An agent-based simulation model using decoupled learning rules to (re)schedule multiple projects

Laura Tosselli[1] , Verónica Bogado[1], Ernesto Martínez[2]

[1] Departamento Ingeniería en Sistemas de Información – UTN FRVM, Argentina
{ltosselli, vbogado}@frvm.utn.edu.ar
[2] INGAR (CONICET – UTN) – UTN FRSF, Argentina
ecmarti@santafe-conicet.gob.ar

Abstract. Competitive pressures and business globalization have led many organizations, mainly technology-based and innovation-oriented companies, to adopt project-based organizational structures. In a multi-project context within enterprise networks, reaching feasible solutions to the multi-project (re)scheduling problem represents a major challenge, where autonomy and decentralization of the environment favor agent-based simulation This work presents and validates a simulation-based multi-agent model using the fractal company concept to solve the complex multi-project (re)scheduling problem in enterprise networks. The proposed agent-based model is tested trough a set of project instances that vary in project structure, project parameters, number of resources shared, unplanned events that affect them, etc. Results obtained are assessed through different scheduling goals, such project total duration, project total cost, leveling resource usage, among others to show that decoupled learning rules allows finding a solution which can be understood as a Nash equilibrium for the interacting agents and it is far better compared to the ones obtained with existing approaches.

Keywords: Multi-project (re)scheduling, project-oriented fractal organization, Multi-agent simulation.

1 Introduction

Multi-Project (re)scheduling is considered as a NP-hard problem, thus becoming a difficult task for project leaders when many tasks and resources are involved which prevents the application of optimization-based methods to find a repaired schedule [1]. Thus, in a multi-project context within enterprise networks, there are conflicting constraints and interrelationships among projects that cause an increase of the complexity, making project rescheduling a difficult problem to be addressed under real-time pressure and selfish behavior of concerned agents [2]. Particularly, the unplanned events impact on due dates and milestones achievement, budget consumption, and resources usage, which in turn affect timing and quality of projects goals delivery because of inefficient responses to such events. Then, the multi-project (re)scheduling problem within an enterprise network requires techniques and tools that allow decision-making in an autonomous and decentralized manner.

However, solving (re)scheduling problems are generally carried out in a centralized way, which makes it difficult to respond to unplanned events autonomously involving selfish agents. Thus, to achieve the project (re)scheduling problem dealing with unplanned events in a decentralized and efficient manner, organizations must adopt structures with a high degree of flexibility, allowing the reconfiguration of their constituent parts. In this context, the concept of a *Fractal Company* [3] has been proposed, which combined with the project-oriented approach gives rise to the needed flexibility, autonomy, decentralization, self-organization, among other features. Therefore, this proposal is based on the Project-oriented Fractal Company Model developed in [4] to implement a *Multi-agent System* (MAS) composed by autonomous and selfish entities that interact to solve the multi-project (re)scheduling problem. These entities do not have complete knowledge about the other strategies and payoffs, and there are also constraints related to the exchange of information. The establishment of *client-server* and *delegation* relationships among end-managers (projects) and mean-managers (resources) provides the flexibility for rescheduling tasks at different abstraction levels.

Several works related to the project scheduling problem have been presented using the agent-based approach [5, 6, 7] where an initial schedule is obtained from the interaction among relevant entities in the problem domain. However, many of these proposals are based on the assumption that the project schedule will be implemented as initially defined. This situation is not representative of the project management environment since 80% of project schedules are affected by unplanned events during their execution, affecting deadlines, costs and estimated resource usage [8].

