Implementing an Agent Based Artificial Stock Market Model in JADE – An Illustration

PN Kumar[1], Rupika R[2], Vennilla S[3], Abinaya K[4], VP Mohandas[5]
[1,2,3,4]Dept. of CSE , [5]Dept. of ECE, Amrita Vishwa Vidyapeetham, Ettimadai, Coimbatore, 641 112, India
pn_kumar@cb.amrita.edu[1], rupika30@gmail.com[2], vennilasivanesan@gmail.com[3],
abinaya.km1991@gmail.com[4] , vp_mohandas@amrita.edu[5]

Abstract— **Agent-based approach to economic and financial analysis is a suitable research methodolgy for developing and understanding the complex patterns and phenomena that are observed in economic systems. In agent-based financial market models, prices can be endogenously formed by the system itself as the result of interaction of market participants. By using agents for the study, heterogeneous, boundedly rational, and adaptive behaviour of market participants can be analysed and its impact assessed. The collective behaviour of such groups is determined by the interaction of individual behaviours distributed across the group. This being the scenario prevailing in stock markets, agent based models are suitable for the study.**

Through this paper, we have attempted to illustrate a detailed implementation of multi agents in an artificial stock market invoking the agent-based methodology on Java Agent Development (JADE) environment, a platform to develop multi-agent systems. The Extended Glosten and Milgrom Model, an agent based artificial stock market model, has been chosen to depict the multi-agent environment model in JADE.

Keyword-Agent based modelling, artificial stock market, investors, market makers, JADE, EGM Model

I. INTRODUCTION

Financial markets are complex entities and hence models are required to study the phenomena prevailing therein. Artificial Stock Markets (ASM) are models for studying the link between individual investor behavior and financial market dynamics. In agent based ASM, prices arise from simulating the interactions of autonomous entities with different profit-making strategies. ASM are often computational models of financial markets, and are usually comprised of a number of heterogeneous and boundedly rational agents. These models are built for the purpose of studying agents' behavior, price discovery mechanisms, the influence of market microstructure etc.

A similar bottom-up approach has been utilized in agent-based computational economics (ACE) [1] to [5], which illustrates the distributed nature of trading in financial markets by computational study of economies modeled as evolving systems of autonomous interacting agents that correspond to the trading parties. A large number of agent-based stock market models have been proposed by researchers to study various aspects of the stock market behaviour [6] to [16]. The Extended Glosten and Milgrom (EGM) model, a multi agent artificial stock market model proposed by Das [17], [18], the ASM proposed to be implemented in this paper, studies the information based market making strategy. The EGM model has been implemented on Java Agent Development **(**JADE), a framework suitable to develop multi-agent systems described in [19]. Moreover, the behaviour of the traders is designed to be continuous and autonomous [22]. Prior to delving into the ASM per se, a brief overview of the market microstructure is given in the next section, which would give a perspective of the intended work and also help clarify the setting of the ASM.

II. THE MARKET MICROSTRUCTURE

The field of market microstructure is concerned with the specific mechanisms and rules under which trades take place in a market and how these mechanisms impact price formation and the trading process. Main factors that describe a market structure are [20], [22]:

A. *Market participants*

- Investors: Traders, not part of the market organization.
- Market Makers: Responsible for an orderly and stabilized market.
- Brokers: Traders, part of the market organization, who handles orders for customers.

B. *Traded instruments*

The objects traded in a market are called traded instruments. Stocks, the instruments traded, are financial assets that represent ownership of corporate assets.

C. *Orders*

- Market order: Specifies size of the stock and type of the trade (buy/sell) for the trade.
- Limit order: In addition to the specifications of a market order, it specifies the length of order validity with respect to time or change in price.

D. *Trading Sessions*

- Discrete: Occurs at well specified time. During a call, all the trade requests are aggregated and a single price is set.
- Continuous: Trade can occur at any time when the market is open.

E. *Execution Systems*

- Quote-driven: Investors or brokers cannot trade with each other; they must involve the market maker.
- Order-driven: Buyers and sellers can trade directly without the intermediation of the market maker.

Rules regulate the market organization, viz trade prices, time interval in discrete trade sessions, quantity and quality of information disseminated to the market participants etc.

III. Extended Glosten AND MILGROM Model

The Extended Glosten and Milgrom (EGM) model is an extension of the Glosten Milgrom information based model, proposed to show the influence of informational asymmetry on the bid-ask spread, based on the learning market maker from Das [17]. The market maker tries to discover the fundamental value of a stock by means of Bayesian learning. He determines the bid and ask quotes based on his expectation of the real value, the order flow, taking into account his prior knowledge regarding the ratio of informed and uninformed traders. A nonparametric density estimation technique is used to maintain a probability distribution over a range of expected true values. The market-maker uses these probability estimates to set bid and ask prices.

A. The Behaviour of the Market Maker

An investor, based on its trading strategy, places a buy/sell/hold order. It is the market maker's task to carry out the rest of the actions within a trading round. On his turn, the market maker needs to carry out the following tasks:

- Receive and execute orders;
- Update the probability density estimate (PDE) based on the received orders;
- Adjust the bid and ask quotes according to the changes in the PDE.

The market maker executes sell orders at the current bid price and buy orders at the current ask price. Information regarding the fundamental value of the stock diffuses from the informed traders to the market maker in this way. A series of sell orders might indicate that the fundamental value is lower than the current bid price, and a series of buy orders might indicate that the fundamental value is higher than the current ask price. However, the market maker will have to take into account the noise incorporated by the orders of the noisily informed traders, and the noise implied by the orders submitted by the uninformed traders. The market maker aims to set bid and ask prices to capture the underlying fundamental value of the stock. The fundamental value is known only by the (perfectly) informed investors, and is not known by the market maker. The main task of the market maker is to learn this value. In order to ensure an efficient market, he tries to track this value by maintaining a probability density estimate (PDE) over a set of possible true values and the probability estimates attached to each of them. The range of possible values, the corresponding probabilities, and the learning process of the market maker is based on a set of current and a-priori known information. The above steps are carried out in a continuous manner, thus resulting in a rather realistic simulation of an actual stock market, and elaborated to form the market making algorithm, given at Section VI.

B. Inventory Control

A realistic model for market-making necessitates taking portfolio risk into account as well, and controlling inventory in setting bid and ask prices. If the market-maker has a long position in the stock, minimizing portfolio risk is achieved by lowering both bid and ask prices (effectively making it harder for the market-maker to buy stock and easier for it to sell stock), and if the market-maker has a short position, inventory is controlled by raising both bid and ask prices. Inventory control can be incorporated into the architecture of our market making algorithm by using it as an adjustment parameter applied after bid and ask prices have been determined.

C. Important Issues

This model consist of “informed" trading agents who decide to trade based on received signals of the true or fundamental value of the stock, and “uninformed" trading agents who trade for exogenous reasons and are modeled as buying and selling stock randomly. This model (EGM) is combined with the learning market maker, which Das [18] describes in a turn-based model, with investors that interact asynchronously and autonomously. The described model is simulated in JADE in continuous time simulation, since most financial markets are continuous and asynchronous in nature. Hence we choose to simulate the same as has been done in [22].

