JOURNAL OF ORGANIZATIONAL COMPUTING
AND ELECTRONIC COMMERCE *13*(1), 1–29 (2006)

On Automated e-Business Negotiations: Goal, Policy, Strategy, and Plans of Decision and Action

Haifei Li
Department of Computer Science
Union University

Stanley Y. W. Su and Herman Lam
Database Systems Research and Development Center
University of Florida

In recent years, there has been increasing interest in automated e-business negotiations. The automation of negotiation requires a decision model to capture the negotiation knowledge of policymakers and negotiation experts so that the decision-making process can be carried out automatically. Current research on automated e-business negotiations has focused on defining low-level tactics (or negotiation rules) so that automated negotiation systems can carry out automated negotiation processes. These low-level tactics are usually defined from a technical perspective, not from a business perspective. There is a gap between high-level business negotiation goals and low-level tactics. In this article, we distinguish the concepts of negotiation context, negotiation goals, negotiation strategy, and negotiation tactics and introduce a formal decision model to show the relations among these concepts. We show how high-level negotiation goals can be formally mapped to low-level tactics that can be used to affect the behavior of a negotiation system during the negotiation process. In business, a business organization faces different negotiation situations (or contexts) and determines different sets of goals for different negotiation contexts. In our decision model, a business policymaker sets negotiation goals from different perspectives, which are called *goal dimensions.* A *negotiation policy* is a functional mapping from a negotiation context to some quantitative measures (or goal values) for the goal dimensions to express how competitive the policymaker wants to reach that set of goals. A negotiation expert who has the experience and expertise to conduct negotiations would define the negotiation strategies needed for reaching the negotiation goals. Formally, a *negotiation strategy* is a functional mapping from a set of goal values to a set of decision-action rules that implement negotiation tactics. The selected decision-action rules can then be used to control the execution of an automated negotiation system, which conducts a negotiation on behalf of a business organization.

Correspondence should be sent to Haifei Li, Department of Computer Science, Union University, 1050 Union University Drive, Jackson, TN 38305 Email: hli@uu.edu

e-business, automated negotiation, negotiation policy, negotiation strategy, negotiation goal, decision and action rules

1. INTRODUCTION

Negotiation has been widely studied in various disciplines such as diplomacy, psychology, sociology, law, business administration, and computer science [1–7]. Negotiation topics vary from trivial matters, such as the selection of a restaurant for dinner, to important business issues such as a negotiation over a multimillion dollar contract. Decision making is one of the important aspects of a negotiation process. In fact, negotiation researchers often view negotiation as a joint decision-making process [8]. This process can be either aided or fully automated by using some communication and computation tools. In recent years, there have been many attempts to develop computer systems for aiding negotiators' decision making in a negotiation process or automating the process without or with little human intervention. We survey some existing works in the next section. In all the implemented systems that we know of, some low-level decision-action rules (called *tactic and strategic rules* in some literature) are implemented to set the initial offer, make concession, generate counteroffers, determine the acceptance or rejection of a negotiation proposal, and so forth. These rules represent the knowledge and experience of the negotiation expert (a role that can be played by more than one person) on how to conduct negotiation under different conditions and in different situations. They are different from the goals and policies set by high-level business executives who play the role of the policymaker. Negotiation policies reflect the policymaker's insight in business. It is the policymaker's responsibility to set reasonable goals and to define suitable policies for different business conditions and situations. For example, goals and policies set for negotiating with a Fortune 500 company can be quite different from those set for negotiating with a one-person company. After the goals and policies are set, the negotiation expert can then determine what strategies and decision-action rules to apply to implement the policies to achieve the defined goals. The decision-action rules drive the execution of an automated negotiation system, which carries out negotiations on behalf of an organization. To our best knowledge, no existing implemented system attempts to relate these low-level decision-action rules that control the behavior of an automated negotiation system to high-level goals and policies of business organizations. We believe that business negotiations should be driven by business goals and policies; changes in goals and policies should directly impact the tactics or strategies used in negotiations so that the results of negotiations will be consistent with companies' goals and policies.

It is well-known that business negotiation is a very complex process. A human-based negotiation process may involve a lot of human intelligence activities, capabilities and emotions such as the ability to learn, and the discovery of the negotiation partners' negotiation policies and strategies, preference solicitation, wants, needs, ego, patience, and so forth. Some negotiation problems are very complicated such as negotiations on joint ventures, takeovers, government issues, diplomatic issues, and arbitrations. They are beyond the scope of our work and, we believe, beyond the current technologies to implement an automated negotiation system that can imitate human intelligence and deal with these complex negotiation problems. In our work, we have restricted our problem domain to bilateral negotiations over multiple attributes (or issues) of simple products or services. We assume that each product or service can be defined by a finite number of attributes, and each attribute can have a finite number of attribute values. The bargaining process between a pair of negotiation servers, which perform negotiations on behalf its clients, is controlled by a well-defined negotiation protocol. In this article, we introduce a formal decision model that defines the concepts of negotiation contexts, goals, policies, strategies, and decision-action rules and captures their interrelations by showing how one affects the other in a formal way starting from negotiation contexts and goals and leading to the decision-action rules that drive an automated negotiation system. We also describe an implemented negotiation system, which provides a computing infrastructure for testing the utility of the decision model in different business negotiation scenarios. This work has three intended contributions. First, we present a formal decision model to show how the high-level business concepts of negotiation contexts, goals, policies, and strategies/tactics can be formally mapped to low-level decision-action rules that control and affect the behavior of a negotiation system in a negotiation process. Second, by implementing the model and a replicable, automated negotiation server, we offer a very useful computing infrastructure over the Internet for experimenting and testing the effects of changing negotiation policies, strategies, and decision-action rules on the negotiation system and thus on the negotiation results. Third, the computing infrastructure can be used to implement and test various business negotiation scenarios. In our work, we have used it to perform automated negotiations on computer purchases [9–11], and negotiations on part purchase and delivery in a multilink supply chain environment [12]. By conducting various experiments, we are able to verify our decision model and to gain the first-hand experience on the feasibility and flexibility of automated negotiations.

The organization of this article is as follows. We survey some related work in Section 2. In Section 3, we give an overview of a bilateral, multi-issue negotiation and present its protocol. In Section 4, we present the decision model for automated business negotiations. In Section 5, we show how negotiation policies are specified. In Section 6, we present some decision-action rules for different states of the negotiation protocol. In Section 7, we show how negotiation strategies are specified to map negotiation policies to decision-action rules. In Section 8, we present the architecture and implementation of an automated negotiation system. In Section 9, we describe two negotiation scenarios, which are used to evaluate and demonstrate the decision model and the implemented negotiation system. We give a conclusion and a discussion on future work in Section 10.

2. RELATED WORK

There are many interesting works on automated e-business negotiations. In this survey, we focus on the decision-making process in different kinds of negotiation systems.

2.1 Negotiation Support System

A negotiation support system (NSS) [13, 14] is a computer system that assists human negotiators in making decisions in a negotiation process. NSSs are usually based on a phase model of negotiation [13]. In the phase model, a negotiation process is roughly divided into three phases: preparation (or prenegotiation) phase, bargaining phase, and postsettlement phase. In the preparation phase, the system solicits the preferences of the individual users and constructs utilities. The main purpose of the preparation phase is to give the users a better understanding of their real preferences. In the bargaining phase, users exchange structured proposals and/or free style messages. When negotiators have reached an agreement in the bargaining phase, they have the option to enter into the postsettlement phase. The postsettlement phase may use a third-party program to check whether the agreement is Pareto-optimal or not. If it is not, the program can suggest possible Pareto-optimal solutions that are more desirable than the agreement reached by both sides. If there is more than one Pareto-optimal suggestion, and the negotiators have different preferences about these suggestions, they can enter into another round of negotiations.

Lim and Benbasat [15] proposed a theoretical model of NSSs. In that article [15], Lim and Benbasat divided an NSS into two major components: decision aid component (i.e., decision support system [DSS]) and an electronic communications component. Due to the information-processing capability of the decision aid component, solutions with NSS are better than those without NSS in terms of the distance from the Nash solution, the distance from the efficient frontier, and the confidence over the final outcome. Compared with a verbal communication channel, an electronic communications channel is more "task oriented" than "social oriented." Therefore, the time to settlement is reduced, and the satisfaction with the system is higher. Like DSSs, the focus of NSSs is still on support. There is no facility for negotiators to specify their negotiation policies and strategies. The human negotiators are expected to apply their own negotiation policies and strategies when composing offers and counteroffers. The focus on NSS is to support the negotiator in a negotiation process, not to make decisions by computers.

Holsapple et al. [16, 17] have proposed a negotiation model. Kersten and Szpakowicz [18] proposed a decision model. The model described in Holsapple et al. [16, 17] is comprehensive and more general than ours from a theoretical perspective. However, Holsapple et al. [16, 17] have not implemented a system based on the model. The model described in Kersten and Szpakowicz [18] has been used in a system called *Negoplan.* Negoplan is a programming environment for decision support instead of a negotiation system such as ours. In our work, the objective is to develop an implemented platform for conducting and testing bilateral automated negotiations over the buying and selling of products or services. We have to simplify our model to make this objective achievable.

