Towards a User-Centric Web Portals Management

Tarek Helmy

College of Computer Science and Engineering,
King Fahd University of Petroleum and Minerals,
Dhahran 31261, Mail Box 413, Saudi Arabia,
Tel: 966-3-860-1967 & Fax: 966-3-860-2174,
E-mail: helmy@ccse.kfupm.edu.sa

Abstract

There is an extensive set of published information on the Internet. Human based approach to discover and utilize this information is not only time consuming, but also requires continuous user interaction. Web portals are the next evolution in Internet services as they provide a more robust one-point access to a variety of core information, and ideally offer a single sign-on point. So far, most of the activities performed on the Web portals can be characterized as solitary ones where users/agents logged in on Web portals can only browse around pages experiencing everything on their own. In this paper we demonstrate the use of agent technology to enable user-centric discovery and utilization of Web portals. We have developed collaborative agents-based architecture aimed at supporting, as a particular kind of collaboration, for user-centric searching and managing of the Web portals. As the system is agent driven, each agent conforms to a communication protocol that allows it to send/receive messages to/from another agent. This paper mainly focuses on presenting the predictive power of capturing the user's preferences, how to use the implicated preferences to expand of user's requests by user-specific demands and wishes, how to filter the URLs not matching a certain profile and how to enhance the communication load between the agents to look for relevant information.

Keywords: Multi-Agent, Web Portals, Customization

I. Introduction

The WWW has had an enormous impact on business and society. It has succeeded largely because of its open architecture and ease-of-use. Although it was originally designed for distributed and interactive information sharing, it has evolved into a powerful business platform in which e-business is driving fundamental change in consumer buying patterns. However, because of the ubiquitous nature of the information on the Internet, customers have an enormous variety of services available to them, effectively overwhelming them with choices. Software agents are the subject of research in many inter-related fields. They are long lived, persistent computations that can perceive, reason, act, and communicate. They have the ability to make decisions independently without human intervention and without influence from other agents. A new agent-based paradigm is in high demand in which software agents can play an important role in automating many activities like modeling of the users, the discovery of interested information, and recommending the selected information. Toward this goal, we are proposing an agent-based framework as an expansion of my experience with agent based Web applications [16, 17]. The proposed framework essentially turns individual Web Portals into communicating and collaborating peers. The proposed framework aims

to give meaningful responses to meaningful requests and to deliver appropriate information to people who need it, when they need it, in a manner that meets their interests. The real power of the proposed architecture will increase exponentially, as more machine-readable Web content and automated services become available [13]. In this paper, we mainly focus on presenting collaborative agents framework to help alleviating the problems of searching/browsing the Web portals to find interest information. We present a new model of how to collect, measure, and evaluate the predictive power of explicit, implicit and mixed interest indicators to capture the User's/Customer's Preferences (UP) and reduce communication loads among agents to look for relevant information to the user's request. In addition we present the methodologies of modeling the customers and routing the customer's requests into the relevant Web portals. Furthermore we introduce the request refining and filtering mechanism by the Customer Interface Agent (CIA). Finally we present the experimental results, related works then conclude by the conclusions and future work.

II. Collaborative Agents Architecture Overview

Software agents have some autonomy and the ability to sense and react to their environment, as well as socially communicate and cooperate with other software agents in order to accomplish their duties, which are delegated from the customer. In a collaborative agent model, a particular agent may not have any prior knowledge, while there may be a number of agents belonging to other customers who have the required knowledge [3], [8], [10], [19], [20]. Instead of each agent re-learning what other agents have already learned through experience, agents can simply ask for help in such cases. This gives each agent access to a potentially vast body of experience that already exists. Over time each agent builds up a trust relationship with each of its peers analogous to the way we consult different experts for help in particular domains and learn to trust or disregard the opinions of particular individuals. In the proposed model, intelligent agents called Portal Agents (PAs) have been used to wrap to the contents of the Web portals. The PA creates software agents called Service Mining Agent (SMA) for each URL in the Web portal. The PA is designed as a special server extension module that learns to function in social environments and where necessary collaborates, completes or negotiates with other PAs. The SMAs could manifest various levels of intelligent behavior from simply reactive to adaptive and learning behavior, where agents actually learn what customers like and dislike. The PA is responsible for starting the transaction process and running of the registered SMAs. Once started, the SMAs, which can be seen as the local representatives of the services/products, can access the local data of the service/product and create the Semantic Policies (SP) [Fig. 1]**.** At the transaction phase, the SMA uses the SP to decide whether or not the customer's request belongs to the SMA. In the system, the PA and SMA use a predefined set of ontology for parsing and interpreting the contents of the Web portal. The system's agent [9], [16], [17] consists of a communication layer and an application layer. The Agent Communication Layer (ACL); comprises the common basic modules shared by all agents; and transmits data from source agents to destination agents through networks. In the system, we simply use TCP/IP, since most Internet applications implement their protocols on top of TCP/IP. The application unit comprises a set of plug-in modules, each of which is used for describing and realizing a specialized or native function of agents. As a Web application, the SMAs process and interpret the contents of the Web pages and exchange the SP freely with each other. We customized and standardized a set of ontology tags to be effective in the e-business application so that the SMA can interpret the contents of the service's/product's page [18]. The customer sends the request through the CIA. The CIA forwards

the request directly to the relevant PAs. The PA delegates the request to its SMAs. The SMA upon receiving a request, attempts to interpret it by itself or its down-chain SMAs. The SMAs return the results to the PA which in turn sends the results normally in the form of HTML/XML to be displayed by CIA's Browser. On the other hand, the PA (at the server side) and CIA (at the client side) will keep track the customer's actions and implicitly infer customer's preference (UP). The PA keeps a record of all previous transactions done by the CIA. Upon returning to the PA again, the customer is greeted by names and a list of recommended services based on the customer's history of purchases/browsing, this will also make it possible for online advertisements that match each customer's interest. By this way; the PA provides a service that would otherwise need to be handled by the sales representatives. Another example that uses these preferences effectively would be a "matching agent," which is an agent that finds the customers who have similar interests by comparing the customer's preferences.

