Design Considerations for a Negotiation Component in a Model E-commerce Agent System

Costin Bădică, Gabriel-George Popa, Mihnea Scafeş
University of Craiova, Software Engineering Department
Bvd.Decebal 107, Craiova, 200440, Romania
badica_costin@software.ucv.ro

Maria Ganzha, Maciej Gawinecki, Paweł Kobzdej, Marcin Paprzycki
Systems Research Institute Polish Academy of Science
ul. Newelska 6, Warsaw, Poland

Abstract

In our recent work we have established that to develop an extensible agent-based e-commerce system it is necessary to logically separate the negotiation process from other parts of the system. This note deals with the design of the "negotiation box"—the component that actually runs the negotiation processes. Furthermore, interaction of the negotiation box with the rest of the system is addressed.

1. Introduction

In the model e-commerce multi-agent system that we are currently developing (see [7] and references collected there for the complete description of the system), price negotiations are a key aspect that deserves special attention. The aim of this note is to outline requirements and discuss solutions for the part of the system responsible for carrying out price negotiations—the *negotiation box*. We start by discussing the negotiation model, protocol and rules. We follow with some examples of actual rules used in representing English and Dutch auctions and conclude by pointing to future work.

2. Negotiation Box

2.1. Negotiation Model

The starting point of our work was the negotiation framework proposed in [4]. Its authors sketched a general framework for implementing price negotiations. There, an abstract negotiation process comprises: a *negotiation infrastructure*, a *generic negotiation protocol* and a *taxonomy of negotiation rules*. The *negotiation infrastructure* defines roles involved in the negotiation process: *Participants* and a *Negotiation Host*. *Participants* are usually *Buyers* and *Sellers* that exchange proposals. Such an exchange is mediated by the *Negotiation Host* (there is no direct exchange of messages between participants) and governed by a *generic negotiation protocol* that defines how and when messages can be exchanged. Therefore, all knowledge required to *facilitate* negotiations is stored within the *Negotiation Host*, which can to control and coordinate negotiations. It is also proposed that this knowledge should be captured as groups of *negotiation rules* responsible for: (1) *checking the validity of negotiation proposals*, (2) *protocol enforcement*, (3) *updating negotiation status and informing participants*, (4) *agreement formation*, and (5) *controlling the negotiation termination* [4]. Negotiation rules are represented and interpreted by a JESS inference engine ([1]). Note that details of the particular negotiation (specific values for parameters that are part of negotiation rules) are provided in a form of a *negotiation template*.

2.2. *Seller Agent*

While, in the above described general framework, roles of the *Seller* (representing the *Shop* and the *Negotiation Host* (an objective negotiation manager) are *conceptually different*, in the present work, for efficiency reasons we have decided to merge them within the same agent—the *Seller Agent*, while a potential conflict of interest has been solved by a clear separation of both roles inside of the agent.

In order to realize its aims, the *Negotiation Host* role incorporates all member objects responsible for checking negotiation rules: *Proposal Validator*, *Protocol Enforcer*, *Information Updater*, *Negotiation Terminator* and *Agreement Maker*. According to the design proposed in [5, 6] the *Negotiation Host* also incorporates two member objects representing the *Negotiation Locale* and the *Blackboard* (see Figure 1), *Negotiation Locale* and *Blackboard* "boxes". The *Negotiation Locale* object holds the *negotiation template* that defines negotiation parameters [4] and the list of participant *Buyer Agents* that were admitted to a given negotiation. The *Blackboard* object encapsulates a JESS rule engine that is initialized with negotiation rules. To improve overall efficiency of the system, we use a single JESS rule engine that is shared by all member objects within each *Negotiation Host* (rather than implementing each object that checks negotiation rules, as a separate rule engine). Since extra attention has to be paid how rules are executed, we decided to utilize JESS modules for partitioning rules and facts managed by the rule engine. Specifically, one JESS module stores blackboard facts and a separate JESS module stores rules used by each member object of the *Seller Agent* (see [5] for details).

