

COLLABORATIVE MULTI-AGENT-BASED E-COMMERCE FRAMEWORK

Tarek Helmy

College of Computer Science and Engineering,
King Fahd University of Petroleum and Minerals,
Dhahran 31261, Mail Box 413, Kingdom of Saudi Arabia,
helmy@ccse.kfupm.edu.sa

ABSTRACT

Software agents offer a promise to change electronic commerce trading by helping traders to purchase products based on their interests and preferences. E-commerce systems are increasingly recognizing the importance of giving additional value to customers by providing customized transactional experiences. We believe that increased support for collaboration is a logical next step in the evolution of the e-commerce systems. The main goal here is to create a collaborative multi-agent based e-commerce framework that allows autonomy, pro-activity, and personalization including an intelligent mall agent, a seller agent, and a buyer agent with their independent profiles. The basic idea is to get a fully qualified broker that can intelligently identify the buyer's needs based on standard parameters that are given to help alleviate the problems of finding interest items. The proposed framework aims at giving meaningful responses to meaningful requests and to delivering appropriate items to people who need it, when they need it, in a manner that meets their interests. To demonstrate the proposed framework a prototype is implemented and tested.

Keywords: Multi-Agent, E-commerce system.

1. INTRODUCTION

With special attention to the application of multi-agent systems in e-commerce, their development is growing rapidly accompanied by the growth of the Internet. There are several multi-agent systems to mediate e-commerce that have been developed, such as (Chavez and Maes, 1996), (Maes P. et al, 1999), (eBay), (MARI), (Fonseca et al., 2001), and (Elhadi and Abu-Draz, 2003a, 2005b). Software agents in e-commerce systems 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 (Bartolini et al., 2002), (Anthony and Jennings, 2003), which are delegated from the customer. These systems are considered to be dynamic networks of interrelated transaction processes that allow P2P communication through trading agents.

The targeted system is a distributed P2P system, where peers will be given the ability to securely, reliably communicate and share their information. The proposed framework allows peers to communicate directly without the use of centralized servers. This makes it more reliable and less dependent on outside sources of possible problems. There are three main components of the system, a Mall Agent (MA), a Buyer Agent (BA), and a Seller Agent (SA). The SA is a peer who has information that needs to be offered or advertised to the BAs. A BA is a peer that needs information about its customer's request. Finally, the MA provides a list of SAs that best match the customer's request by comparing the request with the product features contained in the common database of the MA. Current available e-commerce systems are based on centralized servers. Centralized servers are a weakness of such systems, since they introduce a single point of failure and a bottle neck. That is, if the server goes down for any reason, the whole system will go down. In our framework, even if the MA goes down, the BA could negotiate with the pre-known SAs. We believe that the proposed system will avoid the weaknesses available in some e-commerce systems by having a flexible P2P communication, which is both efficient and reliable.

In this paper, we present the implementation details of the system, the communication protocols, setting the Customer Preferences (CP), routing the customer's requests into the relevant SA and the request expansion by the BA. Finally we present the experimental results, and then conclude by the conclusions and future work.

2. RELATED WORKS

The system presented by (Chavez and Maes, 1996), helps users creating agents to negotiate the buying and selling of goods on their behalf, also allowing the specification of parameters to guide and constrain an agent's overall behavior. However, these agents do not live permanently but only during the completion of a certain transaction, thus not fully exploiting each actor's profile, as well as the proactiveness and semi-autonomy of their agents (e.g., towards new coming offers or requests). Karacapilidis and Moraitis (2001) presented a web-based electronic commerce system in which customers and merchants delegate the related tasks to their personal software agents. Messages passed between these agents can fully encapsulate the associated parties' points of view towards a market transaction.

