LNAI 10444

Vladimír Mařík · Wolfgang Wahlster
Thomas Strasser · Petr Kadera (Eds.)

Industrial Applications of Holonic and Multi-Agent Systems

8th International Conference, HoloMAS 2017
Lyon, France, August 28–30, 2017
Proceedings

Springer

An Agent-based Approach for the Dynamic and Decentralized Service Reconfiguration in Collaborative Production Scenarios

Nelson Rodrigues[1,2], Paulo Leitão[1,2], Eugénio Oliveira[2,3]

[1] Polytechnic Institute of Bragança, Campus Sta Apolónia, 5300-253 Bragança, Portugal, 5301-857 Bragança, Portugal, {nrodrigues, pleitao}@ipb.pt
[2] Artificial Intelligence and Computer Science Laboratory Porto, Portugal
[3] Faculty of Engineering - University of Porto, Rua Dr. Roberto Frias s/n, 4200-465 Porto, Portugal eco@fe.up.pt

Abstract. Future industrial systems endorse the implementation of innovative paradigms addressing the continuous flexibility, reconfiguration, and evolution to face the volatility of dynamic markets demanding complex and customized products. Smart manufacturing relies on the capability to adapt and evolve to face changes, particularly by identifying, on-the-fly, opportunities to reconfigure its behavior and functionalities and offer new and more adapted services. This paper introduces an agent-based approach for service reconfiguration that allows the identification of the opportunities for reconfiguration in a pro-active and dynamic manner, and the implementation on-the-fly of the best strategies for the service reconfiguration that will lead to a better production efficiency. The developed prototype for a flexible manufacturing system case study allowed to verify the feasibility of greedy local service reconfiguration for competitive and collaborative industrial automation situations.

Keywords: Service Reconfiguration, Service Reconfiguration Strategies, Multi-agent Systems, Cyber-Physical system, Smart Manufacturing, Industry 4.0.

1 Introduction

Manufacturing industry is facing a continuous evolution, being the implementation of systems exhibiting flexibility and reconfiguration capabilities one of the many challenges of the manufacturing industry in the coming years. Manufacturing companies must be able to react rapidly and cost-effectively to condition changes, in order to overcome the current problems [1]. Aligned with this vision, predictive maintenance plays an important and necessary role to ensure production efficiency although costs become significantly high [2]. To address the mentioned issues and aligned with the vision of Industry 4.0, the employment of reconfiguration mechanisms to dynamically adapt the needed processes and offered services is crucial. Traditionally, service reconfiguration is performed due to several reasons like, for example, to cope with the unexpected and unpredictable condition changes, to recover from broken processes, to lead to a better production efficiency, to improve the system competitiveness according to the costumer's needs and to respond to new business strategies.

In spite of the current research efforts, the existing reconfiguration strategies are still too much simple, e.g., components' replacement (to react to the harmful effects or breakdowns) and re-planning (to deal with the modified configurations' requirements) [3]. Additionally, traditional production systems are still lacking automated tools that support the dynamic and runtime reconfiguration strategies by discovering new reconfiguration opportunities and exploring new system configurations.

The majority of the deployed service reconfiguration solutions are manually and reactively executed taking into consideration a centralized perspective. In fact, the decisive actions for the system reconfiguration are made after the occurrence of a failure, which can sometimes involve stopping a running process, diagnosing the failure, reconfiguring and restarting the system/device. In practice, the usual behavior when diagnosing a failure is to select the known recovery action that solves the failure; other possibilities that take more time require the understanding of all possible alternative service configurations' solutions that go beyond the human capacity in an acceptable time. Therefore, performing the service reconfiguration manually and, afterwards, restarting the system is not enough to address the dynamics of current industrial needs [4]. As a consequence, new reconfiguration strategies are required to support the evolution of the traditional manufacturing systems by being dynamically performed online and just in time.

This paper describes a flexible and distributed multi-agent system (MAS) approach for the service reconfiguration that allows the pro-active and dynamic identification of the opportunities for reconfiguration and the implementation on-the-fly of the best strategies for the service reconfiguration that will lead to a better production efficiency. In the proposed approach, the distributed and autonomous agents embed intelligent mechanisms for the early detection of reconfiguration opportunities and the selection of the reconfiguration strategies for improving the service properties or updating the catalog of offered services. In order to avoid conflicts arising from the dynamic reconfiguration of the distributed agents, a collaborative service reconfiguration mechanism was introduced. Information sharing among agents support the identification of the best configurations and avoid redundancies or unnecessary reconfigurations that can lead to poorer system performance. The proposed approach was tested in an experimental flexible manufacturing system, and the preliminary results show the benefits of this dynamic and pro-active service reconfiguration.

The remaining of this paper is organized as follows. Section 2 overviews the main concept of service reconfiguration, existing related work and establishes the requirements for a truly dynamic, intelligent and pro-active service reconfiguration process. Section 3 presents the proposed multi-agent based approach for the service reconfiguration. Section 4 describes the experimental case study, the implementation details and discusses the preliminary experimental results. Finally, Section 5 rounds up the main contributions of the paper with the conclusions and the future work.