2.2 Agent Technology

Kasbah [19] is a Web-based multiagent marketplace where users create buying and selling agents to help exchange goods. A user wanting to buy or sell goods can create an agent, initialize it with a particular negotiation strategy, and send it off into the marketplace. Kasbah's agents proactively seek out potential buyers or sellers and negotiate with them on behalf of their owners. Each agent's goal is to complete an acceptable deal subject to a set of user-specified constraints such as a desired price, a highest (or lowest) acceptable price, and a date by which to complete the transaction. Negotiation between buying and selling agents in Kasbah is single-issue (i.e., price) and bilateral. After buying and selling agents are matched up, the only valid action for buying agents to take is to offer a bid to the seller. Selling agents respond with either a binding "yes" or "no." Kasbah provides buyers with one of three negotiation strategies: "anxious," "cool-headed," and "frugal"—corresponding to a linear, quadratic, or exponential function, respectively—for increasing a bid for a product over time. Similar negotiation strategies are provided for sellers. The simplicity of these negotiation heuristics makes it intuitive for users to understand how their agents are behaving in the marketplace, but these negotiation heuristics are almost too simple, even in the viewpoint of the developers. As pointed out by the developers, agents sometime make apparently "stupid" decisions.

Sierra et al. [20] presented a formal model of negotiation between autonomous agents. The model defines a range of strategies and tactics that agents can employ to generate initial offers, to evaluate proposals, and to generate counterproposals. The intelligence of a negotiation agent is manifested by the use of an evaluation function for each negotiation attribute (a simple aggregation function) and a simple, hard-coded, monotonic concession negotiation strategy. It is not clear whether business people, who usually do not have much knowledge about technology, can make use of the model effectively. In our opinion, the model is mainly defined by computer scientists for computer scientists.

2.3 Machine Learning

The field of machine learning attempts to let software agents learn how to negotiate by themselves rather than exhaustively implementing human negotiation knowledge into software agents. Zeng and Sycara [21] presented Bazaar, an experimental system for updating negotiation offers between two intelligent agents during a bilateral negotiation. The article contains a formal analysis of the negotiation state space, which is capable of tracking a rich set of issues and trade-offs that are necessary for a multi-issue negotiation. It explicitly models negotiations as a sequential decision-making task and uses Bayesian probability as the underlying learning mechanism. Zeng and Sycara [21] presented an example by using price as the issue of negotiation.

Another machine learning approach is to use genetic algorithms based on the principle of Darwinian evolution. In the context of negotiations, it works as follows: Each of the software agents begins with a population of various, randomly generated (and not necessarily good) negotiation strategies. It then employs its strategies against the strategies of the other agents in a round of bargaining, which takes place under specific predetermined rules and payoffs. At the end of a "generation," the agent evaluates the performance of each strategy in its current population and crosses over strategies from the current "parent" population to create a "child" generation of bargaining strategies. The more successful the strategies are the higher the probabilities that they are chosen as parents. Mutations can be randomly introduced. The size of the initial population, the number of generations, the crossover rates, and the mutation rate are parameters of the algorithm. In principle, after many trials, the negotiation agent is able to derive a good strategy; however, the actual result depends on many elements: for example, the number of training trials, the algorithm parameter configuration, and the quality of the negotiation strategy of the training opponent. For instance, to achieve a good strategy, the number of trials varies from about 20 generations [22] to 4,000 generations [23], and all runs must be made against opponent(s) whose strategies are as realistic as possible.

The machine learning approach to negotiation is promising in theory because it is able to derive negotiation strategies based on learning algorithms. However, more practical experiences are needed to evaluate the effectiveness. The learning approach used in Bazaar only deals with a single issue: price.

In summary, current research on automated e-business negotiations has tackled the decision-making process from different perspectives; however, a comprehensive decision model that integrates concepts of negotiation contexts, goals, policies, strategies, decision-action rules, and an automated negotiation system based on such a model is still missing.

3. OVERVIEW OF AUTOMATED BILATERAL E-BUSINESS NEGOTIATION

We present an overview of automated bilateral e-business negotiations to give the reader some background information about the restricted negotiation problem we address in this article.

3.1 Human-Based Versus Automated Negotiation Process

We can roughly divide the negotiation process into three phases: preparation, bargaining, and postnegotiation analysis. In the preparation phase, human negotiators collect information about the opponent(s), study previous negotiation cases, and identify both the strengths and the weaknesses of the opponent(s). In the bargaining phase, they state their positions, barter with the opponent(s), and make concessions, if necessary. In the postnegotiation analysis phase, they analyze the negotiation outcome, identify possible mistakes made during the bargaining phase, and adjust their tactics for future negotiations. These phases are similar to those in NSSs; however, all activities in human-based negotiations are conducted without the direct support of a computer system.

Automated negotiations make use of computer networks, especially the Internet, to conduct the negotiation process. Automation itself does not change the nature of the negotiation process. Preparation, bargaining, and postnegotiation analysis are still the main phases in automated negotiations. Figure 1 shows an overview of an automated, server-based negotiation process. Before a negotiation

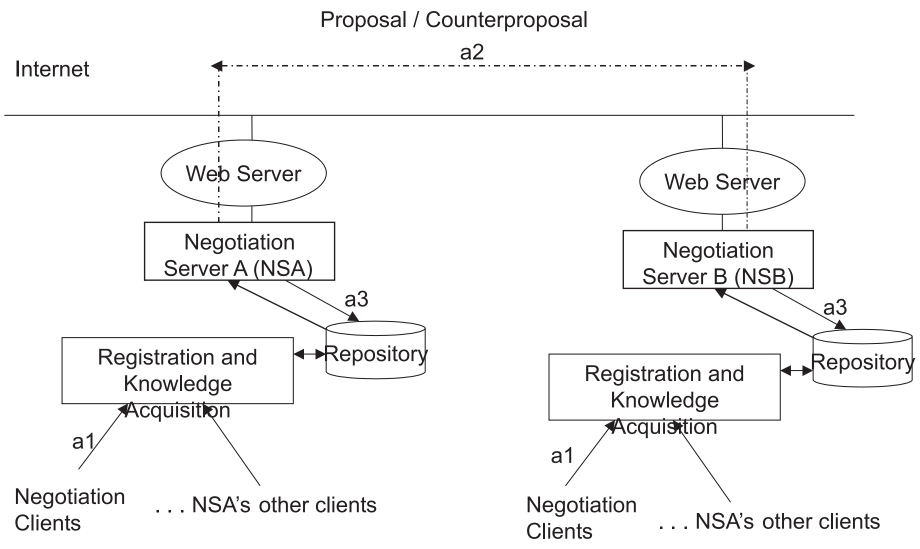
Proposal / Counterproposal
a2
Internet
Web Server
Negotiation Server A (NSA)
a3
Repository
Registration and Knowledge Acquisition
a1
Negotiation Clients
. . . NSA's other clients
Web Server
Negotiation Server B (NSB)
a3
Repository
Registration and Knowledge Acquisition
a1
Negotiation Clients
. . . NSA's other clients

Figure 1. Overview of an automated negotiation process.

begins, potential sellers can publish information about the goods and services on their Web servers. Through searching and browsing or by using available trader services, a buyer can identify potential sellers with whom the buyer wants to conduct negotiations. For buyers and sellers to make use of the automated negotiation service, each of them registers with a negotiation server of their choice and provide it with negotiation knowledge so it can conduct an automated negotiation on his or her behalf. The registration and knowledge acquisition activity is shown as a1 in Figure 1.

After a buyer decides to negotiate with a particular seller, the buyer starts the process by submitting an initial product or service request to his or her negotiation server. The negotiation server (NSA) initiates a bargaining process by creating and sending a call for proposal (CFP) to the counterpart's negotiation server (NSB), which represents the seller. NSB checks the information and constraints specified in the CFP against the registered information of the seller to determine whether the seller can satisfy the requirements and constraints. If not, the conflicting data conditions and constraints can be modified either by using the registered information and knowledge of NSB or through human intervention to produce a negotiation proposal to NSA. The proposal will then be evaluated by NSA against the buyer's registered requirements and constraints.

In the event that a conflict or constraint violation is found, relevant rules, which have been provided by the buyer, are applied to either (a) reject the proposal and suggest possible changes to NSB, (b) call for human intervention, or (c) generate a counterproposal to be returned to NSB. In the latter case, the counterproposal is sent back to NSB, which starts another round of negotiation. The counterproposal is evaluated in the same way by NSB against the seller's registered requirements and constraints. Preregistered decision-making rules of the seller are also applied.

This process of exchanging proposal and counterproposal continues until either an agreement is reached or one side decides to terminate the negotiation process. The exchange of proposal/counterproposal is shown as a2 in Figure 1. After the bargaining process is finished, information gathered during this process is stored in the respective repositories shown as a3 in Figure 1. Activities a1, a2, and a3 in Figure 1 roughly correspond to the preparation phase, bargaining phase, and postnegotiation analysis phase discussed in the human-based negotiation.

3.2 Negotiation Primitive and Protocol

In previous work, eight primitive operations for automated negotiation were introduced [10]. They are CFP, propose proposal, reject proposal, withdraw proposal, accept proposal, modify proposal, acknowledge message, and terminate negotiation. Table 1 gives a brief description of each primitive.

CFP and propose proposal are two alternative ways of initiating a negotiation transaction. Accept proposal and terminate negotiation are two alternative terminators that can bring a negotiation transaction to an end. Acknowledge message is used to acknowledge the receipt of a message. The remaining primitives, together with propose proposal, are iterators that can be used repeatedly in a bargaining process. Because a negotiation party is allowed to proactively propose an offer to the counterpart, propose proposal is used both as an initiator and an iterator. As an initiator, it represents an "unsolicited" proposal sent to a buyer by a supplier or vice versa.