Collins et al. (2002) presented MAGNET, which is a mobile agent-based system that enables buyers to compare products from different sellers. Negotiation between agents is based on multi-attributes. Although MAGNET agents are able to negotiate over multi-attributes, only one buyer agent visits many sellers sequentially. This has a high impact on the system performance when the number of seller agents increases. (Anthony and Jennings, 2003) proposed an agent framework to bid over multiple auctions. Although, they were able to address the problem of segregation, other issues were overlooked. (Wei et al., 2003) present a market-based recommendation system. It is a multi-agent system where agent acts on behalf of its user and sells the sidebar space where recommendations can be displayed. Other agents participate in this auction in order to show their links on this sidebar. 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. As it appears, there are research studies exploiting ideas approximately similar to some of presented in this paper but do not use system structure similar to one used in our paper which supports P2P model. Furthermore XML technology is used since it allows information and services to be meaningfully encoded.

3. THE FRAMEWORK HIGH LEVEL ARCHITECTURE

Developing the framework requires building three independent communicating agents. This requires designing the agents, the communicating protocols, and the GUI interfaces. All the agents built using generic and reusable architecture (Bellosta et al., 2004), (Bartolini et al., 2005). Each agent in the system has a local database that utilizes a profile. The BA's profile stores information about the preferences of its customer, the SA's profile stores the information about the items to be offered. When the BA wants to search for a product, it must send a query to the MA. Then the MA will reply by sending the matching list of SAs that it has. Once the purchaser is satisfied with a product, the BA can establish a negotiation process direct with the SA (Figure 1).

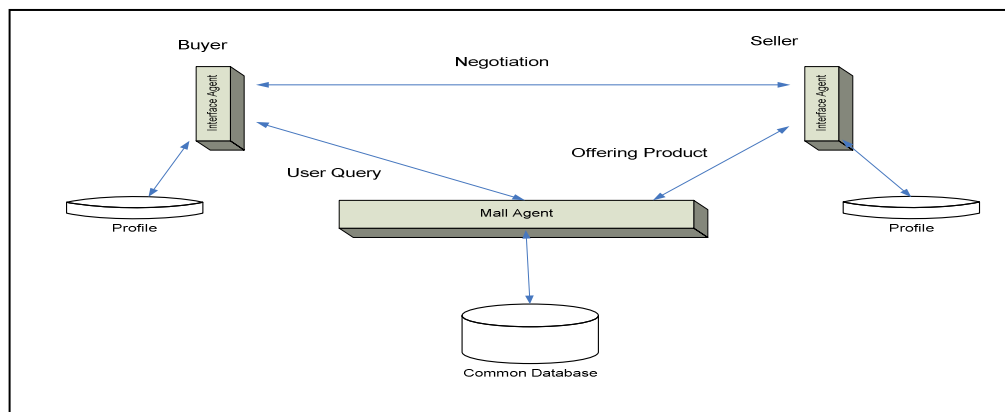


Figure 1: Overall system architecture

3.1 The Buyer/Seller Agent Architecture

The architecture of all the agents is similar; however, each exemplifies a particular architecture that uses specific components to reflect its functionality. The BA/SA architecture in the proposed framework consists of the following main modules (Figure 2). The BA carries product desired by the customer and communicates with the MA to obtain entry to negotiations (if they do not have a pre-known SAs with a relevancy to the current request). When entry is granted, the BA proceeds to negotiate with an appropriate SA (Figure 1). Upon completion of negotiations, the BA informs the customer about the result. A SA offers products and services to BA (directly or through a MA). The SA answers queries for information about its owner's products or services, responds to XML messages, and enters into negotiation with BAs.

