Adapting Price Negotiations to an E-commerce System Scenario

Marcin Paprzycki
SWPS and IBS PAN, Warsaw, Poland
marcin.paprzycki@ibspan.waw.pl

Maria Ganzha
EUH-E, Elblqg and IBS PAN, Warsaw, Poland
ganzha@euh-e.edu.pl

Abstract

In our work we have proposed an agent based e-commerce system in which autonomous agents participate in price negotiations. Recently, our work reached the stage when the main ideas behind price negotiations had to be reevaluated. The aim of this paper is to report how we adapt price negotiations mechanisms to match proposed systems characteristics.

1. Introduction

Currently, we are developing a complete model agent-based e-commerce system ([1, 2]), in which buyer agents attempt at making purchase by participating in price negotiations in e-stores. At the same time e-stores attempt at maximizing profits, resulting from product sales. Within the project we have devoted some work to price negotiations ([2])and even implemented a price negotiation subsystem based on [4]. However, since the system has been re-designed after the initial implementation, our work on price negotiations has been pursued separately from the design of the system itself. It is only now, when a number of specific decisions about the system design have been made, when we can re-evaluate our approach to price negotiations.

Let us note that we understand price negotiations as a process by which agents come to an agreement on a price ([5]). Furthermore, we distinguish between *negotiation mechanisms* that define "rules of encounter" between participants; and *negotiation strategies* that specify behavior of participants aiming at achieving a desired outcome (typically, to maximize "gains"). In this context, *auctions* are one of the most popular and well-understood forms of automated negotiations ([6]) and recently attempts to parameterize the auction design space have been observed (see summary of research results presented in [2]). Among them authors of [4] proposed a complete framework comprising of: (1) negotiation infrastructure: that defines roles of negotiation participants and of a host, (2) a generic negotiation protocol: that defines the three phases of a negotiation: admission, exchange of proposals and formation of an agreement, in terms of how, when and what types of messages should be exchanged between the host and participants, and (3) taxonomy of rules used for enforcing the negotiation mechanism, consisting of: rules for admission of participants to negotiations, rules for checking the validity of negotiation proposals, rules for protocol enforcement, rules for updating the negotiation status and informing participants, rules for agreement formation and rules for controlling the negotiation termination. Finally, they introduced a *negotiation template* that contains parameters specific to a given form of price negotiations. With a large number of results concerning software agents appearing in the context of price negotiations, let us delineate what makes our approach unique in the context of this paper.

1. Typically, price negotiation of a single (collection of) item(s) is contemplated. Once the negotiation is over, agents that participated in it complete their work. We are interested in a more realistic scenario when a number of items of a given product are placed for sale one after another.

2. Since a sequence of items is sold we treat price negotiations as a "discrete process"in which buyers are "collected" and released together in a group to participate in a price negotiation. While the negotiation takes place buyer(s) communicate only with seller(s). At the same time the next group of buyers is collected (as they arrive) for the next negotiation.

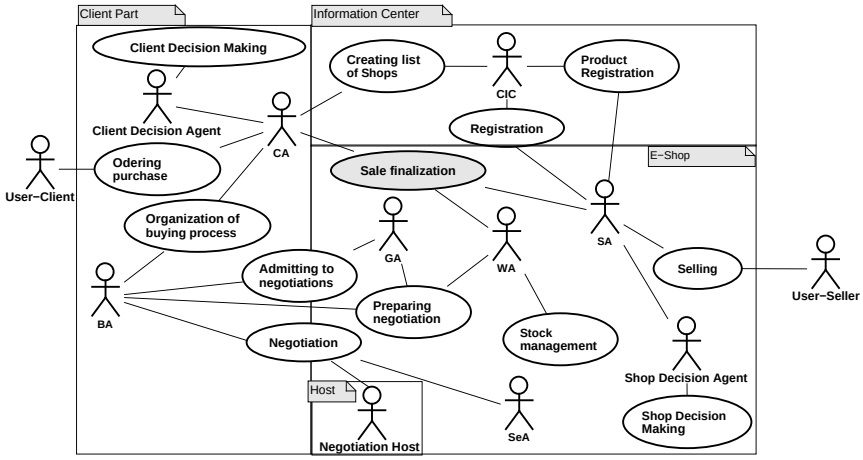

Figure 1. Use Case diagram of the proposed agent-based e-commerce system

3. Since multiple subsequent auctions (involving the same product) take place, price negotiation mechanisms can be dynamically changed. For instance, first 17 items may be sold using English Auction, while the next 42 using fixed price.