2 Related Work

The Service-oriented Architectures (SOA) paradigm [5] is based on the concept of offering and consuming services, each one encapsulating the functionalities of a service provider. The use of service-orientation allows facing interoperability and loose-coupled abstraction in the design of complex systems. In these systems, the concepts of service aggregation, composition, and orchestration are important, to better understand the service reconfiguration concept. Basically, service reconfiguration is related to service adaptation designed to deal with unexpected events, such as failure of a service and loss of the quality of service (QoS) [3].

In this context, several different types of service reconfiguration can be identified:

- *Improvement of the service's behavior and performance*, e.g., changing the calibrating tools and/or switching components of the process to reduce the service's time or improve the service's quality (this can be seen as a weak reconfiguration type).
- *Changing the services' catalog*, i.e. new services are included and others are removed from the catalog offered by an entity to better accommodate with the service demand; e.g., offering a new drilling service (this can be seen as a strong reconfiguration service type).
- *Changing the structure of a composed service*, which is built up through the composition of several atomic services, e.g., reorganizing the atomic services by adding some of them and remove others to better accommodate the evolution in the available atomic services (this can be seen as a strong-reconfiguration type).

Aiming to execute a truly dynamic, intelligent and pro-active service reconfiguration, considering the referred reconfiguration types, the following requirements need to be observed:

- R1: The opportunity to execute a service reconfiguration must be identified internally (regarding the system), automatically and in run time.
- R2: The system needs to have the capacity to select an alternative reconfiguration solution and reconfigure on-the-fly, reducing the perturbation impact.
- R3: Service reconfiguration must be performed in a smooth manner (i.e. avoiding the individual and/or system nervousness).
- R4: Service reconfiguration process should comply with both competitive and collaborative scenarios.

Part of the research that was conducted on the service reconfiguration domain resides in using service composition mechanisms which aims at composing the best service that meets the client's requirements. The process of dynamic service reconfiguration is done according to the following steps [5]:

a) Discover the services in the central service registry, e.g., UDDI (Universal Description, Discovery, and Integration), by selecting the services that match the requirements.
b) Select the optimal service candidates, based on filters such as reputation.
c) Perform the service compositions driven by the QoS, which are the restrictions of the optimization function.

This approach, whenever a disruptive event occurs, conducts a search of appropriate services in a centralized manner (i.e., using the UDDI), to rebuild a composition of services that satisfies the agreed requisites. The structure of the composition is found out by considering a variety of techniques, from optimization techniques that require heuristics algorithms to face the problematic of combinatorial optimization (known to be NP-hard), to an approach that uses Artificial Intelligence (AI) based planning to achieve a near optimal solution and accelerate the execution time. Some authors propose parallelism based on the division of complex tasks into many smaller ones where each sub-task is responsible for a local optimization. Innovative and non-classical solutions, such as the self-organization that was introduced originally by Ashby [6], refer to a cooperation process without any centralized decision. The benefits of decentralization were also investigated in [7], suggesting a decentralization on the service discovery phase by using the social plasticity of the providers. With the aim of improving the system, several authors also suggest an innovative paradigm using integrating agents with SOA to take advantage of agents important features, e.g., loose coupling, decentralization, distribution, and autonomy, to intelligently achieve the client's needs [8].

In addition to the referred works that focus on how to reconfigure the process, a relevant set of methodologies related to defining the moment of change is also proposed, giving relevance to different strategies about (when) reconfiguring. For example, reconfiguring the system due to new consumer policies and requirements [9], when a new service is requested [10] or in the worst cases when an error or disturbance occurs. The work described in [11] covers all undesired events and identifies unexpected opportunities through reactive, predictive, and periodic strategies. Aligned with this trend adoption and with the increasing modification needs, service reconfiguration becomes the *de facto* approach that studies answers to the reconfiguration requirements.

In industrial systems, and in particular in the manufacturing domain, SOA-based paradigms have been proposed for automation and integration of services by extending the SOA paradigm to the domain of embedded low-level devices, such as sensors and actuators [12]. SOA is also used to implement collaborative manufacturing with intelligent Web services [13], and in another work, SOA and MAS are joined to enhance the manufacturing service collaboration as demonstrated in industrial automation [14]. Current trends related to the horizontal and vertical integration is also being faced by using SOA approaches, e.g. to support the increasing or diversity of the system's products or services. In [15], the authors proposed a service model for the dynamic production reconfiguration, in particular to reorganize the machinery in a manufacturing plant to be adapt to a new introduced product.

