Self-Managing Agents for Dynamic Scheduling in Manufacturing

Ana Madureira
Computer Science Department
School of Engineering of Porto
Rua Dr. Bernardino de Almeida, 431
Telefone: +351 22 834 05 24
amd@isep.ipp.pt

Joaquim Santos
Computer Science Department
School of Engineering of Porto
Rua Dr. Bernardino de Almeida, 431
Telefone: +351 22 834 05 24
filipe@dei.isep.ipp.pt

Ivo Pereira
Computer Science Department
School of Engineering of Porto
Rua Dr. Bernardino de Almeida, 431
Telefone: +351 22 834 05 24
i020541@dei.isep.ipp.pt

ABSTRACT
The main purpose of this paper is to propose a Multi-Agent Autonomic and Bio-Inspired based framework with self-managing capabilities to solve complex scheduling problems using cooperative negotiation. Scheduling resolution requires the intervention of highly skilled human problem-solvers. This is a very hard and challenging domain because current systems are becoming more and more complex, distributed, interconnected and subject to rapidly changing. A natural Autonomic Computing (AC) evolution in relation to Current Computing is to provide systems with Self-Managing ability with a minimum human interference.

Categories and Subject Descriptors
I.2 [Artificial Intelligence]: I.2.8 Problem Solving, Control Methods, and Search – *Scheduling.* I.2.11 Distributed Artificial Intelligence - *Multiagent systems*.

General Terms
Algorithms, Design.

Keywords
Autonomic Computing, Multi-Agent Systems, Bio-Inspired Techniques, Dynamic Scheduling

1. INTRODUCTION
Traditionally scheduling resolution requires the intervention of highly skilled human problem-solvers. This is a very hard and challenging domain because current systems are becoming more and more complex, distributed, interconnected and subject to rapidly changing. For these dynamic optimization problems environments, that are often impossible to avoid in practice, the objective of the optimization algorithm is no longer to simply locate the global optimal solution, but to continuously track the optimum, or to find a robust solution that operates optimally in the presence of perturbations [1][12].

The complexity of current computer systems has led the software engineering, distributed systems and management communities to look for inspiration in diverse fields, e.g. robotics, artificial intelligence or biology, to find new ways of designing and managing systems. Hybridization and combination of different approaches seems to be a promising research field of computational intelligence focusing on the development of the next generation of intelligent systems.

A natural Autonomic Computing (AC) evolution in relation to Current Computing is to provide systems with Self-Managing ability with a minimum human interference.

Considering that AC is a grand-challenge vision of the future in which computing systems will manage themselves in accordance with high-level objectives specified by humans, we pretend with the proposed system to give a meaningful contribution in the field of Autonomic Computing application for dynamic scheduling in Manufacturing Systems.

Multi-agent paradigm is emerging for the development of solutions to very hard distributed computational problems. This paradigm is based either on the activity of "intelligent" agents which perform complex functionalities or on the exploitation of a large number of simple agents that can produce an overall intelligent behavior leading to the solution of alleged almost intractable problems.

Bio-Inspired Techniques (BIT) form a class of powerful and practical solution techniques for tackling complex, large-scale combinatorial problems producing efficiently high-quality solutions. From the literature we can conclude that they are adequate for static problems. However, real scheduling problems are quite dynamic, considering the arrival of new orders, orders being cancelled, machine delays or faults, etc. Scheduling problem in dynamic environments have been investigated by a number of authors, see for example [1] [8-9] [12].

The concept of developing the next era of computing systems is driven by the convergence between Biological Systems and the Digital Computing Systems. We belief that our main challenge in this work through autonomic systems is the effort integration of fourth research areas: Autonomic Computing(AC), Multi-Agent Systems (MAS), Bio-Inspired Techniques (BIT) and Real World Scheduling (RWS).

The remaining sections are organized as follows: Section 2 summarizes some related work on dynamic scheduling trough MAS and Bio-Inspired Techniques. Section 3 presents issues and links for Autonomic Computing. Some aspects of MAS and BIT are discussed on section 4 e 5. In section 6 the scheduling problem under consideration is presented. Section 7 presents the AutoDynAgents System and describes implemented mechanisms.

