The Architecture of a Shopping Agent

Assoc.prof. Mihaela OPREA, PhD.
Department of Informatics, University of Ploiesti

Software agents could help buyers and sellers to combat information overload and expedite specific stages of the online buying process. The paper presents a shopping agent architecture, SmartAgent, whose role is to assist users when doing electronic shopping, in the Internet.

Keywords*: agent-based e-commerce, intelligent agents, shopping agent.*

1 Introduction

The Internet and World Wide Web are becoming an important channel for retail commerce as well as business to business transactions. At present, electronic purchases are still largely non-automated. While information about different products and vendors is more easily accessible, and orders and payments can be made electronically, a human is still in the loop in all stages of the buying process, which adds to the transaction costs. In this context, the software agent technologies can be used to automate several of the most time consuming stages of the buying process. A software agent is personalized, continuously running and semi-autonomous. The literature reported several personal agents that assist user with information processing needs by generating, filtering, collecting, or transforming information [1], [2]. On the other hand, internet stores are providing services customized by the needs and interests of individual customers. Such services can be viewed as *seller's agents* with the purpose to push merchandise and/or services on to the users. Therefore, there is a growing need for deploying *shopping agents* or *buyer's agents* whose goal is to best serve the user's interests and to make more informed purchasing decisions. The paper presents the architecture of a shopping agent that is under development at University of Ploiesti [3].

The paper is organized as follows. Section 2 makes a brief description of agent-based e-commerce. Section 3 describes the shopping agent model and section 4 concludes the paper.

2. Agent-based electronic commerce

With the rapid explosion of the Internet and the World Wide Web a different class of agents came to the fore: agents that can gather and collate information on behalf of their users. The Internet produced an abundance of information which is overwhelming. Typical search engines return too many links in response to queries. This limits their usefulness as it becomes difficult to differentiate between the relevant and irrelevant data. Some solutions are given by personalized web-page recommender systems, collaborative filtering mechanisms, and software agents. In this context, the electronic commerce will become in the near future an agent-based electronic commerce.

The agent-based electronic commerce is defined as an electronic commerce in which each participant has an agent that acts in the interest of his owner. In figure 1 it is presented an example of an Internet environment for e-commerce (FIPA-compliant). In this agent-based environment there are seller agents (SA_j), buyer agents (BA_i), agent resource brokers (ARB), directory facilitators (DFs), agent comunication channels (ACCs) and so on. Therefore, the main types of agents that are involved in e-markets are buyers, sellers, mediators, brokers etc. Shopping through the Internet it's lower cost, and enable increased choice and convinience offered to the customer. Agent technology has the potential to make online shopping more than just a set of web-based front-ends to mail order catalogues. The goal is smarter shopping, whereby the best deal for the customer can be located quickly at the lowest cost, and with a minimal effort from the user. Agent-based e-commerce is also benefit vendors by lowering their costs and increasing their customer base. Many companies already take advantage of the lower costs of doing business online.

A buying agent automatically collects information on vendors and products that may fit the needs of the user, evaluate different offerings, make decision on which merchants and products to pursue, negotiate the terms of transactions with these merchants and finally, place orders and make automated payments.

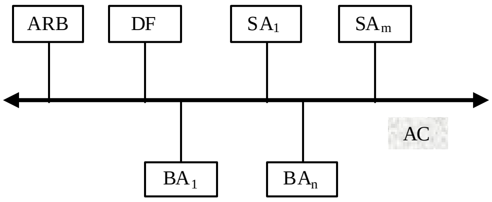

Fig.1. The Internet e-Commerce environment (FIPA compliant)

Several models have been proposed as attempts to capture buying behavior: the Nicosia model, the Howard-Sheth model, the Engle-Blackwell model, the Bettman information processing model, and the Andreasen model [2]. All these models share a list of six fundamental stages of the buying process.

1. need identification
2. product brokering
3. merchant brokering
4. negotiation
5. purchase and delivery
6. product service and evaluation

These stages often overlap and migration from one to another can be non-linear and iterative.

Stage 1, *need identification*, characterizes the buyer becoming aware of some unmet need. The software agent technology offers the so called *monitoring agents* (*monitors*), continuously running programs which monitor a set of sensors or data streams and take actions when certain pre-specified conditions apply. Such agents are used in the stock market as well as on different e-commerce sites. An example is given by Amazon.com which offers to its customers a “notification agent” called “Eyes” which monitors the catalog of books for sale and notifies the customer when certain events (that may be of interest to the customer) occur.

The second stage, *product brokering*, comprises the retrieval of information to help determine what to buy. The buyer has to determine what to buy through a crtical evaluation of retrieved product information. At this stage different techniques could be used (e.g. collaborative filtering, constraint-based techniques, rule-based techniques, data-mining), to discover patterns in customer purchasing behavior.

