INTELLIGENT AGENTS AND MULTI-AGENT SYSTEMS

AN AGENT–ORIENTED ELECTRONIC MARKETPLACE FOR MODELING AND SIMULATION OF DYNAMIC PRICING MODELS BUSINESS LOGIC

Jacek Jakieła, Paweł Litwin, Marcin Olech

***Abstract**: The main goal of the research that preliminary results have been presented in this paper is to develop an agent–oriented electronic marketplace for modeling and simulation of dynamic pricing models, i.e. models in which the price of the item is allowed to fluctuate as supply and demand in a market change. The work provides an overview of forms of dynamic pricing models, with particular emphasis on auctions. After that, the main rationale behind using Multi-agent Systems approach for modeling and simulation of complex business structures has been shown. Then the development process of an electronic marketplace, including agents' architecture as well as implementation environment selection, structure and business logic of e-marketplace have been presented. Last part of the paper comprises conclusions and further research plans.*

***Keywords**: Agent-Based Models, Simulation for MAS development, Agent-oriented Marketplace Design, Multi Agent Based Modeling and Simulation, Dynamic Pricing Models*

***ACM Classification Keywords**: I. Computing Methodologies; I.2 Artificial Intelligence; I.2.11 Distributed Artificial Intelligence; Multi-Agent Systems*

Introduction

The opportunities of using *Multi-Agent Systems* (MAS) and *simulation* are numerous. They have already been applied and coupled in many application domains and for different purposes [Weyns et al., 2009].

The paper describes the first steps undertaken in the new area that is being investigated by authors, but which is the continuation of work that results have already been published

in [Jakieła et al., 2009][Jakieła, Litwin, Olech, 2010a][Jakieła, Litwin, Olech, 2010b][Jakieła, Litwin, Olech, 2011][Jakieła, Litwin, Olech, 2012]. The research conducted so far was focused on application of *Multi-Agent Based Simulation* (MABS) to Supply Chain Management as well as the analysis of Supply Chain behavior (e.g. bullwhip effect analysis).

The goal of the newly started research is to develop an agent-oriented electronic marketplace that will be used for analysis of dynamic pricing models, the models in which price of the item is not fixed but may be changed before transaction is finalized. Activities related to price determination are part of many business processes conducted by firms which are a part of supply chains. For example at the buy-side firm is executing transactions with suppliers, while at the sell-side it interacts with customers. All these activities are increasingly subject to automation as a part of electronic marketplaces.

Automating markets leads to many benefits. One is cost saving from automating functions of non-computational markets such as searching for goods and potential trading partners or price discovery automation. Another benefit would be the ability to extend markets in time and geographic scope by conducting them over networks [MacKie-Mason et al., 2006]. The greatest potential however may be related to the opportunity to deploy market mechanisms that cannot effectively operate without computer automation e.g. dynamic pricing.

When automating market related activities one can take into consideration different alternatives for business logic that will finally be implemented. For example, when the price discovery mechanisms are supposed to be implemented in the form of dynamic pricing model, then *which model is the best choice?* The solution to this problem would be to prepare the experiments which results will be used in the process of business logic selection. Therefore the goal set for the research is to use multi-agent based simulation as a test-bed for this kind of analysis.

The paper is structured as follows. Firstly the advantages of dynamic pricing models applications have been described. After that, the rationale behind using agent paradigm has been explained. Then the problem of e-marketplace design has been characterized. Finally the development process of agent oriented electronic marketplace prototype has been presented in detail.

Virtual Dynamic Pricing as an Efficient Way of Price Discovery

Revenue generation is the core ingredient of every business model used by the contemporary firms. It consists of two main elements: *market mechanisms for price determination* and *revenue sources*. In every firm operating as a part of the supply chain

there are many processes where transaction is executed. It may be the buy-side of the organization where procurement and inbound logistics processes take place. On the sell-side every firm is dealing with intermediaries and customers selling and distributing to them products and services. Wherever transaction is executed, the price for the transaction has to be accordingly determined. There are three commonly used mechanisms of price determination: *menu pricing*, *dynamic pricing* and *bartering*.

In the classical Old Economy context, the basic mechanism is *menu pricing* also known as *fixed pricing*. It works according to the logic, where seller sets the price and the buyer may take it or leave it. This model is used by nearly every retail store where prices for products are fixed and cannot be changed. Although menu pricing has been working for many years it has two main shortcomings. First one is related to the situation in which buyer may be willing to pay more than the price set by the seller. It could be said that seller is leaving money on the table. The second issue is that the menu price is too high and cuts off many buyers. They would have bought the product at a lower price. The stickiness of the prices is the result of two main factors. Firstly, it is not easy to detect changes in consumer preferences quickly enough to effect price changes. Secondly, in an off-line context it is quite hard to implement price changes [Afuah et al., 2002].

These problems may be solved by dynamic pricing which business logic is encapsulated in the functionality of e-marketplace used on-line by buyers and sellers. The first issue is that the price may be changed according to specific protocol. What is more, with the Internet the customer preferences can be detected more easily and cost of changing prices is lower because they all are virtual.

In dynamic pricing model the price is not fixed but may change over time what overcomes the disadvantage mentioned before which lets some customers get away with the price that is less than they would be willing to pay and misses out on customers who would prefer to pay less.

Dynamic pricing may appear in several forms. There are many classifications. In the most popular one, the form of dynamic pricing depends on number of buyers and sellers involved in the process. When there are one buyer and one seller, the pricing is based on negotiation, bargaining or bartering which are the oldest forms that have been practiced for many generations in markets, usually open-air. The final price will be determined by bargaining power of each party, business factors and supply/demand in the item's market [Turban et al., 2011].

In case there is one seller and many potential buyers or one buyer and many potential sellers the pricing model is called an *auction*. If there is one seller and many buyers the auction takes a form of *forward auction*, in which seller entertains bids from multiple

buyers. In the *reverse auction* there is usually one buyer and many sellers. Buyer places an item she wants to buy for bid and potential supplier bid on the item, reducing the price sequentially. Reverse auctions are primarily a Business-to-Business (B2B) pricing mechanism.