In this context, arises the need to solve the multi-project (re)scheduling problem in a decentralized and autonomous way, providing metrics and indicators that support the decision-making process when unplanned events affect the initial schedule. Therefore, the objective of this work is to present and validate a Multi-agent Simulation model to deal with the complex multi-project (re)scheduling problem incorporating in each of the agents that compose it, a decoupled learning rule called *Learning by Trial and Error* [9], following different objectives and obtaining (re)scheduling solutions from the strategic interactions among them. Then, the Agent-based Simulation Model allows evaluating the emergent solution. A standard and representative set of multi-project problem instances [10] that vary in project structure, project parameters and number of shared resources is used to test the presented Multi-agent System against other multi-agent proposals. Results obtained are encouraging and demonstrate the applicability of agent-based simulation.

2 Problem Description

In this section, the multi-project (re)scheduling problem based on a project-oriented fractal model for enterprise networking is considered. This problem can be defined as follows:

- A set of I projects to be (re)scheduled within a project-oriented fractal organization, whose managers (agents) must respond to unplanned events and disturbances. These projects are interdependent and run to a certain extent in parallel. Additionally, each project has properties such as deadline, budget, estimated start and finish time, resource requirements, dependence relationships, precedence relationships, among others. Some of these properties are considered as domain constraints. Each project i is considered as a *fractal unit* and it consists of a *recursive* structure, where a project can be composed in sub-projects, and these sub-projects can be composed of other sub-projects, and so on and so forth. For this reason, the recursive decomposition of a project continues until its minimum expression, i.e., a task that is also considered as a project. Each *fractal unit* or *project* is composed of a project manager and a managed object, where each managed object can be an *end* (e.g., a project, a sub-project, a phase) or a *mean* (a resource). Each project manager carry out functions such as: negotiation, (re)scheduling and has learning capabilities.

- A set R of links of *delegation* and *client-server* type. The delegation relationships are the result of recursive structure and client-server relationships are the result of negotiations, which are auction-based interactions among fractal units that manages projects or resources, where each manager of a project demands resources for fulfilling its scheduled work, whereas the managers of resources sell their specific capabilities and skilled workforce to different projects.
- A set of K available resources with properties such as: processing capacity, maximum availability, resource cost per unit time, list of scheduled tasks for processing. Many of these resources are shared among projects.
- A set of E unplanned events that may affect the execution of the initial schedule. In this work, the types of events are related to variations in the availability of resources.

The scheduling of different goals considered in this work are related to the multiple constraints that should be traded off in a multi-project context, such as time, cost and resources. These goals are:

(1) Minimize the total duration of the project set I.
(2) Minimize the total cost of the project set I.
(3) Minimize the variation of resource utilization in each time period (solution stability) of project set I.

The first goal in addressing the rescheduling of multiple projects is related to the *Resource Constrained Project Scheduling Problem* (RCPSP), which seeks to obtain, for each project, a schedule that have the shortest possible total project duration (TPD), subject to limited amounts of the resources available and precedence/synchronization relationships among projects/tasks [10, 11]. This objective is defined as follows for any project i:

$$TPD=min\ \{max\ \{ft_i\}\} \tag{1}$$

The term ft_i is defined as the finalization time of latest task in the project i schedule.

The second scheduling goal considered in this work aims to obtain solutions that present the lowest possible total cost (TC), giving rise to solutions to rescheduling of multiple projects corresponding to the *Resource Investment problem* (RIP) [12]. In this context, the objective is defined as follows:

$$TC \ = min \ \sum_{p=1,k=1}^{n,q} ral_{pk} \tag{2}$$

where ral_{pk} is the cost of each resource allocation that form part of the schedule for project $\boldsymbol{i}$.

Finally, the third scheduling goal is related to the problem named *Resource leveling problem* (RLP) [12], and it aims to find solutions (schedules) that have the least variation in the resource usage between different time periods (stability). This goal SS tries to minimize the impact on the original schedule of the changes made:

$$SS = min \sum_{k=1}^{q} (ur_{k,t} - ur_{k,t-1}) \tag{3}$$

where $(ur_{k,t})$ represents the amounts of resource usage in the time period t. This goal allows measuring the adaptability of the solution found to the changes in the project environment and leveling the resource usage within the availability constraints to solve resource over-allocations [2, 12].