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-131-6/08/07.

Finally, the paper presents some conclusions and puts forward some ideas for future work.

2. DYNAMIC SCHEDULING

Current global and highly competitive market, enterprises must be aware of momentary market opportunities, and quickly and properly react to customers' demands. The ability to deal with a larger number of smaller quantity orders increases the stress on the scheduling procedure.

Finding good solutions to scheduling problems is very important to real manufacturing systems because the production rate and production costs are very dependent on the schedules used for controlling the flow of work through the system.

Scheduling problems arise in a diverse set of domains, ranging from manufacturing to hospitals settings, transports, computer and space environments, amongst others. Most of these domains are characterized by a great amount of uncertainty that leads to significant system dynamism. Such dynamic scheduling is receiving increasing attention amongst both researchers and practitioners.

Dynamic changes of a problem could arise from new user requirements and the evolution of the external environment. In a more general view, dynamic problem changes can be seen as a set of constraint insertions and cancellations.

For these dynamic optimization problems environments, that are often impossible to avoid in practice, the objective of the optimization algorithm is no longer to simply locate the global optimum solution, but to continuously track the optimum in dynamic environments, or to find a robust solution that operates optimally in the presence of perturbations [11-12]. In spite of all the previous trials the scheduling problem still known to be NP-complete[3], even for static environments. This fact poses serious challenges to conventional algorithms and incites researchers to explore new directions [1][3][15] and Multi-Agent technology has been considered an important approach for developing industrial distributed systems[13][18].

Supply chains are evolving to more coupled organizations like virtual enterprises, though maintaining the single entities autonomy, adaptability and dynamism properties. Such organizations imply organizational and technological developments through agility, distribution, decentralization, reactivity and flexibility. New organizational and technological paradigms are needed in order to reply to the modern manufacturing systems challenges. Some recent trends in manufacturing in particular and business in general, lead to new approaches regarding the organization and software architecture, mainly adopting distributed solutions. Multi-Agent Systems (MAS) is a promising approach for developing applications in complex domains.

3. AUTONOMIC COMPUTING

Autonomic Computing is an initiative started by IBM in 2001. Its ultimate aim is to create self-managing computer systems to overcome their rapidly growing complexity and to enable their further growth [4]. Autonomic systems are intrinsically intended to reduce the complexity of managing systems and human interference through automation.

AC describes the set of concepts, technologies, and tools that enable applications, systems, and entire networks to become more self-managing. Self-management involves four qualities which are often referred to as self-CHOP characteristics: self-Configure, self-Heal, self-Optimize, and self-Protect. The word autonomic is borrowed from physiology; as a human body knows when it needs to breathe, software is being developed to enable a computer system to know when it needs to repair itself, configure itself, and so on.

Computational systems that are to manage themselves have been part of the vision for computer science since the work of Charles Babbage. With the increasing complexity of advanced information technology systems, and the increasing reliance of modern society on the systems, attention in recent years has returned to AC. Such systems have come to be called self-* systems (pronounced "self-star"), with the asterisk indicating that a variety of attributes are under consideration. While an agreed definition on self-* systems is still emerging, aspects of these systems include properties such as: self-awareness, self-organization, self-configuration, self-management, self-diagnosis, self-correction, and self-repair.

Such systems abound in nature, from the level of ecosystems, through large primates (such man) and down to processes inside single cells. Similarly, many chemical, physical, economic and social systems exhibit self-* properties. Thus, the development of computational systems that have self-* properties is increasingly drawing in research in biology, ecology, statistical physics and the social sciences. Recent research on computational self-* systems has tried to formalized some of the ideas from these different disciplines, and to identify algorithms and procedures that could realize various self-* attributes [10].

Computational self-* systems and networks provide an application domain for research and development of agent technologies, and also a contribution to agent-based computing theory and practice, because many self-* systems may be viewed as involving interactions between autonomous entities and components.