The auction is driven by *auction protocol* which determines when transaction is finalized, who will get an item and how much the winner will pay for it.

In *English auction* seller begins by calling out a starting (most often low) price which is gradually raised, apparently to all buyers, usually in small increments. The auction stops after predefined period of time ("deadline" auctions) or when there is only a single bidder who is still interested. Finally the item is sold to the highest bidder.

The *Dutch auction* is open descending price counterpart of English auction. Seller begins with initial price and then gradually decreases it. Starting price is high enough so nobody is interested in buying the item at that price. The price is lowered until some bidder indicates her interest and the item is sold to her at the given price.

In the *sealed-bid first-price auction* bidders submit bids in sealed envelopes. The highest bid wins and the winner gets the item and pays what she bid. The *sealed-bid second price auction* is the process in which bidders submit bids in sealed envelopes and the highest bidder wins but this time she pays second–bid [Milgrom, 2004].

According to the research goals, all of the mentioned auction protocols are supposed to be implemented as business logic of e-marketplace that will finally be tested with the use of agent–oriented simulation. Next section describes the main rationale behind using agent paradigm for this purpose.

Agent-oriented Simulation as a Test-Bed for Dynamic Pricing Business Logic of e-Marketplace

The MABS approach to modeling complex business structures has been becoming more and more popular; its models as well as their advantages have been widely described in [North et al., 2007][Weyns et al., 2009]. Contributions to the MABS domain are periodically published among others in Springer's LNCS and LNAI series [Jakieła, Litwin, Olech, 2010b][Jakieła, Litwin, Olech, 2012].

Multiagent-based simulation (MABS) can be defined as the modeling and simulating real world system or phenomena where the model consists of agents cooperating with one another in carrying out tasks and achieving collective goals. The advent of multi-agent based modeling has introduced an important innovation: behavior of complex systems with many active entities can be simulated by modeling individual entities and their interactions. Importantly, the operation of the system doesn't need to be defined a priori

as set of equations, terms or logical statements, but the whole behavior emerges from individual objects behaviors, their interactions and impact of the environment.

As this paper focuses on agent paradigm application to price discovery on e-marketplace, it is important to look at what have been done so far in this domain. Of course it is possible to design markets without agents. As Marks suggests, in such case one has to have market with demand and supply schedules and what is more, economic efficiency is maximized at the output level where marginal value equals the marginal unit cost, no matter how the social surplus is divided between buyers and sellers [Marks, 2006]. There is only one drawback, as this is optimization the problem has to be well defined. Because of several possible design trade-offs and emergence of unforeseen performance in the system, modeling market system as evolving system of autonomous, interacting agents is increasingly employed. This thesis may be backed by opinion provided by LeBaron who states that agent-based models are well suited to examining market design because they can produce large amount of data and allow testing of market design in a heterogeneous, adaptive environment [LeBaron, 2006]. As many applications have already proved, using agents in market design is quite reasonable choice. The agent paradigm have already been used in analysis of market's micro–structure [Audet et al., 2002], examination of tick sizes [Bottazzi et al., 2003], analysis of double auction process where buyers and sellers, equipped with heuristic rules (belief-based learning) are trying to assess the probabilities that they offers to buy/sell will be accepted, given market history [Gjerstad, 2004]. Agent applications may also incorporate genetic algorithms that are used to encode market decisions that agents can make and to find the optimal one or to encode beliefs about changes of prices from period to period in order to find optimal consumption/savings allocations and market–clearing prices [Arifovic, 1994][Duffy, 2006] [Bullard et al., 1999].

All mentioned pros backed the main goal of the research work which is *to use agent-oriented electronic marketplace as a test–bed for analysis of dynamic pricing models in order to determine which business logic should be encapsulated in the multi-agent system supporting price discovery in procurement and in-bound logistics processes in supply chain*.

The sections below present the development process with all design decisions that had to be made such as: selecting agents' architecture and implementation environment, defining market structure, and finally designing individual agents. In the description that follows only English auction model has been taken into account, but the development process for other models follows the same structure and logic.

The Development Process of Electronic Marketplace in Detail

Multi-Agent Systems are today considered as an interesting way of understanding, modeling, designing and implementing different kind of distributed systems. From the perspective of planned research, the application of MAS has two aspects. Firstly, as Parunak claimed, MAS represent modeling alternative, compared to equation based modeling, for representing and simulating real world or virtual systems which may be decomposed in interacting individuals [Parunak et al., 1998]. At the same time MAS could be used as a programming paradigm to develop software systems. Agent paradigm is especially suited for solutions in which global control is hard or not possible to achieve [Zambonelli et al., 2002]. The modeling aspect requires good understanding of architecture of modeling constructs – agents as well as the whole system – MAS, and the implementation needs–proper environment in which it will be conducted. These two issues will be discussed in the following sections.

Selecting an Agents' Architecture

There are two main approaches for analyzing agent's architecture: *reactive* and *cognitive*. In reactive approach only the perception–action also known as stimuli–response component is considered. The cognitivist approach relies on the mental issues and assumes that an agent has mental states as well as the partial representation of the environment and other agents. There is also third approach called hybrid, trying to get together first two.

The best-known cognitive architecture, which has been employed in the process of e-marketplace design and development, is the BDI (*Belief-Desire-Intention*) architecture. The main quality of BDI architectures is to create a behavior which mimics that of a rational human being [Rao et al., 1992].