IV. JADE

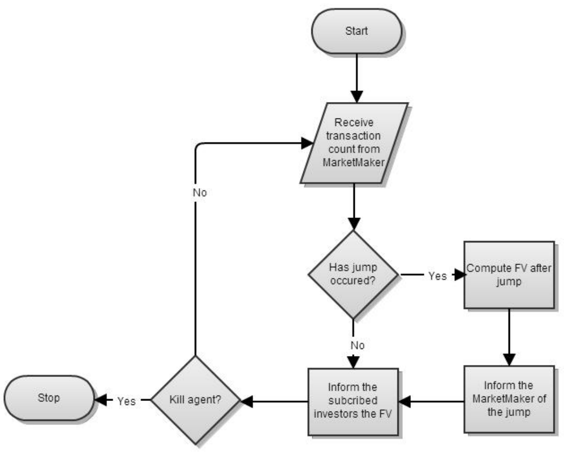
The implementation of the trading environment is based on JADE, a framework to develop multi-agent systems. JADE includes two main products: an agent platform and a package to develop Java agents. JADE (Java Agent Development Framework) is a software development framework aimed at developing multi-agent systems and applications conforming to FIPA standards for intelligent agents. It includes two main products: a FIPA-compliant agent platform and a package to develop Java agents. JADE has been fully coded in Java and an agent programmer, in order to exploit the framework, should code his/her agents in Java, following the implementation guidelines described in this programmer's guide [19]. The steps to launch the project in JADE are illustrated at Appendix.

V. SYSTEM DESIGN

A. The Basic Trading Structure and The Agents

Trading in a market is organized around three main components: the market makers, investors and information source. Based on these roles, three agents are implemented in this system [22]:

- *The Investor:* This agent represents any entity willing to perform trade with the MarketMaker. Depending on the investor’s trading strategy, it buys, sells or holds for a particular instance of the stock prices. The type of investors differs based on its behaviour.
- *The MarketMaker:* The MarketMaker agent is the main authority that controls the market. It is the agent that takes care to open and close the market. It is responsible for the relation between the marketplace and
 the investors.
- *The NewsManager:* The NewsManager represents the information-source component. This agent is started
immediately after launching the marketplace and it generates fundamental value related information. The NewsManager agent sends new information to all agents who have subscribed to it. This agent informs the fundamental value evolved after a jump while it informs the market maker about the occurrence of such a jump (Fig.1). Currently this agent is part of the marketplace application.



Start
Receive transaction count from MarketMaker
No
Has jump occured?
Yes
Compute FV after jump
No
Inform the MarketMaker of the jump
Inform the subcribed investors the FV
Kill agent?
Yes
Stop

Fig. 1: NewsManager agent

B. The investors’ Behaviour

Investors are differentiated based on the information they receive regarding the fundamental value. There are two types of investors:

- Informed traders (perfectly informed/noisily informed).
- Uninformed traders/noisy traders.

Perfectly informed traders observe the correct fundamental value (V_t), while noisily informed investors observe a distorted fundamental value $W_t = V_t + \psi(0,\sigma_w)$. Here, $\psi(0,\sigma_w)$ represents a sample from a normal distribution with mean zero and variance σ. Finally, uninformed traders do not know what the underlying fundamental value is, and they trade randomly. As shown in Fig. 2, informed traders decide whether to trade or not, based on their observation of the fundamental value. An informed trader will buy if the fundamental value that he observes is higher than the market maker's ask price, i.e. if $V_t > P^t_A$ in the case of perfectly informed traders, and $W_t > P^t_A$ in the case of noisily informed traders. He will sell if the fundamental value that he observes is below the bid price, i.e. if $V_t < P^t_B$ or $W_t < P^t_B$. He will place no order if the observed fundamental value is within the bid-ask spread, i.e. $P^t_B <= V_t <= P^t_A$ or $P^t_B <= W_t <= P^t_A$. Note: The noisily informed traders behave very similarly to the perfectly informed traders, the only difference being that the former make their decisions based on a distorted fundamental value, V+ noise. Uninformed traders (Fig. 3) place buy and sell orders with equal probability ($\eta < 0.5$).They can also decide not to place orders with probability ($1-2\eta$).

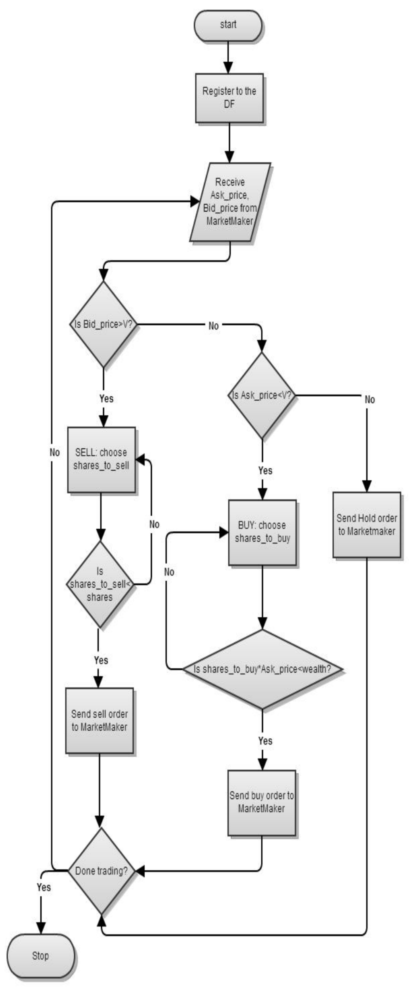

start
Register to the DF
Receive Ask_price, Bid_price from MarketMaker
Is Bid_price>V?
No
Is Ask_price<V?
No
Yes
No
SELL: choose shares_to_sell
Yes
BUY: choose shares_to_buy
Send Hold order to Marketmaker
No
Is shares_to_sell< shares
No
Is shares_to_buy*Ask_price<wealth?
Yes
Send sell order to MarketMaker
Yes
Send buy order to MarketMaker
Done trading?
Yes
Stop

Fig. 2: Perfectly Informed Trader Agent

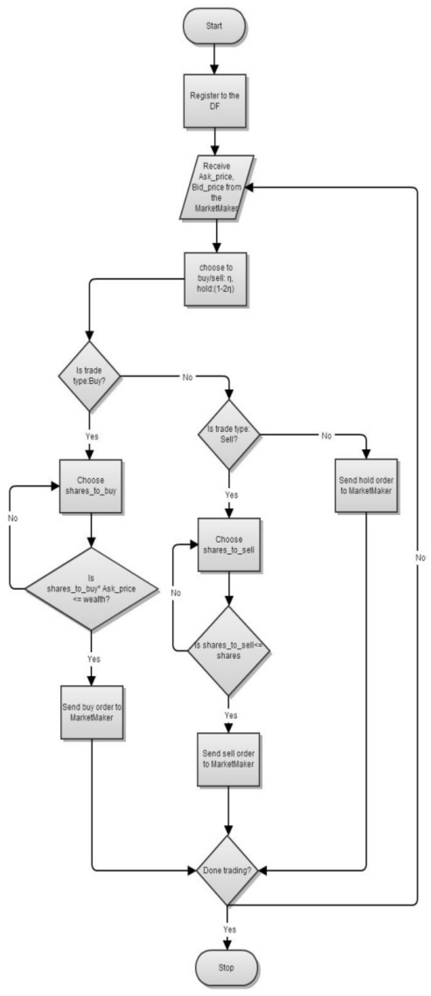

Start
Register to the DF
Receive Ask_price, Bid_price from the MarketMaker
choose to buy/sell: η, hold:(1-2η)
Is trade type:Buy?
No
Is trade type: Sell?
No
Yes
Choose shares_to_buy
Send hold order to MarketMaker
Yes
No
Choose shares_to_sell
Is shares_to_buy* Ask_price <= wealth?
No
No
Is shares_to_sell<= shares
Yes
Send buy order to MarketMaker
Yes
Send sell order to MarketMaker
Done trading?
Yes
Stop