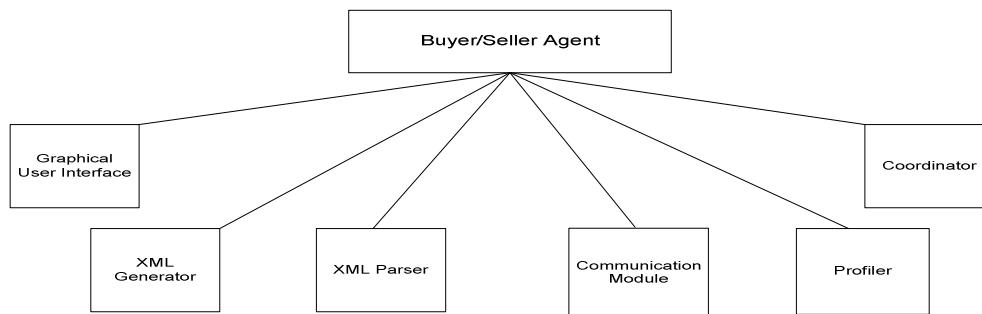


Figure 2: BA/SA module view

3.2 The Communication Protocol Module

The Agents cannot talk or interact to each other unless there is a communication protocol. This module is considered to be the backbone of the whole communicating process among all agents. Communication in the system is accomplished through exchanging XML-based messages. These messages are transferred between the agents through TCP socket clients (Karacapilidis and Moraitis, 2001b, 2000b) (Karacapilidis and Leckne, 2004). Therefore, XML message is a medium that allows agent to talk to each other via components of the message analyzer as followings:

- The message sender: it is responsible for sending XML messages to other agents.
- The message receiver: it is responsible for receiving other XML messages sent by other agents.
- The message parser: it decomposes the message into its individual so that the message analyzer can handle it.
- The message analyzer: it interprets any message received and transforms it into an activity.

There are several XML messages handled by the BA:

- There are five messages, *custSpecMsg*, *custSpecUpdReqMsg*, *custSpecUpdAnsMsg*, *purchInitReqmsg* and *purchInitAnsMsg*. Using the first message, the customer is able to inform (his/her) BA about the product he/she would like to buy together with a related specification (criteria, preferences, constraints, feature, etc). The next two messages are concerned with an update to the specification (at information regarding the customer's opinion for a feature value, a preference, or some argument introduced by the SA). The last two message types concern the interaction that is proactively initiated by BA's request about whether the customer is interested in purchasing a new product (which has just been launched by the SA).

There are several XML messages handled by the SA:

- There are three message namely *merSpecMsg*, *merSpecUpdReqMsg* and *merSpecUpdAnsMsg*. Using the first one, a SA becomes aware of the products it can sell.

In addition, a SA can use a *merSpecUpdReqMsg* message to ask its owner about additional information (e.g., when new criteria or preferences, not currently available in its selling database, appear in an offer request). The merchant's reaction is represented by a *merSpecUpdAnsMsg*. Each time the database of a SA is updated (either when a new product is introduced or an existing product is updated due to promotion reasons), the SA uses *newProdMsg* message to broadcast its characteristics to the possibly interested BAs of the system. Each time a new SA is created and uploaded, it broadcasts *newSellAgMsg* message to announce its presence to the MA/BAs of the system. Each time a SA launches a new product or an offer promoting one of its already existing products, it sends *newOfferAnnMsg* message to its known BAs. Some messages format is as follows:

- An agent registering with a MA:


```

      <?xml version="1.0"?>
      <agent>
      <agent_id>
      </agent_id>
      </agent>
      </keywords>
      </price>
      </price>
      </product>
      
```
- A SA offering a product:


```

      <?xml version="1.0"?>
      <product>
      <product_name>
      </product_name>
      <product_desc>
      </product_desc>
      <product_serial>
      </product_serial>
      <price>
      </price>
      <agentt_id>
      </agentt_id>
      </product>
      
```
- A BA's query:


```

      <?xml version="1.0"?>
      <product>
      <product_name>
      </product_name>
      <keywords>
      </keywords>
      </product>
      </price>
      </price>
      </items>
      </items>
      </detail>
      
```
- MA responses to BA's query:


```

      <?xml version="1.0"?>
      <matches>
      <product_name> </product_name>
      <product_desc></product_desc>
      <agent_id></agent_id>
      <agent_ip></agent_ip>
      <agent_port></agent_port>
      <product_serial></product_serial>
      </matches>
      
```
- BA requesting details from the SA:


```

      <?xml version="1.0"?>
      <details>
      <pid>
      </pid>
      <pname>
      </pname>
      <pdesc>
      </pdesc>
      <price>
      </price>
      <items>
      </items>
      </detail>
      
```

3.3 The Mall Agent

The MA controls the system by providing the required services to other agents (BA, and SA). The MA has several components, see please Figure 3. The MA tries to relate a customer to the correct merchant through analyzing steps. It matches the buyer's request with the available SAs in the system and sends the ordered matching list according to the degree of relevancy to the BA. After that, it sends the seller information when the buyer selects a product. The interaction with the SA includes posting or retrieving the product details. Also, the MA allows offers to be sent to the seller and if s/he agrees, then it will accept the deal. The following diagram summarizes the overall interaction of the system actors.