The remaining part of this paper is devoted to the effects that these features have on one-to-many-type price negotiations. In the next section we briefly describe the agent system under development. Next, we discuss our conceptualization of (1) single-item English and Dutch auctions, (2) multi-item Dutch auction, and (3) sealed bid auctions.

2. System Description

Our system acts as a distributed marketplace in which e-shops sell products to e-buyers. In Figure 1 we present Use Case diagram of the system. Outside of its bounds we can see *User-Client* who will attempt at buying products and *User-Seller* who tries to sell products in her e-store.

Let us now summarize functions of most important agents in the system (for a complete description see [1, 2]). *User-Client* is represented by the *Client Agent* (*CA*). The *CA* is assumed to be autonomous. When the desire to purchase product *P* is communicated by the *User-Client*, it either purchases it or, abandons purchase (e.g. because prices are to high). The *CA* communicates with the *Client Information Center* (*CIC*) agent which contains information which e-stores sell which products. For each store that sells the requested product, the *CA* delegates a single *Buyer Agent* (*BA*) to participate in price negotiations and if successful, possibly attempt at making a purchase (successful price negotiations result in a product reservation for a specific time period). Multiple *BA*s representing the same *CA* can win price negotiations and report back; then the *CA* makes the decision if either of available offers is good enough to make a purchase. *Buyer Agents* can participate in negotiations only if the *Gatekeeper Agent* (*GA*) admits them—if they are trusted. The *GA* is created by the *Shop Agent* (*SA*), who is the central manager of the e-shop. Facilitating the selling process, the *SA* utilizes the (*GA*), as well as a *Warehouse Agent* (*WA*) that is responsible for inventory and reservation management; and a set of *Seller Agents* (*SeA*) that negotiate price with incoming *BA*s.

3. Negotiations—general considerations

In our work we follow the proposal set in [4]. However, the way that our systems and its processes are structured allows for certain simplifications. Furthermore, it allows us to conceptualize multiple forms of price negotiations—depicted in Table 1. In [4] three roles have been proposed: *buyer*(s), *seller*(s) and the *host*. While in that paper (and the code), and in our earlier work [1, 2], distinction between the *host* and the *seller* was blurred, here we make it clear. The negotiation *host* provides the *infrastructure* where the negotiations take place and are managed. The *host* can be facilitated locally by the e-store, or can be located within a certified authority (to assure that the negotiation is not tainted—for instance, in a multi-item Dutch auction it is enough to "pretend" that two messages arrived in a reverse order to favor a one participant over another). Furthermore, we assume that the *host* is a generic infrastructure that can handle any form of price negotiations.

Table 1. Some of more popular forms of auctions

Auctions	Rules
English auction	Seller announces reserve price or a low opening bid. Bidding increases progressively until demand falls. Winning bidder pays highest valuation.
Dutch auction	Seller announces very high opening bid. Bid is lowered progressively until demand rises to match supply.
First-price, sealed bid	Bids submitted in written form with no knowledge of bids of others. Winner pays the exact amount he bid.
Vickrey auction or second-price sealed bid	Bids submitted in written form with no knowledge of the bids of others. Winner pays the second-highest amount bid.
Discriminatory auction	A multiunit auction in which every winning bidder pays a different price which depends on the actual bid placed by each winning participant.
Uniform-price auction	A multiunit auction in which every winning bidder pays the same price, which may or may not be equal to the participants' bids