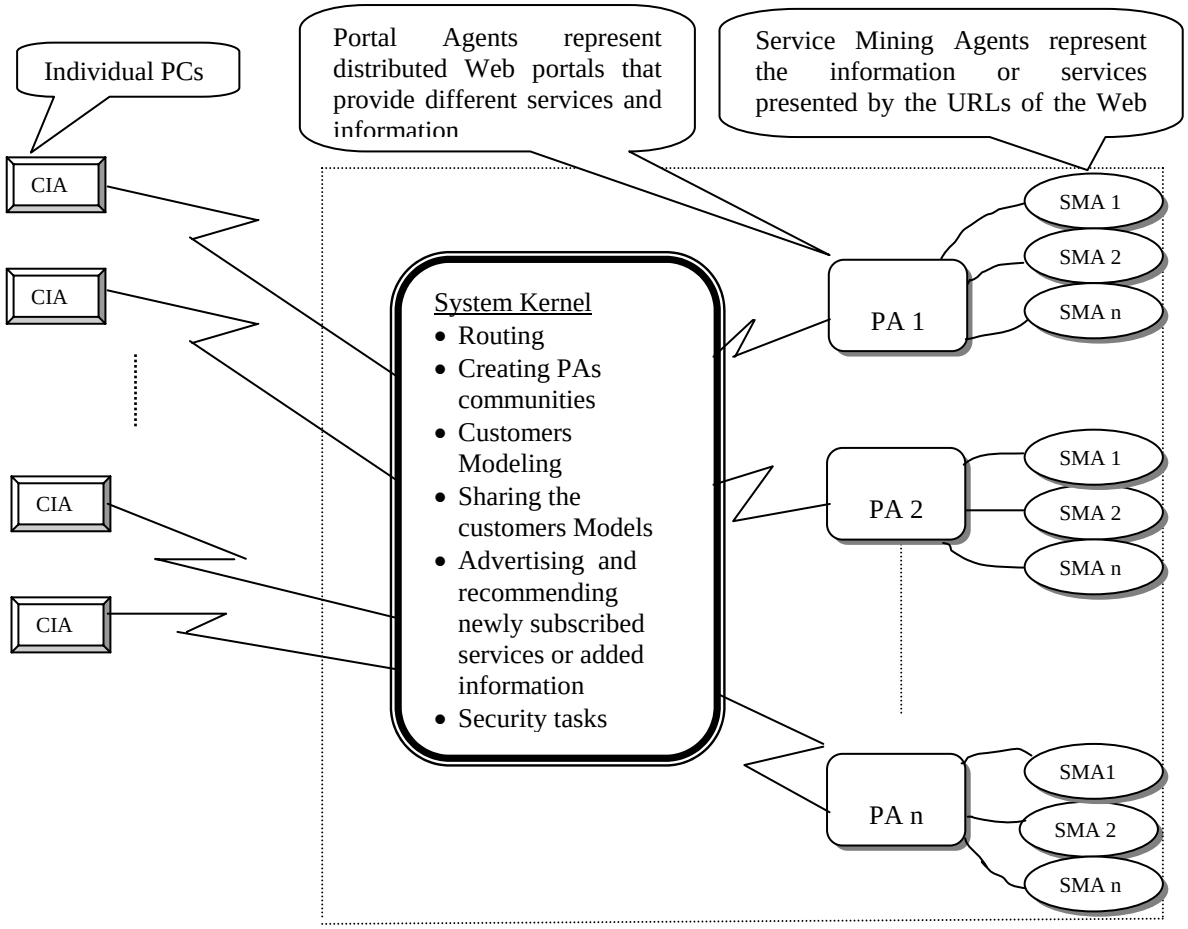

Figure 1: Collaborative Agents Architecture

III. The Customer Interface Agent

Personalization of Web portals search/browse is to carry out retrieval for each customer incorporating his/her interests [4], [8], [12]. Intelligent CIAs rely on UP for playing a fundamental role in actively finding required items on behalf of their customers. The CIA is designed to learn the UP either explicitly or implicitly from his/her browsing behavior [Fig. 2]. The CIA represents UP as

a set of categories; each category is a set of URLs and keywords have similarity over a predefined threshold value, which reflects a specific customer's category of interest [Fig. 3]. The CIA resides in the customer's machine and is usually a running process that operates in parallel with the customer, communicates with the SMAs via the PA to search for relevant services to the customer's request. The CIA looks over the shoulder of the customer and records every action into the UP files. The CIA analyzes the Natural Language (NL) request of the customer, and looks for relevant URLs to the given request in the UP files. If the CIA finds relevant URLs in the UP files, then it shows them to the customer and asks whether he/she is satisfied or wants to search the Web portals. If the CIA could not find in its UP files any relevant services to the given request then the CIA refines the given request and routes it to the relevant PAs, which in turn forward the refined request to their SMAs. The CIA receives the results returned by the PA; the results consist of a set of URLs and their similarity value to the refined request. The customer either explicitly marks the relevant URLs using CIA's feedback menu or the CIA implicitly catches customer's response through his actions like period of visiting, bookmark, saving, printing, copying/pasting, scrolling, following a link, or ordering it. The response is used to adapt the contents of the UP files. Followings are the detail description of CIA's functionalities.