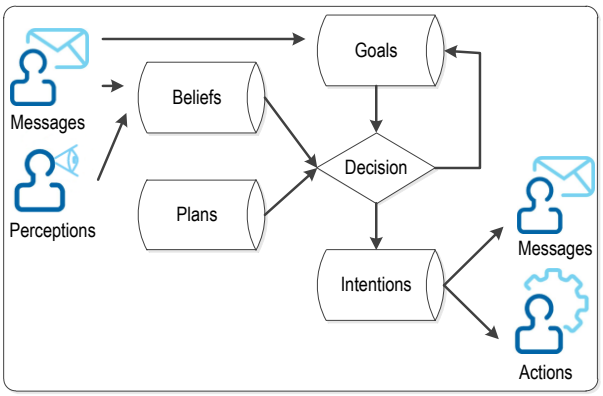

Fig. 1. The elements of BDI architecture

In this architecture there is an assumption that agents infer and act on the basis of knowledge that is represented with a symbolic formalism. What is more, agent possesses mental states such as *beliefs*, *goals* (desires), and *intentions*. Agent acts intentionally in order to achieve goals and this behavior is based on beliefs about world states. Every agent is equipped with library of plans. The plan works as a recipe for a goal achievement and contains a set of actions. When the agent commits itself to achieve the specific goal with the use of specific plan then the intention is set. All the mental states are dynamic what means that are updated during agent's life cycle according to environment's changes and messages from society. The figure 1 illustrates all of the architecture basic building blocks and relationships among them [Ferber et al. ,2009].

As the figure 1 shows beliefs are formed based on perceptions concerning environment changes and messages received from other agents. Each agent may initially have one or more goals. In order to achieve goal, decision component is trying to select plans that are compatible with agent beliefs and executes actions included in plans. Plan that has been chosen for execution is transformed into intentions that finally result in environmental as well as communication actions. The whole process is controlled by BDI engine which works according to logic embedded in implementation environment.

Selecting the Implementation Environment

In recent years, there is extremely rapid increase in amount of research being done on agent-oriented development platforms and programming languages. As a consequence of agent's architecture selection, BDI based implementation environment has been chosen. Presented environment consists of language which is a variant of AgentSpeak programming language and its interpreter called *Jason*. In the following paragraphs, the basic language constructs have been described as well as the structure of an interpreter.

AgentSpeak has its roots in logic programming. Therefore basic representation units, which are used to denote mental attitudes of agents, are predicates. The main language constructs, as in case of BDI architecture, are related to mental states and denote beliefs, goals and plans. *Beliefs* are used to represent information agent stores about environment, other agents and itself. Interesting fact about beliefs in Jason is that they are annotated and therefore may be maintained on the meta-level. There are three main annotations such as: *percept*, *self* and *agent name*. Percept is used to denote information from the agent sensors (received from environment). Self means that the belief is created by agent as a *mental note*. Agent name suggests that source of the belief is other agent.

Goals represent the state of affairs the agent strives for. The representation of goal is the same of a belief expect that it is prefixed by symbol "!". *Plans* constitute courses of

action an agent will execute in order to achieve its goals or to react to changes in its environment. Every agent has the library of plans that determines its behavior.

The plan is structured as presented below.

```
triggering_event : context <- body.
```

The *triggering_event* represents event that will be handled by plan. *Context* describes circumstances under which the plan is suitable to handle the event. The *body* is a sequence of actions that will be executed or new goals for the agent to achieve. The agent behavior may change over time if new plans are acquired during the communication with other agents.

Agent acts according to agent's program which specifies initial beliefs, initial goals of the agent, and the plans in the plans library available to agent when it starts running.

Agent program is executed by Jason interpreter which uses a number of important data structures that allow to implement BDI agents. Figure 2. shows the main components of the Jason interpreter [Bordini et al., 2009].

Belief base is responsible for gathering and storing all the beliefs agent possesses and is updated in every reasoning cycle on the basis of all percepts received by from environment as well as other agents.

Events result from changes in beliefs and goals. Beliefs may be updated and new goals set or received from other agents as a part of the delegation process. Events trigger execution of plans, provided that event matches the *triggering_event* and is applicable the time is chosen.

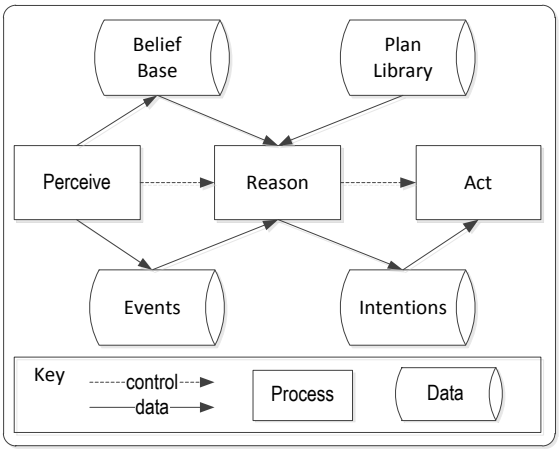

Fig. 2. Simplified view of Jason interpreter components [Weyns et al., 2009]

Plans library stores agent's know-how and includes all the plans written for an agent in AgentSpeak code. In case of simple agents the plan library remains unchanged; however it is possible to change agent behavior by plan exchange using speech-act based communication. Plans may be labeled and annotated what enables meta-level advanced processing in a selection function.

Set of intentions that are created every time the change in environment is perceived by an agent and there is applicable plan for an event. Each intention is a stack of partially instantiated plans and represents a "focus of attention" for the various tasks currently being carried out by an agent. Intention can be dropped or revised.

The body of the plan is composed of actions. Unlike actions, *internal actions* don't change the state of the environment. They are mechanism to allow legacy code to be referenced from the high–level agent reasoning as defined by AgentSpeak code. There are several predefined internal actions provided by Jason to help with various programming tasks. All internal actions start with the "." character [Weyns et al., 2009].

The structures presented above are all essential for BDI agents. Of course there are various other structures used by Jason interpreter but due to the space restrictions of the paper it is not possible to go into enough detail. Complete account of the Jason implementation is available in [Bordini et al., 2007]

The Design of Electronic Marketplaces

Markets play central role in the economy, facilitating the exchange of information, goods, services and payments. Three main functions of every market are: matching buyers and sellers, facilitating the exchange of information, goods, services and payments associated with market transactions, and providing institutional infrastructure (legal and regulatory framework) which enables the efficient functioning of the market [Bakos, 1991].

The emergence of electronic marketplaces introduced several changes in processes used in supply chains, which resulted in [Turban et al., 2011]:

- greater information richness of the transactional environment,
- lower information search costs for buyers,
- diminished information asymmetry between sellers and buyers,
- the ability of buyers and sellers to be in different locations,
- better, more efficient mechanisms for price discovery.