The service reconfiguration in dynamic environments needs to be quickly adaptable in real time and proactive (as stated in R1). Such dynamism can be monitored by means of existing maintenance strategies (e.g., reactive, preventive and predictive), which are covered by [11] where AI techniques were used to go beyond the traditional operational research by speeding up the generation of potential reconfigurations. However, based on the available service reconfiguration literature in manufacturing domain, the main concern relies on the service integration itself, without mentioning in practice any evidence of service reconfiguration. The majority of the existing reconfiguration solutions are performed manually due to the occurrence of failure events or product changeovers, even when the planners predict the actions to be performed. The reconfiguration is usually also achieved by a centralized composition planner, that does not provide the impact of the proposed new solutions (as stated in R2), and does not take into consideration the need for a smooth reconfiguration in case of change (as indicated in R3). The analysis and execution of the reconfiguration process are usually carried out in an individual way without considering the future impact and without regarding a collaborative analysis (as stated in R4).

Having this in mind, the challenge is to develop an approach that takes a step forward by evaluating potential possibilities in advance, having the capability to self-reconfigure the components without the need to stop or re-program the system, reducing the perturbation impact and decreasing the need for external intervention.

3 Dynamic Multi-agent-based Service Reconfiguration

The proposed approach for the service reconfiguration aims to comply with the requirements previously described, and considers the use of MAS principles and intelligent algorithms to support the when and how phases of the reconfiguration process.

In this ecosystem, the resource agents (RA) encapsulate every shop floor stations functionalities, as illustrated in Fig. 1, and publish as services the processes they can offer (i.e. each production RA act as service providers).

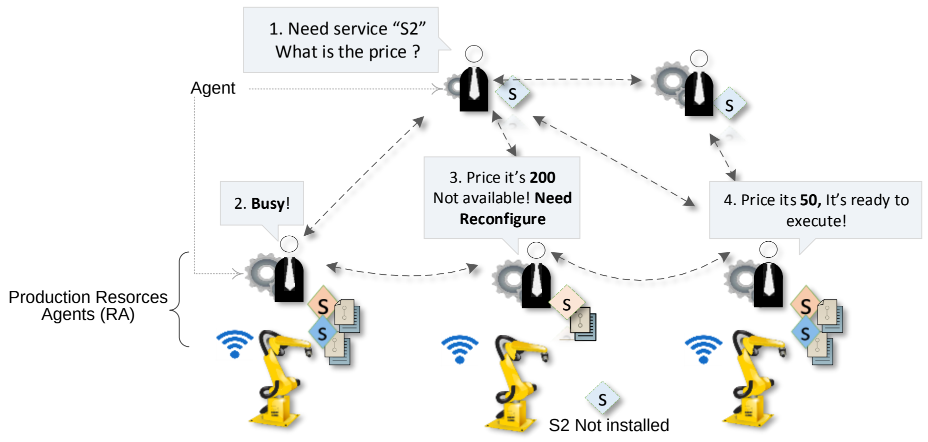
1. Need service "S2" What is the price ?
Agent
S
S
2. Busy!
3. Price it's 200 Not available! Need Reconfigure
4. Price its 50, It's ready to execute!
Production Resorces Agents (RA)
S
S
S
S
S
S
S2 Not installed

Fig. 1. Multi-agent based cable to perform Decentralized Service Reconfiguration.

The service consumers (such as intelligent products) need to consume the production resources services to meet the production demand for this it's necessary to win an auction, thus based on the RA's local schedule, services performance and availability bid at a certain price that shared provide. From the RA' perspective, they try to get as many services invocations as possible, at the highest price. For this purpose, they are continuously aware of the competitiveness of their services and able to execute a service reconfiguration when an opportunity to improve their services is identified. For this purpose, the resource agents embed several intelligent algorithms to handle the when and the how phases.

3.1 Discovering Opportunities and Determining Reconfiguration Solutions in Automatic Manner

Aiming to face the service reconfiguration, each individual agent is continuously collecting data and applying actions to maximize its utility under production uncertainty and demand variability. In this context, a crucial issue is to maintain a competitive catalog of services that addresses the customer demands, which is possible by embedding a reconfiguration module that considers the following components [11], as illustrated in Fig. 2: When to Reconfigure (WtR), How to Reconfigure (HtR) and Decide Reconfiguration Solution (DRS).

The dynamic reconfiguration is challenging and can lead to unpredictable opportunities to evolve based on the fact that several variables are unknown, either from the physical perspective (such as the degradation of quality and the unforeseen plug-in of devices) or from the logical viewpoint (such as the configuration of the manufacturing plant configurations and the scheduling of production orders). In this perspective, predicting these opportunities is wiser than simply reacting, which requires to collect data from the different sources, namely shop floor and customers, to support the several components of the reconfiguration module.