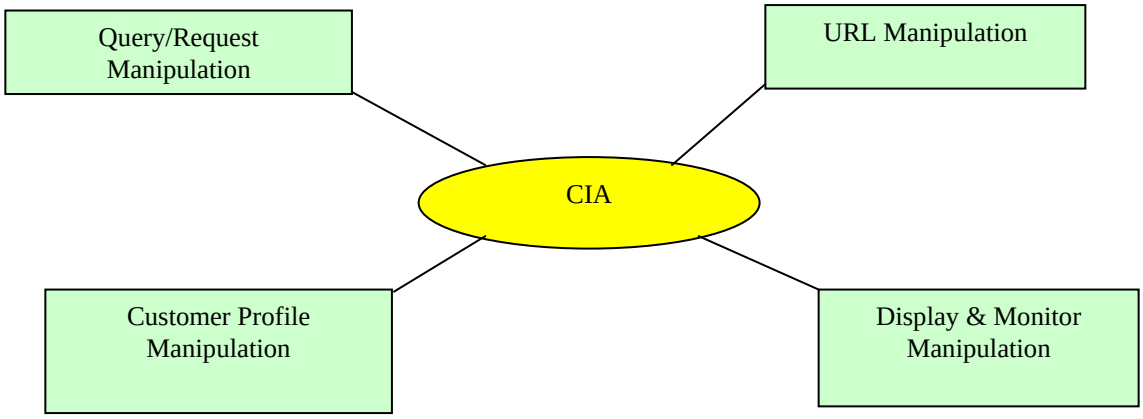

Figure 2: Functionalities of the CIA

III.1 Customer's Profile Manipulation

A. Searching for Relevant URLs in the UP Files

The CIA looks for every URL stored in the UP and compares between every keyword attached to that URL and keywords of the customer's request. For all matched keywords of the same URL, the CIA will add their weights and assign them in temporarily variable as a reference to that URL. At the end, the CIA will order and display the referenced URLs in decreasing order according to the total weights.

Input: Customer's request Q< key_1, key_2, .., key_n>
Output: Sorted array W [] [1, 2, 3, 4] = [Id, URL, title, total_weight]
*/*Id: URL's id number in the URL table, URL: URL address, Title: the title of the Web service, Total weight: weights of all keywords in this URL*/*
m ← count (URLs saved in the URL table)
/ for every URL*/*
for i ← 1 to m
    n ← count (keywords <k> saved in the URL table for URL_i)
    */*find all keywords in URL_i that exist in customer's request*/*
    for j ← 1 to n

```
                    if k_{i,j} belongs to Q then
                    /*sum the weights of all keywords in every URL*/
                            temp ← temp + W_{i,j}
                    end if
            end for
            If W [i][4] > 0 then
              /*if total weight > 0: means if there are keywords found in this URL that are also in customer's
                  request, then add URL's information to the array*/
                    W [i][1] ← Id_i
                    W [i][2] ← URL_j
                    W [i][3] ← Title_i
            end if
    end for
    Sort array W in descending order according to total_weight
    Return (W)
```

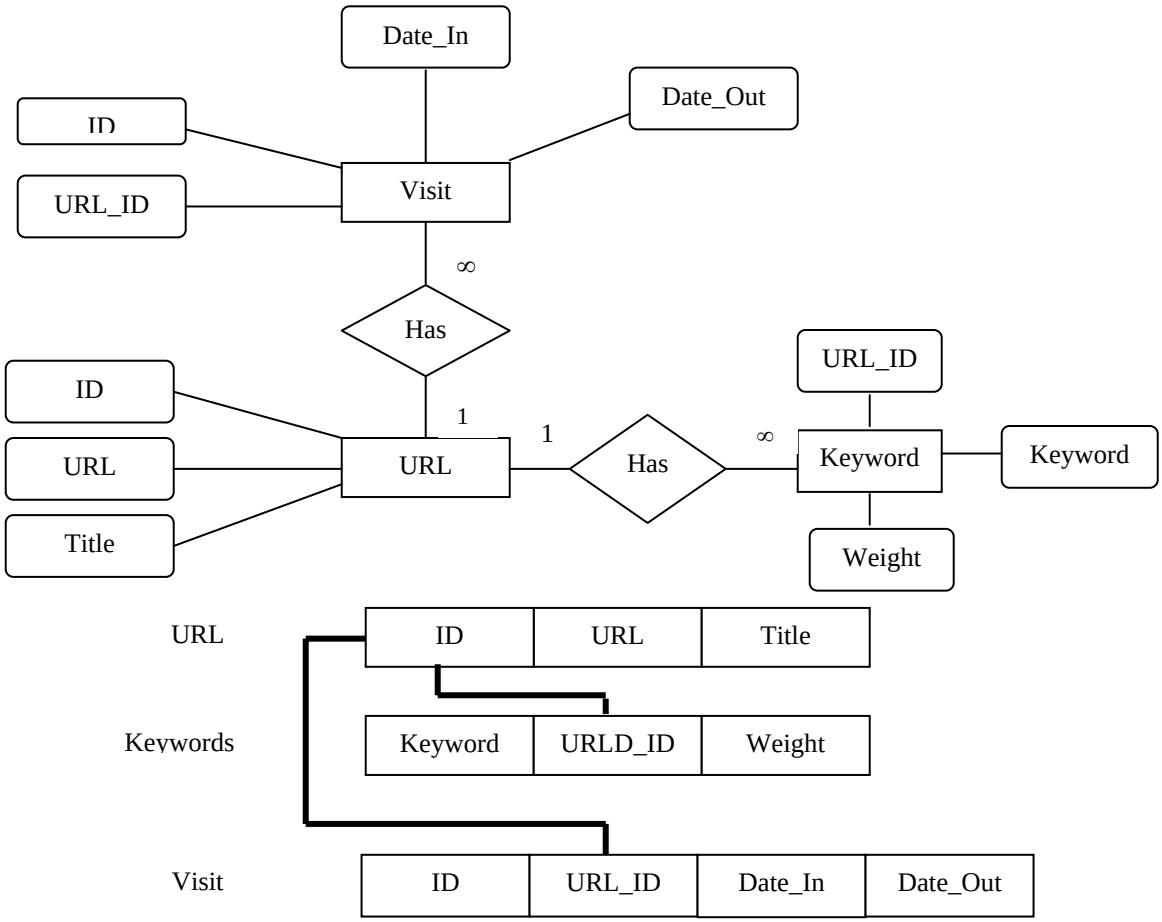

Figure 3: The Entity relationship and relational schema diagram of the UP

B. Adapting the URLs Table

When the customer shows an interest about particular URLs that are displayed to him/her as a response to his/her request, the CIA modifies the contents of the URL table to reflect the new preferences.