Ultimately, the aim is to realize the promise of Information Technology: increasing productivity while minimizing complexity for users. The key message to be drawn from this vision is that it shares many of the goals of agent-based computing, and agents offer a way to manage the complexity of self-* and autonomic systems.

Traditionally, problem determination requires the presence of highly skilled human problem-solvers, but in autonomic computing the systems themselves are equipped with some degree of problem diagnosis and resolution capability with minimum human interference [10].

We consider AC, in AutoDynAgents context, when compared to other paradigms, to be extremely interesting to the dynamic scheduling resolution in the following aspects:

- the role of self-organization, a powerful methodology as demonstrated in nature and well suited to the problems that involve large-scale, distributed, and locally interacting;
- to solve computationally hard problems, e.g., large-scale computation, distributed constraint satisfaction, and decentralized optimization, that are dynamically evolving

and highly complex in terms of interaction and dimensionality;

- to characterize complex emergent behavior in natural and artificial systems which involve a large number of self-organizing, interacting entities;
- to capture the essence of autonomy in natural and artificial systems.

Considering existing methods for modeling autonomy are successful to some extent, a generic model or framework for handling problems in complex manufacturing systems is still missing. Autonomy Oriented Computing offers a new computing paradigm that makes use of autonomous entities in solving computational problems and in modeling complex systems.

Two important questions must be solved at this phase: "*How much human involvement is necessary?*" and "*How sophisticated a model of computational autonomy must be?"*

4. MULTI-AGENT SYSTEMS

Considering the complexity inherent to the manufacturing systems, dynamic scheduling is considered an excellent candidate for the application of agent-based technology.

The main term of Multi-Agent based computing is an Agent. However the definition of the term Agent has not common consent. In the last few years most authors agreed that this definition depends on the domain where agents are used. However there is a general consensus about its two main abstractions:

- An agent is a computational system that is situated in a dynamic environment and is capable of exhibiting autonomous and intelligent behaviour.
- An agent may have an environment that includes other agents. The community of interacting agents, as a whole, operates as a multi-agent system.

Some of most important common properties of computational agents are as follows [21]:

- act on behalf of their designer or the user they represent in order to meet a particular purpose.
- are autonomous in the sense that they control both their internal state and behaviour in the environment.
- exhibit some kind of intelligence, from applying fixed rules to reasoning, planning and learning capabilities.
- interact with their environment, and in a community, with other agents.
- are ideally adaptive, i.e., capable of tailoring their behaviour to the changes of the environment without the intervention of their designer.

Additional agent properties, characteristic in particular domains and applications are mobility (when an agent can transport itself to another environment to access remote resources or to meet other agents), genuineness (when it does not falsify its identity), credibility or trustworthiness (when it does not communicate false information wilfully) and sociality (when agents work in open operational environments hosting the execution of a multiplicity of agents, possibly belonging to different stakeholders (think, e.g., of agent-mediated marketplaces).

In many implementations of MAS systems for manufacturing scheduling, the agents model the resources of the system and the tasks scheduling is done in a distributed way by means of cooperation and coordination amongst agents [14]. There are also approaches that use a single agent for scheduling that defines the schedules that the resource agents will execute [9], [12]. When responding to disturbances, the distributed nature of multi-agent systems can also be a benefit to the rescheduling algorithm by involving only the agents directly affected, without disturbing the rest of the community that can continue with their work.

This fact incites researchers to explore new directions and Multi-Agent technology has been considered an important approach for developing industrial distributed systems [5][17][19-21].

5. BIO-INSPIRED TECHNIQUES

Bio-Inspired Techniques (BIT) is the set of computing techniques inspired by biologically systems that are derived from nature. The distinction between Bio-inspired algorithms and Meta-Heuristics (MH) is largely counterproductive. Although surface-level dissimilarities, the central themes underlying these two classes of heuristic are nearly identical, e.g., intensification versus diversification, mechanisms for escaping local optimum, intelligent design of selection/mutation/crossover operators, and the structure of the fitness landscape. The family of BIT includes, but it is not limited, to Genetic Algorithms, Tabu Search, Simulated Annealing, Adaptive Memory procedures, Scatter Search, Soft Computing, Evolutionary Methods, Ant Systems, Particle Swarm Optimization and their hybrids.

