A Real-Life Experiment in Creating an Agent Marketplace

Anthony Chavez **Daniel Dreilinger** **Robert Guttman** **Pattie Maes**
MIT Media Laboratory
20 Ames Street
Cambridge, MA 02139
anthonyc@microsoft.com; daniel/guttman/pattie@media.mit.edu

Abstract

Software agents help people with time consuming activities. One increasingly popular application for software agents is electronic commerce, namely having agents buy and sell goods and services on behalf of users. We recently conducted a real-life experiment in creating an agent marketplace, using a slightly modified version of the Kasbah system [Chavez96]. Approximately 200 participants intensively interacted with the system over a one-day, six-hour period. This paper describes the setup of the experiment, the architecture of the electronic market and the behaviors of the agents. We discuss the rationale behind the design decisions and analyze the results obtained. We conclude with a discussion of current experiments involving thousands of users interacting with the agent marketplace over a long period of time, and speculate on the long-range impact of this technology upon society and the economy.

1. Introduction

Software agents help people with time consuming activities [Maes95]. In the past, a range of roles for agents have been explored such as filtering information, automating repetitive tasks, and making referrals and introductions [PAAM96] [Bradshaw97]. One increasingly popular application for software agents is electronic commerce, namely having agents buy and sell goods and services on behalf of users. Traditional shopping activities require a large effort from a user and include searching for parties interested in selling or buying what the user wants to buy or sell (e.g., by sifting through catalogs, advertisements in newspapers and television, shelves in stores, etc.), comparing prices and other features of the good or service to help make an optimal purchase decision, and exchanging currency for product through some agreed upon, and ideally secure, channels.

The activity of buying and selling among end-consumers (e.g., classified ads, yard sales and flea markets) is a particularly time-consuming and inefficient form of shopping and often includes additional steps such as negotiating on price or other features. We believe that the effective use of software agents can dramatically reduce transaction costs involved in electronic commerce, in general, and in consumer-to-consumer transactions, in particular.

The pervasiveness and growing popularity of networks such as the Internet and online services have made it possible to facilitate automated transactions. Existing efforts in the area of electronic commerce are still fairly simple, in the sense that they don't radically change the way transactions happen or don't create any new markets. Current efforts include the ability to pay at online stores with a credit card or with electronic cash and online listings of goods for sale which are more easily searchable. Some more interesting experiments which change the way transactions normally occur (and hence raise many issues) include BargainFinder [BF] and Fido [Fido]. Both of these systems present the concept of an agent which can shop for the best price for a good on behalf of a user. The Telescript white paper [Telescript96] also provides some inspiring scenarios for how transactions can be implemented by agents; however, to our knowledge, these scenarios have not yet been implemented.

The Artificial Intelligence (AI) literature includes a substantial body of work on negotiating agents [Rosenschein94], but little of this work has been used in real applications. Since 1994 the Software Agents group of the MIT Media Laboratory has embarked upon a research program to create electronic agent marketplaces. Chavez and Maes describe the architecture of Kasbah, an electronic marketplace where agents buy and sell to one another on behalf of their users [Chavez96]. The current paper reports on an experiment performed in October 1996 which involved a large group of people using a slightly revised Kasbah system.

More specifically, about two hundred people intensively interacted with the system in a one-day, six-hour experiment. The paper reports on the setup for the experiment, the architecture, and the methods used for the electronic market and the agents. We also discuss the rationale behind the design decisions and, most importantly, describe the results obtained. The paper concludes with a discussion of a current experiment which involves up to 10,000 users interacting with the electronic marketplace over a long period of time and, finally, we speculate about the impact of a widespread usage of this technology upon society and the economy.

2. The Experiment Setup

On October 30th 1996, the Media Laboratory organized a symposium on "Digital Life" for 171 attendees from industry. Many of these people were not technically inclined and none had been introduced to the Kasbah concept beforehand. The attendees were given a total of three objects as well as some money (50 "bits", as we called our unit of currency).[1] We invited the attendees to create "selling" agents to sell some of the objects they owned but did not want and to create "buying" agents to buy some of the objects they wanted to own. When a user created an agent, the user would determine its characteristics such as: is it a buying or a selling agent, what does the agent buy or sell (chosen from a limited list of goods), what is the initial asking (or

[1] We decided to use "fake" money, because we did not want to upset our test users in case some things did not work as planned. However, we could have used real money or digital cash/credit, which would

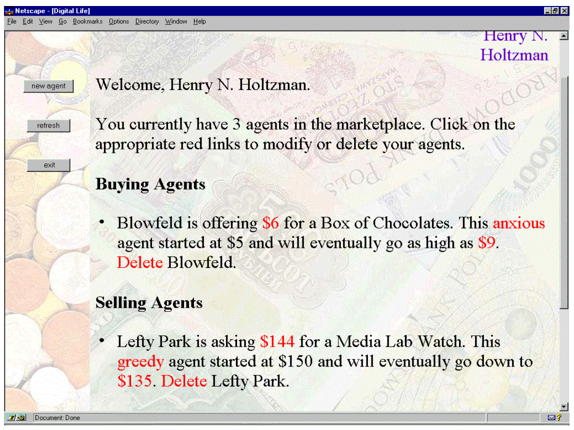
offer) price, what is the final lowest asking (or final highest offer) price and what is the “strategy” the agent will use to lower (or raise) its price over time. Figure 1 shows some users interacting with the agent marketplace.

Figure 1: Participants interacting with the agent marketplace

People were able to create agents all day long. At the end of the day, they could exchange their money for wine (70 bits for one bottle). Hence, wine could be said to correspond to the "gold standard" of the economy created; people would decide how many bits objects were worth to them based on the fact that a bottle of wine was worth 70 bits.

Participants in the agent marketplace experiment were immersed in an environment involving interactive kiosks, a transaction center, and several large-scale displays. Users created and interacted with their agents via 20 keyboardless kiosks which consisted of a computer workstation, monitor, mouse, and bar-code reader. Users initiated a personalized session by scanning their name tag badge with the bar-code reader. Once automatically logged in, users were provided with a list of their active agents (see Figure 2), a list of completed transactions, and options for creating new agents and changing old ones. The interface was entirely mouse driven — users navigated a simple point-and-click interface to create, modify, and monitor their agents. No keyboard input was required. The interface was designed to facilitate short interactions, so that all participants could use the 20 kiosks during the 30 to 60 minute breaks. A typical user session lasted about five minutes.

