A MAS Approach for Port Container Terminal Management

Miguel Rebollo, Vicente Julian, Carlos Carrascosa and Vicente Botti

Departamento Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Valencia, Spain
{mrebollo, vinglada, carrasco, vbotti}@dsic.upv.es

Abstract. This paper presents a system architecture which is based on the multi-agent system paradigm for solving complex problems. This architecture is applied to solve the port container terminal management problem, and specifically to solve the automatic container allocation. The multi-agent systems paradigm seems to fit this problem due to its inherent complexity. This work is part of a project for the integral management of the container terminal of an actual port.

1 Introduction

Over the last few years, the use of the agent/multi-agent system paradigm has increased sharply as an important field of research within the Artificial Intelligence area. This paradigm has been applied to different fields, such as control processes [12], mobile robots [20], air-trafic management [19] and intelligent information retrieval [14].

A lot of agent definitions can be found in the literature but there is no one which has been fully accepted by the scientific community. A definition that is appropiate for our use is the one proposed by Wooldridge and Jennings [27]. According to them, an agent is defined by its *flexibility*, which implies that an agent is:

- Reactive: an agent must answer to its environment.
- Proactive: an agent has to be able to try to fulfill his own plans or objectives.
- Social: an agent has to be able to communicate with other agents by means of some kind of language.

As is stated by this idea of flexibility, to name a tool with the term *agent* this tool should be able to satisfy the above requirements. Nowadays, a small percentage of the existing software follows this definition.

An approach to a system architecture which is based on the multi-agent paradigm for solving the automatic allocation problem in a container terminal is presented in this paper. The operations carried out in this kind of terminal are included in the most complex tasks of the transport industry. This is due to:

- The great diversity of entities acting in the container import and export processes.

– Interaction with a dynamic environment.
– The distributed nature of the problem which is formed by a set of independent systems, but whose individual decisions directly affect the performance of the others.

Because of these factors, it is very difficult to analyse and to develop a single application with all the necessary functionalities. This is the reason for dealing with each task independently without losing sight of the close relationship among the tasks.

The multi-agent system model seems to be an adequate framework for dealing with the design and development of an application which is flexible, adaptable to the environment, versatile and robust enough for the efficient management of a container terminal.

It is very important for the turn-around time of a cargo ship which is in port container terminals to be as short as possible. An average cargo liner spends 60% of its time in port and has a cost on the order of U.S. $1000 for each hour it spends in port [23]. The whole container allocation process must be directed towards minimise the containership stowage time. This is the main objective of the optimisation of the global performance allocation process.

The rest of this paper is structured as follows. Section 2 presents a detailed problem description. Section 3 defines the multi-agent system architecture we propose. Finally, in section 4 the conclusions and future work are presented.

2 Problem description

Container terminal management is a very complex system, then it may be that the only way it can reasonably be addressed is to develop a number of modular components that are specialized for solving a particular aspect of it [13].

The set of operations to be conducted in the terminal is very extensive, but the alternative approaches share some common systems [11]:

– **Marine Side Interface**. This system focuses on loading/unloading containers to/from ships. Normally two or three gantry cranes (GC) are used to move containers for each ship.
– **Transfer System**. It transfers containers from/to the apron to/from the container storage yard. The method used in the terminal is to employ yard trucks (YT) to make the transports. Transtainers are used to pick up or to put down a container on the storage area of the yard (Figure 3).
– **Container Storage System**. Its purpose is to allocate and to control the containers in the yard.
– **Land Side Interface**. It focuses on handling the interactions with the land transportation modes.

2.1 Marine side interface

Load planning is usually carried out as follows. Before the berthing of a vessel, a shipping company provides the work instruction, called *load profile* (Figure 1), where the slot in which each container must be placed is indicated. A load profile shows several clusters of cells, each one of which is assigned to a group of containers with the same length and that have the same destination port. A colour code is used to recognise the different groups of containers. The process of formulation of a load profile is detailed in [25].

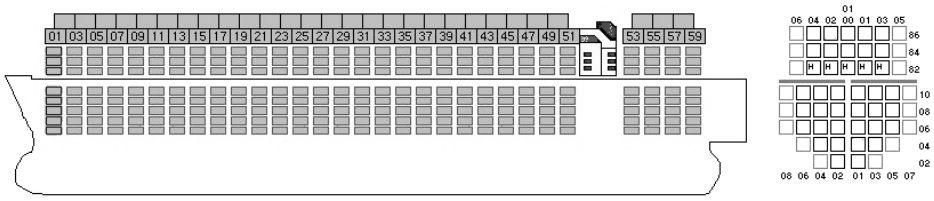

Fig. 1. Ship bay map.

