technologies

MDPI

Article

Solving the Job-Shop Scheduling Problem in the Industry 4.0 Era

Matheus E. Leusin [1,*], Enzo M. Frazzon [1], Mauricio Uriona Maldonado [1], Mirko Kück [2] and Michael Freitag [2,3]

1 Department of Industrial and Systems Engineering, Federal University of Santa Catarina, Florianópolis 88040-900, Brazil; enzo.frazzon@ufsc.br (E.M.F.); m.uriona@ufsc.br (M.U.M.)
2 BIBA—Bremer Institut für Produktion und Logistik GmbH at the University of Bremen, 28359 Bremen, Germany; kue@biba.uni-bremen.de (M.K.); fre@biba.uni-bremen.de (M.F.)
3 Faculty of Production Engineering, University of Bremen, 28359 Bremen, Germany
* Correspondence: leusin@uni-bremen.de; Tel.: +55-47-99174-8955

Received: 30 August 2018; Accepted: 5 November 2018; Published: 16 November 2018

Abstract: Technological developments along with the emergence of Industry 4.0 allow for new approaches to solve industrial problems, such as the Job-shop Scheduling Problem (JSP). In this sense, embedding Multi-Agent Systems (MAS) into Cyber-Physical Systems (CPS) is a highly promising approach to handle complex and dynamic JSPs. This paper proposes a data exchange framework in order to deal with the JSP considering the state-of-the-art technology regarding MAS, CPS and industrial standards. The proposed framework has self-configuring features to deal with disturbances in the production line. This is possible through the development of an intelligent system based on the use of agents and the Internet of Things (IoT) to achieve real-time data exchange and decision making in the job-shop. The performance of the proposed framework is tested in a simulation study based on a real industrial case. The results substantiate gains in flexibility, scalability and efficiency through the data exchange between factory layers. Finally, the paper presents insights regarding industrial applications in the Industry 4.0 era in general and in particular with regard to the framework implementation in the analyzed industrial case.

Keywords: Multi-Agent Systems; Internet of Things; IoT; Digital Manufacturing; Job-shop Scheduling Problem

1. Introduction

Over the years, industry performance has increasingly benefited from technological developments. More recently, new information technologies have given rise to intelligent factories in what is termed as Industry 4.0 (i4.0) [1]. The i4.0 revolution involves the combination of intelligent and adaptive systems using shared knowledge among diverse heterogeneous platforms for computational decision-making [1–3], within Cyber-Physical Systems (CPS).

The emergence of CPS brings up new solving opportunities to industrial problems since it integrates computational with physical processes [2,4], including coordination, monitoring and control of physical operations and engineering systems [5]. A typical example of an industrial opportunity of this kind is scheduling, whose goal is to achieve resource optimization and minimization of the total execution time [6].

Given the characteristics of complexity, dynamism and stochasticity of industrial environments, the resolution of this type of problem may involve the use of very complex solutions. When considering the industrial job-shop, these characteristics include a diversity of products, processes and unpredictable events such as the arrival of new orders, processing delays, machine failures

and unavailability of raw materials, tools, or people [7,8]. Accordingly, the problem of creating a job-shop scheduling, known as Job-shop Scheduling Problem (JSP), is considered one of the hardest manufacturing problems in the literature [9].

The scheduling result determines the resource allocation for each time value and the order of the tasks' execution [6]. Adequate production scheduling enables reducing the time needed to complete tasks, increasing capacity utilization and overall efficiency and thus maximizing the organization' profitability [10]. Currently, most of the job-shop scheduling literature uses a static approach for solving the JSP [11], in part due to the difficulty of considering job-shop dynamics. Among the literature using a dynamic approach to solve the JSP, the use of Multi-Agent Systems (MAS) is remarkable.

In the MAS approach, production resources are established as intelligent agents, which communicate with each other for achieving dynamic reconfigurations and high flexibility [12,13]. Due to agents characteristics of autonomy, robustness to failures and ability to dynamically and flexibly schedule production [14], MAS allows achieving computational agility and better reactivity and adaptability in highly changing environments such as job-shops [15]. As a result, MAS has been recognized as a promising paradigm for the upcoming generation of manufacturing systems [14,16–18], as well as one of the most suitable technologies for dealing with the JSP [19].

