Pattie Maes, Robert H. Guttman, and
Alexandros G. Moukas

Agents That Buy and Sell

Shoppers and sellers alike dispatch them into the digital bazaar to autonomously represent their best interests.

Popular software agents were first used several years ago to filter information, match people with similar interests, and automate repetitive behavior. More recently, agents have been applied to e-commerce, promising a revolution in the way we conduct transactions—business-to-business, business-to-consumer, and consumer-to-consumer. The Internet and World-Wide Web represent an increasingly important channel for retail commerce as well as business-to-business transactions. Recent studies by Forrester Research, International Data Corp., and Nielsen Media Research, have found that the numbers of people buying, selling, and performing transactions on the Web are increasing at a phenomenal pace.

However, the potential of the Internet for transforming commerce is largely unrealized. Electronic purchases are still largely nonautomated. While information about products and vendors is more easily accessible, and orders and payments are dealt with electronically, humans are still in the loop in all stages of the buying process, adding to transaction costs. A human buyer is still responsible for collecting and interpreting information on merchants and products, making decisions about merchants and products, and ultimately entering purchase and payment information.

Software agent technologies can be used to automate several of the most time-consuming stages of the buying process. Unlike so-called traditional software, software agents are personalized, continuously running, and semiautonomous [1]. These qualities help optimize the whole buying experience, revolutionizing commerce as we know it [2]. For example, a company that needs to

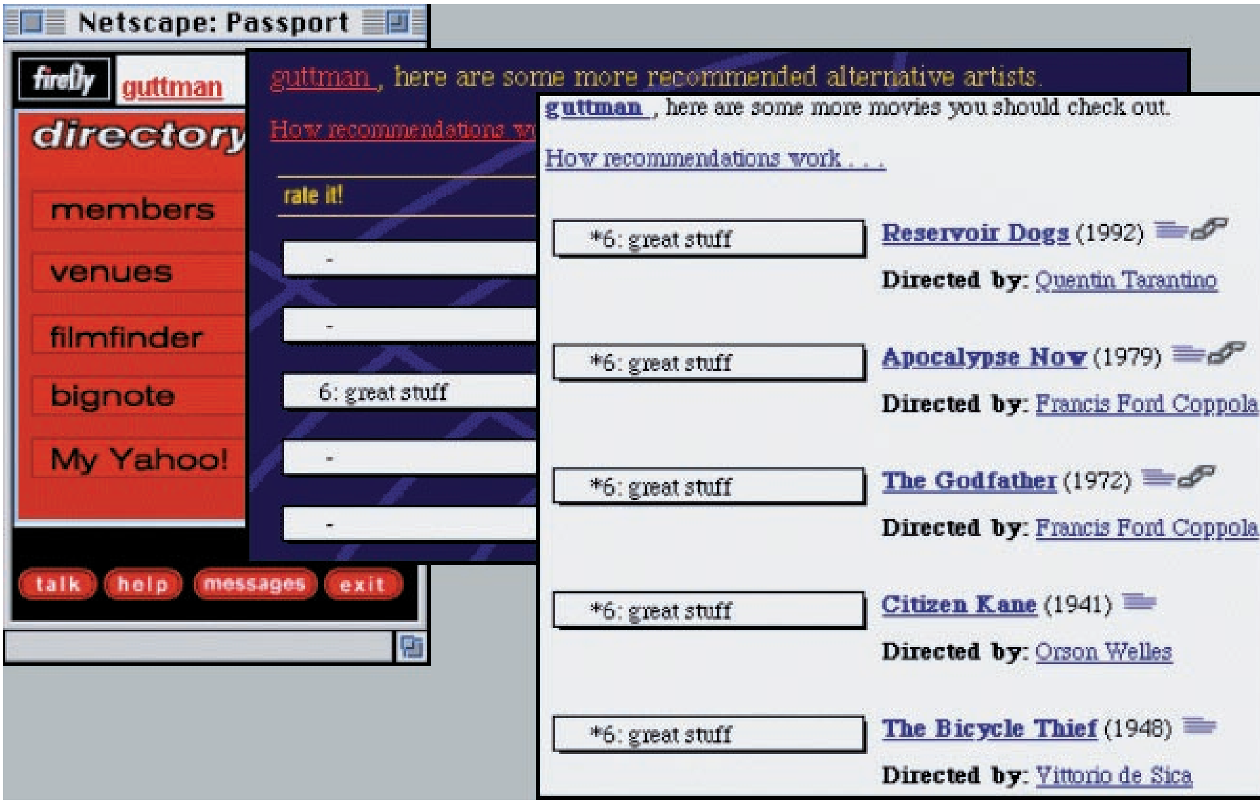

Figure 1. Firefly recommends products based on opinions of like-minded people.

order paper supplies could enlist agents to monitor the quantity and usage patterns of paper within the company, launching buying agents when supplies are low. Buying agents automatically collect information on vendors and products that may fit the needs of the company, evaluate the various offerings, make a decision on which merchants and products to investigate, negotiate the terms of transactions with these merchants, and finally place orders and make automated payments.

As Mediators in E-commerce

It is useful to use a common framework as a context for exploring the roles of agents as mediators in e-commerce. The model we use here stems from consumer buying behavior (CBB) research and includes the actions and decisions involved in buying and using goods and services. Although CBB covers many areas, it is important to recognize its limitations. For example, CBB research focuses primarily on retail markets (although most of its concepts pertain to business-to-business and consumer-to-consumer markets). Even within retail, not all shopping behaviors are captured (such as impulse purchasing). Moreover, e-commerce covers a broad range of issues, some beyond the scope of a CBB model (such as back-office management, supply chain management, and other merchant issues).

