ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 7 N. 4 (2018), 65-72
eISSN: 2255-2863
DOI: http://dx.doi.org/10.14201/ADCAIJ2018746572

Self-Organization through a multi-agent system for orders distribution in large companies

Alberto Montero, Sergio Rodríguez, Felipe Sánchez and Alberto Yébenes

id00714676@usal.es, felipesanchezcalzada@usal.es, id0718167@usal.es, alberyr@usal.es
University of Salamanca

KEYWORD

Multi-agent; Efficiency; Self-organized vehicles; Communication; Self-organization

ABSTRACT

This article presents the development of a multi-agent system in charge of self-managing a delivery system. The article focuses on the delivery management system and not on the movement systems of the different used vehicles.

This system consists of different types of vehicles, each with different characteristics, and there may be several instances of each type of vehicle. There will be three operating agents (Drone Operator, Car Operator and Amphibious Operator), an agent that will be responsible for creating random tasks (used only in simulations) and another one that is responsible for distributing these tasks to the operators taking into account the algorithm. This algorithm follows the bases of backtracking and its main function is to assign a task to a vehicle taking into account the distance, the consumption, the limitations of weight and distances, etc. The whole system has been developed in JADE on java. The described software performs a complete simulation with a console in which it is indicated relevant information such as the tasks that are created, the type of vehicle and the instance of that type of vehicle that resolve the delivery, among others. The purpose of this system is to minimize costs and times.

1. Introduction

Focusing on how the distribution of packages is currently established, we propose a solution that will allow an improvement so that everything can be done automatically through a multi-agent system. Human intervention won't be necessary in the vehicles' handling, thus giving greater security and guarantee to the distribution in large companies, allowing to save great costs, both economical and temporary.

An agent is an encapsulated computer system, located in some environment, within which acts in an autonomous and flexible way to meet its objectives (Wooldridge y Jennings, 1995). A multi-agent system is one which is designed and implemented thinking that it will be composed of several agents who will interact with each other, so that they achieve together the desired functionality (Busman *et al.,* 1993). In this case we must make a greater effort of abstraction, identify mechanisms of learning, coordination, negotiation, etc.

The basis of our project has been the scientific article (Briones *et al.,* 2018). We believe that our work improves the previous one since it adapts better to the real world and the problems related with orders distribution. Our objective is that the vehicles controlled by agents be able to face environmental adversities in an efficient and autonomous way. We have developed this project on JADE with an architecture based on agents, through the intervention of different agents which manage these vehicles and the delivery of the packages assigned to tasks controlled by other agents. Each action will focus on the self-organization of these agents to achieve the most optimal solution in each distribution. After presenting a revision of the state of the art, the MAS architecture is presented and evaluated. Finally conclusions are outlined.

2. State of the arts

This paper presents a MAS, developed for the management of the distribution of materials, packages, orders… We believe that it can suppose a great progress, since it will allow companies to be more efficient. It will mean an improvement in shipping time, as well as various economic benefits (gasoline, increased revenue..). The software made will allow you to choose the most suitable vehicle for each shipment taking into account the route to be made, the distance, the available vehicles... Another aspect to keep in mind is that we have used efficient algorithms to fulfil our purpose. It should also be noted that it is a scalable and generic system so it can be applied in many areas. Although we relied on the article Briones (Briones *et al.*, 2018), which propose a multi-agent system with a drone agent who manage car agents to take a package of a maze and put them outside, we thought that in the real world it wouldn't be necessary or functional, because of that we proposed a multi-agent system with agents that can cross rivers or fly to delivery packages. However, in our project are need more resources, which involve a bigger cost.

In multi-agent systems, due to their complexity, a series of requirements must be followed to ensure that those systems achieve the established objectives. Because an agent could not perform only the tasks that are proposed, given their difficulty, they must be divided into simple sub-tasks which are assigned to an agent (Turner *et al.*, 2013), (Demopoulos *et al.*, 2006), (Roman *et al.*, 2016). In this paper, this division of tasks has been implemented, which has allowed to save computing costs and time.