The interest of this class of approaches is that they converge, in general, to satisfactory solutions in an effective and efficient way (computing time and implementation effort). For literature on this subject, see for example [6].

6. PROBLEM DEFINITION

Real world scheduling problems have received a lot of attention in recent years. In this work we consider the resolution of realistic problems. Most real-world multi-operation scheduling problems can be described as dynamic and extended versions of the classic Job-Shop scheduling combinatorial optimization problem.

In practice, many scheduling problems include further restrictions and relaxation of others [16]. Thus, for example, precedence constraints among operations of the different jobs are common because, often, mainly in discrete manufacturing, products are made of several components that can be seen as different jobs whose manufacture must be coordinated. Additionally, since a job can be the result of manufacturing and assembly of parts at several stages, different parts of the same job may be processed simultaneously on different machines (concurrent or simultaneous processing).

Moreover, in practice, scheduling environment tends to be dynamic, i.e. new jobs arrive at unpredictable intervals, machines breakdown, jobs can be cancelled and due dates and processing times can change frequently[15].

The problem, focused in our work, which we call Extended Job-Shop Scheduling Problem (EJSSP) [11-12], has major extensions and differences in relation to the classic Job-Shop Scheduling Problem (JSSP). In this work, we define a job as a manufacturing order for a final item, that could be Simple or Complex [11]. It

may be Simple, like a part, requiring a set of operations to be processed. We define it as Simple Product or Simple Final Item. Complex Final Items, requiring processing of several operations on a number of parts followed by assembly operations at several stages, are also dealt with.

7. PROPOSED SYSTEM

Traditionally, scheduling resolution requires the intervention of highly skilled human problem-solvers. This is a very hard and challenging domain because current systems are becoming more and more complex, distributed, interconnected and subject to rapidly changing.

Distributed environment approaches are important in order to improve scheduling systems flexibility and capacity to react to unpredictable events. It is accepted that new generations of manufacturing facilities, with increasing specialization and integration, add more problematic challenges to scheduling systems. For that reason, issues like robustness, regeneration capacities and efficiency are currently critical elements in the design of manufacturing scheduling system and encouraged the development of new architectures and solutions, leveraging the MAS research results.

A natural Autonomic Computing evolution in relation to Current Computing is to provide systems with Self-Managing ability with a minimum human interference. Considering that AC is a grand-challenge vision of the future in which computing systems will manage themselves in accordance with high-level objectives specified by humans, we pretend with AUTODYNAGENTS (Autonomic Agents with Self-Managing Capabilities for Dynamic Scheduling Support in a Cooperative Manufacturing System) project to give a meaningful contribution in the field of Autonomic Computing application for dynamic scheduling in Manufacturing Systems.

The concept of developing the next era of computing systems is driven by the convergence between Biological Systems and the Digital Computing Systems. AutoDynAgents is a project envisaging the use of Multi-Agent Systems paradigm for supporting dynamic and distributed scheduling in Manufacturing Systems with Autonomic properties, in order to reduce the complexity of managing systems and human interference.

The objective of this work is to define a Multi-Agent Autonomic and Bio-Inspired based framework with learning and self-managing capabilities to solve complex scheduling problems using cooperative negotiation. The scheduling system will be equipped with some degree of problem diagnosis and resolution capability, incorporating the concept of Autonomic Computing, by self-configuration, self-optimization and self-healing.

The final product of AutoDynAgents is an Autonomic Scheduling System in which communities of agents model a real manufacturing system subject to perturbations. Agents must be able to learn and manage their internal behavior and their relationships with other autonomic agents, by cooperative negotiation in accordance with business policies defined by user manager. Cooperative Negotiation is quite important at this approach; we consider a Multi-dimensional Negotiation process depending upon the effort that the agents want to expend based on business.

We intend that AutoDynAgents system has the Self-Configuring, Self-Optimizing and Self-Healing capabilities.