Table 2. Parameters passed in the template

Parameter name	Description	Auction type
N_ID	Negotiation ID	all auctions
P_ID	Product ID	all auctions
N	Number of products	all multi-item auctions (Discriminatory, Dutch multi-item, Uniform-price)
Delta	Minimal price increment/decrement	English, Dutch (single- and multi-item) auctions
Pr_{min}	Minimal reserved Seller price	all auctions
NP_{min}	Minimal number of participants	Dutch multi-item auction
T	Time parameter (e.g. inactivity or sending proposals)	all auctions
T_{Buyer}	Time of Buyers inactivity, where $0 < T_{Buyer} < T$	all Dutch auctions

In *all* negotiations considered in this paper, the negotiations process is started by the *SeA* that sends to the *host* the negotiation template, containing all necessary parameters, and the list of participants. This information is used to initialize an *instance* of a *host*. This involves initialization of appropriate objects and rule-modules (e.g. modules specific to an English auction [2]). Upon completion of initialization the *host* informs the *SeA* that it is ready. In this way, in turn, the *SeA* knows that the host initialization process has been successfully completed. This process is enclosed in the box *Negotiation initialization* in subsequent Action diagrams.

Recall that *Buyer Agents* are admitted into negotiations in groups, and the *GA* assures that only *BA*s interested in a given product are released to a given negotiation. As a result it is not necessary to check the "product name" during the negotiation process. It could be also possible to assume that it is not necessary to check if a given agent is allowed to participate in a given negotiation. Observe that negotiations are handled by one of the pool of *SeA*s (unknown in advance) and an unknown in advance instance of the *host*. Therefore, only participants of a given negotiation know where to send their messages (bids) to. However, in the case of a multi-item Dutch auction, winning *BA*s are not allowed to bid again. Therefore, for the sake of uniformity, we have decided that validity of sending a proposal to a given negotiation by a given participant has to be checked in all auctions. Unfortunately, due to the lack of space we omit further details related to rule-based mechanisms and focus our attention on the UML-introduced structure of negotiations.

Let us now present, in Table 2, the complete list of parameters passed (in the template) to specific auctions. Note that (1) the Negotiation ID specifies auction to be executed (e.g. Dutch or Vickrey), (2) reason for two timing parameters is related to the fact that it is sometimes assumed that in a Dutch auction there is a certain time within which the *seller* cannot decrement price and within that time buyers can submit their bids.

4. Single-item English and Dutch auctions

Technically, an English auction is a single-item, first-price, open-cry, ascending auctions ([2]), in which a single item (or a collection of products treated as a single item) is sold by a single *seller* to multiple *buyers* bidding for it. A new bid must be higher than the currently highest bid plus a minimal bid increment. All bids are visible to all participants. There are two ways of ending an English auction. In the "eBay model," there is a specific time limit (e.g. 3:00 pm on Sunday, January 7th, 2007). In a classical scenario there is a window of inactivity (e.g. no bids posted within 10 minutes) that ends the negotiations and our system can handle both of then (see in Figure 2). Finally, in the