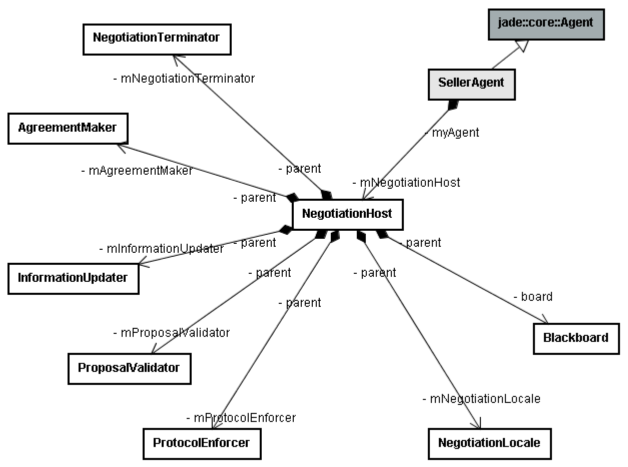

Figure 1. The class diagram showing the structure of the *Seller Agent*

2.3. Generic Negotiation Protocol

The *generic negotiation protocol* imposes a minimal set of constraints on sequences of messages exchanged by the *Host* and *Participants* during a negotiation. Since it is further specialized using negotiation rules, we can conceptually consider that a negotiation mechanism is defined by the following "metaphorical equation": *negotiation mechanism = generic negotiation protocol + specific negotiation rules*.

The *Negotiation Host* coordinates interactions of negotiation participants by managing a negotiation process by facilitating following negotiation activities: (1) proposal (or bid) submission, (2) informing participants about the change of state of negotiation, (3) agreement formation and (4) negotiation termination.

Proposal submission. *Buyer*/*Seller* (depending on the form of price negotiation) will enter the phase of submitting bids after the negotiation was initiated (a number of *Buyer* agents is "simultaneously" released by the *Seller* that sends them a message). The generic negotiation protocol states also that a participant will be notified by the *Seller* if its proposal was either accepted (with an ACL ACCEPT-PROPOSAL) or rejected (with an ACL REJECT-PROPOSAL).

Informing participants. The negotiation protocol requires that participants will always be notified (with ACL INFORM messages) about any new state of the negotiation.

Agreement formation can be triggered at any time during the negotiation. When agreement formation rules signal that an agreement was reached, the protocol states that all participants involved in the agreement will be notified by the host with ACL INFORM messages.

Negotiation termination can also be triggered at any time during the negotiation. When negotiation termination rules signal that the negotiation process reached its final state, the protocol states that all participants will be notified by the *Host* with ACL INFORM messages.

2.4. Negotiation Rules

Enforcement of negotiation rules involves complex activation patterns that are, of course, dependent on particular negotiation mechanism. For instance: (i) rules for proposal validation and protocol enforcement are fired whenever a new bid is received; (ii) acceptance of a bid usually triggers process of updating negotiation status and informing other participants; (iii) optionally, acceptance of a bid might trigger an agreement formation (e.g. in the case of a Dutch Auction); (iv) optionally, agreement formation might trigger negotiation termination (e.g. in the case of an English Auction); (v) timing events (signalling that a specific amount of time elapsed) can optionally trigger negotiation termination or update of the negotiation state (consider an auction comprising an English Auction followed by a Vickrey Auction involving only the three highest bidders from the initial stage [9], where a timing event could signal end of the first stage and beginning of the second stage) and consequently result in informing other participants; (vi) sometimes, negotiation termination might optionally trigger agreement formation (e.g. in the case of an English Auction).

3. Examples

English Auctions. Technically, English Auction is a single-item, first-price, open-cry, ascending auction ([8]). In an English Auction there is a single item (or a collection of items treated as a single item) sold by a single *seller*, and multiple *buyers* bidding against each other to buy it. Usually, (1) there is a time limit for ending the auction, (2) an initial price, (3) a seller reservation price that must be met by the winning bid, (4) a minimum bid increment (to be accepted, a new bid must be higher than the currently highest bid plus that minimum bid increment), and (5) all bids are visible to all participants. Let us now consider a few sample rules for representing an English Auction. We describe them informally using a pseudo-code notation that is independent of any implementation-level language (like JESS).

ENGLISH-POSTING-BUYER rule specifies that a *buyer* participant can post a proposal whenever there is an offer already posted by a *seller* participant. A proposal is called *valid* if it is syntactically well-formed and semantically compliant with the negotiation template. A proposal is called *posted* if it can be posted—depending on the type of proposals that were previously posted by other participants (i.e. if negotiation reached a state that allows the proposal to be posted).

Rule 1 ENGLISH-POSTING-BUYER (*Protocol Enforcer*)

```
IF
    There is a valid proposal $Pr$ of a participant with role 'buyer' and
    There is an active proposal of a participant with role 'seller'
THEN
    Proposal $Pr$ is posted
```

ENGLISH-IMPROVEMENT-BUYER rule specifies that a *buyer* participant must post a proposal that satisfies the minimum bid increment condition. Note that a proposal that passed the improvement tests is called *active*.