Several descriptive theories and models seek to capture buying behavior, including the Nicosia model, the Howard-Sheth model, the Engel-Blackwell model, the Bettman information-processing model, and the Andreasen model. All share six similar fundamental stages of the buying process:

	Persona Logic	Firefly	Bargain Finder	Jango	Kasbah	Auction Bot	T@T
Need identification							
Product brokering	✔	✔					✔
Merchant brokering			✔	✔			✔
Negotiation					✔	✔	✔
Payment and delivery							
Service and evaluation							

Table 1. Online shopping framework with representative examples of agent mediation

- Need identification. Characterizes the buyer becoming aware of some unmet need. The buyer can be motivated through product information.
- Product brokering. Includes retrieval of information to help determine what to buy. Information retrieval includes an evaluation of product alternatives based on buyer-provided criteria. The result is the "consideration set" of products.
- Merchant brokering. Combines the "consideration set" from the previous stage with merchant-specific information to help determine who to buy from. This stage also includes evaluation of merchant alternatives based on buyer-provided criteria (such as price, warranty, availability, delivery time, and reputation).
- Negotiation. Considers how to settle on the terms of a transaction. Negotiation varies in duration and complexity depending on the market. In traditional retail markets, prices and other aspects of a transaction are often fixed, leaving no room for negotiation. In other markets (such as stocks, automobiles, and fine art), the negotiation of price and other aspects of the deal are integral to the buying process.
- Purchase and delivery. Signals either termination of the negotiation stage or occurs sometime afterward (in either order). In some cases, the available payment (such as cash only) or delivery options can influence product and merchant brokering.
- Product service and evaluation. Involves post-purchase product service, customer service, and evaluation of the satisfaction of the overall buying experience and decision.

As with most models, these stages represent an approximation and simplification of complex behaviors. They often overlap, and migration from one to another can be nonlinear and iterative.

They also help identify where agent technologies apply to the shopping experience, allowing us to more formally categorize existing agent-mediated e-commerce systems [3]. We can, for example, identify the roles of agents as mediators in e-commerce. The personalized, continuously running, autonomous nature of agents makes them well-suited for mediating consumer behaviors involving information filtering and retrieval, personalized evaluations, complex coordination, and time-based interactions. These roles correspond most notably to need identification, product brokering, merchant brokering, and negotiation in the buying behavior model. Table 1 lists the six buying behavior stages and which of them are supported by representative agent systems.

To some extent, agent technology can be helpful in automating or assisting the buyer with the need-identification stage. Specifically, agents can help in purchases that are repetitive (such as supplies) or predictable (such as habits). One of the oldest and simplest examples of software agents are so-called monitors, or continuously running programs that monitor a set of sensors or data streams and take actions when certain prespecified conditions apply. Examples are abundant in the stock market, as well as at e-commerce sites. For example, Amazon.com offers its customers a "notification agent" called "Eyes" that monitors its catalog of books and notifies the customer when certain events occur that may be of interest, like when a new book by a particular author becomes available or when a new book in a certain category becomes available.

At the speed of bits, agents will strategically form and reform coalitions to bid on contracts and leverage economies of scale.

After identifying a need to buy something (possibly with the assistance of a monitor agent), the buyer has to determine what to buy through a critical evaluation of retrieved product information. Table 1 lists several agent systems—PersonaLogic, Firefly, and Tête-à-Tête—that lower consumers' search costs when deciding which products best meet their needs. PersonaLogic (www.personalogic.com) enables consumers to narrow the list of products that best meet their needs by helping them define a number of product features. The system filters out unwanted products within a given domain after a shopper specifies constraints on product features. A constraint-satisfaction engine then returns a list of products that satisfy all of the shopper's hard constraints in order of how well they satisfy the shopper's soft constraints.

Tête-à-Tête uses comparable techniques to recommend complex products based on multiattribute utility theory. However, unlike PersonaLogic, Tête-à-Tête also assists buyers and sellers in the merchant-brokering and negotiation stages.

Like PersonaLogic, Firefly (www.firefly.com) and other systems based on collaborative filtering [4] help consumers find products (see Figure 1). However, instead of filtering products based on features, Firefly recommends products through an automated "word-of-mouth" recommendation mechanism called "collaborative filtering." The system first compares a shopper's product ratings with those of other shoppers. After identifying the shopper's "nearest neighbors," or users with similar taste, the system recommends the products the neighbors rated highly but which the shopper may not yet have rated, possi-

Automated Negotiation

The best terms for all concerned

Tuomas Sandholm

Negotiation is a key component of e-commerce. In automated negotiation, computational agents find and prepare contracts on behalf of the real-world parties they represent. This automation saves human negotiation time, and computational agents are often better at finding deals in combinatorially and strategically complex settings. Such benefits need not involve a cost to the other contracted parties, because negotiation is more than just constant-sum bargaining over price. When different users have different preferences, automated negotiation can rapidly find solutions that improve the utility for all parties.

The most promising application areas for automated negotiation include retail e-commerce, electricity markets, bandwidth allocation, manufacturing planning and scheduling in subcontracting networks, distributed vehicle routing among independent dispatch centers, and electronic trading of financial instruments.

