International Journal of u-and e-Service, Science and Technology
Vol.7, No.4 (2014), pp.171-186
http://dx.doi.org/10.14257/ijunnesst.2014.7.4.17

Collaborative Filtering Methods for Identifying Relevant Adverts to a Real Estate Mobile Agents

Jinan Fiaidhi[1], Niki Shakeri[2], Sabah Mohammed[3] and Tai-hoon Kim[4]

[1, 2, 3]*Dept. of Computer Science, Lakehead University,Thunder Bay, ON, Canada*
[4]*Department of Convergence Security, Sungshin W. University, Korea*
[1]*jfiaidhi@lakeheadu.ca,* [2]*nshakeri@lakeheadu.ca,* [3]*mohammed@lakeheadu.ca,*
[4]*taihoonn@sungshin.ac.kr*

Abstract

Critical surveys show that mobile agents are unable to delegate properly the user's requests resulting into having a semi-automated system requiring the intervention of the clients. This paper describes a new brokering architecture called "Meta Broker". It interacts with relevant online real estate websites, based on the user's desires to recommend possible matching advertisements. The proposed architecture includes multiple agents: Shopping Bots; Product Brokering; Recommender System and Data Mining. The objective is to present an intelligent gateway that can elicit the client desires and finds highest possible matches for advertisements on the real estate market. The Meta Broker utilizes collaborative filtering techniques on previously achieved data as well as upon previous client feedbacks to categorize the new client behavior. The extracted knowledge directly affects the ranking of the adverts that are compatible with the user's characteristics and the client queries.

Keywords: *Multi-Agents, Android Apps, Collaborative Filtering, Recommender System*

1. Introduction

Software agents help to automate a variety of tasks including those involved in buying and selling products over the Internet [1, 2]. They differ from "traditional" software in that they are personalized, continuously running, and semi-autonomous. However, "mobile agent (MA) is a software agent that can transport its state from one environment to another, with its data intact, and be capable of performing operations appropriately in the new environment" [3]. Most of the applications for mobile agents are on electronic commerce (e-commerce) [4]. Traditionally clients utilizing agent systems are solely responsible for collecting, interpreting information and comparing information provided by the mobile agents on both merchants and products. Our system, however, delegates many of these human tasks to automated agents where they are programmed with the power of resource awareness. Increasingly, resource awareness plays a central role in many distributed and mobile computing applications. There applications rely on information about the available resources and services in order to provide novel functionality. While many resource-aware application drivers for estate agents already exist in mobile and distributed computing, very little systems research has explored how best to program these applications, to express their constraints, and to allow efficient implementations on highly dynamic real-world platforms. This paper proposes the Meta Broker system architecture, which includes run-time system support for estate agents resource-aware applications. Meta broker allows users to express their interest on renting or buying a new housing unit, as well as trade-offs between quality of result (QoR), latency and cost. The goal is to produce applications that use resources efficiently and that can be run on

ISSN: 2005-4246 IJUNNEST

diverse resource-constrained platforms ranging from laptops to personal digital assistants and to smart phones. Meta broker's run-time system manages QoR and cost trade-offs dynamically by tracking resource availability and locations, brokering usage/pricing agreements and migrating programs to nodes accordingly. Client behavior model permeates the Meta broker system not only to express their resource needs and QoR expectations but also to predict and guide the match making process according to similar clients' selections and feedbacks. Although we are still early in the system development, the initial prototype version demonstrated a promising success.

2. Overview of the Meta Broker Framework:

In this section we introduce the Resource Aware Meta Broker framework, which allows clients to monitor the resources related to their wish list and to programmatically express policies for the selection of such resources. The framework is based on a notion of hierarchical groups of agents, which act as resource containers for the computations they sponsor. Resources are manipulated by the developer using resource descriptors, whose operations are specified by a resource metadata. In this section, we overview the framework and describe its semantics. In the next sections, we discuss a prototype implementation of the Resource Aware Meta Broker framework in Java and Android. Figure 1 illustrates a bird view of the Meta Broker framework.