To ensure effective and meaningful communications between negotiation servers, they must follow a well-defined protocol to exchange negotiation messages. In our work, we use two state transition diagrams to describe the buyer's protocol and the seller's protocol. Figure 2 shows the state and state transitions of the buyer's protocol. S0 is the initial state. S1 is entered after a CFP is sent to a seller, and S2 is entered if a proposal is initiated by the buyer to a seller. From either S1 or

Table 1
Negotation Primitives

Negotiation Primitive	*Description*
CFP	Initiate a CFP
Propose proposal	Send a proposal or a counterproposal that specifies the constraints and conditions acceptable to the sender
Reject proposal	Reject the received proposal with or without an attached explanation, and expect the negotiation counterpart to modify and resend the modified proposal
Withdraw proposal	Withdraw the previous proposal that has been sent
Accept proposal	Accept the terms and conditions specified in a proposal without further modifications
Modify proposal	Modify the CFP or the proposal that was sent
Acknowledge message	Acknowledge the receipt of a message
Terminate negotiation	Unilaterally terminate the negotiation process

Note. CFP = call for proposal.

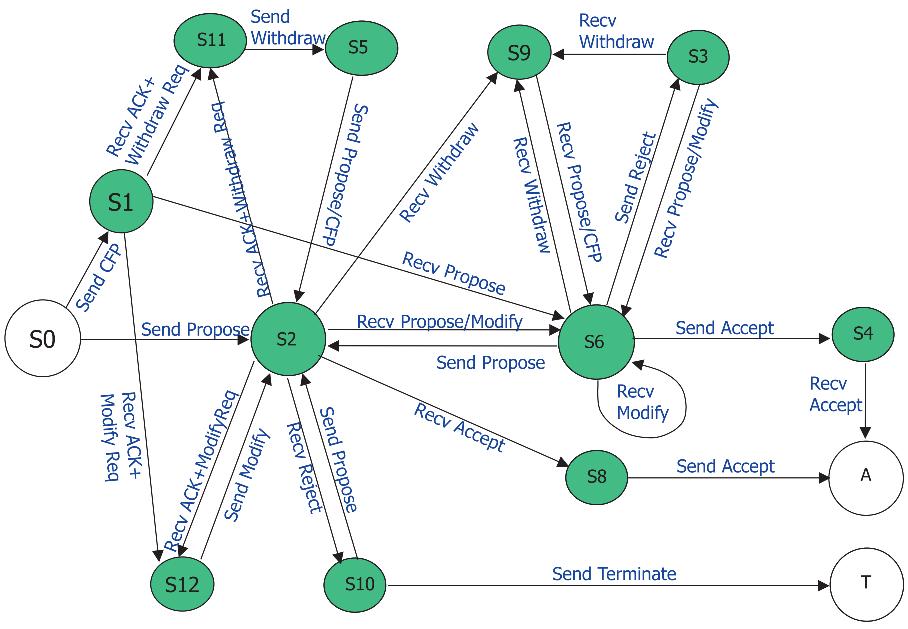

Figure 2. State transition diagram of buyer's negotiation protocol. Recv = receive; ACK = acknowledgment; Req = request; CFP = call for proposal.

S2, the buyer's server will transit to S6 if a proposal or counterproposal from the seller is received. In S6, the buyer's server can either accept the seller's proposal (S6 →S4), reject the proposal (S6 →S3), receive a proposal modification request from the seller (stay in S6), receive a withdrawal message from the seller (S6 → S9), or generate and send a counterproposal to the seller (S6 →S2). In S2, the buyer's server can either receive a rejection message from the seller (S2 →S10), a request from the buyer to modify the previous buyer's proposal (S2 →S12), a request from the buyer without the previous buyer's proposal (S2 → S11), or a counterproposal from the seller (S2 →S6). The buyer's server may "ping pong" between S2 and S6 during the repeated bargaining process. The state "T" is entered when the buyer unilaterally terminates the negotiation by sending a termination message to the seller. The state "A" stands for the state in which an agreement has been reached by both parties after an exchange of acceptance messages. The transition diagram for the seller is very similar to that of the buyer except a state S7 replaces S2 for accepting a CFP, and S0 transits to S6 after receiving a proposal from the buyer. For more details about both protocols, readers are referred to Huang [10] and Su et al. [11].

4. DECISION MODEL FOR AUTOMATED BUSINESS NEGOTIATIONS

In this section, we first introduce our decision model and explain the elements in the model. The model lays the overall framework for the rest of this article. In this section, we also cover three key elements of the model: miniworld, negotiation context, and negotiation goal.

4.1 Decision Model

The decision model is formally defined as a seven-tuple <*W, CX, G, L, P, S, CB*>, where *W* is an enterprise's miniworld, which represents the information, material, financial, personnel, and other resources that the enterprise has access to; *CX* is a set of negotiation contexts specified by an enterprise; *G* is a set of negotiation goals of an enterprise; *L* is a set of plans of decisions and actions used by an automated negotiation server; *P* stands for negotiation policy, which maps a negotiation context to a negotiation goal; *S* stands for negotiation strategy, which maps a negotiation goal(s) to a plan of decisions and actions; and *CB* stands for a cost–benefit evaluation and selection model.

During a negotiation process, a negotiation server may receive a proposal that contains several combinations of product/service features and specifications that are acceptable to its counterpart. It needs a way of evaluating these combinations or alternatives, ranking them in terms of their costs and benefits and selecting the best one for acceptance. When generating a counterproposal, it also needs to evaluate and rank available options and select the proper one to form the counterproposal. We have adopted the cost–benefit decision-making model proposed in Su et al. [24] and have implemented a cost–benefit evaluation server [25] for use as a component of the negotiation server.

Figure 3 shows the interrelation of all of these key elements of the decision model except *CB,* which is part of *L* (a set of plans for decisions and actions used by an automated negotiation server). We explain the figure progressively in the following subsections.

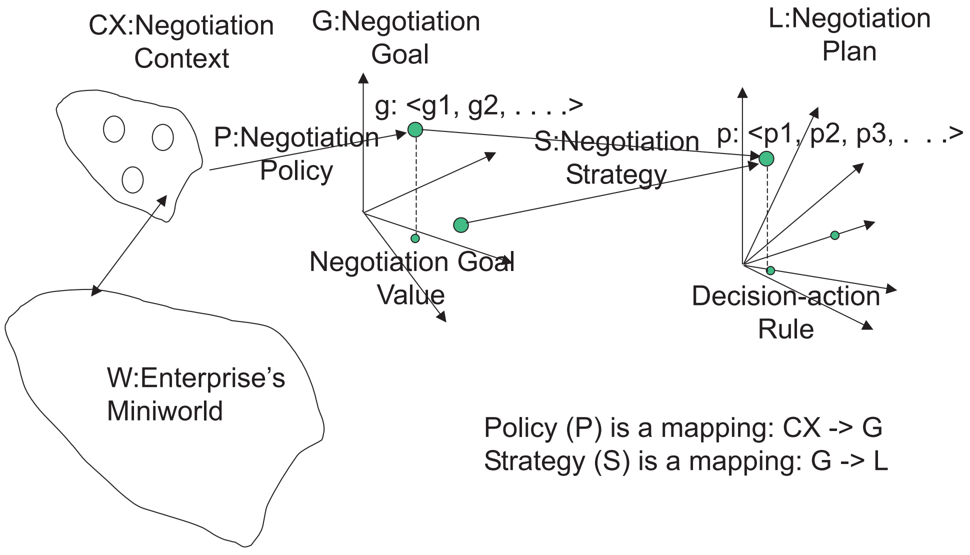

Figure 3. Relationship among key elements.

4.2 Enterprise Miniworld and Negotiation Context

Every business enterprise operates in a miniworld of business in which the enterprise has access to some information, material, financial, and personnel resources that exist in the real world. These resources may be available either in-house or in other enterprises but accessible to the enterprise. They represent the internal and external conditions and states of the enterprise's business. The miniworld of one enterprise can be expected to be quite different from that of another. It changes constantly and reflects the dynamic nature of an enterprise's business.

An enterprise usually conducts different types of negotiations with many different counterparts and under different negotiation conditions or situations. Information about the counterparts and different types of internal and external conditions or situations is important for defining the specific goal of negotiation. For example, information about what types or sizes of companies it deals with, the credit ratings of these companies, the current market conditions, its own inventory level, internal and external business events, or other accessible information in the enterprise's miniworld are all important for setting the goal. Some information may be stored in the enterprise's local database and/or application systems. Others may be accessible from remote databases or application systems, for example, by calling the Web services of remote objects that encapsulate these databases and application systems. Based on the accessible information, an enterprise can define a set of negotiation contexts, each of which can be expressed by a Boolean expression whose truth value can be determined by evaluating the expression against the enterprise's miniworld (see Figure 3). These contextual expressions capture the typical negotiation conditions and situations that the enterprise encounters and for which different negotiation goals can be specified.

4.3 Negotiation Goal and Goal Dimensions

The goal that an enterprise wants to achieve in a particular negotiation can differ from one negotiation context to another. For example, the goal of a supplier may be to achieve the maximum profit and to reach an agreement quickly in a negotiation if the buyer is a small company, without good credit, and ordering a small quantity of a product. On the other hand, if the buyer is a very reputable company with good credit and the order is large, the goal may be to take the minim profit or even no profit for establishing a business relationship with the buyer without any concern over the length of time needed in a negotiation to reach an agreement. The if conditions in the previous example are negotiation context specifications, and negotiation goals are specified in terms of profitability (*PF*), desire for relationship (*D*), and time to agreement (*T*), which are different aspects of a negotiation goal. We call these different aspects of a negotiation goal the goal dimensions. An enterprise can define any number of goal dimensions that are deemed necessary and appropriate to express its goals.