Task reallocation among agents is a key type of negotiation. Reallocation is beneficial when tasks are not initially allocated to the agents that handle them least expensively. That's why I developed a marginal cost-based method for automated task reallocation negotiation that helps reallocate all types of items (beside tasks), such as financial instruments and hours of electricity.

In this method, each agent willingly takes on a task from another agent as long as it is paid more by the other agent than it costs the first agent to handle the task itself. Similarly, an agent willingly gives out a task to another agent as long as it does not have to pay the other agent more than it would cost the first agent to handle the task itself. Using this method in 1990, I built the TRAnsportation COoperation NET (TRACONET) system for automated delivery of task reallocation among freight companies. To my knowledge, this was the first distributed implementation of automated negotiation among self-interested agents. Each software agent, executing in its own Unix process, represents a single company in the negotiation. An agent can take on delivery tasks from some agents and give out tasks to others. Each agent can also recontract-out tasks it had previously contracted-in.

However, negotiation can get stuck in a local opti-

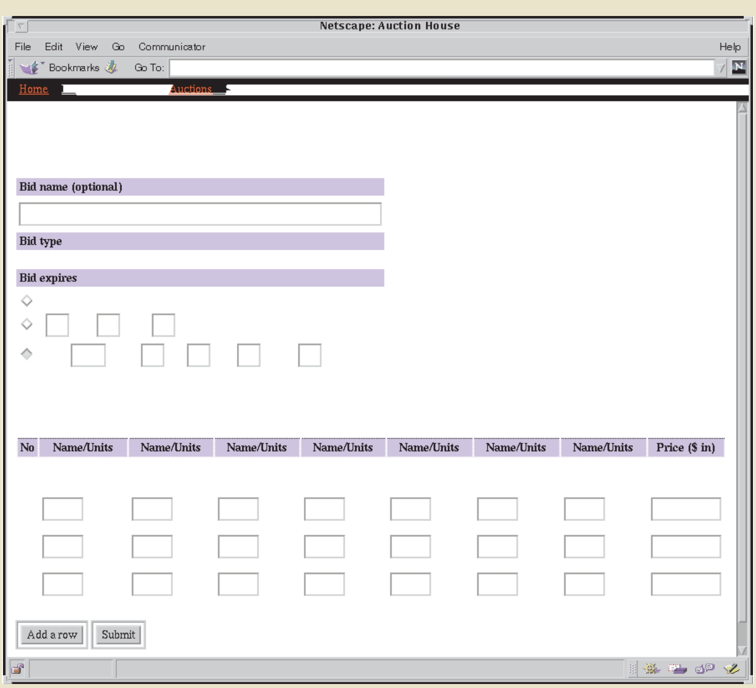

Figure. A combinatorial bid in eMediator. Here, a refinery operator needs three consecutive hours of electricity, preferring to start at 8 a.m., because running the plant then requires the least electricity; starting at 6 a.m. or 7 a.m. would also be acceptable.

bly resulting in serendipitous finds. Firefly uses the opinions of like-minded people to offer recommendations of such commodity products as music and books, as well as more-difficult-to-characterize products, such as Web pages and restaurants.

In addition to constraint-based and collaborative filtering techniques, two other techniques are widely used to implement product brokering and product-recommendation agents. A large set of sites uses simple rule-based techniques, such as those provided by Broadvision, Inc., to personalize product offerings for individual customers. A few sites experiment with data-mining techniques to discover patterns in customer purchasing behavior, exploiting these patterns to help customers find other products that meet their needs.

Whereas the product brokering stage compares product alternatives, the merchant-brokering stage compares merchant alternatives. Andersen Consulting's BargainFinder (bf.cstar.ac. com/bf) is the first

mum where task allocation is suboptimal but where no original (O) contract (transferring one task and a payment) is profitable. To escape such local optima, I introduced into automated negotiation several new contract types: cluster (C) for exchanging multiple tasks for a payment; swap (S) for swapping a task for another task and a sidepayment; multiagent (M) when more than two parties are involved in the same contract; and OCSM for combining these ideas into an atomic contract, guaranteeing that globally optimal allocation is achieved through a finite number of contracts.

While my original implementations of such contracting were based on agent-to-agent negotiation, my colleagues and I recently built an auction server implementing a centrally mediated variant. Agents send bids and tasks for combinations of items to the server (see the Figure). Unlike traditional auctions, these combinatorial auctions allow users to express interrelated valuations of the items—a particularly important function in illiquid, highly volatile, and noncommoditized markets where a participant is unsure whether he or she can buy the items in a desired bundle one at a time.

This auction server is part of eMediator, our e-commerce server, which also provides services other than auctions. To my knowledge, eMediator is the first Internet auction to support combinatorial bidding, bidding via price-quantity graphs, and mobile agents. eMediator determines the winners of the combinatorial auction, identifying profitable contracts for the participating agents. But because optimal winner determination is computationally so complex, we added a highly optimized search-based matching algorithm to work through the problem.

Price-quantity graphs allow users to express continuous preferences. Mobile agents allow users to have their agents participate in the auction while these users are disconnected from the Internet. Mobile agents execute on the agent dock on or near the auction host to reduce network latency—a key issue in time-critical bidding. eMediator uses Mitsubishi's Concordia agent dock to give mobile agents a safe execution platform from which to bid, set up auctions, travel to other auction sites, and observe the activity at various auctions (see Koblick's "Concordia" in this issue).

eMediator also provides an HTML interface through which users specify what they want their agents to do, automatically generating Java code for the corresponding mobile agents before launching them.