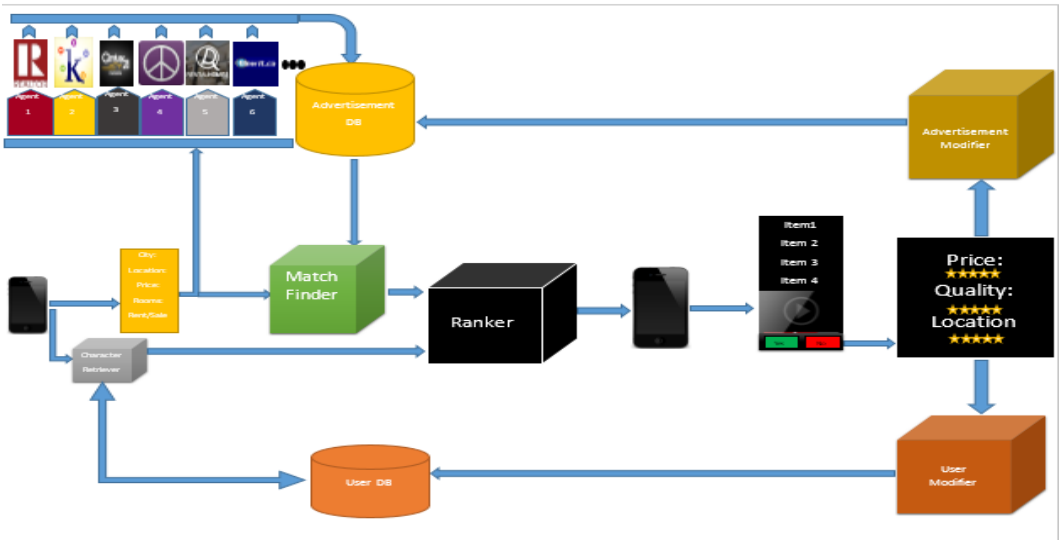


Figure 1. The Meta Broker Framework

The Meta Broker as a black box include three major parts including the android client interface, a server respondent (PHP programs) and resource repositories (in MongoDB). Figure 2 illustrates this framework in its general form. The user can send a request from his/her mobile device or computer to the Server. The Server processes the request and sets the user's characteristic via the database. Then the Server has to concurrently communicate with different websites to extract the required information. The data is then made available on the Server, which sends the data back to the cell phone, enabling the user to check the results and evaluate them with his/her feedback. Feedback will be sent to the Server and the Server has to modify its understanding of both the user and the advertisement.

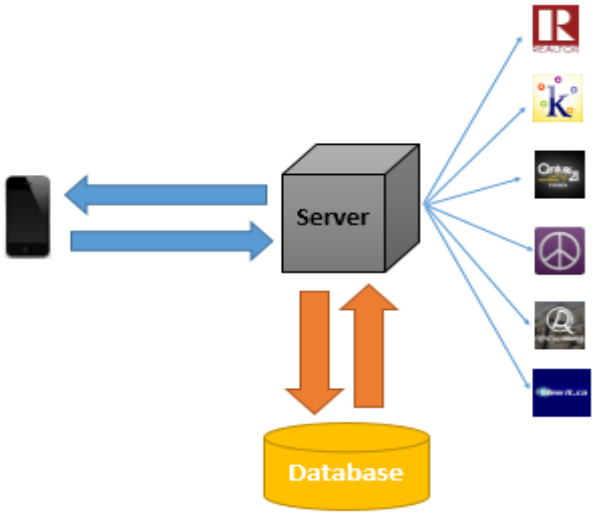


Figure 2. General View of the Meta Broker

Moreover, we can further divide the server respondent component into two parts: *Query Finder* and *Feedback analyzer*. Query finding involves getting the user's query, analyzing it, and displaying it on the screen of the user's mobile device. On the other hand, the Feedback Analyzer is responsible for modifying Meta Broker's understanding of both the user and the advertisements. Together the Query Finder and the Feedback Analyzer complement each other by enabling Meta Broker to present advertisements that are specific to the user's taste (see Figure 3).

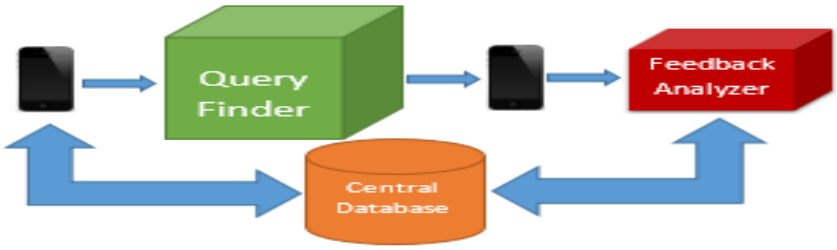


Figure 3. The Server Respondent Components

The details on these two components are provided in the following subsections.

2.1. Query Finder

The Query Finder is responsible for finding all the existing potential advertisements which are related to the user's request and for ranking the advertisements based on the user's characteristics. The Query Finder contains five components including: *Character Retriever*, *Request Inquiry (UI)*, *Contacting or Agents' Room*, *Match Finder*, and *Ranker (see Figure 4)*.

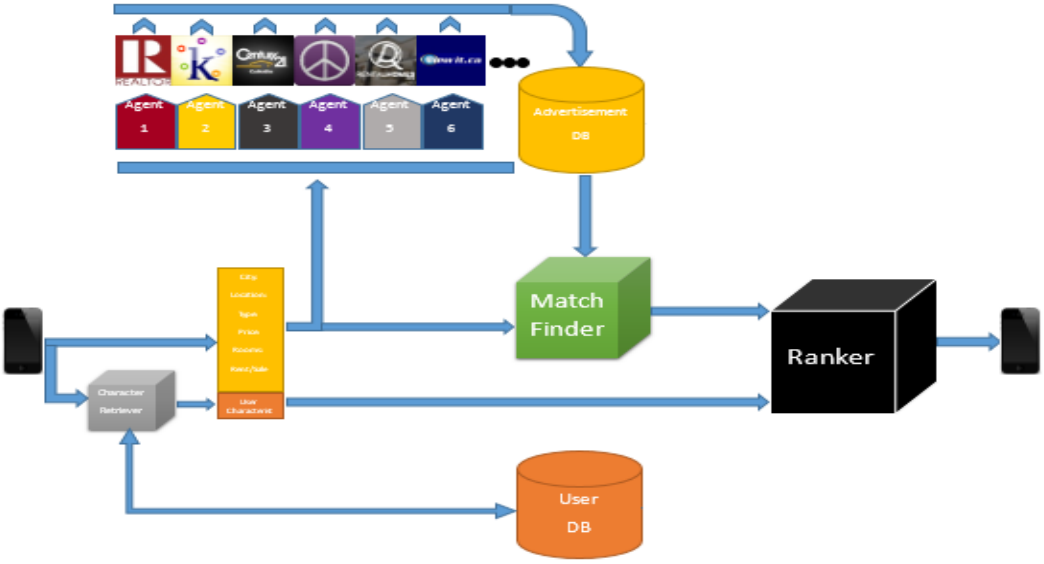


Figure 4. The Query Finder's Components

The logic behind the Query Finder can be summarized as follows:

- Extract the user's characteristics from the database (Character Retriever)
- Get the query through the UI
- Pass the query to the Agent's room as well as to update this component
- Extract the related results from the central database (Match Finder)
- Rank the results based on user's taste (Ranker)

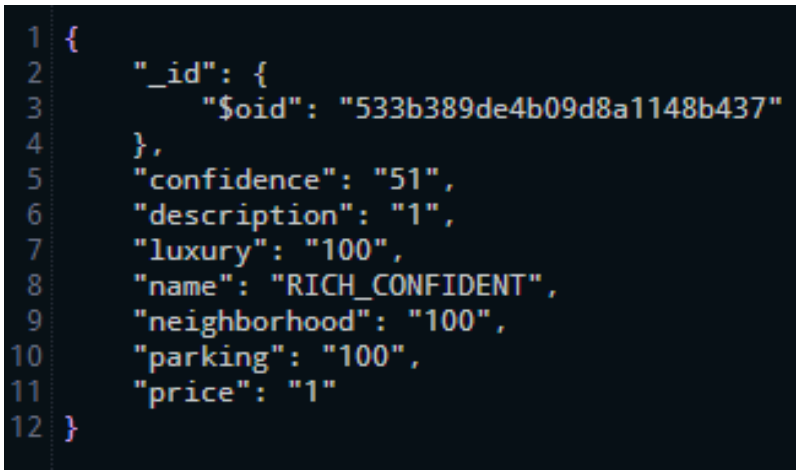
Character Retrieving

For the Meta Broker to learn about the user's behaviour, each user and advertisement is characterized and located in the database. This step is responsible for reading the user's characteristic from the database and for setting the local variables on the mobile application's side. The Ranker component will need this information and use it to match the sorted list with the taste of the user. Each user is modeled through a vector matrix with six variables or categories. For each category the user is assigned a number from 1-100 to illustrate the taste of each individual. If the user is assigned a low value this means that the user is not too concerned about that specific category. In turn, a high value on this scale indicates that the user really cares about that particular category. The following are six categories which determine user's characteristics.

- **Luxury** shows how much the user cares about the quality of the place including both the inside and the outside (back yard, driveway, walkway etc.). Higher numbers in this field represent users that are looking for a luxurious place.
- **Neighborhood**, number 1 in this category means that the user does not pay attention to the quality of the neighborhood.
- **Parking**, higher values in this category indicate that the user has a higher interest in parking availability and quality.
- **Price**, users with limited budgets will be represented by higher numbers because the cost really matters to them.

- **Description** is the fifth category which is geared towards the advertisement itself and indicates the user's interest in the quality of the advertisements' descriptions. The higher value is given to a well described advertisement that has all the required pictures, contacts, and essentially provides information for all the main questions the buyer/renter will have.
- **Confidence** is the last category that measures the confidence level of the other 5 values. Whenever a user provides feedback on an advertisement the confidence level should increase because the more feedback the user provides the easier it is to categorize that consumer's behaviour. For this category, unlike the others, the range is from 1 to infinity. Every new user is designated a confidence level of 1 to start with. The more feedback a new user provides on the advertisements he/she views, the more the confidence level will increase. As the confidence level increases the easier it will become to provide the user with the type of advertisements he/she is interested in. Thus, the user with a higher confidence level has more impact on the advertisement modification.

The rate of change in these numbers is higher for the users with low confidence levels or immature users. The reason is new users are under investigation and Meta Broker needs to gain their feedback to better characterize them. Users with high confidence levels will not have their values change a lot because their consistent feedback enables their behaviour to be learned. The main reason that the user's characteristics are required is to sort the available advertisements and filters out the matching ones. These six categories will be codified at the repository as a JSON file (see Figure 5).



```
{
    "_id": {
        "$oid": "533b389de4b09d8a1148b437"
    },
    "confidence": "51",
    "description": "1",
    "luxury": "100",
    "name": "RICH_CONFIDENT",
    "neighborhood": "100",
    "parking": "100",
    "price": "1"
}
```


Figure 5. A JSON File for Coding a User Characteristic

For a new user registered with Meta Broker, the default values of all the character fields will be set to 50 and the confidence level to 1. Meta Broker will adjust those numbers to reflect the user's real characteristics by iteratively processing the user's subsequent feedback. Meta Broker can model 6 built-in characters (either of poor, normal, rich, joined with mature or immature) to show the list of ranked advertisements is dependent on the user's characteristics. Table 1 illustrates the details of these built-in characters.

