APPLYING HYBRID RECOMMENDATION POLICIES THROUGH AGENT-INVOKED WEB SERVICES IN E-MARKETS

Alexis Lazanas, Nikos Karacapilidis and Vagelis Katsoulis
Industrial Management Lab, MEAD, University of Patras, 26504 Rio Patras, Greece

Keywords: Hybrid recommender Systems, web services, knowledge-based recommendation, intelligent agents.

Abstract: Diverse recommendation techniques have been already proposed and encapsulated into several e-business systems aiming to perform a more accurate evaluation of the existing alternatives and accordingly augment the assistance provided to the users involved. Extending previous work, this paper focuses on the development of an agent-invoked web service that will be responsible for the coordination of the system's recommendation module. The specific service will be invoked through a correspondent software agent that has been already implemented in our system's platform and will perform the tasks of recommendation policy synthesis, as well as the formulation of the appropriate knowledge rules.

1 INTRODUCTION

Recommendation systems accommodate users' preferences for the purpose of providing them with suggestions for purchasing or evaluating elements. Various techniques have been proposed for the creation of recommendations, such as collaborative filtering, knowledge-based and others. These techniques can be combined in hybrid recommendation systems in order to improve their performance.

This paper reports on research conducted in the area of hybrid recommendation systems and proposes a new approach that combines the knowledge-based with the collaborative filtering recommendation technique for the creation of recommendations in transactions taking place via the FTMarket platform. More specifically, this paper extends our previous work on the exploitation of software agent technology in transportation management (Lazanas *et al.*, 2005). We have addressed analysis, design and implementation issues raised during the development of an innovative agent-mediated electronic marketplace, which is able to efficiently handle transportation transactions of various types. Agents of the proposed system represent and act for any user involved in a transportation scenario, such as customers who look for efficient ways to ship their products and transport companies that may - fully or partially - carry out such requests, while they cooperate and get the related information in real-time mode. Our overall approach is based on flexible models that achieve efficient communication among all parties involved, coordinate the overall process, construct possible alternative solutions and perform the required decision-making. In addition, the supporting web-based system is able to handle the complexity that is inherent in such environments, which is mainly due to the frequent need of finding "modular" transportation solutions (Karacapilidis *et al.*, 2006).

This paper focuses on the features and functionalities of a new module integrated in the above system, namely the *recommendation module*, which aims at enhancing the quality of the associated decisions (Lazanas *et al.*, 2006). Recommender systems have been described as systems that produce individualized recommendations as output or have the effect of guiding the user in a personalized way, in environments where the amount of on-line information vastly outstrips any individual's capability to survey it (Burke, 2002). Alternative techniques have been also proposed in the literature in order to handle the above issues (O'Mahony *et al.*, 2002; Sarwar *et al.*, 2000). Having thoroughly considered their pros and cons, our approach follows a hybrid recommendation technique.

To consumers of Web Services, agents can form a powerful means of indirection by masking the

Web Service for purposes such as redirection, aggregation, integration, or even to provide recommendations, as in our case. From this perspective, the architecture proposed in this paper demonstrates a pattern of enabling Web Service clients to invoke application services exposed by agents. In this instance, the type of agent service made available is naturally restricted due to the limited expressivity or typical service invocation calls initiated by Web Service clients (Cho *et al*., 2002). The intrinsic implication here is that applications are now able to seamlessly bridge (in a bidirectional way) the agent and Web Service domains.

2 RELATED WORK

2.1 Hybrid Recommender Systems

Hybrid recommender systems combine two or more recommendation techniques to gain better performance with fewer of the drawbacks of any individual one. Most commonly, collaborative filtering is combined with some other technique in an attempt to avoid the "ramp-up" problem. Hybridization in recommender systems can alleviate some of the problems associated with collaborative filtering and other recommendation techniques.

Knowledge-based recommendation attempts to suggest objects based on inferences about a user's needs and preferences. In some sense, all recommendation techniques could be described as doing some kind of inference. Knowledge-based approaches are distinguished in that they have functional knowledge: they have knowledge about how a particular item meets a particular user need, and can therefore reason about the relationship between a need and a possible recommendation. The user profile can be any knowledge structure that supports this inference. In the simplest case, as in *Google*, it may simply be the query that the user has formulated. In others, it may be a more detailed representation of the user's needs.