Contracts are usually binding in automated negotiation systems for self-interested agents. Such contracts do not allow agents to undo old commitments to accommodate new events, such as tasks that turn out to be more costly than anticipated or the arrival of more lucrative offers later on. To circumvent these problems, I devised a leveled commitment contracting protocol that allows agents to accommodate future events by offering the option of unilateral decommit. The commitment level is chosen by assigning a decommitment penalty to each contract party; to be freed from the contract, the agent pays the penalty to the other parties.

One concern is that rational agents are reluctant to decommit; there is a chance another party will also decommit, in which case the first agent is freed from the contract, does not have to pay a penalty, and collects a penalty from the breacher. However, despite insincere decommitting, the leveled commitment feature increases each contract party's expected payoff, enabling contracts in settings where no full commitment contract is beneficial to all parties. C

TUOMAS SANDHOLM (sandholm@cs.wustl.edu) is an assistant professor of computer science at Washington University in St. Louis where he heads the Multiagent Systems Research Group.

Supported by NSF CAREER Award IRI-9703122, NSF grant IRI-9610122, and NSF grant IIS-9800994.

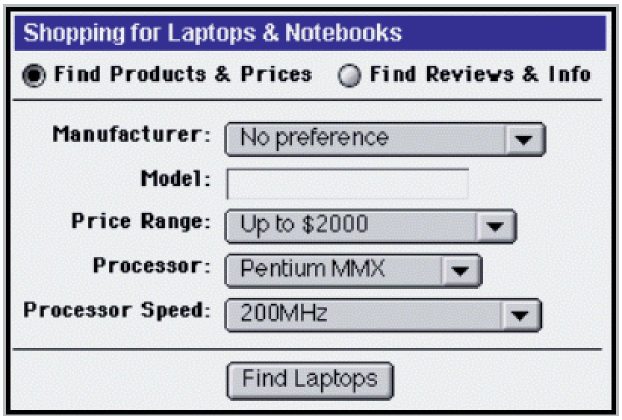

Figure 2. Jango captures shoppers' preferences for price and a limited set of product features and returns a list of product offerings differentiated by price.

shopping agent for online price comparisons. Given a specific product, BargainFinder looks up its price from at least nine different merchant Web sites using Web-browser-like requests. Although a limited proof-of-concept system, BargainFinder offers valuable insight into the issues involved in price comparisons in the online world. For example, a third of the online CD merchants accessed by BargainFinder blocked all of its price requests. One reason was that many merchants don't want to compete on price alone. Value-added services offered on merchants' Web sites were being bypassed by BargainFinder and therefore not likely considered in the consumer's buying decision. However, Andersen Consulting also received requests from an equal number of smaller merchants who wanted to be included in BargainFinder's price comparison. In short, companies competing on price and welcoming exposure wanted to be included; the others didn't.

Jango (jango.excite.com) can be viewed as an advanced BargainFinder (see Figure 2). The original Jango version "solved" the merchant-blocking issue by having the product requests originate from each consumer's Web browser instead of from a central site, as in BargainFinder. This way, requests to merchants from a Jango-augmented Web browser appeared as requests from "real" customers. Such aggressive interoperability makes it convenient for consumers to compare prices from a number of merchants' online catalogs, whether or not merchants welcome such comparisons. While virtual database technology (such as that offered by Junglee, Inc.) and learning techniques for semiautomatically composing "wrappers" for Web sites [5] are helpful for building comparison shopping agents, the process is still done largely by hand and is extremely tedious. In the near future, XML (see Glushko et al.'s article in this issue) as well as mobile agents technology (see Wong et al.'s article in this issue) may make comparison-shopping agents a lot more flexible, open-ended, and easier to implement.

During the negotiation stage, participants settle on price or other terms of the transaction. Most business-to-business transactions involve negotiation (see Sandholm's "Automated Negotiation" in this

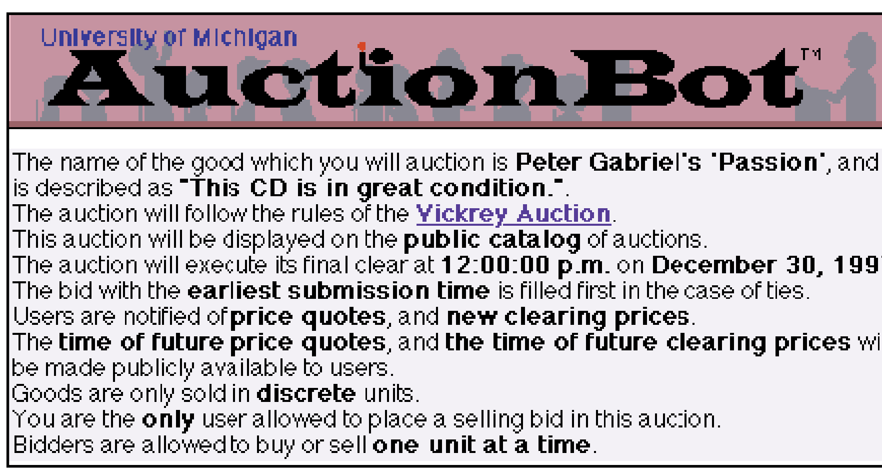

Figure 3. AuctionBot offers many auction protocol permutations.