However, despite the success of MAS approaches for production planning, scheduling and logistics, this approach has still potential to be explored for industrial automation, with most applications involving only laboratory environments or industrial prototypes. One of the main reasons is the lack of standards use for developing such automated systems [3,20]. One of the automation pillars is the use of clearly defined processes. Hence, the use of standards in automated environments is not only common but also essential to creating stable processes and reliable systems [21]. At the same time, standardization is vital for diffusing agent-based technologies in industrial applications [22].

Thus, the main goal of this paper is to propose a dynamic framework for the JSP, based on current industry standards and technologies. Our framework considers a MAS architecture to create an intelligent system, self-adaptable to unforeseen disturbances as they occur in the job-shop. The framework performance is compared to a real industrial scenario setting through simulation. Therefore, among main contributions of this paper one can cite: (i) proposing and validating an intelligent system to handle the JSP, which is highly flexible, robust and adaptable; (ii) analysis regarding the study case, including the verification that balancing the analyzed production line through the use of dispatch rules can provide fully meeting the demand while also minimizing inventory levels (at the cost of increasing the number of setup activities performed); (iii) confirmation that combining real-time data collection and dispatching rules allow for fast adapting to unforeseen disturbances like machine breakdown situations; and (iv) identification of future research on MAS and data-driven technologies.

Thus, following this introduction, Section 2 presents the materials and methods used in this paper. Section 3 describes the theoretical framework related to the JSP, current MAS applications in CPS and industrial standards. Next, Section 4, introduces our framework, followed by Sections 5 and 6, which presents the framework' application to a real industrial scenario and the main conclusions of this work, respectively.

2. Materials and Methods

First, we conducted a narrative review of the literature on three topics:

1. JSP, in order to identify the main characteristics and methods to deal with it;
2. Applications of MAS in CPS, in order to assess the best practices in the development of MAS and technologies currently applied in the implementation of such systems in the CPS domain;
3. Industrial standards, in order to analyze the main standards currently used in the industry, as well as relevant characteristics to be considered for facilitating its implementation.

The search was carried out in three databases: Scopus, Web of Science and Google Scholar. Moreover, conference papers, as well as discussions with experts in the field were used as secondary sources. The proposed framework is the result of this phase.

In the second phase a case study was used to analyze implementation aspects of the proposed framework. The case study comprised a real industrial scenario analysis, where a first test application of the proposed method was simulated and evaluated. A preliminary study was published and discussed in the 16th IFAC Symposium on Information Control Problems in Manufacturing (INCOM) Conference [23]. The company is a manufacturer of mechanical parts, which works primarily in the automotive industry. The company performs monthly production scheduling with Preactor. Two simulation models were developed at this stage: one based on a real monthly scheduling generated by Preactor for the analyzed production line and another in which the proposed framework was used to dynamically coordinate scheduling activities of the same production line. Both simulation models were developed in Anylogic, which offers multimethod features, comprising modeling and simulation tools for system dynamics, discrete event and agent-based approaches [24].

Demand data was provided by the company and represents typical monthly demand. This data is the same currently used by Preactor to perform a monthly scheduling. The framework performance is verified by comparing the results of both simulation models. Figure 1 shows the main steps carried out in this article.

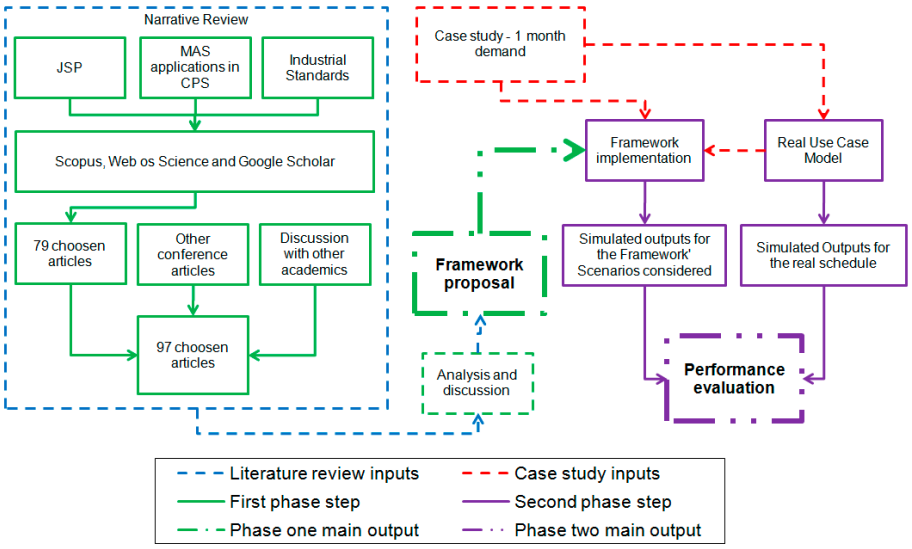

Figure 1. Research Procedure workflow.