The *work scheduling* takes into account the load profile and the availability of GCs. It sets the secuence used to load the bays of the cargo ship, solving the possible interference among the GCs.
There are two studies about the crane scheduling problem in [23] and [6]. Both suggests that algorithms for assigning GCs must be methods that minimise the aggregate delay cost. A strategy of providing buffer spaces in the apron area is provided in [5], to increase the use of the GCs and to reduce the total container loading time. Different strategies are evaluated through simulation.

2.2 Transfer system

After the work schedule is determined, the load sequencing process begins. The actual assignment of containers to specific locations in a ship bay is performed and the load sequence is determined. The sequencing problem is a very complex one, thus it is broken down into two hierarchical problems: the routing problem for transtainers and the pickup sequencing problem.

In the **routing problem** for transtainers, the visiting sequence of each transtainer and the numbers of containers to be picked up at each visited yard bay is determined. In the process, the planner considers the work schedule for the GCs and the yard map, which shows the storage locations of the containers in the yard (Figure 2).

In the **pickup sequencing problem**, the loading for individual containers is determined for the convenience of the handling activities of transtainers and GCs, the stability of the vessel, the maximum staking weight and so on. The final result of the container sequencing problem is called the *load sequence list.*

The existing approaches to this problem attempt to resolve it using operations research techniques. [2] proposes a load planning method in which the main consideration is how to meet the best requirements for ship trim and stability and safety regula-

tions governing hazardous cargo stowage. In [4], Cho develops a methodology for containership load planning. He formulates an integer programming model to assign an individual container to a cell in a bay of a ship. Due to the very large number of containers, the computational time may be an obstacle to applying this approach. In [9], Gifford describes a heuristic procedure for containership load planning in a transtainer-based container port.

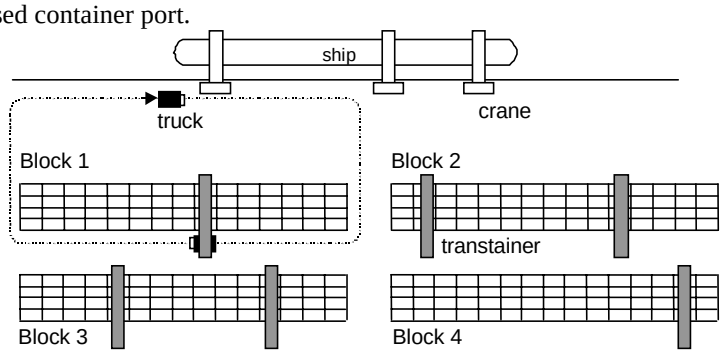

Fig. 2. Yard map.

2.3 Container storage system

The allocation of containers on the yard is a problem that directly affects the previous two systems. A bad container distribution forces the transtainer to make more movements and the GCs to be inactive more time, which increases the loading time.

The way to reduce the useless transtainer movements is to increase the stacking density. Then, all the containers are allocated in close areas and the time dedicated to the movements is reduced.

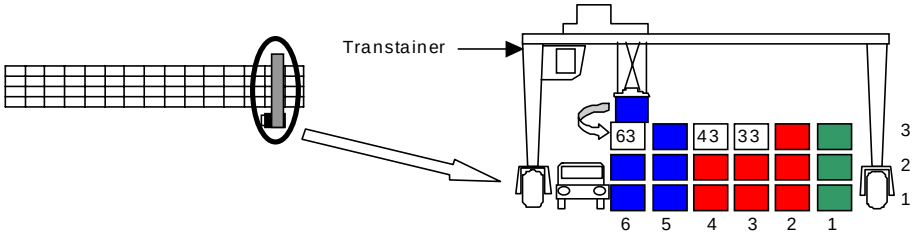

Fig. 3. Transtainer.

Typically, the applications for the management of container terminals divide the work into two tasks [3]: the yard configuration and the automatic container allocation.

The **yard configuration** problem deals with the assignment of the stacks to one shipping company making a specified route. This is called a *Service* in the container terminal. All the containers which have to be unloaded to one of the ports of this route must be allocated to the same area in order to improve the load/unload time. The containers are also organised by taking into account the vessel, onto which they

have to be loaded. Some additional conditions can be taken into account, such as the destination port or the total number of containers that will be unloaded to the same port.