3 Multiagent-based Methodologies

3.1 Agent-based modeling and simulation

Agent-based modeling and simulation (ABMS) is concerned with creating computational models such artificial societies where self-interested and autonomous agents interact among them following different goals [13]. In this proposal, the developed simulation model seeks to imitate the sequence of strategic interactions in the Project-oriented Fractal Organization when responding to events while addressing the multi-project (re)scheduling problem in a decentralized and distributed way. Thus, the multi-project (re)scheduling problem is modeled as a repetitive and interdependent *game* [9], compound of *N* players (agents) and *M* stages (repeated iterations), where each of the agents pursues its specific goals in a selfish way. Furthermore, the simulation model is a key tool to evaluate emergent scheduling behaviors from established *client-server* and *delegation* relationships among interacting agents.

The proposed agent-based model is composed of two kinds of agents, *Project* and *Resource* agents. Each agent class has different goals. A *Project* agent (PA) manages a project, a sub-project or a task, aims at minimizing its duration and total cost. On the other hand, a *Resource* agent (RA) manages a resource aiming to maximize its profits. These individual goals (micro-level) lead to global and emergent behaviors (macro-level) which are aligned to the scheduling objectives defined in *Section 2*. The defining properties for project and resource agents are described in Table 1 and Table 2 respectively.

Table 1. *Project* agent properties

Property	Description
pId	Project ID
pLevel	Project level
pGoal	Project goal
estST	Estimated project start time
estFT	Estimated project finish time
deadline	Project latest finish time
PRi	Set of precedence relationships between other projects and project *i*
DRi	Set of dependence relationships between other projects and project *i*
req_{ij}	Resource requirements for task *j* on project *i*
P_u_i	Payoff function for project agent *i* defined as: $min\ TPD_i + TC_i$

Table 2. *Resource* agent properties

Property	Description
rId	Resource ID
maxAvailability	Resource Maximum available capacity per unit time
rCapacity	Resource Processing Capacity (%)
rType	Define local or global resource type
R_u_i	Payoff function for resource agent *k* defined as: $max \sum_{p=1}^{n} ral_{pk}$

The above defined agent types for interacting managers used in both defining an initial scheduling and during (re)scheduling, where the latter is called for when an unplanned event occurs. The interactions of these agents in the proposed game, through decentralized mechanisms, provide an easier way to obtain a schedule flexibly adapted to the unplanned events mentioned above, where their decisions depend only on the negotiations that they carry out. To respond to abnormal disturbances, the agents must incorporate learning capabilities for (re)scheduling when choosing alternative courses of action and gaining experience from the situations that may arise. In the next section, *learning by trial-and-error* [9] using decoupled rules is incorporated in the proposed simulation model to advantage. The incorporation of decoupled learning rules to solve the multi-project rescheduling problem allows selecting at each strategic interaction among agents those actions that present a greater benefit to achieve agent's goals, according to their payoff functions. More importantly, the resulting repaired schedule corresponds to near Nash equilibrium for all concerned agents.

3.2 Learning by Trial and Error

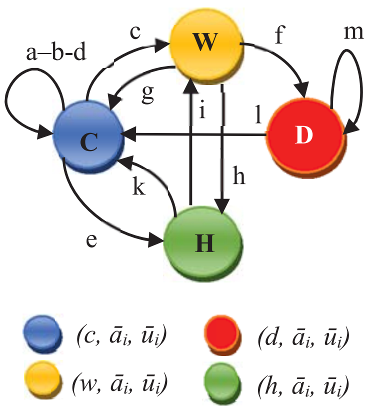
One of the distributed learning techniques having as its main characteristic resorting to uncoupled learning rules for each individual agents is the proposal of Young et al. [9]. In this learning by trial-and-error approach, interacting agents respond to changes in their own rewards, which are affected, indirectly, by other agents' actions. Uncoupled learning rules can be implemented in environments where the agents cannot observe what the other agents might be doing. That is why this learning method has great potential for distributed optimization problems and complex adaptive systems involving many autonomous agents interacting strategically.