The proposed approach in this paper is to work with a centralized system (Peng *et al.*, 2013), unlike (Muñoz *et al.*, 2005), which implies having a leader who coordinates and assigns the tasks to the agents. In our case, the organizer agent is the one that works as a leader, which assigns the tasks to the different vehicle agents through messages sending. We believe that using a leader agent in multi-agent systems allows having a better control of this, since it avoids possible conflicts that may arise in the tasks assignment.

3. Proposal

A Multi Agent bases System has been constructed using JAVA.

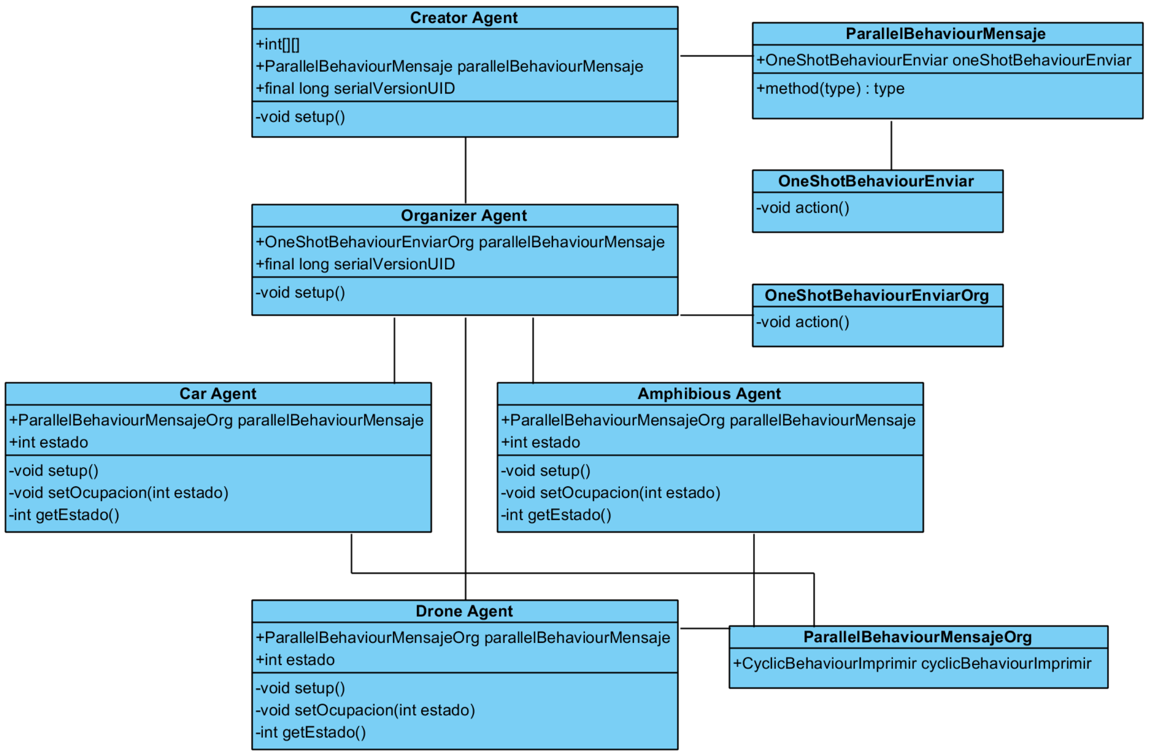

Creator Agent
+int[][]
+ParallelBehaviourMensaje parallelBehaviourMensaje
+final long serialVersionUID
-void setup()
ParallelBehaviourMensaje
+OneShotBehaviourEnviar oneShotBehaviourEnviar
+method(type) : type
OneShotBehaviourEnviar
-void action()
Organizer Agent
+OneShotBehaviourEnviarOrg parallelBehaviourMensaje
+final long serialVersionUID
-void setup()
OneShotBehaviourEnviarOrg
-void action()
Car Agent
+ParallelBehaviourMensajeOrg parallelBehaviourMensaje
+int estado
-void setup()
-void setOcupacion(int estado)
-int getEstado()
Amphibious Agent
+ParallelBehaviourMensajeOrg parallelBehaviourMensaje
+int estado
-void setup()
-void setOcupacion(int estado)
-int getEstado()
Drone Agent
+ParallelBehaviourMensajeOrg parallelBehaviourMensaje
+int estado
-void setup()
-void setOcupacion(int estado)
-int getEstado()
ParallelBehaviourMensajeOrg
+CyclicBehaviourImprimir cyclicBehaviourImprimir

Figure 1: Class Diagram: It is showed the classes relations with the agents and most important methods and attributes.

