Available online at www.sciencedirect.com
ScienceDirect
Robotics and Computer-Integrated Manufacturing 24 (2008) 625–634
Robotics and Computer-Integrated Manufacturing
www.elsevier.com/locate/rcim

A holonic approach to dynamic manufacturing scheduling

Paulo Leitão[a,*], Francisco Restivo[b]

[a]Polytechnic Institute of Bragança, Quinta Sta Apolónia, Apartado 1134, 5301-857 Bragança, Portugal
[b]University of Porto, Faculty of Engineering, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal

Received 29 November 2006; received in revised form 7 April 2007; accepted 24 September 2007

Abstract

Manufacturing scheduling is a complex combinatorial problem, particularly in distributed and dynamic environments. This paper presents a holonic approach to manufacturing scheduling, where the scheduling functions are distributed by several entities, combining their calculation power and local optimization capability. In this scheduling and control approach, the objective is to achieve fast and dynamic re-scheduling using a scheduling mechanism that evolves dynamically to combine centralized and distributed strategies, improving its responsiveness to emergence, instead of the complex and optimized scheduling algorithms found in traditional approaches.

Keywords: Manufacturing scheduling; Intelligent production control systems; Holonic manufacturing systems; Dynamic re-scheduling

1. Introduction

Manufacturing scheduling can be defined as the allocation, over the time, of jobs to machines, within a short temporal horizon and according to a specific criterion, such as cost or tardiness. It is a complex combinatorial problem, more specifically a non-polynomial (NP) problem: the objective is to find the optimal sequence from the $j^{!m}$ possible scheduling sequences, where j is the number of jobs and m the number of machines. The manufacturing scheduling problem becomes even more complex when it takes place in an open, distributed and dynamic environment, where changes in the number of jobs or machines can occur at any time.

The scheduling problem has been widely studied, mainly due to its highly combinatorial aspects, its dynamic nature and its applicability in manufacturing systems [1], and many scheduling methods have been developed, based on different techniques such as heuristics, linear programming, constraint satisfaction techniques, Lagrangian relaxation, neighbourhood search techniques (e.g. simulation annealing or taboo search) and genetic algorithms.

Manufacturing scheduling is traditionally elaborated in a centralized manner using one of referred methods, often calculated off-line and considering that it is a static and deterministic problem. However, in an industrial manufacturing system the things rarely go as expected, mainly because

(i) new tasks arrive continuously to the system, while scheduled tasks are cancelled,
(ii) certain resources become unavailable and additional resources are introduced,
(iii) unexpected events occur in the system, such as machine failures, operator absence, rush orders or unavailability of raw-materials, and
(iv) scheduled tasks may take more or less time than expected.

In these dynamic environments, the optimized schedule produced by the front office can quickly become unacceptable, requiring dynamic re-scheduling, as fast as possible, and done in a short amount of time, to avoid the risk of degradation of the production productivity. Traditional methods do not fulfill the real dynamic re-scheduling needs because they tend to be inflexible and slow.

*Corresponding author. Tel.: +351 273303003; fax: +351 273313051.
E-mail addresses: pleitao@ipb.pt (P. Leitão), fjr@fe.up.pt (F. Restivo).

0736-5845/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.rcim.2007.09.005

Agent-based and holonic approaches to manufacturing[1] approaches suggest the implementation of distributed scheduling, where the scheduling algorithm is distributed over a number of entities, which combine their calculation power and local knowledge to optimize the global performance [2]. Unlike in traditional manufacturing scheduling, using a centralized scheduler, in agent-based manufacturing scheduling systems each agent can locally handle the schedule of its machine, operator, robot or station. The major advantages of the distributed scheduling are the improvement of reaction to disturbances and the parallel computation.

Some of the distributed scheduling approaches use traditional algorithms embedded in distributed entities, while others are based on emergent behaviour, like market-based and net protocol algorithms. Among others, Yet Another Manufacturing System (YAMS) [3] applies a contract net technique to a hierarchical model of manufacturing system, including agents to represent the shop floor. Autonomous Agents at Rock Island arsenal (AARIA) encapsulates manufacturing components as autonomous agents focusing on the dynamic scheduling, dynamic reconfiguration and on the control of manufacturing systems that fulfill the delivery dates [4]. CORTES [5] uses micro-opportunistic techniques to solve the scheduling problems and constrained heuristic search techniques for the decision making related to the scheduling. Sousa and Ramos [6] propose a dynamic scheduling system supported by a holonic approach, using forward and backward influence in the negotiation leading to the task allocation, to handle the temporal constraints and to solve conflicts, and Gou et al. [7] present a holonic manufacturing scheduling approach using Lagrangian relaxation, where capacity constraints of a scheduling problem can be relaxed and replaced by a penalty cost. Lin and Solberg [8] use a market-based control model to implement resource allocation and distributed scheduling, based on multiple-step negotiation, allowing the real-time coordination of agents. Markus et al. [9] propose a market model to solve dynamic order processing and scheduling problems, and Sugimura et al. [10] model the manufacturing operations using an object-oriented approach and propose a real-time scheduling mechanism for assembly lines. Hino and Moriwaki [11] introduce a recursive propagation technique based on sending messages regarding schedule changes to agents responsible for subsequent tasks, and Logie et al. [12] extend this concept by limiting the focus of the agents to tasks within a specified time window. Rabelo and Camarinha-Matos [13] propose a multi-agent-based dynamic scheduling, supported by negotiation techniques and the dynamic formation of consortia of manufacturing resources. Heikkilä et al. [14] propose a holonic approach for manufacturing scheduling and control in a manufacturing cell.

The motivation of the work reported in this paper is to use simple scheduling algorithms embedded in distributed entities, supported by adaptive and dynamic mechanisms, which increase the system performance in industrial scenarios characterized by the need for fast scheduling solutions. This paper describes a holonic approach to the dynamic manufacturing scheduling, introduced by adaptive holonic control architecture (ADACOR) for distributed manufacturing systems [15], which combines distributed decision-making embedded in each distributed holon, to achieve fast response to change, with coordination entities, to allow global optimization.