Entree and several other recently developed systems employ techniques from case-based reasoning for knowledge-based recommendation. Schafer, Konstan and Riedl call knowledge-based recommendation the 'Editor's choice' method (Schafer *et al*., 2000). The knowledge used by a knowledge-based recommender can also take many forms. *Google* uses information about the links between web pages to infer popularity and authoritative value. *Entree* uses knowledge of cuisines to infer similarity between restaurants. Utility-based approaches calculate a utility value for objects to be recommended, and in principle, such calculations could be based on functional knowledge. However, existing systems do not use such inference, requiring users to do their own mapping between their needs and the features of products, either in the form of preference functions for each feature in the case of *Tete-á-Tete* or answers to a detailed questionnaire in the case of *PersonaLogic*.

2.2 Agents and Web Services

Typical agent architectures have many of the features of Web Services. Agent architectures provide "yellow" and "white pages" directories, where agents advertise their distinct functionalities and other agents search to locate the agents so they can request those functionalities (Klusch and Sycara, 2001). However, agents extend Web Services in several important ways:

- A Web Service knows only about itself, but not about its users, clients, or customers (Wang *et al.,* 2004). Agents are often self-aware at a meta-level, and through learning and model building gain awareness of other agents and their capabilities, as interactions among the agents occur..
- Web Services, unlike agents, are not designed to use and reconcile ontologies. If a service's client and provider happen to use different ontologies, the result of invoking the Web Service would be incomprehensible to the client.
- Agents are inherently communicative, whereas Web Services are passive until invoked. Agents can provide alerts and updates when new information becomes available. Current standards and protocols make no provision for even subscribing to a service to receive periodic updates.
- A Web Service, as currently defined and used, is not autonomous. Autonomy is a characteristic of agents, and it is also a characteristic of many Internet-based applications.
- Agents are cooperative, and by forming teams and coalitions can provide higher-level and more comprehensive services. Current standards for Web Services do not provide the means for composing functionalities.

3 OUR APPROACH

3.1 Recommendation's Module Overview

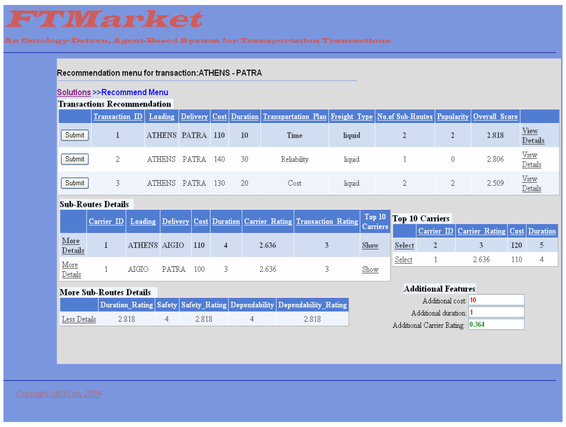
Our system retrieves all possible transportation routes that can be constructed for a given transaction request. These routes are presented to the user through an appropriate designed webpage. This webpage gives to the user the opportunity either to select one of the proposed routes (in this case he will be asked to complete the transaction), or to be redirected to an evaluation webpage where he can access the whole number of routes and the evaluation will be based on specific criteria. The evaluation takes place in a transactions and popularity rating web page and is based on various criteria, such as:

- Cost, Duration, Safety and Reliability;
- Average scores on the above carriers' elements that will be used;
- Average scores on the above sub-routes' elements that will be used;
- The number of times that a specific route has been selected by other customers;
- Customer's preferences;
- Number of sub-routes (assuming that transloadings are associated to a negative factor).

A final score is extracted for each route with the help of a mathematical formula which includes weight factors for the above criteria. At the beginning, the system renders the five best routes according to the final score without any further details. Though, the user is given the opportunity, if he wishes, to see the subroutes and the particular score details. The user can also compare a sub-route with an alternative which is provided by one of the ten best carriers. In this case, if a user chooses one of the alternatives, apart from the selected carriers the difference in the cost, time and score of the carrier is presented. When the user selects the desired itinerary, he is transferred to the transaction confirmation page.

Having executed the optimal routes retrieval algorithm (Lazanas *et al.*, 2006; Crauser *et al.*, 1998), the user is prompted by the system to press the *Recommendation* command button, in order to be transferred to the itineraries' evaluation webpage, where results are presented. Furthermore, this triggers the execution of the itineraries evaluation algorithm (executed in the website's server). The recommendation process is based on an evaluation algorithm, which applies to both the carrier and the itinerary participating in a proposed solution. More specifically, for every solution that the optimal route algorithm has retrieved for the requested transaction, we analyze the sub routes; for each of them, we calculate the average score that the carrier has received for its reliability during the transaction, as well as the average score that the specific route has for the duration of the transaction. During the calculation of the above average grades, the scores that each carrier or each route has received are multiplied by the user's reliability factor (through the *User_Reliability* table which is upgraded every time that a user evaluates a route). This is performed in order to add a level of significance into a reliable user's opinion compared with a less reliable one. This certain reliability is calculated by the number of times that a user has rated an itinerary and not by the fact that his evaluation was considered strict or not. Additional to the above duration reliability evaluation, there is a similar procedure for the safety and the general carrier's reliability during the transaction. So, taking into consideration the priorities (preferences) that the user has indicated (cost, safety, duration, reliability), we calculate the average of the carrier and the route under consideration (relying on the sum of the particular elements multiplied by gravity factors depending on the priorities of the user).