Table 1. User Characteristics

Type of User	Confidence	Ad Descriptio	Luxury	Neighborh ood	Parking	Price	Description of this type of user/profile
Poor immature	1	1	1	1	1	100	This user mostly cares about price, Meta Broker is uncertain about this users profile
Poor confident	51	1	1	1	1	100	This user mostly cares about price, Meta Broker is certain about the user's profile
Normal immature	1	50	50	50	50	50	This user's feedback is an accurate depiction of the ad, Meta Broker is uncertain about the user's profile
Normal confident	51	50	50	50	50	50	This user's feedback is an accurate depiction of the ad, Meta Broker is certain about the user's profile
Rich immature	1	50	100	100	100	1	This user is very picky about everything related to the quality, because he/she has a lot of money and wants the best possible choice
Rich confident	51	50	100	100	100	1	This user is very similar to the above with one difference which is Meta Broker is certain about the user's characteristics

Request Inquiry

To make an enquiry the user needs to complete a form, providing details through User Interface (UI). The information that the users provide includes: City (Meta Broker covers 111 cities in Canada. By selecting any of those 111 cities the search request can be started), Minimum Price, Maximum Price, Number of Bedrooms, Location, Type of Rent (Apartment/Condos, House Rental, Room Rental, Commercial or Storage parking) and Type of Sale (House, Condos, Land, or Commercial). All of these parameters are designed to help users view the most relevant advertisements by selecting their preferences. Figure 6 shows the layout for Meta Broker's UI. The location textbox can be filled with any common notation such as, an address or postal code. It can even affect the filter by mentioning a street name. After filling in the location, there are two spinners which show the options for rent and sale. If the user does not pick any of them, it is supposed that the user is looking for both options. Then there are four radio buttons (poor, normal, rich, new) and a switch (confidence on or off) which are all connected to the Character Retriever. All of these characters will not be affected by the User Modifier because they are surrounded by a lock/unlock mechanism so they are always usable. Also, we can pick "New" which is a kind of "Normal immature" user, which has the ability to be altered to converge to the user's behavior.

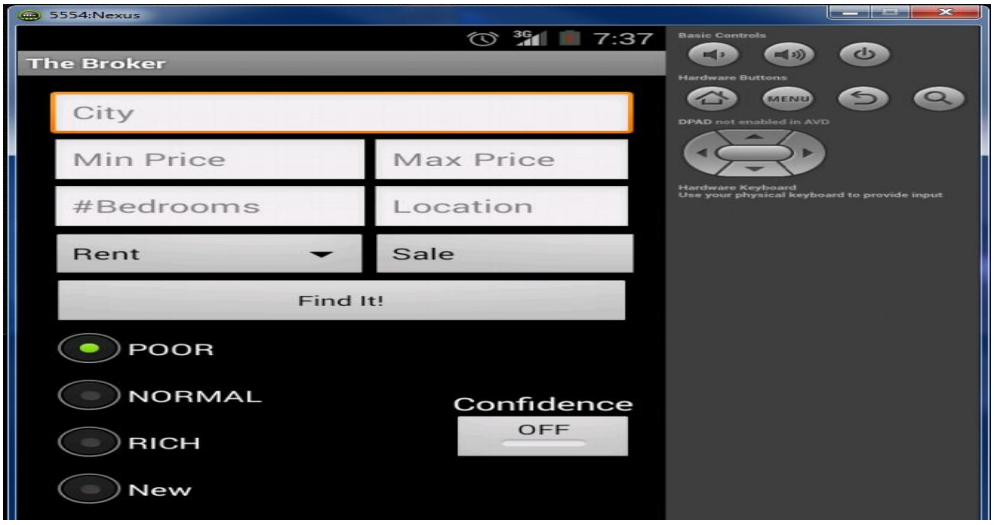


Figure 6. The User Interface on the Android Emulator Side

Contacting Agents

The information gathered by the User Interface will then be connected to the Agent's Room, which is a room full of agents. Each agent is responsible for extracting the information from the assigned website and updating the database (see Figure 7). Each Agent concurrently contacts their corresponding website and passes the user's request to that site. Then the agents extract the relevant information and update the central database.

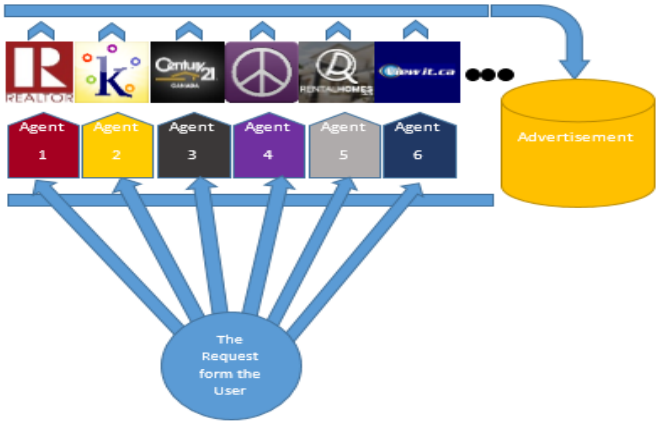


Figure 7. Agents' Room Architecture

The central database may not contain all the required information because if there is no search for advertisements with particular criteria, there is no need to gather and save those advertisements. If there is a frequent request from a number of users, advertisements pertaining to that request will be consistently updated in the central database. So, the central database is compatible with users' requests. The architecture of the Agents' Room follows a multi-agent system pattern. Each agent should be set up and added as a module to the system based on the related website. The extracted information is coded with 12 factors including: Current time to find how old the ad is, Text that refers to the title of the advertisement, Link

to the advertisement which will take the user to the website the advertisement is listed on, Number of rooms that the advertisement lists, Price as it appears in the advertisement, Location of this house/apartment, Rent/Sale type, and Confidence level of the user related to the advertisements. The rest of the extracted factors are Price Percent, Luxury Percent, Neighborhood Percent and the Parking Percent. Meta Broker evaluates those factors out of 100%. For example, a house that is considered to be 100% Price Percent category means that it is a high quality. Likewise, a house valued at 50% has a reasonable price for the condition of the house. So in general, the Agents' Room is a multi-agent system design for extracting the data from the internet and converting that data into useable information that will be added to the central database.