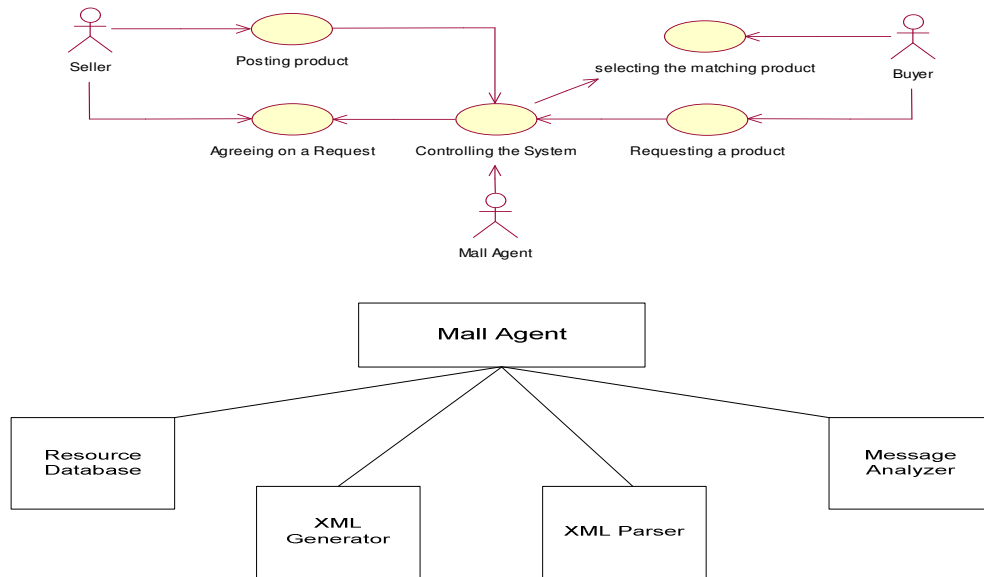


Figure 3: MA module view

3.4 Profiler Database

The profile module is responsible for keeping track of all customer transactions. In addition, it allows the customer to retrieve his or her profile and makes changes to it. All profiles are stored in the local database. In order to store or retrieve a profile it needs to interact with the responsible agent. Figure 4 shows the databases; in the BA and SA sides; used for storing the historical items (s/he) bought and the keywords used for describing the items. Furthermore the common database of the system that stores the overall information regarding the products and the agents is shown in Figure 5.

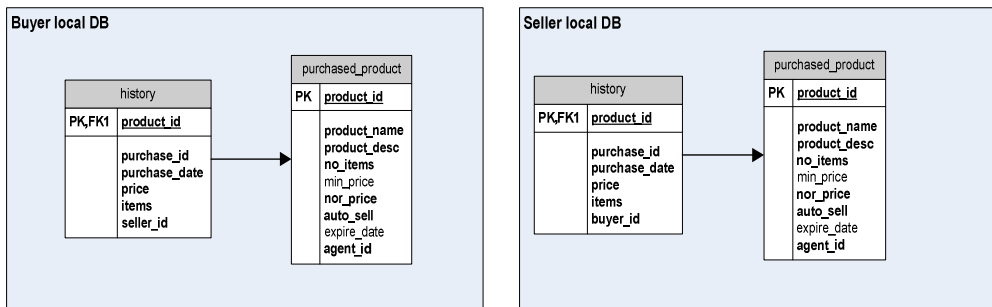


Figure 4: Buyer/Seller agent's local databases

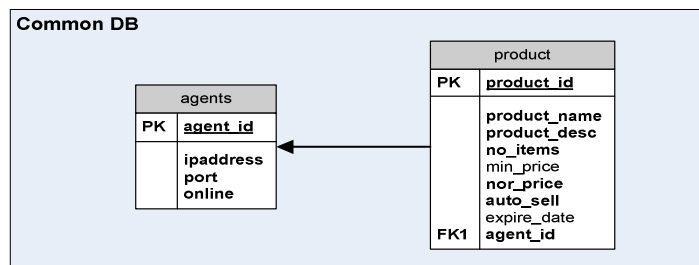


Figure 5: Common database (controlled by the MA)