Both the average of the specific criteria (duration, reliability, safety, general reliability) and the general average are stored in the database, in the fields which correspond to the specific sub route. When this procedure is completed for all the sub-routes of the selected itinerary, the average of all the scores which were stored for each sub route is calculated in order to get the overall score for the carriers and the sub-routes, which will be used for the complete itinerary. The final score of the itinerary is the sum of the overall grade for the carriers and the overall grade for the subroutes, normalized by a percentage of the sum according to the overall cost and the number of transloadings for the specific itinerary. The overall and the particular grades are stored in tables corresponding to the current itinerary. When this procedure is performed for all itineraries, the user gets transferred to the popularity calculation webpage, where it is calculated whether each of the above itineraries has been completed and its correspondent frequency. Aim of this procedure is to observe if a specific itinerary is particularly popular for the selected transaction. The popularity of each route is an element which is presented later to the user, in order not to affect his final decision (see the web page illustrated in Figure 1).


FTMarket

An Ontology-Driven, Agent-Based System for Transportation Transactions

Recommendation menu for transaction:ATHENS - PATRA

Solutions >>Recommend Menu

Transactions Recommendation

	Transaction ID	Loading	Delivery	Cost	Duration	Transportation Plan	Freight Type	No.of Sub-Routes	Popularity	Overall Score	
Submit	1	ATHENS	PATRA	110	10	Time	liquid	2	2	2.818	View Details
Submit	2	ATHENS	PATRA	140	30	Reliability	liquid	1	0	2.806	View Details
Submit	3	ATHENS	PATRA	130	20	Cost	liquid	2	2	2.509	View Details

Sub-Routes Details

	Carrier ID	Loading	Delivery	Cost	Duration	Carrier Rating	Transaction Rating	Top 10 Carriers
More Details	1	ATHENS	AIGIO	110	4	2.636	3	Show
More Details	1	AIGIO	PATRA	100	3	2.636	3	Show

Top 10 Carriers

	Carrier ID	Carrier Rating	Cost	Duration
Select	2	3	120	5
Select	1	2.636	110	4

More Sub-Routes Details

	Duration_Rating	Safety	Safety_Rating	Dependability	Dependability_Rating
Less Details	2.818	4	2.818	4	2.818

Additional Features

Additional cost:	10
Additional duration:	1
Additional Carrier Rating:	0.364

Copyright: IMISLab, 2004

Figure 1: The recommendation module.

As shown, the five optimal itineraries for the transaction are presented, according to the final grade assigned to each one of them. Suppose that in this specific case the user is interested in a transaction from Athens to Patras. At the beginning, the page contains a single table with the retrieved itineraries and some of their basic features such as cost, duration, number of transloadings, popularity of the itinerary and its final grade. By pressing the *Submit* button, the user gets transferred directly to the transaction confirmation page. Apart from the initial table, the user is given the possibility to see particular details of the specific itinerary if he wishes. By pressing the *View Details* button, a second table appears which includes the subroutes' elements of the selected itinerary and some overall grades for each sub-route. By pressing the *More Details* button, the user is given the possibility to see analytical details for each sub-route, like grades of the duration's, safety and general reliability. The user is also given the ability to compare the candidate carrier of the sub-route he chose with the ten best transporters that exist for the particular sub-route by pressing the *Show* button of the *Top10 Carriers* field (a common *top-n* practice in collaborative filtering). Finally, the user is given the possibility, by pressing the *Select* button on the table which includes the ten best carriers, to compare the carrier of the selected sub-route, with one of the ten best carriers which are displayed beside. The comparison is being performed according to the cost and the time of the sub-route as criteria, as well as the overall score of the carriers. This comparison is presented under the name *Additional Features*. When the compared carrier excels the existent one (the one that the user has temporally selected), its elements are displayed green, while its elements are displayed red in the opposite case.