The container allocation problem focuses on assigning a free allocation on the yard for the container to be stored until it is loaded into a ship (or, if it comes from a vessel, until a truck takes it away from the terminal). When a container arrives to the terminal, the system decides which allocation is the most appropiate depending on the assigned cargo ship, its destination port, its dimensions, its weight and, if it is available, the allocation into the bay of the ship.

2.4 Land side interface

The goal of this system is to control the access to the terminal of the trucks, which carry or take away the containers.

The information is introduced into the system using EDI messages. The shipbroker can send the container data through Internet to the terminal. When the container arrives, their data and the EDI message are compared to check their accuracy.
Another task is the control of the terminal access gates, identifying both the truck and the container using artificial vision techniques. In this way, an unattended gateway system, which speeds up the truck admission, is achieved and global productivity of the terminal is increased.

3 System architecture

To design the system architecture, the system has been divided according to its main tasks. Therefore, a different kind of agent for each one of the main tasks to be done has been modelled. These agents are mainly characterised by their independence from the rest of the system elements. They are able to coordinate and to communicate some decisions to the rest of the system. The communication between agents is done by means of asynchronous messages, which are based upon the FIPA-ACL standard [7], using radio frequency (RF).

This design, which is designed according to the agent paradigm, allows us to divide the problem into subproblems. Each subproblem can be solved by a specific agent. In this way, the design is simplified and a highly robust development is allowed. In this case, each agent can be considered as a autonomous reasoner [26].

Five agent classes can be found in this system, as can be observed in Figure 4:

- The Ship agents: they control the ships load and unload sequence scheduling process
- The Stevedore agents: they manage the loading and unloading of all the ships docking in the port.
- The Service agents: they distribute the containers in the port terminal.
- The Transtainer agents: they optimise the use of these machines.
- The Gate agents: they interact with the land transport (I/O of containers by land).

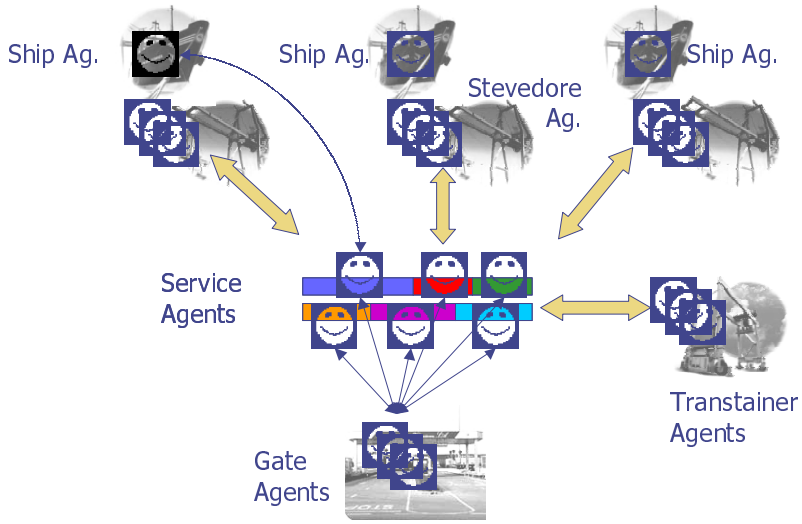

Fig. 4. System Architecture.

The main features of the agents presented in this architecture are specified in the following sections.

3.1 Agent Description

Ship Agent

An agent will be created for each ship docking in the port to dynamically manage its load/unload. Therefore, in response to the arrival of a ship (Ship agent creation event) the system will create a new Ship agent instance for this ship and its load profile. Its goals are: to minimize the gantry crane idle time, to maximize its utilization, to minimize the ships load/unload time, and to minimize the derived costs from the stowage process. This work is closely related to the Stevedore agents involved, which the Ship agent will have to co-ordinate with.

Each Ship agent faces a scheduling problem. In this problem, a set of resources (the cranes) must be assigned to the different operations (container load/unload) establishing a resource use time (container load/unload time). The resolution method for this problem will be considered as a constraint satisfaction problem (CSP) [1][8] [17][24].

This agent must have the following information: the number of assigned gantry cranes to the ship, the ship load profile, the ship characteristics, the number of containers to load/unload, the name of the port where the containers must be unloaded, the type, the length and the height of each container, and finally, the operation time of

the gantry cranes.

