# A Search Engine based on Multi-Agents-System

*Lecturrer* **Noora A. Al-Saidi**[*]

## Abstract

A multi-agents system (MAS) is a system composed of multiple interacting agents. MAS is used to solve difficult and impossible for an individual agent or monolithic system to solve. Usually these problems are decomposable to sub-problems, each one will be the duty of one or more agents.

In this paper a new approach for the search engine is proposed depending on a MAS. The main functional subsystems of the engine such as crawler, indexer, updater, and query builder are implemented as agents. The proposed search engine based on multi-agents system, SEMAS, is designed to provide fulltext, up to date database, query builder, and general search engine web application. The agents in SEMAS are communicating via simple syntax agent-communication facts (ACF). SEMAS was tested by a population of web documents of 1000 web sites of various subjects. The results show that SEMAS provide accurate ranking depending on the visual effect of the page, the up to date database, and ease of query tuning obtained by the query builder. Furthermore many software agent features are attained such as reusability, distributed problem solving (DPS), distributed artificial intelligence (DAI), and parallelism.

[*] AL-Rafidain University College

## .1- Introduction

Using the internet has adapted an idea to use WWW which is the World Wide Web. Then it becomes the most popular tools of use to surf the internet, because of many factors. These factors depends on the richness of information in the internet page, and using sounds, videos and more colours has leads using the internet from many people[1].

Because of the increasing of using the internet and the growth of the numbers of websites, the searching and find such information from websites becomes more difficult. Therefore it raise the need of tools to ease the access of these information , and this tools called search engine which is a powerful engine of use to find sites and graphics or whatever the users need[2].

The search engine is tool to find a list of results from their database. It works as a unit from many layers levels starting from crawler, indexer, searcher and ranker. When a word or group of words called, the crawler is the first use of these words and it make sure that there is no more than one time use of the page can be crawled. After saving the page in the database, it goes to the next level called indexer processing and tests the text (extracting keywords and their attributes and occurrences). The last level the ranker it chooses the high probability elected depends on their weights.

As a result of that it shows the need to build a system which can use many agents and make them work in homogenously way with each other to get accurate solution.

And because of the speedy way of developing software technology, it becomes using and programming the tools software much greater. However when discuss of using new agent-based on the building software environment. It becomes more important to find the kind of experience that the creator has to make software which can sense and interact with the software system that going to use it for the future [3]. Therefore the agent is a system or program works independently to do very specific tasks in visual or real environment. And these agents behave independently on behalf of the owner in the whole the world. Like an example the use of it in (networks, expert systems, information retrieval, and others) [2, 4, 5].

That is why we need a multi-agents system which consisted of two or more to solve complicated programs and find reasonable solution in the searching tools like MAS, and the example of using it in online trading [6], disaster response[7], modelling social structures[8], e-mail monitoring[9].

Therefore the agents in multi-agent system have many important characteristics [10].
1- autonomy: the agents should be kind of independent or partially autonomous.
2- local views: there is no perfect agent for full global understanding of system or any kind of complexity that make it practical use in this knowledge.
3- decentralization: does not exist any type of agent could practically reduce monolithic system.

The agent's works with any system by sharing information between each other and using any kind of communications protocol to do it. In this research will discuss a specific way of formatting massages passing mechanism to use between any system and agent for the type of sharing knowledge.

## 2. Aim of the research

The research is based on different factors which is as below:
1- Create a search engine which should work with full text of entry for any site registered in their database.
2- Make a new agent application that share and works effectively with the search engine and has all the important applications which are (Crawler, Indexer and Updater).
3- Build a mobile agent as query builder installed in the client computer to reduce the trafficking and the communication protocol processing in the main server computer. And adjust the type of format query that could be sending to the server applications (the searcher).
4- Build a repository for the web search engine has genuine databases system.

It can show that building web search engine has more complexes and more variety needs to be able having a good search engine, which we are going to explain it in block diagram in detail.