The rest of the paper is organized as follows: Section 2 presents the main principles of the proposed holonic scheduling architecture, focusing on the distributed components, dynamic adaptation model and distributed scheduling. Section 3 describes the cooperation mechanisms to support the dynamic re-scheduling and Section 4 describes the prototype implementation. Finally, Section 5 rounds up the paper with conclusions.

2. Holonic dynamic scheduling architecture

The idea beyond our scheduling approach is that a global optimized schedule should be used whenever possible, and a fast re-scheduling solution should be used in case of disturbances, because this is preferable to waiting a significant amount of time for the optimized schedule, which is not likely to be optimized again soon. The proposed dynamic scheduling is based on the following main foundations:

- *Distributed approach*, with decision-making distributed by a community of autonomous entities, each one having a partial knowledge of the problem.
- *Scheduling engines embedded in distributed holons*, both in the low-level and coordinating entities.
- *Dynamic adaptation mechanism*, allowing the evolution of the overall system in order to combine centralized and distributed scheduling strategies.

The next sections detail the main concepts of the proposed holonic dynamic scheduling architecture.

2.1. Holonic manufacturing components

ADACOR holonic control architecture [15] is based on a community of distributed and autonomous control entities, the holons, representing the manufacturing components. Three types of holons are identified to handle the scheduling and control at shop floor level [15], as illustrated in the Fig. 1:

[1]Holonic Manufacturing Systems (HMS) ⟨http://hms.ifw.uni-hannover.de/⟩ translates to the manufacturing world the concepts developed by Arthur Koestler for living organisms and social organizations. Holonic manufacturing is characterized by holarchies of holons (i.e., autonomous and cooperative entities), which represent the entire range of manufacturing entities. A holon, as Koestler devised the term, is a part of a (manufacturing) system that has a unique identifier, may be made up of subordinate parts and, in turn, can be part of a larger whole.

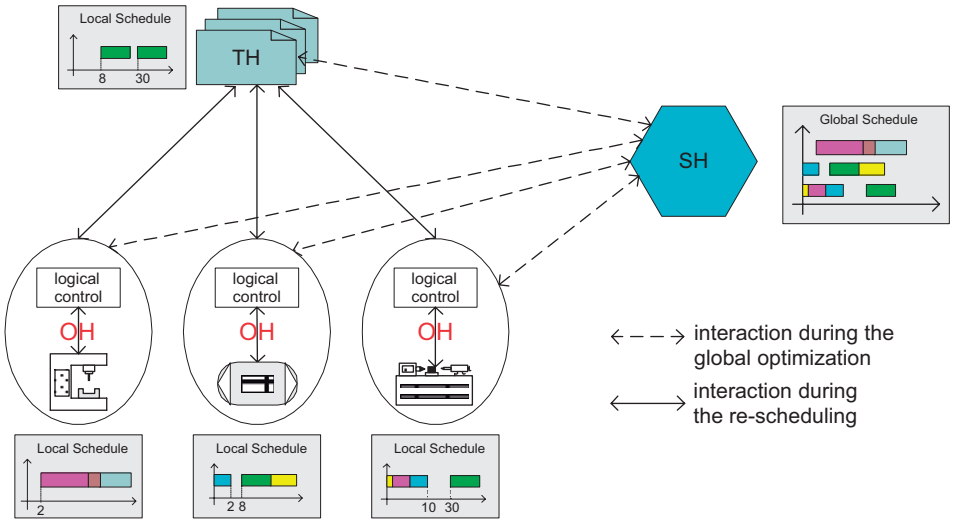

Fig. 1. Interaction between distributed manufacturing components.

- Task holons, representing production orders launched on the shop floor, and containing information about the production of the product, and about the progress of the production order execution.
- Operational holons, representing physical resources or operators available at shop floor, each one with a set of skills and knowledge.
- Supervisor holons, representing the logical coordination of a group of operational and/or supervisor holons, providing co-ordination and optimization services to the holons under their supervision, and thus introducing hierarchy in an otherwise decentralized system.

Each holonic control unit has decision-making capabilities, performing, among others, control and scheduling functions. These embedded mechanisms are dependent on the holon type, its behaviour and objectives.

On one side, the scheduling mechanism embedded in a supervisor holon deals with the multiple machines and multiple jobs scheduling problem, may take a large computational effort and time, but it is an optimal plan since the supervisor has a global view of the system.

On the other hand, each operational holon deals with the relatively simple problem of scheduling multiple jobs for a single machine, based on its local knowledge, but due to the lack of global information, the overall result may not be an optimal global schedule.

The motivation for ADACOR holons to execute the manufacturing actions is regulated by a credits system. When it is launched, each task holon receives a fund (π) to execute a production order and a penalty value for the delay. The task holons manage the fund received, trying to maximize its final amount.

During the interaction to allocate the operations, task holons try to pay as less as possible and the operational holons try to receive as much as possible. After the negotiation, each task holon agrees to pay a price of ξ credits to the operational holon that will execute a certain operation and to receive a penalty of φ credits from the operational holon if it does not fulfill the contracted due date.

Table 1
Evolution of credits during the holon life cycle

Phase	Task holon	Operational holon
Resource allocation process	Contracts the execution by ξ and the penalty by φ	Contracts the execution by ξ and the penalty by φ
Finish of an operation with success	Pays the value ξ ($\pi \leftarrow \pi - \xi$)	Increases its credits by ξ ($\mu \leftarrow \mu + \xi$)
End of an operation with delay	Pays the value ξ and receives the value φ ($\pi \leftarrow \pi - \xi + \varphi$)	Decreases its credits by φ and increases by ξ ($\mu \leftarrow \mu + \xi - \varphi$)
Operation cancelled (delay, failure, etc.)	Receives the value φ ($\pi \leftarrow \pi + \varphi$)	Decreases its credits by φ ($\mu \leftarrow \mu - \varphi$)

Table 1 summarizes the evolution of the credits of task and operational holons during their life cycles.