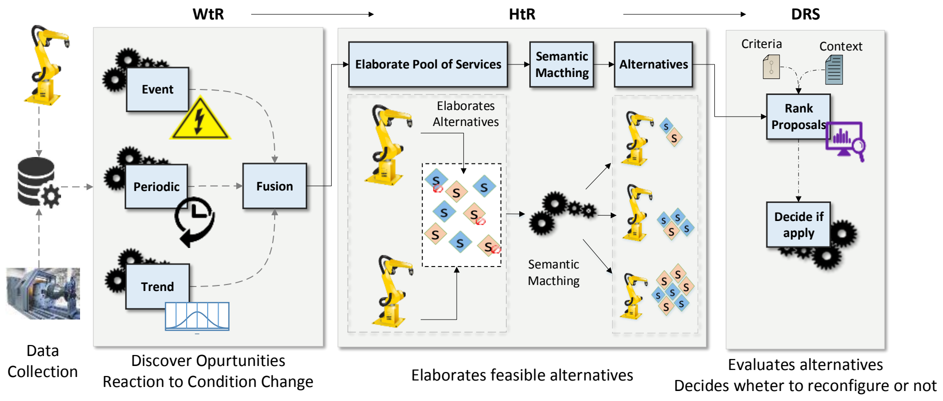

Fig. 2. Service reconfiguration module implemented in each agent.

3.1.1 When to Reconfigure

The step in the proposed service reconfiguration approach, performed by the WtR component, is related to monitor and analyze the collected data to identify the triggers or opportunities for the reconfiguration, e.g., a performance or quality degradation, a failure occurrence or the introduction of new products. The WtR model relies on three different triggering strategies to detect possible situations to reconfigure [11]: event, periodic and trend.

The event triggering strategy uses an event-driven approach to detect events related to the system condition changes, e.g., a resource failure, the addition of a new resource/component or the removal of an existing resource/component. This strategy permits a good reaction to face unexpected events, which is an important feature in dynamic and unpredictable environments.

The trend triggering strategy is responsible to identify, as earlier as possible, a tendency or pattern in the degradation of a service performance, allowing the earlier implementation of actions to improve its performance or to reconfigure this service by another more useful. Several algorithms can be used to identify these opportunities, namely the anomaly detection, the cluster analysis-based and the structural break [11]. The anomaly detection and cluster analysis-based methods are more appropriate to discover anomalies in patterns [16], and the structural break method is more appropriate to perform a simple trend analysis.

The periodic triggering strategy uses a periodic check to verify the current service performance and decide about the opportunity to reconfigure. The triggering time interval should be dynamically adjusted to better fit the system dynamics, i.e. increasing or decreasing this value, taking into the consideration the application of proper machine learning algorithms. Q-learning [17] is a suitable approach to address this challenge, since it provides a positive/negative reinforcement feedback that handles the system's dynamics, allowing to converge to an optimal value.

3.1.2 How to Reconfigure

After being identified an opportunity to reconfigure, the HtR component is triggered with the responsibility to determine how the service reconfiguration can be implemented. The process comprises the elaboration of a pool of possible alternatives for the service reconfiguration, followed by a semantic checking that reduces the dimension of the alternative solutions (see [4] for more details). The generation of alternative solutions considers the improvement of the resource's utility and consequently the improvement of the services' behavior and/or the changing of the service's catalog.

3.1.3 Decide Reconfiguration Solution

After the calculation of the set of alternative reconfiguration solutions, it is necessary to evaluate the effectiveness of each alternative and determine the best one. The evaluation method uses a reconfiguration index (RI) [4, 18] that quantifies the advantage of performing a certain reconfiguration, considering the ratio between the reconfiguration effort with the expected profit that the reconfiguration can bring [4].

At the end, the several alternative solutions are ranked according to the evaluation method and considering the criteria defined by the system managers.

This component is also responsible to decide if the best reconfiguration solution should be implemented or not, taking into consideration the nervousness control of the resource. In fact, the system stability is a very important issue and each resource agent must control its nervousness to avoid falling into a chaotic system. The system should be pro-active to identify opportunities for reconfiguration but should not be constantly changing the service reconfiguration because it implies a performance degradation.

3.2 Decentralized Mechanism for Service Reconfiguration

The described proposed approach for the service reconfiguration is carried out in a self-interested, autonomous and competitive way. Each agent, in this competitive situation, is running individually the service reconfiguration mechanism and does not share its objectives with the other agents. However, in collaborative environments, the lack of control or a normative environment using self-interested agents can lead to problematic situations that are damaging to the entire system, namely:

- Conflict situations: conflict of interest among agents have to be managed, e.g., in case several agents want to reconfigure to provide the same service.
- Deadlock situation: simultaneous individual service reconfigurations based on the interest of the most valuable services can lead to situations where no one offers the least profitable but necessary services.

In this sense, the adoption of a decentralized service reconfiguration approach and the design of a well-defined collaborative interaction protocol facilitate the avoidance of deadlocks [19], allowing to reach a mutual agreement that benefits the collaborative system behavior, namely improving the competitiveness of the system and balancing the resources' utilization rate, and avoiding a service reconfiguration carried out in an uncoordinated and chaotic way. To deal with this, an interaction protocol permits to collect the agent's intentions of its interests in adapting/reconfiguring its catalog of services. The protocol works in a synchronous manner by transferring data and control of the reconfiguration design among the agents (rather than using a central agent), to acquire all the data and understand if a global configuration is feasible. In particular, the protocol considers several resource agents, as illustrated in Fig. 3, one acting as an initiator of the interaction (i.e. the one that wants to change its service) and others participating in the collaborative interaction.