The main purpose of AUTODYNAGENTS is to create a Multi-Agent system where each agent represents a resource (Machine Agents) in a Manufacturing System.

Each Machine Agent must be able:

- to find an optimal or near optimal local solution trough Genetic Algorithms or Tabu Search meta-heuristics.
- to deal with system dynamism (new jobs arriving, cancelled jobs, changing jobs attributes, etc).
- to change/adapt the parameters of the basic algorithm according to the current situation.
- to switch from one Meta-Heuristic algorithm to another.
- to cooperate with other agents.

The original Scheduling problem defined in section 2, is decomposed into a series of Single Machine Scheduling Problems (SMSP)[1],[5]. The Machine Agents (which has an Meta-Heuristic associated) obtain local solutions and later cooperate in order to overcome inter-agent constraints and achieve a global schedule.

7.1 AUTODYNAGENTS Architecture

The proposed Team-Work architecture is based on three different types of agents. In order to allow a seamless communication with the user, a User Interface Agent is implemented. This agent, apart from being responsible for the user interface, will generate the necessary Task Agents dynamically according to the number of tasks that comprise the scheduling problem and assign each task to the respective Task Agent.

The Task Agent will process the necessary information about the job. That is to say that this agent will be responsible for the generation of the earliest and latest processing times, the verification of feasible schedules and identification of constraint conflicts on each job and the decision on which Machine Agent is responsible for solving a specific conflict.

Finally, the Machine Agent is responsible for the scheduling of the operations that require processing in the machine supervised by the agent. This agent will implement meta-heuristic and local search procedures in order to find best possible operation schedules and will communicate those solutions to the Task Agent for later feasibility check (Figure 1).

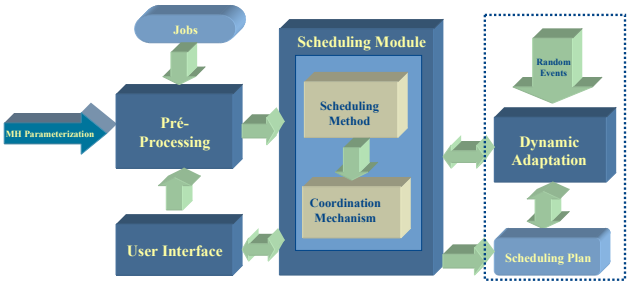

Figure 1. AutoDynAgents System Architecture

7.2 Coordination Considerations

In a real manufacturing system a product is produced, step by step, passing on several machines. In each machine it will be performed at least one operation (job) of the process plan. In our approach we have one agent for each machine. However, if we join solutions obtained by our machine agents we will observe that, some times, they will not be feasible. In fact, if operation op1 (in machine m1) precedes operation Op2 (in machine m2) and Op2 precedes Op3 (in machine m3) in a manufacturing process, it is not guaranteed that the initial time for Op2 in m2 will be after the end of Op1 in m1 nor that the end of Op2 in m2 will be before the start of Op3 in m3.

Two possible approaches, to deal with this problem, could be used. In the first, the AUTODYNAGENTS system waits for the solutions obtained by the machine agents and then apply a repair mechanism to shift some operations in the generated schedules till a feasible solution is obtained (Repair Approach). In the second, a coordination mechanism is established between related agents in the process, in order to interact with each other to pursuit common objective trough cooperation. These coordination mechanisms are prepared to accept agents subjected to dynamism (new jobs arriving, cancelled jobs, changing jobs attributes). The latter approach is the one implemented in the proposed system.

7.3 Self-Managing Mechanisms for Autonomic Agents

Generally, self-organization can be defined as the process by which systems tend to reach a particular objective with no external interference. All the mechanisms dictating its behavior is internal to the system e.g. are autonomous. This field of research has received much attention trough Autonomic Computing paradigm [4].

We envisage to define Self-Managing mechanisms for a Cooperative Scheduling System considering that AutoDynAgents must be able to perform scheduling in highly dynamic environments where there is incomplete information and changes often occur; modify previously formed schedules considering recent dynamic information, minimizing the disruption of earlier schedules and still aiming for the most effective possible use of resources and achievement of goals and provide flexibility to react robustly to any disruption in an efficient and timely manner.