Fig. 3: Noisy Trader Agent

VI. DESIGN OF THE EGM MODEL

A. *System Design of the EGM Model*

In both the original Glosten and Milgrom [21] model and the extended EGM version described by Das [17], [18] trading sessions are continuous and the execution system is quote-driven. There is one stock traded. One market maker and multiple investors are represented. The stock does not pay dividends. It is assumed that the stock has an underlying fundamental value, which is generated exogenously to the market.The underlying fundamental value of the stock at time t is V_t which follows a jump process, being constant most of the time, and changing occasionally. Jump in the fundamental value occurs with some probability at every trading period that is at every discrete point in time. The jump process is modeled as a random process following $V_{t+1} = V_t + \omega(0,\sigma)$ where $\omega(0,\sigma)$ represents a sample from a normal distribution with mean zero and variance σ2. The NewsManager agent takes care of this. Once a jump has occurred, it informs its subscribers, the informed trader agents, of the new fundamental value (the noisily informed traders are aware of the new fundamental value weighted by a noise factor). The NewsManager then informs the MarketMaker agent of the occurrence of such a jump. The system design is illustrated in Fig. 4.

B. The Information Set of the Market Maker

The market maker does not know the fundamental value, but, in order to ensure an efficient market, he tries to capture it by maintaining a PDE over a range of expected true values [22]. The probability estimates are initialized according to the normal distribution. The initial bid and ask prices are calculated from this initial PDE and a-priori expectations of the market maker. After initialization, trading rounds start. The market maker executes sell orders at the current bid price (P_B) and buy orders at the current ask price (P_A). The private information regarding the fundamental value is revealed implicitly by the type of the orders submitted by the (informed) traders.

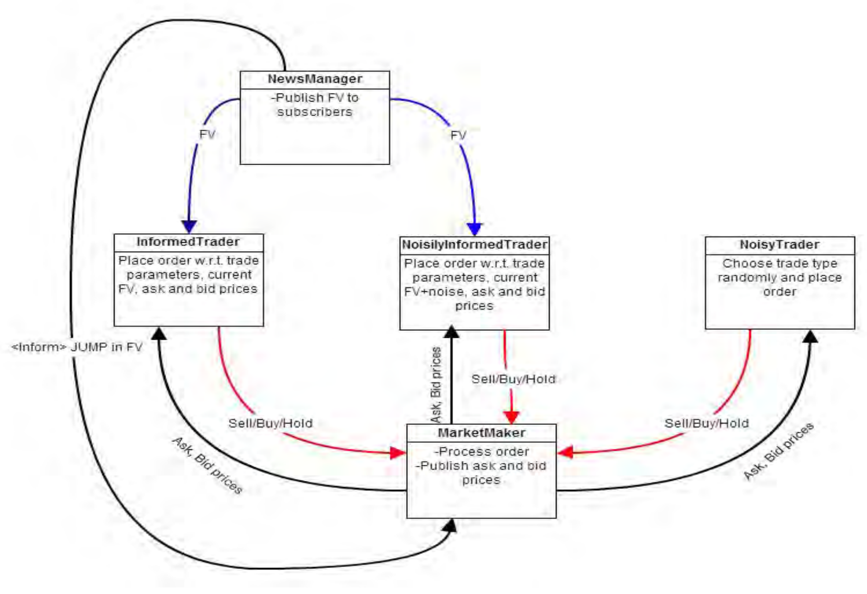

NewsManager
-Publish FV to subscribers
FV
FV
InformedTrader
Place order w.r.t. trade parameters, current FV, ask and bid prices
NoisilyInformedTrader
Place order w.r.t. trade parameters, current FV+noise, ask and bid prices
NoisyTrader
Choose trade type randomly and place order
<Inform> JUMP in FV
Ask, Bid prices
Sell/Buy/Hold
Sell/Buy/Hold
Sell/Buy/Hold
Ask, Bid prices
Ask, Bid prices
MarketMaker
-Process order
-Publish ask and bid prices

Fig. 4 : System Design

Information regarding the fundamental value of the stock diffuses from the informed traders to the market maker in this way. A series of sell orders might indicate that the fundamental value is lower than the current bid price, and a series of buy orders might indicate that the fundamental value is higher than the current ask price. However, the market maker will have to take into account the noise incorporated by the orders of the noisily informed traders, and the noise implied by the orders submitted by the uninformed traders.

The market maker aims to set bid and ask prices to capture the underlying fundamental value of the stock. The fundamental value is known only by the (perfectly) informed investors, and is not known by the market maker. The main task of the market maker is to learn this value. As mentioned above, he tries to do this by maintaining a set of possible true values with probability estimates attached to each of them. The range of possible values, the corresponding probabilities, and the learning process of the market maker is based on a set of current and a-priori known information. Current information refers to the actual order placed, and a notification when a change in the fundamental value occurs. The a-priori information set of the market-maker contains:

- Fraction of perfectly informed traders (α_in),noisily informed traders (α_nia) and noisy traders(α_no);
- Probability for an uninformed trader to trade (η);
- Initial fundamental value Vo;
- Distribution function of the jump process (ω (0,σ));
- Distribution function of the noise process ($\psi(0,\sigma_w)$).

C. The Behaviour of the Market Maker

After an investor has placed an order, it is the market maker’s task to carry out the rest of the actions within a trading round. On his turn, the market maker needs to carry out the following tasks:

- Receive and execute orders;
- Update the PDE based on the received orders;
- Adjust the bid and ask quotes according to the changes in the PDE. The above steps are carried out in a continuous manner, thus resulting in a rather realistic simulation of an actual stock market [22]. These steps are elaborated to form the following market making algorithm.