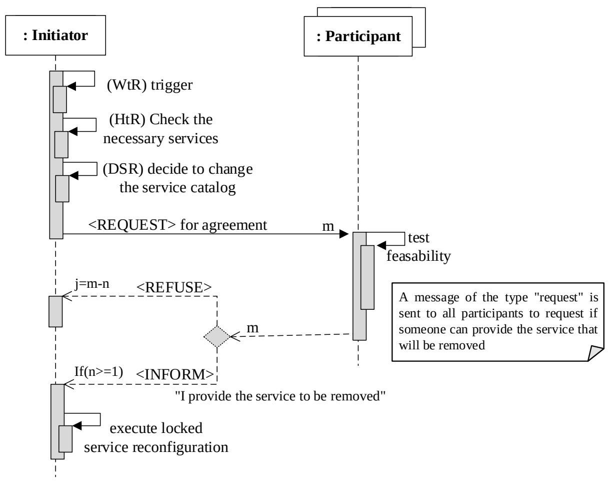
: Initiator
: Participant
(WtR) trigger
(HtR) Check the necessary services
(DSR) decide to change the service catalog
<REQUEST> for agreement
m
test feasability
A message of the type "request" is sent to all participants to request if someone can provide the service that will be removed
j=m-n
<REFUSE>
m
If(n>=1)
<INFORM>
"I provide the service to be removed"
execute locked service reconfiguration

Fig. 3. Collaborative protocol for the interaction among agents.

The initiator, after identifying an opportunity to reconfigure, by using the WtR module that decides to implement a potential service reconfiguration, notifies its intention to implement a service reconfiguration and waits for the non-objection of all participants, aiming to control the system nervousness and to avoid entering into a chaotic situation (e.g., a non-feasible configuration, where no one is providing a necessary service). For this purpose, the initiator sends a "REQUEST" message to all participants, inquiring if someone can provide the service that will be removed. Each participant will reply with "INFORM" or "REFUSE" messages, according to its possibility to provide the service or not.

After, the initiator is waiting for the replies from the participants. If the initiator receives at least one "INFORM" then the system has achieved a feasible collaborative reconfiguration (since at least one participant offers the service that will be reconfigured), otherwise the proposed service reconfiguration leads to a non-feasible configuration (in the collaborative perspective), and the initiator will ignore the opportunity to reconfigure. Note that despite being beneficial for one service provider, the reconfiguration is not beneficial for the whole collaborative system, and consequently should not be implemented.

The proposed approach, considering the individual and system perspectives, has a high focus on flexibility in many forms:

- The dynamic individual service reconfiguration is directly mapped in competitive situations, where the self-fish behavior of the agents leads to a truly dynamic and decentralized service reconfiguration.
- The decentralized interaction mechanism permits to build a better and more robust reconfiguration approach, as well as to smooth the nervousness problems, because the conflicts of competition between services are avoided and even if there are several individual service reconfiguration interests, the resource agents decide if they are worth for the system benefits (also avoiding the implementation of non-feasible configuration solutions).

As a drawback, this approach does not ensure the optimality of the service reconfiguration solution. However, as stated by [19], such type of approaches improves the performance regarding increased throughput and lower response. Nevertheless, both methods facilitate scaling the system to new agents. From one hand, they can be non-cooperative perspectives, performing autonomously the service reconfiguration, and from the other hand, they can be cooperative where the overall reconfiguration emerges from the participants without joining in just one agent the entire image of the system.

4 Experimental Validation

4.1 Description of the Case Study

The proposed approach for the service reconfiguration was tested using the flexible manufacturing system AIP-PRIMECA FMS [20], which comprises 5 workstations linked by a conveyor system. The workstations offer a set of services related to the execution of several operations (i.e. sub-products produced in this system, namely the letters A, B, E, I, L, P, and T), which combined can produce the final products BELT and AIP. This case study created based on batch production forcing to set-up and reconfigure the production equipment according to the demand. As illustrated in Table 1, each sub-product has its assembly process plan that needs to be followed to complete its production, e.g., to generate the sub-product T, the sequence of operations is to load the assembly base plate into the shuttle, followed by assembling 2 Axis, r and L components, performing the inspection and finally unloading the product from the shuttle.