4. SETTING AND MODIFYING THE CP

A Customer Preferences (CP) contains predictive information about the customer's interest toward product features. The BA allows the customer to set the initial preferences when running it for first time, i.e. the number of results to be displayed to the customer, and the default MAs (e-commerce portal) that will receive the customer's request. Also, s/he can modify these preferences at any time later (Degenmis, 2004). The customer can enable/disable the e-commerce portals to represent his/her preference regarding which portal s/he would like to send the request to. Moreover, the BA 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 request & the default e-commerce portals*/
[CP].Customer_results/request ← 10
e-commerce_portal [1] ← 'MA1'
e-commerce_portal [2] ← 'MA2'
e-commerce_portal [3] ← 'MA3'
/*if the customer wants to modify the number of results to be displayed per request*/
if Customer.modify (Customer_choicoe_results/request) then
/*allow him/her to modify only if the number is greater than or equal to 10*/
if Customer_NewChoice >= 10
Customer_choicoe_results/request ← Customer_NewChoice
end if
end if
/*if the customer wants to add more e-commerce portals to the default ones*/
If Customer.add_ e-commerce portal (Customer_NewPortal) then
/*check that s/he is not trying to add e-commerce _portal that has been added*/
If Customer_NewPortal does not belong to the default e-commerce Portals then
e-commerce Portal ← e-commerce _Portal + Customer_NewPortal
end if
end if
/*If the customer wants to remove a e-commerce Portal from the default list*/
If Customer.delete_ e-commerce portal (Customer_Oldportal) then
/*check that s/he is not trying to remove one of these portals*/
if Customer_OldPortal does not belong to ('MA1';'MA2';'MA3') then
e-commerce Portal ← e-commerce portal-Customer_Oldportal
end if
end if

```

4.1 Request Filtering by the BA

When the customer enters the request in NL, the BA removes what is called noisy words from it before it starts searching the CP, (Daniel Lemire, 2004). These words like the prefix, suffix, and other words. Examples are (re, -or, -er, at, the, on, in, an, his, our, we, are, etc.). The BA 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 Keyi belongs to (noisy words N)
/*if there are noisy words in customer's request, remove them*/
Q ← Q - Keyi
end if
end for
Return (Q)

```

4.2 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 BA is different from the one submitted by the customer to be more representative of the customer's intent. The BA adds to the request the keywords attached to the items that have the most relevancies to the customer's request to represent the customer's intention and delegates it to the MA.

Input: Request in NL but already filtered: $Q \langle \text{key}_1, \text{key}_2, \dots, \text{key}_n \rangle$, Sorted array W (Id, SA, title, total weight)

Output: Request refined by adding some other keywords from the profile

$m \leftarrow \text{count}(\text{Id in } W)$

*/*for all items in W: array of results retrieved from the URL table*/*

for $i \leftarrow 1$ to m

 if $W[i][4] > 0.5$ then

*/*If the weight of this item > 0.5, then*/*

$n \leftarrow \text{count}(\text{keywords } \langle k \rangle \text{ saved in the SA for } Id_i)$

 for $j \leftarrow 1$ to n

*/*for every keyword in this item but not in customer request Q, add it to the Q*/*

 if $k_{i,j}$ does not belong to Q then

$Q \leftarrow Q + K_{i,j}$

 end if

 end for

 end if

end for

Return (Q)

5. SYSTEM IMPLEMENTATION

A prototype implementation of the system has been developed. In this prototype, three different types of agents are implemented as mentioned. The implementation of the agents varies depending upon whether they act at the buyer or at the seller side. The agents are implemented and run in a Java Virtual Machine (JVM) JDK 1.4 on Windows XP machine. Agents can access relational databases, for which they are able to access and manipulate a set of tables that contain models of the others.