4 RECOMMENDATION POLICY

4.1 Mathematical Score Model

In this subsection, we present the mathematical model and the calculation procedure of the itineraries' overall score (*Overall_Score*) through the recommendation production from the system. The *Overall_Score* receives values in the range from 0 (minimum) to 5 (maximum) and is the criterion on which we base our suggestions to the users of our system, in order to assist them selecting the optimal itinerary. From the evaluation of the subroutes and the carriers involved, the *Overall_Score* takes an initial value. After that, in the initial value the cost of each route and the number of its transloadings is counted, thus obtaining the final value. The score model for assigning the *Overall_Score* value is:

$$\text{Overall\_Score} = \sum_{0}^{i} \begin{array}{l} \{[a*(average(ct*ur)+average(tt*ur))/2]+ \\ [b*(average(cs*ur)+average(ts*ur))/2]+ \\ [c*(average(cr*ur)+average(tr*ur))/2]/[(a+b+c)*i]\} \end{array} \qquad (1)$$

$$ct = Carrier\_Rating.Time$$
$$cs = Carrier\_Rating.Safety$$
$$cr = Carrier\_Rating.\mathrm{Re}iability$$
$$tt = Transaction\_Rating.Time$$
$$ts = Transaction\_Rating.Safety$$
$$tr = Transaction\_Rating.\mathrm{Re}liability$$
$$ur = User\_\mathrm{Re}liability$$
.

With the variable *average ()* we refer to the average of the registrations (they are included in the data base), of the variable which are in the parenthesis (*i* represents the number of subroutes of each itinerary). The variables *a, b, c, d* are weight factors and are related with the preferences of the user (cost, time, safety, reliability). Weight factor *d* is used later for the introduction of cost in the *Overall_Score*. The variables *a, b, c, d* assign their values by using a simple "case statement" which is presented in the pseudo-code below:

```
switch (Transportation_Plan)
{
case Time: a←2;
case Safety: b←2;
case    Reliability: c←2;
case Cost: d←2;
}
```

At this point, in the *Overall_Score* the normalized cost of the route and the number of transloadings are included. As normalized cost, we define the cost of the itinerary that we are examining, divided by the minimum cost of the itinerary. Furthermore, the number of transloadings is being penalized (a great number of transloadings could evoke damage in the product and increase the transaction's overall time).

5 AGENTS AND WEB SERVICES

5.1 Using Agents to Invoke Web Services

Web Services are often characterized as message-based. Interaction via message exchange means that instead of a client invoking functionality exposed as a Web Service, it sends a request to the Web Service to have the functionality invoked (Greenwood and Calisti, 2004). Or in other words, the thing that a Web Service exposes is the functionality of receiving a message. Instead of a *"GetRecommendationQuote"* (GRQ) port type, for example, a Web Service would expose a *"ReceiveMessage"* port type, to which messages requesting recommendation quotes are to be sent. This has the advantage of correctly describing the system's control boundaries. For example, if a system exposes a process GRQ service that implies that it is the customer who causes the GRQ to be processed. Of course, the system inserts some sort of control point into the code that gets invoked, while the system makes the decision of whether to really process the RFQ (e.g., by producing a recommendations' quote and sending it back), or whether to refuse the customer's request.

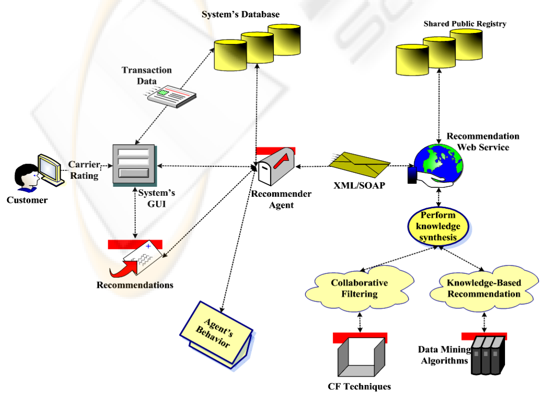


Figure 2: The recommendation module architecture.

We adopted a generic message interchange policy, which means that delivery of message content is independent of its format. Inputs to port types that can receive messages are sufficiently flexible that any content can be delivered in them. In effect, the *"ReceiveMessage"* port types should take arbitrary XML documents as input, regardless of their schema. In generic messaging, arbitrary message content may be exchanged by two interacting parties, even in cases where the recipient of a message is unable to recognize its meaning, make decisions about it, or even, perhaps, parse it. There are two fundamental reasons for this:

- Proper assignment of function. Constraining the set of messages that may be sent or received is like programming a telephone to send or receive only a fixed set of words (Kuno and Sahai, 2002). It is a basic misplacement of function. The messaging infrastructure should not to act as a supervisor defining what may and may not be said.
- Unexpected messages may turn out to be valuable, because they may contain clues as to how they should be handled. The simplest example of this is a message containing a non-standard abbreviation, which may be guessed at and, by a further exchange of messages, confirmed. Similarly, generic messaging provides a crucial feedback path by showing the recommender agent the way in which its correspondent web service is attempting to contact it.