Table 1. Process plans for the catalog of products.

	product B	**product E**	**product L**	**product T**	**product A**	**…**
1	Loading	Loading	Loading	Loading	Loading	
2	Axis	Axis	Axis	Axis	Axis	
3	Axis	Axis	Axis	Axis	Axis	
4	Axis	Axis	Axis	r_comp	Axis	
5	r_comp	r_comp	I_comp	L_comp	r_comp	
6	r_comp	r_comp	I_comp	Inspection	L_comp	
7	I_comp	L_comp	Screw_c	Unloading	I_comp	
8	Screw_c	Inspection	Screw_c		Screw_c	
9	Inspection	Unloading	Inspection		Inspection	
10	Unloading		Unloading		Unloading	

Table 2 represents the catalog of services offered by each machine, indicating the processing time for each provided service. For example, the “r_comp” service can be executed by the workstation M3 while the workstation M2 offers the service “Axis”. Aiming to increase the flexibility of the FMS and to create a richer scenario to test the service reconfiguration approach, a slight change was introduced in the scenario described in [20]. This change is related to expand the number of services provided by the machines, and particularly services that are available but are not currently offered in the machines’ catalog. For example, the machines M2 and M3 have the possibility to change their catalog of services by offering, respectively the services “r_comp” and “L_comp”. In case the agents decide for the service reconfiguration, a maintenance intervention is required to improve the service performance (taking 20 seconds) or to change the service provided (taking 30 seconds).

Table 2. Catalog of services provided by each machine (processing times in seconds).

Service	**M1**	**M2**	**M3**	**M4**	**M5**
Loading	I (10)				
Unloading	I (10)				
Axis		I (20)			
r_comp		NI (20)	I (20)		
I_comp			I (20)		
L_comp			NI (20)	I (20)	
Screw_c				I (20)	
Inspection					I (5)

Legend: I – installed in the catalog; NI – available but currently not offered in the catalog

To simulate realistic scenarios for evaluating the reconfiguration hypotheses the occurrence of disturbances is considered. For this purpose, machines M2 and M3 have a probability of failure of 25% for all services in their catalogs, and when an improvement of the service performance is executed, its failure is reduced by 3%.

The designed MAS was implemented using the JADE framework [21], being the iteration among the agents performed by using FIPA-ACL compliant messages. Each resource agent contains the implementation of the “when” strategies that allow identifying opportunities to reconfigure. In this work, the following strategies of the WtR module were implemented [4, 11]:

- Event: related to the identification of reactive and critical situations, e.g., new production requests and resource/service failures.

- Trend: related to the earlier identification of patterns that result in deviations and anomalies, e.g., loss of quality of a service and decrease of resource usability.

These strategies were implemented on the monitoring behavior of each agent. In respect to the Event strategy, it was triggered by the monitoring procedure that contains the monitoring features to detect new products and disturbances, e.g., service or resource failures. The trend strategy requires more information (i.e., historical and contextual production data) to produce better real-time analysis and statistical computation to support the identification of potential deviations and anomalies patterns. In this case, the algorithms performing data analysis were implemented in R language [22] and accessed by the agents by using the RServe API connected through TCP/IP, which acts as a back-end for web services. From the R-side, the anomaly detection and cluster analysis based methods are continuously running in background to detect the degradation of a service performance. However, when facing production changeover situations, these two algorithms may create some confusion leading to identifying the characteristics of a new product as an outlier that consequently will result in bad configurations. In these cases, the Event module notifies the Trend module about a product change, allowing the adaptation of its trend analysis for a specific product, aiming to perform more accurate predictions.

In the same manner, agents incorporate the HtR module that allows generating potential reconfiguration solutions based on the different types of service reconfiguration, namely weak reconfiguration by improving the service performance, and strong reconfiguration by replacing the service catalog. A special aspect of the HtR algorithm [4] to ensure the feasibility of service reconfiguration solutions is the semantic verification of resources and pool of services, using JENA, to reduce the number of these alternatives.

4.2 Experimental Results

Some testing scenarios were designed to assess the described service reconfiguration approach, exploiting the impact of enabling the individual service reconfiguration performed by the distributed resource agents and enabling the collaborative mechanism to avoid conflicts and chaotic behaviors. In the experiments, the catalog of orders included the production of 20 BELT products. Fig. 4 illustrates the experimental results for the different scenarios.

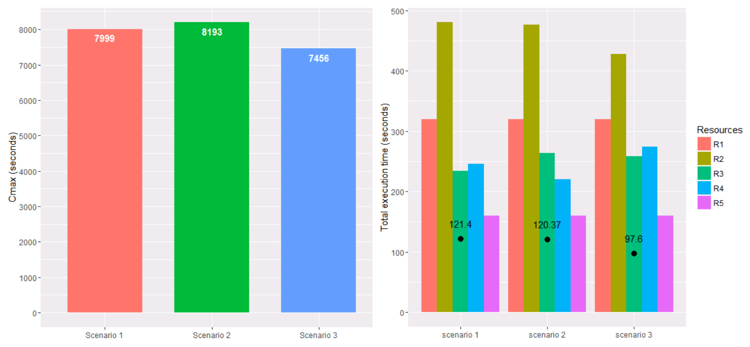

Fig. 4. Experimental results for scenarios with and without service reconfiguration.