Rule 2 ENGLISH-IMPROVEMENT-BUYER (*Protocol Enforcer*)

```
IF
    Negotiation is on goods $A$ and
    Bid increment is $Inc$ and
    Currently highest bid is $B$ and
    Proposal $Pr$ on goods $A$ with price $P$ was posted by a buyer and
    $P > B + Inc$
THEN
    Proposal $Pr$ is active
```

ENGLISH-AGREEMENT-FORMATION rule specifies that when the agreement formation is triggered, if the currently highest bid is greater than the *seller* reservation price, an agreement is formed between the submitter of the highest bid and the *seller*.

Rule 3 ENGLISH-AGREEMENT-FORMATION (*Agreement Maker*)

```
IF
    The currently highest bid is $B$ and was submitted by buyer $S1$ and
    There is an active proposal of seller $S2$ with price $P$ and
    Negotiation is on goods $A$ and
    $B \geq P$
THEN
    An agreement of $S1$ with $S2$ to transact goods $A$ at price $P1$ is formed
```

Dutch Auctions. We have chosen the following conceptualization of a Dutch Auction ([6]): "In a Dutch auction, bidding starts at an extremely high price and is progressively lowered until a buyer claims an item by {...} pressing a button {...}. When multiple units are auctioned, normally more takers press the button as price declines. In other words, the first winner takes his prize and pays his price and later winners pay less. When goods are exhausted, the bidding is over.". Note that when *Buyer* becomes a winner it cannot return to the same negotiation. Technically, we describe this type of auction as: single-item multi-unit (homogenous) discriminatory, descending one. Observe that this conceptualization is in accordance with the FIPA Dutch Auction Interaction Protocol [3]. It can be observed that the FIPA Dutch Auction Interaction Protocol is under-specified (a similar observation concerning the FIPA English Auction Interaction Protocol was made in [4]), as it does not specify important parameters like: (a) how often is the *Seller* allowed to shout the price? for example, shouting prices "very fast" may not leave enough time for the *Buyers* to "think" if to bid or not; (b) how much must the *Seller* decrement the price in the next bid? for example, an auction might require a minimum value for this decrement (similarly to the minimum increment in an English Auction); (c) how much time without any activity is allowed before terminating the auction? note that it might happen that neither *Buyers* bid nor the *Seller* wants to decrement the price, and in such a case the auction must be terminated (even if the *Seller* didn't sell all the inventory).

Therefore, we have added the following parameters to the specification of Dutch Auction: i) minimum value by which the *Seller* must decrement the price at each announcement: $H > 0$; ii) Minimum time limit that the *Seller* must wait before issuing the next announcement: $Tm > 0$; iii) Maximum time window of inactivity in the auction that, when observed, will terminate the auction: $Tw > 0$; obviously, the following must hold: $Tm < Tw$. Additionally, number of units N of the product to be sold is also placed in the negotiation template. This value is decremented with the number items "sold" whenever a successful bid is received from a *Buyer*.

While parameters of the negotiation mechanism together with negotiation rules are public, i.e. known to all participants, each participant may also use a private strategy that dictates how it should act during the negotiation. In a Dutch Auction *Buyer* strategy dictates when exactly the *Buyer* should accept price shouted by the *Seller*. It could be extremely simple and require acceptance of shouted price that falls below a given threshold. It could also be rather complex and involve passage of time, number of still available items and/or speed with which items are being purchased by other *Buyers*. On the other hand, *Seller* strategy could contain following parameters: i) initial price X_0, ii) timing R_i of reducing price at each new shout i (variable at each shout);

the following condition must hold: $Tm \leq R_i \leq Tw$ for all i to assure that rules are not violated and negotiation does not terminate; iii) reservation price $Xres$; *Seller* will stop bidding if the price reaches a value below this limit.

Let us present sample rules that are a part of our representation of the Dutch Auction. We define: *Pr*—proposal (or bid); $X(Pr)$—price, $T(Pr)$—time when the proposal has reached the *Negotiation Host*, and $M(Pr)$—number of units "accepted" for purchase.

The *Seller* is allowed to shout a new price when one of the following conditions holds: (i) it is the first shout; (ii) one of *Buyers* submitted a bid that was successful; (iii) no successful bids were received and at least Tm time units elapsed since the last shout. DUTCH-POSTING-SELLER rule checks the third of these conditions. Note that when the *Seller* offer passes all checks it becomes *posted*.