D. The Market Making Algorithm

A round in the market making algorithm (Fig. 5) consists of the following steps:

I. Initialise Vo, Ask_price and Bid_price as 100.
II. Compute V_{min}=Vo - 4σ and V_{max}= Vo + (4σ-1).
III. Populate the PDE 2-D array as pde[V_i][Pr(V= V_i)], following a uniform distribution.
IV. Register itself to the Directory Facilitator (DF).
V. Call RequestPerformer:

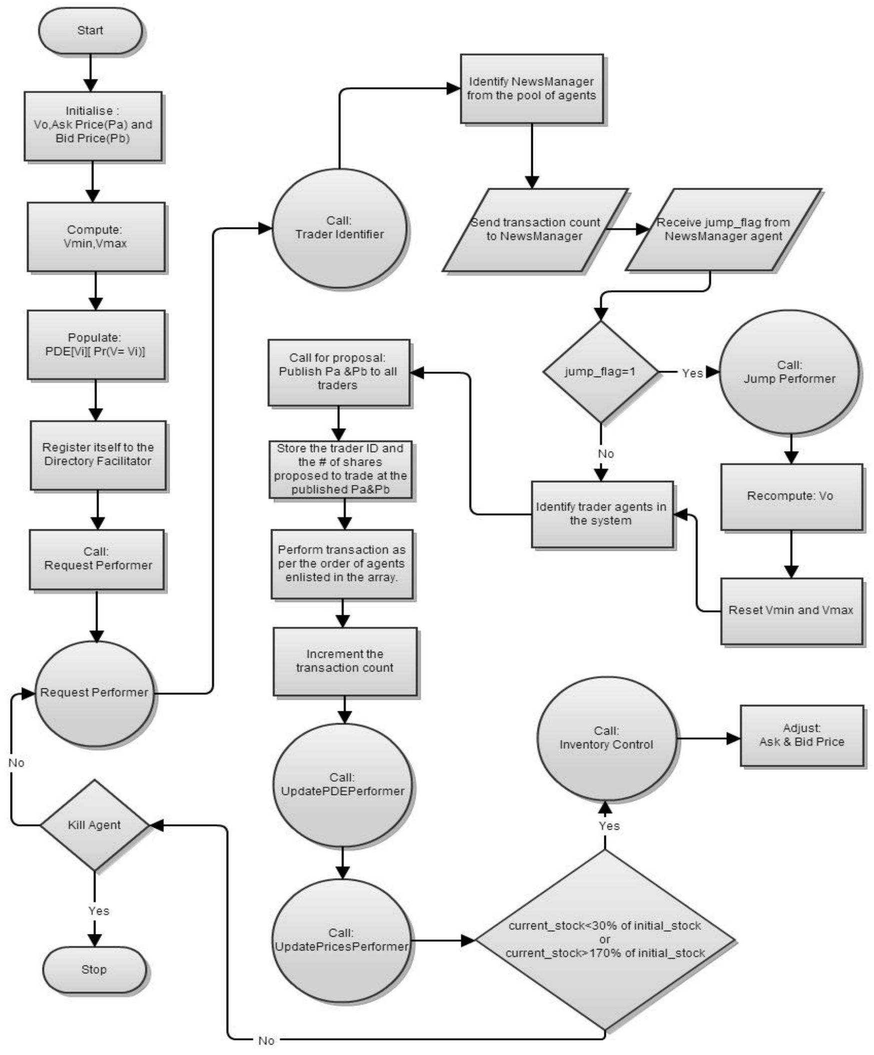

Start
Initialise : Vo,Ask Price(Pa) and Bid Price(Pb)
Compute: Vmin,Vmax
Populate: PDE[Vi][Pr(V= Vi)]
Register itself to the Directory Facilitator
Call: Request Performer
Request Performer
Kill Agent
No
Yes
Stop
Call: Trader Identifier
Identify NewsManager from the pool of agents
Send transaction count to NewsManager
Receive jump_flag from NewsManager agent
jump_flag=1
Yes
No
Call: Jump Performer
Recompute: Vo
Reset Vmin and Vmax
Identify trader agents in the system
Call for proposal: Publish Pa &Pb to all traders
Store the trader ID and the # of shares proposed to trade at the published Pa&Pb
Perform transaction as per the order of agents enlisted in the array.
Increment the transaction count
Call: UpdatePDEPerformer
Call: UpdatePricesPerformer
current_stock<30% of initial_stock or current_stock>170% of initial_stock
Yes
No
Call: Inventory Control
Adjust: Ask & Bid Price

Fig. 5: MarketMaker Agent

RequestPerformer: Behaviour of the MarketMaker that takes care of the continuous trade. This happens cyclically until the MarketMaker agent is manually terminated.

1. Call TraderIdentifier.

   TraderIdentifier: *Behaviour of the MarketMaker called to identify all the trader agents and the NewsManager agent.*

   a. Identify NewsManager.
   b. Send transaction count to the NewsManager.
   c. Identify trader agents in the system.
   d. Receive jump_flag from NewsManager agent; if jump_flag=1, jump has occurred, call JumpPerformer.

   JumpPerformer*: Behaviour of the MarketMaker that is invoked each time jump_flag=1, ie. the MarketMaker is informed of the occurrence of a jump.*

   a. Compute Vo=Avg[∑(Vi*Pr(V=Vi))+Avg(Prices)]

   where **Avg(Prices)**: average of the sum of various Ask_price and Bid_price occurrences since the last jump.

   b. Reset V_{min} and $V_{max.}$

2. Send call for proposal message, publishing the Ask_price and the Bid_price, to all the available traders.
3. Store the Agent identities (Aid) of the traders who proposed, their corresponding number of shares to trade and the prices at which trade is proposed in respective arrays.
4. Perform a transaction as per the order of agents enlisted in the arrays.
5. Increment the transaction count.
6. Call UpdatePDEPerformer.

   UpdatePDEPerformer: As explained earlier, the PDE array is updated.

7. Call UpdatePricesPerformer.

   UpdatePricesPerformer: As explained earlier, the ask and bid prices are adjusted.

8. If current_stock < 30% of initial_stock OR current_stock > 170% of initial_stock,
9. Call InventoryControl.

   InventoryControl: Behaviour of the MarketMaker that is invoked whenever the current stock is out of the tolerant range, ie. 80% of initial_stock < current_stock < 120% of initial_stock.

   a. Set Adjustment factor= - γ *((current_stock-initial_stock) /initial_stock) where $\gamma = 10$.
   b. Reset Ask_price= Ask_price+ Adjustment factor
   c. Reset Bid_price= Bid_price+ Adjustment factor

10. Go to step 1.

Note: Continuous trading occurs through Step 2 to Step 7.

E. The PDE

The range of possible values maintained by the market maker to estimate the fundamental value, as proposed by Das [18] varies from Vmin = Vo-4σ to Vmax = Vo + 4σ-1. (1)

The values are given in INR, in intervals of 25 Paise. The market maker keeps a PDE over the whole range of possible values determined in this way. Whenever a jump in the fundamental value occurs, the range of possible values is re-centered. That is, after each jump the market maker sets Vo to the current expected value, as follows:

$$V_o=\sum Vi*Pr(V=Vi)))+Avg(Prices)] \quad (2)$$

where Avg(Prices) is the average of the sum of various Ask_price and Bid_price occurrences since the last jump. V_i is the possible fundamental values already present in the PDE and $Pr(V=V_i)$ is its corresponding probability value. The range of possible values is reset at 4σ distance from the new V_o.

F. Initialization

At the beginning of the simulation, and after a jump in the fundamental value, the probabilities are initialized to follow a normal distribution within the given range of possible values. Accordingly:

$$Pr(V= V_i) = \int N(0, 4\sigma)dx;\ \text{where } V_i\ \varepsilon\ \{Vo-4\sigma,\ldots,Vo+4\sigma-1\} \text{ and integrated from } V_i \text{ to } V_{i+1} \quad (3)$$

Here, N is the normal density function in *x* with mean zero and standard deviation 4σ. The array is kept in a normalized state at all times, so the entire probability mass for V lies within it. The probabilities are updated whenever a new order arrives. It should be noted that the probabilities for the V values at the beginning of the simulation are of uniform distribution.