From the perspective of this paper especially important is the last issue from the list above. The goal is to design an electronic marketplace used for analysis of dynamic pricing models.

Designing markets is rather a new discipline. Marks lists the following five examples of designed market [Marks, 2006]:

- *Simulated stock markets* for new financial instruments and derivatives. They have been developed after Black and Scholes solved the problem of pricing options.
- *Markets for pollution emissions* where market mechanisms are used to control emission of sulfur dioxide and carbon dioxide.
- *Markets for electro–magnetic spectrum* where bands of local spectrum to be used for new communication technologies are sold.
- *Markets for electricity exchange* where several types of new market mechanisms have been introduced because classical ones were not appropriate.
- *On-line markets* also known as *e-marketplaces* that provide opportunities to buy and sell on-line with the use of Internet infrastructure. There are several types of them. Major Business-to-Consumer (B2C) e-marketplaces are *virtual storefronts* and *e-malls*. Business-to-Business (B2B) e-marketplaces include *sell-side* and *buy-side* e-marketplaces and *exchanges*.

The business logic that is supposed to be tested on implemented agent–oriented e-marketplace may be used in the design of sell-side, buy-side e-marketplaces as well as B2B exchanges.

The Structure and Business Logic of e-Marketplace

According to MacKie-Mason there are mainly three stages, parties must go through in order to execute a transaction [MacKie-Mason et al., 2006]: (1) *connecting* that is responsible for discovery of opportunity to engage in a market interaction; (2) *interaction* which is the negotiation of terms; (3) *exchanging* – execution of terms of the finalized transaction. The paper focuses mainly on the first two stages. Connecting is realized with the use of communication protocols the agents use. Negotiation of terms, as will be shown later, is done according to selected dynamic pricing models protocols.

In order to organize presentation of the development process, the framework of marketplace system has been used, that consists of the following elements:

- *Marketplace system* – the society of agents that participate in resource allocation problem, together with market mechanisms used during interaction in order to finalize a transaction.
- *Mechanisms* – the rules specifying permissible actions and the outcomes as a function of agent actions.
- *Market participants* – agents with BDI architecture that represent autonomous decision making locus in the system of many decision making entities.

Depending on the dynamic pricing model analyzed, agents operate according to some decision rules (market mechanisms).

The general structure of the marketplace system is quite simple. It consists of buyer and seller sides. Seller–side is composed of seller agents and buyer–side includes buyer agents. The system has been parameterized what enables to easily change the number of agents.

Because the main aim of the marketplace is to analyze the business logic related to dynamic pricing models, these mechanisms are based on auctions protocols. Protocols selected are those, the most often used nowadays. There are English Auction, Dutch Auction, First-Price Sealed–Bid Auction (FPSB) and Second–Price Sealed–Bid Auction (SPSB). All of them have shortly been described in the section entitled *Virtual Dynamic Pricing as an Efficient Way of Price Discovery*.

Every protocol is characterized by the set of parameters that have been presented in the table 1. During simulation the values of parameters are supposed to be randomly generated. Specific distribution may be selected by modeler. In the default settings *uniform distribution* is used.

Table 1. Parameters of auction protocols used

Protocol	Reservation price	Initial product price	Reaction time	Duration of the auction	Decreasing price's time	Decreasing price's value
English	*Yes*	*Yes*	*Yes*	*Yes*	*No*	*No*
Dutch	*Yes*	*Yes*	*Yes*	*No*	*Yes*	*Yes*
FPSB	*Yes*	*No*	*Yes*	*Yes*	*No*	*No*
SPSB	*Yes*	*No*	*Yes*	*Yes*	*No*	*No*

The sell–side of the marketplace works according to the following rules:

- Seller agent is supposed to sell products on the marketplace.
- Seller agent has *reservation price* established, that is a minimal acceptable value of transaction, and the *reaction time* that determines when the auction winner will be announced and when the product will be offered on the marketplace.
- Seller agent also knows the *duration of the auction* in case of Sealed-bid auctions as well as English auctions – after specified period of time goes by, the system closes an auction.
- Seller agent sets the *initial product price* for English and Dutch auctions. This is the value the potential buyers start to bid with.

- Seller agent sets two additional parameters for Dutch auction which are related to the velocity of price decrease and the value which is used to decrease the price.

The buy–side of the marketplace operates with regard to the following assumptions:

- Buyer agent is supposed to buy a product offered on the marketplace.
- Buyer agent has *private valuation of product* established which is the maximum price she is willing to pay, and the *reaction time*.
- Buyer agent has two additional parameters for English auction. The first one concerns the value which is used to increase the product's price. Second one regards the time after which the buyer will re-bid.
- Buyer agent bidding process consists of the following steps:
    - Checking all open auctions.
    - Evaluating all open auctions utility function values.
    - Removing all auctions that do not meet established criteria (the utility function value is lower than the threshold that has been set).
    - Selection of the auction with highest utility function value.
    - Bidding until the auction is open.

The business logic presented above has been implemented in the e-marketplace. As the seller side is quite simple, next sections describe mainly implementation details of buyer agents.

Buyer Agents Implementation

Buy–side of the e–marketplace consists of two types of agents. The first type is *dummy agent* that operates randomly, according to parameters' distributions selected by modeler. The second category is *smart agent* which is able to use different tactics during the bidding process. All agents' behavior is driven by BDI interpreter according to agents plans implemented. Simple business logic, as in case of dummy agents has been developed with the use of *AgentSpeak*. More complex behaviors that required sophisticated calculations have been implemented in Java language. Thank to flexible architecture of *Jason* interpreter as well as Java interfaces it has, these two implementations can be relatively easily put together and fully integrated.

Dummy Agent Implementation

The operation of dummy agent consists of several activities carried out as internal and external actions. The first action is responsible for setting the parameters presented in the table 2.