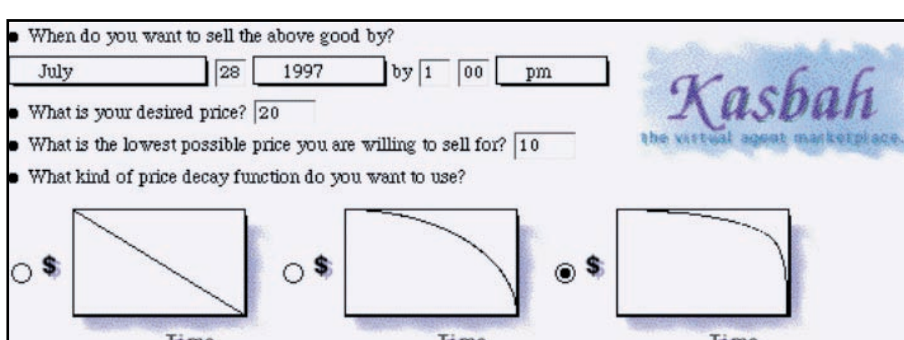

Figure 4. Kasbah is one of the first online agent systems for negotiating the price of consumer products.

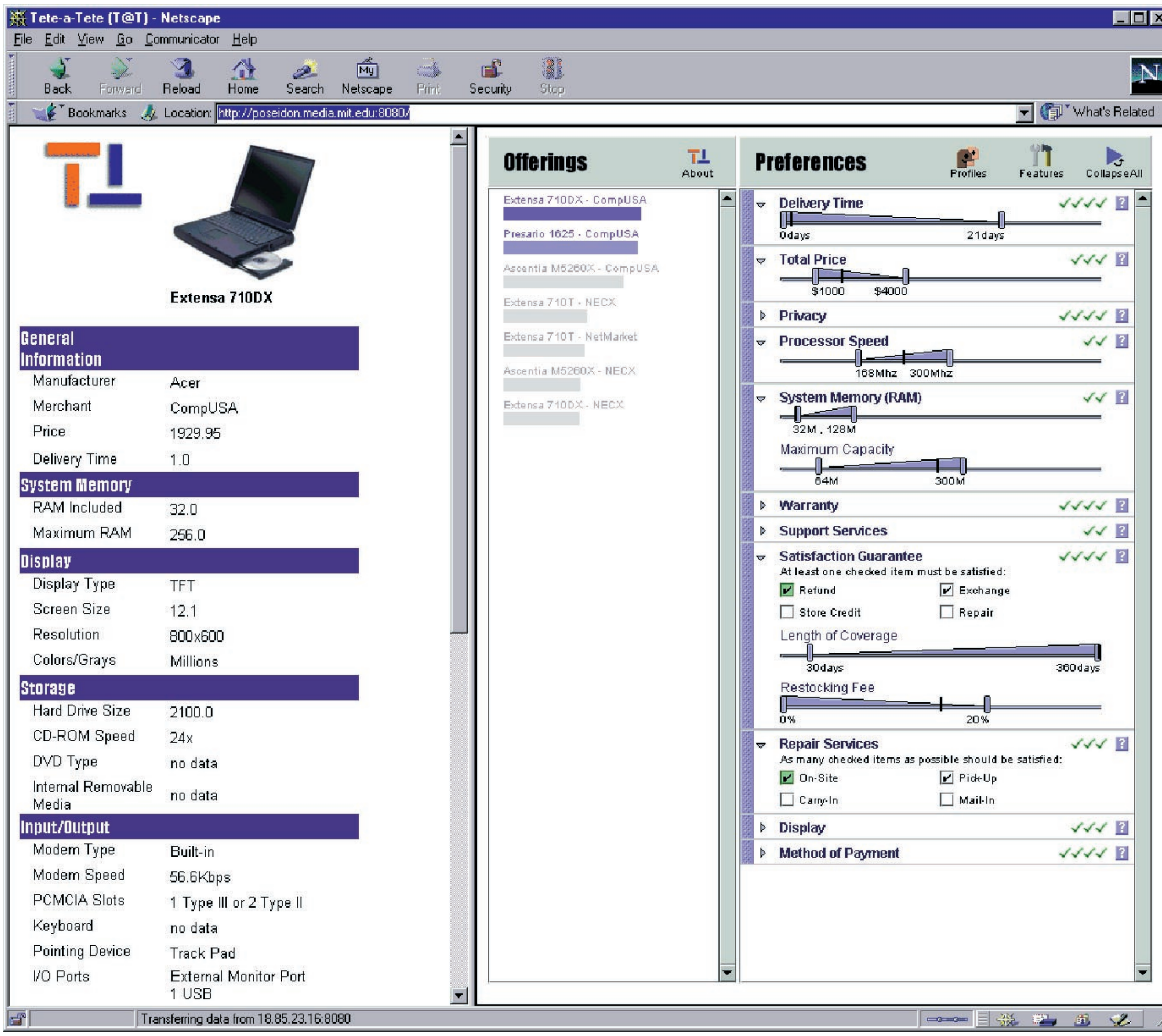

Figure 5. Tête-à-Tête's shopping interface

issue). In retail, we are familiar mostly with fixed prices, even though fixed-price selling was introduced only about 100 years ago. The benefit of dynamically negotiating a price for a product instead of fixing it is that the merchant is relieved from having to determine—a priori—the value of the good. Rather, this burden is pushed into the marketplace. As a result, limited resources are allocated fairly, that is, to those who value them most. However, there are impediments to using negotiation. In the physical world, certain types of auctions require all parties to be geographically colocated in, say, an auction house. Negotiating may also be too complicated or frustrating for the average consumer. Moreover, some negotiation protocols perform over an extended period of time that does not suit impatient or time-constrained consumers. In general, real-world negotiations accrue transaction costs that may be too high for either consumers or merchants.