Input: Input (URL, title, Q <key_1, key_2, …, key_n>, interest_value)

*/*URL: URL address, title: the title of the Web service, Q: customer's request, Interest_value: the value that will be assigned to the keywords*/*
i ← 0
*/*search for the given URL address in the URL table until you find it or you finish searching all the records*/*
While (i < count (URLs saved in the URL table) and found = 0)
    i ← i + 1
    If $URLi$ = Input.URL then
        found ← 1
    end if
end while
*/*if it finds the given URL address in the URL table*/*
if found = 1 then n ← count (keywords <key> in Q)
/ for every keyword in the customer's request Q*/*
    for j ← 1 to n
        */*if this keyword is saved in this URL*/*
        if key_j belongs to (keywords <k> saved in the URL table for URL_i) then
*/*add to its weight the 'interest_value': such that the final value will not exceed 1*/*
            $w_{i,j} \leftarrow w_{i,j}$ + interest_value
            if $w_{i,j} > 1$ then
                $w_{i,j} \leftarrow 1$
            end if
        */*if this keyword is not saved in this URL*/*
        else
*/*add this keyword to the list of keywords that belong to this URL and makes its weight equal to the 'interest_value'*/*
            $k_i \leftarrow k_i + key_j$
            $w_{i,j}$ ← interest_value
        end if
    end for
*/*if it didn't find the given URL address in the URL table*/*
else */*add a new entry: URL reference #, URL address, the title of this URL*/*
    Id ← i + 1
    $URL_{i+1} \leftarrow URL_{i+1}$ + Input.URL
    $Title_{i+1}$ ← Input.title
    n ← count (keywords <key> in Q)
    */*for every keywords in the customer's request Q*/*
    for j ← 1 to n
*/*add this keyword to the list of keywords of this URL and make its weight equal to the 'interest_value'*/*
        $K_{i+1,j} \leftarrow K_{i+1,j} + Key_{i+1,j}$
        $w_{i+1,j}$ ← interest_value
    end for
end if

C. Setting and Modifying the UP

The CIA allows the customer to set the initial preferences when running it for first time. Also, s/he can modify these preferences at any time later. Some of these preferences are: the number of results to be displayed to the customer, and the default PAs that will receive the customer's request. The customer can modify these portals and enable/disable them to

represent his preference regarding which portal s/he would like to send the request to. Moreover, the CIA allows the customer to add other portals.

```
/*the default information set in the preference unless the customer modifies it*/
/*i.e. number of results to be displayed per page & the default Web portals*//
        [UP].Customer_results/page ← 5
        Web_ portal [1] ← 'PA1'
        Web_ portal [2] ← 'PA2'
        Web_ portal [3] ← 'PA3'
/*if the customer wants to modify the number of results to be displayed per page*/
if Customer.modify (Customer_choioce_results/page) then
/*allow him/her to modify only if the number is greater than or equal to 5*/
        if Customer_NewChoice >= 5
                Customer_choioce_results/page ← Customer_NewChoice
        end if
end if
/*if the customer wants to add more Web portals to the default ones*/
If Customer.add_Webportal (Customer_NewPortal) then
    /*check that s/he is not trying to add Web_portal that has been added*/
        If Customer_NewPortal does not belong to the default Web Portals then
                Web Portal ← Web_Portal + Customer_NewPortal
        end if
end if
/*If the customer wants to remove a Web Portal from the default list*/
If Customer.delete_Webportal (Customer_Oldportal) then
/*check that's/he is not trying to remove one of these portals*/
        if Customer_OldPortal does not belong to ('PA1';'PA2';'PA3') then
                WebPortal← Webportal-Customer_Oldportal
        end if
```

III.2 ***Request Manipulation by the CIA***

A. ***Filtering the Request***

When the customer enters the request in NL, the CIA removes what is called noisy words from it before it starts searching the UP. These words like the prefix, suffix, and other words. Examples are (re, -or, -er, at, the, on, in, an, his, our, we, are, etc.). These noisy words will be predefined in a database file. The CIA looks up in the table specified for these words and removes from the request whatever words match any one of them.

```
Input: Request in NL: Q <key1, key2, …, keyn>
Output: Request after filtration from noisy words
m← count (keywords in Q)
/*for all keywords in customer request*/
for i ← 1 to m
        if Key_i belongs to (noisy words N)
        /*if there are noisy words in customer's request, remove them*/
                Q ← Q – Key_i
        end if
end for
Return (Q)
```

B. ***Refining the Request***

The request-refine process can serve as a context to disambiguate the words in the customer's request. This means that the request used internally by CIA is different from the one submitted by the customer to be more representative of the customer's intent. The CIA adds

to the request the keywords attached to the URLs that have the most relevancies to the customer's request to represent the customer's search intention and delegates it to the PA.