The Model includes a fleet of 3 vehicles (drone, car, amphibious) with different characteristics which are the ones used to decide the most suitable vehicle:

1. Drone
    a. Carries a weight of 1kg
    b. Travel a distance of 3km
    c. Consume 1 unit per km
    d. Cannot face strong winds
2. Car
    a. Carries a weight of 3kg
    b. Travel a distance of 5km
    c. Consume 3 units per km
    d. Cannot cross rivers
3. Amphibian
    a. Carries a weight of 7kg
    b. Walk an unlimited distance
    c. Consumes 10 units per km
    d. Can go through all kinds of conditions

The environment used consists of a matrix where there will be different conditions (strong winds, rivers or ground) in each square of the matrix. Each matrix number is a condition; 1 means ground, 2 means river, 3 means wind and 5 means river and wind.

1	1	1	1
2	5	3	1
2	2	2	1
2	1	1	1

Figure 2: The matrix who represents the map. Each number represents a type of terrain.

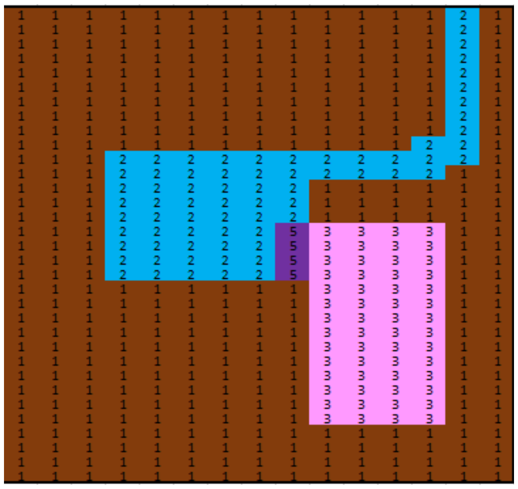


Figure 3: A matrix which could represents a real case: blue squares mean water, brown mean ground, rose mean wind and purple wind and water.

For the resolution of the problem there are 5 types of agents:

1. Drone Operator: is in charge of controlling the drone vehicle.
2. Car Operator: is responsible for controlling the car vehicle.
3. Amphibious operator: is responsible for controlling the amphibious vehicle.
4. Creator: creates tasks automatically, giving a route and a weight of the package for each distribution.
5. Organizer: organizes the different tasks to assign them to the most optimal vehicle.

The execution is based on the following steps:

1. The 'creator' agent creates a task, which consist in give 2 matrix coordinates, 1 of the actual position (x_1 and y_1) and 1 of the destination (x_2 and y_2). For the message sending will be used the method ParallelBehaviourMensaje which sends through OneShotBehaviourEnviar. This agent is simple reflexive type thus it generates and sends tasks.
2. The 'creator' agent sends a message about the task to be accomplished to the 'organizer' agent.
3. The 'organizer' agent, through the matrix in which details the different environmental obstacles, calculates the shortest path for each 'vehicle' agent (in the case they can perform it) through a backtracking algorithm and then it selects the shortest of the 3.
4. The 'organizer' agent send a message to the 'vehicle' agent, 'amphibious', 'car' or 'drone', to make the task. If the 'vehicle' agent is occupied, the task is sent to other agent. If every agent is occupied, the task will wait until one agent be free. It will apply OneShotBehaviouEnviarOrg indicating to the pertinent

vehicle the performance of a task. This agent is based on goals since it needs information to decide which vehicle choose.

5. The ‘vehicle’ agent accomplish the task using the method ParallelBehaviourMensajeOrg.

They are simple reflexive agents due to they use condition-action rules to establish the connection between the receives tasks and their implementation.