3. Literature Review

3.1. The Job-Shop Scheduling Problem

The Job-shop Scheduling Problem (JSP) is one of the most important problems in manufacturing [9] due to its effect on the whole company and supply chain performance. The JSP involves sequencing a set of jobs, which have their own chain of operations, to be processed for a given length of time in a set of machines [9].

Finding an ideal solution to this problem in an acceptable amount of time is challenging due to job-shop environments high complexity [25]. Because of the factorial explosion of possible solutions, JSPs belong to the intractable numerical problems class, also known as NP-hard [9,25,26]. In this class of problems, exact or complete methods providing optimal solutions might take longer periods to be computed, whereas approximate methods can provide nearly optimal solutions in more appropriate time [27,28]. Due to this, there are different approaches and methods to deal with the JSP,

each impacting in different ways the scheduling in the job-shop. These approaches and methods are shown in Figure 2.

Job-shop Scheduling Problem	Approaches for dealing with the JSP	Heuristic Rules		
		Classical Optimization		
		Artificial Intelligence (A.I.) approaches		
	Scheduling Flexibility	Classical Approach		
		Dynamic approach	Rule used for creating the scheduling	Completely reactive scheduling
				Predictive-reactive scheduling
				Predictive-reactive robust scheduling
				Robust pro-active scheduling
			Decision modes	Simulation based approaches
				Artificial intelligence based approaches
				Agent-based approaches
	Scheduling updating and Rescheduling	Rescheduling policies	Event-based rescheduling	
			Periodic rescheduling	
			Hybrid rescheduling	
		Updating an existing scheduling	Right-shift rescheduling	
			Complete rescheduling	
			Partial rescheduling	

Figure 2. Job-shop Scheduling Problem approaches and methods.

The distinct approaches and methods directly influence the volume, frequency and intensity of data exchange in the job-shop, as well as the scheduling quality. These characteristics affect the configuration and implementation of MAS and consequently the production performance of a company. Thus, the following subsections discuss these approaches and methods.

3.1.1. Approaches to Deal with the JSP

There are three main approaches used for handling the JSP: Heuristic Rules, Classical Optimization and Artificial Intelligence (A.I.) approaches. All of them can be used both for creating a scheduling in advance (classical approach) as for dealing with scheduling dynamically (dynamic approach).

Heuristic rules, also known as dispatching rules, priority rules, or scheduling rules, are used for determining which job will be selected next among the variety of jobs in queue to be manufactured on a specific machine, without previously examining the outcome of the choices [29]. In addition of being one of the most common methods adopted for scheduling, there is a variety of dispatching rules acknowledged for a diversity of scheduling criteria. Reference [30] did a survey related to the use of dispatching rules for scheduling, presenting 26 distinct dispatching rules used for the most variable purposes. Some of these purposes include for example the use of dispatching rules for minimizing completion or execution times, maximizing the standard deviation or for a Just-in-time production.

Classical Optimization methods, in turn, are divided into two primary classical approaches, both based on enumeration techniques: the branch and bound method and dynamic programming. Branch and bound is based on lower bounds of the optimal problem solution, used for guiding a branching process that divides the problem into smaller and mutually exclusive sub-problems searched until the best solution is found. In dynamic programming, on the other hand, the set of possible schedules is generated through accounting each job as the last and checking up how well this particular choice would help in optimizing a given criterion. In this way, the optimal schedule is discovered by picking up the minimal of this set [29]. Based on this definition, Meta-heuristics can also be considered dynamic programming (although not cited in Reference [29]): they are high-level heuristics that guide nearest neighborhood search methods, which follow the idea of searching neighborhoods for escaping from local optima [31]. As examples of Meta-heuristic methods,

reference [31] highlight simulated annealing, tabu search and genetic algorithms, which may accept poor solutions for escaping local optima or may focus on finding out more adequate starting solutions (instead of just picking them up randomly) for the local search.