Input: Request in NL but already filtered: Q <key_1, key_2, …, key_n>, Sorted array W (Id, URL, title, total weight)
Output: Request refined by adding some other keywords from the URL table
m ← count (Id in W)
*/*for all URLs in W: array of results retrieved from the URL table*/*
for i ← 1 to m
    if W [i][4] > 0.5 then
    */*If the weight of this URL > 0.5, then*/*
        n ← count (keywords <k> saved in the URL table for Id_i)
        for j ← 1 to n
    */*for every keyword in this URL but not in customer request Q, add it to the Q*/*
            if $k_{i,j}$ does not belong to Q then
                Q ← Q + $K_{i,j}$
            end if
        end for
    end if
end for
Return (Q)

III.3 ***URLs Manipulation by the CIA***

A. ***Delegating the Request to the Web Portal Agent***

The CIA takes the manipulated request and sends it to the default Web Portal agent (PA).

Input: Request Q after refining process
Output: Index (Number of Web Portals received this request)
*/*if the customer choose to send his/her request to the default Web portal*/*
If RecvPortal = default then
    m ← count (Web portals exist in default Web portals list)
*/*for every Web portal in the list*/*
    for i ← 1 to m
*/*If the Web portal set to receive queries*/*
        if Web_Portal_i.enable = true then
    */*send the request Q to it*/*
            $Web\ Portal_i$. Request ← Q
        end if
    end for
*/*if the customer chooses to send his/her request to the Web Portal of his/her choice*/*
else if RecvPortal = Customer_choice_portal then
        */*send the request Q to the portal chosen by the customer*/*
    Customer_choice_portal. Request ← Q
end if

B. ***Receiving Request Results***

The CIA presents the received results to the customer through the system's browser.

Input: total_portal (Number of Web Portals received the request)
Output: Array R contains the retrieved results (first 5 results from each Web Portal (PA))
    R (Web Portal, URL $address_1$, URL $address_2$, …, URL $address_5$)
/ if the customer chooses to send his/her request to the default Web portals*/*
If RecvPortal = default then
    for Index ← 1 to tot_portals
      */*save the first five results retrieved from of every Web portal [PA] in the array*/*
            R [Index][1] ← URL_1

```
                    R [Index][2] ← URL2
                    R [Index][3] ← URL3
                    R [Index][4] ← URL4
                    R [Index][5] ← URL5
                end if
            end for
            /* if the customer chooses to send his request to Web Portal of his/her choice [new PA]*/
else if RecvPortal = Customer_choice_Portal then
            /*save the first five results retrieved from of this Web portal in the array*/
                    R [1][1] ← URL1
                    R [1][2] ← URL2
                    R [1][3] ← URL3
                    R [1][4] ← URL4
                    R [1][5] ← URL5
end if
Return (R)
```

III.4 ***Filtering and Monitoring by the CIA***

The CIA uses UP contents to improve the task of finding relevant services, and for filtering of the retrieved URLs. The CIA takes a set of keywords from the UP; whose similarity to the submitted request is over a predefined threshold value; and combines them to form a new contextual request [16], [17]. The CIA uses the contextual request vector to represent the current customer's interest and to filter the retrieved URLs. For each of the retrieved URLs a keyword vector is calculated and compared with the contextual request vector [7]. Comparing means, which URLs in the retrieved set are relevant (have a similarity value with the contextual request vector over a predefined threshold value) and which are non-relevant with the current customer's category of interest.

A. ***Displaying the Results***

When the customer clicks on specific URL, the CIA opens its browser and displays the contents of the URL.

```
Input: Array of A (URL Addresses, URL titles)
Output: Displayed URL contents in the browser
/*display the results, where the number of results per page is retrieved from the UP*/
CIA [results/page] ← [UP].Customer_choioce_results/page
CIA ← (URL Addresses, URL titles)
/*if the customer clicks on specific URL, display it in the browser*/
If Customer.click = URLi then
        Browser ← URL i
end if
```

B. ***Monitoring User Responses***

For the CIA to be truly useful, customer's interests must be inferred implicitly from actions and not only obtained exclusively from explicit content ratings provided by the customer through the menu bar provided by the system's browser, because having to stop to enter explicit ratings may not be of interest for many customers [6], [11]. The CIA monitors the customer's actions implicitly like copying, scrolling, saving, book marking, printing the content of the URL which means that s/he is interested in it but not as much as if s/he rated it explicitly.

```
Input: URL displayed in the browser
```

```
Output: the customer's interest value about the URL
/*display document in the browser*/
Browser ← URL document
/*save the customer's explicit feedback about this URL into variable 'interest_value'*/
if customer_response = Interest then
        interest_value ← 1.0
else if customer_response = Neutral then
        interest_value ← 0.75
else if customer_response = mildly interest then
        interest_value ← 0.50
else if customer_response = Not very interest then
        interest_value ← 0.25
else if customer_response = Useless then
        interest_value ← 0.0
/*monitor & save the customer's implicit feedback into variable 'interest_value'*/
else if customer_response = Save then
        interest_value ← 0.6
else if customer_response = Print then
        interest_value ← 0.5
else if customer_response = Bookmark then
        interest_value ← 0.4
else if customer_response = Copy and Past then
        interest_value ← 0.3
else if customer_response =remain visiting the page while scrolling then
        interest_value ← 0.2
end if
Return (interest_value)
```

IV. System Implementation and Agent Interaction

The proposed agent system comprises the hierarchical structured agent communities based on a PA model. The PA is the representative of the community and allows all SMAs in the community to be treated as one normal agent outside the community. A PA has its role limited in a community, and itself may be managed by another high-level PA. A PA manages the SMA members in a community and can multicast a message to them. Any SMA in a community can ask the PA to do multicasting for its message. All agents form a logical world which is completely separated from the physical world consisting of agent host machines. That means agents are not network-aware, but are organized and located by their places in the logical world. This model is realized with the agent middle-ware. The agent middle-ware is primarily designed to act as a bridge between two distributed physical networks, and to create an agent-friendly communication infrastructure on which agents can be organized in a hierarchical fashion more easily and freely. Followings are the scenario of agent interactions:

- The customer submits the request to the CIA.
- The CIA looks for relevant URLs to the given request in the UP files.
- If the CIA finds relevant URLs in the UP files, then it shows them to the customer and asks whether he/she is satisfied or wants to search the Web portals.
- If the CIA could not find in its UP files any relevant services to the given request or the user wants to send the query direct to the Web portals.
- The CIA starts by initiating a communication request with the predefined PAs.
- The PA identifies the CIA and replies with either accept/reject response.
- In case of accepting, the CIA sends the original/refined customer's request to the PA.

