An Implementation Model of a Declarative Framework for Automated Negotiation

Laura Surcel

University of Craiova
Blvd. Decebal, nr. 107, RO-200440, Craiova, Romania
laura_surcel@yahoo.com

Abstract

The subject of automated negotiations has received a lot of attention in the Multi-Agent Systems (MAS) research community. Most work in this field on the auction design space, on its parametrization and on mechanisms for specific types of auctions. One of the problems that have been recently addressed consists in developing a generic negotiation protocol (GNP) capable of governing the interaction between agents that participate in any type of auction. Though much has been said on this matter, the current results stop at the XML representation of specific negotiation mechanisms. In this paper we propose a declarative approach for specifying a generic auction protocol by using Belief-Desire-Intention (BDI) agents and the Jason programming language to represent the entities that communicate in an auction. In order to validate the claim on the generality of the proposed approach we have used the GNP to model two negotiation mechanisms: one for the English auction and one for the Dutch auction.

1998 ACM Subject Classification I.2.11 Distributed Artificial Intelligence

Keywords and phrases Auctions, automated negotiations, multi-agent systems, Jason, generic negotiation protocol

Digital Object Identifier 10.4230/OASIcs.ICCSW.2012.129

1 Introduction

Generally speaking, a negotiation is a bargaining (give or take) process between two or more parties (each with its own aims, needs and viewpoints) seeking to discover a common ground and reach an agreement to settle a matter of mutual concern or resolve a conflict[1]. Negotiation is employed in many areas, such as: law, business, air and marine traffic management, trade, parenting, hiring a.o.

A notable subfield of negotiations is represented by auctions which are also the focus of the present paper. An auction is a process of exchanging possessions through buying and selling by offering them up for bid. A person or an organization may use auctions to sell goods or services by making them available to the public, setting an initial price and then receiving offers. In the general case the winner of the auction is the buyer that makes the greatest bid.

The importance of this subject is supported by the popularity and the use of hundreds of web sites that have transformed auctions into an open process, in which thousands of items may be offered for bidding by anyone from anywhere at any time[2]. All these sites offer the

[1] http://www.businessdictionary.com

[2] The five most reliable, renowned and diversified online auction sites with respect to user options, according to a 2012 survey conducted by TopTenReviews (http://online-auction-sites.toptenreviews.com/), are: eBay, WebStore, eBid, OnlineAuction and OZtion

2012 Imperial College Computing Student Workshop (ICCSW'12).
Editor: Andrew V. Jones; pp. 129–134
OpenAccess Series in Informatics
OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
ICCSW

possibilities of selling and buying which must be performed by human operators. This last observation leads to some interesting questions that form the starting point of the current research: is there a way to automate the negotiation process? Can negotiations be carried out between computers with minimum human input? And can such a process be universally applied to any kind of auction?

Researchers [1, 2] that tried to answer these questions reached the following results: any negotiation can be considered as an interaction between a mechanism and a strategy. The mechanism (or protocol) is a set of rules that must be followed by participants in order to communicate and it is public, while the strategy describes the behavior of a negotiation participant and it is used for reaching his or her private goals [1]. Although many models of automated negotiations such as: bargaining, auctions[4, 5], multi-commodity negotiations a.o. have been brought forward, this paper discusses the model proposed in [1] and describes a possible extension of the GNP from an English auction to a Dutch auction using the Jason agent programming language [6].

Our choice is justified by the fact that [1] identifies a minimal generic protocol and a declarative approach of representing rules and constraints specific to negotiation types, it highlights a set of basic concepts that could be part of a core negotiation ontology and it presents the initial prototype for the English auction which served as a starting point for the present implementation. With respect to the chosen programming language, there were three reasons that made its employment appropriate: (1) Jason supports the declarative programming paradigm, closer to logic programming; (2) it is implemented in Java which makes it multi-platform; (3) a Jason multi-agent system can be easily distributed over a network (for example, by using JADE).

The paper is structured as follows. In Section 2 we outline the context that led to this approach of a GNP. Section 3 demonstrates the specific approach by providing code samples and by offering details on how the initial prototype from [1] was extended. Finally, Section 4 summarises the results and presents the conclusions.

2 Background and Related Work

Most of the papers that deal with the subject of automated negotiations are focused on describing specific protocols for different kinds of interactions. However, in what follows, we will only produce a brief description of those that tackle the problem of a generic protocol and that have influenced the present work.

Article [3] presents an electronic market architecture whose main asset is a declarative auction language (DAL) expressed by means of an XML representation. Rolli *et al.* claim that such a language makes possible the design of auction mechanisms, code generation, deployment of market instances and modelling of participants' behaviour. Albeit its expressiveness, the computational complexity of the model is a real drawback especially when there are introduced more mechanisms. On the other hand, using an interpreter for declarative programming languages like Jason is more efficient. Additionally, its similarity to logic programming simplifies the process of specifying new negotiation mechanisms.