## 3. Proposed Search Engine Design

The website search engine is a continent of many applications working as one package called Search Engine based on Multi Agents System (MASSE) and linked to huge source of databases called repository. There are two applications represent the search engine which are (Windows and web

Application). The window represent an agent application while the web is for the on line application, as shown in figure (1).

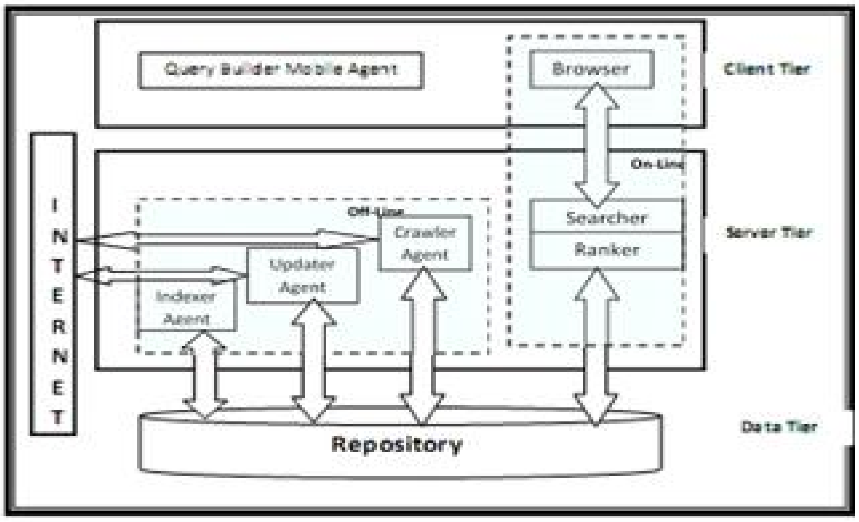

Figure (1): SEMAS Block Diagram

The main structure of the SEMAS consists of (Client, Server and Data tier). And in addition there is the Web application which has two parts (searcher and ranker), while the Agent application branch are (Crawler, indexer and updater), with one mobile agent called (query builder). And this research is going in to details as follows.

## 3.1 **The Core structure of the** SEMAS.

### 1- Client Tire

The client tire is a link between the user and the mobile agent throw the browser interface.

A. the browser:

The browser as (the internet explorer, Mozilla, Safari, etc) are applications runs from the user side to be able calling any site through a server. Therefore when the user inter a website address, the browser send this address to the server and the server provide the page site then send it back to the user. But before see the site in the user side, the document should be decoded using application like Java or Visual Basic script to be able seeing the site in html or xml languages.

When the user calls for a web site in the SEMAS (search engine) as (http://myserver/afm/index.htm); the web server user sends the address and adding the Query builder mobile agent embedded with it. And as a result it should be seen the MASSE home page in the client user side.

B. Query Builder Mobile Agent:

When the user calling a website, the server application program send back all the information requested to the user. But with the query builder mobile agent which is a mobile codes and used as a tool send with any address.

The purpose of using the query builder can be explained as below:

1- Avoid interning the URL queue website if it has been called for the first time and this can make a Crawler agent crawl once. After that it can
be added as a form in the URL queue in the (SQL) the Structured Query
Language Statement.

2- Make the right kind of the query which called from the user, and make it in the suitable correct form, then send it back to the user server prototype.

Therefore the use of the mobile agent is to speed up the processes of all the application that been used by the server applications. And in Figure (2) it shows the activities of sharing the data betweenthe server applications and the mobile agent.