The different Ship agents which are active at any given moment must co-ordinate with each other as a whole to minimise the possible blockages between the assigned cranes. The goal of this minimisation is to maximise the active time of all the cranes and to reduce the load/unload time of each ship.

Stevedore Agent

Each one of the gantry cranes of the terminal (I/O of containers by sea) is controlled by a Stevedore agent. When a gantry crane is active loading or unloading containers from an specific ship, the Stevedore agent will try to obtain the most appropiate scheduling to manage the container stowage in the ships load/unload sequences. At this point, the agent will use informed search techniques like heuristic search algorithms (A*, IDA*, RBFS) [10] [15] [16], that allow for the reduction of the problem solution space. Moreover, the restrictions that limit the Stevedore agent due to the necessity of adjusting to the gantry crane load/unload scheduling , makes it possible to apply hard prunings and to achieve a fast convergence to the optimal solution of the search process. The agent will have to know the following information in order to perform its task: the gantry crane load/unload sequence, the yard trucks assigned to this crane and the positions of the different containers in the terminal (known by means of the service agents involved).

To develop these goals, the agent is co-ordinated with the rest of the active Ship agents and the suitable Service agents. Therefore, the agent will try, in a co-ordinated way, to minimise the empty movements of the employed machinery and the number of the necessary machines in the internal transfer.

Service Agent

The Terminal has been divided according to the different services. Each service has assigned some specific stacking ranges. The main goal of this kind of agent is to determine the appropriate allocation for the arriving containers in the Terminal from a specific service (allocation problem) and the suitable configuration of the portion of the yard the agent controls.

For the development of its allocation function, the agent must focus the search process towards an approximate solution, which can be progressively refined (by applying something similar to an IDA* algorithm) [22]. The agent tries to obtain short response times for the container allocation and satisfactory responses, which are near the optimal values. To do this, the agent must know the yard map of its assigned portion, the container characteristics (type, length, weight, destination port, ship, …) and the stacking factor. Also, the agent has to coordinate with the other service agents in order to resolve any conflicts; for instance, their assigned stacks are full and the container must be allocated to another place.

To solve the configuration problem, the goal of the service agents is to maximise the stacking density in its yard portion. To reach this goal, the agent uses non-

supervised learning techniques [18], which allow it to automatically adjust the yard distribution goodness through the system obtained results as its utilization time increases. The service agent launches this process automatically, when the agent considers it to be necessary (pro-activity), based on criteria such as, time, stack allocation conflicts —slots without use or cargo ships that run out of reserved areas—, low staking density, and so on.

Transtainer Agent

Each system transtainer is modelled as an autonomous agent whose goal is to efficiently perform the stacking operations of the containers in the yard.

The transtainer agent has to minimize its empty movements. To do this, it must obtain the most accurate sequence for the container movement to/from its correct position in the yard. Each one of these agents is waiting for stacking requests from the different service agents, who facilitate the transtainer agent with:

- The containers to be moved from the stack and where they are located: this is done for vessel or external truck loading.
- The containers to be moved to the stack and where they must be placed: this is done for vessel or external truck unloading.

Gate Agent

Each one of the terminal gates, that is, the containers input and output by land, is controlled by a *Gate* Agent. To do this, the agent will have to manage the terminal gate assigned, informing the corresponding service agent when necessary. It will have to inform the corresponding service agent of the new containers' arrival (to store them) and of the trucks' arrival (to retire containers from the yard).

When a container arrives to the terminal, it will have to check the correctness of its data. If the data is correct, it will ask for a container location form the appropiate service agent, according to the service the container is assigned to. After the location is known, it is communicated to the truck in order to assign it to the appropiate stack. On the other hand, if the container leaves the terminal, the process is similar to the above.

3.2 Agent Communication

The message interchange between all the agents forming the system is based on the FIPA-ACL standard [7]. Only the protocols needed for the terminal management are implemented, depending on the type of the message.

FIPA-query

This protocol is used to recover information about the current terminal state asking for it from the agent in charge of maintaining it. Its main use is to show the human system users the current terminal state through the ad-hoc designed interfaces. The message must allow the specification of the language used for the query (EDI, SQL, own language, …).

FIPA-request

This protocol is used to ask an agent for the execution of a specific action. For instance, to ask a service agent for a proper allocation for a container that is entering the terminal through a gate.

```
(request
     :sender gate1
     :receiver service3
     :content "...."
     :reply-with cont_8714442_ub
     :language ASCII
     :protocol FIPA-request )
```

FIPA-request-when

This protocol provides terminal information when a condition is satisfied (or an event is triggered). The most common uses are to notify the load profile when a cargo ship arrives or the arrival of a transtainer to a specified stack (before the beginning of the stacking operation).