Updating the Database

The results from the agents will be inserted into the database and update the central advertisements' database as shown in Figure 8.


```
{
    "_id": {
        "$oid": "5376552777660830db000088"
    },
    "confidence": "2",
    "text": "PLEASANT_2_BDRM_APARTMENT_VICTORIA_PARK_ST_CLAIR_E",
    "price": 1025,
    "link": "http://www.kijiji.ca//v-2-bedroom-apartments-condos/city-of-toronto/
    pleasant-2-bdrm-apartment-victoria-park-st-clair-e/589021839?src=topAdSearch",
    "location": "",
    "rooms": "",
    "rentorsale": null,
    "time": "2014-05-16",
    "pricepercent": "50",
    "luxurypercent": "25",
    "neighborhoodpercent": "25",
    "descriptionpercent": "25",
    "parkingpercent": "25",
    "image": "http://i.ebayimg.com/00/s/NDEzWDU1MQ==/z/e3cAAOxy9X5TY6dg/$_2.JPG"
}
```


Figure 8. A Sample Advertisement in the Database

Match Making

The relevant advertisements from the central database will be filtered to be compatible with the initial query. For example, if the user desires a house with higher than three rooms, the Match Finder will filter all the non-relevant advertisements such as those with one or two bedrooms. Once the Match Finder is done all the results are relevant to the request. The Match Finder has access to the database which contains manipulated data. In general, displaying the related hits without sorting them is the duty of this component.

Ranking

Meta Broker will then rank the advertisements based on the user's profile. The Ranker has access to the characteristics of all the users and all the advertisements in the target list, through the Match Finder. With the usage of both data sets, the Ranker tries to sort the advertisements for users based on what they are looking for.

The formula below (Equation 1) shows the compatibility of an advertisement (A) for a user (U).

$$CUA = \sum_{1}^{5} A[i] \times U[i]$$

Equation 1: User and Advertisement Compatibility

A1, A2, A3, A4 and A5 are five numbers out of 100 which show Meta Broker's understanding of the advertisement's parking, neighborhood, luxuriousness, price, and description, respectively. Also, U1, U2, U3, U4 and U5 are the ratings, out of 100, based on a specific user's preferences for parking, neighborhood, luxuriousness, price, and description, respectively. The Ranker uses Equation 1 to calculate the user's and advertisement's compatibility, CUA. The resulting CUA value will then be used to rank the advertisements in a list that is sent back to the user. Essentially, each user has his/her own weight for the different fields of an advertisement. For example, some people might care more about the luxuriousness of a place compared to the price. The Ranker ensures that users with a high interest in the luxuriousness of a place will get different results from users who are more concerned with the price, because for either type of user the Ranker will produce a different list.

Suppose that we have two users called User1 and User2, also two advertisements called Ad1 and Ad2. Table 2 shows the characteristics of each of them. Now after applying Equation 2 to the test sample, Table 3 illustrates the calculated value for each pair and the ranking for each user. The weighted average used in the Ranker component helps to find the compatibility between each pair (user, advertisement) and uses it to achieve the proper ranking for our users and give the best performance. The table shows that users are getting differing results because they have opposite desires. The table shows that the CUA for user 1 is higher for ad 2, which means that ad 2 better fulfills user1's request. Whereas, User2 has a high CUA for Ad 1, meaning Ad1 is a better choice for User2 and as a result will be ranked higher than Ad2.

Table 2. Sample Characteristics of Users and Advertisements

User1	User2	Ad1	Ad2
{	{	{	{
U1:80	U1:20	A1:40	A1:60
U2:50	U2:50	A2:50	A2:50
U3:50	U3:50	A3:50	A3:50
U4:50	U4:50	A4:50	A4:50
U5:20	U5:80	A5:60	A5:40
}	}	}	}

Table 3. The Ranking for the Sample Test

	Ad1	Ad2	Ranking
User1	CUA=11900	CUA=13100	{Ad2,Ad1}
User2	CUA=13100	CUA=11900	{Ad1,Ad2}

At the end, advertisements will be presented to the user as a ranked list. The user will get access to the advertisements' information.

Feedback Analyzer

The Feedback Analyzer is responsible for showing the list of results and receiving feedback from the user for updating the users' profiles and advertisements characteristics. Figure 9 shows the architecture of the Feedback Analyzer.