G. Updating the PDE

When an order arrives, the market maker updates the probabilities for V_i by scaling the distribution, based on the type of the order. The new probabilities at time t+1, $Pr(V = V_{t+1}^i)$ will be set to the current probabilities given that the corresponding order arrived $Pr(V = V_{t+1}^i|Order)$. The values are updated using Bayes' Rule according to the following generalized equation:

$$Pr\ (Vi\ |\ Order) = [Pr\ (Order\ |\ Vi) * Pr\ (V = Vi)\ ]/Pr\ (Order) \quad (4)$$

where the Order can be:Buy,Sell, or No order/Hold.

The prior value probabilities Pr(V = Vi) refer to the current probability estimates. Pr(Order) represents the prior probability of a certain type of order, while Pr(Order|V =Vi) is the conditional probability of a certain type of order. The prior probability of an order is the cumulative conditional probability of that action weighted by the probability estimate of the given values.

For V from V_{min} to V_{max}, $Pr\ (Order) = \sum Pr\ (Order|Vi) * Pr\ (V = Vi)$ (5)

The conditional probability of placing a certain order depends on the fraction and type of the investors involved. In general:

$$\begin{aligned}\Pr(\text{Order}|V = Vi) = &(\alpha\_no) \Pr(\text{Order from noisy investors}|V = Vi) + \\ &(\alpha\_in) \Pr(\text{Order from perfectly informed investors}|V = Vi) + \\ &(\alpha\_nia) \Pr(\text{Order from noisily informed investors}|V = Vi) \quad (6)\end{aligned}$$

The model assumes that in case of uninformed traders the probabilities are known and these are independent of the current market situation. Accordingly, Pr(Buy from uninformed investors|V = Vi) = Pr(Sell from uninformed investors|V = Vi) = η. Consequently, Pr(No order from uninformed investors|V = Vi) = 1-2η. Let us now elaborate on how the conditional probabilities of the various order types can be computed for the market system with the three types of investors.

Case (i): Sell Order

If $V_i<P_B$, $\Pr(\text{Sell}|V=V_i) = (\alpha\_no*\eta)+ \alpha\_in+ \alpha\_nia*\Pr(V_i+noise<P_B)$ (7)

If $V_i>=P_B$, $\Pr(\text{Sell}|V=V_i) = \alpha\_no* \eta$. (8)

Case (ii): Buy Order

If $V_i>P_A$, $\Pr(\text{Buy}|V=V_i) = (\alpha\_no*\eta)+ \alpha\_in+ \alpha\_nia*\Pr(V_i+noise>P_A)$ (9)

If $V_i<=P_A$, $\Pr(\text{Buy}|V=V_i) = \alpha\_no* \eta$. (10)

Case (iii): No Order

If $P_A<V_i<P_B$, $\Pr(\text{Hold}|V=V_i) = (\alpha\_no*(1-2\eta))+ \alpha\_in+ \alpha\_nia*\Pr(P_A<V_i+noise<P_B)$. (11)

Otherwise, $\Pr(\text{Hold}|V=V_i) = \alpha\_no*(1-2\eta)$. (12)

Once the market maker receives an order or (an acknowledgement of no order), the agent updates the PDE as shown in Eq.(6.4) using one of the equations between (6.7) to (6.12) depending on the type of order and the V_i value. In other words, for the range of V values present in the PDE, the agent iteratively calculates the $\Pr(\text{Order}|V=V_i)$. Consequently, Pr(Order) is computed. These values are then substituted in equation (6.4).

It should be noted that receiving a sell order would lead to density functions that are symmetric to the ones resulted after receiving a buy order. The case of no orders leads to different results. If the selected investor decides not to place any order the market maker believes that he managed to capture the bid and ask values. Consequently, he increases the probabilities for the values in between the bid and ask, and lowers the probabilities for the rest of the values.

H. Adjusting the Bid and Ask Quotes

The market maker tries to set the bid and ask prices such that these reflect the fundamental value of the stock. As he does not know this value, the new bid and ask prices will be based on the expected value, given the adjusted PDE after arrival of an order. The bid price is set to the expectation of the true value given the probability that a sell order will arrive. Similarly the ask price is set to the expectation of the true value given the probability that a buy order arrives. In other words,

$$P_B=E[V|\text{Sell}] \quad (13)$$

$$P_A=E[V|\text{Buy}] \quad (14)$$

By definition, in order to estimate the expectation of the underlying value, it is necessary to compute the conditional probability that the true price equals a certain value (V = x; x > =0) given that a particular type of order is received. With [Vmin; Vmax] as the range of possible values, the refined equations are arrived at:

$$P_B = \sum_{Vi=Vmin}^{Vmax} \text{Vi} * \Pr(\text{Vi} \,|\text{Sell}) \quad (15)$$

$$P_A = \sum_{Vi=Vmin}^{Vmax} \text{Vi} * \Pr(\text{Vi}|\text{Buy}) \quad (16)$$

Using the PDE values, the above equations are updated as follows:

$$P_B = \frac{1}{\Pr(sell)} \sum_{Vi=Vmin}^{Vmax} \text{Vi} * \Pr(\text{Sell}|\text{Vi}) * \Pr(V = \text{Vi}) \quad (17)$$

$$P_A= \frac{1}{\Pr(Buy)} \sum_{Vi=Vmin}^{Vmax} \text{Vi} * \Pr(\text{Buy}|\text{Vi}) * \Pr(V = \text{Vi}) \quad (18)$$

Thus, the new prices emerge as a result of each trade taking place in the market.

I. Implementing the Inventory Control

A control over the stock inventory is maintained by ensuring that the stock stays within a particular pair of lower and upper bounds. Consider that the lower bound is set at 30 percent of the initial stock while the upper bound is set at 170 percent. Logically speaking, the stock tends to go below the lower limit when there is an excess of buy orders. The stock is then said to be in short position. This problem can be controlled by increasing the ask and bid prices. This will lead to an increase in the sell orders. On the other hand, the stock tends to go above the upper limit when there is an excess of sell orders. The stock is then said to be in long position. In this case, the ask and bid prices are decreased to encourage buy orders. To implement inventory control, a risk aversion factor, γ, is introduced. The adjustment is calculated as follows: Adjustment= - γ* ((current_stock - initial_stock) / initial_stock) (19)

The prices are then modified by being appended with the calculated adjustment factor as mentioned below.

$$P_A = P_A + \text{Adjustment} \text{ and } P_B = P_B + \text{Adjustment}. \qquad (20)$$

One might think that 80% (120%) of the initial stock is too high (low) a value to be used as a proper lower (upper) limit for the stock. But one must keep in mind that the trade takes place in a continuous manner. Despite knowing that the stock is in short (long) position, the market maker has to finish the trades that he had agreed to before realizing the necessity for inventory control. Hence the rather immoderate limits give the market maker a sort of buffer period within which the inventory can be brought back under control.

VII. EXPERIMENTAL EVALUATION AND CONCLUSION

A. *Experimental Settings*

1) *Fundamental value related settings*

- The initial value (V_0) is Rs. 100.
- The standard deviation in the distribution function of the jump process (ω(0,σ)) is Re. 1.
- The jump in the fundamental value is set to occur after every 10 transactions.
- The σ used to compute V_{min} and V_{max} is 50 paise.