It is desirable to have a quantitative way of specifying negotiation goals with respect to a set of defined goal dimensions. For the previously discussed purpose, we first introduce the concept of goal space. The goal dimensions defined by an enterprise form a multidimensional goal space as shown in Figure 3. Each dimension has an index with a value in the range between 0.0 and 1.0 to serve as a specification of the degree of importance for a company to achieve the negotiation goal in that goal dimension. The profitability index is used by the policymaker to specify the importance of making monetary gain. The *PF* value indicates the minimum level of profit that must be made on a deal. It will influence the "bottom line" and the decision-action rules used in a negotiation process. A profitability index *PF* that is close to 1.0 indicates that a high profit must be made before the deal is accepted. A low *PF* value indicates that to satisfy other goals, the policymaker is willing to make a lower profit. If the negotiation party is a supplier, a high value for *PF* indicates the supplier prefers a high price. If the negotiation party is a buyer, a high *PF* value indicates the buyer prefers a low price.

Time is an important factor in most business activities. Thus, time to reach an agreement should be considered as an important dimension of a negotiation goal. In most business negotiations, if the time to reach an agreement is too long or passes a deadline, the final result of the negotiation is no longer relevant. For example, if a company produces a product suitable as Christmas gifts and requires 25 days to manufacture and deliver the product, the company may have no interest in any negotiation on a raw material purchase if the delivery day is after the first day of December. This goal is represented in the goal space by the time-to-agreement index *T*. If a quick resolution (either agreement or termination) is desired or required, then the value for *T* should be specified closer to 0.0. The value 1.0 would indicate that the enterprise does not wish to set a time limit on the negotiation process.

The desire-for-relationship index represents how strongly the policymaker wants to establish a business relationship with the counterpart in a negotiation. An index *D* of 1.0 means it is a "must have" deal for whatever the reason. For example, a start-up company is eager to establish a long-term relationship with a Fortune 500 company for future credit reference. Its policymaker would most likely set the *D* value close or equal to 1.0. A value that is close to 0.0 means that a business relationship with the company is not at all important.

A negotiation goal is therefore a point in the multidimensional goal space, which can be quantitatively represented by a triplet (in our three-dimensional example) as (*pf, t, d*), where *pf, t,* and *d* are values corresponding to the goal dimensions *PF, T,* and *D,* respectively. It should be noted that *PF, T,* and *D* are only examples of goal dimensions. Different enterprises can have different types and numbers of goal dimensions. Also, these goal dimensions are not completely independent. For example, if a company wants to have a good relationship with another company, it may need to reduce the value of the profitability dimension.

Given the fact that there are multiple- (three in our study) goal dimensions, it is natural to measure the distance between different goal points in the goal space. A software agent or a person needs to be able to express his or her preference for a given value of *PF, T,* or *D* and have a way of aggregating the preference score for *PF, T,* and *D* values to come up with a global preference score so that it can be compared with the global preference score derived for the other (*PF, T, D*) triplet. This is a cost–benefit decision problem, which is out of the scope of this article. The cost–benefit decision model and the cost–benefit evaluation system used for evaluating (*PF, T, D*) triplets are presented in Liu et al. [25].

5. NEGOTIATION POLICY

5.1 Definition of Negotiation Policy

The term *policy* in English has different meanings in different contexts. In the *American Heritage Dictionary of the English Language* (3rd ed.) [26], policy has two entries: The first entry has three definitions. The first one is "a plan or course of action, as of a government, political party, or business, intended to influence and determine decisions, actions, and other matters" (p. 1401). Examples of policies are American foreign policy and the company's personnel policy. The second definition is "a course of action, guiding principle, or procedure considered expedient, prudent, or advantageous" (p. 1401). One example is "Honesty is the best policy." The third definition is "prudence, shrewdness, or sagacity in practical matters" (p. 1401). This definition is often used to modify another noun such as policy statements and policy issues.

The second entry has two definitions. The first one is "a written contract or certificate of insurance" (p. 1401). Examples of such policies are the "return policy" or "refund policy" of purchased goods. This definition of policy is essentially the "terms and conditions" usually specified in purchase contracts. In our opinion, they are a part of the attribute set to be negotiated. For example, return policy and refund policy can be represented as attributes associated with a negotiation transaction (i.e., transactional attributes instead of product/service attributes). The second definition is "a numbers game" (p. 1401).

Our definition of a business negotiation policy is based on the concepts and terms given in the first 2 definitions of the first entry. A business negotiation policy is the following: "A general guiding principle for achieving a business negotiation goal under some specified conditions. It is intended to influence and determine the decisions or actions to be taken in a business negotiation." A business negotiation policy does not specify what specific decisions and actions should be taken to achieve the goal. The specifications of decisions and actions for implementing negotiation policies are called negotiation strategies. We address negotiation strategies in Section 7.

5.2 Rule Representation of Negotiation Policy

Negotiation policies are specifications that relate specific negotiation contexts to specific goals. The negotiation contexts, goal dimensions, and policies specified by one enterprise can be different from those of others. It is the responsibility of the people who play the role of policymaker in an enterprise to define them. We formally define negotiation policy as a function *P,* which maps from the set of negotiation contexts *CX* to a set of goal points *G* in the goal space: that is, $P: CX \rightarrow G$ as illustrated in Figure 3. A specific policy can thus be specified by the following general expression: If *CXi,* then *Gj,* where *CXi* is a member of *CX* and *Gj* is a member of *G* for some *i* and *j*. For example, a general policy defined by a medium-size company can be as follows:

> If the counterpart is a Fortune 500 company AND the counterpart is a company with which this company previously has a good business relationship

AND the monetary value of the deal to be negotiated is large
THEN $P = 0.3$, $T = 0.9$, $D = 0.9$.

The business goal represented by (0.3, 0.9, 0.9) specifies that the company is willing to take a low profit for establishing a long-term relationship with that Fortune 500 company. It is also willing to spend time in negotiation to reach a deal. The preceding policy specification is not in a formal language. In our work, we use the rule specification language and graphical user interface (GUI) tools that have been developed for an event-trigger-rule (ETR) server [27] for policy rule specifications.

6. DECISION-ACTION RULES

The automated negotiators (ANs) must follow the state diagram (i.e., the negotiation protocol) shown in Figure 2 to exchange negotiation primitives and data during a negotiation process. At each state, an AN needs to make a decision or take action based on the received data before transiting to the next meaningful state. The specification of that conditional decision or action can be in the form of a decision-action rule. Generally speaking, there are alternative decision-action rules that can be used by an AN in each transition of the negotiation protocol. Based on the bilateral negotiation protocol shown in Figure 2, the transitions in various states of the protocol are shown below:

- The initiation of a negotiation transaction (issuing the initial negotiation proposal or call for proposal)
- The acceptance of a proposal
- The rejection of a proposal
- The termination of a negotiation transaction
- The modification of a proposal
- The withdrawal of a proposal
- The generation of a counterproposal

In the following subsections from 6.1 to 6.7, we suggest some useful alternative decision-action rules associated with the preceding transitions. Decision-action rules are defined by people in a business enterprise who play the role of negotiation expert. Decision-action rules capture the techniques/tactics used by skillful human negotiators and are used by an AN to conduct its negotiation on behalf of the enterprise.

6.1 Initiation of a Negotiation Transaction

In our negotiation protocol, a negotiation transaction instance is started by a negotiation server who issues either a CFP or a propose proposal on behalf of a company. Decisions need to be made with respect to the contents of the CFP or propose proposal. Alternative decision-action rules can be used to create the contents. For example, if the negotiation attributes such as price, quantity, and delivery date form a part of the data contents of a propose proposal, the initial values provided for these attributes can either be very close to what the issuer desires (i.e., the bottom line) or exceed what the issuer desires with the expectation that the counterpart may bargain them down. Other rules can be applied to determine the proper initial values that are appropriate for achieving a negotiation goal defined in terms of profitability, time to agreement, and desire for relationship as discussed before.

The initiation of a negotiation by a CFP or a propose proposal represents two different negotiation styles. If a negotiator is quite familiar with the counterpart and the negotiation environment, the negotiator may start a negotiation by issuing a proposal to the counterpart. On the other hand, if a negotiator is not familiar with the counterpart and the negotiation environment, the negotiator may choose to issue a CFP to ask the counterpart to give the initial offer. The choice of issuing a CFP or a propose proposal may depend on the personality of a negotiator. A competitive and hasty negotiator may choose to issue the initial proposal to get the "first move" advantage; however, a cooperative and cautious negotiator may choose not to issue the initial proposal but to wait for a proposal submitted by the counterpart.

6.2 Acceptance of a Proposal

Let us assume that the negotiation server of a buyer receives a propose proposal from the negotiation server of a supplier in response to the buyer's CFP or propose proposal. After the analysis of its contents, some decision-action rules need to be applied to guide the acceptance decision. For example, the proposal is accepted if

1. All attributes of the proposal are within *X*% difference of the desired values.
2. All major attributes are satisfied.
3. Some key attributes are satisfied, and some attributes are within *X*% difference.
4. Global preference score derived from all the attribute values exceeds a certain threshold.