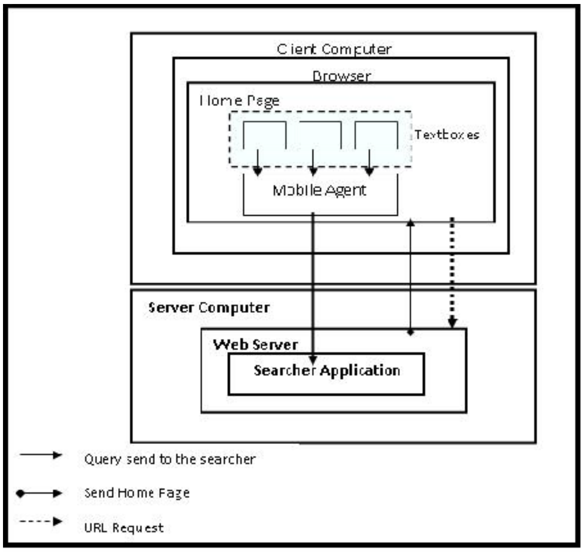

Figure (2): Data flow between mobile agent and server application

However the SQL works as a database (Repository) which should be used as stored procedure queries. While some of it could be dynamically generated by the Query builder mobile agent as shown in figure (3).

```
Algorithm (1):(Input: text word(s) entered from user )
              ( Output: send query to searcher)
1- Construct the use query as SQL query
2- If the query is previously processed then   Return its result.
3- Else
4-     If it is a stored procedure then   Invoke stored
       procedure.
5-     Else   Apply the query by search application
6-     end if
7-   end if
```

Figure (3): Query builder agent algorithm

- **Server Tier**

The server tier is an application which runs from the server computer during a request from the client computer browser. This server application has two types of program which are searcher and ranker.

**1- The Searcher:**

The searcher get request from the client computer which is the (Query builder mobile agent). Then the query builder gets the information needed from the user side and generates SQL form. and then send it back to the searcher.

The searcher will then have the result from the repository after analyzing the query and pass it to the ranker. The ranker would adjust the best results and send it back as HTML code form to the client.

On this level the searcher will return the Word_id after comparing it with the query. And will be extracted in details to:
(a) Page_id. (b) Position. (c) Style. (d) Color. (e) Font name. (f) Size.(g) Occurrences of this word in the page.

All these type of information can be stored in temporary place as buffer to be used. And if there are more than one word then it would be treated depends on the type of operators as shown below in table [1].

***Table (1) : Operator Effect Table***

| Operator | Effect |
|---|---|
| NOT | Select all pages that did not contain the term, NOT works with on term only. |
| NEAR | Select all pages that contain term1followed by term2. |
| AND | Select all pages that contain term1 and term2. |
| OR | Select all pages that contain term1 or term2. |

**2- The Ranker**

The ranker send the highly recommended pages to the user and will not pass any other pages not related to the word that been called. And this result can be reach by using the ranking algorithm. Table (2) elucidates the notation used in the ranking algorithm
.

***Table (2) Notation Used In The Proposed Algorithm***

| **Notation** | **Description** |
|---|---|
| Word.No | The number of word in the page. |
| W(i) | the weight value of word i in the page. |
| p.w(i) | the weight value of the word I in the page |
| oc.w(i) | the occurrence of the word (i) in the |
| cl.w | the weight value of the color |
| s.w | the weight value of the font size. |
| st.w | the weight value of the style type of the font. |

This algorithm is used by the SEMAS as Visual Effect ranking algorithm. Therefore when a word called, then this word will be tested against many factors like (appearance, position and occurrences). After that the ranker will weight the word relay on a calculation used by equations as below:

$$T.W=\sum_{i=1}^{word.No} W(i)\ldots\ldots\ldots\ldots\ldots 1$$

$$W(i)=p.w(i) + appearance\_w(i) + oc.w(i) \quad \ldots\ldots 2$$

$$Appearance\_w(i)=cl.w + s.w + st.w \quad \ldots\ldots\ldots 3$$

Some of the symbols used above in the equations is going to be explained as tables and show every symbols and it is value in the calculation used by the Ranking Algorithm.

```
Algorithm (2)
1- List.page= stored all the pages that match with the query
2- T.W=0
3- M=0
4- For all pages contain the keyword do
5-     Fetch page P
6-     T=0
7-      For i= 0 to Word.No in P do
8-         W(i)=p.w(i)+cl.w(i)+s.w(i) +st.w(i) + oc.w(i)
9-          T=T+W(i)
10-     end {for}
11-     T.W(M)=T
12-     M=M+1
13- end {for}
14- L=sorted list of pages sorted in descending order dependingon T.W
15-Return(L)to the client
```

Figure(4)Visual effect ranking algorithm.