Table 2. Dummy agent parameters

Parameter's name	Value Range	Distribution
Reaction time	(5; 8)	Uniform
Product's max price	(100;250)	Uniform
Ace value	⟨1;20)	Uniform
Ace interval	⟨1;5)	Uniform

After parameters' values are generated the beliefs are created and saved in the agent's beliefs base.

In the next step external action causes that list of all auctions recorded in the system is saved and made available to the agent. Auctions have been divided into two groups: open and closed. In addition all open auctions are counted and the result of this operation is stored in agent's beliefs base. The figure 3 presents example beliefs base of an agent after the operation is executed.

engAuction(0.6578,1,sellerEng2)

engAuction(closed,5,sellerEng4)
engAuction(closed,2,sellerEng3)
engAuction(0.2413,3,sellerEng1)
engAuction(0.6772,4,sellerEng5)

Fig. 3. The agent's beliefs base content

The first *engAuction* predicate argument is the value of utility function for auctions that are open or the information that auction has already been closed. The second is an *auction ID* and the third is ID of a seller.

Auctions with utility function value smaller than predefined threshold should be removed. Therefore internal action has been defined for deletion of such auctions from the agent's belief base. When all such auctions have been removed, the number of open auctions agent can bid for has been calculated. Next, the check is done if there are any auctions we can choose from. Finally the internal action finds the maximum element in the list which represents the best option.

The last activity the agent undertakes is to bid for selected auction. Bidding action has been implemented in the environment which is responsible for managing offers. The action ends after the agent receives *noAuction* belief from the system.

The bidding process is conducted according to the following steps:

1. Is the auction open? If it is, then go to step 2; if it is not, go to step 7.

2. Does the last offer is my offer? If it does, go to step 8; if it does not, go to step 3.
3. Calculate value of the bid.
4. Does the calculated value is greater than my private product's estimation value? If it does, go to step 9; if it does not, go to step 5.
5. Place an offer, store transaction information in the system, print information to the log file.
6. Return to the internal action (run the bidding process once again).
7. Print message: "No auction found in system or auction has been closed", add percept *noAuction* – bidding process has been terminated.
8. Print message: "I submitted the last offer, no action is required this time". Return to the internal action (run the bidding process once again).
9. Print message: "Product's price exceeds the private product's estimation value", add percept *noAuction* – bidding process is terminated.

Smart Agent Implementation

The agent which is presented in this section uses English auction protocol. The main difference between *smart agent* and *dummy agent* is that the former is able to determine price dynamically according to carefully selected tactics. All tactics have been based on decision functions proposed in [Faratin et al., 1998].

The smart agent's business logic includes the following tactics for dynamic price determination:

a) *remaining time* tactic:

$$f_{rt}(t) = \alpha_{rt(t)} \cdot p_r \tag{1}$$

where:

f_{rt} – calculated product price according to remaining time tactic,

t – current time,

p_r – private value of the product (reservation price),

$\alpha_{rt}(t)$ – remaining time tactic coefficient, which is calculated according to formula presented below.

$$\alpha_{rt(t)} = k_{rt} + (1 - k_{rt}) \cdot \left(\frac{t}{t_{max}}\right)^{\frac{1}{\beta_{rt}}} \tag{2}$$

where:

k_{rt} – proportion between initial (p_i) and reservation price (p_r); the initial price is value the agent wants to start bidding process with,

t_{max} – time when the last active action in the English protocol will be terminated,

β_{rt} – adjust shape of the price's curve, it belongs to positive real numbers.

b) *remaining auction* tactic:

$$f_{ra}(t) = \alpha_{ra}(t) \cdot p_r \tag{3}$$

where:

f_{ra} – calculated product price according to remaining auction tactic,

t – current time,

p_r – private value of the product (reservation price),

$\alpha_{ra}(t)$ – remaining auction tactic coefficient, which is calculated according to the formula presented below

$$\alpha_{rt(t)} = k_{ra} + (1 - k_{ra}) \cdot e^{\frac{|L(t)|}{\beta_{ra}}} \tag{4}$$

where:

k_{ra} – proportion between initial (p_i) and reservation price (p_r),

L(t) – number of the open auctions at this time,

β_{rt} – adjust shape of the price's curve, it belongs to positive real numbers.

c) *desire of product purchase* tactic:

$$f_{ba}(t) = \alpha_{ba}(t) \cdot (p_r - \omega(t)) \tag{5}$$

where:

f_{ba} – calculated product price according to desire of purchase product tactic,

t – current time,

$\alpha_{ba}(t)$ – desire of purchase product tactic coefficient,

p_r – private value of the product (reservation price),

ω(t) – minimum purchase price.

$$\alpha_{ba(t)} = k_{ba} + (1 - k_{ba}) \cdot \left(\frac{t}{t_{max}}\right)^{\frac{1}{\beta_{ba}}} \tag{6}$$

where:

k_{ba} – proportion between initial (p_i) and reservation price (p_r),

t_{max} – time when the last active action in the English protocol will be terminated,

β_{ba} – adjust shape of the price's curve, it belongs to positive real numbers.

$$\omega(t) = \frac{1}{|L(t)|} \cdot \left(\sum_{1 \le i \le |L(t)|} \frac{t - \sigma_i}{\eta_i - \sigma_i} \cdot v_i(t)\right) \tag{7}$$

where:

L(t) – number of the open auctions at this time,

σ_i – time when the i-auction started,

η_i – time when the i-auction will end,

$v_i(t)$ – current price of the i-auction.

Algorithm presenting smart agent operation has been shown in the figure 4.

Before agent is initialized, it is possible to modify initial beliefs and rules using AgentSpeak Language. Initial parameters are related to:

- *agent's reaction time* – this value can be deterministic or stochastic. In the prototype, randomly generated values with uniform distribution (5;8) have been set up,
- *parameters for dynamic price calculation tactics* – parameters values can be constant or randomly generated. In the system constant values have been selected,
- *coefficients used in dynamic price calculation tactics* – values that have been set are partially constant and partially random.