The global performance of operational holons in terms of credits is given by the sum of rewards received minus the sum of penalties paid for the delays. These rewards and penalties reflect the reputation of the holon.

2.2. Dynamic scheduling model

The dynamic scheduling model is the result of the dynamic interaction between task, operational and supervisor holons.

At the operation level, the interaction process leading to the achievement of the manufacturing schedule has the following constraints:

- A part cannot be started before its preceding part(s) is finished.
- An operation cannot be started before its preceding operations are finished.

- Each machine can only process one operation at time t.
- A resource R_j possessing the set of skills S_j, has abilities to execute the operation o_{ik}, having a list of requirements $B_{ik} = \{B_{ikz} | z \in I\}$, if

$$B_{ik} \subseteq S_j \Leftrightarrow \forall x : x \in B_{ik} \rightarrow x \in S_j,$$

i.e., the resource has abilities to execute an operation if it fulfils all the requirements presented by the operation.

The self-organization capability of each ADACOR holon is the key concept to support the adaptive production control and scheduling mechanism. The self-organization is regulated by the *autonomy factor*, which fixes the level of autonomy of each holon, and evolves dynamically in order to adapt the holon behaviour to the changes in the environment where it is placed. The evolution is governed by a decision mechanism, and the overall efficiency of the self-organization is dependent on how the learning mechanisms are implemented, and on how new knowledge influences its parameters.

Fig. 2 illustrates a small example about how the adaptive and dynamic scheduling approach works. In normal operation, i.e. without the occurrence of unexpected disturbances, the holons are running in a hierarchical structure, with supervisor holons coordinating several operational and/or supervisor holons, and operational holons having a low autonomy factor. Periodically, regulated by their internal clocks, supervisor holons generate globally optimized manufacturing scheduling plans.

The optimized schedule plans are offered, as advices, to the holons under their coordination domains, which have the capability to accept or reject them. Normally, they follow the schedule advices proposed by the supervisor holons since they have a low level of autonomy.

In turbulent scenarios or once an unexpected disturbance (e.g. a machine failure or deviation from plan) is detected, the autonomy factor of each ADACOR holon is increased dynamically according to a function that takes in consideration its current value, the estimated time to recover from the disturbance, and the level of impact of the disturbance. This way, the system is forced to evolve to a heterarchical structure, characterized by totally decentralized decision-making mechanisms.

During this transient state, which should be as short as possible, the system operates without the presence of coordination levels, the manufacturing scheduling being achieved in a distributed manner. The distributed scheduling results solely from the interaction between task and operational holons. The cooperation strategy built into each holon is therefore the key to the success of this approach. In the example illustrated in Fig. 2, two operations that are allocated to a broken resource are reallocated to other similar resources.

After the transient phase, the autonomy factors are reduced, the system evolving to a new control structure

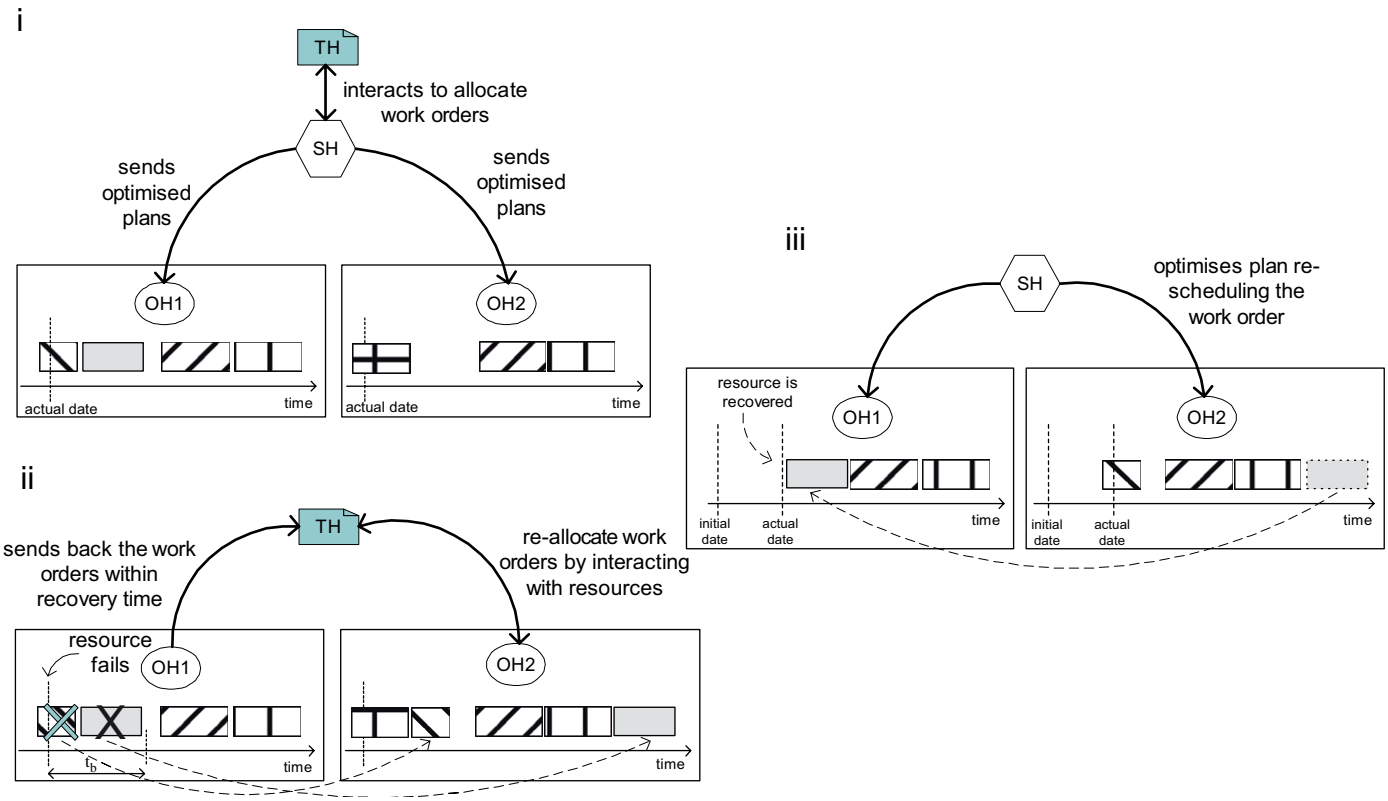

Fig. 2. Dynamic holonic scheduling. (i) Normal operation; (ii) reaction to disturbance; (iii) plan optimisation.