In addition to the kiosks, several other devices and displays were used to disseminate information. Each participant received a personal alphanumeric pager which would notify them whenever one of their agents had made a deal. For example, if Andy had created an agent named James Bond to sell a lunch pail, and this agent then made a deal with one of Pattie's agents, Andy was paged with the message: *“Andy, I sold your Media Lab lunch pail to Pattie Maes for $53. --James Bond".* Pattie was paged with a similar message from her agent. At one point, about halfway through the day, buying and selling agents sent suggestions to users' pagers when it seemed unlikely that they would make a deal by the end of the day with the price parameters given by the user. For example, if a user created an agent to buy a box of chocolates


Netscape - [Digital Life]
File Edit View Go Bookmarks Options Directory Window Help
new agent
refresh
exit
Document: Done

Henry N. Holtzman

Welcome, Henry N. Holtzman.

You currently have 3 agents in the marketplace. Click on the appropriate red links to modify or delete your agents.

Buying Agents

- Blowfeld is offering $6 for a Box of Chocolates. This anxious agent started at $5 and will eventually go as high as $9. Delete Blowfeld.

Selling Agents

- Lefty Park is asking $144 for a Media Lab Watch. This greedy agent started at $150 and will eventually go down to $135. Delete Lefty Park.

Figure 2: Part of the welcome screen shown to a user upon login

and the maximum price the agent would offer was 20 percent lower than the average selling price of chocolates throughout the day, that agent would page the user with the message: *"Tip: you may want to raise my maximum offer if you want to buy those chocolates – James Bond."* (If we had two-way pagers, we would have provided the option to tell the agent to change the price accordingly.)

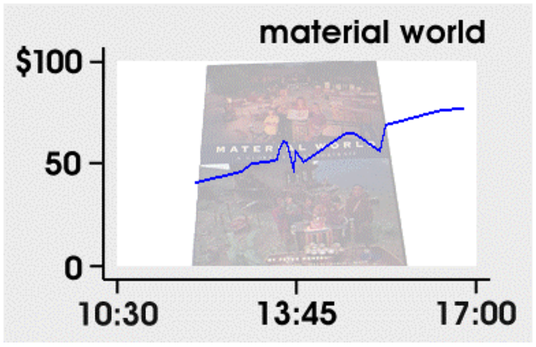


Figure 3: Screenshot taken from the time-series display

An 8 foot high projection display and two large monitors provided visualizations of marketplace statistics including histograms of buying and selling prices for each of the nine good types and a time-series projection illustrating the trend in actual transaction prices for each good over time. Figure 3 shows one of the time-series displays. A scrolling LED 'ticker-tape' displayed up-to-the-minute market information about each of the goods being traded, such as the highest bid, lowest asking price, and last transaction amount.

Upon being notified of a transaction (either via the pager or the user interface), users were told to report to the transaction center to exchange the good for the price agreed upon. We anticipated situations where one party would arrive several minutes after the other party (or fail to show up altogether) and wanted to avoid making the participants wait around since they had very busy schedules. Thus, when two users arrived asynchronously, the transaction center, which had a supply of extra money and surplus goods, completed the one-sided deals. If the two users arrived at the same time, they carried out the transaction privately between themselves. The transaction center was also used to provide general information and assistance, to make change, and to sell the wine at the end of the day.

3. Architecture and Implementation

3.1. Overall Architecture

The architecture of the agent marketplace system comprises three primary components:

1. **Front-end**: a Web interface handles all of the user interaction. It consists of a set of Perl CGI scripts located on a Web server.
2. **Back-end**: the marketplace engine is where the agents actually "live" and interact with one another. The back-end is implemented in Java.
3. **Auxiliary components**: there are also several auxiliary components that generate the visual display files, send out notification pages to users, send marketplace data to ticker tape displays, and implement the login process.

The architecture was designed to be fairly well modularized, so that changes to one component would cause at most minimal changes to be made to the others. The interfaces between components were designed to be as general and flexible as possible. This design decision proved to be a wise one, as the front-end design was constantly in a state of flux, yet most of these modifications were made without requiring constant changes to the back-end. Figure 4 shows the architecture of the agent marketplace system.

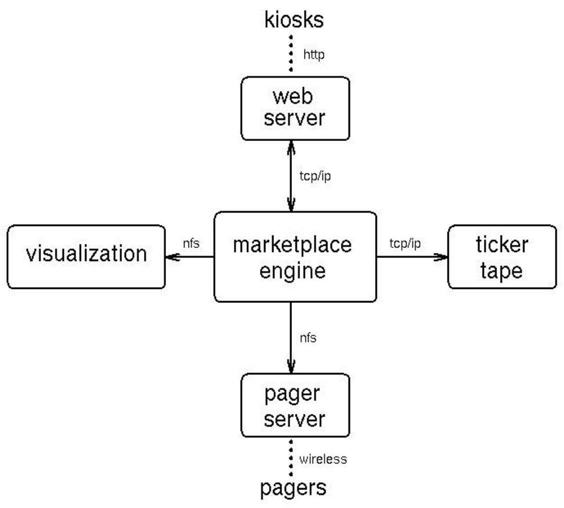


Figure 4: Architecture of the Agent marketplace system

The physical topology of the system is as follows. The front-end CGI Perl scripts resided on two Web servers. The back-end marketplace engine was located on a single high-performance workstation. The other auxiliary components were scattered across several other servers. The user kiosks consisted of 20 Intel computers running a specially modified version of Microsoft Internet Explorer. The kiosks were set up so that user interaction was done entirely via the mouse and special barcode scanners (to read people's name badges and log them on). The front-end and back-end communicated with one another via TCP/IP sockets. The user kiosks communicated with the two front-end servers via HTTP.