2) *Investor related settings:*

- The fraction of perfectly informed traders, noisily informed traders and noisy traders is set at 0.3, 0.5 and 0.2 respectively; here a total of 50 traders for the experiment are considered.
- The probability (η) that uninformed traders place a buy order (and, respectively, sell order) is set to 0.5. Consequently, the probability that uninformed traders do not trade is 0 (these values ensures that noisy agents never hold, maintaining liquidity in stock).
- All investors place market orders for a chosen quantity of the risky stock. The investors do not withdraw their order once it is submitted.
- The noise value for the noisily informed traders is set to range from 1 to 1.5, incrementing at 0.05; this is selected using the random function during the creation of the trader agent. While in prevalent ASM systems, a noise factor is fixed, common to all the noisily informed traders, in this system varying noise values are chosen as an attempt to create heterogeneity among the noisily informed traders.
- To maintain an elementary sense of inventory control among the traders, constraints are added such that the trader does not trade beyond a limit value in wealth and stock.

3) *Market maker related settings:*

- The market maker knows the fraction of traders present in the system.
- The market maker also knows the noise distribution.
- For inventory control, the short stock position is at 30% of the initial stock while the long stock position is at 170% of the initial stock. γ is taken as 10.

B. *Experimental Results and Inferences*

Fig.6 shows the graphical outputs of the system with inventory control and Fig.7 without inventory control. Initially, the ask and bid prices are set equal to the Vo value, 100. Then the trade begins. After each order, the PDE is updating and if it is a buy order, the ask price is modified, else, if it is a sell order, the bid price is modified. As is evident in both the graphs, there are quite some fluctuations. This is due to the presence of noisily informed traders and noisy traders in the market. Their erratic behaviour leads to unpredictable orders flowing to the market maker.

In our experiment, a jump is set to occur after every 100 transactions. If this was increased to say, 500 transactions, the fluctuations will reduce as the market maker is provided with ample time to settle into a steady pace. The blue lines in the graphs denote the fundamental value estimated by the market maker as a result of his Bayesian learning. This is nothing but the average of the pair of asks and bid prices since the last jump. The actual fundamental value of the stock, as maintained by the NewsManager agent is shown in Fig. 8. Comparing the blue lines of the graphs with the fundamental value in the above figures, it is evident that the market maker does a satisfactory job of tracking the fundamental value. Graphically it is seen that inventory control results in lesser fluctuations in the prices. The result of using inventory control, although not evident in Fig.7, is quite significant when analyzing the output data. In the output data corresponding to the system without inventory control, the market maker's stock diminishes to a mere 1. This has the danger of the market maker running out of stocks to trade. This, in real life, spells a market catastrophe. The main role of the market maker is to maintain liquidity in the market. This is an impossible task if the market maker runs out of stock and is solely dependent on a sell order to get back into the market. Thus, the system implementing inventory control is the better choice for emulating an efficient stock market.

C. Deviation from the Existing Model

- *Computing Vo after a jump*
  Whenever the MarketMaker gets to know that a jump has occurred, the Vo is recalculated. It is equated to the average of the estimated V value (computed using the PDE) and the average of the ask and bid prices that the agent has been maintaining since the last jump.
- *Probabilistic estimation of the order*
  This refers to the calculation of Pr(Order|V_i). In the prevalent papers, consideration is given to either perfectly informed traders and noisy traders or noisily informed trader and noisy traders when estimating Pr(Order|V_i). But in this system, all the three types of traders coexist. Accordingly, modification has been done to the probability estimation formula to make room for the three traders.

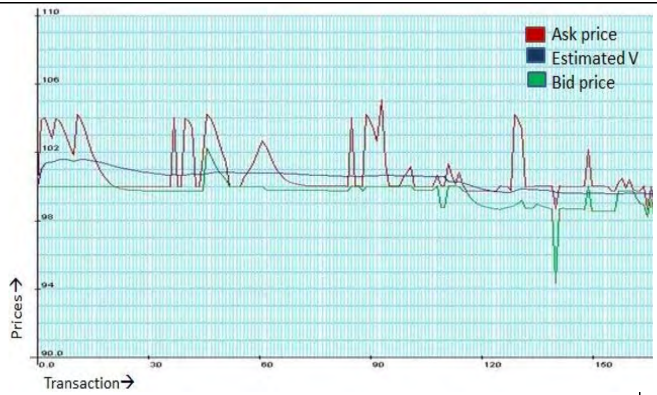
D. Conclusion

A continuous-time, autonomous, asynchronous agent-based model of the ASM based on the theoretical concepts enunciated in EGM has been implemented in JADE. A step by step illustration of implementing a multi-agent system has been elaborated. This system can be used to carry out comparative studies of the real and artificial stock markets by exploiting this flexibility, as follows:

- By adjusting the ratio of the different type of traders and comparing the resulting graph with the real-stock data, the distribution of the varied traders in the real market can be sufficiently assumed.
- For a given initial stock value, the most efficient inventory control tolerance range can be ascertained.

E. Future Work

- Multiple stocks can be considered, and investors could be modeled as portfolio managers in multiple stocks.
- More market makers could operate on the same market.
- Brokers could be implemented.
- In accordance with the implementation of our system, inclusion of new traders could be done dynamically.
- To create a more realistic simulation of the stock market, trade can be carried out over a distributed platform.



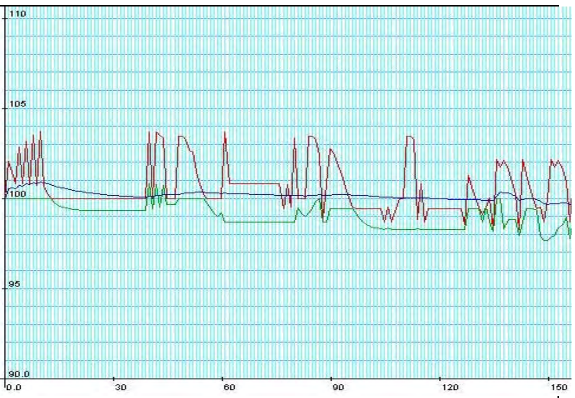


Fig. 6: The emergent prices with inventory control	Fig. 7: The emergent prices without inventory control

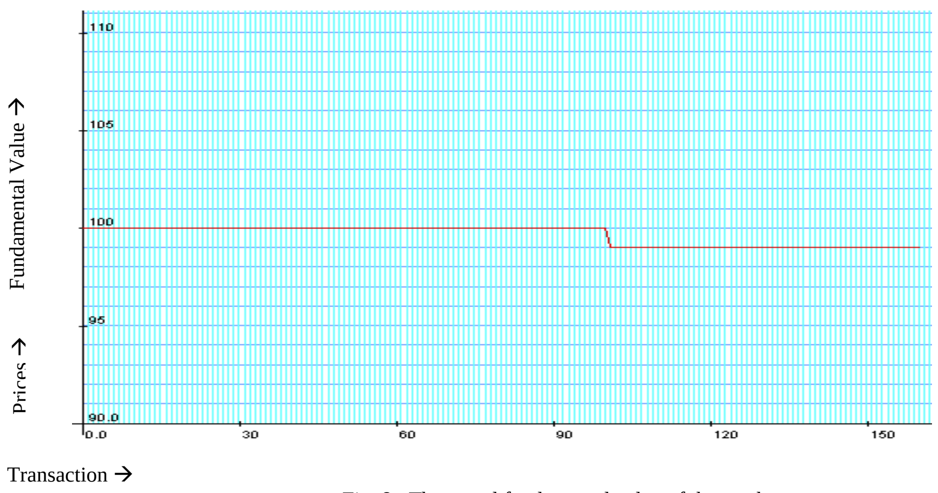


Fig. 8: The actual fundamental value of the stock

Appendix

EXPERIMENTAL SET UP : STEPS TO LAUNCH THE PROJECT IN JADE

1). To execute .java files, java compiler should be available to create the respective .class files. Firstly, download JDK (latest version) and set