(often returning to the same hierarchical one), according to the knowledge of each holon. At this stage, supervisor holons enter again in action and return to their function of optimizing the scheduling plan for the current jobs and machines.

2.3. Distributed resource allocation schema

The distributed scheduling mechanism, introduced in this work, uses a resource allocation mechanism based on a multi-round contract net protocol (CNP) [16], extending the original CNP schema with the capability to apply the contract net schema several times, and capability to contract partial quantities. The negotiation process is illustrated in Fig. 3.

In presence of operation announcements, each operational holon decides, based on its skills and capacity, its availability to execute the operation. In case of availability, the operational holon calculates the price to be proposed to the task holon. The price may be calculated according to the following function:

$$p_{jik} = C_s + C_p \times d_{ik} + C_b \times (2 - e^{-\sigma \times \beta} \times (1 - \gamma)),$$

which models the market laws, increasing or decreasing the final price in function of the actual load of the resource (reflected by the parameter β) and of the actual bid acceptance rate (reflected by the parameter γ, with $0 \leqslant \gamma \leqslant 1$). The holon uses the knowledge learned from the previous bids to adjust the final price: reducing the parameter γ if the acceptance rate is low or increasing it in the opposite case.

The task holon evaluates the proposals sent by operational holons to allocate the operation to the best bid. The decision procedure takes into account, among others, the proposed price, the location of the resource and the confidence degree about the holon. The confidence degree reflects the trust that the task holon has in an operational holon, and it is based on the knowledge learned from previous interactions. In case of an inconclusive evaluation, the task holon can start another iterative negotiation, reformulating the bid parameters, for example, the due date and announcement specifications.

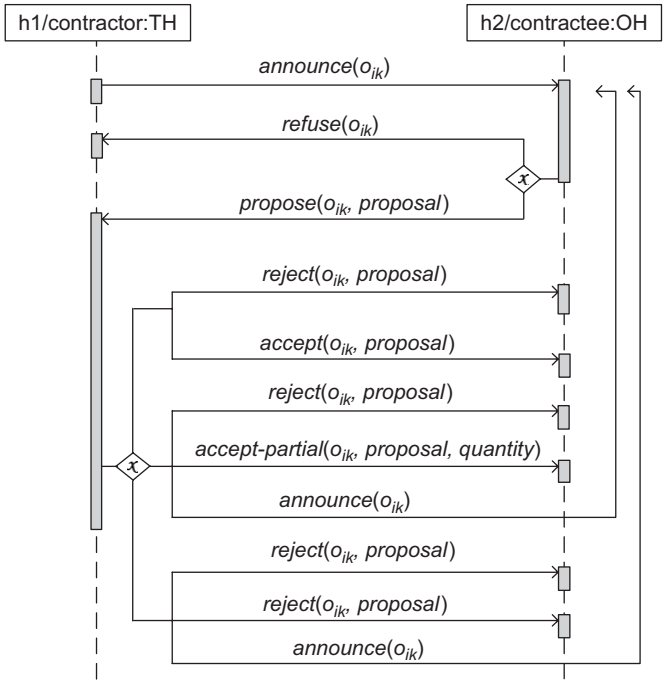

Fig. 3. Interaction sequence for the distributed scheduling.

3. Cooperation mechanisms for re-scheduling

The dynamic scheduling algorithm must respond dynamically and promptly to emergent and unexpected disturbances. An important design factor is the *size* of the disturbance that activates the rescheduling mechanisms. Re-scheduling mechanisms can be divided into

- *periodic re-scheduling*, which considers all disturbances (usually many small disturbances) at once, generating new optimized schedules periodically, and
- *event-based re-scheduling*, which is more suited for larger, single disturbances, like machine breakdowns or rush orders.

In the next sections, several types of re-scheduling mechanisms are described.

3.1. Re-scheduling for cancellation of orders

The cancellation of a production order can be considered a simple disturbance at shop floor level that requires only local re-scheduling, in order to optimize the schedule. After generating a new schedule, the operational holon notifies the supervisor holon about its new schedule, allowing the synchronization of both agendas and the optimization of the global schedule. It must be noticed that this type of disturbance may open free spaces in the agenda, allowing to execute earlier some operations that were eventually delayed.

The modification of the order attributes, such as the temporal window to execute the order, normally requires the re-scheduling of the related operations, trying to minimize deviations from the initial plan. In case of operations that cannot fulfill the due date in the new schedule, the operational holon notifies the task holon and waits for its agreement. In affirmative case, the schedule is confirmed, but otherwise, the operation is cancelled.

3.2. Re-scheduling for machine breakdowns

In the case of occurrence of a machine failure, the operational holon determines the state of the machine and of the part after the failure, and estimates how long the downtime will be. The diagnostic can lead to different scenarios: the machine can become immediately available or stay out of service for a longer repair intervention, and

the part may have been destroyed or not. If the part is not destroyed and the machine is ready to re-execute the operation, no action has to be performed; however, other scheduled operation(s) may become delayed, being then treated as a delay disturbance.

In the other cases, both operational and task holons have specific tasks to perform. The operational holon

- in case of destruction of the part, removes the proper operations from its agenda, and notifies the task and supervisor holons about the occurrence.
- in case of machine breakdown, notifies the task holon about its impossibility to execute the operations on the scheduled dates.

The operational holon also executes a re-scheduling to optimize the plan.