- Data Tier (Repository)

The repository is kind of storage contains the WebPages that has been downloaded from the crawler. And it will arrange all the pages in order by enhanced indexing system. And the table below show the repository content.

1- URL Table:

Table (3) shown below evaluate the way that the crawlers use the pages and the type of information can be taken through this process.

Table (3) URL Table

| page_id (unique) | page_URL (full address) | File_type (.html or .htm) | Crawled (as a flag) | crawl_date (date) |
|---|---|---|---|---|

The table has five levels explained as below:

- Page_id: it is a unique number which identified each individual page.
- Page_URL: it is full identification for the webpage.
- File_type: it is the type of page been called from the user and that could be (HTML, PDF, etc)
- Crawled: it is unique flag to be attached to the crawled page to be identified from other new WebPages could be called later.
- Crawl_date: it is the date of the crawling.

Therefore this operation would reduced the times of processing when the user call a page from been repeated, relaying on the updater agent.

## 2- Lexicon and Stop_Word Table:

As shown in table (4) the lexicon table using the connections between words and entries.

| Table (4) Lexicon Table | |
|---|---|
| word _id (unique) | Word (stemmed) |

- Word_id: we need to reduce the size of the word by using numbering each word and these identifications compared with the page_id in the index table. Then the id can be used rather than the exact word.
- Word: the original word that been called at the first time.

Stop_Word the table which has words may not relate with the real content of the user. Therefore if it is added to the unwanted words then it will not appear in the original searched page.

## 3-Inverted Index Table:

It shows in the table that every file or text must go through process to make data index. then it will build a structure of mapping the data for the content need to be indexing. And these help to speed up the search and calculated by (Term-Based Ranker).

| Table (5) Inverted Index Table | | | | | | | |
|---|---|---|---|---|---|---|---|
| Page_id (unique) | word_id (unique) | position (offset) | Importance | style | Color | Font name | size |

- Page_id: it make a unique number to use it as id which can be found in the Word_id field.

- Word_id: it use identification number for every word and identified it in the Inverted_index table for size reduce.

- Position: is the place of the exact word in the web page. And the number reflects the number or words in the page.

- Importance: the importance of the words depends on their location.And that is mean if the word was in the Title of subject, then it will get high propriety.

- Style: it mean relay on the word style (Bold, Italic, Underline). Therefore each style has unique identified number. When the style word is Italic that is meaning the value is 2.

Table (6) Page_Information Table

| page_id (unique) | Page_ title | author_ name | Keywords | descriptor | modification_ date | publishing_ date |
|---|---|---|---|---|---|---|

- Color: it is the font color for the word used.

- Font name: it is the font face for the word used.

- Size: it is the font size for the word used.

**4-Page_Information Table:**

This table has the seven fields as it shown in table (6).

- Page_id: as explained in table 5

- Page_title: the title can be taken either from the <TITLE> or the <META> tag. But when the identity names the same as title then the content can be represent as a title. An example of that <META name=”title” content=”Web Developer.com guide to building intelligent web site with JavaScript”>

- Author_name: the author name is generated from the <META> tag when it is identification name equals author. Then the content of the identified content will be the web author name. Example for that<META name= “author” content=”Nigel Ford”>

- Keywords: the keywords is taken from the <META> tag when it matching the same sign of the content for the webpage. Then it can be the right keyword for that webpage topic.

- Descriptor: the descriptor should be taken from the <META> tag as it should compare with the name of the page title. After that the descriptor could illustrate the mean topic of the webpage.

- Modification_date: it means the date of the last modification has been developed for the webpage to be loaded to the crawler. It can be concentrated from the <META> tag when its characteristics related to (update, modfdate, or other words related to modification_date). Therefore this character could be designate for the modification_date.

- Publishing_date: it means the first date of the webpage when loaded for the first time to the crawler. It is concentrated from the <META> tag when its characteristics related to (pubdate, publishdate, or others words related to publishing_date). After that this character could be designate for the publishing_date.

**5-Page_Form_Information Table:**

Table (7) below has five departments.