Initially, the system was running in normal mode with the service reconfiguration and collaborative mechanisms disabled (scenario #1), measuring the Cmax value (i.e. the makespan that is defined as the total amount of time to process a given manufacturing order), which is represented in the left graphic of Fig. 4. The right graphic of the same figure represents the standard deviation (σ) of the service utilization for each machine, aiming to verify how well distributed and balanced is the production. In this case, σ values close to zero represent a good balanced production. The scenario #1 presents a value of 7999 seconds for Cmax and 121.4 seconds for σ, working as a baseline to compare the proposed service reconfiguration approach.

The second scenario (scenario #2) is related to the enabling of the service reconfiguration mechanism in each resource agent in a self-fish mode, which means that the agents will execute their service reconfiguration individually and without any collaborative procedure. The results, depicted in Fig. 4, show a decrease of the system performance in 1,8%, reflected in the need to have more 194 s to produce the 20 BELT products, and a slightly more balanced effort among the machines. The small increase in the Cmax may be due to conflict situations, since agents are performing service reconfiguration procedures in a self-fish manner in a very typical collaborative environment, which means that they are reconfiguration to maximizing their individual interests and not the overall system goals.

The third scenario (scenario #3) is related to enabling the collaborative mechanism operating over the decentralized service reconfiguration mechanism being performed individually by the several resource agents. In this case, the results show an increasing of the system performance (illustrated by the reduction of 6,7% of the Cmax), as well as the reduction of the σ value to 97.6 seconds, which means a better disturbance of the resource utilization. This scenario clearly shows the advantages of applying the collaborative mechanisms to harmonize the service reconfiguration performed individually and in a non-cooperative manner by the distributed agents to reach collaborative environments.

This set of experiments allowed to verify that enabling the individual service reconfiguration, the system production efficiency is slightly improved, which small improvement is due to some possible contradictory and conflictual reconfigurations. The activation of the collaborative mechanism, with the agents taking the final decision about the service reconfiguration, not only considering its own perspective but also the benefit of the whole system, permits to achieve a higher production efficiency.

A fourth (scenario #4), with a batch of 30 BELT products, was considered to test the dependency of the proposed service reconfiguration approach with the dimension of the production order set. Fig. 5 summarizes the achieved results for this scenario.

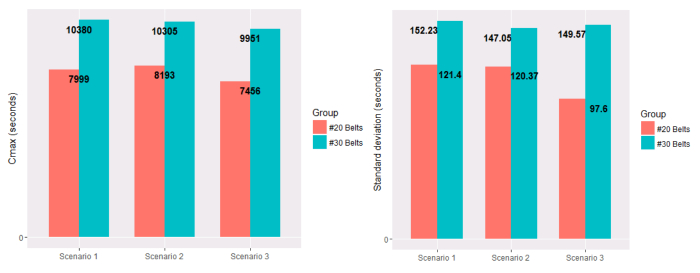

Fig. 5. Experimental results for different batch sizes (20 and 30 batch sizes).

The analysis of the results shows a considerably higher service utilization rate for the batch of 30 BELT, when compared to the batch of 20 BELT, which means that the service reconfiguration considering a proper triggering mechanism represents a wiser and maximized utilization of the services. The makespan value is more dependent on the dimension of the production batch than the standard deviation parameter, being possible to see improvement in the Cmax value for bigger production batch sizes.

5 Conclusions and Future Work

The dynamic service reconfiguration process is, nowadays, a hot topic in manufacturing systems, aligned with the cyber-physical systems context, and particularly with the Industry 4.0 initiative. A literature review on this field clearly shows that the service reconfiguration is usually performed in a manual, offline and centralized manner, and traditionally considers the service integration such as CPS, without mention the truly service reconfiguration.

The proposed approach described in this paper considers the challenge of performing a dynamic, online and decentralized service reconfiguration, where intelligent software agents apply different strategies to identify opportunities to reconfigure in a proactive way. These agents, after identifying opportunities to reconfigure their catalog of services, execute adequate algorithms to determine the alternative possibilities to evolve, and decide for the most promising one. This approach addresses two different situations: a non-cooperative or competitive environment, where the reconfiguration is decided and executed individually by each one of the distributed agents, and a collaborative environment, where the reconfiguration is triggered individually by the distributed agents but it is only executed if is seen as beneficial for the whole system. A decentralized collaborative mechanism was designed to allow addressing this second situation, avoiding reaching non-feasible configurations and also promoting the balance of the resources utilization.

The proposed service reconfiguration solution was implemented in JADE and tested in a flexible manufacturing system use case. Both in competitive and collaborative scenarios, our service reconfiguration approach has been proven to display better performance than the normal operation, materialized in lower "makespan" (Cmax) values and also better distribution of the resources utilization. The increase of the batch size also positively affects the use of the proposed service reconfiguration approach. Thanks to the multiagent-based system, the proposed on-the-fly service reconfiguration can be implemented dynamically, automatically and proactively to improve the service profitability, contributing for the beneficial of the individual entities as well as to the entire system (in case of the collaborative environment).