XML technology; as mentioned previously is used, since it allows information and services to be encoded with a meaningful structure. The agent framework comprises the hierarchical structured agent communities based on a portal-agent model. The MA is acting in this application as a portal agent. The MA is the representative of the community and allows all BAs/SAs in the community to be treated as one normal agent outside the community. A MA has its role limited in a community, and another high-level portal agent (MA) may manage the MA itself. A MA manages all BAs/SAs in a community and can multicast a message to them. Any BA/SA in a community can ask the MA to do multicasting its message. This model is realized with the agent middle-ware. The agent middle-ware is primarily designed to act as a bridge between distributed physical networks, creating an agent-friendly communication infrastructure on which agents can be organized in a hierarchical fashion more easily and freely like:

- The BA starts by initiating a communication with the MA using the agent communication protocol.
- The MA identifies the BA and replies with either accept/reject response.
- The BA sends the original/refined customer's request to the MA.
- The MA delegates the request to the registered SAs.
- The SA upon receiving a request attempts to interpret it by itself. If interpretation is successful, the SA will report to the MA with a certainty value.
- IF the SA cannot interpret the request as its own then it reports with zero certainty value.
- The SA returns the result to the MA.

5.1 Request Routing by the MA

The MA creates its own SA's attributes table and modifies it over time automatically to reflect the popularity of each SA. These SA's attributes reflect the profile of each SA to the BA and will be used in the routing mechanism. The MA will do the routing based on the following criteria:

- The MA learns the profile and popularity of each SA for upcoming relevant request based on customer's historical responses (how much the SA services this BA's previous requests) and creates its own communities of SAs.

6. EXPERIMENTAL RESULTS AND DISCUSSION

We have conducted experiments to make a consistent evaluation of the system's performance, how much the BAs support and negotiation/collaboration will enhance the customer's satisfaction and the cost of getting the results. In the evaluation phase, the system has been installed on a PC running Windows XP. In the experiment, we have chosen a simple scenario with 20 BAs and 1000 SAs being registered with the MA. *In the first phase of the experiment*, each customer submitted 20 different requests for the first time to his/her BA, which in turn refines the requests; based on the customer's preferences. As soon as the customer makes his/her request, the BA formulates an XML message of the refined query and routes the XML message to the MA directly asking for the relevant SAs. When MA receives this request, it responds with the matched SAs that have been registered with item's description relevant to the submitted request. Then the BA proceeds to negotiate with an appropriate SA. Upon completion of negotiations, the BA informs the customer about the results; the transaction procedure shown in Figure 6. Then we calculated the Customer's Satisfaction (CS) and the cost of getting the results. *In the second phase of the experiment*, each customer submitted again 20 requests being submitted before to his/her BA. In this case, the BA should proceed to negotiate with specific SA direct, the transaction procedure shown in Figure 7. Then we calculated the cost of getting the results. CS is the ratio of the number of relevant items returned to the total number of irrelevant and relevant items returned. While the cost of getting the results is the ratio of the number of messages exchanged in the second phase to the total number of messages exchanged in the two phases. We captured the number of messages exchanged in the experiment using a dedicated agent. The results depicted in Figure 8 prove that our framework makes sense and allows locating the desired items in a more qualitative way consistent with the customer's need. The results also prove the hypothesis that the cost of getting the result is being improved, and in fact also confirms the fact that our system acts in a peer-peer fashion.