Table (7) Page_Form_Information Table

| page-id (unique) | number of words (No of words) | Color | face | Size |
|---|---|---|---|---|

- Page_id: as explained recently.
- Number of words: it is restricted to the present and calculated the numbers of different words in the current webpage which is the (Page_id).
- Color: produce the conventional font color for the present Page_id.
- Face: interpret the conventional font face for the present Page_id.
- Size: illustrate the commonplace font size for the present page_id.

**6-Content Table:**

This table (8) has just two departments as shown below.

Table (8) Content Table

| page-id (unique) | Page_content (Full text document) |
|---|---|

- Page_id: as it was explained before.
- Page_content: this department accommodate the text of the webpage.

These database designed by Microsoft SQL Server 2000.

## 2.1.2 Agent Applications of the Proposed System (Off-line Part)

This section is a process that works independently at the end user. It lies on different parameters of applications which are (crawler, indexer and updater) agents. And these agents undertake the downloading page, update databases and extract keywords, as going to be explained as follows.

- **Crawler Agent**

The crawler is also known as spider web works automatically or manually to register or copy all the WebPages visited, and keep them for the later use, to speed up processing of the search engine and provide fast searcher. However crawling is more easy to use when it becomes using the site-specific search engine, as the developer has wide and variety of access to these sites. And we are going to show the functions of the crawler in figure (5), as it contain three parts.

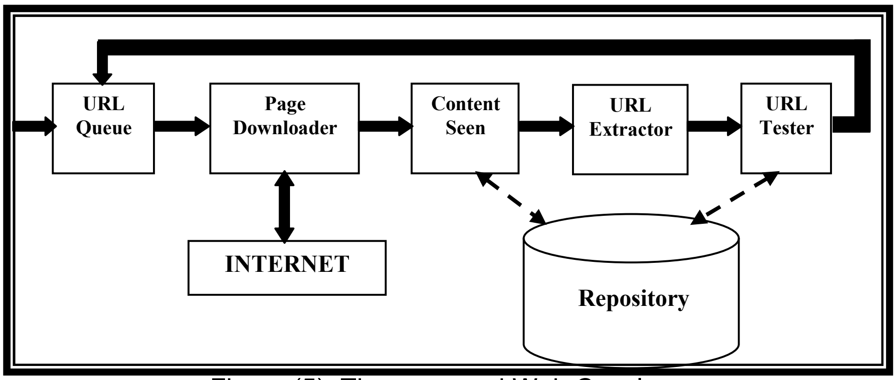

Figure (5): The proposed Web Crawler.

**1-URL Queue:** This elemental use the mobile agent to get the URL then passes it in the alternate storage queue. And this queue has no Occurrences, considering every URL in this queue is new and not visited. All this process made through the mobile agent and the URL tester.
**2-Page Downloader:** The page downloader is element which copies the page content of the related URL and sends it back to the user. And it dissolves the URL and finds the right process for downloading the page. An example for that is the image cannot be proceeding while the HTML or Portable Document File (PDF) documents is been downloaded.

**3-Content Seen:** The content seen is tool to avoid referencing the same URL, hence this tool correlate the page content and check the repository through the content table. Then restrain having more than one copy in the indexer agent for the page content, if it is the same in two URL. However it will keep the two URL of the same page as a copy.
**4-URL Extractor:** The URL extractor is element to test the text of the webpage and find the URL links of it, then send it to the URL tester components.
**5-URL Tester:** When crawling a page it is extracting from the URL tester and must not inspect again. Any new URL must be send to the URL queue for crawling and any visited URL would be abandoned. The crawler is designed to performance manually or automatically, as it shown in figure(6).

```
Algorithm (3): Input: URL Queue. output: Page Content Table
1-  For all URL in URL queue do
2-      dequeue URL depending on priorities
3-      download the content of URL
4-      if the pages contain new URLs add URLs to URL queue
        using content seen test
5-      If the page is a new page save its contents and its id in
        content table repository.
6-              Retrieve all URLs in this page
7-              Retain the unvisited URL only
8-              Inqueue the unvisited URLs in the URL queue
9-      Else insert the URL information record in the repository.
10-   End {if}
11-  End {for}
```

Figure (6)Crawling Algorithm

- **Indexer Agent**

The indexer agent is working in the search engine without instructions, therefore the indexer agent proceeding information, while the URL in the URL table is not indexed. And this agent is the sophisticated program as it keeps every word of information for particular web page. The diagram is shown in figure (7) explain the indexer.