Note that the value assigned to the parameter, *X*, of a parameterized decision-action rule determines the degree of flexibility a user or organization wants its negotiation server to pursue that negotiation. If *X* is a small value, the negotiation plan is an inflexible one because the rule requires that the counterpart should move very close to the negotiator's position. On the other hand, the plan is a flexible one if *X* has a large value. This is because the negotiator is willing to accept the counterpart's offer even if it is not very close to the negotiator's desired value. Different *X* values used in the parameterized rules (1 and 3 in the preceding list) represent alternative rules.

6.3 Termination of a Negotiation Process

The negotiation protocol provides a set of well-defined "rules of engagement" for both parties to exchange proposals and counterproposals. The acceptance and termination states of the protocol are the states to which a negotiation process should converge; however, the protocol itself does not guarantee that these states will be reached because the negotiation parties may engage in an endless exchange of proposals. To ensure a negotiation transaction's convergence to one of the terminal states, decision rules, which explicitly specify the conditions for termination, must be specified and applied. These conditions may depend on the data conditions presented in the previous proposals that have been received. For example, the negotiation server can use the previous exchange of proposals to determine if the negotiation is converging or diverging and if the counterpart is acting reasonably or not. If not, the negotiation process should be terminated. Some examples of termination rules are given following:

1. The global preference scores derived from the received proposals indicate that the negotiation has been moving away from the desired direction for *X* number of times.
2. The global preference scores derived from the received proposals are oscillating.
3. The negotiation time exceeds *X* time units.
4. The number of proposal exchanges exceeds *X* times.
5. Some combinations of the preceding items.

Parameterized decision-action rules for termination provide a quantitative means to make termination decisions, and the negotiation history allows a negotiation server to detect the convergent or divergent trend of a negotiation in a traceable and justifiable manner. To support such decisions, a component to record the history of negotiations is needed. The history of an ongoing negotiation transaction is required for the enforcement of the preceding decision-action rules. The history of previous negotiation transactions with a counterpart is useful in the selection of the proper negotiation policy and strategy when negotiating with the counterpart, thus increasing the likelihood that an agreement can be reached in a timely fashion.

6.4 Rejection of a Proposal

A negotiation server may decide to reject the current proposal. A rejection is different from a termination in that the negotiation process would continue. It simply informs the counterpart's negotiation server that the received proposal is not acceptable and requests the sender to modify and resubmit the proposal. Some decision-action rules for guiding the rejection decision are given below:

1. The data conditions specified in the proposal are below the acceptable bottom-line values.
2. Insufficient information is provided in the proposal.
3. The data conditions specified in the proposal are acceptable, but the receiver of the proposal wants its counterpart to do further concession.

6.5 Modification of a Proposal

During the negotiation process, the user, who is monitoring the progression of the negotiation, may issue a request to his or her negotiation server that he or she wants to modify a proposal that has been sent. Rules can also be defined to guide a negotiation server to make some modification to a proposal automatically. For example, if the negotiation server has not received a response for a previous proposal from its counterpart after *X* time units, a decision-action rule may trigger the modification of the delivery date, the price, and so forth that are specified in the previous proposal. In another case, if the prices of a company's products to be sold are updated monthly or quarterly, the negotiator may need to modify the proposal that has been sent if no response has been received and the time limit is approaching or has reached the monthly or quarterly boundary.

6.6 Withdrawal of a Proposal

Similar to a proposal modification, a user may initiate the withdrawal of a proposal. Rules can be defined to guide a negotiation server in making the withdrawal decision automatically. For example, if the financial condition of the issuer has changed, if the customer has changed his or her order, if the inventory has dropped to a certain level, or if the counterpart has not responded to the previous proposal after *X* time units, the proposal should be withdrawn.

6.7 Counterproposal Generation

Counterproposal generation is one of the most important tasks in a bargaining process. In the analysis of a received proposal, a negotiation server may find that some of the data conditions are not acceptable (meaning that they do not satisfy the requirements and constraints of its client). In that case, some of these conditions need to be changed to form a counterproposal. However, very often there are many alternative data values that can be used in the counterproposal. A decision needs to be made to select a suitable combination of values to form the counterproposal. In this case, the *CB* model (*CB* decision model) must be used to evaluate the alternative combinations and derive global cost–benefit scores for them. The selection of a proper combination can be guided by a decision-action rule, which can be one of the following:

1. Purely selfish (i.e., select the one with highest global cost–benefit score).
2. Purely unselfish (i.e., select the one with lowest global cost–benefit score).
3. Some point in between.

In generating a counterproposal, a negotiation server may have to make concessions based on some decision-action rules. A negotiation position can be lost or gained depending on the concession strategy used. Concessions in negotiation can be broadly classified into two types: static concession and dynamic (or adaptive) concession.

6.7.1 *Static Concession*

Static concession controls the behavior of a negotiation server based on some predefined functions. Some examples of static concession are as follows: One useful function is the Sigmoid function [28]: $f(x) = 1/(1 + \exp(-g \times (x - 4)))$. If this function is used in a negotiation setting, x can be interpreted as the xth counterproposal to be generated, and the value returned from $f(x)$

can be interpreted as the value to be sent to the counterpart. The preceding Sigmoid function results in Figure 4 for $g = 1.5$.

As a decision-action rule, this function can be used to control the rate of concession. As shown in the graph, the concession rate is initially very slow. It increases drastically in the middle of the negotiation. Then the concession rate slows down again toward the end of the negotiation when the bottom-line value of a negotiated attribute is approaching. By choosing a proper value for g, the concession rate can be controlled. By controlling the concession rate, the speed of convergence of a negotiation transaction can also be controlled (e.g., the higher the concession rate, the faster the negotiation can reach an agreement.) This curve has one turning point (when $x = 4$). It is possible to define functions that have multiple turning points. Alternative parameter values of a parameterized decision-action rule (e.g., g values in this case) determine how competitively the strategy is to be carried out.

Other functions such as $f(x) = k \times x$, which gives concessions at a constant rate, can be used in a counterproposal generation. Another rule is not to give concession at all under certain situations (some people call the phenomenon "Boulwarism" [7]). Users can also define arbitrary functions based on user-selected pivotal points and interpolation and extrapolation techniques.

6.7.2 Dynamic Concession In the static approach, the function used for concession is selected, and its parameters are set based on a selected decision-action rule. Once selected and set, it proceeds without any regard to what the counterpart does. Dynamic concession strategies take the negotiation behavior of the counterpart into consideration; the function can be adaptive. We explain two examples of dynamic concession strategies following.

1. We assume that the attribute (say price) under negotiation is independent of the other attributes. If a negotiation server keeps track of the attribute values presented in the previous proposals sent by its counterpart, it is possible to determine the concession applied by the counterpart (i.e., counterpart_concession (x)). The concession of the negotiation server is also known based on the static concession

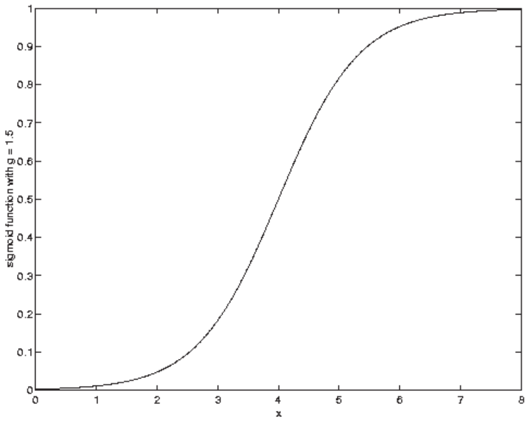

Figure 4. Graph for Sigmoid function with $g = 1.5$.

function used by the server (i.e., own_concession (*x*)). We can use the following equation to control the actual concession:

$$f(x) = A \times \text{own\_concession}\ (x) + (1.0 - A) \times \text{counterpart\_concession}\ (x),$$

where *A* is the adjustment factor whose value is in the range of 0.0 to 1.0, own_concession (*x*) is the function to compute the concession used by my own side, and counterpart_concession (*x*) is the function to compute the concession used by the counterpart.

Note that when $A = 1.0$, counterpart_concession (*x*) is ignored, and the function is reduced to a static concession function. When $A = 0.0$, own_concession (*x*) is ignored, and *f* is equal to counterpart_concession (*x*). In the latter case, the server totally adapts to the concession speed used by the counterpart. When *A* is chosen to be some value between 0.0 and 1.0, the concession pattern is a combination of both.

There are several ways to compute the counterpart_concession (*x*). One simple way is to take the average of the concession values found in the received counterproposals. For example, if the counterpart has the concession values of 20.0, 17.0, 15.0, and 12.0 in the previous four counterproposals, we calculate the counterpart_concession (*x*) to be $(20.0 + 17.0 + 15.0 + 12.0)/4 = 18.0$. Another way is to use extrapolation techniques to derive a polynomial curve and get the next value. For example, we can treat the preceding numbers as four points—(1, 20.0), (2, 17.0), (3, 15.0), and (4, 12.0)—in the *x* and *y* coordinates then compute coefficients for a 4-degree polynomial curve. We can then get the *y* value for $x = 5$.

2. A negotiation server can assign an "importance factor" (from 0.0 to 1.0) to each attribute under negotiation based on the input of its client. For example, 1.0 implies very important, and 0.0 implies not important. The server can also compute the importance factor for the same attribute of the counterpart. For example, one heuristic could be that the counterpart's importance factor is inversely proportional to the counterpart's concession speed for that attribute. A tactic that can be followed is to compare the importance factors associated with all the attributes of both sides. If some of them are different, it would be beneficial to increase the concession speed of those attributes that are not important to my own side but important to the counterpart side and decrease the concession speed of those attributes that are important to my own side but not important to the counterpart side.