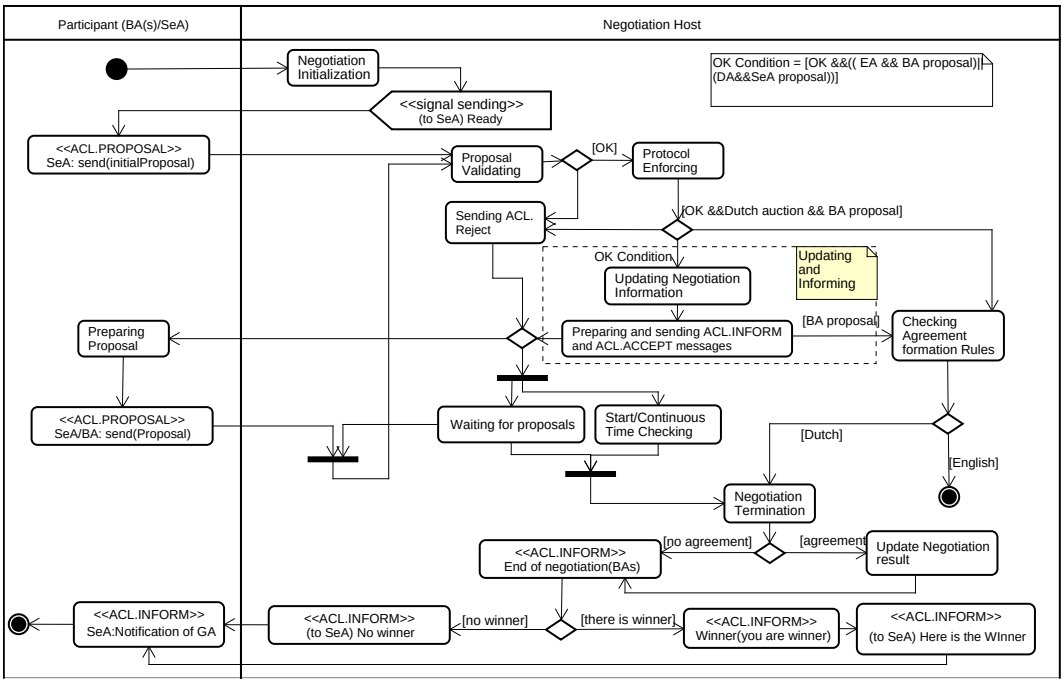

Figure 2. Activity Diagram of single-item English and Dutch auctions

English auction there is usually a seller reservation price that must be met by the winning bid.

In a number of e-commerce sites different auctions are dubbed Dutch auction (see [3] for a discussion). In our work we utilize the "classical definition" in which we have a single *seller* and multiple *buyers* that compete for a single item (or a collection of products sold as a single item). The *seller* starts bidding from a high price and subsequently decreases it. Each consecutive bid has to be smaller by at least a minimal bid decrement. The end of a Dutch auction involves either an acceptance of the proposal by a buyer, or by seller reaching its reservation price. The latter is represented by the seller not submitting another proposal for a specific "time of inactivity," which when expires terminates the negotiation. In Figure 2 we see the Activity diagram that serves both single-item English and Dutch auctions.

After negotiation initialization the *host* informs the *SeA* that it is ready. Interestingly enough, in both forms of auctions it is the *SeA* that "starts" the process. In the case of an English auction it submits the starting price. In the case of a Dutch auction it submits an initial price. Either price is submitted as a proposal and validated by the system. Note that if they do not pass, the *host* sends a REJECT message and awaits for the correct initial proposal; if such proposal does not come then the inactivity constraint terminates the negotiation. Upon successful validation an initial proposal is posted on the blackboard as an active proposal, and a message to all *BA*s is send (informing them about the blackboard update) and a timer is re-started. Notice that the timer can either establish fixed time of ending the negotiation (eBay model of English auction) or count time of inactivity (remaining auctions). The *host* then awaits proposals. In an English auction *BA*s send bids that, if validated, become posted on the blackboard (and the agreement is prepared—see the *Checking Agreement Formation Rules* box—to be used in case if this bid will be the accepted one—thus only bids that are above the reservation price trigger agreement preparation). In the Dutch auction the proposal can come from the *SeA* and contain the decremented price that, after validation, becomes posted on the blackboard, or from the *BA*, which after successful validation, result in negotiation termination. Note that validation of proposals would eliminate proposals submitted by the *SeA* too early (violating the T_{Buyer} constraint). Upon negotiation termination the results is established and information is distributed to all participants describing their outcome.

