A Market Architecture for Multi-Agent Contracting

John Collins, Ben Youngdahl, Scott Jamison,
Bamshad Mobasher, Maria Gini

Department of Computer Science and Engineering
University of Minnesota

Abstract

We present a testbed for multi-agent negotiation, implemented as a generalized market architecture called MAGNET (Multi AGent NEgotiation Testbed). In contrast with other approaches to multi-agent negotiation, we introduce an explicit intermediary into the negotiation process. We show how this approach helps in controlling fraud and discouraging counterspeculation. MAGNET provides support for a variety of types of transactions, from simple buying and selling of goods and services to complex multi-agent contract negotiations. It is organized around three basic components: the exchange, the market, and the session. We also present a negotiation protocol for planning by contracting that takes advantage of the services of the market. Finally, we describe the open, extensible implementation of MAGNET.

1 Introduction

In recent years, many researchers and practitioners have focused on the design of market architectures for electronic commerce, while others have focused on protocols governing the interaction of self-interested agents engaged in such transactions. However, existing architectures for multi-agent virtual markets typically leave the actual negotiation to the agents themselves, lacking explicit facilities and infrastructure for handling multiple and varied negotiation protocols. We believe there is advantage to be gained by providing a market-based intermediary for inter-agent negotiations.

Our goal in this research is to design, implement, and analyze a generalized multi-agent market architecture that can provide explicit and integrated support for complex agent interactions, such as in automated contracting, as well as other types of negotiation protocols, including sealed-bid auctions and open-bid or advertised-price buying and selling.

We show how the existence of an independent market infrastructure can add value and practicality to contracting protocols, by providing protection against fraud and misrepresentation, and by curtailing unproductive value-based or time-based counterspeculation by participating agents. We also introduce a flexible contracting protocol which can take full advantage of the proposed market architecture to facilitate agent interactions.

Proc. 2nd Int'l Conf on Autonomous Agents, Minneapolis, MN, May 1998.

Automated contracting protocols that have been developed recently generally assume direct agent-to-agent negotiation. For example, Smith [14] pioneered research in communication among cooperating distributed agents with the Contract Net protocol. The Contract Net has been extended by Sandholm and Lesser [11] to self-interested agents. In these systems, agents communicate and negotiate directly with each other. In Sandholm's TRACONET system [13] both the bidding and contract execution mechanisms are complicated by the need to operate in an environment where agents cannot trust each other. They do not assume or take advantage of an independent market infrastructure that can act as a trusted intermediary and affect the timing and/or functionality of the protocol elements.

To the extent that we require the existence of an external market mechanism as an intermediary, our proposed framework is similar to that of Wellman's market-oriented programming used in AuctionBot [20] and The University of Michigan Digital Library [19]. AuctionBot, for instance, supports a variety of auction types each imposing a set of market rules on the agent interactions. Hence, the auctions, themselves, become the intermediaries. The entity that sets up the auction can specify certain parameters for the auctions. In contrast, our framework provides explicit market mechanisms which can not only specify and enforce auction parameters, but also support more complex interactions such as automated contracting in a variety of types of markets based on a market ontology. Furthermore, these market mechanisms also enforce general market rules and "social laws", such as government regulations, by which all participants must abide.

Rosenschein and Zlotkin [10] analyze a variety of domains and propose a classification of problems into domains that are characterized by different types of negotiation among agents. They show that the behavior of multiple, interacting agents can be influenced by the set of rules (the protocol) that the system designers choose for the agents' environment. The purpose of these rules is to allow the agents to make constructive agreements. Their analysis assumes that the negotiating agents have similar capabilities. The protocol we present in this paper does not require that assumption.

Mechanisms to reduce counterspeculation, such as the Clarke tax mechanism [2] or the Vickrey auction [17] have been proposed for automated negotiation of self-interested agents. The architecture we present can support the Vickrey auction, and eliminates one of its limitations by providing a structure that can act as a trusted auctioneer [12].

A variety of architectures have been proposed for single and multiple agents in different domains (see, for instance, [3, 5, 7, 9]). MAGMA [16], an open architecture for agents interested in buying and selling, supports both manual and automated negotiation with a limited form of Vickrey auction. Even though MAGMA already includes many of the features of the architecture we present here, MAGMA is intended for a more limited domain.

Substantial work is underway in standardizing an open architecture for electronic commerce [15, 8]. Our architecture improves on these proposals by adding support for more complex negotiation protocols. Our architecture could be implemented by extending the framework presented in [15].

This paper is organized as follows: in Section 2 we identify the requirements for a generalized multi-agent market architecture that can support complex agent interactions, and we present a novel architecture that satisfies these requirements. In section 3, we show how this architecture can be applied to a multi-agent contracting domain, and we present a flexible contracting protocol that can take full advantage of our proposed architecture. Section 4 discusses the implementation with an example. Finally, in section 5, we conclude and discuss further research opportunities.