To support the adaptive concession strategies, a component to record the history of negotiations is needed. For example, the history of an ongoing negotiation transaction (i.e., proposal exchanges) can be used to determine the speed of concession applied by the counterpart. In addition, the system can learn from experience (i.e., the history of previous negotiation transactions with a counterpart) to generate new policies and strategies and/or to modify the existing policies and strategies dynamically at run time.

6.8 Plan of Decision and Action

If *D*1, *D*2, … *Dn* (*n* is equal to 7 in our protocol example) are domains of alternative decision-action rules associated with *n* transitions, a negotiation plan is a combination of n decision-action rules, each of which is selected from a domain of alternative rules to guide the decision and action of a negotiation server during a negotiation process. Different combinations of decision-action rules form different negotiation plans. Thus, formally, the set of negotiation plans (L) defined by a business organization can be represented by a set of tuples having the general structure $< d_1, d_2, \ldots d_i, \ldots d_n >$, where d_i in D_i for $(1 \leq i \leq n)$ is a decision-action rule. Conceptually, the transitions of a negotiation protocol form an n dimensional space, which is called the *plan space.* Negotiation plans are points in the plan space as illustrated in Figure 3. Along each dimension or axis, a number of decision-action rules can be defined and arranged in an order based on how competitive or cooperative these rules are.

7. NEGOTIATION STRATEGY

To have a clear and precise definition of business negotiation strategy, it is useful to first make reference to some definitions of *strategy* as found in the dictionary [26] mentioned in Section 5.1. We show them here:

1. The science and art of using all the forces of a nation to execute approved plans as effectively as possible during peace or war.
2. The science and art of military command as applied to the overall planning and conduct of large-scale combat operations.
3. A plan of action resulting from strategy or intended to accomplish a specific goal.
4. The art or skill of using stratagems in endeavors such as politics and business. (p. 1775)

The third and fourth definitions of strategy are closer to what we think the definition of business negotiation strategy should be. We define a *business negotiation strategy* as follows: "A plan of decisions or actions for accomplishing a business negotiation goal."

Having formally defined negotiation goals and negotiation plans, we can formally define a negotiation strategy as a function $S: G \rightarrow L$, where G is a set of goals in the goal space and L is a set of negotiation plans in the plan space, and S maps from G to L. Conceptually, strategies are mappings from some points in the negotiation goal space to some points in the negotiation plan space as illustrated in Figure 3. Each mapping can be either one to one, many to one, many to many, or one to many. In the case of a one-to-one mapping, a person who plays the role of the negotiation expert in a business enterprise has identified a specific goal to achieve and knows a specific plan of decisions or actions, which can be used by a negotiation server to achieve the goal. In the case of a many-to-one mapping, the expert specifies that a number of goals can be achieved by a single negotiation plan. For example, a strategy specification may state that if P dimension in the goal space is in the range of (0.2, 0.5), T in the range of (0.6, 0.7), and D in the range of (0.6, 0.8), then the negotiation plan should use the decision-action rule R1 for the initiation of a negotiation transaction, parameterized rule R5 with parameter value set to 0.05 for the acceptance of a proposal, R11 for the termination of a proposal, and so forth. The range specifications for P, T, and D, in effect, allow a number of goals to be mapped to a specific negotiation plan. In the many-to-many mapping case, multiple negotiation plans can be applied to achieve multiple negotiation goals. In the one-to-many mapping case, a goal can be realized by multiple plans. This represents the case that different negotiation experts have different opinions on how to achieve a specific goal.

Although the specification of a negotiation strategy can be one of the preceding four types, when a business enterprise enters into a particular negotiation, the enterprise's miniworld is defined, and the negotiation contexts specified in policy rules can be verified against the miniworld. A specific goal will be selected based on a selected policy, which maps the verified negotiation context to the goal. A strategy specification may map the goal to one or multiple plans. In the latter case, the negotiation server would need the input of a human to select one from the alternative plans because a single plan, with a combination of decision-action rules, should be used for driving the negotiation process of this negotiation transaction in its initiation, rejection, termination, and so forth. At run time, if the enterprise's miniworld has been changed, new policy and strategy may have to be applied to determine a new plan of decisions or actions. A negotiation system and its architecture should support such dynamic changes.

8. SYSTEM ARCHITECTURE AND IMPLEMENTATION

In this work, we have developed several system components that implement the new decision elements we presented in this article and integrated them with the negotiation server reported in Huang [10], Su et al. [29], and Su et al. [11]. Figure 5 shows the architecture of the integrated system. The new components are shown in gray color. During the build time, the policymaker and the negotiation expert of a business enterprise specify their business goals, policies, strategies, and decision-action rules by using the GUIs provided by a knowledge acquisition processor. Some other negotiation knowledge, such as miniworld information, is imported to a knowledge repository by using the application program interfaces provided by a knowledge manager. Negotiation policy rules, strategic rules, and decision-action rules are stored in the rule base of the ETR server. During a negotiation session, these rules are triggered by events (e.g., an event that represents a transition in the negotiation protocol) posted by the negotiation server to the ETR server.

The build-time activities are divided into two parts: activities for each negotiation transaction and activities for each negotiation session. A negotiation transaction is a sequence of negotiation sessions. Each session is an exchange of a negotiation message and its processing. Activities for each transaction select the appropriate decision-action rules to drive the execution of the negotiation server. They are accomplished by the negotiation plan selector and the ETR server. When a negotiation transaction is triggered by an external component, the miniworld information is retrieved from the knowledge repository through the knowledge manager. The miniworld information is used to evaluate the contextual expressions in policy rules. There is a possibility that multiple contextual expressions are evaluated to true. Therefore, multiple sets of goal values are selected. The negotia-

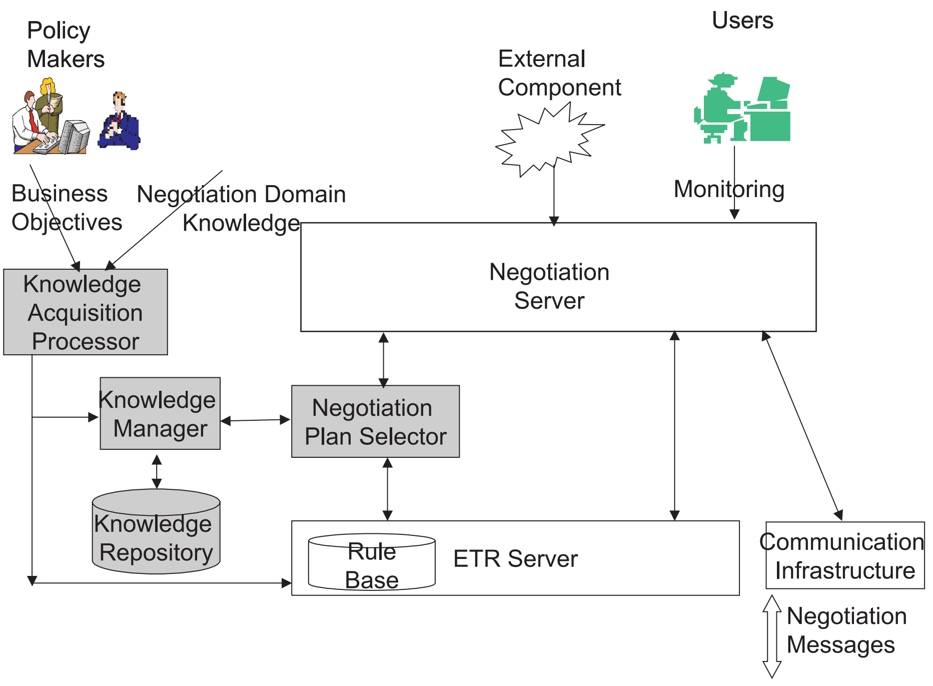

Figure 5. System architecture. ETR = event trigger rule.

tion plan selector takes these sets as inputs and derives a final set of goal values based on a predefined conflict resolution mechanism (e.g., by taking the average of the goal values, the maximum or the minimum value of each attribute). After the final set of goal values is determined, a strategic rule is matched, and a final set of decision-action rules is selected for use by the ETR server.

After the build-time activities for each transaction are completed, activities for each session begin. A negotiation server is ready to generate negotiation messages or process incoming messages. It consults with the ETR server to make a decision at each transition of the negotiation protocol. The ETR server makes the decision based on the selected decision-action rules and sends the decision back to the negotiation server. Once the decision is made available, the negotiation server generates an appropriate negotiation message and sends it to the counterpart's negotiation server through the communication infrastructure. The process continues until either a mutual agreement is reached or one side unilaterally terminates the transaction.

9. SYSTEM EVALUATION

We have evaluated the system described in the preceding section in two application scenarios: (a) negotiation over time, delivery period, and quantity in a computer purchase and (b) a two-tier, multilateral negotiation in a supply chain. We briefly describe them following. Detailed descriptions of these two applications can be found in Li [30], Lodha [12], and Su et al. [31].