The task holon can take two different actions:

- If the part is destroyed, it re-allocates from the beginning all operations belonging to the production order.
- If the machine became unavailable, it re-schedules the operations taking into consideration the information from previous resource allocation processes. The achieved allocation can lead to delays in the posterior operations, requiring an adjustment of the temporal window to execute each operation.

In both cases, the re-scheduling is performed using the distributed resource allocation schema.

3.3. Re-scheduling for delays

An operation delay can occur after a disturbance, when the operational holon cannot fulfill the scheduled due date of an operation. In this situation, the operational holon notifies the task holon about the delay, proposing a new date. The decision about the acceptance of the operation delay is dependent of the actual state of the operation. If the operation is already in execution, this notification is seen as a warning of delay, being necessary to re-schedule all the posterior operations affected by the delay. If the operation is waiting for the execution, the task holon can try to find alternative resources to allocate the operation by asking other operational holons about their capacity to execute the operation.

Based on the proposals sent by the operational holons and on the estimated delay, the task holon decides if it accepts the proposal for the estimated delay or if it changes the allocation to another operational holon. In this case, the operational holon removes the operation from its local schedule, and triggers a local scheduling optimization procedure. Additionally, the task holon re-schedules the operations in its agenda, adjusting the scheduled start and due dates.

3.4. Re-scheduling for new rush orders

A rush production order is an order, usually of high priority, that arrives to the system and must be processed immediately, since it has a near due date. As the schedule plans are elaborated periodically by the supervisor holons, these orders cause a disturbance in the system.

In this situation, the rush task holon interacts directly with the operational holons to allocate its operations, using the distributed resource allocation mechanism. The problem appears if the rush production order has to be executed in a time window already occupied by other operations, which requires a special negotiation to relax these and to introduce new ones.

Since each order has an associated priority, operational holons take this information into consideration. In the case that the rush order has maximal priority, i.e. it must be executed as soon as the current operation is completed, the operational holon tries to re-schedule the operations that have lower priority, i.e. those operations that can probably be delayed beyond the due date without major loss, to find capacity to execute the rush operation.

In case of impossibility to find capacity to allocate the rush production order, the task holon needs to negotiate with the other task holons that have operations allocated to the resources occupying the requested time window. The task holons can use the trade of credits units for rewards and penalties for this purpose.

In the negotiation process, as illustrated in Fig. 4, the rush task holon interacts with other task holons, requesting the desired time window and offering a reward. Each one of the other task holons analyzes the offer and in case the reward covers the penalty that the task holon must pay for the delay, it accepts; otherwise, it rejects the proposal.

In case all task holons reject the reward, the rush task holon must increase the reward value and make another offer. The task holon should repeat this procedure until one task holon accepts the offer or the offered reward value reaches the maximum value, which is equal to the penalty to be paid in case of delay.

In case one task holon accepts the offer, it will notify the operational holons to decrease the priority to free the time window. In parallel, the rush task holon announces again the production order to the operational holons.

3.5. Re-scheduling for optimization

After the execution of an event-based re-scheduling, performed in a distributed manner, it is necessary to synchronize and optimize the global schedule. The synchronization is required because supervisor holons do not know what kind of schedule was achieved during the distributed re-scheduling. For this purpose, lower-level operational holons must notify the supervisor holon about their new schedule plans.

Supervisor holons start the optimization of the re-schedule plan achieved using the distributed scheduling

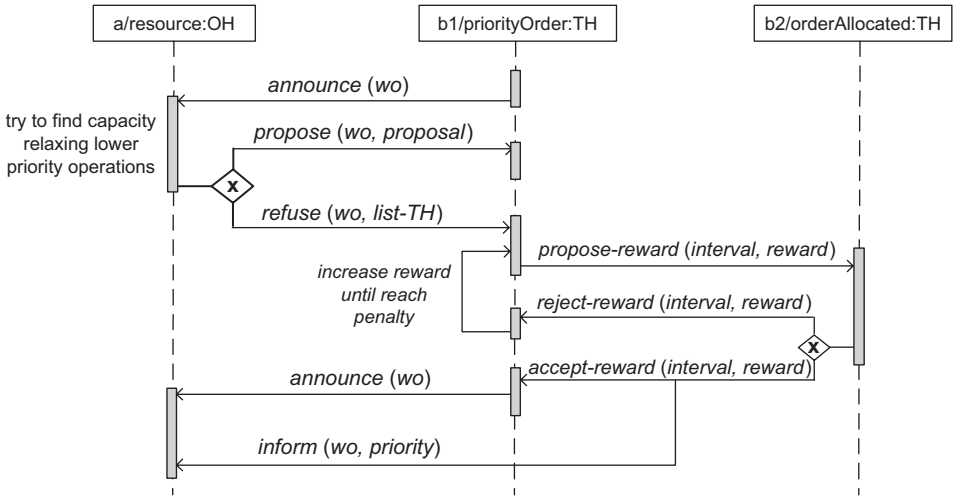

Fig. 4. Interaction protocol for the negotiation between task holons.

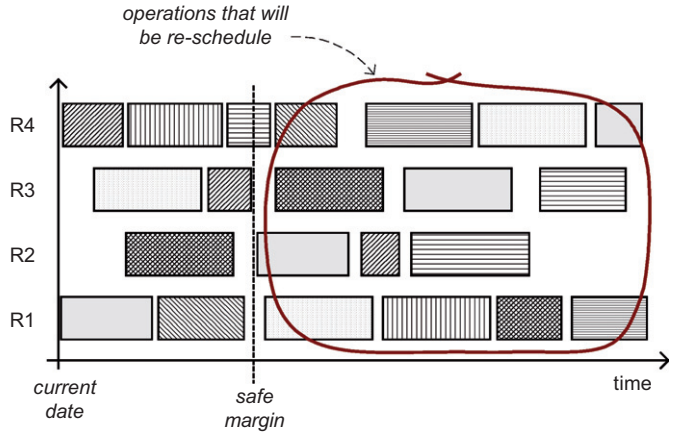

Fig. 5. Optimized re-scheduling after the distributed scheduling.

schema. The elaboration of this optimized re-scheduling is performed in background and does not consider the operations included in a safe time window, as illustrated in Fig. 5.