FIPA-contract-net

One of the most important tasks is the coordination and the negotiation between the agents. Each one is in charge of one independent part of the terminal (a ship, a stack range, a transtainer, …). The global terminal performance depends on their working optimally. The protocol's main uses are:

- To negotiate which transtainer is in charge of manipulating the container.
- To ask another service agent for an available stack in its assigned stacks range for containers of a full service.

This protocol (Figure 5) allows for the efficient distribution of the decisions taking place in the terminal, avoiding the need for a coordination module that must follow any kind of incident closely.

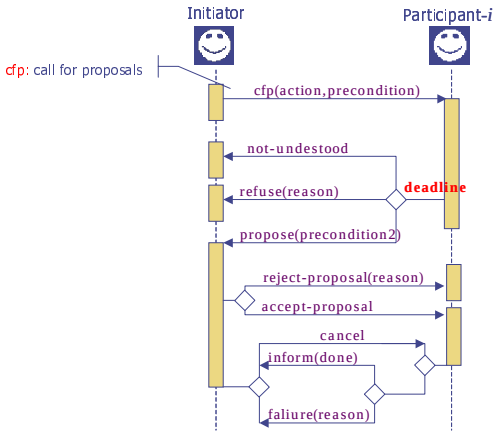

Fig. 5. FIPA-contract-net protocol [21].

Implementation

A first prototype of this system is currently being developed. Some of the agents presented here are already created. The interaction at both an internal level (between the forming different agents) as well as at an external level (with the system users) is shown in Figure 6.

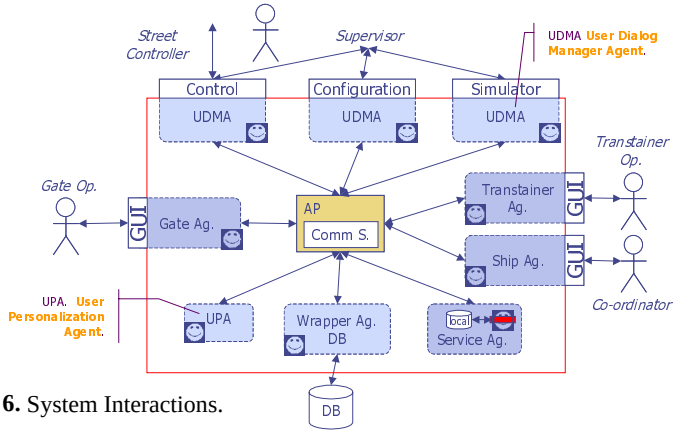

Fig. 6. System Interactions.

The directives of FIPA standard have been used as much as possible to carry out the implementation of this prototype. A set of auxiliary agents has appeared throughout the implementation process:

– A wrapper agent: provides access to the database, along with communication with external software.

– The UDMA (User Dialog Management Agent): interface agent with human users.
– The UPA (User Personalization Agent): manages the explicit profiles and preferences of the registered human users.

There is an agent platform (AP) to support the entire architecture. It provides the following services: yellow pages, white pages, a communication channel and an agent platform security manager.

4 Conclusions

In this paper, a multi-agent system architecture for the automatic allocation problem in a port container terminal has been presented. Apart from the benefits obtained from a multi-agent approach, the independence which is obtained in all of the presented subsystems must be pointed out. This architecture provides a maintenance of the necessary co-operation in order to minimise the time the ships are in the container terminal.

A preliminary version of the system is currently being implemented, which models the container terminal function of a real port. This prototype will soon be integrated with a yard simulator that is being developed at the same time. The main goal of this project is to implement the system as the integral management of a container terminal in the port which is associated with this project.

5 Acknowledgements

This research is partly supported by FEDER-CICYT research project 1FD97-2158-C04-01. The authors are grateful to Marítima S.L., who have made significant contributions to the project.