- The PA delegates the request to its SMAs with an "*Is-This-Yours*?" performative.
- The SMA upon receiving a request, attempts to interpret it by itself.
- If the interpretation is successful done, the SMA will report to the PA using the "*It-Is-Mine*" performative with a certainty value.
- The SMA returns the result to the PA.
- The PA retunes the result to the CIA to be displayed by CIA's Browser.
- If the SMA cannot interpret the request as its own, before reporting "*Not-Mine*", it must check with its down-chain SMAs.

A. ***Request Routing by the CIA***

The CIA creates its own PA's attributes table and modifies it over time automatically to reflect the popularity of each PA [17]. These attributes reflect the profile of each PA to the CIA and will be used in the routing mechanism. The CIA learns the profile of the PA as following:

- The CIA learns the profile and popularity of each PA for upcoming relevant request based on customer's historical responses and creates its own communities of PAs.
- The PA sends a message (know me) to the known CIA enclosed with its attributes and other information as an advertisement.
- The CIA gets these attributes while making the first transaction with the PA.

V. Experimental Evaluation

We have conducted several experiments to make a consistent evaluation of the system's performance. As this paper is mainly focusing on the CIA, we conducted some experiments for evaluating how much the CIAs will enhance the customer's satisfaction and reduce the communication loads to look for relevant information to a user's request.

A. ***Experimental Setup***

In the experiments, the CIAs have been installed on 10 PCs running Windows XP for about 3 weeks. We used three Web portals (Yahoo [21] Amazon [1], AOL [2]) for the experiments. The Web pages used from each portal are limited to 3 categories; Books & magazines, Music, and Computers & electronics. The PA is assigned to each Web portal and the SMAs are created and activated in the experiments. We set the maximum number of SMAs per each Web portal to be 300 agents and thus 100 SMAs are assigned to each category. All agents are realized by describing their functions into plug-in modules of the agent's application unit. Thirty male customers participated in this experiment as follows. Twenty undergraduate students with different background (management, computer science & engineering) and ages (freshmen, sophomores, juniors, seniors), 5 faculties and 5 graduate students of different background (management, computer science & engineering) have been asked to use the system to search for some books, music CDs, digital cameras, and computer accessory from the Web portals. Customers were not told the purpose of the experiments.

B. ***Experiments Scenario***

Experiment 1: The focus is mainly for evaluating how much the request refining and filtering of the retrieved URLs by the collaborative CIAs will enhance the customer's satisfaction. *In the experiment*, ***first***, each customer submitted 30 different requests to his CIA, which in turn routed the requests to the PAs directly and then we calculated the Precision (customer's satisfaction). ***Second***, each customer submitted the same 30 different requests to his CIA, which in turn

refined the requests; based on the customer's preferences, routed the requests to the PAs and then we calculated the Precision. ***Third***, we set the CIAs to be able to collaborate with other predefined CIAs. This means the UP files can be shared among many CIAs. After that, each participant submitted again the same 30 different requests to his CIA, and then we calculated the Precision. **Precision** is the ratio of the number of relevant URLs returned to the total number of irrelevant and relevant URLs returned. The results shown in [Fig. 4] are the average Precisions of the individual 30 Precisions of each request. The results prove that the way of complementing Web portal search with CIAs support makes sense and allows performing Web portals search in a more qualitative way consistent with the customer's need. The results also prove the hypothesis that after certain number of interactions, the Precisions of the refined requests are being improved and this confirms the fact that CIAs are able to refine well the customer's requests after learning and adapting his profile.

The following comments were received from the experiment participants:

- The system helped me to easily find the interested information form the Web portals.
- It was very pleasant to receive information about various items from other customers.
- I want more information from other Web portals.
- The system takes a little bit more time to retrieve the results but with lower noise URLs.

Some comments show that the system provides satisfied information to the customers. On the other hand, there are some negative comments; we will try to inspect them by mining the details of the customer's profile and scaling up the system to cover more Web portals. For the moment we have not run yet experiments for a number of PAs bigger than 3. However, we suppose that after a number of PAs reach a certain level, increasing of the number of community members will cause only moderate increase of performance characteristics.