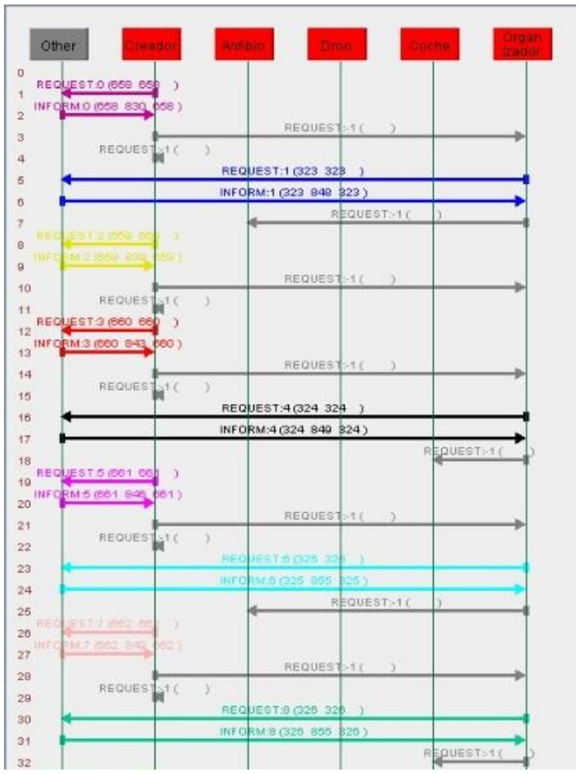


Figure 4: Communication between the agents. It can be seen the communication between creator and organizer agents. and then to vehicle agent.

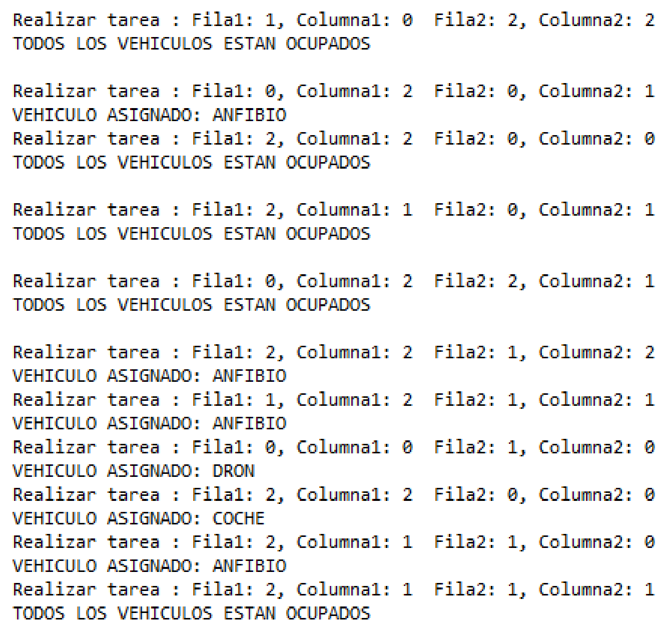
The messages sent by the agents to communicate are (according to our code in JADE):

Creator to organizer: an object of type *CaracteristicasPaquete* is sent, which has an associated starting point, end point and package weight.

Organizer to drone, car or amphibious: an object of type *Object* which is actually a list of points where the vehicle should go. This assignment is made in *setContentObject ((Serializable) objEnv)* where *objEnv* is the named list.