Merchant brokering combines the resulted set of products from stage 2 with merchant-specific information to help determine from where to buy. It is made an evaluation of merchant alternatives based on buyer’s criteria (e.g. price, warranty, delivery time etc). The first shopping agent for online price comparisons is BargainFinder from Andersen Consulting [4]. Another shopping agent, advanced than BargainFinder is Jango [5].

The stage 4, *negotiation*, identify how to settle on the terms of the transaction. In certain markets (such as automobile, fine arts, stocks) the negotiation of price or other aspects of the deal are integral of the buying process [6]. Two popular web sites, Onsale and eBay, that sell refurbished and second-hand products are using a choice of auction protocols. In such situations, the software agents could assist the customer in negotiating the terms of a transaction. Several agents that support negotiation were reported in the literature (e.g. AuctionBot and Kasbah).

The fifth stage, *purchase and delivery*, can signal the end of the negotiation stage or can occur sometime afterwards. The available payment or delivery options can influence product and merchant brokering.

The last stage, *product service and evaluation*, is a post-purchase stage and involves product service, customer service and an evaluation of the satisfaction of the overall buying experience and decision.

In an agent-based e-commerce, a software agent can help the user with all aspects of online shopping:

1. helps the user to decide what product to buy (e.g. by listing what products of a certain type are available);
2. finds specifications and review of the products recommended;
3. makes recommendations;
4. comparison shopping to find the best price for the desired product;
5. monitoring *what's new* lists and other sources to discover new relevant online information sources;
6. watching for special offers and discounts.

In agent-based e-commerce, the agents need to have the capability of learning about customer's preferences. At present, there are some learning agents in the Internet. For example, Firefly [7] helps the user find music that they are likely to enjoy, and uses information gathered from others, similar to the users' in tastes and opinions, to suggest new music.

A major value of employing software agents with intranet, Internet, and extranet applications is that they are able to assist in locating and filtering all the data. They save time by making decisions about what is relevant to the user.

The remarkable growth in agent-oriented internet-based applications is encouraging. Still, most of these applications appear to open up new possibilities or choices for the user without providing much guidance or help about how best to use this additional information. Therefore, considerable research is directed to the area of comparison shopping agents [8], [9].

3. The shopping agent architecture

The shopping agent will serve the interest of the user by understanding the user's goals and recommending products/services or suggesting modification to user queries or requirements that will be more likely to produce results at a higher level of user satisfaction. The consumer's initial choice or preference can be modified in the light of new information from rapidly changing marketplace (e.g. the latest options, deals, package offerings etc). The shopping agent must be able to track changing market conditions and to inform the user about interactions between stated constraints in queries and the prevailing market.

When designing an intelligent web agent we need to address several questions:

1) To what extent can intelligent agents understand information published on the Web sites?
2) Is the agent's understanding sufficient to provide genuinely useful assistance to users?
3) Is site-specific hand-coding necessary, or can the agent automatically extract information from unfamiliar Web sites?
4) What aspects of the Web facilitate this ability?

These four questions are related to four essential characteristics of a shopping agent: *the ability*, *the utility*, *the scalability* and *the environmental constraint*. Unfortunately, the Web is less agent-friendly than we might hope. Although Web pages are written in HTML, this language only defines how information is to be displayed, not what it means. A solution is XML which has becoming more widely used. Also, research efforts are directed to the development of the second generation of Web, the Semantic Web.

We have designed a shopping agent model, SmartAgent, whose architecture is described in figure 2. The shopper interacts with the user through a graphical user interface (GUI) based on the domain description. The agent is able to parse product descriptions and to identify several product attributes, including price and warranty. It achieves this performance without sofisticated natural language processing, and requires minimal knowledge about the domain of the products. The agent can extract information from online vendors by using an heuristic search, pattern matching and an inductive learning technique.