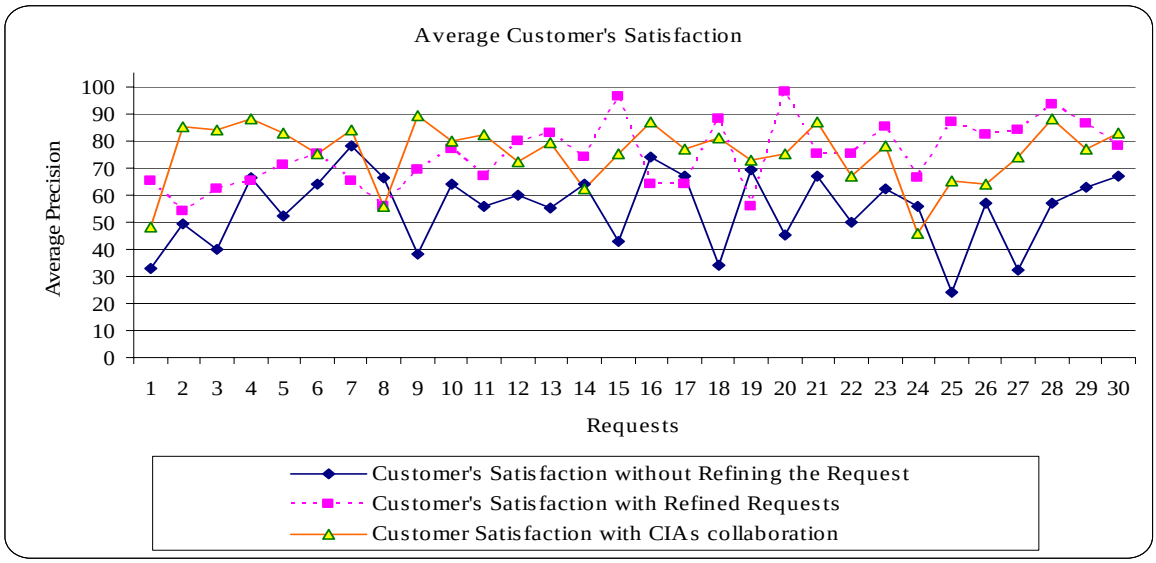

Figure 4: Customer's satisfaction

Experiment 2: The focus is mainly to show the effectiveness of using user's browsing history, refined user's requests and collaboration among the CIAs on the communication load between CIAs and PAs to look for relevant information to a user's request. We compared the traffic of the communicating messages in 4 cases. The results shown in [Fig. 5] prove that the number of exchanged messages among the agents when searching for relevant information to a query without using the user's browsing history (UP) is much more for most of the queries input, while that of using user's browsing history, refining user's requests and collaboration among the CIAs

were being reduced. It is also shown that the number of exchanged messages among the agents while allowing the CIAs to collaborate is a little bit larger than in case of refining the request based on the UP of the user himself.

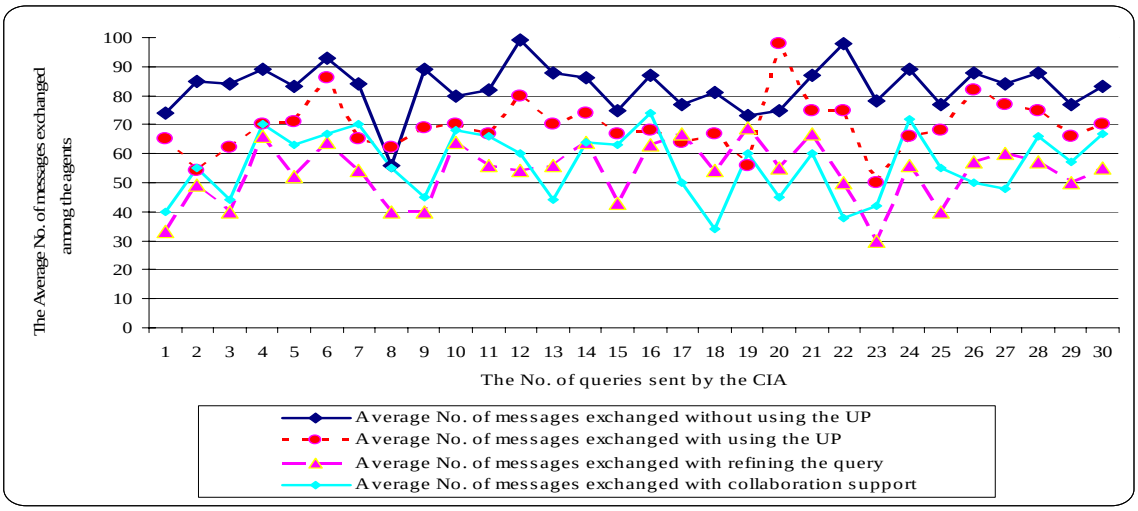

Figure 5: The average No. of messages exchanged among the agents

VI. Related Works

In this section we give a small survey of the work related to this paper. The SurfAgent [15] is used to assist the user while browsing the Web. SurfAgent uses TF-IDF vectors [14] for representing its UP in several topics of interest. Learned profiles are used for generation of queries to search engines in order to automatically find new pages which are interesting to the user. Authors in [19] present a market-based recommendation system. It is a multi-agent system where agent acts on behalf of its user and sells the slidebar space where recommendations can be displayed. Other agents participate in this auction in order to show their links on this slidebar. The agent-initiator of the auction chooses the most profitable offers and displays them to the user. After the user accepts some results, his/her personal agent rewards the providers of the accepted links while other agents receive no reward. Thus, agents try to make better suggestions in order to increase their profit. A multi-agent referral system described by [18], in which each user has his/her own personal agent, the agent interacts in order to provide the user with answers to his/her question. From agent's point of view the other agents are classified as neighbors and acquaintances and their status in this classification determines the way of contacting them. The system uses ontologies to facilitate knowledge sharing among agents and the ontologies have to be predetermined and shared among all the agents. Differently from the proposed system, their agents do not answer all questions but only those are related to their own user interests. The paper is focused more on knowledge search rather than on Web portals search. Finally, their system is mail-based while the proposed system is a Web portals based. Authors in [8] presented a recommendation system incorporating collaborative filtering and learning UP techniques. Thus, this system combines collaborative approach with analyzing Web page content. The knowledge about users is represented in UPs and used within the collaborative filtering algorithm to reduce the time of the recommendation generation. The collaborative multi-agent Web mining system described by [5] implements the post-retrieval analysis and implies cross-user collaboration in Web search. In order to provide the user with recommendations, there is a special agent that performs profile matching to find the information potentially interesting to the user before specifying the area of the interest and privacy or publicity of the search. One of the sufficient differences between this system and the proposed system is that the user should analyze excessive

output because he/she has to browse a number of similar already finished search sessions. As it appears, there are research studies exploiting ideas approximately similar to some of presented in this paper but do not use system structure and techniques similar to one used in this paper.