This safe time window guarantees that the current schedule plan can be executed in the factory plant during the elaboration of the optimized re-schedule and is defined according to the estimated time required to optimize the schedule plan.

4. Prototype implementation

The proposed holonic manufacturing scheduling approach was implemented in a prototype, using agent technology, namely the Java Agent Development Framework (JADE) framework. All three types of designed holons were implemented by extending the Agent Java class provided by the JADE framework. The communication between agents is performed by encoding messages according to the FIPA- Agent Communication Language) (ACL) communication language.

The decision component embedded in an ADACOR holon uses a rule-based system, applying declarative knowledge expressed in terms of rules, to regulate the holon behaviour. For this purpose, the Java Expert System Shell (JESS) rule-oriented programming infrastructure is used. The decision component also uses procedural knowledge, embodied in procedures that are triggered as actions by some rules. The scheduling algorithm is an example of this type of knowledge.

In the prototype, the scheduling mechanism embedded in the supervisor and operational holon uses simple algorithms that guarantee rapid and reliable scheduling. As the ADACOR architecture is built upon functional blocks, similar to Lego® components, these scheduling algorithms can be easily modified in the future, by plugging more powerful scheduling algorithms. The mechanisms to implement the distributed scheduling were developed in a similar way. The mechanisms for the propagation of re-organization using ant-based techniques and the factor of autonomy were also implemented.

The proposed holonic manufacturing scheduling approach was tested in a prototype for the flexible manufacturing system of CIM Centre of Porto, illustrated in Fig. 6, extended with two virtual manufacturing cells (cells B and C that do not exist in the real platform), to provide the hardware/software redundancy and flexibility for accommodating alternative solutions at the production planning level [17]. This redundancy in the manufacturing plant is needed to be taken advantage of using agile and reconfigurable systems.

This experimental case study considers the production of four (sub)products, named the base, body, cover and handle, also illustrated in Fig. 6, which, when assembled, can create two different final products: ashtray and box. The ashtray product comprises the assembly of the base and the body subproducts, and the box product comprises the assembly of the four subproducts.

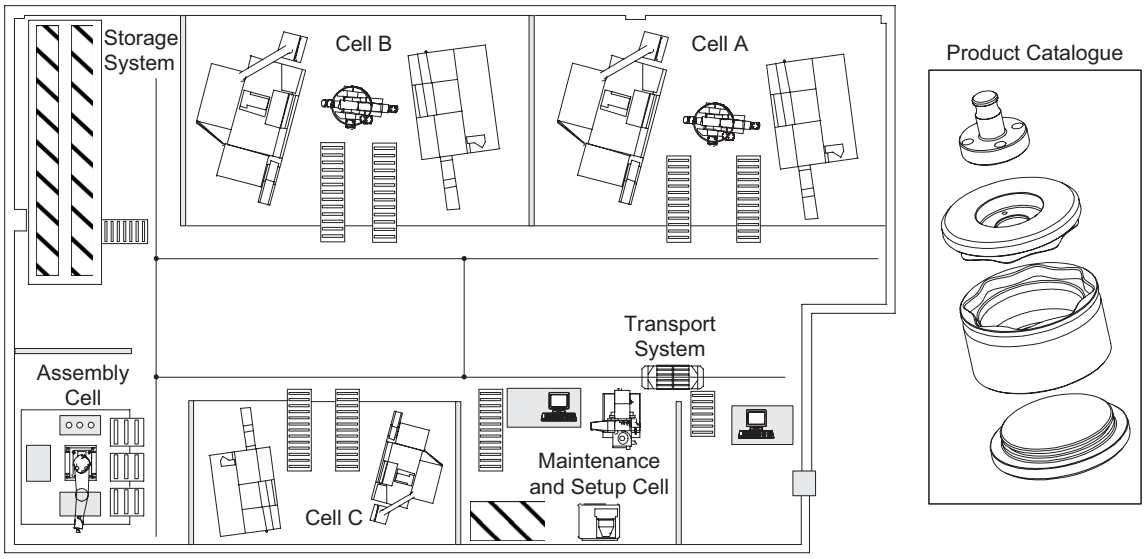

Fig. 6. Case study: production system.

The experiment considers three different plant scenarios: (i) the first plant scenario considers that no unexpected disturbance will occur, (ii) the second plant scenario considers the occurrence of failures in one turning machine (cell B), with a probability of 25%, and that, in case of failure, the part is destroyed and the machine is down for 60 s for the recovery procedures, and (iii) the third plant scenario considers the occurrence of failures in the turning machines of cells B and C, with the same disturbance model of the previous scenario.

Each individual book of orders comprises the production of 6 production orders: 2 bodies, 2 bases, 1 handle and 1 cover. The experimental tests consider different plant loads, for example, a book of orders comprising 18 production orders (involving 51 operations). All the production orders belonging to the same book of orders arrive to the production system at the same time, but different books of orders arrive sequentially to the production system.

Fig. 7 illustrates the system prototype that shows graphically the optimized schedule elaborated by a supervisor holon and the local schedule performed by an operational holon.

The prototype operation showed, in the first instance, the correctness and applicability of ADACOR control system, and particularity the holonic scheduling. It was also proved that the proposed holonic scheduling approach presents fast responsiveness and better flexibility, scalability and robustness, particularly for unexpected situations. A set of experimental results were presented in [17], and summarized in Fig. 8, where this approach was evaluated and compared with two other different control approaches, namely a heterarchical-like control approach similar to that presented by [8] and a hierarchical-like control approach using a supervisor entity similar to that presented by [13].

In order to guarantee a fair benchmarking, all the three referred control approaches used the same developed prototype platform: (i) in the hierarchical-like control approach, the holons are organized in a hierarchical control structure, using the supervisor holon to act as the shop floor controller, (ii) in the heterarchical-like control approach, the holons run on a completely decentralized control structure, without the presence of supervisor holons, and (iii) in the ADACOR control approach, the holons are organized in a hierarchical control structure, using the supervisor holon as the shop floor controller, but enabling the self-organization capability of the operational holons to support the agile re-organization of the control structure in case of emergency.

