a set of requests and a list of relevant documents for each request. The corpus we used contains 500 documents and 40 requests.

The relevance is measured by two factors: the precision and the recall [24]. The recall measures the ability of the system to retrieve all relevant documents. The precision measures the ability of the system to retrieve only relevant documents and reject all irrelevant documents.

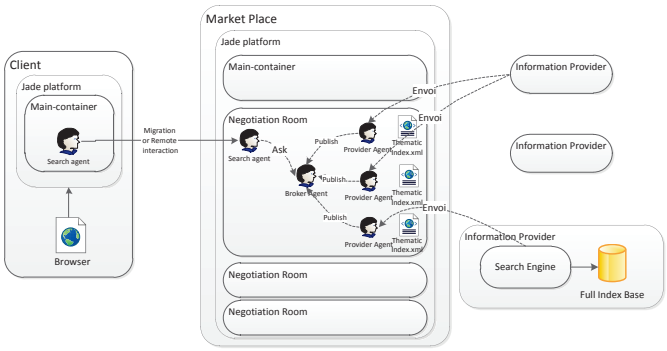

Fig. 14. Overview of MP-IR Jade prototype

7.1 Basic Tests

We perform our tests on the Inex benchmark. In order to evaluate the quality of our system, we have chosen assessment files in the Inex collection containing requests and corresponding relevant documents. To do our tests, we have chosen six requests (***Q1..Q6***) as shown in the table 1.

Table 1. Chosen Inex requests

Q	*Requests*	Assessment files
1	*Problems physical limits miniaturization microprocessor*	*206. xml*
2	*Mining frequent pattern itemset sequence graph association*	*209. xml*
3	*Gibbs sampler*	*213. xml*
4	*User-centered design of web sites*	*217. xml*
5	*Computer assisted composing music notes MIDI*	*218. xml*
6	*Capabilities limitations commercial speech recognition software*	*221. xml*

- **Using category and theme search**

We calculate the recall and precision measures corresponding to the requests ***Q1***…***Q6*** for searches with and without consideration of category and theme (see table 2 and figure 15). We have fixed the following parameters:

- *Rdmax (see algorithm 1) = 1*
- *Number of providers = 3*

Table 2. Recall-precision with and without category and theme

	Basic search		*Advanced search*	
Request	**Recall**	**Precision**	**Recall**	**Precision**
Q1	0,38	0,13	0,85	0,66
Q2	0,45	0,27	0,75	0,42
Q3	0,28	0,7	0,2	0,88
Q4	0,2	0,18	0,72	0,53
Q5	0,4	0,11	0,67	0,5
Q6	0,5	0,24	0,84	0,57
Average	0,37	0,27	0,67	0,59

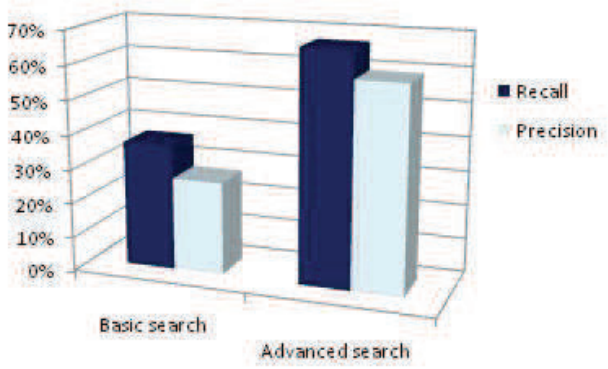

Fig. 15. Recall-precision measures with and without category and theme

We can notice that the recall/precision values of the advanced search are better than those of a simple search. This is due to the fact that the number of relevant documents in a given category and theme is higher than the number of the relevant documents in a general collection.

- **Varying number of negotiation rounds**

We study the impact of the number of negotiation rounds on the relevance. We have calculated the recall/precision measures for the requests ***Q1***…***Q6*** by varying the number of rounds from 1 to 3 (see table 3 and figure 16). We have fixed the following parameters:

- *Number of providers=3*
- *Advanced search (choosing category and theme)*

Table 3. Recall-precision in function of the number of rounds

	1 round		*2 rounds*		*3 rounds*	
Request	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**
Q1	0,85	0,66	0,92	0,66	0,92	0,78
Q2	0,75	0,42	0,9	0,42	0,9	0,47
Q3	0,2	0,88	0,4	0,75	0,2	0,92
Q4	0,72	0,53	0,72	0,57	0,72	0,57
Q5	0,67	0,5	0,8	0,47	0,7	0,41
Q6	0,84	0,47	0,8	0,5	0,84	0,5
Average	0,67	0,58	0,76	0,56	0,71	0,61