5. Multi-item Dutch auction

In a multi-item Dutch auction a single *seller* attempts at selling *N* units of product (*P*) to multiple *buy-*

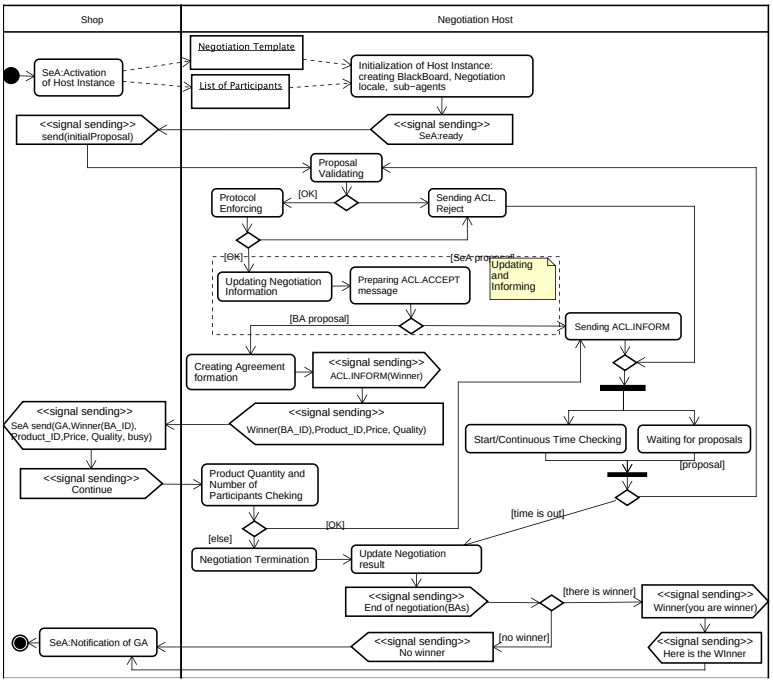

Figure 3. Activity Diagram of Dutch multiple-item auction

ers. It starts bidding at a high price and subsequently reduces it. At a certain moment one of buyers informs that it is ready to purchase M units of P (where $M \leq N$) at a price equal to the current bid. At this stage, the number of available units is reduced by M and the winning *BA* is removed from the negotiation to process its reservation (we assume that each buyer is allowed to submit at most one successful bid). Specifically, its ID is removed from the list of negotiation participants and this prevents it from bidding again. If $N - M > 0$, process continues until either (1) all N units are sold, or (2) there are no more buyers left, or (3) the minimal reservation price of the seller is reached (time of inactivity condition will kick-in). In Figure 3 we present a UML Activity Diagram of a multi-item Dutch auction, which is very similar to that in Figure 2. There are two main differences. First, when one of *BA*s is a winner (submitted a bid for M items of a given product). This fact is confirmed to both: the *BA* and the *SeA* and suspends processing in the *host*. The reason is that it is possible that all products have been sold, or that there are no more *buyers* left and we do not want the negotiation termination to take place at the same time when the *SeA* is communicating with the *GA*. The *SeA* sends a message to the *GA* containing all necessary information about the winner and afterward signals the *host* that it can continue. Second, after a successful bid, termination rules related to the number of products left and to the number of remaining participants are checked and a message to all remaining *BA*s is send (informing them about the changes). Observe that if the original proposal was from the *SeA* and contained a price decrease then on the blackboard would be updated and message would also be send to all *BA*s. Rules for time-based negotiation termination are checked in the same way as in single-item auctions.

6. Sealed bid auctions

In Table 1 we have presented brief definitions of four such auctions: (1) First-price, sealed bid, (2) Vickrey auction or second-price sealed bid, (3) Discriminatory auction, and (4) Uniform-price auction. From the point of view of our current interest, all these forms of auctions have the same structure, which we have depicted in an Activity diagram presented in Figure 4. The only difference is in the way that the agreement formation rules (a) select the winner/winners, and (b) establish the winning price. While we have listed four sealed bid auctions, we believe that the same schema applies to all of them. This, combined with composition of JESS modules taking place during initialization of a *host instance* allows us to easily support all of them in our system.