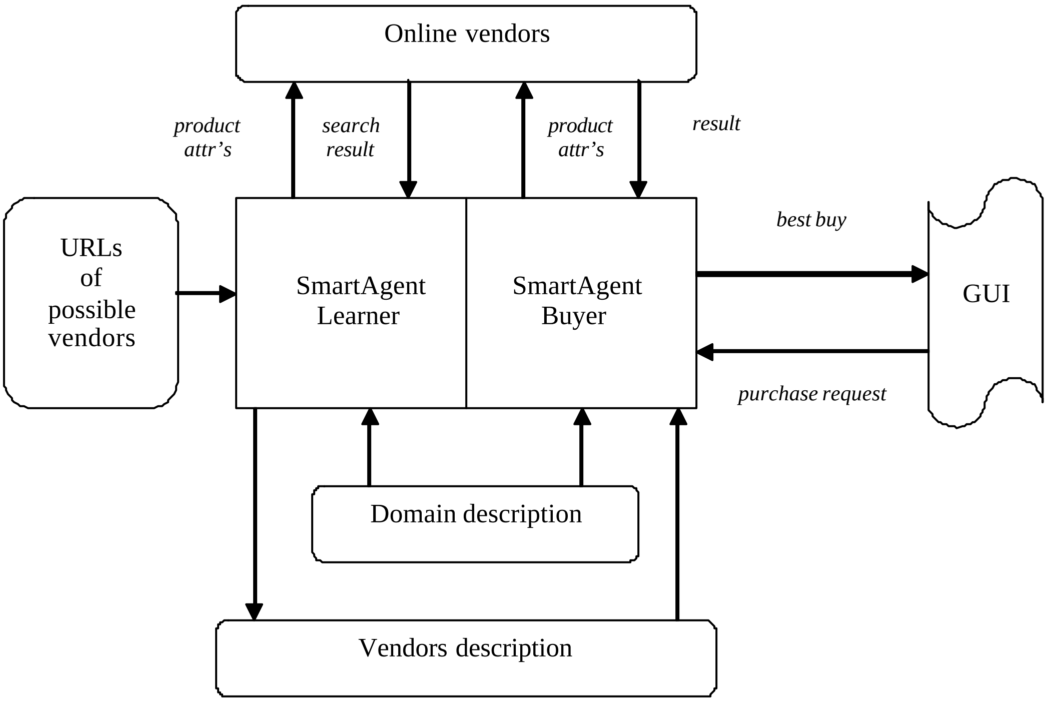

Fig.2. The architecture of the shopping agent *SmartAgent*

The shopping agent may receive proposals from multiple sales agents. Each proposal defines a complete product offering including a product configuration, price, warranty, and the merchant's value-added services. The shopping agent evaluates and orders these proposals based on how they satisfy its owner's preferences (expressed as multi-attribute utilities).

The shopping agent has two main components: the learner and the buyer. In the learning stage an offline learner creates a vendor description for each merchant. In the buying stage a real-time shopper uses the vendors description to help the user decide which store offers the best price for a given product. Given the home pages of several online stores, SmartAgent autonomously learns how to shop at those vendors. After learning, the agent is able to speedily visit over a dozen software vendors, extract information and summarize the results for the user. The agent uses knowledge about different product domains. SmartAgent does an heuristic search and uses pattern matching and an inductive learning technique [10].

The problem of extracting the product description from the site

Figure 3 describes the information extraction learning problem. Starting from an incomplete domain model and the URLs of vendor's pages, the goal is to determine a procedure which accesses the vendor site to look for a given product and to return a set of product descriptions.

The architecture of SmartAgent is useful for stores that provide a searchable index. Some heuristics help the learner to discard forms that are clearly not searchable indices, e.g. forms that prompts the user for name, address and phone number.

Input data:

1. Incomplete domain model:
   - example products: $P_1, P_2, \ldots, P_n$
   - attributes of the products
2. The URL for the home page of a vendor

Goal:

Determine a procedure which accesses the vendor site to look for a given product and returns a set of strings, each corresponding to a product description returned by the vendor.

Fig.3. The information extraction learning problem

Example of a vendor description:

- the URL of a page containing a form for a searchable index;
- a function mapping product attributes to fields of that form;
- functions for extracting product data from pages returned by the index:
  - a function that extract a set of individual product descriptions from the remaining text on a successful page;
  - a function that strips header and tailer information from successful pages;
  - a function that recognizes failure pages ('Product not found');

A page tipically contains not only one or more product descriptions, but also information about the store itself, meta-information about the shopping process, headings, subheadings, links to related sites, and advertisements. Therefore, the problem is difficult and moreover, the product name often appears in other places on the page, not in product descriptions. The format of product descriptions varied widely and no intuitive rule can work robustly across different products and different vendors. Still, some regularities can be found and used by the learning approach.

Examples of regularities are:

1) the vertical separation regularity – merchants use whitespace to facilitate customer comprehension of their catalogs;
2) the uniformity regularity – vendors attempt to create a sense of identity by using a uniform look and feel.

The learning algorithm search through the space of possible abstract formats and pick the best one. The problem of learning a vendor description has three steps:

(1) identifying an appropriate search form;
(2) determining how to fill in the form;
(3) discerning the format of product descriptions in pages returned from the form.