The trial and error learning technique is incorporated in each agent to obtain the best course of action to be followed through a simple implementation at each decision stage in the repeated game (presented in previous sub-section) to solve the multi-project (re)scheduling problem and to obtain a feasible schedule. Each interacting agent's state (z_i) is made up of a type of *mood* m_i (which can be *content*, *discontent*, *hopeful* or *watchful*), a reference action ($\bar{a}_i$) and a reference reward ($\bar{u}_i$). According to his mood and the obtained reward, each agent decides its next action to be selected. In Table 3, the properties for the implementation of decoupled learning rules in each agent are shown. The parameter ε defines the rate of exploration in the next game stage.

Table 3. Agent properties related to *Learning by Trial and Error* rules.

Property	Description
$z_i = (m_i, \bar{a}_i, \bar{u}_i)$	Reference state of agent.
$z_i = (m_i, a_i, u_i)$	Current state of agent.
ε	The exploration/exploitation rate of each agent

At the end of each game stage, once all the agents define their actions to follow, they simultaneously collect the stage payoffs according to the selected actions and agents' state transitions occur. The transitions between the different states that an agent can experiment during the simulation of the agent-based model are depicted in Fig. 1. The transitions "a" to "k" depends on agents' actions and payoffs only, while transitions "l" and "m" depends on a probability function called *response* function, which is monotonically increasing in u_i and monotonically decreasing in $\bar{u}_i$ for project agents, and conversely for resource agents.



Id	Initial State	Transition conditions	Resulting state
a	$(c, \bar{a}_i, \bar{u}_i)$	$a_i \neq \bar{a}_i$, $u_i(a) \leq \bar{u}_i$	$(c, \bar{a}_i, \bar{u}_i)$
b		$a_i \neq \bar{a}_i$, $u_i(a) > \bar{u}_i$	$(c, a_i, u_i(a))$
c		$a_i = \bar{a}_i$, $u_i(a) < \bar{u}_i$	$(w, \bar{a}_i, \bar{u}_i)$
d		$a_i = \bar{a}_i$, $u_i(a) = \bar{u}_i$	$(c, \bar{a}_i, \bar{u}_i)$
e		$a_i = \bar{a}_i$, $u_i(a) > \bar{u}_i$	$(h, \bar{a}_i, \bar{u}_i)$
f	$(w, \bar{a}_i, \bar{u}_i)$	$a_i = \bar{a}_i$, $u_i(a) < \bar{u}_i$	$(d, \bar{a}_i, \bar{u}_i)$
g		$a_i = \bar{a}_i$, $u_i(a) = \bar{u}_i$	$(c, \bar{a}_i, \bar{u}_i)$
h		$a_i = \bar{a}_i$, $u_i(a) > \bar{u}_i$	$(h, \bar{a}_i, \bar{u}_i)$
i	$(h, \bar{a}_i, \bar{u}_i)$	$a_i = \bar{a}_i$, $u_i(a) < \bar{u}_i$	$(w, \bar{a}_i, \bar{u}_i)$
j		$a_i = \bar{a}_i$, $u_i(a) = \bar{u}_i$	$(c, \bar{a}_i, \bar{u}_i)$
k		$a_i = \bar{a}_i$, $u_i(a) > \bar{u}_i$	$(c, \bar{a}_i, u_i(a))$
l	$(d, \bar{a}_i, \bar{u}_i)$	prob $\phi(u_i(a), \bar{u}_i)$	$(c, \bar{a}_i, u_i(a))$
m		prob $1\text{-}\phi(u_i(a), \bar{u}_i)$	$(d, \bar{a}_i, \bar{u}_i)$

Fig. 1. State transitions experienced by agents using decoupled learning rules.