3.2. The Back-end Marketplace Engine

The back-end engine implements a request-response service: a client sends it a request, the back-end services that request, and then sends back to the client a response. From a high-level functional viewpoint, these are the marketplace services that the back-end provides:

- Add a new user to the marketplace
- Create a new selling or buying agent for a given user
- List all the agents and their corresponding properties for a given user
- Delete a specified agent for a given user
- Modify parameters of a specified agent for a given user
- Get current market data

The front-end sends requests for the above services to the back-end via a simple protocol that we devised. In addition to handling requests from the front-end, the back-end is where the agents do their negotiating and deal-making. Conceptually, one can think of the back-end as the place where agents are "running around" talking to one another, “haggling” and trying to find the best possible deal on behalf of their users, all in parallel.

3.3. Selling and Buying Agents

Software agents are long-lived programs that perform some task on behalf of a user [Maes95]. In our case, the agents try to buy or sell some good for the best possible price on behalf of the user which created them. To do this, the agents negotiate with other users’ agents in the marketplace, trying to find the best deal subject to a set of user-specified constraints. When two agents “make a deal” with one another, they notify their respective users so that the transaction can be physically consummated.

The design of the agents is derived from that of Kasbah [Chavez96]. However, the functionality was simplified after early user tests indicated that people may feel uncomfortable delegating a buying/selling task to an agent whose behavior is complex and difficult to predict. In fact, the tests which we performed lead us to believe that the level of intelligence or sophistication of a buying/selling agent may be limited more by user acceptance constraints than by limitations of AI technology. This was particularly the case when agents have the authority to complete transactions as in our experiment.

Selling and buying agents are architecturally symmetrical. The relatively simple set of agent parameters include:

- good to sell (buy): in order to keep the user interface for the experiment simple, we decided to only have nine good types which users could sell (buy) — three books, a mug, a camera, a box of chocolates, a lunch pail, a watch, and a canvas bag.
- desired price to sell (buy) for: this is the price for which the user ideally would like to sell (buy) the good, i.e. the initial price which the agent will ask (offer).
- lowest acceptable price (highest offer): this is the lowest price (highest offer) for which the user would ultimately be willing to sell (buy) the good.
- date to sell (buy) by: this parameter was hard-coded to 5:00pm for the experiment reported upon in this paper (i.e., users were not allowed to set it).
- agent strategy: this allows the user to specify the negotiating “behavior” of their agent. There were three options to choose from: “anxious”, “cool-headed”, and “frugal”.

By specifying these parameters, the user defines and constrains the behavior of his or her agent. The behavior of a selling agent is as follows (with buying agents behaving similarly). The agent will try to sell the specified good. It will initially start by asking the desired price. If it is able to find someone willing to pay that price or more, then it makes a deal (at the highest price). Otherwise, the agent lowers its asking price and continues to negotiate. By the end of the day (the hard-coded date to sell by), if the agent has not yet made a deal, it will have lowered its asking price to the lowest

The way in which a selling agent lowers its price throughout the day is dependent upon its strategy. An anxious agent, in a big hurry to sell, will lower its price very quickly towards the lowest acceptable price. A greedy agent, on the other hand, will lower its price slowly, waiting until the very end of the day before lowering its price substantially. A cool-headed agent will strike a balance between these extreme strategies. The section on agent behavior explains how these strategies were implemented.

From observations made during the experiment, it appeared that users of the system had little difficulty in understanding what their agents were doing. They seemed to perceive the agent behavior in the way described above. Some users even created three agents with different strategies to sell the same good, so as to test whether the agents displayed the right behavior.

3.4. Agent Communication

In order for agents to negotiate in the marketplace, they need to be able to communicate with one another. Agent communication is based upon a request-response protocol and is strictly agent-to-agent. There is no broadcasting of messages and an agent cannot eavesdrop on a conversation between two other agents. In the implementation used for the agent marketplace experiment, there were just two types of request messages an agent could send:

- “What are you currently asking?”: the agent responds with its current asking price (i.e., the price at which it is willing to buy/sell)
- “I offer X. Will you accept?”: the agent responds with either “yes” or “no”. Note that this request message is considered binding; that is, the agent making the offer (sending the message) is implying that if the response is “yes”, it is bound to that deal. Likewise, the agent responding “yes” has bound itself as well.

Clearly, the current conversation space is rather simple. For more sophisticated agents interactions, a richer language will be needed. The current work section expands upon this.

3.5. Agent Behavior

Now that we have described the user’s perceived behavior of their selling or buying agent, let us turn to the implemented behavior of the agent. It should be obvious that the behavior of the agent needs to correspond closely to the user’s perception of how it will behave; if not, then the user is sure to be disappointed or possibly upset. This is a general challenge facing agent research, namely to make sure that the agent lives up to the user’s expectations.

Given this criteria, here is a selling agent’s actual implemented behavior (again, the strategy for a buying agent would be the intuitive opposite). An agent has an internal variable called “current ask price”. This is always set to the price which the agent is currently willing to sell the good for. As time passes, the agent will adjust this price — selling agents will lower it, buying agents will raise it. The price will never be set lower than the lowest acceptable price that the user had specified. When the agent is first created, “current ask price” is set to the desired price. How the agent adjusts its “current ask price” over time is described below.

Agents must be able to adjust their asking prices over time in order to be able to make deals. Currently, these price adjustments are deterministic, i.e. they follow some function of time. In the future, we plan on building agents which can adjust their price according to real-time marketplace factors (e.g., the number of other agents selling the same good, the number of interested buyers, etc.). The three agent strategies described above map directly to price adjustment curves, as shown in Figure 5. These price adjustment curves show the agent’s “current asking price” as a function of time. As you can see, the “current asking price” always starts at the user-specified desired price. By the date to sell by (a hard-coded date in this particular experiment), the current asking price is always at the user-specified lowest acceptable price.

