Agent-Based Simulation of Grid Resource Management Using Auction Market

Plamenka Borovska[1], Ognian Nakov[2], and Adelina Aleksieva-Petrova[3]

***Abstract* – Agent-based simulation is a new approach to modelling systems comprised of autonomous, interacting agents. The primary goal of this article is to describe the construction of an agent-based system that simulates auctions for grid resource management. This system can be used to evaluate the efficiency of auctions and as a tool to dynamically manage and distribute resources throughout grid systems.**

***Keywords* – agent-base simulation, Grid, resource management, auction.**

I. INTRODUCTION

Agent-based simulation refers to a model in which the dynamic processes of agent interaction are simulated repeatedly over time, as in systems dynamics, and time-stepped, discrete-event, and other types of conventional simulation [1].

Markets have emerged as a new paradigm for managing and allocating resources in complex systems. Several research systems have explored the use of different economic models for trading resources to manage resources in different application domains: CPU cycles, storage space, database, query processing, and distributed computing [2]. Such systems are Popcorn, Java Market, Mungi, Enhanced MOSIX, Nimrod-G and so on.

The primary goal of this article is to describe the construction of an agent-based system that simulates auctions for grid resource management. This system can be used to evaluate the efficiency of auctions and as a tool to dynamically manage and distribute resources throughout grid systems.

II. CASE STUDY

In this study, we simulate the auction mechanism of resource management.

In auctions the sellers (suppliers) submit their available resources to an auctioneer for processor time and another auctioneer for disk space at minimum prices; i.e. with the average profit per slot and per file formed on the basis of the value of the function of demand.

Buyers submit their requests by means of tasks (jobs), every task specifying the type, number and duration of resources needed by it.

Every buyer has a defined available budget ($G) with which to pay for the resources needed for its tasks, which is current for a fixed period of time.

Buyers submit requests for resources as long as they have tasks and a current budget.

The auctioneer maintains a list of available resources, running through the resources in each round in the order in which they are listed, implementing a second-price auction (in a concrete arrangement this can generally be Danish, English, etc.).

When a given resource is turned over for use by a consumer it is removed from the list of available resources and the consumer's budget decreases by a corresponding amount ($G).

After receiving resources for the current task, the buyer submits requests for its next task.

After every auction there are buyers who have already rented some resources (and for which they have already paid) who await additional resources to be provided to them from the next auctions in order to complete the current task..

In order to outline the architecture of this agent-based architecture three primary roles have been defined for agents:

- PRODUCER (seller, producer of resources)
- CONSUMER (buyer, application)
- AUCTIONEER (auctioneer, broker)

There are two types of producers: suppliers of processor time and suppliers of disk space.

Buyers (consumers) submit their requests through tasks. For each task the type, duration and number of utilized resources is described. Every buyer has a discrete budget available for paying for utilized services, which is current for a particular period of time and is subject to updates.

The auctioneer maintains a list of all the participants in the auction: CONSUMERS and PRODUCERS. Likewise, the auctioneer maintains a list of all the resources provided by the suppliers and a list of all the requests made by the buyers.

At every round the producers submit their available resources to the appropriate auctioneer at the price determined by the function of supply mentioned above.

At every round buyers submit their bids (requests), specifying the number, type and duration to use the resources, as well as the price they would pay for such resources according to their means.

It is important for us to note that at every round buyers can bid at most for one task.

Every round is conducted on the principle of the second-price auction in order to determine the winner. With this type of auction the buyer who wins is the one who has submitted

[1]Plamenka Borovska is with the Faculty of Computer Systems and Control, TU-Sofia, bl. Kliment Ohridski 8, 1000 Sofia, Bulgaria, E-mail: pborovska@tu-sofia.bg

[2]Ognian Nakov is with the Faculty of Computer Systems and Control, TU-Sofia, bl. Kliment Ohridski 8, 1000 Sofia, Bulgaria, E-mail: nakov@tu-sofia.bg

[3]Adelina Aleksieva-Petrova is with the Faculty of Computer Systems and Control, TU-Sofia, bl. Kliment Ohridski 8, 1000 Sofia, Bulgaria, E-mail: aaleksieva@tu-sofia.bg

the highest offer, thereby receiving the resource, and if there is a buyer who has submitted a second-highest offer that is above the minimum price set by the supplier, the final price for the resource is set at this second-highest bid.

III. PROGRAM IMPLEMENTATION

The Java software program JADE (Java Agent Development Framework) [3] is used in creating the forthcoming application. It facilitates the introduction of a multi-agent system in an environment that abides by FIPA specifications [4] and supports the conscription of graphical instruments to maintain and eliminate errors. The agent platform may be dispersed across computers and the configuration may be remotely controlled through a GUI (Graphical User Interface).