Article [2] opens the way for a universal protocol by proposing a generic software framework for automated negotiation. Bartolini *et al.* argue that the presented interaction protocol is simple enough and it can be used in all circumstances. Moreover, a taxonomy for negotiation rules is presented, which identifies and outlines the different roles of agents within an auction. Although very helpful for this research, the illustrated approach is relatively limited because it only addresses the problem from the perspective of the negotiation host, i.e. the agent

that controls the negotiation process. Nevertheless, the paper remains important because the proposed taxonomy has been incorporated into the declarative framework in order to delimitate between the different stages of an auction.

Article [1] represents the foundation for the present implementation model since it addresses the problem from the perspective of both the auction host and the participant. The authors bring to the forefront of the architectural model three types of agents:

- *Auction Service (AS)* It manages the auction related activities like: auction creation, auction termination and it keeps track of all the current auctions by containing an auction directory;
- *Auction Host (AH)* (see Sec. 3.1);
- *Auction Participant (AP)* (see Sec. 3.2).

Another interesting observation of the authors is that, from a conceptual point of view, the above agents can be described by means of the following equations:

$$AH = GNP_{rolehost} + DNM_{host}$$
$$AP = GNP_{roleparticipant} + DNM_{participant} + CNS$$

where:

- **GNP** is the Generic Negotiation Protocol that governs the interaction between agents;
- **DNM** is the Declarative Negotiation Mechanism (see Sec. 3.3);
- **CNS** represents the Custom Negotiation Strategy and it is specific to a given AP as it can be noticed from the second equation. Nevertheless, the CNS must be consistent with the DNM. Its purpose is to help the agent take the appropriate steps (that remain within the limits of the DNM) in order to achieve its own aims[3].

Since the proposed prototype was rather simple and illustrated only the workflow of an English auction, the general character was not fully achieved. As a result, we have set the goal for this paper the extension of its applicability. In what follows, we will discuss the changes on the original model and the improvements that we propose.

3 Implementation Details

From the three agent types specified in the previous section, only the AH and the AP will be presented at large, since they illustrate better the improvements brought to the model proposed in [1]. Nevertheless, one important observation must be made on the AS: it creates a new auction instance (AIN) based on a short description (which consists of the auction type and the product name) offered by the auction initiator participant (AIP). This can be considered the first situation of auction parametrization because the AIN is informed of the specific type of auction that it will manage.

3.1 Auction Host

The auction host is a unique agent per auction instance and it represents the authority that governs the auction. According to [2], AH plays the following roles: gatekeeper (decides which agents can be submitted to negotiation), proposal validator (verifies if a proposal

[3] For more details on the architecture please consult [1]

satisfies the negotiation template), protocol enforcer (determines the circumstances in which a participant may post a proposal), information updater (changes the parameters of the negotiation as it unfolds), negotiation terminator (specifies when no more proposals may be posted) and agreement maker (chooses from a set of valid proposals the one (those) that should be turned into an agreement(s)).

Due to the fact that Jason agents pursue their goals by applying plans which are compliant with their beliefs and rules (for more details see reference [6]), the definition of an AH's behavior comes easily.

```
+register[source(A)]
    : can_register(A)
    <- +registered(A);
       ?buy_it_out(Sum);
       ?increment(Increment);
       ?items(Left_items);
       ?last_offer(Offer);
       ?state(State);
       .send(A, tell, registered(info(Sum,Increment),
        status(Offer, Left_items, State))).
```

```
+fold[source(A)]
    : registered(A)
    <-  -registered(A);
        .send(A, tell, not_registered).
```

Listing 1 The Auction Host – gatekeeper role implementation.

In order to illustrate its role as a gatekeeper the *register* and *fold* plans were considered (see listing 1). That is, every time the AH receives a request of registration or withdrawal from an AP, it applies one of the implemented plans based on some conditions (can_register(A), registered(A)). The conditions may vary from one auction to the other so they are specified by the DNM. The other roles of the AH have been outlined through a set of plans and goals illustrated in listing 2.

```
+bid(Offer,Items)[source(A)]
    : check_protocol(A)
    & check_proposal(Offer,Items)
    & not(terminate(Offer,Items))
    <- -+state(processing);
       !update_status(A,Offer,Items);
       !inform_participants(A,Offer);
       -+state(bidding).
```

```
+!close
    : check_winner
    <- ?initiator(I);
       -+state(closed);
       ?bidders(A);
       .send(I, tell, winner(A));
       .send(A, tell, winner).
```

Listing 2 The Auction Host – proposal validator, protocol enforcer, information updater and negotiation terminator roles' implementation.

The modifications made with respect to the initial model are the following: the "buy-out" and the fold options were introduced and new goals were created (update_status and inform_participants) in order to outline the different roles of the AH and also the phases of an auction.

3.2 Auction Participant

The AP only registers for an auction, receives a minimal description of it (which was not present in [1]) and, based on that description, it chooses the strategy that bests suits its goals. In this case, there have been implemented three kinds of strategies: firstly, if there is a "buy-out" sum and it is public (the Sum parameter must be a non-zero number), then it offers the sum immediately (aggressive bidder); secondly, if there is no "buy-out" sum and it is an ascending auction (Increment is positive), then strategy one is applied, i.e. it continuously bids and increases the last offer by Increment units until the auction closes or it is declared winner; thirdly, if there is no "buy-out" sum and it is a descending auction

(Increment is negative), then strategy two is applied, i.e. as long as the last accepted bid is greater than its available amount of money it remains idle and when the offer is lowered enough it starts bidding by decreasing its last offer until the auction closes or it is declared winner. Listing 3 displays the selection of the second strategy (which follows the rules of an English auction), the others being similar. We also present the implementation of the third strategy that conforms to the rules of a Dutch auction.