Future work will be devoted to developing a completely decentralized collaborative mechanism to regulate the service reconfiguration and to develop rules to control the system nervousness avoiding falling into a chaotic situation. This approach, although introducing a higher complexity effort, permits to evolve smoothly easily respond to future disturbances.

References

[1] Y. Koren et al., “Reconfigurable Manufacturing Systems,” CIRP Ann. - Manuf. Technol., vol. 48, no. 2, pp. 527–540, 1999.

[2] J. Lee, J. Ni, D. Djurdjanovic, H. Qiu, and H. Liao, “Intelligent prognostics tools and e-maintenance,” Comput. Ind., vol. 57, no. 6, pp. 476–489, 2006.

[3] C. Zeginis and D. Plexousakis, “Web Service Adaptation: State of the art and Research Challenges,” Tech. Rep. n. 410. University of Greece, Institute of Computer Science FORTH- ICS, 2010.

[4] N. Rodrigues, P. Leitão, and E. Oliveira, “Dynamic Service Reconfiguration with Multi-agent systems”, Service Orientation in Holonic and Multi-Agent Manufacturing Studies in Computational Intelligence (SOHOMA’16), vol. 694, Springer, pp. 307–318, 2017.

[5] T. Erl, “Service-Oriented Architecture: Concepts, Technology, and Design”, Prentice Hall/Pearson PTR, 2005.

[6] W. R. Ashby, “Principles of the self-organizing dynamic system,” J. Gen. Psychol., vol. 37, pp. 125–128, 1947.

[7] E. del Val, M. Rebollo and V. Botti, “Combination of self-organization mechanisms to enhance service discovery in open systems”, Information Sciences, vol. 279, pp. 138-162, 2014.

[8] N. Rodrigues, P. Leitão, and E. Oliveira, “Dynamic Composition of Service Oriented Multi-agent System in Self-organized Environments”, Intelligent Agents and Technologies for Socially Interconnected Systems (IAT4SIS’14), Prague, Czech Republic, pp. 1-6, 2014.

[9] Y. Bar-Yam, “Dynamics of complex systems”, Perseus Books, Cambridge, MA, USA, 1997.

[10] E. M. Maximilien and M. P. Singh, “A Framework and Ontology for Dynamic Web Services Selection,” IEEE Internet Comput., vol. 8, no. 5, pp. 84–93, 2004.

[11] N. Rodrigues, P. Leitão, and E. Oliveira, “Triggering Strategies for Automatic and Online Service Reconfiguration”, Proceedings of the 11th Iberian Conference on Information Systems and Technologies (CISTI’16), Gran Canaria, Spain,pp. 76–82, 2016.

[12] A. Cannata, M. Gerosa, and M. Taisch, “SOCRADES: A framework for developing intelligent systems in manufacturing”, Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management (IEEM’08), pp. 1904-1908, 2008.

[13] W. Shen, Y. Li, Q. Hao, S. Wang, and H. Ghenniwa, “Implementing collaborative manufacturing with intelligent Web services”, Proceedings of the Fifth International Conference on Computer and Information Technology (CIT’05), pp. 1063-1069, 2005.

[14] J. M. Mendes, P. Leitão, F. Restivo, and A. W. Colombo, “Service-Oriented Agents for Collaborative Industrial Automation and Production Systems”, in Holonic and Multi-Agent Systems for Manufacturing: 4th International Conference on Industrial Applications of Holonic and Multi-Agent Systems, HoloMAS 2009, Linz, Austria, Springer, Berlin, Heidelberg, pp. 13-24, 2009.

[15] D. Marcos-Jorquera, F. Macia-Perez, V. Gilart-Iglesias, and J. A. Gil-Martinez-Abarca, “Service model for the management of industrial environments. Dynamic reconfiguration of production elements”, Proceedings of the 5th IEEE International Conference on Industrial Informatics (INDIN’07), pp. 249-254, 2007.

[16] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” ACM Comput. Surv., vol. 41, n. 15, pp. 1-58, 2009.

[17] C. Watkins and P. Dayan, “Q-learning”, Mach. Learn., vol. 8, n. 3-4, pp. 279–292, 1992.

[18] P. Neves, “Reconfiguration Methodology to improve the agility and sustainability of Plug and Produce Systems,” Ph.D. dissertation, 2016.

[19] G. B. Chafle, S. Chandra, V. Mann, and M. G. Nanda, “Decentralized Orchestration of Composite Web Services”, Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters, pp. 134-143, 2004.

[20] D. Trentesaux, C. Pach, A. Bekrar, Y. Sallez, T. Berger, T. Bonte, P. Leitão, and J. Barbosa, “Benchmarking flexible job-shop scheduling and control systems,” Control Eng. Pract., vol. 21, n. 9, pp. 1204-1225, 2013.

[21] F. Bellifemine, G. Caire, D. Greenwood, “Developing Multi-Agent Systems with JADE”, Wiley, 2007.

[22] R Core Team, “R: A Language and Environment for Statistical Computing”, R Foundation for Statistical Computing, Vienna, Austria, 2015.