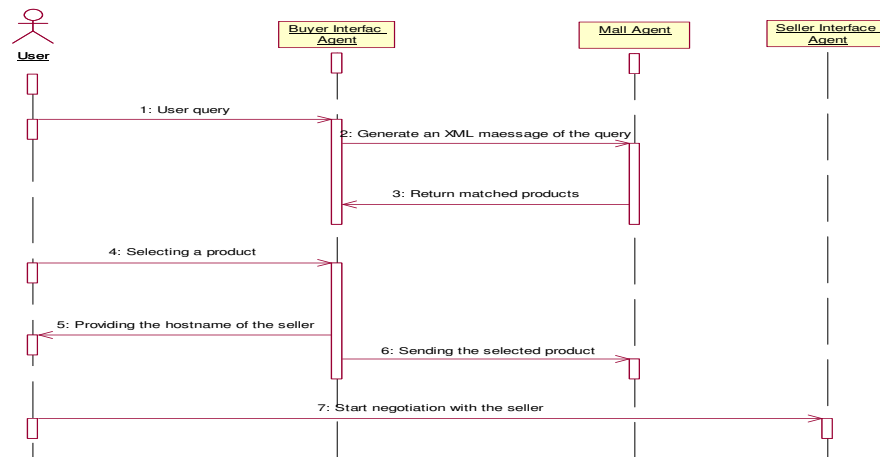


Figure 6: Transaction procedure while requesting a product for 1st time

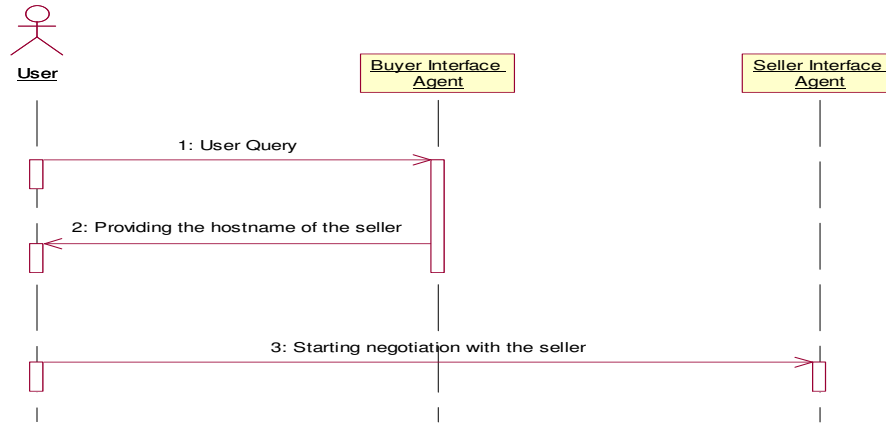


Figure 7: Transaction procedure while requesting a product being requested before

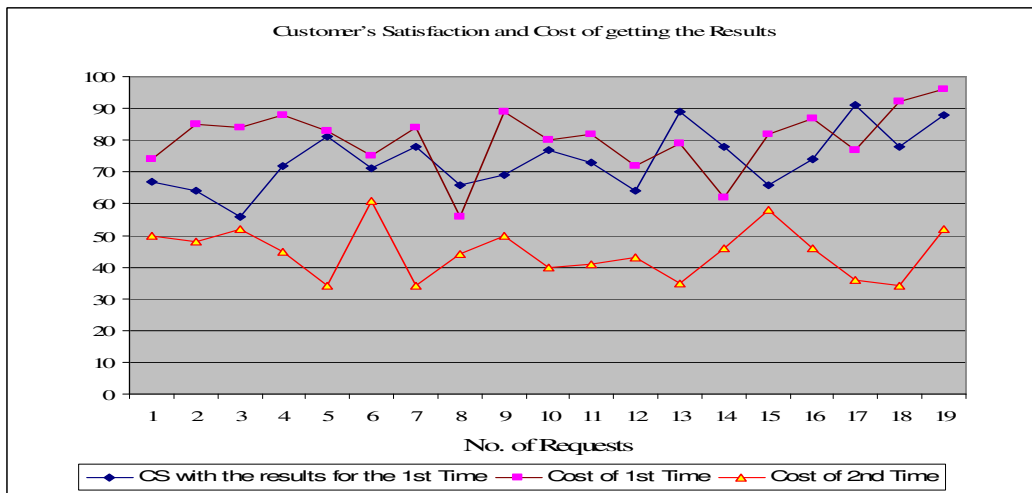


Figure 8: Customer's satisfaction and cost of getting the results

7. CONCLUSIONS AND FUTURE WORK

The main goal of this paper has been to develop an agent-mediated framework for e-commerce. This goal has been achieved with a design of multi-agent system architecture. A prototype of the system is implemented and tested. As a future work, we have set up a goal of studying system scalability and identifying performance bottlenecks, extending the ontological support beyond that very simple ontology of customer preferences, and adding more realistic strategy modules. We will report on our progress in subsequent papers.