Three primary packages and two supplementary packages

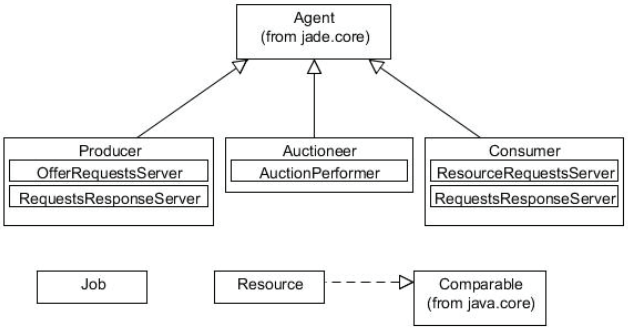

Fig. 1. General scheme of main packages

are utilized:

- Producer– with two subcategories: OfferRequestsServer and RequestsResponseServer, which records the behavior of the seller agent.
- Consumer – with two subcategories: ResourceRequestsServer and RequestsResponseServer, which records the behavior of the buyer agent.
- Auctioneer – with one subcategory: Auctionperformer, which records the behavior of the auctioneer (the resources broker).
- Job – a supplementary category that is used by the Consumer in order to generate tasks.
- Resource– a supplementary category that records each given resource (type, price and owner).

All three primary categories – Producer, Consumer and Auctioneer – are derived from the Agent category, which is a built-in JADE category.

In the setup() portion all of the characteristics of the agent are initialized and the different behaviors that the agent will express are assigned. These behaviors are described via internal categories derived from the primary category Behavior.

All of the categories describing different behaviors are part of jade.core.behaviours.*.

In the system categories the following behaviors are utilized:

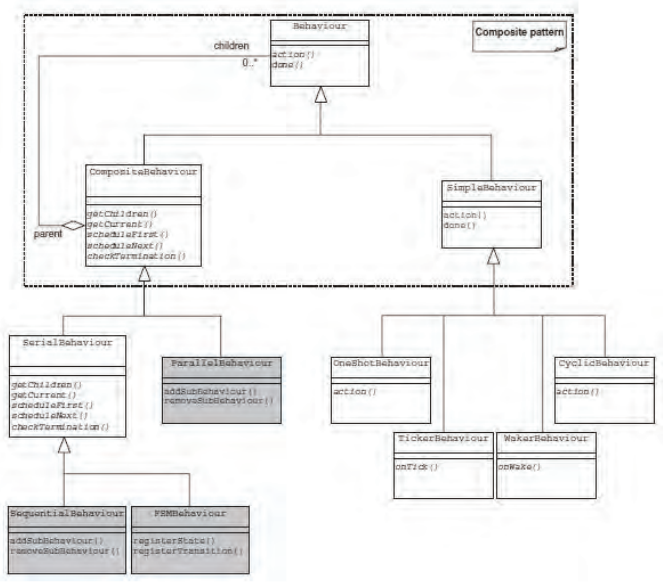

Fig. 2. The hierarchy of behavior packages in JADE

A. Producer

We have two subcategories describing two different behaviors:

OfferRequestsServer class – derived from CyclicBehaviour, which describes a periodically repeating behavior.

Each agent maintains its own message trail. The primary parts of the message are the body (contents), sender, recipient(s), the type (according to the standards of FIPA), the topic of the "conversation", etc.

In our case, the agent, in keeping with its assigned behavior, waits for a CFP (Call For Proposals) message, which serves as an "invitational" message. As long as there are no such messages on hand in the message trail the agent is blocked from using processor resources.

When such a message arrives the agent is activated and the corresponding actions are carried out – in this case it estimates the minimum resource price and sends all of its available resources at this price.

RequestsResponseServer class – derived from CyclicBehaviour. In this case this class is used for periodically receiving the auction results (whether a given agent has sold a resource and received a profit).

The agent awaits an ACCEPT_PROPOSAL message, which means "the offer has been accepted". As long as there are no such messages on hand in the message trail the agent is blocked from using processor resources.

When such a message arrives the agent is activated and the corresponding actions are carried out – in this case if a resource has been sold the profit is increased and the corresponding resources are marked as being in use for the specified duration for which has been paid.

B. Consumer

We have two subcategories describing two different behaviors:

ResourceRequestsServer category – derived from CyclicBehaviour. This class is used to periodically send requests to the auctioneer for resources needed in order to complete a given task.

The agent, in keeping with its assigned behavior, awaits a CFP (Call For Proposals) message, which serves as an "invitational" message. As long as there are no such messages on hand in the message trail the agent is blocked from using processor resources.

When such a message arrives the agent is activated and the corresponding actions are carried out – in this case the price that the agent can afford to spend is calculated based on its average speed of spending and its current budget, then it sends a request containing the number, type, duration and price it is willing to pay for a given resource to the auctioneer.

RequestsResponseServer class – derived from CyclicBehaviour. In this case this category is used to periodically receive the results from the auction (whether a given agent has won a resource and at what price).