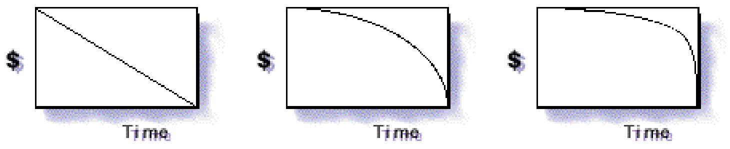


Figure 5: Price adjustment curves for selling agents
(left to right: “anxious”, “cool-headed”, “greedy”)

Between highest and lowest price is where the curves for the three strategies differ. For the anxious strategy, the curve decays linearly. For the greedy strategy, the curve decays much more slowly following an inverse-cubic shape. Intuitively, this makes sense — a greedy selling agent will naturally lower its asking price more slowly than one that is anxious to sell. Naturally, the cool-headed strategy’s curve lies between that of the anxious and greedy strategies following an inverse-quadratic shape. At any point in time, the agent can compute its current asking price using these curves. This calculation is done whenever the agent needs to access the “current asking price” variable in order to make a decision (e.g., whether or not it should accept an offer).

When an agent (buying or selling) makes a deal, it sends out a notification message to its owner (the user who created it) telling him or the identity of the other party and what the deal price is. For the agent marketplace experiment, the notification messages were delivered via pagers. Once an agent makes a deal, it ceases to negotiate with other agents and asks the marketplace to remove it from the list of “active” agents. This ensures that other agents will not be able to send it messages. For all intents and purposes, the agent has terminated.

Agents have two behavior modes: passive and pro-active. The passive behavior is what the agent will do when other agents make initiatives towards it. The pro-active behavior is what steps the agent will take towards making the best possible deal. We believe it is important for agents to have both passive and pro-active behaviors, especially in long-term marketplaces that include a multitude of agents whose internal behaviors and strategies are not publicly known. If an agent has only a passive strategy, and makes no active attempts to find deals, there is the danger that all other agents in the marketplace will also only have a passive strategy. In this case, no deals will ever be made, because all agents are waiting for others to make contact, which no one ever does.

In the experiment performed, an agent's passive strategy includes the following. If the agent is made an offer for a good which is greater than or equal to the "current ask price", it will accept it. Otherwise, it will reject it. The agent's passive behaviors are only triggered when it is made an offer by another agent. An agent's pro-active behavior includes finding the agent X who is willing to pay the most for the good it is selling. Ask agent X what it is currently willing to pay. If X's offer is greater than the agent's "current ask price", then make an offer to X at that price. Otherwise, make an offer at the "current ask price". The point of the agent first asking X its own asking price is that X might be willing to pay more than the agent is currently asking. After all, the agent's job is to find the best possible deal; if there's a customer in the marketplace who will pay more than the asking price, then the agent should grab that deal if it wants to be considered competent by its owner. The agent's pro-active behavior is triggered by the marketplace. The marketplace acts as a scheduler which cycles through the agents and tells each to "do its thing". Note that in running its pro-active behavior, the agent will send a message to another agent, thus triggering that agent's passive behavior.

3.6. The Marketplace

The marketplace maintains all agents and user information. It keeps track of:

- all the "active" agents: the agents which have not yet made deals
- all the agents for each user: the marketplace tracks all of their agents, including the "terminated" ones which have already made a deal
- marketplace data: statistics on deals that have been made, current asking prices for each type of good, etc.

The marketplace also provides a useful "brokering" service to the agents which inhabit it. By providing this service, the marketplace frees the agents from having to do this additional work. This "brokering" service works as follows. An agent can ask the marketplace to find an agent, either buying or selling, who has the current best asking price for a particular good. For selling agents, this would be the agent with the current lowest asking price. For buying agents, this would be the agent with the current highest asking price. Recall from above that the agent negotiation strategy is to always make offers to the agent with the current best asking price (for a specified good).

The marketplace is also responsible triggering the agent's "pro-active" behaviors described above. The back-end is implemented as a Java program; the marketplace and the agents are Java objects within that program. The behaviors of the agents (as well as the marketplace) are embedded as methods in these objects. For each agent, there is a "run-pro-active-behavior" method which, when called, will take the appropriate actions. We chose not to have each agent run within an individual thread because of synchronization difficulties. Thus, every agent needs to have its "run-pro-active-behavior" method called frequently, in order to effectively simulate all agents running "in parallel". The continuously cycles through all the active agents in the marketplace and calls their "run-pro-active-behavior" method. When it makes this call, the marketplace passes execution control of the thread to the agent. The agent can then send requests to the marketplace, to other agents, or whatever else its "run-pro-active-behavior" method implements.

The "run-pro-active-behavior" method of each agent should terminate within a reasonable amount of time for resource fairness. Since we built the agents for the agent marketplace experiment, this was easy to ensure, but in a more open market system where third-parties can submit their own agents, it will be necessary to take more explicit steps to ensure fairness (e.g., using a resource allocation scheme, perhaps even a market-based approach). As the marketplace cycles through all of the active agents and "runs" them, some of them will make deals and remove themselves from the set of active agents. Others will expire (run past the user-specified date to sell by) and be removed by their owner. The order in which the marketplace "runs" the active agents is constrained by the following rule: once an agent is "run", it will not be "run" again until all other active agents have also been "run". This rule ensures that no agent will have more opportunities to make a deal than any other agent.

In addition to running the agents, the marketplace also services the requests coming from the Web server scripts. It was a design criteria that requests to the marketplace be processed as quickly as possible, so that the end-user standing at the kiosks would experience only minimal delays. To achieve this goal, the marketplace interleaves running the agents with polling the socket from the web server to see if a request has arrived. The marketplace gives priority to processing Web server (i.e., user) requests. If there is a queue of five pending requests, all of these will be processed before another agent is run.

4. Results, Analysis and Observations

The quantitative and qualitative results from our agent marketplace experiment were largely as expected with a few surprises. We first look at and analyze quantitative data and then discuss some qualitative observations.

4.1. Quantitative Analysis

Between 11:00am and 5:00pm, the 171 participants created 625 agents of which 510 made deals as shown in Figure 6. This graph also shows how many agents were created and how many made deals where made for each good. For example, many agents were created for cameras (145) although fewer were created for lunch pails (14). Two factors that we observed which determined how many agents were created for each good were the overall quantity of the good (which varied widely) and the participants' interest in trading it.

Figures 6 and 7 show the total number of agents, the number of active agents (those agents that have not yet made deals), and the cumulative number of completed deals made over the course of the day. The kiosks were accessed heavily during program intermissions and more moderately during lunch. This coincides with the number of agents spikes at the start of the experiment and at particular times - roughly between 11:00am and 12:00pm, 1:00pm and 2:00pm, and again at 3:00pm. The noticeable increase of agents just before the 5:00pm market close was due to the transaction center infusing agents into the marketplace to distribute the remaining goods.