2 A Generalized Multi-Agent Market Architecture

The MAGNET architecture is a distributed set of objects that can support electronic commerce in a variety of domains, from the simple buying and selling of goods to situations that require complex multi-agent negotiation and contracting.

We begin by discussing a set of requirements that this architecture is intended to address, and then turn to a description of our proposal.

2.1 Requirements for a Market Architecture

The architecture should provide a framework for secure and reliable commerce among self-interested agents. Existing architectures and proposals provide such a framework [15, 8].

We focus on three areas of requirements that have not been fully addressed in existing proposals, and that will be required to support the contract negotiations needed to formulate and carry out plans in a virtual organization. Support for complex multi-agent negotiation

Support for Complex Multi-agent Negotiation

Protocols that require agents to do complex marginal cost computations [13, 14], or plan generation and composition [1, 10], may require extended periods of time to complete, during which a context must be maintained. The time during which the negotiated transaction extends can also span significant periods of time, at least in the range of weeks to months. In order to support this, the market infrastructure must maintain the state of each transaction over time.

This is a prerequisite for many other functions we expect the market to perform, and it makes the system more robust in the face of hardware and communication failures. Most negotiation protocols involve time limits, such as a deadline for receipt of bids. All parties to a time-sensitive negotiation process must have a common time reference. We expect the market architecture to provide this. The architecture must also have the ability to validate non-performance and assess negotiated decommitment penalties.

Protection against fraud and misrepresentation

We must assume that participating agents will take advantage of any opportunities that exist in the design of the market to gain advantage. The structure of the market must recognize and protect against situations that allow agents to gain unfair advantage at the expense of other agents. Strategies that can result in this type of "unfair" gain, and related architectural requirements, include:

- Hiding one's identity or taking on the identity of another.
- Dishonest auctioneer - In Vickrey-type auctions [18, 17], the motivation for truth-telling on the part of participants is predicated on their belief in the honesty of the auctioneer.
- Miscommunication of the rules under which an auction is being conducted.
- Failure to follow through on commitments.

Discouragement of counterspeculation

Opportunities for counterspeculation arise when the rules of negotiation allow agents to gain advantage by making use of factors other than their own capabilities and valuations, such as their estimates of the capabilities and valuations of the customers or other suppliers [6]. We are concerned with two general types of counterspeculation. Value-based counterspeculation [10, 13, 18, 17] occurs when agents use their own estimates of each other's valuations to set bid prices. In [1], we identified two classes of time-based counterspeculation opportunities in a contracting domain that can be controlled by the settings of certain timing parameters. One of these situations occurs when supplier agents are allowed to expire their bids before the customer's call-for-bids expires, and the other situation occurs when customers are perceived by suppliers to be considering bids and formulating plans before bidding is closed.

2.2 Architectural Elements

We now turn to the architectural design of a market structure that will satisfy these requirements. The fundamental elements of this architecture are the *exchange*, the *market*, and the *market session*. A primary motivation for this architecture, and especially the session construct, is to support *Planning by Contracting* [1], an activity in which an agent, in order to formulate its plans and fulfill its goals, must contract with other self-interested agents for all or part of the necessary tasks.

The Exchange

An *exchange* is a collection of domain-specific markets in which goods and services are traded, along with some generic services required by all markets, such as verifying identities of participants in a transaction, or a Better Business Bureau that can provide information about the reliability of other agents based on past performance. Architecturally, an exchange is a network-accessible resource that supports a set of markets and common services, as depicted in Figure 1.

Markets

Each *market* within an exchange is a forum for commerce in a particular commodity or business area. There would be markets devoted to banking, publishing and printing, construction, transportation, industrial equipment, etc. Each

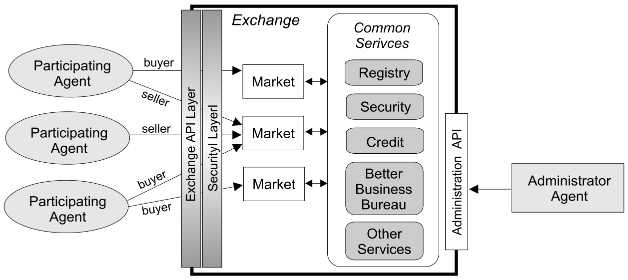

Figure 1: The Structure of an Exchange

market includes a set of domain-specific services and facilities, as shown in Figure 2, and each market draws upon the common services of the exchange.