- Path
  - Right click on My computer→Properties→Advanced Properties→Environment Variables→ Path→(augment) C:\Program Files\Java\jdk1.6.0\bin\;
  - create a new system variable named, CLASSPATH and set it to: .;C:\Program Files\Java\jre6\lib

Open CMD. Type,

- java
- javac

Note: All the commands available in java and javac will be printed if the path and classpath were set correctly.

- Compile the required java files in the jade folder: (this creates the .class files)

```
C:\ jade>javac -classpath lib\jade.jar –Xlint:unchecked -d classes examples\bookTrading\*.java
```

Note: Without -Xlint:unchecked , javac will not compile because the .java files are interlinked (GUI and java program).

2). Basic launch commands:

- To launch JADE GUI: java jade.Boot –gui (by default JADE uses port# 1099)
- To launch a main-container with port(1111),platform-id (MarketPlace):

```
java –cp lib\ jade.jar;classes jade.Boot -gui -platform-id MarketPlace -port 1111
```

- To launch two platforms on the same machine, in the first CMD prompt, enter:

```
java jade.Boot –gui (by default JADE uses port# 1099 )
```

- And in the second one, enter:

```
java jade.Boot –gui –port 1111
```

- To invoke an agent for, say, a predefined agent class: PingAgent :

```
java -cp lib\jade.jar;classes jade.Boot –gui  –agents ping1:examples.PingAgent.PingAgent
```

- To invoke multiple agents in the environment:

```
java -cp lib\jade.jar;classes jade.Boot –gui  –agents
ping1:examples.PingAgent.PingAgent;ping2:examples.PingAgent.PingAgent;ping3:examples.PingAgent.PingA
gent
```

3). To invoke this project:

- Type the following command in cmd.in C:\jade>

```
java -cp lib\jade.jar;classes jade.Boot -gui -platform-id MarketPlace -port 1111 -agents
```

nia1:examples.stockMarket.NoisyInAgent;nia2:examples.stockMarket.NoisyInAgent;nia3:examples.stockMarket.NoisyInAgent;nia4:examples.stockMarket.NoisyInAgent;nia5:examples.stockMarket.NoisyInAgent;in1:examples.stockMarket.InformedAgent;in2:examples.stockMarket.InformedAgent;in3:examples.stockMarket.InformedAgent;no1:examples.stockMarket.NoisyAgent;no2:examples.stockMarket.NoisyAgent;pg:examples.stockMarket.PingerAgent

Note: Invoke the MarketMakerAgent using the GUI. A prior invocation of the MarketMakerAgent, before the establishment of the market, will create erroneous results. Fig. A.1 shows 10 agents (as per the above command).

- To invoke MarketMakerAgent: Right click 'Main-Container' and select 'Start New Agent'(Fig. A.2).

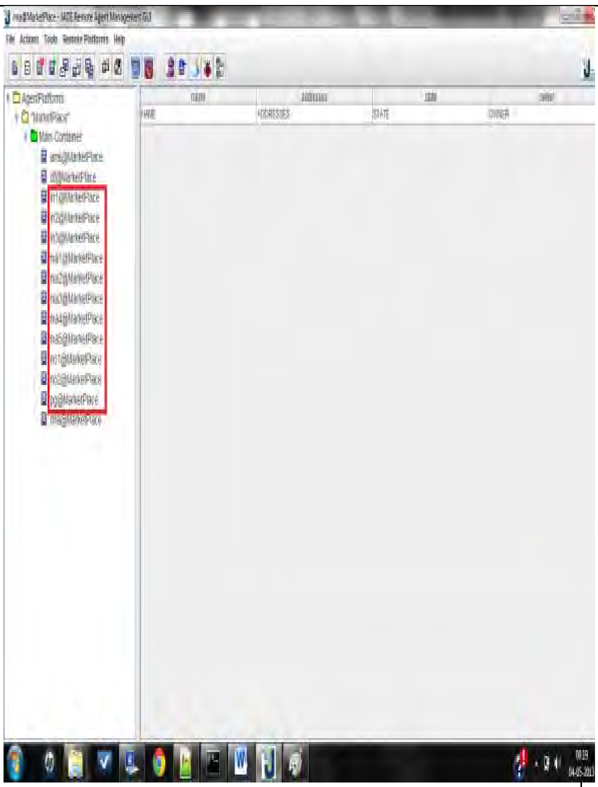


Fig A.1: Agents invoked in JADE environment

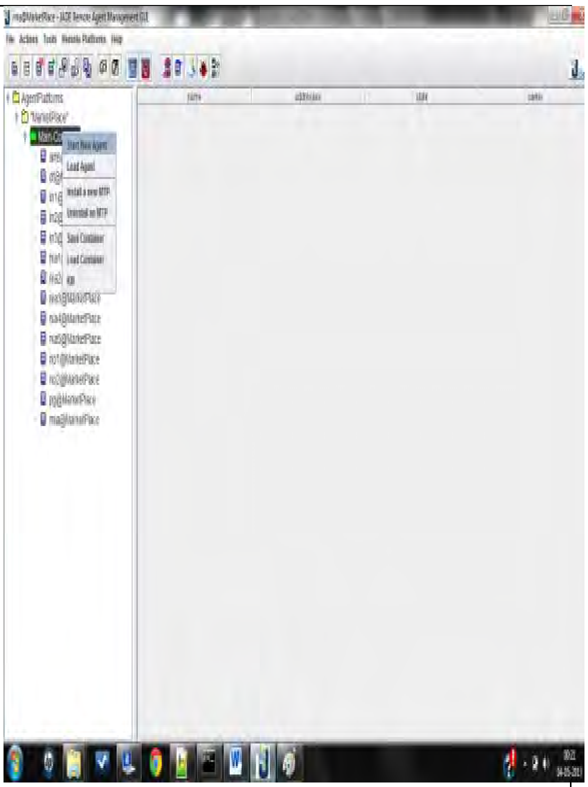
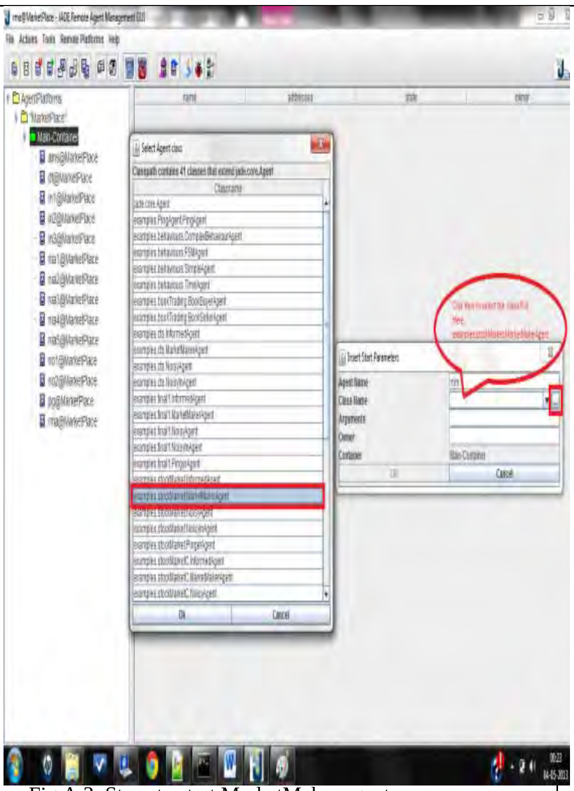


Fig A.2 Market Maker Agent

- Give the MarketMakerAgent a name and choose the .class file (Fig.A.3).
- Click on OK. The MarketMakerAgent is invoked (Fig. A.4).