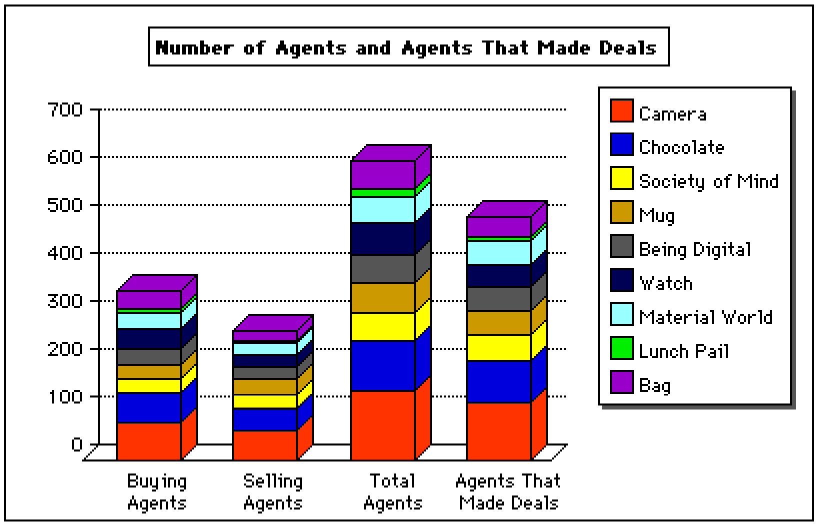


Figure 6: Number of agents and agents that made deal

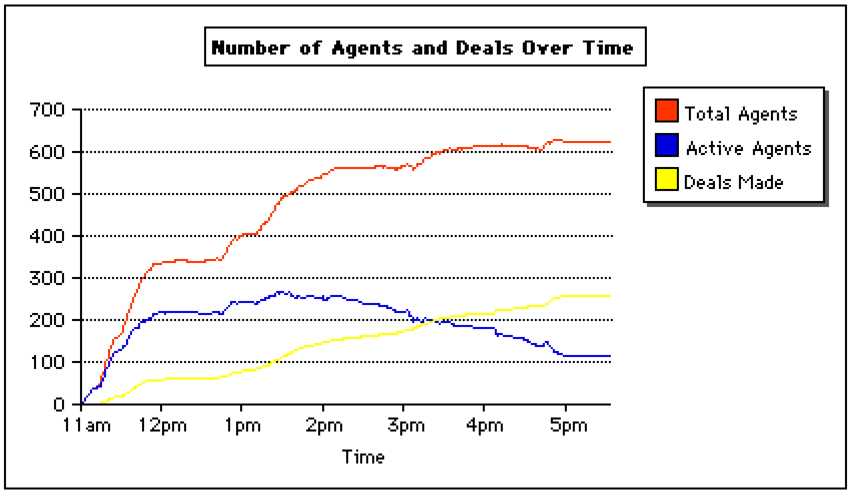


Figure 7: Number of agents and deals over time

We can see that after about 1:30pm, the number of active agents left in the marketplace began to decrease due to active agents making deals. This corresponds with the substantial rise in deals made around that 1:30pm time frame. The flatline of agents and deals at 5:00pm is due to kiosks being switched off and the convergence of each agent's negotiation price to its termination price at its 5:00pm hard-coded deadline.

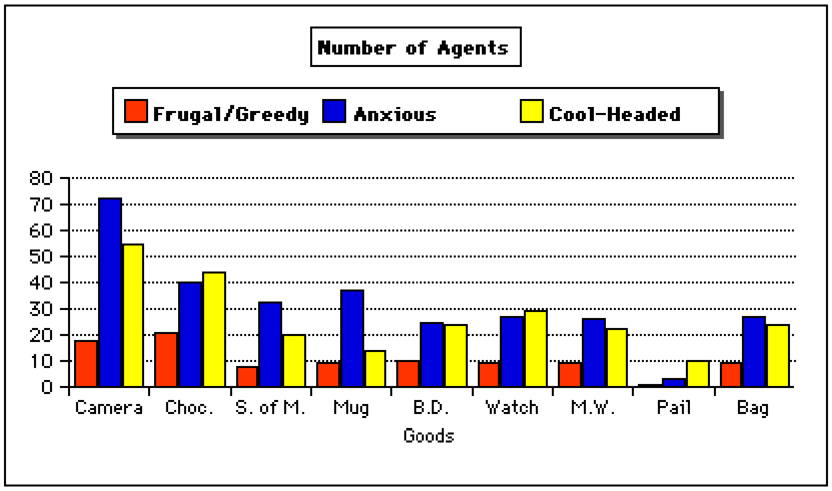


Figure 8: Number of agents (of each strategy for each good)

One intriguing aspect of the agent marketplace experiment was the participants' ability to choose for each agent one of three negotiation strategies: frugal (or greedy for a selling agent), anxious, or cool-headed as discussed in the previous section. Among other factors (e.g., initial price), the chosen strategy helps determine the likelihood that the agent will make a deal and at what price. Figure 8 shows a histogram taken after the market closed at 5:00pm of how many agents there were of each strategy broken down by good.

It is interesting to see in Figure 8 that most agents were given an anxious strategy and very few agents were given a frugal or greedy strategy especially since the user interface did not default to any particular choice and that anxious was the last listed of the three strategies. Either the participants were eager to trade goods and chose the anxious strategy accordingly or the names "frugal" and "greedy" were deterrents due to their negative connotations in our society. A few informal surveys indicate the former.