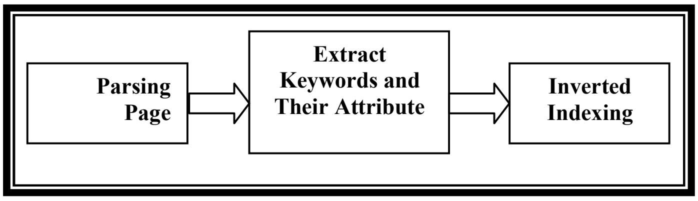

Figure (7): Indexer Agent.

In this research, we appreciate indexer is considered. Considering of having the right webpage, then it must having kind of superiority of selection of information, to be able having definitive of characterize the contented of the indexed web pages. Therefore reinforce of indexer reads a reasonable text file from the web page, and this will explained as follow:

## 1- Page Parsing:

By analyzing the process of the Enhanced indexer, it takes every page from the repository and makes it as a text file that has the most information of the HTML tags. And these tags are important to work together with the indexer to reach the exact information. Then we end of having perfect characteristics for all the words in the indexed page.

## 2- Extracting keywords and their attributes:

In the HTML page it can be establish from the tags as a representative (tokens). Therefore the word processed by the simple stemmer to reduce the size of the lexicon to step up the search engine functioning, and the tags operation in a later stage. And also by wipe the noise words from the token stream, by filtering using the negative dictionary that has a stop words. In addition these two agents (lexicon and negative dictionary (stop_word) are part of the search engine indices.

## 3-Inverted Indexing:

The final step is algorithm (4) control the token. Therefore if the token is tags then the handling by tag_Process (token), as it analyze the characteristic of relevant words. And there are different processing steps because it depends on the tags type.

However if the token is not a noisy word (mean not stop (token)). In this case the word descriptor indication will be selected, therefore the word (word_id, identification number) and the same word in Lexicon (return to table 3), will be retrieve from lexicon and stored in the Inverted Index Table in the Search Engine Indices. After that the structure among the word_id and Page_id in the Inverted Index will be completed.

```
Algorithm (4): **Input**: tokenized page. **Output**: Inverted Index table
1-  while not file.eof do
2-    if tag(token i) then    tag_process(tokeni)
3-    else   if word(tokeni) and not stopped(tokeni) then
4-              store(tokeni,attribute)
5-              move next token i
6-     end if
7-  end while
```

Figure(8):Token processing Algorithm

The information that used and selected in this step has been arranged in to three sectors:

## A. Word Descriptor Attributes:

These words are reserved in the inverted index table after been evaluated from the indexed word independently. The attributes are: *Word Position, Word Style, Word Importance, Word Font Color, Word Size, and Word Font Face.* The configuration for the Inverted Index Table will be illuminated in Table (6).

## B. Page Information Attributes:

other classification characters are selected and saved through the index running interval processing. Other saved in Page_Information Table, and these references are: Page Title, Author Name, and the Keywords use in the web Page, Descriptor Terms and Publishing Date. The Modification Date is the formula of the Page_Information as shown in Table (7).

## C. Page Descriptor Attributes:

There are many other attributes processed and saved in the Page_Rorm_Information during the indexing, and it is all part of the web page. These attributes are:

The general font face, the number of words, the general font color and the general font size. All these characters are explained in the table (7), as it collaborator to classify the ranking score literally in Term_Based Ranking phase.