Many of these impediments disappear in the digital world. For example, OnSale (www.onsale.com) and eBay's AuctionWeb (www.ebay.com/aw) are two popular Web sites selling refurbished and second-hand products through a choice of auction protocols. Unlike physical auction houses, these sites do not require participants to be colocated geographically. However, these sites still require consumers to manage their own negotiation strategies over an extended period—and is where agent technologies come in. Table 1 lists several agent systems—AuctionBot, Kasbah, and Tête-à-Tête—that assist customers negotiating the terms of a transaction.

AuctionBot (auction.eecs.umich.edu) is a general-purpose Internet auction server at the University of Michigan. Its users create new auctions by choosing from a selection of auction types and then specifying its parameters (such as clearing times, method for resolving tie bids, and number of sellers permitted) (see Figure 3). Buyers and sellers can then bid according to the auction's multilateral distributive negotia-

Seven Good Reasons for Mobile Agents

Dispatch your agents; shut off your machine.

Danny B. Lange and Mitsuru Oshima

Mobility is an orthogonal property of agents, that is, not all agents are mobile. An agent can just sit there and communicate with its environment through conventional means, such as remote procedure calling and messaging. We call agents that do not or cannot move "stationary agents." A stationary agent executes only on the system on which it begins execution. If it needs information not on that system or needs to interact with an agent on another system, it typically uses a communication mechanism, such as remote procedure calling.

In contrast, a mobile agent is not bound to the system on which it begins execution [1]. It is free to travel among the hosts in the network. Created in one execution environment, it can transport its state and code with it to another execution environment in the network, where it resumes execution. The term "state" typically means the attribute values of the agent that help it determine what to do when it resumes execution at its destination. Code in an object-oriented context means the class code necessary for an agent to execute.

A mobile agent has the unique ability to transport itself from one system in a network to another in the same network. This ability allows it to move to a system containing an object with which it wants to interact and then to take advantage of being in the same host or network as the object.

Our interest in mobile agents is not motivated by the technology per se but rather by the benefits agents provide for creating distributed systems. There are at least seven main benefits, or good reasons, to start using mobile agents:

They reduce the network load. Distributed systems often rely on communication protocols involving multiple interactions to accomplish a given task. The result is a lot of network traffic. Mobile agents allow users to package a conversation and dispatch it to a destination host where interactions take place locally. Mobile agents are also useful when reducing the flow of raw data in the network. When very large volumes of data are stored at remote hosts, that data should be processed in its locality rather than transferred over the network. The motto for agent-based data processing is simple: Move the computation to the data rather than the data to the computation.

They overcome network latency. Critical real-time systems, such as robots in manufacturing processes, need to respond in real time to changes in their environments. Controlling such systems through a factory network of substantial size involves significant latencies. For critical real-time systems, such latencies are not acceptable. Mobile agents offer a solution, because they can be dispatched from a central controller to act locally and execute the controller's directions directly.

They encapsulate protocols. When data is exchanged in a distributed system, each host owns the code that implements the protocols needed to properly code outgoing data and interpret incoming data. However, as protocols evolve to accommodate new requirements for efficiency or security, it is cumbersome if not impossible to upgrade protocol code properly. As a result, protocols often become a legacy problem. Mobile agents, on the other hand, can move to remote hosts to establish "channels" based on proprietary protocols.

They execute asynchronously and autonomously. Mobile devices often rely on expensive or fragile network connections. Tasks requiring a continuously open connection between a mobile device and a fixed network are probably not economically or technically feasible. To solve this problem, tasks can be embedded into mobile agents, which can then be dispatched into the network. After being dispatched, the agents become independent of the process that created them and can operate asynchronously and autonomously. The mobile device can reconnect at a later time to collect the agent.

They adapt dynamically. Mobile agents can sense their execution environment and react autonomously to changes. Multiple mobile agents have the unique ability of distributing themselves among the hosts in the network to maintain the optimal configuration for solving a particular problem.

They are naturally heterogeneous. Network computing is fundamentally heterogeneous, often from both hardware and software perspectives. Because mobile agents are generally computer- and transport-layer-independent (dependent on only their execution environments), they provide optimal conditions for seamless system integration.

They are robust and fault-tolerant. Mobile agents' ability to react dynamically to unfavorable situations and events makes it easier to build robust and fault-tolerant distributed systems. If a host is being shut down, all agents executing on that machine are warned and given time to dispatch and continue their operation on another host in the network.

But be warned: Do not waste your time searching for the killer application for mobile agents. There are no mobile agent applications, but there are plenty of applications that benefit from using mobile agents. Several applications clearly benefit from the mobile agent paradigm:

E-commerce. Mobile agents are well suited for e-commerce. A commercial transaction may require real-time access to remote resources, such as stock quotes and perhaps even agent-to-agent negotiation. Different agents have different goals and implement and exercise different strategies to accomplish them. We envision agents embodying the intentions of their creators, acting and negotiating on their behalf. Mobile agent technology is a very appealing solution for this kind of problem.