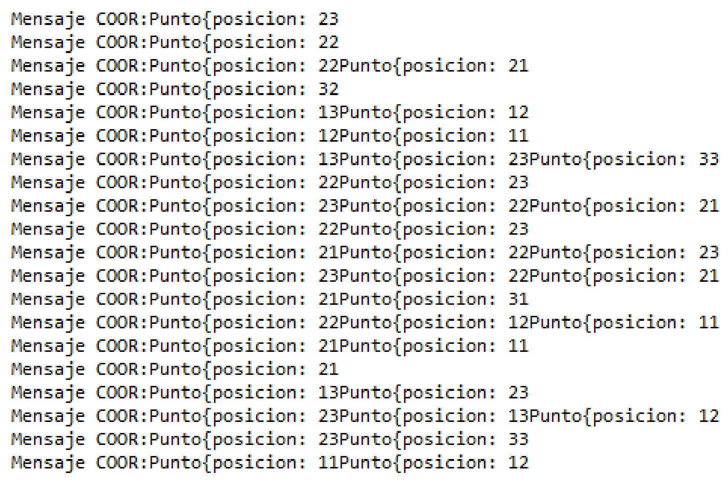
4. Results and Conclusions

The evaluation and validation of the model for this study will be done through the Java IDE *Eclipse,* developed by the Eclipse Foundation. So as to evaluate the proposed architecture, a series of experiments were conducted with the purpose of simulating the behavior of the different vehicles controlled by the agents in the performance of their tasks. The obtained results have made it possible to evaluate if the system was able to perform the tasks without any problem. Such distribution was carried out in the most efficient way possible, as well as the distribution of the packages. Given the environmental limitations of the drone and car agents, the amphibious agent was usually the most used, although if it was busy, the others performed the task without any problem.



```
Realizar tarea : Fila1: 1, Columna1: 0  Fila2: 2, Columna2: 2
TODOS LOS VEHICULOS ESTAN OCUPADOS

Realizar tarea : Fila1: 0, Columna1: 2  Fila2: 0, Columna2: 1
VEHICULO ASIGNADO: ANFIBIO
Realizar tarea : Fila1: 2, Columna1: 2  Fila2: 0, Columna2: 0
TODOS LOS VEHICULOS ESTAN OCUPADOS

Realizar tarea : Fila1: 2, Columna1: 1  Fila2: 0, Columna2: 1
TODOS LOS VEHICULOS ESTAN OCUPADOS

Realizar tarea : Fila1: 0, Columna1: 2  Fila2: 2, Columna2: 1
TODOS LOS VEHICULOS ESTAN OCUPADOS

Realizar tarea : Fila1: 2, Columna1: 2  Fila2: 1, Columna2: 2
VEHICULO ASIGNADO: ANFIBIO
Realizar tarea : Fila1: 1, Columna1: 2  Fila2: 1, Columna2: 1
VEHICULO ASIGNADO: ANFIBIO
Realizar tarea : Fila1: 0, Columna1: 0  Fila2: 1, Columna2: 0
VEHICULO ASIGNADO: DRON
Realizar tarea : Fila1: 2, Columna1: 2  Fila2: 0, Columna2: 0
VEHICULO ASIGNADO: COCHE
Realizar tarea : Fila1: 2, Columna1: 1  Fila2: 1, Columna2: 0
VEHICULO ASIGNADO: ANFIBIO
Realizar tarea : Fila1: 2, Columna1: 1  Fila2: 1, Columna2: 1
TODOS LOS VEHICULOS ESTAN OCUPADOS
```

Figure 5: Organizer agent. It shows which vehicle performs the task given by the organizer agent or if they are all occupied.



```
Mensaje COOR:Punto{posicion: 23
Mensaje COOR:Punto{posicion: 22
Mensaje COOR:Punto{posicion: 22Punto{posicion: 21
Mensaje COOR:Punto{posicion: 32
Mensaje COOR:Punto{posicion: 13Punto{posicion: 12
Mensaje COOR:Punto{posicion: 12Punto{posicion: 11
Mensaje COOR:Punto{posicion: 13Punto{posicion: 23Punto{posicion: 33
Mensaje COOR:Punto{posicion: 22Punto{posicion: 23
Mensaje COOR:Punto{posicion: 23Punto{posicion: 22Punto{posicion: 21
Mensaje COOR:Punto{posicion: 22Punto{posicion: 23
Mensaje COOR:Punto{posicion: 21Punto{posicion: 22Punto{posicion: 23
Mensaje COOR:Punto{posicion: 23Punto{posicion: 22Punto{posicion: 21
Mensaje COOR:Punto{posicion: 21Punto{posicion: 31
Mensaje COOR:Punto{posicion: 22Punto{posicion: 12Punto{posicion: 11
Mensaje COOR:Punto{posicion: 21Punto{posicion: 11
Mensaje COOR:Punto{posicion: 21
Mensaje COOR:Punto{posicion: 13Punto{posicion: 23
Mensaje COOR:Punto{posicion: 23Punto{posicion: 13Punto{posicion: 12
Mensaje COOR:Punto{posicion: 23Punto{posicion: 33
Mensaje COOR:Punto{posicion: 11Punto{posicion: 12
```

Figure 6: Amphibious agent. It is showed the point sequence that it should follow to perform the task.