- **Updater Agent**

It works as the Updater Agent enabled automatically to search for updates from an updated URL. The agent can automatically update it as needed to communicate with the updated URL in each period of time or the update could be a requested from the website. In either way it will be updated if there are any changes in the publishing or updating date. The updater show how it works in algorithm (5)

```
Algorithm (5):
Input: URL for test to update  Output: The page either updated or not.
1-  while there is a request to update do
2-    if page_has_publication_date  then
3-        if pub_date_insaved_page<pub_date_in_online_page then
              crawl this page again and index it.
4-        Else    this page does not need updating.
5-        End { if}
6-    Else    do content_seen_coparison
7-    if no match in comparison then crawl this page again and index it.
8-    Else    this page dose not need updating.
9-    End{ if}
10- End{ if }
11- End {while}
```

Figure(9):Updating processing Algorithm

## 4. The Results of Proposed Search Engine

The search engine approved from the huge database of websites, and could be tested through the client and user computers server. Then we get these results:

1- The Crawler agent crawls only the new web pages have never been visited before and never crawling more than once, and this operation reduce the crawling.

2- The mobile agent increasing the performance of the search engine by accommodates the system of build the sufficient query.

3- The updater agent keep the system up to date of any changes in the web pages, then no crawling more than one time.
4- Overcome the time spend of the query by having a good save proceeding in the SQL server.
5- The MASSE database gets updated by new websites through the registration page.
As a result of that we have been tested the search engine inside closed network (intranet) due to a group of users and table (9) is show the result below.

Table (9) Population Table

| People Type | Quantity |
|---|---|
| Beginner users of Internet | 30 |
| Expert users of Internet | 15 |
| Programmers | 10 |
| Web application programmer | 6 |
| SQL server 2000 programmer | 5 |

And then from the use of 1000 website through the users, we can show the web documents database as shown in table (10).

Table (10) Fittest of the results

| Type of page | Percent of fittest |
|---|---|
| HTML document | 87% |
| ASP document | 80% |
| PDF document | 50% |
| Image document | 12% |

Now if we compare the effectiveness of the system with the (Speed and accuracy), and by using a computer with Pentium processer 4 (1.8 GHz Dual core of speed) and using Windows XP and Server 2003, With 60 users, we found out that the accuracy through using mobile agent is (12%) higher than without using it, and the speed is (11%) higher than without using mobile agent.

And by monitoring the network functionality by adapting the agents ( Crawler, Indexer, updater and mobile agent) compare it with the (speed, accuracy, maintenance compliance, reusability and updatability). And take the same parameters and measured without using the multi agents, then we found out the resulted as shown in table (11) below:

Table (11) Multi-agent system with non-agent system comparison

| System type | Speed | Accuracy | Maintenance | Reusability | Updatability |
|---|---|---|---|---|---|
| Multi-agent system | 85% | 80% | 95% | 85% | 93% |
| Non-agent system | 67% | 60% | 60% | 40% | 57% |

## 5- Conclusions

As we used MASSE system and examined with the experience from using it through the users we reach the following:

1- Through the process of using the MASSE program by testing it in many kinds of characteristics, and by recall most websites. We have positive development for the timing processed of the server applications when it use the user and builder query. And by testing the result we can be sure that the search has reached it is target.

2- The webpage indexing process takes long time because of the full text indexing process. Considering the search process is scan for every part of the webpage document and not only the keyword tags.

3- The crawler using single thread program and that is why it takes long time for processing a site. However the crawler speed depends on the registry priorities for the site that has been looking for.

4- All the information needed is up to date due to the updater agent always providing the latest updated to the database.

5- The SQL server is reducing the timing process during the operation of the agents' activities considering the size of the database structure. And it makes it more effective of using the data information, as it have the correct data for later use.

6- By making agent created for every component in the search engine system to be used and maintain in high active performance. Therefore the update and maintain these component will be accurate and effective.

7-these components is not limited of use in the search engine system, but for other use that could be helpful to adapt, like using it in the electronics books library.

The SEMAS has restrictions and achievements should be reached in the future. These restrictions can be summarised by the following:

1- The program can be used just for English only as it has English language for the interface, index and the websites.