References

1. Baptiste, P.; Le Pape, C. Nuijten, W.; “Incorporating Efficient Operations Research Algorithms in Constraint-Based Scheduling”, First International Workshop on Artificial Intelligence and Operations Research, Timberline Lodge, Oregon, (1995).
2. Beliech, D.E.; “A Proposed Method for Efficient Preload Planning for Containerized Cargo Ships”, Master’s thesis, Naval Postgraduate School, Monterey, California, (1974).
3. “Recent developments in information technology for container terminal”, Cargo Systems, 1999.
4. Cho, D.W.; “Development of a Methodology for Containership Load Planning”, Ph. D. Dissertation, Oregon State Univ., (1982).
5. Chung, Y.G.; Randhawa, S.U.; McDowell, E.D.; “A Simulation Analisys for a Transtainer-based Container Handling Faciliyty”, Comp. Indust. Eng., Vol. 2, 14:113-125, (1988).

6. Daganzo, C.F.; "The Crane Scheduling Problem", Transportation Research, Part B 23:159-175, (1989).
7. "Foundation for Intelligent Physical Agents: FIPA 97 Specfication. Part 2, Agent Communication Language", (1997).
8. Fox, M.S.; Sadeh, N.; "Why is Scheduling Difficult? A CPS Perspective", 9th European Conference on Artificial Intelligence, 34:1-38, (1990).
9. Gifford. L.; "A Containership Load Planning Heuristic for a Transtainer Based Container Port", Master's thesis, Oregon State Univ., (1981).
10. Hart, P.E.; Nilsson, N.J.; Raphael, B.; Correction to "A formal basis for the heuristic determination of minimum cost paths", SIGART Newsletter, 37: 28-29, (1972).
11. Holguín-Veras, J.; Jara-Díaz, S.; "Optimal pricing for priority service and space allocation in container ports", Transportation Research, Part B 33:81-106, (1999).
12. Jennings, N. R.; Corera, J. M.; Laresgoiti, I.; "Developing industrial multi-agent systems". In Proceedings of the First International Conference on Mulit-agent Systems, (ICMAS-95), 423-430, (1995).
13. Jennings, N. Wooldridge, M. "Applications of Intelligent Agents". Queen Mary & Westfield College. University of London (1998).
14. Julian, V.; Carrascosa, C.; Soler, J.; "Una Arquitectura de Sistema Multi-Agente para la Recuperación y Presentación de Información", IV Congreso ISKO-España, EOCONSID'99, María José López-Huertas y Juan Carlos Fernández-Molina (editores), 291-296, (1999).
15. Korf, R.E.; "Linear-space best-first search", Artificial Intelligence 62:41-78, (1993).
16. Korf, R.E.; "An optimal admissible tree search", Artificial Intelligence 27: 97-109, (1995).
17. Kumar, V.; "Algorithms for Constraint Satisfaction Problems: A Survey", AI Magazine, Vol.1 , 13:32-4, (1992).
18. Kung, S.Y. ;"Digital Neural Networks", Prentice Hall, Inc.; (1993).
19. Ljungberg, M.; Lucas, A.; "The OASIS air-traffic management system". In Proceedings of the Second Pacific Rim International Conference on Artificial Intelligence, PRICAI'92, Seul, Korea, (1992).
20. Neves, M. C.; Oliveira, E.; "A Control Architecture for an Autonomous Mobile Robot". Agents'97, ACM, (1997).
21. Odell, J.; Parunak, H.; Bauer, B.: "Representing Agent Interactions Protocols in UML". Paper Submitted to AAAI Agents 2000 conference. Barcelona 3-7. June 2000.
22. Onaindia, E.; Barber, F.; Botti, V.; Carrascosa, C.; Hernández, M.A.; Rebollo, M.; "A Progressive Heuristic Search Algorithm for the Cutting Stock Problem", Lectures Notes in Artificial Intelligence, Springer-Verlag, pp:25 – 35, (1998).
23. Peterkofsky, R.I.; Daganzo, C.F.; "A Branch and Bound Solution Method for the Crane Scheduling Problem", Transportation Research, Part B, Vol. 3, 24:159-172, (1990).
24. Sadeh, N.; Hildum, D.; Laliberty, T.; Smith S.; Nulty, J., Kjenstand, D.; "Mixed-initiative management of integrated process-planning and production-scheduling solutions", Proc. Workshop A.I. and Manufacturing Research, Alburquerque, (1996).
25. Shields, J.J.; "Containership Stowage: A Computer-Aided Preplanning System", Marine Technology, Vol. 4, 21:370-383, (1984).
26. Shoham, Y.; "Agent0: A simple agent language and its interpreter". In Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91), 704-709, (1991).
27. Wooldridge, M.; Jennings, N. R.; "Intelligent Agents: Theory and Practice", The Knowledge Engineering Review, vol. 10 (2) pp. 115-152 (1995).