Rescheduling is necessary due to two classes of events [11]:

- **Partial events** imply variability in jobs/operations attributes such as processing times, due dates or release times
- **Total events** imply variability in neighborhood/population structure, resulting from new job arrivals, job cancellations, machines breakdown, etc.

We intend to define the following Self-Managing mechanisms:

- **Self-Configuring** - enable agents to adapt to changing conditions by changing their own configurations, allowing the addition and removal of resources without service disruption. Machine Agents will be prepared to handle dynamism by adapting the solutions to external perturbations. While, on one hand, partial events only require redefining job attributes and re-evaluation of the objective function, total events require a change on solution structure and size, carried out by inserting or deleting operations, and also re-evaluation of the objective function. Therefore, under a total event, the modification of the current solution is imperative, trough Job arrival integration mechanisms (When a new job arrives to be processed), Job elimination mechanisms (When a job is cancelled) and Regeneration mechanisms in order to unsure a dynamic adaptation of population/neighborhood.
- **Self-Optimizing** – the ability of the agent to monitor its state and performance and proactively tune itself to respond to environmental stimuli. Each machine agent adopt and provides self-parameterization of the solving method in accordance with the problem being solved (parameters can change in run-time). Each machine agent must be able to define which BIT will be used and define initial parameters of BIT according to the current situation or even to commutate from one algorithm to the other according to current state and previous learning. Agents self-optimize through learning and experience.
- **Self-Healing** – giving agents the capacity to diagnose deviations from normal conditions and take proactively action to normalize them and avoid service disruptions. For example AutoDynAgents architecture is composed by a set of autonomous agents organized by Team-Work objectives, each agent could know the actual state of a neighbor, and when its neighbor "dies" for some reason, it could be able of instantiate a new agent with the recovered state. This will eliminate large costs for systems when a small part of him falls. When a machine is breakdown the respective agent could be replaced by another machine agent (alternative machine) and maintaining system stable.

In this paper we consider that BIT self-parameterization could permit a better adaptation to the dynamic situation being considered. The idea is that each agent adopts and provides self-parameterization in accordance with the problem being solved: the method and/or parameters can change in run-time, the agents can use different MH according with problem characteristics, etc.

BIT can be adapted to deal with dynamic problems, reusing and changing solutions/populations in accordance with the dynamism. We will use the Dynamic Adaptation Mechanisms defined in [11], which includes a method for neighborhood regeneration under dynamic environments, increasing or decreasing it according to new job arrivals or cancellations.

8. CONCLUSIONS AND FUTURE WORK

We believe that a new contribution for the resolution of more realistic scheduling problems (Extended Job Shop Problems) was described in this paper. The particularity of our approach is the procedure to schedule operations, as each machine will first find local optimal or near optimal solutions, succeeded by the interaction with other machines trough cooperation mechanism as a way to find an optimal or near-optimal global schedule.

As result of this project we expect to prove some ideas for which we are now claiming. From these we refer the following:

- Use Autonomic Computing to reproduce Life-like behavior in computation to explain, predict, reconstruct and deploy complex systems, with a minimum human interference.
- Multi-agent Systems are adequate to model and support dynamic and distributed scheduling with Cooperative Negotiation.
- Multi-agent paradigm is often inspired by biologically systems. Observing Manufacturing Systems like evolution-based social systems will be important in order to allow a better understanding and integration between the machinery and humans.
- Learning in Multi-agent Autonomic Systems is a challenging problem, so does Optimization. Optimization in such environments must deal with dynamism.
- Bio-Inspired Techniques can be adapted to deal with dynamic problems, reusing and changing solutions in accordance with system dynamism.

9. ACKNOWLEDGMENTS

The authors would like to acknowledge FCT, FEDER, POCTI, POCI for their support to R&D Projects and the GECAD Unit.