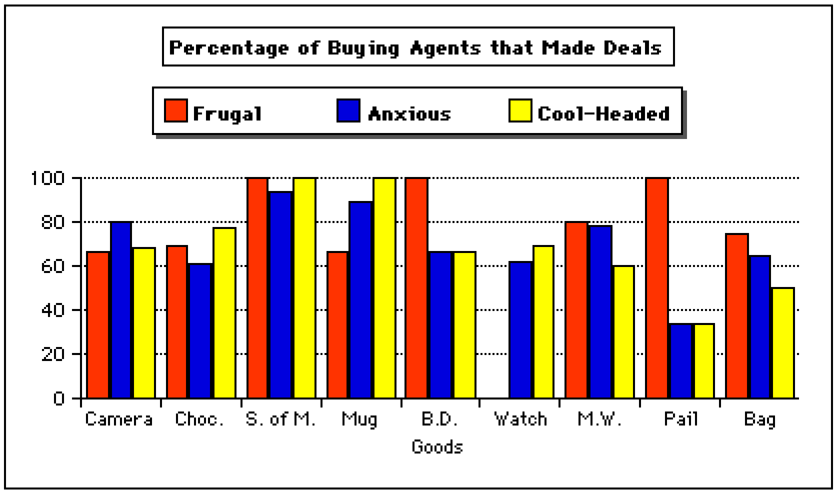


Figure 9: Percentage of buying agents that made deals

A more interesting question is how did each strategy perform in general? One would expect to see a higher percentage of deals being made by anxious agents than frugal (or greedy) agents since they more quickly change their price over time and thus would more quickly find an agent willing to trade. This is true for mugs as shown in Figure 9. For mugs, 90% of all anxious buying agents made deals whereas only 65% of all frugal buying agents made deals. However, the other goods show non-intuitive results. In most cases, frugal agents made the highest percentage of deals.

A possible explanation for this is that participants were able to modify their agents prices throughout the day in order to make deals more quickly. In fact, users made 6.5 agent modifications or deletions on average. This behavior of manually negotiating for an agent would change the nature of any negotiation strategy. An alternate explanation is that we should instead be looking at each agent's lifespan (i.e., how much time it took each agent to make a deal). Unfortunately, this data was not adequately captured to perform such an analysis.

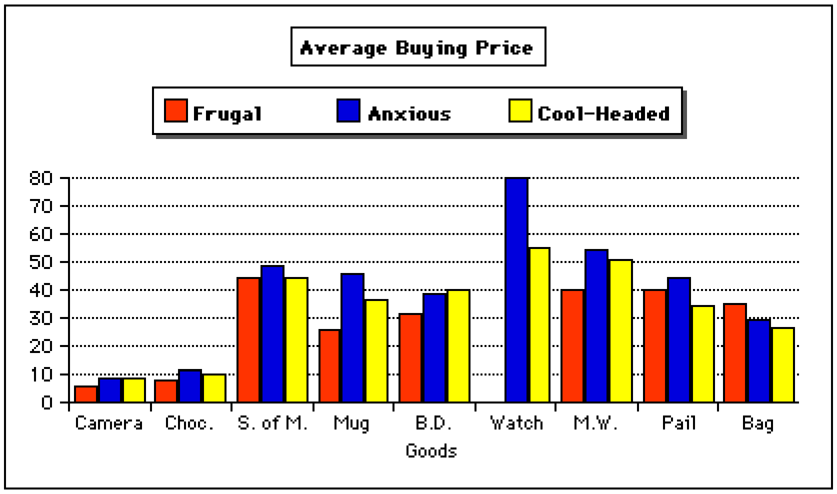


Figure 10: Average buying price

Another expectation based on strategy is that purchases made by anxious agents should be higher priced on average than deals made by frugal agents since they more quickly increase their price over time. (The inverse would be expected for sales.) As shown in Figure 10, this expectation holds for virtually all goods. If we look at mugs again, we see that the average price paid by an anxious agent is about $45 whereas the average price paid by an frugal agent is about $25.

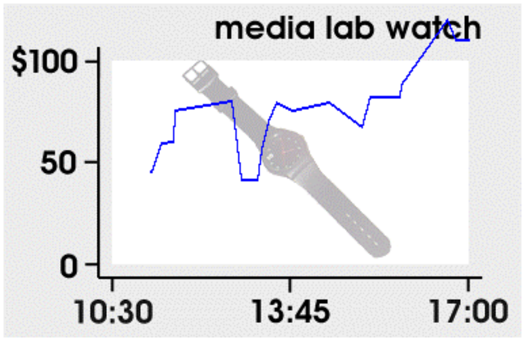


Figure 11: Price of watches over the day

Two other incidents happened which can be identified in Figure 11. At some point in the middle of the day, one of the participants spread a rumor saying that at the end of the day, every person present would receive a free Media Lab watch. As a result the price of watches plummeted because people hurried to change the prices of their watch-buying agents. When it was pointed out later that this news was false, the price of watches picked up again. Another phenomenon that took place is that the prices of goods generally rose towards the end of the day. This is the case because at the end of the day, people could only buy wine if they had 70 bits. Anyone owning less than 70 bits was happy to spend all their money on whatever good they could buy, because the money became useless at the end of the experiment.

4.2 Qualitative Analysis

Looked at from an economics and game theory perspective, the concept of our agent marketplace transactions follow a "continuous double auction" similar to classified ads in newspapers and on various Web sites. Characteristics of continuous double auctions include:

- sellers (and therefore goods) continuously enter and leave the marketplace
- buyers continuously enter and leave the marketplace
- sales occur whenever the buyer's price is greater than or equal to the seller's price

Strategic analysis of this type of auction is lacking in the economics and game theory literature relative to other auction types [Friedman93]. Our future research will explore potential strategies for this market model, including how dependencies among auctions influence them. This will help us create more sophisticated buying and selling strategies than the simple (but effective) ones used in this experiment.

Although the concept of our agent marketplace model is a continuous double auction, the implementation of our agent marketplace had elements of "continuous English auctions" and "continuous Dutch auctions" [Milgrom89]. Buyers continually raising their prices over time is consistent with the protocol of English (first-price, open-cry) auctions and sellers continually lowering their prices over time is consistent with the protocol of Dutch (first-price, sealed-bid) auctions. However, there are no formal auction protocols in our system that require agents to change their price in any fashion (although we will likely explore formal auction protocols in the future). Also, since our future work should bring us closer to a "truer" continuous double auction model (in that third-party, heterogeneous agents may have any arbitrary strategy), the strategies used in English and Dutch auctions (which are generally better understood although not necessarily trivial to analyze and compute) would likely not apply.