9.1 Computer Purchase Scenario

This scenario is designed to show how high-level policy and strategic rules influence the low-level decision-action rules used to generate proposals and counterproposals in a bilateral bargaining process over product price, quantity, and delivery period and to terminate the bargaining process. We show and describe some rules used in one of the bargaining processes in this scenario following. The following is the policy rule used:

```
CONDITION [rulePar.roleName eq "S1"]
rulePar.mWorld.compInfo.companySize eq "F500" AND
rulePar.mWorld.compInfo.creditRating eq "excellent" AND
rulePar.mWorld.orderSize eq "large"
ACTION
rulePar = new CommonData.RuleResponse();
rulePar.initialResult.myGoal.p = 0.3;
rulePar.initialResult.myGoal.t = 0.9;
rulePar.initialResult.myGoal.d = 0.9;
return rulePar;
ALTACTION
rulePar = new CommonData.RuleResponse();
return rulePar;
```

To simplify the description, related event and trigger information are not shown. The preceding policy rule describes a possible corporate negotiation policy: If the counterpart is a Fortune 500 company, its credit rating is excellent, and the size of the order is large, set the *P* (profitability) to 0.3, *T* (time to agreement) to 0.9, and *D* (desire for relationship) to 0.9. The values are returned in an object instance of CommonData.RuleResponse. The policy says that the company is willing to take less profit, does not mind if it takes a long time to make a deal, and has a high agree of desire to establish a business relationship with the stated type of companies.

The following is the strategic rule used:

```
CONDITION [rulePar.roleName eq "S1"]
(rulePar.finalResult.myGoal.p ≤0.4) AND
(rulePar.finalResult.myGoal.t ≥ 0.6) AND
(rulePar.finalResult.myGoal.d ≥ 0.8)
ACTION
rulePar = new CommonData.RuleResponse();
rulePar.acceptancePrice = "S1A1Price";
rulePar.acceptanceDelivery_period =
"S1A1Delivery_period";
rulePar.difPrice = 0.2;
rulePar.difDelivery_period = 0.2;
rulePar.terminateRuleName = "S1T1";
rulePar.counterRuleName = "S1C1";
rulePar.modifyRuleName = "S1M1";
```

```
return rulePar;
ALTACTION
return null;
```

The preceding strategic rule specifies that if *P* value is less than or equal to 0.4, *T* value is greater than or equal to 0.6, and *D* value is greater than or equal to 0.8, the following rules are chosen. Rule "S1A1Price" is chosen for the acceptance criterion of price, and Rule "S1A1Delivery_period" is chosen for the acceptance criterion of delivery period. The acceptable difference in price is 20% (a parameter of S1A1Price), and the acceptable difference in delivery_period is also 20%. The rule for termination is "S1T1," the rule for counterproposal generation is "S1C1," and the rule for modification is "S1M1." The following is a part of the termination rule:

```
CONDITION [rulePar.terminateRuleName eq "S1T1"]
rulePar.numOfProposals > rulePar.terminateProposalNum
ACTION
rulePar = new CommonData.RuleResponse();
rulePar.terminateFlag = true;
return rulePar;
ALTACTION
rulePar = new CommonData.RuleResponse();
rulePar.terminateFlag = false;
return rulePar;
```

The preceding termination rule specifies that if the rule name is S1T1 and the number of proposals that have been exchanged (rulePar.numOfProposals) exceeds a certain threshold specified by rulePar.terminateProposalNum, the negotiation transaction should be unilaterally terminated. In this case, an instance of the object CommonData.RuleResponse is created with its termination flag (rulePar.terminateFlag) set to true and returned to the negotiation server. Otherwise, the alternative action to return the object instance with its termination flag is set to false.

The preceding scenario serves to illustrate how a high-level corporate policy can be mapped to low-level decision-action rules that control the behaviors of an automated negotiation server through a strategic rule. The distinction made between policy and strategy is very useful because a business policy that represents a business organization's long-term goal can stay the same. By changing the strategic rule used to implement the policy, a different set of decision-action rules can be applied by the negotiation sever to achieve a different negotiation result. This distinction reflects the real-world situation in which skillful negotiators (e.g., sales persons) can apply different negotiation strategies in business transactions within the guideline of corporate policies. We also point out that because a policy is quantified and represented by the values of *P*, *T*, and *D*, the mapping conditions of a strategic rule can be expressed by a combination of ranges of values instead of just single values. This allows the same strategic rule to be applied to implement different but related policies.

9.2 Supply Chain Negotiation Scenario

To verify the utility of our negotiation system, we have developed and implemented a set of supply chain scenarios [12]. These scenarios include major players of a supply chain such as retailers, distributors, and manufacturers. They describe business processes, which include determining qualified suppliers, requesting quotes, and automating negotiations between any two entities in the supply chain. They also include inventory replenishment, quote evaluation, transporting decision, cost–benefit evaluation, and order processing for any entity in the chain. We describe one of the scenarios, a two-tier negotiation scenario, in the following.

In the two-tier scenario shown in Figure 6, the retailer first sends the distributor a request for quote (RFQ) for the purchase of some units of a product. The distributor checks its inventory and finds that it cannot satisfy the quantity requested by the retailer. It thus sends an RFQ to the manufacturer for the purchase of sufficient units of the product to satisfy the retailer's order. On the receipt of the quote from the manufacturer, it generates and sends a quote to the retailer. The retailer checks the quote and finds that some conditions given in the quote are not satisfactory (e.g., the price is too high, the delivery date is too late, etc.). The retailer then begins a negotiation process with the distributor to address these conditions. The negotiation server, which acts on behalf of the retailer, sends a proposal to the negotiation server of the distributor. On receiving the proposal from the retailer, the distributor's negotiation server checks the proposal against the distributor's policies and constraints and realizes that it, in turn, needs to start a negotiation process with the manufacturer in an attempt to satisfy the retailer's demand. In this case, the negotiation with the retailer would have to depend on the outcome of the negotiation with the manufacturer. The distributor has to temporarily suspend its negotiation with the retailer and to complete the negotiation with the manufacturer first.

In each of the two bilateral negotiation processes described previously, negotiation proposals and counterproposals can be sent between a pair of business entities until either an agreement is reached, or the negotiation takes too long, or one party unilaterally terminates the negotiation process. The allowable number of proposal exchanges can be controlled by each side using a termination rule or a time-out mechanism.

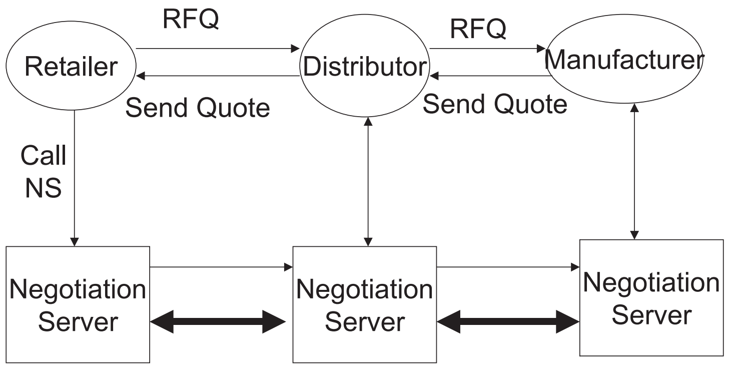

Figure 6. Two tier scenario. RFQ = request for quote; NS = negotiation server.

There are many other scenarios in a supply chain in which the automated negotiation service is needed. For example, while in the middle of a product delivery, if the distributor's transportation truck has an accident, then the delivery will not be completed on the specified date/time; hence, a new delivery date/time is to be negotiated. If the manufacturer's machinery stops because of a mechanical problem that causes a production delay, then the manufacturer will have to negotiate with the distributor on a new delivery date, which may, in turn, invoke some simultaneous renegotiations with retailers. The decisions made and the actions taken by the business entities in a supply chain during the negotiation process as well as the decision to enter negotiations with other entities are affected by the high-level policies and strategies in the same way as illustrated in the computer purchase scenario.

We have used the previous two application scenarios to demonstrate and evaluate our system. However, they do not constitute a formal validation of the model and approach presented in this article. One of the most challenging issues is how to quantitatively assign values to the goal dimensions. If the assigned goal values are not reasonable, the decision-action rules selected by the system will not produce the desirable negotiation result. The assignment of suitable values and the design of suitable decision-action rules would depend on the knowledge and experience of policymakers and negotiation experts. Negotiation is an art rather than a science. The objective of our research and development effort is to allow us to vary the goal dimension values and decision-action rules to examine their effects on the outcomes of negotiations. Needless to say, more application scenarios need to be implemented and tested to further validate our model in our follow-up work.

10. CONCLUSIONS, DISCUSSION, AND FUTURE WORK

In this article, we have presented a formal decision-making model that captures the key elements of the decision-making process in an automated e-business negotiation. The model shows how the negotiation context, defined on the basis of a business enterprise's miniworld information, can be formally mapped to a negotiation goal point in a multidimensional goal space using a negotiation policy specification. A negotiation goal or a number of goals can then be formally mapped to plans of decisions and actions using a negotiation strategy specification. A plan of decisions and actions can be specified in terms of a set of decision and action rules, which can be applied to determine the transitions of a state diagram that defines the negotiation protocol. These decision and action rules are used by a negotiation server to make initial offer, acceptance, rejection, termination, concession, withdrawal, and modification decisions. Different from the previous work in this area, we provide an approach to link high-level business concepts to low-level decision rules used by an automated negotiation server. Negotiation goals and policies are set by policymakers and the strategies and plans of decisions and actions are set by negotiation experts to achieve the goals and policies. In formulating the decision model, we have studied the "terms and conditions of sales" of many corporate Web sites and incorporated the key concepts and requirements in our model. We believe that the decision model provides a sound framework for building the decision-making components of an automated negotiation system. The automated negotiation system based on the decision-making model has been implemented and presented. The implemented system provides a desirable computing environment for testing the effect of applying and changing policies, strategies, and decision-action rules to the automated negotiation system and thus to the negotiation outcome. To verify our decision model and negotiation system we have implemented a number of business scenarios that deal with computer purchase and supply chain management.