ACKNOWLEDGMENTS

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

REFERENCES

A. Chavez and P. Maes (1996), *Kasbah: An Agent Marketplace for Buying and Selling Goods*, In the Proceedings of the First International Conference on the Practical Application of Intelligent Agents, London, UK, 75-90.

- Bartolini, C., Preist, C., Jennings, N.R. (2002), *Architecting for Reuse: A Software Framework for Automated Negotiation*, Proceedings of International Workshop on Agent-Oriented Software Engineering, Italy, LNCS 2585, Springer, 88–100.
- Bartolini, C., Preist, C., Jennings, N.R. (2005), *A Software Framework for Automated Negotiation*, Proceedings of SELMAS, LNCS 3390, Springer, 213–235.
- Daniel Lemire, 2004, *Scale and Translation Invariant Collaborative Filtering Systems*, *Information Retrieval*, Vol.7, 1-22.
- Degemmis M., Lops P., Semeraro G., Costabile M.F., Lichelli O., Guida S.P. (2004), *A Hybrid Collaborative Recommender System Based on User Profiles*”, Proceedings of the Sixth International Conference on Enterprise Information Systems, Vol. 4, 162-169.
- Elhadi Shakshuki, Saad Abu-Draz (2005), *Agent-mediated E-commerce System*, Proceedings of the 19th International Conference on Advanced Information Networking and Applications.
- eBay: <http://www.ebay.com>
- J. Collins, W. Ketter and M. Gini, (2002), *A Multi-agent Negotiation Tested for Contracting Tasks With Temporal and Precedence Constraints*, *Int’l Journal of Electronic Commerce*, 7(1):35-57, 2002.
- M. Bellosta, I. Brigui, S. Kornman and D. Vanderpooten (2004), *A Multi-criteria Model for Electronic Commerce*, Proceedings of the ACM Symposium on Applied Computing, 759-765.
- Maes P., Guttman R., and Moukas Al., (1999), *Agents that Buy and Sell*, *Communications of the ACM* 42(3), pp. 81-91.
- M. C. Jonker and V. Robu (2004), *Automated Multi- Attribute Negotiation with Efficient Use of Incomplete Preference Information*, Proceedings of 3rd International Joint Conference on Autonomous Agents, New York.
- MARI, <http://www.media.mit.edu/~gtewati/MARI>.
- N. Karacapilidis and P. Moraitis (2001), *Building an Agent-Mediated Electronic Commerce System with Decision Analysis Features*, *Journal of Decision Support Systems*, Vol. 32, No 1, 53-69.
- N. Karacapilidis and T. Leckne (2004), *A Recommendation Based Framework for Online Product Configuration*, In Proceedings of the 6th International Conference on Enterprise Information Systems, ICEIS Press, Vol. 4, 303-308.
- P. Anthony and P. Jennings (2003), *Developing a Bidding Agent for Multiple Heterogeneous Auctions*, *ACM Transaction on Internet Technology*, Volume 3, 185-217.
- S. Fonseca, M. Griss and R. Letsinger, 2001, *An Agent- Mediated E-commerce Environment for the Mobile Shopper*, HP Technical Report HPL-157.
- S. Abu-Draz and E. Shakshuki (003), *Agent-Based Online Trading System Advances in Artificial Intelligence*, Proceedings of 16th Conference of the Canadian Society for Computational Studies of Intelligence, Canada.
- Wei Y., Moreau L., Jennings N., (2003), *Recommender Systems: A Market-Based Design*, Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-agent Systems, ACM Press, 600-607.

Received: March 12th 2005

Accepted in final format: August 29th 2006 after one revision.

About the author:

Tarek Helmy is teaching in the College of Computer Science and Engineering at King Fahd University of Petroleum and Minerals, KSA. He holds a PhD from Kyushu University, Japan in 2002. His research interests include Operating Systems, Multi-Agent Systems, Artificial Intelligence, Web services, and Intelligent Systems.