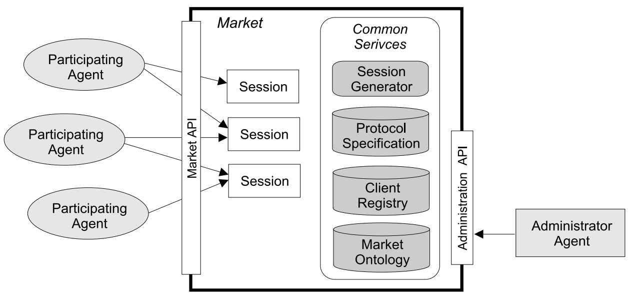

Figure 2: The Structure of a Market within the Exchange

An important component of each market is a set of current *market sessions* in which the actual agent interactions occur. Agents participating in a market may do so as either session initiators, or as clients, or both. As detailed in the next section, each session is initiated by a single agent for a particular purpose, and in general multiple agents may join an existing session as clients. Important elements of the market include:

- An *Ontology* that is specific to the domain of the market, specifying the terms of discourse within that domain. In a commodity-oriented domain, it would include terms for the products or services within the domain, as well as terminology for quality, quantity, features, terms and conditions of business, etc. In a planning-oriented domain, specifications of services would be in a form that supports planning.
- A *Protocol Specification* that formalizes the types of negotiation supported within the market. Within a planning-oriented market domain, these specifications would be limits on parameters of the negotiation protocol outlined in Section 3, such as the maximum decommitment protocol, whether bids can be awarded before the bid deadline, etc.
- A *Registry* of market clients who have expressed interest in doing business in the market. Entries in this registry would include the identity of a client, a catalog [4] (or a method for accessing a catalog) of that client's interests, products or capabilities, which can be used to locate clients to meet requests for new session participants, and a client agent that is empowered to negotiate contracts on behalf of the supplier. Client catalogs are required to express their interests and offerings in terms of the market's ontology.

Market Sessions

A *market session* (or simply a session) is the vehicle through which market services are delivered dynamically to participating agents. It serves as an encapsulation for a transaction in the market, as well as a persistent repository for the current state of the transaction.

We have chosen the term "session" to emphasize the temporally extended nature of many of these interactions. For example, in a contracting market, if an agent wishes to build a new house, it initiates a session and issues a call-for-bids.

The session extends from the initial call-for-bids through the negotiation, awards, construction work, paying of bills, and final closing. In other words, the session encloses the full life of a contract or a set of related contracts. The session mechanism ensures continuity of partially-completed transactions, protects against fraud, limits counterspeculation, and relieves the participating agents from having to keep track of detailed negotiation status themselves.

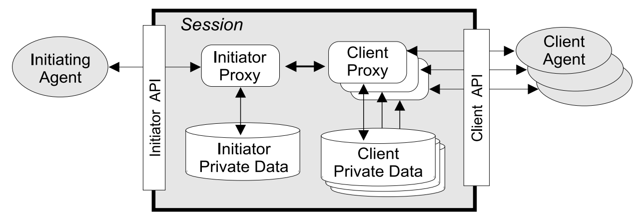

Figure 3: The Structure of a Market Session

Agents can play two different roles with respect to any session. The agent who initiates a session is known as the *session initiator*, while other participating agents are known as *session clients*. A session can be initiated either for the purpose of buying or selling, depending on the type of market. In the above example of building a house, the initiating agent was the buyer or customer, and the other participants would be sellers or suppliers, whether they were supplying materials, labor, advice, credit, or other services. A session could also be initiated to sell items or services at auction.

At any given time, a session can be *open* to new participants, or *closed*. A public auction would typically be open to new participants, while the house-building session described above would be closed once the contracts were let. The market maintains a list of open sessions which may be accessed by participating agents.

Figure 3 shows the structure of a session. Two APIs are exposed, one for the session initiator and one for session clients.

Each session contains an Initiator Proxy that implements the Initiator API and persistently stores the current state of the session from the standpoint of the initiator.

A Client Proxy is provided for each client that similarly provides a Client API to the client agent, and persistently stores the current state of the session from the standpoint of the client. Proxies are market entities that act on behalf of the agents and enforce market rules.

There are two reasons for the existence of the proxy components. The first is related to security: client proxy components cannot see the private data of the initiator or of other clients. The second is that in a distributed system environment, the processing and persistent data elements of the initiator and clients would presumably be at different locations in the network to maximize performance.

3 A Contracting Market Structure

A primary motivation for the session construct is to support *Planning by Contracting* [1], an activity in which an agent, in order to formulate its plans and fulfill its goals, must contract with other self-interested agents for all or part of the necessary tasks.

3.1 Protocol for Planning by Contracting

The Planning by Contracting protocol is a three step process which begins after the session has been initiated by a customer agent: the customer issues a call-for-bids, suppliers reply with bids, and the customer accepts the bids it chooses with bid-accept messages. We have avoided the need for open-ended negotiation by means of bid break downs and a time-based decommitment penalty as described below.