Fig. 8 illustrates some quantitative indicators, namely the lead time (i.e. the total time required to process a given product through the factory plant), the throughput (i.e. the ratio between the number of parts produced in the experience and the batch time necessary to execute the experience), the repeatability (i.e. the mean value of the standard deviation of the percentage of utilization of all resources of the system over the several experiences) and the tardiness (i.e. the difference between the order completion date and the due date when this difference is positive). Also, a qualitative indicator, the agility (i.e. the capability to react in a short period of time to the occurrence of unexpected disturbances), can be extracted from these experimental results, by obtaining the percentage of reduction of the throughput of the system running under a disturbance scenario in relation to the throughput of the system running in a scenario with no disturbances [17].

The results showed that the proposed approach has potential to improve the system performance, mainly combining agility (i.e. smaller lost of productivity

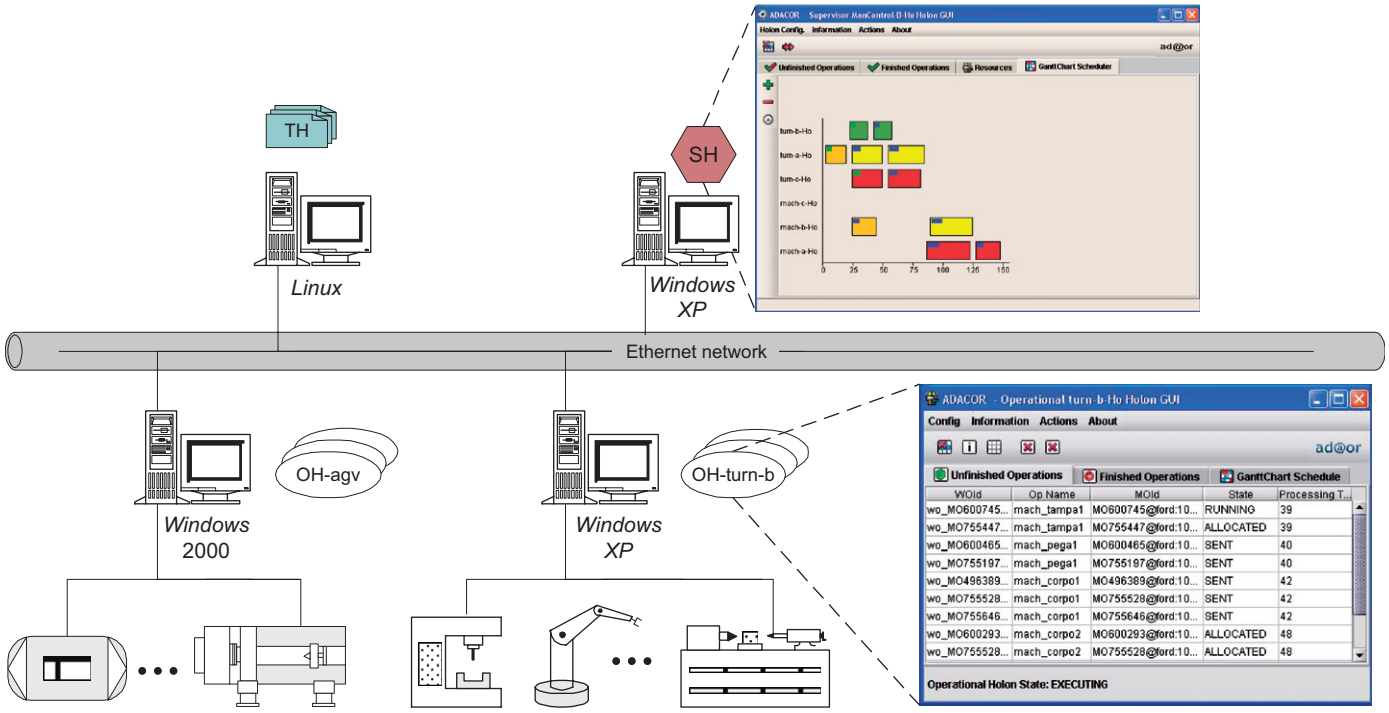

Fig. 7. The central and local schedules in the distributed ADACOR entities.