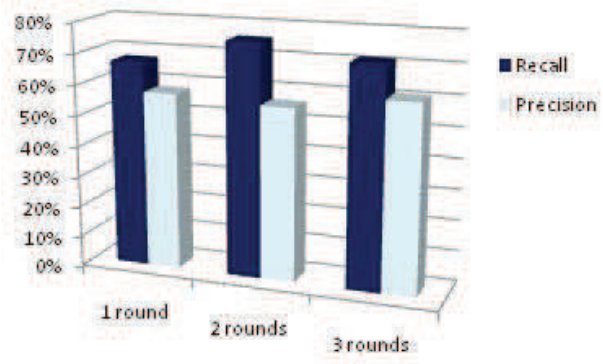

Fig. 16. Recall-precision in function of the number of rounds

We can notice that there is an optimal *Rdmax* in which the recall-precision is the best. In our case, when *Rdmax*=2, the recall-precision is optimal.

- **Varying number of providers**

We study the impact of the number of providers on the relevance. We have then calculated the recall-precision measures for the requests ***Q1***…***Q6*** by varying the number of providers from 3 to 7 (see table 4 and figure 17). We have fixed the following parameters:

- *Rdmax* = 1
- *Advanced search*

Table 4. Recall-precision in function of the number of providers

	3 providers		*5 providers*		*7 providers*	
Request	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**
Q1	0,85	0,66	0,9	0,61	0 ,8	0,55
Q2	0,75	0,42	0,84	0,43	0,86	0,4
Q3	0,2	0,88	0,27	0,9	0,27	0,9
Q4	0,72	0,53	0,76	0,48	0,85	0,56
Q5	0,67	0,5	0,9	0,41	0,9	0,55
Q6	0,84	0,47	0 ,8	0,51	0,85	0,53
Average	0,67	0,58	0,61	0,56	0,62	0,58

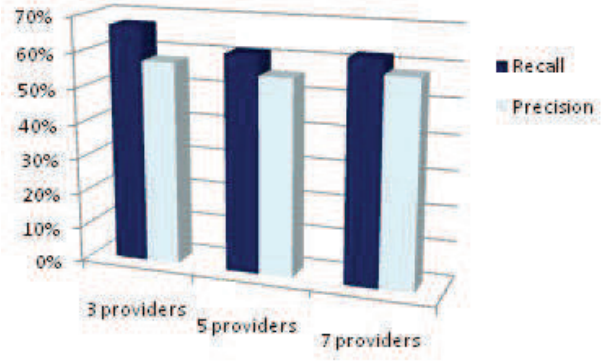

Fig. 17. Recall-precision in function of the number of providers

We can notice that when the number of providers grows, the recall-precision reaches first an optimum and then decreases. In our case, the optimum is with *3 providers*.

7.2 Comparison with classical IR system

7.2.1 Using the Inex benchmark

Using Inex benchmark, we now compare the precision and the recall of both systems. We have fixed the following parameters:

- Advanced search
- Rdmax=1
- Number of providers = 3

The measures we did are based upon the recall and precision curves. Table 5 summarizes the results and figure 18, shows the recall-precision curves. Globally, it is interesting to note that, although Terrier is better, both curves are almost similar.

Table 5. Precision/recall values of both systems

	MP-IR		*Terrier*	
Request	**Recall**	**Precision**	**Recall**	**Precision**
Q1	0,85	0,66	0,9	0,4
Q2	0,75	0,42	0,33	0,4
Q3	0,2	0,88	0,15	0,95
Q4	0,72	0,53	0,5	0,6
Q5	0,67	0,5	0,85	0,21
Q6	0,84	0,47	0,92	0,41

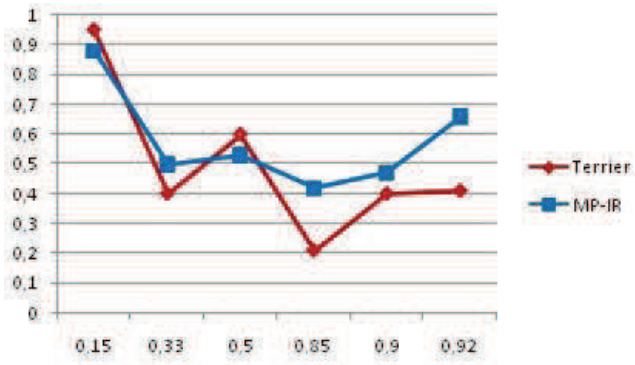

Fig. 18. Recall-precision curves of both systems

7.2.2 Using the personalized benchmark

We also did our tests on the personalized benchmark. Both systems use the same corpus. The table 6 summarizes the average precision depending of the recall for the 40 requests.