The agent waits an INFORM message, which means "the offer has been accepted". As long as there are no such messages on hand in the message trail the agent is blocked from using processor resources.

When such a message arrives the agent is activated and the corresponding actions are carried out – in this case if it has won a resource the agent's budget is reduced and the corresponding work is marked as completed (if all the necessary resources for it have been supplied).

C. Auctioneer

Here we have one primary behavior: AuctionPerform category – derived from Behavior

In this case we have "round-specific behavior", whereby in each round the auctioneer carries out specific actions, after which it proceeds.

Round 1: The auctioneer sends messages to all agents (sellers and buyers) announcing the start of the auction, then waiting for their offers;

Round 2: After the auctioneer has announced the start of the auction it waits for all agents to send their replies – requests from buyers and offers from sellers. Only when the auctioneer has received responses from all agents does it proceed.

Round 3: After information from all participants is received then it is possible for the auction to be carried out. The action auction() is called upon, which carries out the second-price auction.

Round 4: After this auction is concluded and all resources have been distributed the corresponding messages are sent to all participants and the auctioneer moves on to the next auction.

The critical moment that needs to be mentioned is the registration of the agents into the so-called "yellow pages". Every agent is registered in them in order to easily be found in the future along with the type of services they offer as well as a brief additional description. Every agent has the ability both to register and to search for services.

In our case buyers and sellers register one service each.

This registration is subsequently used by the auctioneer in order to restrict the relevant agents according to their type.

IV. Simulation

The following parameters are given for the agents participating in the experimental formulation:

Producers:

Agent 1P → CPU Producer, 5 Slots, 6$/slot/time unit

Agent 2P → CPU Producer, 7 Slots, 8$/slot/time unit

Consumers:

Agent 1C → budget 100$, next budget refresh in 19 time units

Agent 1C Jobs:

Job1 → 4 CPU slots for 2 time units

Job2 → 3 CPU slots for 3 time units

Agent 2C → budget 150$, next budget refresh in 18 time units

Agent 2C Jobs:

Job1 → 2 CPU slots for 1 time unit

Job2 → 3 CPU slots for 1 time unit

The results from the simulation that was carried out are summarized in the following graph. Time is tracked along the X-axis, while the average price paid by all agents in each round is tracked along the Y-axis.

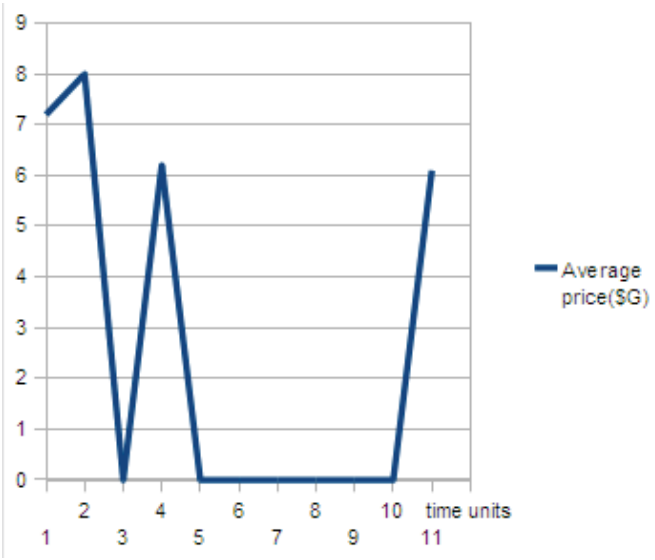

Fig. 3. Average price paid by all agent in simulation

V. Conclusion

Agent-based technologies are aimed at assisting in application development as they are based on a decentralized approach to intelligent agents. We have discussed the case study and implementation of the agent-based simulation of auction for resource management. The advantage of this

system is its agent-based architecture, which approximates to the greatest possible degree the actual organization of grid systems, along with its own requirements and problems.

REFERENCES

[1] Macal, C., Michael J. North: AGENT-BASED MODELING AND SIMULATION: ABMS EXAMPLES, Proceedings of the 2008 Winter Simulation Conference

[2] Li Chunlin, Li Layuan, The use of economic agents under price driven mechanism in grid resource management, Journal of Systems Architecture 50, 2004, pp.521–535

[3] JAva Platform DEvelopment Framework (JADE), http://jade.tilab.com/

[4] FIPA, URL: http://www.fipa.org/

[5] Qiang Tong, Zhidong Shen, Research of Mobile Agent Technology for GIS Grid, Ninth International Conference on Hybrid Intelligent Systems, 2009, 5p.

[6] Gang Wang, Tao Wen, Quan Guo, Xuebin Ma, A Knowledge Grid Architecture Based on Mobile Agent, Proceedings of the Second International, Conference on Semantics, Knowledge, and Grid, 2006, 4p.