Our proposal is especially focused on the field of automotive organization for large companies or organizations that need this service in order to be more efficient, which can be positively affected as the algorithm used to assign the vehicles stops being based on ‘backtracking’ and adapts to the needs of the company that uses it, either to send packages (such as Amazon) or to organize vehicles in other areas (such as hospitals, police, firemen ..). Apart from the improvement of the algorithm in reduction of time, several benefits are experienced such as the reduction of costs making the company improve their income, since we also focus on the consumption of vehicles in certain tasks based on the needs of the company.

5. References

Bechlioulis, C., Rovithakis, G. A., 2016. Decentralized Robust Synchronization of Unknown High Order Non-linear Multi-Agent Systems with Prescribed Transient and Steady State Performance. Pages 123-134. IEEE

Briones, C. D., Machaen, Z. A., Martín, L. M., Torres, G. A., 2018. Self-organizing mobile robots based on multi-agent coordination techniques implemented with aerial vision and communication gateway between Wi-Fi and RF.

Castelfranchis, C., 1997. Modeling social interaction for Al agents. Pages 1567-1576.

Castelfranchis, C., Dignum, F., Jonker, C., Treur, J., 1999. Deliberative normative agents: Principles and architecture. Pages 364-378. Springer, Berlin, Heidelberg.

Chamoso, P., De la Prieta, F., Bajo, J., Corchado, J. M., 2019. Conflict resolution with agents in smart cities. Pages 695-713. IGI Global.

Corchado, J. M., Bajo, J., De Paz, Y., Tapia, D., 2008. Intelligent environment for monitoring Alzheimer patients, agent technology for health care. Pages 382-396. North-Holland.

Corchado, J. M., Lasa, R., 2003. Constructing deliberative agents with case-based reasoning technolgy. Pages 1227-1241. Wiley Subscription Services, Inc., A Wiley Company.

Corchado, J.M., Lees, B., 2001. A hybrid case-based model for forecasting. Pages 106-127- Applies Artificial Intelligence.

Dimopoulos, Y., Moraitis, P., 2006. Multi-agent coordination and cooperation through classical planning. Pages 398–402. IEEE/WIC/ACM.

Fyfe, C., Corchado, E., González, M., Corchado, J.M., 2002. Analytical Model for Constructing Deliberative Agents. CRL Publishing.

Guo, G., Ding, L., Han, Q., 2014. A distributed event-triggered transmission strategy for sampled-data consensus of multi-agent systems. Pages 1489-1496. Pergamon.

Jennings, N.R., Wooldridge, M., 1998. A roadmap of agent research and development. Pages 7-38. Springer Netherlands.

Jennings, N.R., 2000. On agent-based software engineering. Pages 117. Elseiver.

Jennings, N.R., Wooldridge, M., Kinny, D, 2000. The Gaia methodology for agent-oriented analysis and design. Pages 284-312. Springer Netherlands.

Jennings, N.R., Wooldridge, M., 1994. Agent theories, architectures and languages: a survey. Pages 1-39. Springer, Berlin, Heidelberg.

Muñoz, R., Aguirre, R., García-Silvente, M., Gomez, M., 2005. A multi-agent system architecture for mobile robot navigation based on fuzzy and visual behavior. Pages 689–699.

Pelrine, R., Hsu, A., Cowan, C., Wong-Foy, A., 2017. Multi-agent systems using diamagnetic micro manipulation - From floating swarms to mobile sensors. IEEE.

Peng, Z., Wen, G., Rahmani, A., 2013. Leader-follower formation control of multiple nonholonomic robots based on backstepping. Pages 211–216. ACM.

Román, J. A., Rodríguez, S., 2016. Improvement in the distribution of services in multi-agent systems with SCODA. Pages 31-46, v. 4, n. 3. ADCAIJ

Shouwenaats, T., De Moor, B., Feron, E., How, J., 2001. Mized integer programming for multi-vehicle path planning. Pages 2603-2608. IEEE.

Turner, J. R., 2013. Multiagent systems as a team member. Pages 73–90. International Journal of Technology, Knowledge & Society, vol. 9, no. 1.

Wooldridge, M., 1997. Agent-based software engineering. Pages 26-37. IET.

Wooldridge. M., 2009. An introduction to multiagent systems.