Finally, Artificial Intelligence methods also have broad application to handle the JSP in a variety of ways and combinations, including its use in heuristics and optimization methods as well. Reference [29] highlight Neural Networks, which comprise a system of individual computational neurons. Each neuron of this system has interconnections, which receive weighted inputs that go through an activation function, leading to a learning process [29]. Other A.I. methods cited in Reference [31] include (but are not limited to) knowledge-based systems, fuzzy logic, case-based reasoning and Petri nets.

3.1.2. Scheduling Flexibility of the JSP

Classical Approach

In the classical approach, also known as static scheduling or offline scheduling, the entire planning horizon is considered, being all the jobs scheduled at once. Thus, an a priori (predictive) schedule is created based on some conditions assumed prior to the creation of scheduling [32]. One of these assumptions may be the inclusion of disturbances that have a high probability of occurring in an attempt to create a predictive scheduling already adapted for these disturbances.

Other common assumption is that all information related to work and machines availability are previously known and that these conditions will not change during the execution of the scheduling [33,34]. However, job-shops are inherently complex, dynamic and stochastic with a diversity of products, processes, levels of production and occurrence of unforeseen disturbances, which makes a scheduling based on this assumption more sensitive to disturbances. These disturbances may include, for example, the variation in orders (cancellations, generation of new orders, delays or changes in ongoing orders), processing delays, machine failures and the unavailability of tools, raw materials, or workforce [7,8].

Dynamic Approach

Dynamic scheduling, also known as online scheduling [35], deals with scheduling considering real-time disturbances [31]. Thus, the program execution is considered step by step and priorities are determined and applied dynamically [28].

According to the rule used for developing the scheduling system, dynamic scheduling can be divided into four categories [36–38]:

- Completely reactive scheduling: no pre-schedule is generated and scheduling is done in real-time.
- Predictive-reactive scheduling: a schedule is generated beforehand and a rescheduling is considered for responding to real-time disturbances.
- Predictive-reactive robust scheduling: a schedule is generated beforehand and a rescheduling is performed when the impact of disturbances on performance measures is significant.
- Robust pro-active scheduling: the schedule is generated beforehand anticipating the impact of disturbances in the manufacturing system.

Regarding the use of dispatching rules, dynamic scheduling can further be classified according to the decision mode used for defining these rules. Reference [7] classify and define decisions modes in dispatching rules as (1) simulation-based; (2) Artificial Intelligence (A.I.)-based and (3) agent-based approaches. According to the authors, a simulation model or computational/knowledge model must be developed for the first two approaches for resembling the existing manufacturing system as similar as possible. Simulations are then used as a decision support tool for evaluating distinct dispatching rules for dynamic control, or A.I. techniques are applied for finding optimal solutions in the AI-based approach. Considering that the responses of this system should be taken quickly, time is of great

concern in these two types of approaches. Therefore, agent-based approaches have the advantage of quick response to dynamic changes through local decision-making. The main advantages and disadvantages of each type of dispatching rule under this classification are highlighted in Table 1.

Table 1. Decisions modes' advantages and disadvantages highlighted in Reference [7].

Decision Mode	Advantages	Disadvantages
Simulation-based	• Easier and cheaper to implement • Useful to answer "what if" questions	• Challenging to validate • Greater difficulty to adapt itself to real time system changes • Results may be difficult to interpret • Dependent on centralized computing
Artificial intelligence	• Best optimization results in relation to the other two approaches, achieving optimal results depending on the used algorithm	• Requires high computational capacity and long computing times, depending on the system's complexity • Limited to make real-time decisions • Time consuming to implement • Low flexibility
Agent-based or Multi-Agent Systems	• Rapid response to changes in the manufacturing environment • Highly flexible to meet new system configurations • Modularity, reconfigurability, adaptability, fault tolerance and extensibility characteristics • Supports distributed computing.	• Suboptimal results • Hard to implement • Highly dependent on communication resources between agents

Reference [39] emphasize that great advances in computational capacity have deployed the development of methods combining simulation and optimization, so that now simulation is used for its great detail potential while optimization techniques are used to find optimal or sub-optimal solutions. Regarding this, References [15,31] highlight the need for combining distinct scheduling approaches with A.I. to provide the scheduling system with the adaptability and robustness needed for handling the JSP.