Accordingly, the SmartAgent learner will make three decisions which are strongly interdependent. First, the learner will find a set of candidate forms (possibilities for the first decision). For each form F it will compute an estimate of how successful the comparison-shopping phase should be if form F is chosen by the learner. In order to do the estimation, the learner will determine how to fill in the form and then will make several *test queries* using the form to search several popular products. The results of the test queries are used as training examples (to induce the format of product descriptions) and to compute the learner's success in finding the popular products. Finally, the learner will pick the form with the best estimate and records a vendor description.

The SmartAgent learner first queries each form with several "dummy" products such as "qrscd dummynosuchproduct" to determine what a "Product Not Found" result page looks like for that form. The learner builds a generalized failure template based on these queries. Next, the learner queries the form with several popular products given in the domain description. It matches each result page for one of these products against the failure template. The page that does not match the template is assumed to represent a successful search. If the majority of the test queries are failures the learner assumes that this is not the appropriate search form to use for the vendor. Otherwise, the learner records generalized templates for the header and tailer of success pages, by abstracting out references to product attributes and then finding the longest matching prefixes and suffixes of the success pages obtained from the test queries. After that, the learner uses the bodies of these pages from successful searches as training examples from which will induce the format of product descriptions in the resulted pages for this form. Each such page contains one or more product descriptions (including particular products searched) that matched the query parameters. Extracting these product descriptions is difficult because their format varies widely across vendors. From the training examples, the SmartAgent learner will induce the product descriptions by using the ILA algorithm [10]. The algorithm breaks the body of each resulted page into logical lines representing vertical-space-delimited text, and then considers abstract formats (i.e. strings of XML/HTML tags and/or keywords – e.g. "<li>*text*<a>*text* </a>*text*.") that correspond to at least one of the logical lines in one of the resulted pages. The learner uses a heuristic ranking process to choose which format is most likely to be the one the store uses for product descriptions. The ranking function is the sum of the number of lines of that format in which some text was found, plus the number in which a price was found, plus the number in which one or more of the required attributes were found. The SmartAgent learner repeats the procedure for each candidate form. Finally, the agent has to decide which is the best choice, i.e. which is the best form to use for a comparison shopping. The choice is based on making an estimation for each form of how successful the comparison-shopping stage would be if that form was chosen by the learner.

The operation of the shopping agent is simple. Once it has received a request from the user via the GUI, it goes in parallel to each online vendor's searchable index and fills out and submits the forms. For each resulting page not matching the vendor's failure template, it strips off the header and tailer, and looks in the remaining XML/HTML code for any results (any logical lines matching the learned product description format. After that, it sorts the results by ascending order of price and generates a summary for the user.

4. Conclusion

With the rapid explosion of the electronic commerce stores, the consumer may be overwhelmed by the volume and diversity of information available on the net and may not have time to search the available information to make a judicious choice. Therefore, the solution offered by software agents could be adopted. The software agent technologies help to manage ambiguous content, personalized preferences, complex goals, changing environments and disconnected parties [11]. The first generation of agent-based electronic commerce systems are creating new markets and begin to reduce transaction costs in a variety of business processes (e.g. industries such as gas, electricity, books etc). In this paper we have presented the architecture of a shopping agent, SmartAgent, that allow the improvement of the buying process by trying to make the best deal.

References

[1] Jennings, N.R., and M. Wooldridge, Software Agents, *IEE Review*, pp. 17-20, January 1996.

[2] Nwana, H.S., Software Agents: An Overview, *The Knowledge Engineering Review*, vol. 11(3), pp. 205-244, 1996.

[3] Oprea, M., SmartAgent: A Shopping Agent Model, *research report*, University of Ploiesti, 2002, (under publication in an international journal).

[4] BargainFinder URL: bf.cstar.ac.com/bf

[5] Jango URL: www.jango.com/

[6] Oprea, M., Adaptability and Embodiment in Agent-based E-Commerce Negotiation, *Proceedings of the International Workshop Adaptability and Embodiment Using Multi-Agent Systems AEMAS'01*, July 2001, Prague, Czech Republic, pp. 257-265.

[7] Firefly URL: <http://www.firefly.com/>

[8] Maes, P., R.H. Guttman, and A.G. Moukas, Agents that buy and sell, *Communications of the ACM*, 42(3), March, 1999.

[9] Etzioni, O. and D. Weld, Softbot-based interface to the internet, *Communications of the ACM*, 37(7), pp. 72-76, July 1994.

[10] Oprea, M., Rule Generation Versus Decision Tree Induction, *Proceedings of the 20th International Conference Applied Informatics AI'02*, Feb. 2002, Innsbruck, Austria, ACTA Press, pp. 395-398.

[11] Schlimmer, J., and L. Hermens, Software agents: Completing patterns and constructing user interfaces, *Journal of Artificial Intelligence Research*, 1993, pp. 61-89.