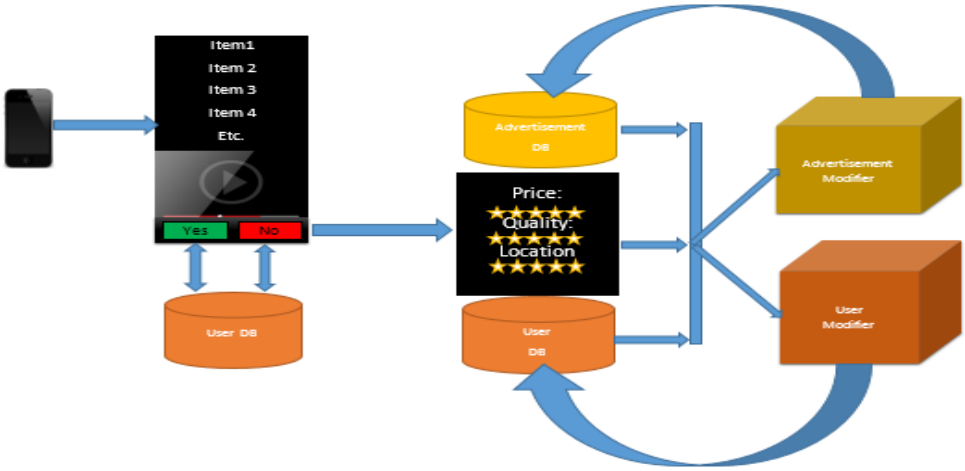


Figure 9. The Feedback Analyzer's Architecture

After the Query Finder phase a list of ranked advertisements appears in the User Interface. As it is shown in Figure 9, the UI includes a list of several advertisements, an image view, and yes and no buttons. Each advertisement is displayed with a title, price, and a picture. If the user clicks on the advertisement, the images become an animation composed of all the existing pictures available on the website. Figure 10 illustrates the structure of the window.

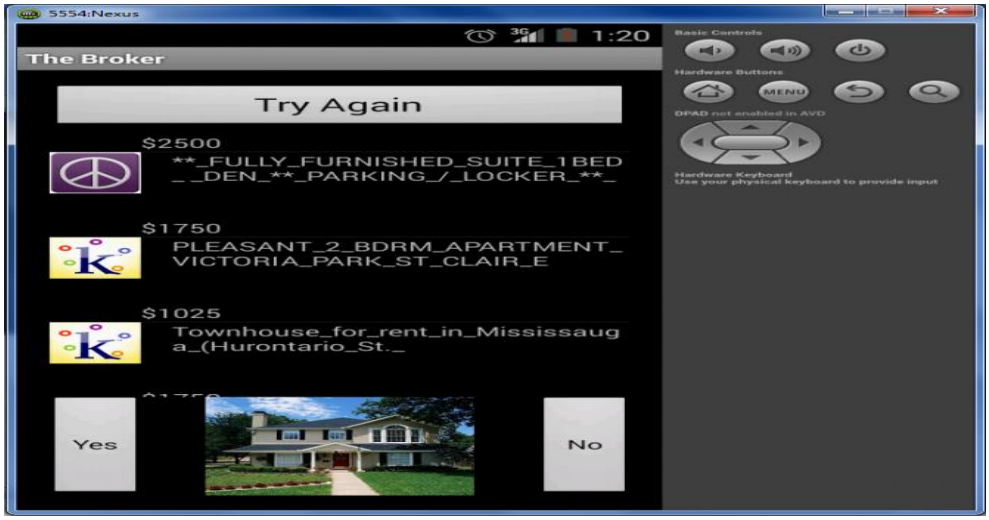


Figure 10. UI for a List of Results

For the yes/no buttons, If the user clicks yes, the Feedback Analyzer saves the advertisement and adds it to the user's profile found in the user database. Otherwise, that advertisement will be removed from the current list if the user is not interested in it. With the no option, a new window will pop up requiring the user to rate the discarder advertisement with a 5 star method based on 5 criteria including Price, Luxury, Neighborhood, Parking, and Advertisement Description. Figure 11 shows the Feedback Pop-Up Window.

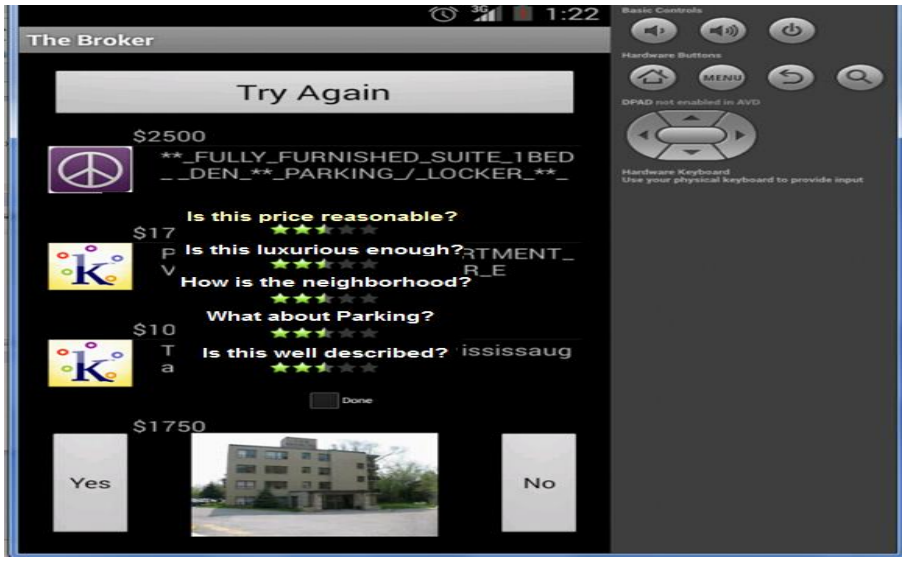


Figure 11. Feedback Pop-Up Window

The user rates the advertisement by clicking on any number of stars, ranging from 0 to 5, with 5 indicating that the advertisement was excellent with regard to that particular aspect. The ratings will then be converted to a value ranging from 0 to 100 and will be sent to two components, the *Advertisement Modifier* and the *User Modifier*.