4 Computational results

This section presents and discusses results obtained through different simulations of the Multi-agent model for a Project-oriented Fractal Organization (MAS-MPR) developed to solve the multi-project (re)scheduling problem in enterprise networking. These results allow assessing the response and performance of the MAS-MPR. The Multi-agent model was implemented in **Netlogo**, a multi-agent simulation environment for agent-based modeling that allows generating emergent behaviors resulting from on-going interactions among autonomous, learning agents [13]. As a consequence of the initial concurrent projects scheduling received as input to the multi-agent model, the global (re)schedule of such projects is obtained in an autonomic and decentralized manner, so as to accomplish the scheduling goals defined in *Section 2*. The multi-agent model presented in this work is based on the prototype presented in [14]. This prototype has been extended to incorporated new goals based on additional project constraints such cost and resource leveling (since only the project duration is considered in [14]), and to eliminate assumptions defined during the initial implementation, such as that a resource can only process one task at a time.

Instances of multi-project problems with different features are used to test the proposed multi-agent system (Table 4). These problem instances are available in http://www.mpsplib.com. To adapt the problem instances to the simulation problem, different levels (L1, L2, L3) are defined for each problem, describing the number of project agents (managing either a project, sub-project or task) that interact at each level of fractal hierarchy. Then, four resource agents are available for each project, divided into shared (G) and local (L). In the scheduling phase of the multi-agent simulation model, 10 games per problem instance were considered, and 10 stages for each game are defined. Thus, the overall number of experiments performed in this work is 400.

Table 4. Multi-project problem instances

Set of Instances	L1	L2	L3	N. resources	
				G	L
MPJ30_a2	1	2	30	1	3
MPJ30_a5	1	5	30	1	3
MPJ30_a10	1	5	30	2	2
MPJ30_a20	1	5	30	2	2

In the virtual simulated world implemented using Netlogo, each estimated project schedule is graphically depicted by means of a Gantt diagram, on which changes will be made according to each agent action during interactions (Fig. 2). At each stage, agent decisions define the multi-project schedule, which is recorded as output of such stage. The MAS-MPR simulated response and performance can be evaluated through domain indicators (project total duration, project cost, leveling resources) based on obtained agent's payoffs after simulation execution, which can be utilized by project leaders to assess the multi-project (re)scheduling problem generated by unplanned events, thus providing a tool for find a solution in which all agents have no incentives to deviate, hence is a Nash equilibrium.

The scheduling efficiency for each project in the solution generated by the MAS-MPR is evaluated using a measure of performance defined as *Average Project Delay* (APD) [5], which in this work is calculated as follows:

$$APD = \frac{1}{N}\sum_{i=1}^{N}(ft_i - edd_i) \tag{4}$$

where ft_i is the finalization time of the latest task in the generated schedule for project i, edd_i is the estimated due date for project i (considered as the critical path for the project), and N represents the number of game stages.