Personal assistance. Mobile agents' ability to execute on remote hosts makes them suitable as assistants performing tasks in the network on behalf of their creators. Remote assistants operate independently of their limited network connectivity; their creators can even turn off their computers. For example, to schedule a meeting with several other people, a user can send a mobile agent to interact with the agents representing each of the people invited to the meeting. The agents negotiate and establish a meeting time.

Secure brokering. An interesting application of mobile agents is in collaborations in which not all the collaborators are trusted. The parties could let their mobile agents meet on a mutually agreed secure host where collaboration takes place without risk of the host taking the side of one of the visiting agents.

Distributed information retrieval. Instead of moving large amounts of data to the search engine so it can create search indexes, agent creators can dispatch their agents to remote information sources where they locally create search indexes that can later be shipped back to the system of origin. Mobile agents can also perform extended searches that are not constrained by the hours during which a creator's computer is operational.

Telecommunication networks services. Support and management of advanced telecommunication services are characterized by dynamic network reconfiguration and user customization. The physical size of these networks and the strict requirements under which they operate call for mobile agent technology to function as the glue keeping the systems flexible yet effective.

Workflow applications and groupware. The nature of workflow applications includes support for the flow of information among coworkers. Mobile agents are especially useful here, because, in addition to mobility, they provide a degree of autonomy to the workflow item. Individual workflow items fully embody the information and behavior they need to move through the organization—independent of any particular application.

Monitoring and notification. This classic mobile agent application highlights the asynchronous nature of these agents. An agent can monitor a given information source without being dependent on the system from which it originates. Agents can be dispatched to wait for certain kinds of information to become available. It is often important that the life spans of monitoring agents exceed or be independent of the computing processes that created them.

Information dissemination. Mobile agents embody the so-called Internet push model. Agents can disseminate information, such as news and automatic software updates, for vendors. The agents bring the new software components, as well as installation procedures, directly to customers' computers where they autonomously update and manage the software.

Parallel processing. Given that mobile agents can create a cascade of clones in the network, another potential use of mobile agent technology is to administer parallel processing tasks. If a computation requires so much processor power that it must be distributed among multiple processors, an infrastructure of mobile agent hosts can be a plausible way to allocate the related processes. **C**

Reference

1. Lange, D., and Oshima, M. *Programming and Deploying Java Mobile Agents with Aglets*. Addison-Wesley Longman, Reading, Mass., 1998.

Danny B. Lange (danny@genmagic.com) is director of agent technology in General Magic, Inc. in Sunnyvale, Calif.
Mitsuru Oshima (oshima@genmagic.com) is a technical lead in General Magic, Inc. in Sunnyvale, Calif.

tion protocols. In a typical scenario, a seller bids a reservation price after creating the auction and lets AuctionBot manage and enforce buyer bidding according to the auction's protocols and parameters. What makes AuctionBot different from most other auction sites, however, is that it provides an application programming interface for users to create their own software agents to autonomously compete in the AuctionBot marketplace. However, as with the Fishmarket Project [6], users encode their own bidding strategies. Fishmarket is not currently being used as a real-world system, but it has hosted tournaments for comparing opponents' handcrafted bidding strategies along the lines of Axelrod's prisoner's dilemma tournaments [7].

In general, real-world negotiations accrue transaction costs that may be too high for either consumers or merchants.

MIT Media Lab's Kasbah (kasbah.media.mit.edu) [8] is an online, multiagent consumer-to-consumer transaction system. Users who want to buy or sell an item create an agent, give it some strategic direction, and send it off into a centralized agent marketplace (see Figure 4). Kasbah agents proactively seek out potential buyers or sellers and negotiate with them on behalf of their owners. Each agent's goal is to complete an acceptable deal on behalf of its user—subject to a set of user-specified constraints, such as initial asking (or bidding) price, a lowest (or highest) acceptable price, a date by which to complete the transaction, and restrictions on which parties to negotiate with and how to change the price over time. Kasbah's agents automate much of the merchant-brokering and negotiation stages for both buyers and sellers.

Negotiation in Kasbah is straightforward. After buying agents and selling agents are matched, the only valid action in the negotiation protocol is for buying agents to offer a bid to selling agents with no restrictions on time or price. Selling agents respond with either a binding "yes" or "no." Given this protocol, Kasbah provides buyers with one of three negotiation strategies: anxious, cool-headed, and frugal—corresponding to a linear, quadratic, and exponential function respectively for increasing its bid for a product over time (similar functions exist for selling agents). The simplicity of these negotiation heuristics makes it intuitive for users to understand what their agents are doing in the marketplace. This understanding is important for user acceptance, as observed in a recent Media Lab experiment [8]. A larger Kasbah experiment now under way at MIT allows students to buy and sell books and music.

The Kasbah system also incorporates a trust and reputation mechanism called the "better business bureau." Upon completion of a transaction, both parties may rate how well their counterpart managed his or her half of the deal (such as for accuracy of product condition and completion of transaction). Kasbah agents use accumulated ratings to determine whether they should negotiate with agents whose owners fall below a user-specified reputation threshold. More details on the specific algorithms used in Kasbah's better business bureau are in [9].