Call for bids

Once the customer has developed a plan of subtasks chosen from the market's ontology, it will send a call-for-bids message. The call-for-bids message will include, for each subtask listed, a time window during which the work must be done. The call-for-bids message will also include, among other information:

1. a bid deadline, or the time by which the suppliers must respond with bids,
2. the time at which the customer will begin considering the bids,
3. the earliest time at which bid acceptances will be sent,
4. penalty functions for each subtask, which will be assessed against the supplier if the supplier commits to work, but fails (or decides not) to do it. These penalty functions are piecewise-linear functions of time that are intended to encourage suppliers to perform the work they commit to. If a supplier is unable to perform, the increasing value of the penalty function encourages it to explicitly decommit as early as possible.

This call-for-bids message, once created, is passed to the market session, which makes it available to all of the appropriate suppliers (those who are registered with the market, and are able to perform the necessary tasks.) In this sense, the call-for-bids message is public, while all of the remaining messages are private. Before forwarding it, the market session may check the message to make sure that it conforms to all market and exchange rules which may exist.

Bidding

Each supplier will inspect the call-for-bids, and will decide whether or not it should respond with a bid, according to its resources, time constraints, and knowledge of the work to be done, according to the catalog of services provided by the market agent.

If it chooses to respond, it will send a bid message, which will be private (i.e. other suppliers will not see the contents of the bid). This bid message can include a combination of subtasks, which must be a subset of the subtasks listed in the call-for-bids. The content and number of bid messages will be monitored and may be recorded by the market session, before they are validated and forwarded to the customer.

In the bid, the supplier must indicate the cost (to the customer), the time window, and the estimated duration of the work for the whole subtask combination, and this same data for each of the separate subtasks (please see the explanation for this in the next section.) The bid-accept deadline must also be included, as well as a penalty function for each subtask which the customer will have to pay if it commits to giving this supplier the work but then decides to decommit. This penalty function will have the same structure as the supplier penalty function.

Each supplier can send multiple bids for each call-for-bids, each including different costs and time windows, but each supplier will be awarded only one bid combination (or part of one). This is to enable the supplier to send many bids, but not over commit itself.

This bid is a commitment by the supplier to do work listed in the bid, should the customer accept it. If the supplier sends no bid message before the customer's bid deadline, the customer will assume that the supplier has decided not to send a bid for this particular call for bids. Thus, rejection is passive.

Bid acceptance

Having received the bids, the customer must decide which of the bids to accept, using knowledge about the bids, the task and subtask values, its own time constraints and the bidder (perhaps provided by the market agent). After completing this process, the customer must decide to do one of three things for each bid that it has received:

1. accept the whole bid,
2. accept a subset of the subtasks in the bid, or
3. reject the bid (again, this is done passively).

The motivation for these choices is to make open-ended negotiation unnecessary. If no acceptable set of bids together would cover every subtask to the satisfaction of the customer, then the customer can avoid negotiation because it knows how the supplier will break down the costs of the accepted subtasks, should it become necessary for the customer to accept a subset of the original bid combination.

This scheme in conjunction with the time-based decommitment penalty functions makes it possible to avoid open-ended negotiation without loss of generality.

The bid-accept message will be sent through the market session, which will verify, validate and time-stamp it before forwarding it to the customer. Note that either of the first two choices are commitments to give the supplier the work and at the point in time that this message is sent (according to the market session's time stamp), both the supplier and the customer penalty functions will be set into effect.

A failure to send a bid-accept message means the customer is rejecting the supplier's bid.

Once commitments have been made, an agent may determine that it cannot do the tasks it has committed to, or that it would disadvantageous to do so. In these situations, the agent must send a decommitment message to the other agent, describing what parts of its commitment it will not be satisfying. Included in the decommitment messages will be an acknowledgment of the penalty that the agent will be paying as a result of the decommitment.

3.2 The Role of the Market in Planning by Contracting

We present a brief outline of the activities of customer and supplier agents engaged in planning by contracting:

1. Using the ontology provided by the market, a customer agent develops a partial plan, and estimates tentative values for plan components based on the goal value and the "criticality" of each component.