After the *negotiation initialization* the *host instance* sends a call for proposals to all participants—

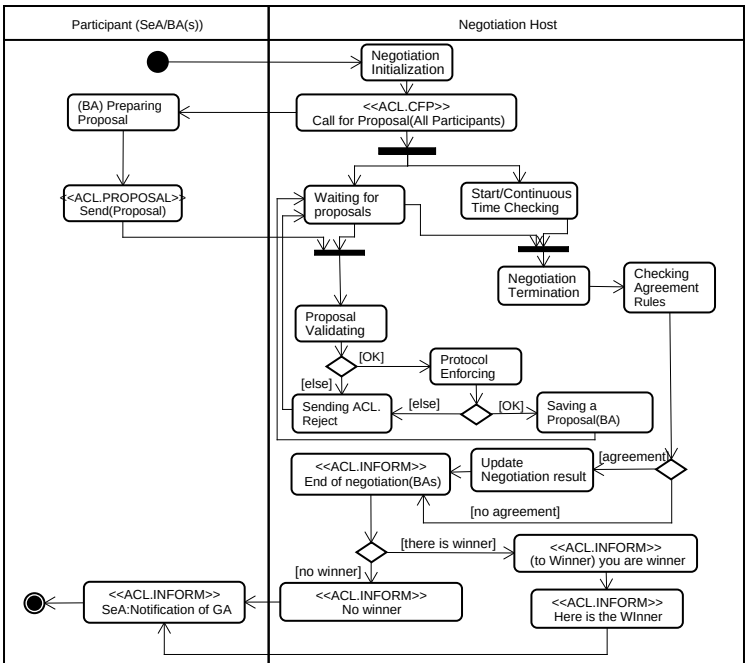

Figure 4. Activity diagram of Sealed bid auction

this includes the *SeA*. In this way the *SeA* learns that the initialization has been successful. Proposals from *BA*s are collected until a timer forces negotiation termination, which in turn results in agreement formation rules selecting winner(s) (depending on the form of a sealed bid auction). At the final stage all necessary information is passed to the winner(s) and to the *SeA* that notifies the *GA* accordingly. Notice that this part of the process matches that found in the remaining two auction mechanisms. This observation is relatively important as it shows that all mechanisms have a certain common structures (related to initial and final stages of negotiations) that can be turned into re-usable modules shared by all of them.

7. Conclusions and Future Work

In this paper we have described how the price negotiation mechanisms should be modified to match the e-commerce scenario that is realized in our agent-based e-commerce system. We have illustrated, through UML diagrams, detailed mechanisms of single item English and Dutch auctions (with an English auction having two different ending conditions), a multi-item Dutch Auction as well as multiple sealed bid auctions. We have shown that all sealed bid auctions share the same UML structure. Furthermore, all eight auctions share the same negotiation initialization and negotiation closing mechanisms. These observations are relatively important in the context of auction implementation (code re-use and modularity), which is currently taking place.

References

[1] Bădică, C., Bădită, A., Ganzha, M., Paprzycki M.: Developing a Model Agent-based E-commerce System, in: Jie Lu et. al. (eds.) E-Service Intelligence–Methodologies, Technologies and Applications, Springer, Berlin, 2007, 555–578

[2] Bădică, C., Bădită, A., Ganzha, M., Paprzycki M.: Implementing Rule-Based Automated Price Negotiation in an Agent System. J. of Universal Comp. Sci., to appear

[3] Bădică, C., Ganzha, M., Gawinecki, M., Kobzdej, P., Paprzycki, M.: Utilizing Dutch Auction in an Agent-based Model E-commerce System, in: Proceedings of the 14th International Enformatika Conference, World Enformatika Society (2006) 7–12

[4] Bartolini, C., Preist, C., Jennings, N.R.: A Software Framework for Automated Negotiation, in: *Proc. of SELMAS'2004*, LNCS 3390, Springer-Verlag (2005) 213–235

[5] Lomuscio, A.R., Wooldridge, M., Jennings, N.R.: A classification scheme for negotiation in electronic commerce. In: F. Dignum, C. Sierra (Eds.): *Agent Mediated Electronic Commerce: The European AgentLink Perspective*, LNCS 1991, Springer Verlag (2002) 19–33

[6] Wurman, P.R., Wellman, M.P., Walsh, W.E.: A Parameterization of the Auction Design Space. In: *Games and Economic Behavior*, 35, Vol. 1/2 (2001) 271–303