5.2 Determining Recommendation Policy through Web Services

The overall architecture followed in our approach is illustrated in Figure 2. As shown, the recommendation tool is appropriately wrapped in order to describe the kind of service to be provided. To be easily located by users, such descriptions of services will be placed in a shared public registry. It is through this registry that users may look up for the services they need each time. The correspondent agent, which needs functions that can be provided by the specific Web Service, sends the appropriate request as an XML document in a SOAP envelope. Web Services may make requests of multiple services in parallel and wait for their responses. The set of services to be provided in the FTMarket platform will be increased and will constitute a services repository. It is noted that it is not necessary for all these services to be provided through a single server; multiple servers, located in distinct providers, may be used. Finally, it is made clear that the foreseen services, through the associated tools, communicate with a set of (local or remote) knowledge and model bases.

6 CONCLUSIONS

This paper presented our approach on the integration of hybrid recommendation techniques into an agent-based transportations transaction management platform. First of all, we proposed a hybrid approach that combines different recommendation techniques, in order to provide the user with more accurate suggestions. The overall process is coordinated by a recommender agent, who is responsible for invoking a correspondent Web Service which carries out multiple tasks, such as knowledge rules application, the appropriate recommendation technique selection and performs the knowledge synthesis through the exploitation of collaborative filtering techniques and data mining algorithms. The presence of the recommendation agent guarantees that the user will be provided with continuous recommendation and dynamic update of the recommendation.

REFERENCES

Burke, R. 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, 12, pp. 331-370.

Cho, Y., Kim J. and Kim, S. 2002. A personalized recommender system based on web usage mining and decision tree induction. *Expert Systems with Applications*, 23, pp. 329-342.

Crauser A., Mehlhorn K., Meyer U. and Sanders P., 1998. A Parallelization of Dijkstra's Shortest Path Algorithm. *In Proceedings of 23rd International Symposium, MFCS'98*, Brno, Czech Republic.

Greenwood D. and Calisti M., 2004. 'Engineering Web Service - Agent Integration'. In *IEEE Conference of Systems, Man and Cybernetics*, The Hague.

Karacapilidis N., Lazanas A., Megalokonomos G. and Moraitis P., 2006. On the Development of a Web-based System for Transportation Services. *Information Sciences, Vol.176 (13)*, pp. 1801-1828.

Klusch, M. and Sycara, K., 2001. Brokering and Matchmaking for Coordination of Agent Societies: A Survey. In A. Omicini et al. (eds.), Coordination of Internet Agents, Springer, 197-224.

Kuno, H. and Sahai, A., 2002. 'My agent wants to talk to your service: personalizing web services through agents'. *Proc. of the Ist International Workshop on Challenges in Open Agent Systems*, Bologna, Italy.

Lazanas A., Evangelou C., and Karacapilidis N. 2005: Ontology-Driven Decision Making in Transportation Transactions Management. In W. Abramowicz (ed*.), Proc. of the 8th International Conference on Business Information Systems (BIS 2005),* Poznan, Poland, April 20-22, pp. 228-241.

Lazanas A., Karacapilidis N. and Pirovolakis Y., 2006. Providing Recommendations in an Agent-Based Transportation Transactions Management Platform. In the *8th International Conference on Enterprise Information Systems*, Paphos, Cyprus.

O'Mahony M, Hurley N., and Silvestre C., 2002. Promoting Recommendations: An attack on Collaborative Filtering: *In: Proceedings of DEXA 2002 Conference, LNCS,* Vol. 2453, Springer-Verlag, Berlin, pp. 494-503.

Sarwar, B.M., Karypis, G., Konstan, J.A. and Riedl, J., 2000. Analysis of Recommender Algorithms for E-Commerce. In: Proceedings of the ACM Conference on e-Commerce, Minneapolis, MI, October 17-20, 2000, ACM Press, pp. 158-167.

Schafer, J.B., Konstan, J. and Riedl, J., 2000. Electronic Commerce Recommender Applications. *Journal of Data Mining and Knowledge Discovery,* 5 (1/2), pp. 115-152.

Wang, H., Huang, J., Qu, Y. and Xie, J. (2004) 'Web services: problems and future directions', Journal of Web Semantics, Vol. 1, pp.309–320.