3. Advertisement Modifier

The Advertisement Modifier is responsible for updating the central database's characteristics of the advertisement based on the user's five star ranking. After receiving feedback from the user, Meta Broker has to update its understanding of both that Advertisement and that user. The Advertisement Modifier has access to the vector matrix of that specific advertisement which will be shown as {Aj1… Aj5} and the confidence level of it as C(Aj). Also on the user side we have {Ui1… Ui5} and C(Ui) as the confidence level, plus the feedback values represented by {fij1 … fij5}. Equation 2 illustrates the updating process. The UAC represents the Updated Advertisement Characteristic, AC is the Advertisement Characteristics which we had from before, and the RAC is the newly generated characteristic based on the user's feedback and the user's information and called Recreated Advertisement Characteristics.

$$UAC = AC + RAC$$

Equation 2: Updating the Advertisements Characteristics

$$AC = \frac{C(A_J)}{c(A_J) + c(U_i)} \begin{bmatrix} A_{J1} \\ . \\ . \\ . \\ A_{J5} \end{bmatrix}$$

Equation 3: Portion of the Old Advertisement's Characteristics

$$RAC = \frac{C(U_i)}{c(A_J) + c(U_i)} \begin{bmatrix} N(U_{i1}, f_{ij1}) \\ . \\ . \\ . \\ N(U_{i5}, f_{ij5}) \end{bmatrix}$$

Equation 4: Portion of the Recreated Advertisement's Characteristics

The first multiplier in Equation 3 is the portion of the advertisement's confidence, $C(A_J)$, out of the sum of both the advertisement and the user's confidences, $c(A_J) + c(U_i)$ which will be multiplied by the previous characteristics which Meta Broker had from before. Also the first part of Equation 4 is another multiplier which shows the weight of the new advertisement's characteristics, $\frac{C(U_i)}{c(A_J)+c(U_i)}$, and another vector matrix calculated by using the user and feedback matrices. Equation 5 shows how each field of the new advertisement will be calculated.

$$N(U_{ik}, f_{ijk}) = U_{ik} + f_{ijk} - 50$$

Equation 5: Recreated Advertisements' Characteristics Based on the User Feedback Combination

Equation 4 where N represents the new advertisement characteristics based on a user's feedback, U_{ik} represents how much the user cares about k, parking, neighborhood, luxuriousness, price, and description, respectively, and f_{ijk} represents the feedback of user I on advertisement j in the field of k. For a better understanding of this formula, an example follows. If (U_{ik} =50), it means the user has average preferences and gives fairly accurate feedback about the reality of each advertisement. So if we replace U_{ik} with 50, the new user's characteristics will be exactlyf_{ijk}.

If the U_{ik} is above 50 but the feedback provided for the advertisement is as before, *i.e.*, f_{ijk} = 50, it means that a picky person has the same idea about an advertisement, thus the advertisement's value needs to be elevated. This is based on the assumption that a picky person would generally rate advertisements lower than the advertisement deserves because a

picky person would have high standers. If a user with a U_{ik} = 100 publishes feedback for an

advertisement around 0, Meta Broker's new evaluation of that advertisement becomes 50 because 50 is a normal value for an advertisement. This is based on the supposition that a super picky person hates something of average quality. Essentially, higher feedback and higher user characteristics increase the new vector. This formula shows the balance of exploitation and exploration. The AC relies on the information which was calculated up to this point and the RAC tries to change and update the advertisement's characteristics. The expanded edition of the Updating Advertisement Modifier Equation and the pseudo code of it can be found in Equation 6 and Figure 12 respectively.

$$\begin{bmatrix} A_{J1} \\ \cdot \\ \cdot \\ A_{J5} \end{bmatrix} = \frac{C(A_J)}{c(A_J) + c(U_i)} \begin{bmatrix} A_{J1} \\ \cdot \\ \cdot \\ A_{J5} \end{bmatrix} + \frac{C(U_i)}{c(A_J) + c(U_i)} \begin{bmatrix} N(U_{i1}, f_{ij1}) \\ \cdot \\ \cdot \\ N(U_{i5}, f_{ij5}) \end{bmatrix}$$

Equation 6: Expanded Edition of Updating Advertisements' Characteristics

```
Total Confidence= Confidence (A[j]) + Confidence (U[i])
AC= Confidence (A[j]) / Total Confidence
RAC= Confidence (U[i]) / Total Confidence
For k from 1 to 5{
        New Ad = U[i][k] + f[i][j][k] -50
        A[j][k] = AC * A[j][k] + RAC * New Ad
}
```

Figure 12. Pseudo Code of the Advertisement Modifier Functionality When User[i] Publishes Feedback[i][j] on Advertisement[j]

4. User Modifier

User modification is also required to make the users' characteristics. By using the information from the advertisement and the received feedback, the central database will be able to classify the users' preferences. The new behaviour of the user, as gathered from his/her feedback, will update the original perception of the user's preferences and enable better predictions of advertisements that will interest that user. The User Modifier has a similar formula to equation 2, the equation for updating the user's characteristics can be seen in Equation 7.