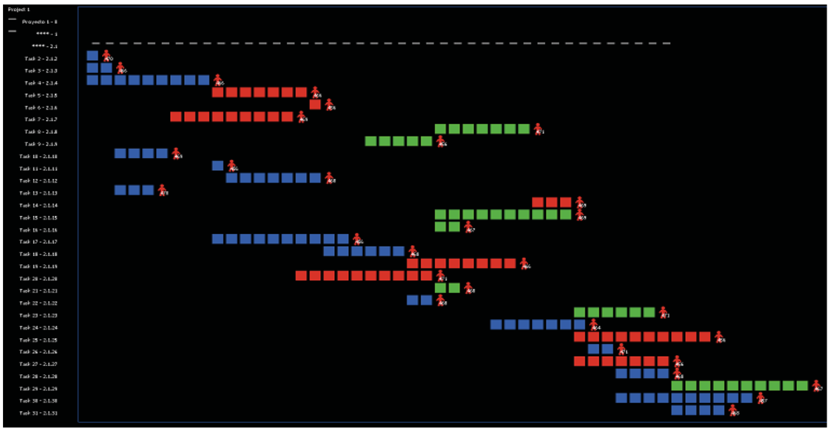


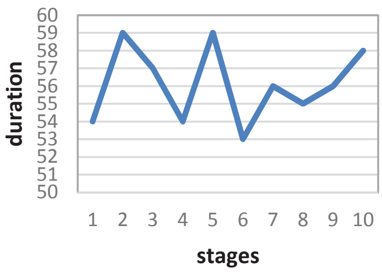
Fig. 2. Example of a project schedule obtained of MAS-MPR simulation as result of one stage.

The APD values obtained through the simulation of the MAS-MPR model are shown in Table 5, together with the values for the different proposals presented in [5]. The project leader can analyze the solutions provides by the MAS-MPR, and assess each problem instance to obtain the best solution, i.e., the schedule that presents the minimum APD value.

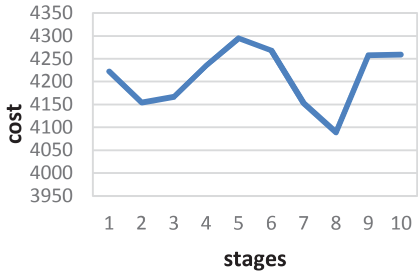
Table 5. Results obtained by different proposals

Set of Instances	MAS-MPR	CMAS/ES	CMAS/SA	DMAS/ABN	MAS/CI
MPJ30_a2	10,59	13,8	12,8	15,9	19
MPJ30_a5	14,82	18,52	19,28	21,2	27
MPJ30_a10	69	84,4	99,3	87,5	84,14
MPJ30_a20	154	198,4	223,95	207,96	182,05

The obtained results in Table 5 vividly demonstrate the advantages of incorporating learning to the interacting agents in the resolution of the multi-project (re)scheduling problems, comparing against the results obtained in other proposals. From Table 5 it is seen that the MAS-MPR obtains better results for the four subsets of presented problems.These results correspond to the objective evaluation defined as minimization of the project total duration. Furthermore, the MAS-MPR provides other results to evaluate the different solutions obtained. In Fig. 3, the project total duration and costs after a game simulation is shown. To facilitate calculating the costs of the resulting project schedule, different processing costs were assigned to the shared resources. Thus, the project leader can choose the solution that is closest to the estimated duration and cost. For example, the global schedule in the *stage 6* presents the minimum duration, whereas in the *stage 8* presents the lowest cost. In addition, if the stage 2 is analyzed, it is observed that the cost of the project decreases, but the total duration increases. This represents the situation in which project agents are selecting the most expensive resources. Thus, the project leader can analyze these two variables, namely time and cost, and then choose the generated schedule that fulfills the restrictions of its project.



(a)Multi-Project Total Duration in a game



(b)Multi-Project Total Cost in a game

Fig. 3. Project indicators in a simulated game of MAS-MPR.

It is worth noting that when using the MAS simulation model, computational times to find the solution to the multi-project (re)scheduling problem allows evaluating MAS-MPR response time in seconds. In this work, the average of such time is obtained from the following relation (where M is the number of stages played in a game):

$$AACT = \frac{1}{M} \sum actual\_clock\_time \tag{5}$$

To obtain a feasible schedule, the required real time (in seconds) changes from one problem instance to another. The average time simulation for each instance simulated in this work is shown in Table 6. The MAS-MPR also can carry out the multi-project (re)scheduling in similar simulation times to those used during the scheduling phase. The simulations were executed on a computer Intel Core I5 (2.5GHz, 8GB RAM).

Table 6. Average actual clock time for a simulated stage on MAS-MPR.