After all parameters have been set up, agent starts execution. During initialization stage all coefficients, which are defined in the initial beliefs and rules section are generated and stored in the beliefs base of the agent as constant values. Then agent starts bidding process which is composed of many steps coded with the use of AgentSpeak and Java languages. Bidding is the main plan of the agent. It is executed in the loop until stop conditions are met. Instead of describing every line in detail, the overall idea of agent operation with regard to dynamic pricing will be presented.

The main steps are the following:

- if the agent is going to bid, it waits for a while,
- all prices calculated in the previous iteration are cleared – this is an external action executed by environment,
- all prices are calculated according to tactics – agent's plans for calculation of every tactic's price are almost the same, they take some coefficients as parameters, but all prices are calculated finally in the environment part of the system,
- final bid's value is determined – calculated prices and weights for every tactic are sent to the outside environment to calculate final bid's value,
- auction (for which bid will be submitted) is selected – this is done according to expression that takes under consideration such issues as time remaining to auction's end (auctions which are closer to the end are preferred) and randomly generated value with uniform distribution, which is agent's personal valuation of the product,
- selected auction's ID is saved in agent's beliefs base,
- offer is made with a given auction number and calculated bid value,

- whole process is running in the loop until stop conditions are met.

The stop conditions are stored in the context part of the agent's bidding plan. It is possible that the plan execution will be interrupted with regard to the following reasons:

- bidder has bought the product – bidder successfully executed plan and is not allowed to bid in another auction during current simulation experiment,
- bidder cannot identify any English auction in the system - it means that there is no seller who offers product and therefore it is impossible to bid in the current simulation experiment,
- bidder has not found any auction that is open – if all auctions are closed buyer is not able to submit a bid,
- bidder has already bid – it is prohibited to bid again until the offer is the highest bid.

The business logic that has been described so far has fully been implemented with the use of AgentSpeak language. However there are areas of the prototype developed, where logic programming paradigm needs imperative paradigm support. In these areas Java language has been used for implementation purposes. As a part of smart agent behavior, six external actions have been implemented. They are the following:

- clearing prices calculated during last iteration of the algorithm. This action is quite simple because calculated prices have been stored as beliefs with annotation «percept» denoting the source. Removing them from the beliefs base is done with *clearPercepts* action,
- price calculation according to pricing tactics with given parameters. Calculated price is stored in the beliefs base of the agent,
- bid value calculation according to weights related to every tactic and calculated prices. Final value is stored once again in beliefs base of the agent,
- evaluation and selection of an auction to bid for – evaluation process is performed for every auction which is open. If there is no open auctions, the percept *no(auction)* is returned, which interrupts the bidding process.
  To evaluate every auction value, the following expression is used:

$$f_{val} = rand_{val} \cdot \left(\frac{t-\sigma_i}{\eta_i-\sigma_i}\right) \quad (8)$$

  where:
  $rand_{val}$ – value randomly generated with uniform distribution in a range <0,1)

  other symbols' meaning is the same like in ω(t) expression in formula (7).
  The expression (8) simulates agent's personal valuation of the product, and promotes auctions which are close to the termination. This expression can be easily modified according to the specific bidding strategy planned.

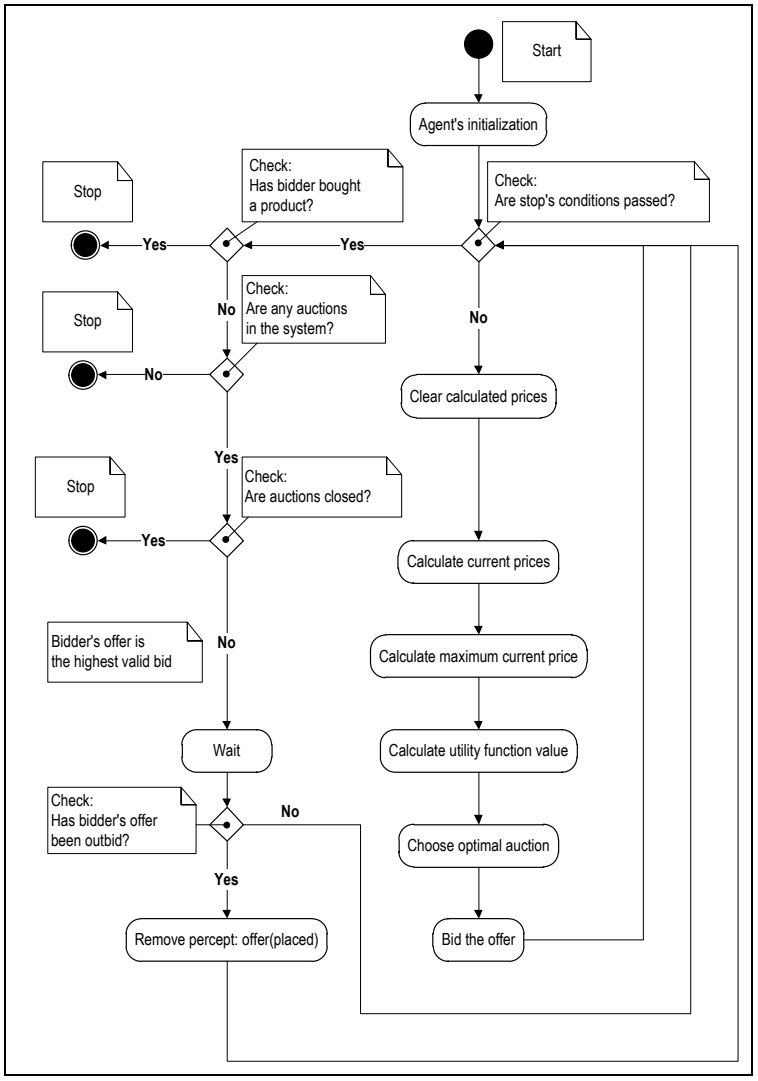

Figure 4. The activity diagram presenting business logic of Smart Agent

- making a bid – as a result of the previous internal and external actions, agent's beliefs base contains two important beliefs: ID of the auction with the highest utility function value and agent offer's value. Both beliefs are bound to the logical variables, which are sent as parameters of the *makeBid* external action. The algorithm of the action is as follows:
    - find auction with given ID,
    - check if the auction is open – if it is not, print message to the console window and terminate action; if it is, go to the next steps,

- check if the last offer does not belongs to the agent – if it does, terminate action and store beliefs *bid(offer)* in agent's beliefs base; if it does not continue operation,
- check if agent's bid value is smaller than or equal to the actual product price – if it is, terminate action and print message to the console window, if it is not, continue operation,
- make a bid,
- save information related to the bidding process in the system.