3.2. Multi-Agent Systems

Most scheduling systems used in industrial environments currently operate centrally and hierarchically [36], that is, with decisions being made at a single point in the system and being passed down to the remaining stations from top to bottom. Although centralization may allow a reliable overview of the system state [31], the use of a centralized top-down approach for task distribution causes rigidity and limits real-world problem-solving capability [40,41]. Hence, a hierarchical and centralized scheduling is ineffective in highly dynamic environments where unexpected events are more frequent.

In this sense, Multi-Agent Systems (MASs) are seen among the most promising approaches to solve scheduling problems in these environments [18,19]. At the same time, the use of MASs offers significant advantages for industrial automation, including automated or semi-automated problem solving and capabilities of distributed control and processing [42]. The autonomous characteristics provided by the use of agents can still contribute to the development of autonomous processes, or even systems based on autonomous control.

For a system to be considered autonomous, two fundamental characteristics must be present: the capability of self-control and independence from its neighbor systems and from its environment [43]. Autonomous control, in turn, describes decentralized decision-making processes in heterarchical structures. It infers interacting components which have the ability to take decisions independently in non-deterministic systems [44], such as autonomous agents in MASs. However, these agents

(or elements) also need to be coordinated through specific policies [36,45]. Through coordination policies, it is possible to coordinate an innumerable number of autonomous agents towards a common goal. Thus, a good coordination policy, which can deliver effective adapting to changing circumstances and efficient global performance, is crucial for the system performance [45,46].

The most common coordination mechanisms used for agents are based on message passing, followed by the Blackboard [47]. Differently from message passing, in the Blackboard paradigm agents share a global memory instead of exchanging direct messages. This might avoid communication overhead that decreases agent's decision reaction to system' disturbances and causes them to employ more time acting on messages than performing their functions [7]. The use of a coordinating agent to control other agents is also an option to avoid these situations. With supervisory control introduction, the system is capable of generating integrated solutions with better overall performance [48–50].

Only local information is needed for a subsystem' autonomous control. For this, the logistic objects must be capable of receiving local information, processing it and deciding about their next action [43]. This is achievable with local agents. Besides, the logistic structure must supply distributed information on local states and different options for enabling decisions generally [51], which is possible with the use of a coordinating agent. Therefore, at the same time that distributed cooperation among agents and autonomous decision can enable self-x characteristics and high flexibility, the coordinating agent centralizes the feedback of the various system agents to achieve high global efficiency. With decision-making located where information arises, global information reaches its minimum. At the same time, scheduling turn into dynamic and resources like machines and products turn into intelligent objects that can interact for cooperation [43,52]. Thus, the system is decomposed into functional and simplified modular parts, leading to self-x characteristics like self-organization, self-adaptation and self-optimization [43].

By also using the Blackboard paradigm for indirect communication between local agents and its coordinator, the exchange of information directly between agents is reduced, avoiding communication overhead. Thus, the construction of a structure with these characteristics for dealing with the Job-shop Scheduling Problem would allow: (i) greater agility and flexibility to deal with stochastic events in the job-shop; (ii) less need for data distribution capacity; (iii) a simplified and modular view of the system, facilitating analysis, improvements and integration with other autonomous systems of the factory; and (iv) an autonomous system with autonomous control.

MASs Applications in Cyber-Physical Systems

MAS approaches are currently being used to build-up some typical CPS functions. Between these, one can cite [5,12,53] for Cloud computing, references [54,55] for Internet of Things (IoT) and [2] for a Service Oriented Architecture (SOA), besides MAS use as a means of integrating other system functions in References [3,56].

In particular, reference [12] highlights the use of a coordinator agent to process large amounts of data and take globally oriented decisions. The authors present an intelligent factory structure combining supervisory control terminals with intelligent shop floor objects and industrial cloud. Intelligent objects are machines, transporters and products, modeled as agents using negotiation mechanisms for cooperation. The coordinating agent, located in the cloud, processes a large amount of data collected from the plant, coordinating the distributed intelligent objects behavior and issuing alerts on the performance indicators for control terminals. Intelligent machines communicate their status and process information to this agent, while distributed sensors continually transfer their data to it.