Throughout the day, we had several students stationed around the kiosks to assist participants. The feedback observed and reported to these students was quite positive. Most participants had no difficulty logging into the system (by scanning their badge), creating agents, or checking the status of their agents. Except for some people having trouble operating their pagers and overcrowding at the transaction center, the participants found the experiment to be highly compelling to the point where some skipped sessions to be at the kiosks following their agents' every move.

Although the focus of the experiment was in the digital trading realm, we also anticipated the occurrence of black-market trading. This is because we see the electronic agent marketplaces as complementing rather than competing with existing means of transactions. In general, we see agent marketplaces as helping reduce transaction costs associated with certain types of transactions (e.g., classified ads) where financial, time, and trust issues can impede negotiations and commerce. Many black-market trades were observed throughout the day.

Another anticipated behavior we observed was participants buying more than they had money to spend. Interviewing participants revealed several reasons for this behavior. A few people misunderstood the negotiation tactics of their agents. Others were not able to sell goods fast enough to cover purchase costs. Some just assumed credit was available or they didn't expect any negative consequences for their actions (afterall, the money was "play money"). While some of these reasons are user interface issues, the underlying issue involves the inherent discrepancy between how much money the owner has and how much money the owner's agents "think" they can offer. This problem has a number of potential solutions.

One solution could be to let agents know how much money an owner has to spend and ensure that collectively they don't make bids greater than that amount. This would require greater inter-agent communication and collaboration than currently exists in the system. While not a foolproof solution (because the user could spend some money in real-life as well), it could be useful to help place price caps on total agent spending. Another solution may allow agents to directly access an owner's financial account(s) (e.g., credit cards or digital cash). In this scenario, no mapping of funds between the real and digital worlds would be necessary since agents would be transacting with actual currency. Obviously, issues of trust, security, and price caps are still relevant here.

A more interesting solution to the problem of buying more than one has money to spend could involve a collaborative way for an owner's agents to negotiate. For example, a user may specify conditions on when an agent or group of agents should buy or sell a good. Perhaps an owner is interested in buying one of three goods and doesn't care much which good is purchased as long as one of the three goods is bought and the price paid is reasonable. In this scenario, an owner could create three agents in a mutually-exclusive relationship where once one of the agents makes a deal, the other two stop negotiating (until instructed otherwise). This addition to the system was considered but left out of the actual system deployed in the experiment because we assumed it would take too much time to teach the participants about instructing agents.

5. Related Work

There are many Web sites where people can go to find, buy, and sell goods [BF] [Fido] [OnSale] [UCE]. However, none of these sites has the notion of an autonomous agent which will actually do the work of *negotiating* to find the best possible deal on your behalf.

The closest system we have seen to ours is that of Tsvetovatyy and Gini [Tsvetovatyy96], although there are significant differences. The main difference is that they attempt to more completely model the real world, representing things such as banks, accounts, money, etc. Our approach is to keep things as simple as possible, and focus primarily on the negotiation aspects of transactions. We believe this allows our system to be integrated into the already existing real world of buying and selling more easily.

Autonomous or intelligent agents have been actively researched for many years. The overarching goal of this work is to help people deal with “information and work overload” [Maes95]. The notion of autonomous agents is not a new one. It appears extensively throughout computer science literature, in several different contexts. In the field of Distributed AI, agents are entities which collaborate to solve a specific problem [Demazeau90]. In Decentralized AI, the focus is more on the interactions of agents with different motivations. The underlying notion, though, is still that the agent interaction should further some organizational goals [Demazeau90].

Agents are often seen as a general technique for solving problems, be they very specific (planning the path of a robot) or broader (managing resources). This notion of agents is somewhat different from the one we take, which is more task-oriented. Also, our agents not only don't share common goals, they have diametrically opposing aims. This contrasts to a system such as Firefly [Shardanand95], where agents serve individual users (to make music recommendations), yet cooperate and exchange information in a mutually beneficial fashion.

There has also been a lot of work done on market-based systems [Clearwater96]. The typical approach is to model an optimization problem, such as resource allocation, as a marketplace consisting of multiple agents, each trying to maximize their own “utility”. Notice how this differs from the traditional AI approach mentioned above where the agents are cooperating. Here the agents are not cooperating but acting selfishly. This can be a powerful and effective technique when dealing with difficult optimization problems. A good example of a market-based system is the Challenger system for allocating processing resources within a network of workstations [Chavez97]. For two earlier market-based agent approaches to resource allocation and equilibrium problems, see Enterprise [Malone88] and WALRAS [Cheng]. Much of this market-based research has stemmed from earlier work in economics and game theory [Rosenschein94].

Our agent marketplace is in a sense very similar to a market-based system, in that we are trying to efficiently and optimally match buyers and sellers. Unlike most market-based systems, though, which exist strictly in a virtual world, our system is intended to be used by real people. Because of this human factor, a lot of issues have to be considered that do not in more “artificial” systems.

Finally, a multiple agent system implies agent communication. This is also an area which has been extensively researched. KQML is perhaps the most notable attempt to design a general purpose agent language [Labrou94]. We chose not to use KQML, since our agents are all locally built and thus can be made to communicate via a predefined set of methods.

More advanced agents might require a richer semantics of communication to enable more complex and subtle negotiations. The field of speech acts has investigated such theoretical issues in depth [Winograd86].

6. Current Work

We received a lot of useful feedback from our first agent marketplace experiment. Our current efforts include using this feedback (along with our own analysis) to create an agent marketplace for the MIT-wide community. We will initially allow MIT members and sponsors to trade books, music, and Magic Cards. We intend to have this system up and running by early 1997. This experiment will run for several months.

Unlike the first experiment, we will not have a transaction center for one party to complete their half of a transaction if the other party doesn't show up to complete the deal. Having parties renege on deals made by their agents can lead to many unhappy people. To provide a means for recourse for this likely scenario, we are implementing a limited "Better Business Bureau" feedback mechanism where users can "report" untrustworthy people. The logistics of this facility is judiciously being worked out now; however, one use of this information is to disallow one's agents from negotiating with anyone's agents who are under a user-specified trust level.