VII. Conclusions and Future Work

In this paper, an agent-based framework for user-centric searching of Web portals has been introduced. In the proposed approach agents play an important role in automating many activities like modeling of the users, the discovery of interested information, and recommending the selected information. This approach essentially turns individual Web Portals into communicating and collaborating peers. The proposed framework aims to give meaningful responses to meaningful requests and to deliver appropriate information to people who need it, when they need it, in a manner that meets their interests. The paper presented a new model of how to collect and evaluate the predictive power of explicit, implicit and mixed interest indicators to capture the customer's preferences and routing the customer's requests into the relevant PAs. The main advantages of the proposed system that confirmed by the encouraging results obtained in the experimental phase are no explicit feedback is required, the query refining feature and the relevance ranking computation. The results showed also the efficiency to reduce communication loads between agents when searching for relevant information to a query. Next step is to tune the agent parameters in order to get a more efficient performance and to test the agent with more complex user's profiles. As a future work, we want to scale up the scope of the experimentation to make a precise evaluation and solve the negative comments of the experimenters. Due to the emergence of the semantic Web, the SMA should be able to process and semantically interpret the contents of the Web pages and exchange the SP freely with each other.

Acknowledgments

I would like to thank King Fahd University of Petroleum and Minerals for providing the utilized computing facilities. The author is grateful to anonymous reviewers for their insightful comments and feedback.

References

1. Amazon, http://www.amazon.com/
2. AOL, http://www.aol.coml
3. Billsus D., Pazzani M.,"A Personal News Agent that Talks, Learns and Explains," Proceedings of the 3rd International Conference on Autonomous Agents, 1999, Seattle, WA.
4. Bonett Monica, "Personalization of Web services: Opportunities and Challenges," Ariadna 2001, ISSN: 1361-3200.
5. Chau M., Zeng D., Chen H., Huang M., Hendriawan D., "Design and Evaluation of a Multi-agent Collaborative Web Mining System," In Decision Support Systems 35 (2003) 167-183.
6. Claypool, M., Le, P., Waseda, and M., Brown, D., "Implicit interest indicators," Proceedings of the 6th International Conference on Intelligent User Interfaces (IUI 2001), USA, pp. 33-40.
7. Daniel Lemire, "Scale and Translation Invariant Collaborative Filtering Systems," Information Retrieval, 7, pp.1-22, 2004.

8. Degemmis M., Lops P., Semeraro G., Costabile M.F., Lichelli O., Guida S.P.,"A Hybrid Collaborative Recommender System Based on User Profiles," Proceedings of the Sixth International Conference on Enterprise Information Systems, 2004, Vol. 4. pp. 162-169.
9. Guoqiang Zhong, Satoshi, Amamiya, Kenichi Takahashi, Tsunenori Mine and Makoto Amamiya, "The Design and Implementation of KODAMA System," IEICE Transactions on Information and Systems, E85-D (4):637-646, April 2002.
10. Good N., Schafer B., Riedl J., "Combining Collaborative Filtering With Personal Agents for Better Recommendations," Proceedings of the AAAI-'99 conf., pp 439-446.
11. Jung, K., "Modeling web user interest with implicit indicators," Master Thesis, Florida Institute of Technology, 2001.
12. Liu, F., Yu, C., and Meng, W., "Personalized Web search by mapping user queries to categories," Proceedings of the 11th International Conference on Information and Knowledge Management (CIKM 2002), USA, pp. 558-565.
13. Markup for Web Services, http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/
14. Salon G., McGill M.,"Introduction to Modern Informational Retrieval," McGraw-Hill (1983).
15. Somlo G., Howe A., "Using Web Helper Agent Profiles in Query Generation," ACM Proceedings of the 2nd Inter. Joint Conference on Autonomous Agents and Multi-agent Systems 2003, pp. 812-818.
16. Tarek Helmy, Satoshi Amamiya, Tsunenori Mine, Makoto Amamiya, "A New Approach of the Collaborative User Interface Agents," Proc. of IEEE/WIC/ACM Int. Conference on Intelligent Agent Technology (IAT'2003), pp. 147-153, Oct. 13-17, Canada.
17. Tarek H. El-Basuny, "A Ubiquitous Approach for Next Generation Information Retrieval System," Proc. of the IEEE International Conference on Information and Computer Science 2004, KFUPM, Saudi Arabia, pp. 501-513.
18. Uit Beijerse, R.P., "Questions in Knowledge Management: defining and conceptualizing a phenomenon," Journal of Knowledge Management, 2000, 3(2):94–109.
19. Wei Y., Moreau L., Jennings N., "Recommender Systems: A Market-Based Design," ACM Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-agent Systems 2003, pp. 600-607.
20. Wooldridge M., Kinny D., "The Gaia Methodology for Agent-Oriented Analysis and Design", Journal of Autonomous Agents and Multi-Agent Systems, pg. 285-312, 2000.
21. Yahoo, http://shopping.yahoo.com/

Tarek Helmy is an assistant professor in the department of Information and Computer Science, College of Computer Science and Engineering at King Fahd University of Petroleum and Minerals. He holds a PhD. From Kyushu University, Japan in 2002. His research interests include Operating Systems, Multi-Agent Systems, Artificial Intelligence, Personalized Web services, and Cooperative Intelligent Systems. He has published more than 30 papers in major international journals and conferences in the filed of intelligent agents and operating systems.