$$UUC = UC + RUC$$

Equation 7: Updating Users' Characteristics

Equation 7 is the engine of the user Modification. As before we have the previous information and we try to recalculate it again. UUC means the Updated User Characteristics, UC is the User Characteristics from before and RUC is the Recreated User Characteristics.

$$UC = \frac{C(U_i)}{c(A_J) + c(U_i)} \begin{bmatrix} U_{i1} \\ . \\ . \\ . \\ U_{i5} \end{bmatrix}$$

Equation 8: Portion of The Advertisement's Old Characteristics

$$RUC = \frac{C(A_J)}{c(A_J) + c(U_i)} \begin{bmatrix} N(A_{j1}, f_{ij1}) \\ . \\ . \\ . \\ N(A_{j5}, f_{ij5}) \end{bmatrix}$$

Equation 9: Portion of the Advertisement's Recreated Characteristics

Both UC and RUC are explained in Equation 8 and Equation 9 respectively. The User Modifier has a different N equation as follows, see Equation 10.

$$N(A_{jk}, f_{ijk}) = A_{jk} - f_{ijk} + 50$$

Equation 10: Recreated User Characteristics based on the Advertisement-Feedback Combination

By using the new values for the advertisement and feedback matrices the new values for that user can be calculated. Higher advertisements field and lower feedback increase the value of the new field. The reason being, is if an advertisement is of a place (office, house, apartment, etc…) that is of high quality based on the majority of users' feedback, and a user gave low feedback this indicated that the user who gave low feedback is picky. Also, if we suppose that the advertisement field is fixed and feedback varies, by publishing low feedback, $N(A_{jk}, f_{ijk})$ goes higher and shows that the user cannot be satisfied easily.

For instance if f_{ijk} = 50, A_{jk} represents the new Uik because if Aik is high Uik should be high

and vice versa. If we combine the N equations from both modifiers, we will have equation 11 which shows the N equations are compatible with what we are looking for.

$$N(A_{jk}, f_{ijk}) = (U_{ik} + f_{ijk} - 50) - f_{ijk} + 50 = U_{ik}$$

Equation 11. The Amortized Value of Recreated User Characteristics

A situation that shows the importance of the multipliers is when an immature user provides feedback on a mature advertisement or a mature user provides feedback on an immature advertisement. The multipliers prevent Meta Broker from discrediting valuable knowledge which came to the system over time, regarding the mature users and mature advertisements. The expanded edition of the Updating User Modifier Equation and the pseudo code of it can be found in Equation 12 and Figure 13 respectively.

$$\begin{bmatrix} U_{i1} \\ . \\ . \\ . \\ U_{i5} \end{bmatrix} = \frac{C(U_i)}{c(A_J) + c(U_i)} \begin{bmatrix} U_{i1} \\ . \\ . \\ . \\ U_{i5} \end{bmatrix} + \frac{C(A_J)}{c(A_J) + c(U_i)} \begin{bmatrix} N(A_{j1}, f_{ij1}) \\ . \\ . \\ . \\ N(A_{j5}, f_{ij5}) \end{bmatrix}$$

Equation 12: Expanded Edition of Updating the User Characteristics

```
Total Confidence= Confidence (A[j]) + Confidence (U[i])
UC= Confidence (U[i]) / Total Confidence
RUC= Confidence (A[j]) / Total Confidence
For k from 1 to 5{
        New User = A[j][k] - f[i][j][k] + 50
        U[i][k] = UC * U[i][k] + RUC * New User
}
```

Figure 13. Pseudo Code of the User Modifier's Functionality When User[i] Publishes Feedback[i][j] on Advertisement[j]

5. Discussion and Conclusion

In this paper, we designed and implemented a prototype of a Multi Agent recommender system that is empowered by collaborative filtering methods and resource-aware policies. The prototype involves an android mobile app that is capable of recommending the closest available real estate advertisements to the ideal list of advertisements for each user. The core contribution of this paper was two-fold. First, we presented a Multi Agent method to be able to unify all the online real estate stores and create an intermediate interface between the real estate customers and real estate online advertisements. As the second main contribution, we developed a collaborative filtering method to sort the fetched advertisements based on the users' characteristics, through modeling users, advertisements, and users' feedback. We are still in the process of developing this prototype to be used by other mobile platforms as well as to measure users' satisfactions with the prototype.

References

[1] M. Klusch and K. Sycara, "Brokering and matchmaking for coordination of agent societies: a survey", In Coordination of Internet agents, Springer-Verlag, (**2001**), ISBN:3-540-41613-7
[2] J. Fiaidhi, S. Mohammed and M. Hahn," Developing a Brokering Architecture for Multimedia Learning Objects on the Semantic Web, IJCSNS Int. Journal of Computer Science and Network Security, vol. 7, no.1, (**2007**) January.
[3] P. Karthikeyan, and E. Sathiyamoorthy, "A Survey on Applications of Mobile Agents in E-Business", International Journal of Scientific and Engineering Research, vol. 3, no. 3, **(2012)** March.
[4] L.-G. Li, "Study of e-commerce system based on mobile agent", 2010 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), pp. 705-708, 09 Jul - 11 Jul 2010, Chengdu, China.