Classpath contains 41 classes that extend jade.core.Agent

jade.core.Agent
examples.PingAgent.PingAgent
examples.behaviours.ComplexBehaviourAgent
examples.behaviours.FSMAgent
examples.behaviours.SimpleAgent
examples.behaviours.TimeAgent
examples.bookTrading.BookBuyerAgent
examples.bookTrading.BookSellerAgent
examples.ds.InformedAgent
examples.ds.MarketMakerAgent
examples.ds.NoisyAgent
examples.ds.NoisyInAgent
examples.final1.InformedAgent
examples.final1.MarketMakerAgent
examples.final1.NoisyAgent
examples.final1.NoisyInAgent
examples.final1.PingerAgent
examples.stockMarket.InformedAgent
examples.stockMarket.MarketMakerAgent
examples.stockMarket.NoisyAgent
examples.stockMarket.NoisyInAgent
examples.stockMarket.PingerAgent
examples.stockMarketC.InformedAgent
examples.stockMarketC.MarketMakerAgent
examples.stockMarketC.NoisyAgent

Fig A.3: Steps to start MarketMaker agent

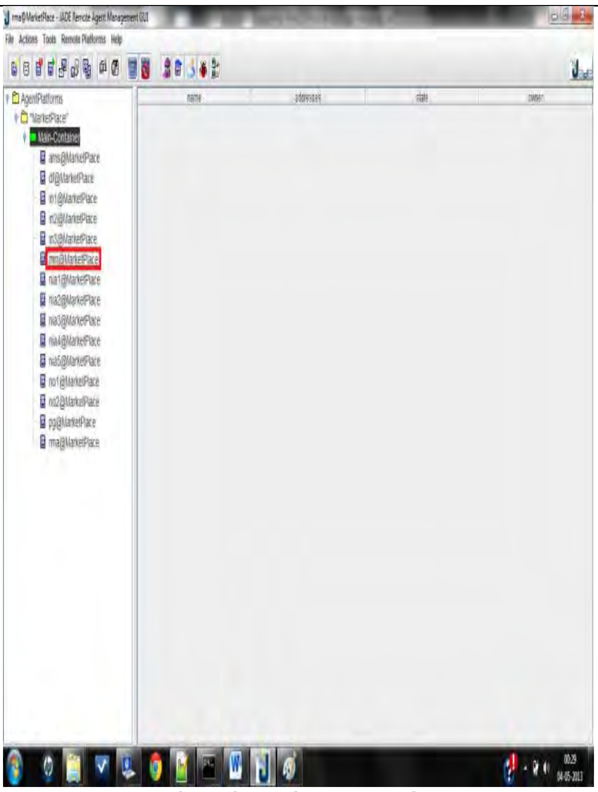


Fig A.4: MarketMaker in the agents pool

REFERENCES

[1] LeBaron B., “Empirical regularities from interacting long- and short-memory investors in an agent based stock market”, IEEE Transactions on Evolutionary Computation, 5(5):442–455, 2001.
[2] LeBaron B., Arthur W.B., and Palmer R., “Time series properties of an artificial stock market”, Journal of Economic Dynamics & Control, 23:1487–1516, 1999.
[3] LeBaron B., “A builder’s guide to agent based financial markets”, Quantitative Finance, 1(2), 254–261, 2000..
[4] Levy M., Levy H, and Solomon S., “Microscopic Simulation of Financial Markets: From Investor Behaviour to Market Phenomena”, Academic Press, 2000.
[5] Tesfatsion L., “Agent-based computational economics: Growing economies from the bottom up”, Artificial Life, 8:55–82, 2002.
[6] LeBaron,B, “Agent-based computational finance: Suggested readings and early research”, Journal of Economic Dynamics and Control
24, 679–702,2000.
[7] Jon Palin., “Agent-based stock market models: calibration issues and applications”, Submitted for the degree of M.Sc. in Evolutionary and Adaptive Systems, September 2002.
[8] M. Levy, H. Levy, and S. Solomon, “A microscopic model of the stock market”, Economics Letters, 45:103–111, 1994.
[9] T. Lux and M. Marchesi, “Scaling and criticality in a stochastic multi-agent model of a financial market”, Nature, 397:498–500, February 1999.
[10] Chan, N.T. and Shelton, “An electronic market maker”, Working Paper AI Memo 2001–005, Massachusetts Institute of Technology, 2001.
[11] B. LeBaron, “ Calibrating an agent-based financial market to macroeconomic time series”, Working Paper, Brandeis University 2002.
[12] B. LeBaron, “Short-memory traders and their impact on group learning in financial markets”, Proceedings of the National Academy of Science, 99:7201–7206, 2002.
[13] J. D. Farmer and S. Joshi, “The price dynamics of common trading strategies”, SFI Working Paper 00-12-069, 2000.
[14] G. Genotte and H. Leland, “Market liquidity, hedging, and crashes”, The American Economic Review,80(5):999–1021, December 1990.
[15] N. F. Johnson, D. Lampert, P. Jeffries, M. L. Hart, and S. Howison, “Application of multi-agent games to the prediction of financial time-series”, Physica A, 299:222–227, 2001.
[16] D. Lamper, S. Howison, and N. F. Johnson, “Predictability of large future changes in a competitive evolving population”, http://xxx.lanl.gov/cond-mat/0105258, May2001.
[17] Das S., “An agent-based model of dealership markets”, in Proceedings of the International Workshop on Complex Agent-based Dynamic Networks, Oxford, 2003.
[18] Das S., “A learning market-maker in the Glosten-Milgrom model”, Quantitative Finance 5(2), 169–180, 2005.
[19] Bellifemine F., Caire G., Tucco T., Rimassa G., “JADE Programmer’s Guide”, 2010.
[20] Harris L., “Trading and Exchanges: Market Microstructure for Practitioners”, Oxford University Press, 2003.
[21] Glosten L.R. and Milgrom P.R., “Bid, ask and transaction prices in a specialist market with heterogeneously informed traders”, Journal of Financial Economics 14, 71–100, 1985.
[22] Boer K., “Agent-Based Simulation of Financial Markets- A Modular, Continuous-Time Approach”, PhD Thesis, RSM Erasmus University / Erasmus School of Economics, 2008.