Table 6. Average precision depending on the 11 recall levels

	Standard recall levels (%)				
	0	10	20	30	40
Terrier	69.96	55.45	48.67	31.48	22.07
MP-IR	60.51	50.44	43.92	29.17	20.05

Standard recall levels (%)					
50	60	70	80	90	100
16.03	12.65	10.88	9.12	5.55	1.75
15.00	11.64	10.88	9.12	5.55	1.75

The figure 19 shows the average precision curve using the 11 standard recall levels for the 40 requests. Globally, it is interesting to note that, although Terrier is a bit better, both curves are similar.

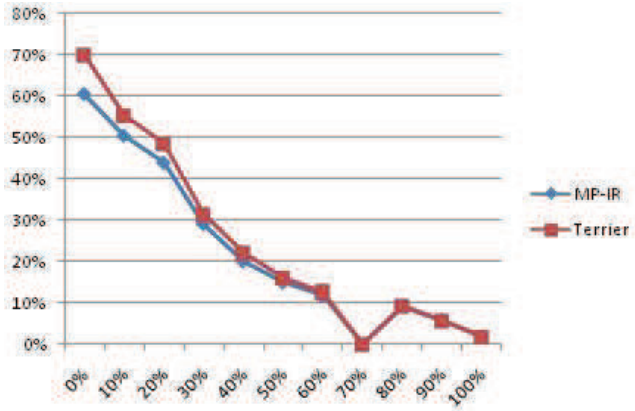

Fig. 19. Average precision of the Terrier and MP-IR

8 CONCLUSION AND FUTURE WORKS

There is an increasing need for World Wide Web Information Research systems to offer a very high level of relevance. However, while

the volume of information increases, the index bases grow and the relevance of the documents returned to users' requests tends to dramatically decrease. Many approaches have been proposed to improve the relevance but still do not satisfactorily succeed.

Using MA to distribute indexes and to convey users' requests is a good idea. However, MA address the issues of security and interoperability. To answer those issues, we have proposed a novel MA interaction model, the SB model, in which buyer agents meet seller agents only in market places, and developed a global architectural design called MP architecture based upon the SB model. To achieve our proposition, all interactions between agents are based upon market mechanisms such as negotiation and competition. Finally, we apply MP architecture to IR systems. The IR framework based upon MP architecture is called MP-IR.

The MA approach reveals to be efficient to distribute indexes: indexes are managed by search engine agents (or index agents) and users' requests are conveyed by search agents (or metasearch agents). In this approach, we proposed to implement the IR process, notably the matching between the users' queries and the indexed sources of information, through market mechanisms that create competition between index agents. Therefore, search agents and index agents meet in market places and interact by means of competing negotiation. We have proposed an algorithm of negotiation based upon CFP and reverse auction. The negotiated price is supposed to be the QoS (to optimize) of the application, thus the relevance (to maximize).

The experimentations show that our approach has given similar results as those of well-known classical IR systems. In fact, the idea to distribute indexes by means of mobile agents and implement the IR process by means of market interactions between agents is promising and can give better results if we improve our algorithm 2. We believe that integrating competition and negotiation in the IR process will give better relevance.

In further work, we intend to complete the MP-IR prototype by improving the algorithm 2 and to perform deeper experiments against classical IR systems by using the whole Inex collection.