Reference [54], on the other hand, present a concept using agents together with Raspberry Pi devices to form an IoT network. This approach, called AgPi, uses MAS to command various network activities. According to the authors, the Pi device is a low footprint and low-cost mini-computing device. Agents operate on each Pi device, making decisions autonomously. The authors point out that although RFID is generally a good option, it presents as major disadvantage the need for proximity

between transmitter and receiver devices. In this sense, the use of Bluetooth Low Energy (BLE) technologies, such as Pi devices, may offer a superior solution to RFID tags in cases where emitters and receivers are far apart.

Furthermore, reference [55] present an IoT infrastructure for the collaborative fulfillment of warehouse orders based on RFID, MAS and ambient intelligence. This structure consists of an Enterprise Application Integration (EAI) platform middleware (based on Universal Plug and Play protocol), a MAS created using JADE multi-agent platform, an ERP and a physical devices layer (trucks, pallets, forklifts and packing machines equipped with RFID technology). For comparison purposes with the aforementioned Pi devices, RFID technology can be divided in passive (tags) and active (readers). Tags cost only a few cents but are unable to perform computations such as Pi devices. RFID readers, which have a computational capacity and are used to read these tags, cost a few tens of dollars. Thus, the authors highlight that RFID industrial deployment reduces investment costs, encouraging companies to use a bottom-up approach to warehouse management. It is also worth noting that agents are not embedded in the products as the previous presented implementation but run on "normal" desktop systems or servers, exchanging information with the ERP and Middleware environment architectures using XML interfaces and EAI middleware.

MAS can still be integrated with SOA by forming up a service-oriented multi-agent system that not only shares services but also forms a distributed agent network integrating SOA principles [2]. In these systems, the front layer includes services that encapsulate the functionalities provided by agents. These agents, in turn, provide intelligence, autonomy and control. Reference [2] present and discuss four European industrial projects, seen as headlight projects that reflect a broader on transition thinking for the community about CPSs. Of these four projects, two in particular use agents in their implementation: FP7 GRACE (MAS in Manufacturing Automation) and FP7 ARUM (MAS merged with SOA for Production Management).

The FP7 Grace project integrates processes and quality control adopting MAS for implementing dynamic self-tuning procedures and feedback controls with the goal of improving production quality and efficiency. The designed agent-based solution was implemented through distributing the multiplicity of agents by 8 computers arranged on the production line, interconnected by TCP/IP protocols over an Ethernet network. The other MAS based example, ARUM FP7, applies mitigation strategies for acting more quickly and appropriately to unexpected events in the production of highly customized and complex products. For this, agents are used to developing planning, scheduling and production optimization tools, which display their internal functionality as services following SOA principles. An Enterprise Service Bus (ESB) architecture is used as a backbone to support interoperability between the MAS knowledge-based decision support tools developed. Although its potential, both projects are prototypes hardly replicable due to its very particular development characteristics and disregard to industrial standards.

The MAS' characteristics of modularity, autonomy, flexibility, robustness and adaptability can also be implemented to fulfill functions of integration between different factory operations. On this subject, reference [56] present an architecture for Agent-based MES dedicated to short-series production support. Both workflow and information exchange follow ISA-95 standard and are implemented in a dynamic, agent-based environment. Holons are a cybernetic part of this application, collecting information from the actual production system. Agents use internet services, which are also available to human users, for processing this information.

Finally, reference [3] present the Watchdog Agent (WA) application, which consists of different analysis methodologies to evaluate the deterioration process of machines and their components. The WA is a multi-sensor performance evaluation and prediction toolbox that enables quantitative evaluation and performance degradation levels analysis for critical components of machines [57]. The application is able to predict machine deterioration with self-conscious intelligence, avoiding potential problems or failures. To achieve this, the WA collects and processes a large amount of data from various production facilities and distinct platforms. For predictive analyzes, recognized

methodologies for processing, prediction and forecasting were implemented. Based on their knowledge of both process and equipment to be supervised, the agent becomes aware of the current situation, facilitating the dynamic adjustment of the tools available for predictive analysis. In addition, the agent is able to remember observed situations and, as a consequence, can identify circumstances that have never been observed before.