2- The SEMAS works with PDF documents only. Therefore it would be useless if we try to use it for different format.
3- The database of the MASSE works just within the register website in the system. And cannot be updated and used from other search engines and web directories.
4- It has no security and authority check for the website register in the system to be able confirming the system is true and real.

## References

[1] Saba Abdul Kaliq Abdullah Al-khadady, “Internet and Arabic Search Engines”, M.Sc. thesis, Al- Nahreen University, 2002.
[2] Wei-Cheng Lai, Edward Chang and Kwang-Ting (Tim) Cheng, “An Anatomy of Large-scale Image Search Engine”, Morpho Software Inc., 3944 state street, Suite #340, Santa Barbara, CA 93105, (2005).
[3] Serenko, A., Ruhi, U. and Cocosila, M. (2007). "Unplanned effects of intelligent agents on Internet use: Social informatics approach." Artificial Intelligence & Society 21(1-2):141-166.
[4] Robert j. k., “Internet Search engine”, University of New York, December 6, 2004, “http://library.albany.edu/internet/engines.html”.
[5] Booth, D. ,Hoas, H., Mccabe, F., Newcomer, E., Champion, M., Ferris, C., et al. . Web Services Architecture. 2010, from "http://www.w3.org/TR/ws_arch/"
[6] A. Rogr, E. david, and J.Schiff, "The Effects of Proxy Bidding and Minimum Bid Increments within eBay Auctions, ACM Transaction on the Web, 2007.
[7] Nathan Schurr and Janusz Marecki, "The Future of Disaster Response: Humans Working with Multi-agent Teams Using DEFACTO", 2005.
[8] Ron sun and Isaac Naveh, "Simulating Organizational Decision-Making Using a Cognitively Realistic Agent Model", Journal of Artificial Societies and Social Simulation, 2009.
[9] Michael Wooldrige, "An Introduction to Multi-Agent Systems", John Wiley & Sons Ltd, 2002, ISBN 0-471-49691-X.
[10] Tamas Mahr , Jordan Srour , Mathijs M. de Weerdt and Rob Zuidwijk (2010). Can agents measure up? A comparative study of an agent-based and on-line optimization approach for a drayage problem with uncertainty. Transportation Research: part C 18(1): 99-119.

# مـحرك بـحث معتمد على نـظام مـتعدد الوكلاء

*م. نورا احمد مولى الساعدي

## المستخلص

النظام المتعدد الوكلاء، هو نظام متكون من عدة وكلاء متفاعلة فيما بينها. انه يستخدم لحل مشاكل معقدة ومن الصعب لنظام أحادي الوكيل حلها. هذه المشاكل عادةً ما تكون قابلة للتجزئة الى مشاكل اصغر كل واحدة منها توكل لوكيل او أكثر لحلها.

هذا البحث يقدم نظرة جديدة مقترحه لمحرك بحث يعتمد على نظام متعدد الوكلاء. ان المكونات الفرعية الوظيفية الرئيسية للمحرِّكِ مثل (الزاحف) و (المفهرس) و (المحدث) و (مكون الاستفسار) تم تصميمها كوكلاء تتعاون لأداء الوظيفة الرئيسية. ان محرك البحث المقترحة والتي سميت (محرك البحث متعدد الوكلاء) تم تصميمها لتزويدنا بنص كامل، قاعدة بيانات محدثة، مؤلف استفسارات و محرك بحث على الويب. تم اختبار النظام على شبكة الأنترانيت و من قبل عينة من الناس و قاعدة البيانات الخاصة بالنظام تمثل عينة من 1000 موقع تستعمل على مواضيع مختلفة. و اظهرت نتائج مبنمة دقة عالية اعتماداً على التاثيرات المرئية في الصفحة، وقاعدة البيانات المحدثة دائماً وسهولة توليف الإستفساراتِ التي تبنى بالاعتماد على مؤلف الأستفسارات. علاوة على ذلك العديد مِنْ ميزّاتِ البرمجيات المُوَكلة قد تحققت مثل إعادة استخدام البرامج، حل المشاكل الموزع، والذكاء الاصطناعي الموزع، والمعالجة المتَوازية.

* كلية الرافدين الجامعة