10. REFERENCES

[1] Aytug, Haldun, Lawley, Mark A, McKay, Kenneth, Mohan, Shantha, & Uzsoy, Reha. Executing production schedules in the face of uncertainties: A review and some future directions. European Journal of Operational Research, Volume 16 (1), 86-110, 2005.

[2] Babiceanu, R.F and Chen, F.F, Development and Applications of Holonic Manufacturing Systems: A Survey, Journal of Intelligent Manufacturing, 17: 111-131, 2006.

[3] Blazewicz, Jacek, Ecker, Klaus H,.& Trystram, Denis, Recent advances in scheduling in computer and manufacturing systems. European Journal of Operational Research, 164(3), 573-574.(1), 86-110, 2005.

[4] EMA., Practical Autonomic Computing: Roadmap to Self Managing Technology - A White Paper Prepared for IBM , Enterprise Management Associates, 2006.

[5] FARMS LAB - Laboratory for Fundamental and Applied Research in Multi-agent Systems, Multi-Agent Scheduling in Manufacturing Systems: http://farm.ecs.umass.edu/~pschiegg/bib/lit.html

[6] Gonzalez, Teofilo F. Handbook of Approximation Algorithms and Metaheuristics. Chapman&Hall/Crc Computer and Information Science Series, 2007.

[7] Horling, Brian and Lesser, Victor, A Survey of Multi-Agent Organizational Paradigms, University of Massachusets, 2005.

[8] Jain, S. and Meeran, S., Deterministic Job Shop scheduling: past, present and future, European Journal of Operational Research, nº113, 390-434, 1999.

[9] Logie, S., Sabaz, D., and Gruver, W.A., Sliding Window Distributed Combinatorial Scheduling unsing JADE, IEEE International Conference on Systems, Man and Cybernetics, 2004.

[10] Luck, M., McBurney, P., Shehory, O., Willmoth, S., and et al., Agent Technology: Computing as Interaction. A Roadmap for Agent-Based Computing, AgentLink III, http://www.agentlink.org/roadmap/al3rm.pdf, 2005.

[11] Madureira, Ana, Meta-Heuristics Application to Scheduling in Dynamic Environments of Discrete Manufacturing. PhD Dissertation. University of Minho, Braga, Portugal, 2003 (in portuguese).

[12] Madureira, Ana, Ramos, Carlos and Silva, Sílvio C., Toward Dynamic Scheduling Through Evolutionary Computing. WSEAS Transactions on Systems. Issue 4. Volume 3, 2004, pp.1596-1604.

[13] Monostoria, L., Vánczaa, J., and Kumara, S.R.T, Agent-Based Systems for Manufacturing , CIRP Annals - Manufacturing Technology, Volume 55, Issue 2, Pages 697-720, 2006.

[14] Nwana, H., Lee, L. and Jennings, N., Coordination in Software agent systems, BT Technol J Vol 14 No 4 October, 1996.

[15] Pinedo, M., Planning and Scheduling in Manufacturing and Services, Springer-Verlag, New York, ISBN:0-387-22198-0, 2005.

[16] Portmann, M. C., Scheduling Methodology: optimization and compu-search approaches, in the planning and scheduling of production sys-tems, Chapman &Hall, 1997.

[17] Russel, S., and Norvig, P, Artificial Intelligence: A Modern Approach, Prentice Hall/Pearson Education International: Englewood Cliffs (NJ), (2nd Ed), 2003.

[18] Shen, W., and Norrie, D., Agent-based systems for intelligent manufacturing: a state of the art survey, Int. J. Knowl. Inform. Syst., vol. 1, no. 2, pp. 129– 156, 1999.

[19] Shen, W., Hao, Q., Yoon, J. and Norrie, D.H., Applications of Agent Systems in Intelligent Manufacturing: An Updated Review, Advanced Engineering Informatics, Volume 20, Issue 4, October 2006, Pages 415-431, 2006.

[20] Weiss, Gerard, MultiAgents Systems – A Modern Approach to Distributed Artificial Intelligence, The MIT Press, 1999.

[21] Wooldridge, M., An Introduction to Multiagent Systems, John Wiley and Sons, 2002