REFERENCES

1. Hock, R.: Web search engines: Features and commands: Search Engine Section, Online, Vol.23, No.3, pp.24-28 (1999)
2. Liu F. and Yu C.: Personalized web search for improving retrieval effectiveness. In IEEE Transactions on knowledge Data Engineering, volume 16, pages 28–40 (2004)
3. Feigenbaum, Lee, Ivan Herman, Tonya Hongsermeier, Eric Neumann, and Susie Stephens: "The Semantic Web in Action." Scientific American, vol. 297, Dec. 2007, pp. 90-97 (2007)
4. Grey D.J., Dunne G. and R.I. Ferguson: "A Mobile Agent Architecture for Searching the WWW", Scientific Commons, paper available under: http://en.scientificcommons.org/42430563 (2007)
5. Coté M., Chaib-Draa B. and Troudi N.: "NetSA: A reusable multi-agents architecture for rich information environments", Cépadues-Edition (2002)
6. Bauer T. and Leak D.B.: "Handling Complex Information Environments: A Multi-Agent Framework », *Proceedings of the Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2002 at AAMAS*02)*, Bologna, Italy. Editors.: Paolo Giorgini, Yves Lespérance, Gerd Wagner, Eric S. K. Yu, Publisher : CEUR-WS.org, Series: CEUR Workshop Proceedings, Volume 59 (2002)
7. Leriche S., Arcangeli J.-P.: Flexible Architectures of Adaptive Agents: the Agentphi Approach. In: International Journal of Grid Computing and Multi-Agent Systems, Serials Publications, Vol. 1 N. 1, p. 51-71, january 2010 (2010)
8. Vigilson Prem M., Swamynathan S.: "Role of Mobile Agent in Medical Information Retrieval in Mass Casualty Scene – A Performance Study in Web Environment". WSEAS Transactions on Information Science and Applications, ISSN: 1790-0832, Issue 10, Volume 8, October 2011 (2011)
9. Effanga E.N., Asuquo D. E. and Williams E. E.: "Information Retrieval using Jade Mobile Agent System". World Journal of Applied Science and Technology, ISSN: 2141 – 3290, Vol.3. No.1, pp 106-117 (2011)
10. Tilab. Jade official site: http://jade.tilab.com [Accessed in July 2012]
11. Coulouris G.F., Dollimore J., Kindberg T. and Blair G:. Distributed Systems, Concepts and Design. Addison-Wesley, 5th Edition (2012)
12. FIPA. Official site of the Foundation for Intelligent Physical Agents. Obtained through the Internet: http://www.fipa.org, [accessed 15/06/2009].

13. Milojicic D., Breugst M., Busse I., Campell J., Covaci S., Friedman B., Kosaca K., Lange D., Oshima M., Tham C., Virdhagrisswaran S., White J.: « MASIF – The OMG Mobile Agent System Interoperability Facility », In Proceeding of the 2nd International Workshop on Mobile Agents, Ed. by K. Rothernel & F. Hohl, pp. 50-67, Lecture Notes in Computer Science, No. 1477, Springer (1998).
14. Menacer, D. E.; Drias, H.; Sibertin-Blanc, C.: The MP Architecture: Towards a Secure Framework for Mobile Agents. *International Journal of Agent-Oriented Software Engineering*, Volume 4, Number 4, November 2011, pp. 390-414 (2011)
15. Menacer, D. E.; Drias, H.; Sibertin-Blanc, C.: Towards a security solution to mobile agents. In Proceeding of the 2013 World Conference on Information Systems and Technologies (WorldCIST'13), Ed. by Á. Rocha, A. M. Correia, T. Wilson, K. A. Stroetmann, pp. 969-979, Advances in Information Systems and Technologies, No. 206, Springer (2013)
16. Wang Y., Tan K. L. and Ren J.: A Study of Building Internet Marketplaces on the Basis of Mobile Agents for Parallel Processing. Journal of World Wide Web, Volume 5 Issue 1, pp. 41-66, Kluwer Academic Publishers Hingham, MA, USA (2002)
17. Adams C. and Lloyd S.: Understanding PKI: Concepts, Standards, and Deployment Considerations. 2nd Edition, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA ©2002 ISBN:0672323915 (2002)
18. Xu H. and Shatz S. M.: A Framework for Model-Based Design of Agent-Oriented Software. IEEE Transactions on Software Engineering (IEEE TSE), January 2003, Vol. 29, No. 1, pp. 15-30 (2003)
19. Hsieh F-S.: Analysis of contract net in multi-agent systems. Journal of Automatica, vol. 42, no5, pp. 733-740, Elsevier, Kidlington, UK (2006)
20. Bellifemine F., Caire G. and Greenwood D.: Developing Multi-Agent Systems with JADE. Published Online: 20 FEB 2007. DOI: 10.1002/9780470058411.ch6. Copyright © 2007 John Wiley & Sons, Ltd. (2007)
21. Fortino G., Garro A., Mascillaro S. and Russo W.: Using event-driven lightweight DSC-based agents for MAS modeling. International Journal of Agent-Oriented Software Engineering, Volume 4, Number 2/2010, pp. 113-140 (2010)
22. Terrier, http://www.terrier.org [Accessed in July 2012]
23. Inex. http:// code.google.com/p/inex/downloads/list [Accessed in August 2012]
24. Turpin A., Scholer F.: User Performance versus Precision Measures for Simple Search Tasks. Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR'06, pp. 11-18, ACM New York, NY, USA (2006)