Another example presented in Reference [3] is the demonstrator myJoghurt, developed to evaluate new Cyber-Physical Production Systems (CPPSs) approaches. The demonstrator encompass a production process for daily consumer products, more specifically the manufacturing of mass-customized yogurt. The myJoghurt demonstrator performs the interconnection of locally distributed production facilities, establishing a CPPS network through an internet connection. These distributed facilities have the capacity of communicating with each other through defined protocols and interfaces based on MAS. After a configuration of a personalized yogurt recipe is completed by costumers, the request is forwarded to a provider agent that represents different supplier plants. This provider agent decides which vendor's factory will receive an assignment to supply its production services. In this way, the main task of the MAS and its decision making is to choose an appropriate factory (considering individual price/delivery period characteristics) to produce the yogurt product [58].

3.3. Industrial Standards for Data Exchange

The difficulty of integrating systems with multiple points of interest in production has led software developers to bundle multiple management components into unique and integrated solutions [59–61]. Among these solutions are two highly used in industry today: ERP and Manufacturing Execution System (MES).

ERP has emerged for many manufacturing companies as a standard in the market for generic information systems [62]. It provides a unified platform for managing the company's business in all areas where is possible to automate information [63], maintaining important business data and supporting business processes [60]. However, ERPs do not allow controlling production operations in real-time. Thus, decisions made at the strategic level of the ERP should be executed and monitored in real time by control systems [56]. In this sense, MESs are service-oriented interfaces connecting business with production operations [56]. Thus, MESs enable linking real-time control devices to high-level management, providing a common user interface and data management system (e.g., SCADA and Graphical User Interface (GUI) systems) [56,60].

Despite enabling the capture of data and monitoring machines conditions in real time [64], MESs generally have a static hierarchy, with a pre-established set of services and fixed interfaces for production facilities, making these systems very difficult to modify [56]. In addition, information integration difficulties can arise when data is exchanged between heterogeneous application systems, such as ERP and MES. Therefore, standard models and manufacturing information schemas are required, playing important roles in sharing information and integrating heterogeneous business applications [65].

Among these models, ISA-95 is highly recognized in the industry due to its focus on the integration between these two types of solutions (ERP and MES) and its high level of application in the industrial environment [63,66,67]. ISA-95, also known as IEC 62264, provides consistent terminologies, models and interfaces for integrating data exchange between different levels of a production system's hierarchy [67,68]. The standard defines the MES data structure and its services associated with manufacturing operations such as (i) product definition; (ii) production planning; (iii) production capacity management and (iv) production performance evaluation.

ISA-95 also defines the information exchanged between finance, sales and logistics processes and quality, production and maintenance systems [56]. It distinguishes these operations through a clear description of requirements and functions, at the same time that unites them through a well-defined communication interface. This communication interface is performed by an XML application of the

ANSI/ISA-95 family of standards named Business to Manufacturing Markup Language (B2MML). This application is based on UML (Unified Modeling Language) models, also primarily used for developing standard interfaces between MES and ERP systems [56,65].

ISA-95 defines processes as a five-level (0–4) functional hierarchy model, distinguishing two distinct domains inside a manufacturing company: the corporate (Level 4) and the manufacturing (Level 3 and levels below) domains. The corporate domain (Level 4) includes decisions on determining inventory levels and establishing the plant' basic schedule. Therefore the time frame considered for decisions on this domain comprises from months to days or shifts. The manufacturing domain on the other hand, defined by the standard as Manufacturing Operations Management (MOM), details all activities supporting manufacturing in quality, production, maintenance and inventory areas [60]. This includes activities such as following the workflow, keeping records and optimizing the production process (Level 3), monitoring and supervisory control (Level 2), sensing and manipulating the production process (Level 1), and the physical processes themselves (Level 0).

Thus, ISA-95 business processes are executed in the top layer (ERP), which transmits the data to the MOM layer, where the requests are scheduled and production plans are created [66]. The orders are then divided into many subtasks that are handled by distinct fabrication cells. This process creates production orders of the subtasks that are performed by the control level. Running production information is collected in a bottom-up manner.

Among the 5 parts that make up ISA-95, Part 3 encompasses activities related to production operations management, which include scheduling activities. In his part, the "Production information model" presented in Part 1 is expanded into four categories, with the aim of better detailing the flow of information in the company. In the activity model presented in Part 3 for production operations management, eight functions are presented for production operations management, namely: Detailed Production Scheduling, Production Resource Management, Production Dispatching, Production Tracking, Product Definition Management, Production Execution, Production Data Collection and Production Performance Analysis.