All auctions and submitted offers are stored in the *ArrayList* data structure, which is created for every auction protocol type. Most important information concerning the bidding process is stored in a text file with semicolon as a delimiter, so data can be further analyzed.

- monitor the auction the agent has bid for - if the agent has successfully bid, making new bid is prohibited until offer has not been outbid. Therefore the action related to monitoring of bidding process of the agent has been implemented.
  If the auction is closed, agent removes belief *bid(offer)* from beliefs base and check if the offer has not been outbid. If it has not, the belief *bought(product)* is added to beliefs base. If the auction is open agent simply checks if the offer is the highest one, and if it is, action ends with no changes made to beliefs base; if it is not, *bid(offer)* belief is removed from beliefs base. Depending on the changes in the beliefs base the agent's activity can be finished, *!bid(auction)* plan can be initialized or monitoring process can be executed once again.

Presented development process concerns the e-marketplace prototype. So far, the main elements of the architecture have been implemented. Of course there is a need for further development but basic business logic has already been implemented. What is more e-marketplace architecture is so flexible that may be easily extended whenever new research objectives will be formulated.

Conclusions and Further Research

The paper presented the first steps of research undertaken in the intersection of MAS and simulation. MABS allows analysis of systems that are too complex to analyze using closed–form techniques. The advantage of using simulation is that it provides us with light where the analytical techniques cast little or none, in our metaphorical search, so we are no longer restricted to working with models which we hope will prove tractable to our analytical tools [Marks, 2006]. After there is an understanding reached thank to simulation, with how the elements of phenomenon of concern work together, it is possible to ask the question of how better to design it.

The final goal the planned research is supposed to achieve is to find the optimal variant of dynamic pricing models business logic that can be implemented in an agent–oriented system supporting management of contracts with suppliers. The analysis process is conducted with the use of MABS, where electronic market plays a role of a workbench for testing different variants of business logic that drive bidding tactics of agents.

Making conclusions from technical perspective leads to the statement, that carefully selected agent's architecture as well as implementation environment seem to be sensible choice. The modeling and implementation process done with BDI architecture and Jason & AgentSpeak tandem have not been so easy. However, having two programming paradigms, object–oriented and logic, in one place gives many advantages. It is possible to easily code the business logic and at the same time make computations when needed.

Further research works will be concerned with equipping agents with intelligent mechanisms such as genetic algorithms in order to improve the performance of agents' bidding tactics and simulation based comparative analysis of these mechanisms.

Bibliography

[Afuah et al., 2002] Afuah, A., Tucci, C., L.: Internet Business Models and Strategies. Text and Cases. McGraw-Hill, (2002).

[Arifovic, 1994] Arifovic, J.: Genetic algorithm learning and the cobweb model. Journal of Economic Dynamics and Control 18, 3–28, (1994).

[Audet et al., 2002] Audet, N., Gravelle, T., Yang, J.: Alternative trading systems: does one shoe fit all?, working paper 2002-33 (Bank of Canada, Ottawa), (2002).

[Bakos, 1991] Bakos, J. J.: A Strategic Analysis of Electronic Marketplaces. MIS Quarterly 15, no. 3 (1991).

[Bellifemine et al., 2005] Bellifemine, F., Bergenti, F., Caire, G., Poggi, A.: JADE — a java agent development framework. In Bordini, R. H., Dastani, M., Dix, J., El Fallah Seghrouchni, A., (eds): Multi-Agent Programming: Languages, Platforms and Applications. No 15 in Multiagent Systems, Artificial Societies, and Simulated Organizations. Springer-Verlag, chapter 3, pp. 69–94, (2005).

[Bordini et al., 2005] Bordini, R. H., Dastani, M., Dix, J., El Fallah Seghrouchni, A., (eds): Multi-Agent Programming: Languages, Platforms and Applications. Number 15 in Multiagent Systems, Artificial Societies, and Simulated Organizations. Springer-Verlag, (2005).

[Bordini et al., 2007] Bordini, R., H., Hubner, J., F., Wooldridge, M.: Programming Multi-Agent Systems in AgentSpeak Using Jason. Wiley Series in Agent Technology. John Wiley & Sons, (2007).

[Bordini et al., 2009] Bordini, R., H., Hubner: Agent-Based Simulation Using BDI Programming in Jason. In: [Weyns et al., 2009] Weyns, D., Uhrmacher, A.M. (eds.): Multi-Agent Systems Simulation and Applications. Computational Analysis, Synthesis, and Design of Dynamic Models Series. CRC Press, Florida (2009).

[Bottazzi et al., 2003] Bottazzi, G., Dosi, G., Rebesco, I.: Institutional architectures and behavioural ecologies in the dynamics of financial markets: a preliminary investigation", Technical Report, Laboratory of Economics and Management, Sant' Anna School of Advanced Studies, Pisa, Italy, (2003).

[Bullard et al., 1999] Bullard, J., Duffy, J.: Using genetic algorithms to model the evolution of heterogeneous beliefs. Computational Economics 13 (1), 41–60, (1999).

[Davidsson , 2002] Davidsson, P.: Agent based social simulation: a computer science view, J. Artif. Soc. Social Simulation, 5, (2002).

[Duffy, 2006] Duffy, J.: Agent-based models and human-subject experiments. in: Tesfatsion, L. Judd, K. L. (Eds): Handbook of computational economics. Volume 2 –Agent-based Computational Economics, North Holland, (2006)

[Faratin et al., 1998] Faratin P., Sierra, C., Jennings, N., R.: Negotiation Decision Functions for Autonomous Agents, In: Journal of Robotics and Autonomous Systems, pp. 159-182., (1998).