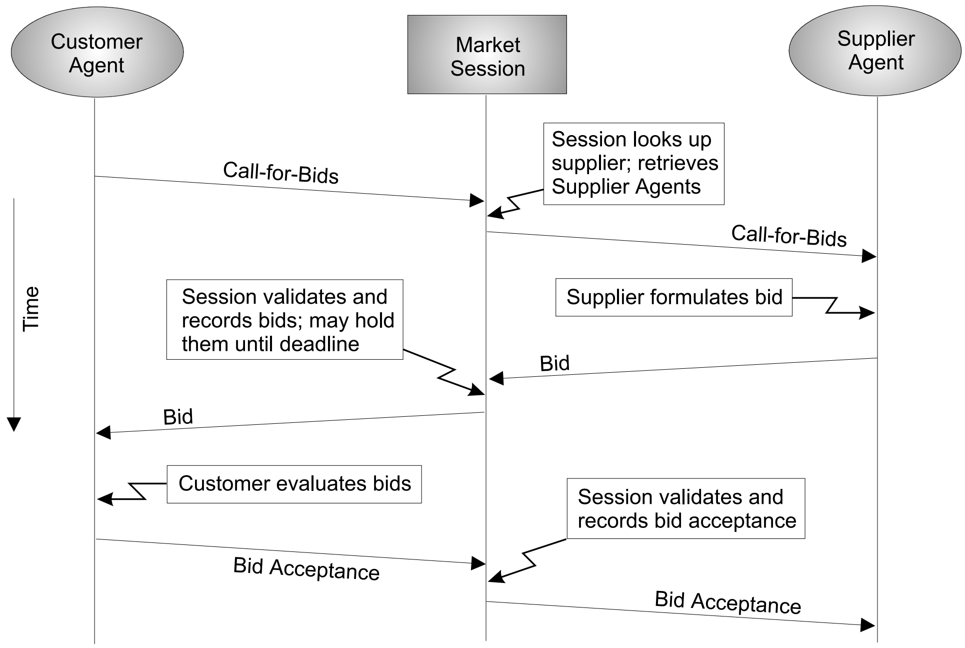

Figure 4: A Typical Session-Mediated Negotiation

2. The customer initiates a session in the market.
3. The customer announces, through the session, tasks to be bid upon in a call-for-bids.
4. Suppliers join the session as clients, either because they have been notified by the market or because they have queried the market and found a session of interest.
5. Suppliers evaluate the call-for-bids, decide which subtasks to bid upon, and return bids through the session.
6. The customer receives bids from suppliers.
7. For each bid, the customer maps the bid to the plan and evaluates bid price vs. value. The value calculation could include time limits, decommitment penalty, task coverage, location of work, identity of bidder, etc.
8. The customer extends or modifies the plan, and may announce further bids.
9. The customer awards contracts to selected suppliers.
10. The customer monitors execution of the plan, making necessary adjustments as suppliers perform or fail to perform their contracted tasks.
11. Customer and suppliers finalize financial settlements and the session is terminated.

Within our market architecture, these activities are encapsulated in a market session. Once a session is obtained, the customer is able to issue a call-for-bids and conduct other business. The interactions involved in the basic bidding cycle among the customer, supplier, and market session are illustrated in Figure4. The customer's call-for-bids is passed to the session rather than directly to the supplier agents, and all interaction passes through the session. This allows the session to perform several important services:

- The customer need not search the market to find suppliers who might be interested in bidding. The session does this by searching the market's registry to find suppliers for which the intersection of services or goods requested in the call-for-bids and the advertised goods and services is non-empty. The customer can override this with its own list, or it may retrieve the list from the market ahead of time and prune it. If the customer supplies the list, it has the option of specifying that the session be closed to other suppliers.
- The customer need not find a way to describe the services that it requires. It can use the catalog of services (and its syntax) which the market has listed in the ontology.
- The session time-stamps all interactions involved in the transaction, in order to avoid dispute among customers and suppliers over performance with respect to deadlines.
- The session truthfully informs the suppliers of the conditions under which the bidding is being conducted, and then enforces those conditions. This eliminates the possibility of the customer misinforming the suppliers of bid conditions, and therefore unfairly affecting their pricing strategies.
- The session can limit the number of bids sent by each supplier to avoid overwhelming the customer.
- If the customer desires or the market allows, the session may provide the customer with information about the suppliers for use in choosing which bids to accept.
- The session enforces the "rules of the market" with respect to deadlines, penalties, disclosure of identity, and auction rules. For example, if a sealed-bid auction is advertised, the session will hold bids back from the

customer until the bid deadline has passed. This eliminates the possibility of temporal counterspeculation as identified in [1].

- If the customer desires or the market requires, the session can act as a "trusted auctioneer" in a Vickrey auction.

4 Implementation

The MAGNET system is implemented in Java as a set of CORBA services. Since it is designed as an environment for automated agents, its user interface is somewhat primitive. In order to permit direct viewing of the operation of the market, we have constructed a human interface applet called Incognito that allows humans to masquerade as agents by making objects and their behaviors visible.

The three major components of MAGNET, the exchange, market, and session, are exposed via interfaces defined in CORBA IDL. At each level of the MAGNET structure, there are a set of objects visible, some of which are primitive attributes and some of which are composite objects. Selecting an object in the left-hand pane makes certain of its behaviors available in the right-hand pane. In general, the set of behaviors presented depends on the identity and role of the participant; for example, customer agents will see different behaviors than supplier agents will see.

Using the CORBA trader service, an exchange provides a "yellow pages" of "exchange approved" services and markets. An agent might request a recommended better business bureau, an accepted credit source, or a market related to a specific ontology and contracting protocol.