It is clear to us that more work needs to be done in the area of dynamic concession. Concession is one of the cornerstones of business negotiations. Although we have proposed two methods for dynamic concession, we have not tested the validity of the assumptions. In the first method, we assume that the attribute under negotiation is independent of the other attributes. To reduce the complexity, it is possible (even desirable in some cases) to negotiate attributes separately. However, attributes are often negotiated together in human-based negotiations. In the second method, we assume that the counterpart's importance factor is inversely proportional to the counterpart's concession speed for that attribute. Generally speaking, the assumption is valid. For example, suppose that the counterpart is very sensitive to price; it would be safe to assume that the counterpart will make little concession on price. It would make sense for my negotiation server to assign a relatively large value to the importance factor for price. However, if the counterpart tries to "outsmart" my system by guessing and manipulating the concession algorithm used by my system, my assumption about the counterpart would be questionable. The situation will become even more complicated if my negotiation system wants to outsmart the counterpart's system, which wants to outsmart mine. In this article, we do not consider complicated situations such as this.

In an e-business negotiation environment, a buyer may negotiate with multiple sellers, and a seller may negotiate with multiple buyers concurrently. Bilateral negotiation we discussed in this article is just one "thread" in a multibilateral negotiation effort. In some negotiation situations, it is possible to carry out a multibilateral negotiation by multiple bilateral negotiations. However, many additional problems need to be dealt with due to the added complexity. The progress and the result of each bilateral negotiation need to be coordinated with those of the other bilateral negotiations. A decision to be made in one thread of the negotiation may depend on the progress or result of the other. For example, if one supplier is able to supply a large quantity of parts needed at a good price, the buyer may have to terminate some of the ongoing negotiations with other suppliers or reduce the quantity requested from other suppliers. In another case, if one or more sellers have decided to make a big concession, the buyer has the power to concede very slowly in negotiations with other sellers. Compared with bilateral negotiation, multibilateral negotiation requires more sophisticated decision-making techniques to keep track of dynamic changes and to coordinate the concurrent negotiation activities.

ACKNOWLEDGMENTS

This work is a part of the EECOMS (Extended Enterprise coalition for integrated COllaborative Manufacturing Systems; http://www.cs.umbc.edu/eecoms/) project funded by the National Institute of Standards and Technology. The project was

managed by the CIIMPLEX (Consortium for Integrated Intelligent Manufacturing PLanning and Execution) office of IBM. It focused on the research, development, and prototyping of technology to enable the integration of manufacturing applications in a multicompany supply chain planning and execution environment. The negotiation system we reported in this article is a key component of a supply chain management system developed by the consortium and has been demonstrated to many industrial companies and government agencies.

REFERENCES

[1] D. A. Lax and J. K. Sebenius, *The Manager As Negotiator: Bargaining for Cooperation and Competitive Gain.* New York: Free Press, 1986.

[2] G. Lo and G. E. Kersten, "Negotiation in electronic commerce: Integrating negotiation support and software agent technologies," presented at the 5th Ann. Canadian Operational Research Society Conf., Halifax, Nova Scotia, Canada, 1999.

[3] C. L. Karrass, *Give and Take: The Complete Guide to Negotiating Strategies and Tactics.* New York: HarperCollins, 1993.

[4] H. Raiffa, *The Art and Science of Negotiation.* Cambridge, MA: Belknap Press of Harvard University Press, 1982.

[5] I. Ståhl, *Bargaining Theory.* Stockholm, Sweden: Economics Research Institute, 1972.

[6] G. R. Shell, *Bargaining for Advantage: Negotiation Strategies for Reasonable People.* New York: Viking, 1999, pp. 156–176.

[7] L. Thompson, *The Mind and Heart of the Negotiator.* Upper Saddle River, NJ: Prentice Hall, 1998.

[8] D. G. Pruitt, *Negotiation Behavior.* New York: Academic, 1981.

[9] J. Hammer, C. Huang, Y. H. Huang, C. Pluempitiwiriyawej, M. S. Lee, H. Li, et al., "The IDEAL approach to Internet-based negotiation for e-commerce," in *Proc. Int. Conf. Data Engineering,* San Diego, CA, Los Alamitos, CA: IEEE Computer Society, pp. 666–667, 2000.

[10] C. Huang, "A Web-based negotiation server for supporting electronic commerce," University of Florida, unpublished doctoral dissertation, 2000.

[11] S. Y. W. Su, C. Huang, J. Hammer, Y. Huang, H. Li, L. Wang, Y. et al., "An Internet-based negotiation server for e-commerce," *VLDB Journal,* vol. 10, no. 1, pp. 72–90, 2001.

[12] R. Lodha, "An event-trigger-rule-based supply chain management system over the Internet," Department of Computer and Information Science and Engineering, University of Florida, unpublished master's thesis, 2002.

[13] G. E. Kersten and S. J. Noronha, "WWW-based negotiation support: Design, implementation, and use," *Decision Support Systems,* vol. 25, no. 2, pp. 135–154, 1999.

[14] A. Rangaswamy and G. R. Shell, "Using computers to realize joint gains in negotiations: Toward an 'electronic bargaining table,'" *Management Science,* vol. 43, no. 8, pp. 1147–1163, 1997.

[15] L. Lim and I. Benbasat, "A theoretical perspective of negotiation support systems," *Journal of Management Information Systems,* vol. 9, no. 3, pp. 27–44, 1993.

[16] C. Holsapple, H. Lai and A. Whinston, "Implications of negotiation theory for research and development of negotiation support systems," *Group Decision and Negotiation,* Vol. 6, no. 3, 1997, pp. 255–274.

[17] C. Holsapple, H. Lai, and A. Whinston, "Analysis of negotiation support systems: Roots, progress, and needs," *Journal of Computer Information Systems,* vol. 35, no. 3, pp. 2–11, 1995.

[18] G. E. Kersten and S. Szpakowicz, "Decision making and decision aiding: Defining the process, its representations, and support," *Group Decision and Negotiation,* vol. 3, no. 2, pp. 237–261, 1994.

[19] A. Chavez and P. Maes, "Kasbah: an agent marketplace for buying and selling goods," in *Proc. 1st Int. Conf. Practical Application of Intelligent Agents and Multi-Agent Technology,* pp. 75–90, London, Apr. 1996.

[20] C. Sierra, P. Faratin, and N. Jennings, "A service-oriented negotiation model between autonomous agents," in *Proc. 8th European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW–97),* Ronneby, Sweden, 1997, pp. 17–35.

[21] D. Zeng and K. Sycara, “Bayesian leaning in negotiation,” *International Journal of Human Computer Systems,* vol. 48, no. 1, pp. 125–141, 1998.

[22] J. R. Oliver, “A machine-learning approach to automated negotiation and prospects for electronic commerce,” *Journal of Management Information Systems,* vol. 13, no. 3, pp. 83–112, 1997.

[23] G. Dworman, S. Kimbrough, and J. Laing, “On automated discovery of models using genetic programming in game-theoretic contexts,” *Journal of Management Information Systems,* vol. 12, no. 3, pp. 97–125, 1996.

[24] S. Y. W. Su, J. Dujmovic, D. S. Batory, S. B. Navathe, and R. Elnicki, “A cost-benefit decision model: Analysis, comparison, and selection of database management systems,” *ACM Transactions on Database Systems,* vol. 12, no. 3, pp. 472–520, 1987.

[25] Y. Liu, F. Yu, S. Y. W. Su, and H. Lam, “A cost-benefit evaluation server for decision support in e-business,” *Journal of Decision Support Systems and Electronic Commerce,* vol. 36, no. 1, pp. 81–97, 2003.

[26] A. H. Soukhanov, D. A. Jost, K. Ellis, M. Severynse, M. S. Berube, J. P. Latimer, et al., *American Heritage Dictionary of the English Language*. Boston: Houghton Mifflin, 1996.

[27] M. Lee, S. W. Y. Su, and H. Lam, “Event and rule services for achieving a Web-based knowledge network,” in *Proc. 2001 Int. Conf. on Web Intelligence (WI–2001),* Maebashi City, Japan, 2001, pp. 205–216.

[28] D. Von Seggern, *CRC Standard Curves and Surfaces*. Boca Raton, FL: CRC, 1993.

[29] S. Y. W. Su, C. Huang, and J. Hammer, “A replicable Web-based negotiation server for e-commerce,” in *33rd Hawaii Int. Conf. Systems Science (HICSS–33),* Wailea, Maui, Hawaii, 2000, p. 219.

[30] H. Li, “Automated e-business negotiation: Model, life cycle, and system architecture,” Department of Computer and Information Science and Engineering, University of Florida, unpublished doctoral dissertation, 2001.

[31] S. Y. W. Su, H. Lam, R. Lodha, S. Bai, and Z. J. Shen, “Collaboration technologies for supporting e-supply chain management,” in *Applications of Supply Chain Management and E-Commerce Research in Industry,* E. Akcaly, J. Geunes, P. M. Pardalos, H. E. Romeijn, and Z. J. Shen, Eds. New York: Springer-Verlag, 2004.