[Ferber et al., 2009] Ferber, J., Michel, F., Drogoul A.: Multi-Agent Systems and Simulation: A Survey from the Agent Community's Perspective. In: [Weyns et al., 2009] Weyns, D., Uhrmacher, A.M. (eds.): Multi-Agent Systems Simulation and Applications. Computational Analysis, Synthesis, and Design of Dynamic Models Series. CRC Press, Florida (2009).

[Gjerstad, 2004] Gjerstad, S.: The impact of bargaining pace in double auction dynamics. Department of Economics, University of Arizona, (2004).

[Jakieła, 2006] Jakieła, J.: AROMA – Agentowo zorientowana metodologia Modelowania orgAnizacji. WAEiI, Politechnika Ślaska, Gliwice (2006)

[Jakieła et al., 2009] Jakieła J., Pomianek B.: Agent Orientation as a Toolbox for Organizational Modeling and Performance Improvement. International Book Series "Information Science and Computing", Book 13, Intelligent Information and Engi-neering Systems, INFOS 2009, pp. 113-124, (2009).

[Jakieła, Litwin, Olech, 2010a] Jakieła J., Litwin P., Olech M.: Toward the Reference Model for Agent-based Simulation of Extended Enterprises. In: Setlak, G., Markov, K.: Methods and Instruments of Artificial Intelligence, pp. 34-66, (2010).

[Jakieła, Litwin, Olech, 2010b] Jakieła, J., Litwin, P., Olech, M.: MAS Approach to Business Models Simulations: Supply Chain Management Case Study. In: KES AMSTA-2010, Jędrzejowicz, P. Nguyen, N., T., Howlett, R., Lakhmi, C. J., (Eds.), Part II, LNAI 6071, pp. 32-41, Springer-Verlag, Berlin Heidelberg, (2010).

[Jakieła, Litwin, Olech, 2011] Jakieła J., Litwin P., Olech M.: Prototyp platformy symulacji wieloagentowej rozszerzonych przedsiębiorstw. Studia Informatica, vol. 32, Number 2B (97), pp. 9-23, Gliwice (2011).

[Jakieła, Litwin, Olech, 2012] Jakieła J., Litwin P., Olech M.: Multi-Agent Based Simulation as a Supply Chain Analysis Workbench. Issues 6, Springer-Verlag, Berlin Heidelberg, Transactions on Computational Collective Intelligence, LNCS, vol. 7190, pp. 84-104, (2012).

[LeBaron, 2006] LeBaron, B.: Agent-based computational finance in: Tesfatsion, L., Judd, K. L. (Eds): Handbook of computational economics. Volume 2 –Agent-based Computational Economics, North Holland, (2006).

[Luck et al., 2003] Luck, M., McBurney, P., Preist, C.: Agent technology: enabling next generation computing. A roadmap for agent based computing, (2003), www.agentlink.org.

[MacKie-Mason et al., 2006] MacKie-Mason, J. K., Wellman, M. P.: Automated Markets and Trading Agents In: Tesfatsion, L., Judd, K. L. (Eds): Handbook of computational economics. Volume 2 – Agent-based Computational Economics, North Holland, pp. 584-634 (2006).

[Marks, 2006] Marks R.: Market Design Using Agent-Based Models. in: Tesfatsion, L., Judd, K.L. (Eds) : Handbook of computational economics. Volume 2 –Agent-based Computational Economics, North Holland, (2006).

[Milgrom, 2004] Milgrom P.: Putting Auction Theory to Work. Cambridge University Press, (2004).

[North et al., 2007] North, M., J., Macal, C.M.: Managing Business Complexity. Discovering Strategic Solutions with Agent-Based Modeling and Simulation. Oxford University Press (2007).

[Parunak et al., 1998] Parunak, H., V., D., Savit, R., Riolo, R., L.: Agent-based modeling vs. equation-based modeling: A case study and users' guide. In J. S. Schiman, R. Conte, and N. Gilbert, editors, Proceedings of the 1st Workshop on Modeling Agent Based Systems, MABS'98, volume 1534 of LNAI. Springer-Verlag, (1998).

[Rao et al., 1992] Rao, A., S., Georgeff, M., P.: An abstract architecture for rational agents. In: Nebel, B., Rich, C., Swartout, W., R. (Eds), Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR'92), pp. 439–449. Morgan Kaufmann Publishers, (1992).

[Turban et al., 2011] Turban, E., King, D., Viehland, D., Lee, J.: Electronic commerce. A managerial perspective. Prentice-Hall, (2011).

[Weyns et al., 2009] Weyns, D., Uhrmacher, A.M. (eds.): Multi-Agent Systems Simulation and Applications. Computational Analysis, Synthesis, and Design of Dynamic Models Series. CRC Press, Florida (2009).

[Zambonelli et al., 2002] Zambonelli, F., Parunak, H., V., D.: From design to intention: signs of a revolution. In Proceedings of the first international joint conference on Autonomous agents and multiagent systems, pp. 455–456. ACM Press, (2002).

Authors' Information

Jacek Jakieła, Ph.D., Eng. *– Department of Computer Science FMEA RUT; Powstancow Warszawy ave. 12, 35-959 Rzeszow, Poland; e-mail: jjakiela@prz.edu.pl*

Major Fields of Scientific Research: Software Development Methodologies, Agent and Object-Oriented Business Modeling, Internet Enterprises Models, Computational Organization Theory and Multi-Agent Based Simulation of Business Architectures.

Paweł Litwin, Ph.D., Eng. *– Department of Computer Science FMEA RUT; Powstancow Warszawy ave. 12, 35-959 Rzeszow, Poland; e-mail: plitwin@prz.edu.pl*

Major Fields of Scientific Research: Applications of Neural Networks in Mechanics, Computer Simulations, Finite Element Method.

Marcin Olech, M.Phil., Eng. *– Department of Computer Science FMEA RUT; Powstancow Warszawy ave. 12, 35-959 Rzeszow, Poland; e-mail: molech@prz.edu.pl*

Major Fields of Scientific Research: Multi-agent Based Simulations, Artificial Intelligence Applications in Industry.