The market and session components of MAGNET maintain private, authoritative models of their respective domains. Contracting agents accomplish their goals by perceiving and manipulating these models within the rules imposed by MAGNET. For example, a market may filter who can see specific sessions, as well as how such sessions are manipulated.

Similarly, a session models the state of an ongoing negotiation. It enforces a negotiation protocol by filtering what contracting participants can discern about the state of the session model, in addition to the requests made to manipulate it.

The market and session expose and protect their models via a shared set of interfaces (Perceptible/Tangible). Through these interfaces, sessions and markets provide a filtered set of object references to the agents (perception). These object references act as proxies to the actual elements within the model. A contracting participant may not directly call the methods of the objects behind the proxies, but rather may only access the methods (actions) of the proxy implementations. These proxies, created by the session and market, work in tandem to enforce the consistency of the model by filtering what the contracting participants can perceive and do.

We now describe the implementation using an example. A startup company, Acme Software and Screen Doors, Inc., is releasing version 1.0 of its new killer application. The product manager, who is also the president and chief programmer, has a goal: produce and ship 1000 shrink-wrap packages, including media, manuals, license forms, registration cards, and license key stickers. The old method would be to find a broker or publisher, and hand over the job.

Now assume Acme has an intelligent agent that can act as a broker in the Exchange. The manager gives the goal to

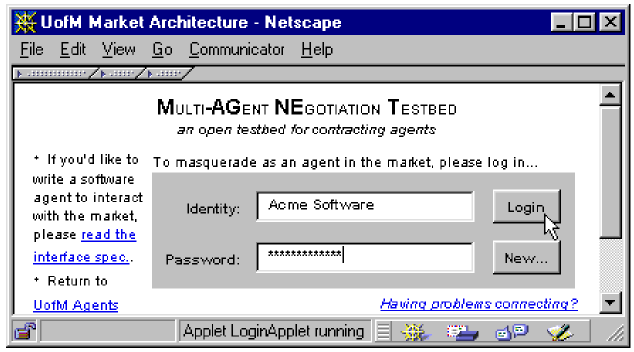

Figure 5: MAGNET Login Screen

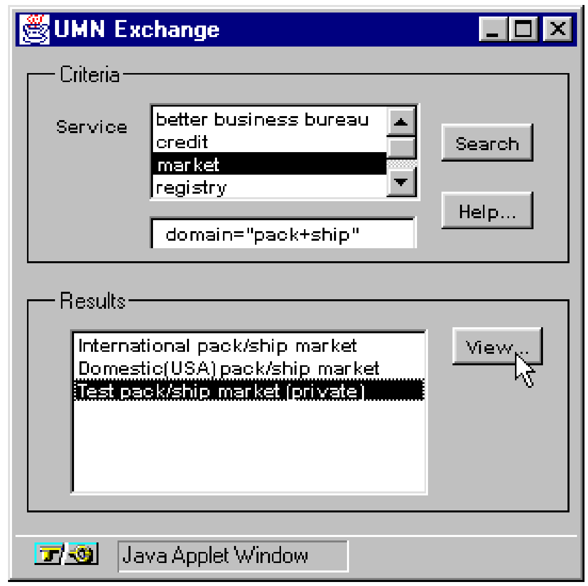

Figure 6: Finding a Market

her automated agent, which enters the exchange to assemble a set of contracts to satisfy the goals. We will assume the agent is already registered with the exchange. Figure 5 shows the login dialog. The agent searches the exchange and finds three markets that provide services to satisfy its goals. One deals with printing services, one with digital media, and one with packaging and shipping. In our example implementation, markets are found using a simple query interface that operates against a CORBA Trader Service, as shown in Figure 6. Using operators obtained from the ontologies in the selected markets, the agent formulates a plan to satisfy its goals. Its plan looks like Figure 8.

Next, the agent establishes the necessary credit with exchange, as specified by the chosen markets, and requests contract negotiation sessions in all 3 markets. The plan is now broken down by market, and the agent submits a call-for-bids to each session. It is acceptable if the submitted subplans overlap. This is because some potential contractors may give lower bids on combinations of multiple tasks than they would on single tasks. Figure 7 shows the customer agent in the process of setting timing parameters for its call-for-bids.