Instance	*AACT* (per stage)
MPJ30_a2	11 seconds
MPJ30_a5	70 seconds
MPJ30_a10	255 seconds
MPJ30_a20	720 seconds

5 Conclusions and Future Work

Multi-project (re)scheduling is considered a critical problem for organizations, mainly in those that share resources. Its importance is due to the inclusion of many resources, tasks, precedence constraints, unforeseen events, etc. In this context of high uncertainty and dynamism, project leaders claim for tools to evaluate alternative repaired schedules on a short notice, allowing the analysis of possible scenarios before making decisions. These scenarios can include unavailable resources, new tasks, technological complexity that causes delays, among other unplanned events that can generate project abnormal situations. In this work, the multi-project (re)scheduling problem is solved through a multi-agent model of the project-oriented fractal organization that decreases the inherent environment complexity when unplanned events occur. This proposal through the agent-based simulation aims to provide different indicators that makes room for the project leader to evaluate project (re) scheduling alternatives, taking into account unplanned events that may arise during its project execution. Thus, multi-project (re)scheduling in real time using the proposed MAS, where feasible schedules emerge from simple learning mechanisms, permits an early analysis of different (re) scheduling solutions before the total project execution. Therefore, agents do not need to know the complete status of the project schedule, nor the actions taken by other agents. The results obtained highlight indicators that permit the analysis of the project total duration, cost, leveling resources, etc., and provide a quality evaluation of the MAS generated schedule. Current research work is about incorporating the multi-agent simulation models in project management tools with an user-friendly interface supporting the decision-making process and allowing training activities for project leaders that favor seeking a solution to solve the problem of project (re)scheduling in the framework of game theory and multi-agent learning.

References

1. Garey, M., Johnson, D. Computers and Intractability: A Guide to the Theory of NP Completeness. New York: W. H. Freeman (1979).
2. Project Management Institute, Guía de los Fundamentos para la Dirección de Proyectos (Guía del PMBOK), 4ª ed. (2008).
3. Warnecke, H.J. The Fractal Company: A Revolution in Corporate Culture, Springer-Verlag, Berlin (1993).
4. Canavesio, M., Martinez, E. Enterprise modeling of a project-oriented fractal company for SMEs networking, Computers in Industry, vol. 54, pp. 794–813 (2007).
5. Adhau, S., Mittal, M., & Mittal, A. A multi-agent system for distributed multi-project scheduling: An auction-based negotiation approach. Engineering Applications of Artificial Intelligence, vol. 25, pp. 1738–1751 (2012).
6. Bodea, C., Badea, I., Mogos, R. A Multi-Agent System with Application in Project Scheduling. Management & Marketing, vol. 6, No.4, pp. 573-590 (2011).
7. Araúzo, J., Pajares, J., López-Paredes, A. Gestión eficiente de carteras de proyectos. Dyna Ingeniería e Industria, vol. 84 (9), pp. 761-772 (2009).
8. Chaos Report. Standish Group. In https://www.infoq.com/articles/standish-chaos-2015, last access 07/07/2017.
9. Young, H.P., Learning by trial and error-Games and Economic Behavior, vol. 65, pp. 626-643(2009).
10. Homberger, J., A (μ, λ)-coordination mechanism for agent-based multi-project scheduling. OR Spectrum (2009).
11. Artigues, C., Demassey, S., Néron, E. Resource-constrained project scheduling- Wiley Online Library (2008).
12. Shahsavar, A., Najafi, A.A.,Niaki, S.T.A. Three self-adaptive multi-objective evolutionary algorithms for a triple-objective project scheduling problem. Computers & Industrial Engineering, Elsevier, vol. 87, pp. 4-15 (2015).
13. Railsback, S., Grimm, V. Agent-Based and Individual-Based Modeling: A practical Introduction, Princeton University Press (2012).
14. Tosselli, L., Bogado, V., Martínez, E. Un Enfoque de Sistemas Multiagente para la Gestión Ágil de Riesgos en la Compañía Fractal Mediante la (Re) Planificación de Proyectos. Conaiisi 2015, Bs. As., Argentina (2015).