Depending on the feedback we receive from this new MIT-wide agent marketplace experiment, we will add more types of goods with a means for users to create their own good templates. These will be used for defining goods that the system has not yet seen. We are also exploring protocols and semantics for defining more extensible agent marketplace communications and are researching several alternative protocols [Labrou94] [Telescript96]. The chosen protocol will be used to support the distribution of agents so they can either execute on the marketplace server or on a user's local machine. This marketplace protocol will be "open" to allow third-party agents (with their own unique strategies) to participate in the agent marketplace. For example, we envision developers creating sophisticated commerce agents that require a non-trivial amount of resources to complete its market analysis. Such agents could potentially be run more efficiently on a user's local machine and communicate / negotiate with other agents via the open marketplace protocol. Other technologies we're looking at include electronic cash/credit and all-digital transactions involving online information.

At the same time that we are building an architecture to support distributed and third-party agents, we are also exploring how agents can negotiate better in the marketplace on their users' behalf. To do this, we are beginning to merge microeconomics theories with AI techniques such as collaborative filtering and machine learning. Ultimately, agents can be created that make better deals than their owners without the owner needing to intervene. We anticipate that our agent marketplace combined with sophisticated agent strategies will offer a more compelling consumer experience over today's online auctions and marketplaces.

7. What We Learned

We created an electronic marketplace where agents buy and sell goods and negotiate with interested parties on behalf of their users. In general, we see such agent marketplaces as helping reduce transaction costs associated with certain types of transactions (end-consumer to end-consumer transactions) where currently financial, time, and trust issues can impede negotiations and commerce. User testing of such agents has revealed that the level of intelligence or sophistication which a buying/selling agent could ultimately demonstrate is limited more by user-agent "trust" issues than by limitations of AI technology. In particular, in order for these agents to be widely accepted, it is crucial that the agent's behavior can easily be understood and controlled by the user.

The experiment raises a lot of issues about the implications of this type of marketplace for our society and economy. While the experiment was limited to a very small set of goods, one could envision setting up similar marketplaces for all consumer goods as well as for services. It is still an open issue what the effects of such wide deployment would be. One could argue that this type of market will approach the concept of an ideal market more so than our current markets resulting in a more efficient and effective economy. Another potential effect may be that more types of goods will be bought and sold because it is easier in this kind of system to find a buyer/seller for a good, even if it's a good that very few parties might be interested in. For example, we hope that the wider deployment of this technology will encourage people to buy and sell more second-hand goods, thereby supporting the reuse of objects in our society. Finally, it is an open issue what happens to current brokers in this future scenario, since an important role of brokers, namely to bring buyers and sellers together, is automated by this agent system.

8. Acknowledgments

Many people contributed to the success of our multi-agent electronic marketplace experiment. These include Kathi Blocher, Debbie Cohen, Judith Donath, Henry Holtzman, May-Li Khoe, Yishai Lerner, Andy Lippman, Carl Malamud, Nelson Minar, Karen Modrak, and Greg Tucker.

9. References

[BF] BargainFinder. http://bf.cstar.ac.com/bf/

[Bradshaw97] J. Bradshaw. *Software Agents.* MIT Press, Cambridge, MA. (Upcoming March 1997)

[Chavez96] A. Chavez and P. Maes. "Kasbah: An Agent Marketplace for Buying and Selling Goods." *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM'96)*, pp. 75-90. London, UK, April 1996.

[Chavez97] A. Chavez, A. Moukas, and P. Maes. "Challenger: A Multi-agent System for Distributed Resource Allocation." *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*. Marina del Ray, California, February 1997.

[Cheng] J. Cheng and M. Wellman. "The WALRAS Algorithm: A Convergent Distributed Implementation of General Equilibrium Outcomes." Submitted for publication.

[Clearwater96] S. Clearwater. *Market-Based Control: A Paradigm for Distributed Resource Allocation*. World Scientific Publishing, Singapore, 1996.

[Demazeau90] J. Demazeau and J. Muller. *Decentralized Artificial Intelligence*. Elsevier Science Publishers, North Holland, 1990.

[Fido] FIDO: the Shopping Doggie! http://www.shopfido.com/

[Foner96] Foner, Lenny, "A Multi-Agent Referral System for Matchmaking", Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, pp. 245-261, London, UK, April 1996.

[Friedman93] D. Friedman and J. Rust, editors. *The Double Auction Market: Institutions, Theories, and Evidence.* Addison-Wesley, New York, 1993.

[Labrou94] Y. Labrou and T. Finin. "A Semantics Approach For KQML: A General Purpose Communication Language for Software Agents." *Proceedings of CIKM'94*, New York, 1994.

[Maes95] P. Maes. "Intelligent Software." *Scientific American*, Vol. 273, No. 3, pp. 84-86. Scientific American, Inc., September 1995.

[Malone88] T. Malone, R. Fikes, K. Grant, and M. Howard. "Enterprise: A Market-like task scheduler for distributed computing environments." In B. Huberman, editor, *The Ecology of Computation*, pp. 177-205. North-Holland Publishing Company, Amsterdam, 1988.

[Milgrom89] P. Milgrom. "Auctions and Bidding: A Primer." *Journal of Economic Perspectives*, pp. 3-22. Summer 1989.

[OnSale] OnSale: Live Online Auction House. http://www.onsale.com/

[Rosenschein94] J. Rosenschein and G. Zlotkin. *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers*. MIT Press, Cambridge, MA, 1994.

[PAAM96] Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, London, UK, April 1996.

[Shardanand95] U. Shardanand and P. Maes. "Social Information Filtering: Algorithms for Automating Word of Mouth." *Proceedings of Computer-Human Interaction '95 (CHI'95)*. Denver, CO, 1995.

[Telescript96] "Telescript Technology: The Foundation for the Electronic Marketplace." http://www.genmagic.com/Telescript/Whitepapers/wp1/whitepaper-1.html

[Tsvetovatyy96] M. Tsvetovatyy and M. Gini. Toward a Virtual Marketplace: Architectures and Strategies. *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM'96)*, pp. 597-613. London, UK, April 1996.

[Winograd86] T. Winograd and F. Flores. *Understanding computers and cognition: A New Foundation for Design*. Addison-Wesley, Reading, MA, 1986.

[UCE] United Computer Exchange. http://www.uce.com/