Each session now invites potential contractors who have expressed interest, through the market registry, in the sub-

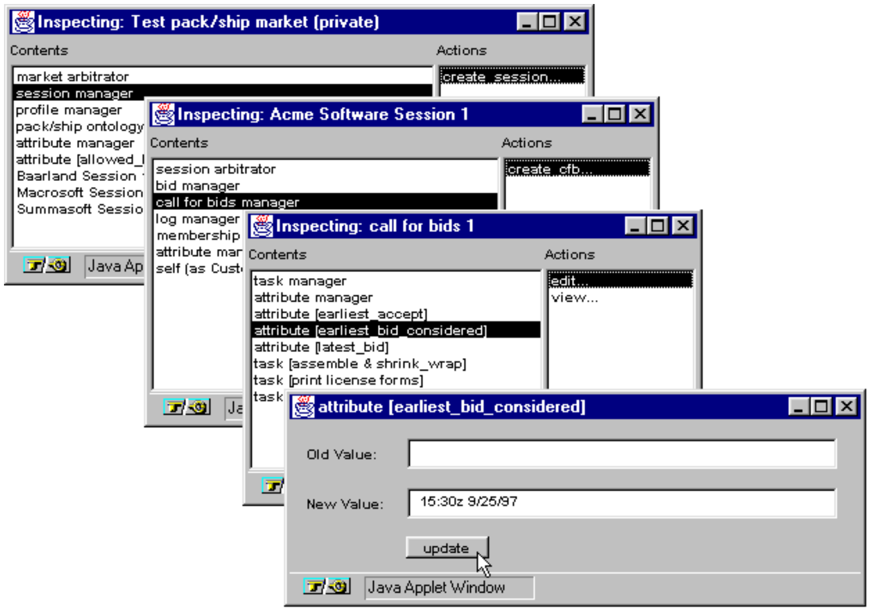

Figure 7: Customer Issues a Call-for-Bids

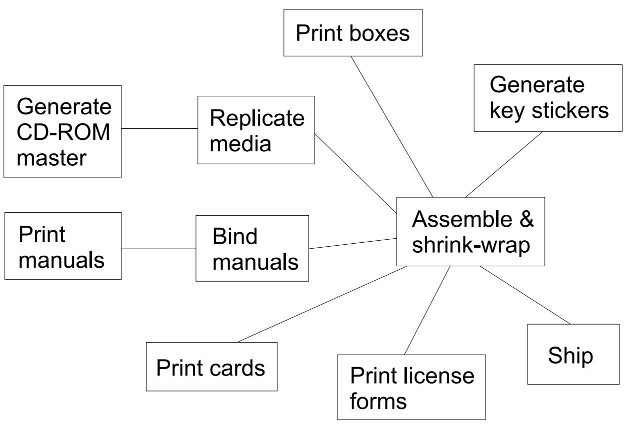

Figure 8: Plan for Release 1.0 Production

ject matter of the bids, to join the sessions as clients. Some of those contractors submit bids. Figure 9 shows a supplier agent in the process of inspecting the call-for-bids in Acme's session.

The customer agent receives the bids, and attempts to assemble them into an optimal feasible plan. Unfortunately, no supplier has bid on printing the license key stickers. At this point, the customer agent must decide whether its own resources can satisfy this need, whether the plan can succeed without the stickers, or whether to risk starting plan execution on the assumption that a supplier for the stickers can be found before plan execution reaches the point where they are needed. It decides (or is told by our manager) that the stickers can be done in-house.

The customer agent now awards the selected bids, and work commences. Part way through the process, the box printer backs out, and pays the required decommitment penalty. Our agent re-opens bidding for box printing, receives and awards a bid. The customer agent monitors task progress, posting payments as agreed at task or subtask completion. Once all transactions are completed, the customer agent requests the respective markets to terminate the sessions.

4.1 Conclusions and Future Work

In this paper we have brought together ideas from recent work in market architectures for electronic commerce, and work in multi-agent contracting protocols. We have presented a generalized market architecture that provides support for a variety of transaction types, from simple buying and selling to complex multi-agent contract negotiations. We have also presented a protocol that takes advantage of the services of the market. Our market architecture is organized around three basic components: the exchange, the market, and the session. We have shown how the existence of an appropriate market infrastructure can add value to a multi-agent contracting protocol by controlling fraud and discouraging counterspeculation.

We are currently exploring the possibility of moving our testbed from CORBA to Voyager, a distributed agent technology from ObjectSpace (www.objectspace.com). Voyager provides many agent-oriented services that are either difficult to find or implement within CORBA. Chief among these are a filterable distributed event service, a persistence mechanism that easily supports agent life cycles, and a facility to migrate agents and objects between machines.

Additionally, we have begun implementing software-agents to interact within MAGNET, and intend to benchmark their performance against human participants as well as each other. We are also exploring how modifying the parameters of the market and session protocols can effect the performance of the system. Eventually, we would like to open our testbed to outside participation over the Internet.