Note that selecting a suitable bidding technique only depends on whether there is a "buy out" sum and whether the increment is positive or negative which naturally leads to a poor strategy. Additionally, the type of AP displayed here is that of an aggressive bidder.

```
+registered(info(Sum,Increment),
status(Offer, Left_items, _))
    : Sum==0 & Increment>0
    <- -+current_quote(Offer);
       +increment(Increment);
       -+items(Left_items);
       +strategy(1);
       !bid_strategy1.
```

```
+!bid_strategy2
: not(accepted) & amount(Amount) &
  items(No) & No>0 & current_quote(Quote) & Quote<=Amount
<-  ?auction_host(AuctionHost);
   -+current_offer(Quote);
   .send(AuctionHost, tell, bid(Quote, 1)).
```

Listing 3 The Auction Participant - strategy determination and implementation.

3.3 Declarative Negotiation Mechanisms for English and Dutch Auction

The DNM depends on the type of negotiation and it is used for customizing the GNP. It is a layer of the architecture which differentiates between auction types. In the present implementation model it consists of a set of rules (and goals) that the AH applies (and pursues) during the course of an auction and it follows the taxonomy proposed in [2]. This classification identifies conditions for: admission of participants (can_register(A)), proposal validity (check_proposal(Offer,Items)), protocol inforcement (check_protocol(A)), updating status and informing participants (update_status(A,Offer,Items), inform_participants(A,Offer)), lifecycle of negotiation (terminate(Offer, Items)), agreement formation (this part of the auction has not been implemented yet but it is considered for future work). The rules and goals that differ the most between the two auction types are illustrated by listing 4:

```
// Dutch auction
check_proposal(Offer,Items)
    [state(bidding)] :-
    asked_price(AskedPrice) &
    items(Left_items) &
    Left_items>=Items &
    Offer <= AskedPrice.

+!update_status(A,Offer,Items)
    <- .time(H,M,Sec2);
       ?items(I);
       ?increment(Increment);
       ?asked_price(AskedPrice);
       ?bidders(B);
        -+last_offer(Offer);
        -+items(I-Items);
        -+bidders([A|B]);
        -+asked_price(AskedPrice+Increment);
        -+last_update(Sec2).
```

```
// English auction
check_proposal(Offer,Items)
[state(bidding)] :-
    increment(Increment) &
    last_offer(Quote) &
    items(Left_items) &
    Left_items>=Items &
    Offer >= Quote + Increment.

+!update_status(A,Offer,Items)
    <- -+last_offer(Offer);
       -+bidders([A]).
```

Listing 4 Dutch versus English auctions DNM.

4 Conclusions

In this paper we have presented a new version of the prototype of a GNP proposed in [1]. The improvements relate mainly to AH and AP implementation. On the one hand, the AH displays a better separation of roles which also leads to a better organization of an auction's workflow. On the other hand, the AP may choose from different strategies the one that complies with the current rules based on a description of the auction and not its name. Thus, there is no need to create one agent for each kind of auction.

We believe that this approach to a declarative specification of a GNP may bring satisfactory results in the pursuit of creating a generic framework for automated negotiations. However, the current results suggest that the more comprehensive the protocol becomes, the harder it is to determine a variety of negotiation strategies. A compromise may consist in simple strategies that fail to use all the choices that a type of auction offers. We think that this is an interesting direction that should be pursued in future research.

References

1 Alex Muscar, Costin Badica. *Exploring the design space of a declarative framework for automated negotiation: initial considerations (**in press**)*. AIAI'2012 Proceedings, 2012

2 Claudio Bartolini, Chris Preist, Nicholas R. Jennings. *A generic framework for automated negotiation*. In: R. Choren, A.F. Garcia, C.J.P. de Lucena, A.B. Romanovsky (eds.) SELMAS, *Lecture notes in computer science*, vol. 3390, pp. 213-235, Springer, 2004

3 Daniel Rolli, Stefan Luckner, Henner Gimpel, Christof Weinhardt. *A descriptive auction language*. In: *Electronic Markets*, vol. 16, issue 1, pp. 51-62, Routledge, 2006

4 Peter R. Wurman, Michael P. Wellman, William E.Walsh. *A parametrization of the auction design space*. Games and Economic Behavior 35(1-2), 304-338, 2001

5 Adriana Dobriceanu, Laurentiu Biscu, Amelia Badica, Costin Badica. *The design and implementation of an agent-based auction service*. IJAOSE 3(2/3), 116-134, 2009

6 Rafael H. Bordini, Jomi F. Hubner, Michael Wooldridge. *Programming multi-agent systems in AgentSpeak using Jason*. Wiley, 2007