Tête-à-Tête (ecommerce.media.mit.edu/tete-a-tete/) [10] provides a unique negotiation approach to retail sales. Unlike most other online negotiation systems that competitively negotiate over price, Tête-à-Tête's consumer-owned shopping agents and merchant-owned sales agents cooperatively negotiate across multiple terms of a transaction, including warranties, delivery times, service contracts, return policies, loan options, gift services, and other merchant value-added services. Based on bilateral argumentation [11], Tête-à-Tête's integrative negotiations comprise an exchange of XML-based proposals, critiques, and counterproposals. For example, a shopping agent may receive proposals from multiple sales agents. Each one defines a complete product offering, including a product's configuration and price and the merchant's value-added services. The shopping agent evaluates and orders these proposals based on how well they satisfy its owner's preferences (expressed as multiattribute utilities) (see Figure 5). Shoppers dissatisfied with the proposals can critique them along one or more dimensions. The shopper agent broadcasts these preference changes to the sales agents that, in turn, use them to counterpropose better product offerings.

Tête-à-Tête considers product features and merchant features equally throughout negotiations to help the shopper simultaneously determine what to buy and whom to buy from. This integration of product and merchant brokering through integrative negotiations enables constraints on product features to influence the decision of whom to buy from. For example, only a certain merchant may provide a particular product configuration. Likewise, constraints on merchant features can influence the decision of what to buy. So, if no merchant can accommodate the overnight delivery of a specific product, an alternate product that can be delivered overnight may be deemed a better overall value.

Future Directions

Software agents are helping buyers and sellers combat information overload and expedite specific stages of the online buying process. Today's first-generation agent-mediated e-commerce systems are already creating new markets (such as low-cost consumer-to-consumer goods) and beginning to reduce transaction costs in a variety of business processes. The industries affected first are those dealing with perishables (such as travel, theater and concert tickets, and network bandwidth availability), and surplus inventory and commodities (such as gas, electricity, pencils, music, and books).

However, we still have a way to go before software agents transform the way companies conduct business. This change will occur as agent technologies mature to better manage ambiguous content, personalized preferences, complex goals, changing environments, and disconnected parties. The greatest changes may occur once standards are adopted and resolved to unambiguously and universally define goods and services [10], consumer and merchant profiles, value-added services, secure payment mechanisms, and interbusiness electronic forms.

Looking even further into the future, agents will explore new types of transactions in the form of dynamic relationships among previously unknown parties. At the speed of bits, agents will strategically form and reform coalitions to bid on contracts and leverage economies of scale—in essence creating dynamic business partnerships that exist only as long as necessary. It is in this third-generation of agent-mediated e-commerce that companies will be at their most agile, and marketplaces will approach perfect efficiency. C

References

1. Maes, P. Agents that reduce work and information overload. *Commun. ACM 37,* 7 (July 1994), 31–40.
2. Moukas, A., Guttman, R., and Maes, P. Agent-mediated electronic commerce: An MIT Media Laboratory perspective. In *Proceedings of the International Conference on Electronic Commerce* (Seoul, Korea, Apr. 6–9). ICEC, Seoul, 1998, pp. 9–15.
3. Terpsidis, I., Moukas, A., Pergioudakis, B., Doukidis, G., and Maes, P. The Potential of electronic commerce in reengineering consumer-retail relationships through intelligent agents. In *Advances in Information Technologies: The Business Challenge*, J.-Y. Roger, B. Stanford-Smith, and P. Kidd., Eds., IOS Press, Amsterdam, Denmark, 1997.
4. Shardanand, U., and Maes, P. Social information filtering: Algorithms for automating 'word of mouth.' In *Proceedings of the Computer-Human Interaction Conference CHI'95* (Denver, Colo., May 7–11). ACM Press, N.Y., pp. 210–217.
5. Doorenbos, R., Etzioni, O., and Weld, D. A scalable comparison-shopping agent for the World-Wide Web. In *Proceedings of the First International Conference on Autonomous Agents Agents'97* (Marina del Rey, Calif., Feb. 5–8). ACM Press, N.Y., 1997, pp. 39–48.
6. Rodriquez, J., Noriega, P., Sierra, C., and Padget, J. FM96.5: A Java-based electronic auction house. In Proceedings of the Second International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology PAAM'97 (London, U.K., Apr.). Practical Application Company, London, 1997.
7. Axelrod, R. *The Evolution of Cooperation*. Harper Collins, N.Y., 1984.
8. Chavez, A., Dreilinger, D., Guttman, R., and Maes, P. A real-life experiment in creating an agent marketplace. In *Proceedings of the Second International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology PAAM'97* (London, U.K., Apr.). Practical Application Company, London, 1997.
9. Zacharia, G., Moukas, A., and Maes, P. Collaborative reputation mechanisms in electronic marketplaces. In *Proceedings of the HICSS-99 Conference, Electronic Commerce Minitrack* (Maui, Hawaii, Jan. 5–9). IEEE Computer Society, 1999.
10. Guttman, R., and Maes, P. Agent-mediated integrative negotiation for retail electronic commerce. In *Proceedings of the Workshop on Agent-Mediated Electronic Trading AMET'98* (Minneapolis, May 1998).
11. Parsons, S., Sierra, C., and Jennings, N. Agents that reason and negotiate by arguing. *J. Log. Comput.*

Pattie Maes (pattie@media.mit.edu) is an associate professor in MIT's Media Laboratory where she founded and directs the Software Agents Group.

Robert H. Guttman (guttman@media.mit.edu) is a cofounder of Frictionless Commerce, Inc., a start-up building the next generation of shopping advisory solutions for retail e-commerce.

Alexandros G. Moukas (moux@media.mit.edu) is a cofounder of Frictionless Commerce, Inc., an AT&T Fellow, and a doctoral candidate in MIT Media Laboratory's Software Agents Group (on leave).