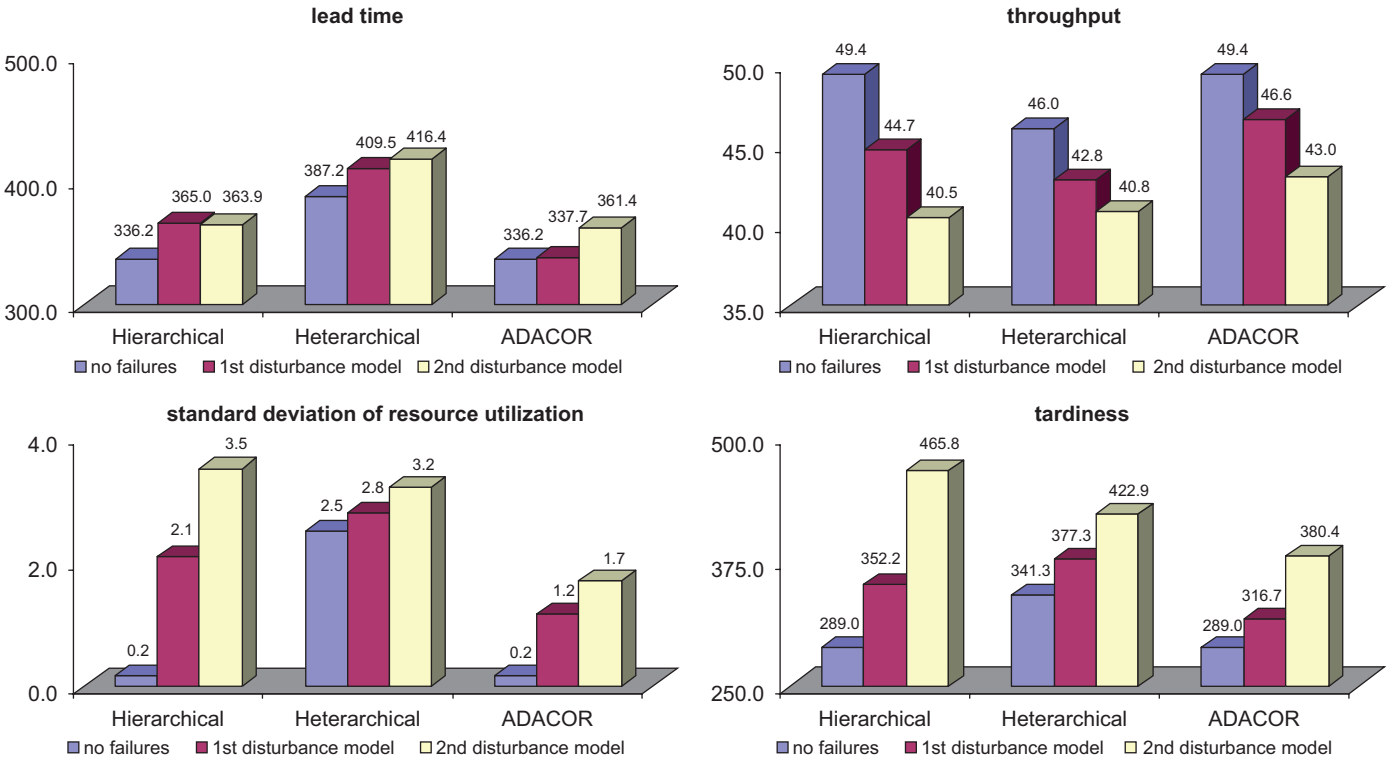

Fig. 8. Experimental results.

presented by ADACOR approach) and global production optimization (i.e. better results presented by ADACOR approach in terms of throughput, lead time and tardiness).

5. Conclusions

Manufacturing scheduling is traditionally elaborated in a centralized manner and doesn't consider dynamic

re-scheduling. This paper presented a holonic approach to dynamic manufacturing re-scheduling, combining the need of fast re-scheduling with the need to maintain global optimization. The architecture is based on the following main foundations:

- distributed approach, with decision-making distributed by a community of autonomous entities, each one having partial knowledge about the problem;
- in normal operation the scheduling is achieved in a central manner, using coordination entities to achieve optimization, and in abnormal situations, the scheduling is elaborated in a distributed manner aiming for fast re-scheduling;
- the dynamic adaptive mechanism allows the evolution of the overall system in order to combine centralized and distributed scheduling strategies.

At this stage, the objective is not to have complex scheduling algorithms but to achieve fast re-scheduling combined with global optimization, using simple local scheduling algorithms embedded in the holons. In further work, the embedded local scheduling mechanisms will be improved in order to achieve high quality scheduling in a timely fashion.

References

[1] Shen W. Distributed manufacturing scheduling using intelligent agents. IEEE Intell Syst 2002;17(1):88–94.
[2] Bongaerts L. Integration of scheduling and control in holonic manufacturing systems. Ph.D. Thesis: Katholieke Universiteit, Leuven, Belgium. 1998.
[3] Parunak HVD. Manufacturing experience with the contract net. Distributed artificial intelligence, vol. 1. Morgan Kaufmann; 1987. p. 285–310.
[4] Parunak HVD, Baker A, Clark S. The AARIA Agent Architecture: an example of requirements-driven agent-based system design. Proceedings of the first International Conference Autonomous Agents, ACM Press, New York, 1997, p. 482–3.
[5] Sadeh N, Fox M. CORTES: An exploration into microOpportunistic job-shop scheduling. Proceedings of the Workshop on Manufacturing Production Scheduling, Detroit, 1989.
[6] Sousa P, Ramos C. A distributed architecture and negotiation protocol for scheduling in manufacturing systems. Comput Ind 1999;38(2):103–13.
[7] Gou L, Luh P, Kyoya Y. Holonic manufacturing scheduling: architecture, cooperation mechanism and implementation. Comput Ind 1998;37:213–31.
[8] Solberg J, Lin G. Integrated shop floor control using autonomous agents. IIE Trans 1992;24(3):57–71.
[9] Markus A, Vancza T K, Monostori L. A market approach to holonic manufacturing. Ann CIRP 1996;45:433–6.
[10] Sugimura N, Hiroi M, Moriwaki T, Hozumi K. A study on holonic scheduling for manufacturing system of composite parts. Japan/USA Symposium on Flexible Manufacturing; 1996.
[11] Hino R, Moriwaki T. Decentralized scheduling in agent manufacturing system. Proceedings of the second international Workshop on Intelligent Manufacturing Systems, Belgium, 1999, p. 41–7.
[12] Logie S, Sabaz D, Gruver WA. Sliding window distributed combinatorial scheduling using JADE. Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics Netherlands, 2004, p. 1984–9.
[13] Rabelo R, Camarinha-Matos LM. Negotiation in multi-agent based dynamic scheduling. Robot Comput Integ Manuf 1996;7(4): 257–70.
[14] Heikkilä T, Jarviluoma M, Juntunen T. Holonic control for manufacturing systems: design of a manufacturing robot cell. Integ Comput Aided Eng 1997;4:202–18.
[15] Leitão P, Restivo F. ADACOR: a holonic architecture for agile and adaptive manufacturing control. Comput Ind 2006;57(2):121–30.
[16] Smith R. Contract net protocol: high-level communication and control in a distributed solver. IEEE Trans Comput 1980;C-29(12): 1104–13.
[17] Leitão P, Restivo F. Experimental validation of ADACOR holonic control system. In: Marík V, Brennan R, Pechoucek M, editors. Holonic and multi-agent systems for manufacturing, LNAI 3593. Springer; 2005. p. 121–32.