This work raises several interesting questions for future research. A game-theoretic analysis of the protocol could be done to determine optimal strategies for its use by agents. In particular, how should the decommitment penalties be used, and how should proposed decommitment functions be

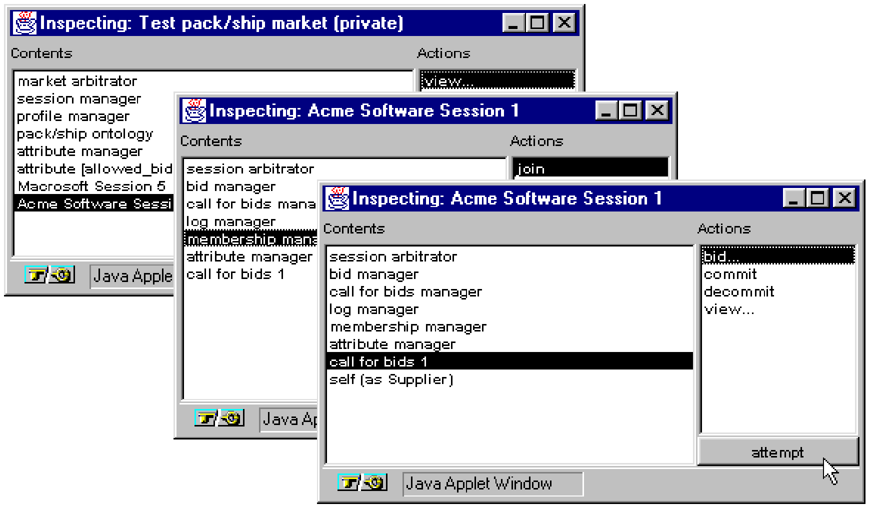

Figure 9: Supplier Inspecting a Call-for-Bids

evaluated when computing marginal costs of plan alternatives. The methods for composition of possibly overlapping bids into feasible or even optimal plans must be worked out.

References

[1] J. Collins, S. Jamison, M. Gini, and B. Mobasher. Temporal strategies in a multi-agent contracting protocol. In *AAAI-97 Workshop on AI in Electronic Commerce*, July 1997.

[2] E. Ephrati and J. Rosenschein. Deriving consensus in multiagent systems. *Artificial Intelligence*, 87:21–74, 1996.

[3] J. Eriksson and N. Finne. Marketspace: an open agent-based market infrastructure. Master's thesis, Uppsala University, Computing Science Department, 1997.

[4] A. M. Keller. Smart catalogs and virtual catalogs. In R. Kalakota and A. Whinston, editors, *Readings in Electronic Commerce*. Addison-Wesley, Reading, Mass., 1996.

[5] D. Kinny and M. Georgeff. Modeling and design of multi-agent systems. In M.J. Wooldridge J.P. Mller and N.R. Jennings, editors, *Intelligent Agents III*, Lecture Notes in Artificial Intelligence. Springer, Berlin, 1997.

[6] D. M. Kreps. *A Course in Microeconomic Theory*. Princeton University Press, 1990.

[7] M. Lejter and T. Dean. A framework for the development of multi-agent architectures. *IEEE Expert*, 1996.

[8] S. McConnell, M. Merz, L. Maesano, and M. Witthaut. An open architecture for electronic commerce. Technical report, Object Management Group, Cambridge, MA, 1997.

[9] J. A. Rodriguez, F. Noriega, C. Sierra, and J. Padget. FM96.5 - a Java-based electronic auction house. In *Second Int'l Conf on The Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM'97)*, London, April 1997.

[10] J. S. Rosenschein and G. Zlotkin. *Rules of Encounter*. MIT Press, Cambridge, MA, 1994.

[11] T. Sandholm and V. Lesser. Issues in automated negotiation and electronic commerce: Extending the contract net framework. In *1st International Conf. on Multiagent Systems*, pages 328–335, San Francisco, 1995.

[12] T. W. Sandholm. Limitations of the Vickrey auction in computational multiagent systems. In *Second Int'l Conf on Multi-Agent Systems)*, Kyoto, Japan, December 1996.

[13] T. W. Sandholm. *Negotiation Among Self-Interested Computationally Limited Agents*. PhD thesis, University of Massachusetts, 1996.

[14] R. G. Smith. The contract net protocol: High level communication and control in a distributed problem solver. *IEEE Trans. on Computers*, 29(12):1104–1113, December 1980.

[15] J. M. Tennenbaum, T. S. Chowdhry, and K. Hughes. eCo System: CommerceNet's architectural framework for internet commerce. Technical report, Object Management Group, Cambridge, MA, 1997.

[16] M. Tsvetovatyy, M. Gini, B. Mobasher, and Z. Wieckowski. MAGMA: An agent-based virtual market for electronic commerce. *Journal of Applied Artificial Intelligence*, (6), 1997.

[17] H. R. Varian. Economic mechanism design for computerized agents. In *USENIX Workshop on Electronic Commerce*, New York, NY, July 1995.

[18] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, 16:8–37, 1961.

[19] M.P. Wellman, W.P. Birmingham, and E.H. Durfee. The digital library as a community of information agents. *IEEE Expert*, 11(3):10–11, June 1996.

[20] P.R. Wurman, M.P. Wellman, and W.E. Walsh. The michigan internet Auctionbot: A configurable auction server for human and software agents. In *Second Int'l Conf. on Autonomous Agents*, May 1998.