Rule 4 DUTCH-POSTING-SELLER (*Protocol Enforcer*)

```
IF
    There is a valid proposal $Pr$ submitted by the 'Seller' participant **and**
    'Buyer' participants didn't submit successful bids since the last shout **and**
    There is an active proposal $Pr_1$ from the 'Seller' participant **and**
    $T(Pr) - T(Pr_1) \geq Tm$
THEN
    Proposal $Pr$ is posted
```

A *posted* offer that is not the first offer must also satisfy the improvement tests in order to become *active*. This is realized by the DUTCH-IMPROVEMENT-SELLER rule that asks for the shouted price to be at most equal to the last shouted price minus a template-specified decrement. When this condition holds, the seller offer becomes *active*.

Rule 5 DUTCH-IMPROVEMENT-SELLER (*Protocol Enforcer*)

```
IF
    Participant with role 'Seller' has posted proposal $Pr$ **and**
    Value of last offer posted by 'Seller' is $B$ **and**
    Minimum decrement is $H$ **and**
    $X(Pr) \leq B - H$
THEN
    Proposal $Pr$ is active
```

Note that the Dutch Auction rule-based description involves also rules for controlling when a *Buyer* bid may be posted and accepted (see [6]). When a *Buyer* bid becomes active, it will result in: (i) triggering rules for informing participants (e.g. *Buyers* how many units are still available for sale); (ii) triggering rules for agreement formation and negotiation termination. An agreement formation DUTCH-AGREEMENT FORMATION rule looks for an active *Seller* offer and an active *Buyer* bid and generates a deal by matching those two proposals.It also updates number of units available for sale.

Since the agreement formation rule updates number of items still available for sale, a negotiation termination rule has to be activated. It checks if there are any items left and if they are none the negotiation is terminated.

Rule 6 DUTCH-AGREEMENT FORMATION (*Agreement Maker*)

```
IF
    There is an active bid submitted by a *Buyer* participant $Par_1$ **and**
    There is an active offer shouted by a *Seller* participant $Par_2$
THEN
    An agreement between $S_1$ and $S_2$ to transact according to $X(Par_1)$ is
    formed **and** Available quantity $N$ is updated
```

4. Conclusions and Future Work

This note discussed design of a negotiation component of a multi-agent e-commerce system. We focused our attention on: (i) interface of the negotiation component with the rest of the system; (ii) negotiation model and protocol; (iii) rule-based representation of negotiation mechanisms. Our current work involves implementation and integration of the negotiation component into our system.

Acknowledgment. Work of M. Ganzha, M. Gawinecki, P. Kobzdej and M. Paprzycki has been partially sponsored by the M. Curie IRG grant, project E-CAP.

References

[1] JESS: Java Expert System Shell. http://herzberg.ca.sandia.gov/jess/.

[2] Agorics: http://www.agorics.com/Library/Auctions/.

[3] FIPA: Foundation for Physical Agents. http://www.fipa.org.

[4] C. Bartolini, C. Preist, and N. R. Jennings. A software framework for automated negotiation. In R. Choren, A. F. Garcia, C. J. P. de Lucena, and A. B. Romanovsky, editors, *Software Engineering for Multi-Agent Systems III, Research Issues and Practical Applications [the book is a result of SELMAS 2004].*, volume 3390 of *Lecture Notes in Computer Science*, pages 213–235. Springer, 2005.

[5] C. Bădică, A. Bădiţă, M. Ganzha, A. Iordache, and M. Paprzycki. Implementing rule-based mechanisms for agent-based price negotiations. In H. Haddad, editor, *SAC '06: Proc. of the 2006 ACM Symposium on Applied Computing*, pages 96–0100, New York, NY, USA, 2006. ACM Press.

[6] C. Bădică, M. Ganzha, M. Gawinecki, P. Kobzdej, and M. Paprzycki. Utilizing dutch auction in an agent-based model e-commerce system. In *International Enformatika Conference, IEC'06*, pages 101–107, 2006. (to appear).

[7] C. Bădică, M. Ganzha, and M. Paprzycki. *Developing a Model Agent-based E-commerce System*. Springer Verlag, 2006.

[8] K. Laudon and C. Traver. *E-commerce. business. technology. society (2^{nd} ed.).* Pearson Addison-Wesley, 2004.

[9] D. Rolli and A. Eberhart. A descriptive auction language. *Electronic Markets – The International Journal*, 2005.

[10] M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons, 2002.