The potential of agents to perform different functions autonomously enables all the eight functions presented to be deployed in a MAS, including the process control functions presented in Level 0-1-2. The development of a MAS based on ISA-95 standards, in turn, facilitates its integration with other existing industrial solutions developed based on this standard, highlighting the importance of considering ISA-95' popularity for avoiding applicability issues. In addition, the use of standards contribute to achieving the high levels of automation desired in the CPS era, as well as facilitating the implementation of later modifications to adapt processes or to include new functions, potentially eliminating the difficulty of replicating existing MAS' solutions for other industrial cases.

Thus, considering the points mentioned, the proposed MAS framework for dealing with the JSP is presented in the next section.

4. A MAS Framework to Dynamically Deal with the JSP

4.1. Framework Proposal

Considering the information presented, as well as the models and information established in ISA-95 standard, the framework for handling the JSP dynamically is presented in Figure 3.

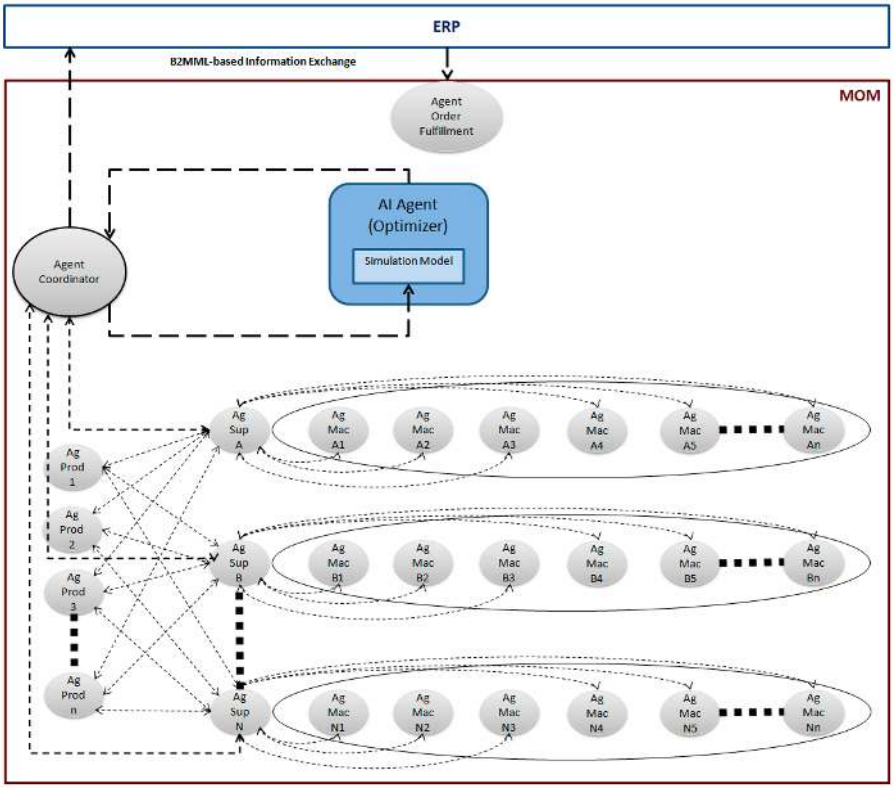

Figure 3. Novel framework for dynamically dealing with the JSP in CPSs.

The framework distinguishes between the two distinct domains considered by ISA-95 (corporate and manufacturing domains), exchanging data through them using the XML implementation known as Business to Manufacturing Markup Language (B2MML). Six types of agents are considered, with the following functions:

- Agent Order Fulfillment: This type of agent receives demand information from the ERP and separates them into orders, creating a new agent for each product to be manufactured (Ag Prod). For this, Agent Order Fulfillment must have data about the variety of possible products to be made, as well as the operations required for their production. This agent records priority numbers on these newly created Agents Product according to a given dispatching rule (order of arrival, priority, due time, etc.), which is used for determining their order in waiting queues;
- Agents Product (Ag Prod): Each agent of this type contains information of a given order to be made, such as due time, processes sequence, priority and so forth, according to the information provided by Agent Order Fulfillment. This agent still records on itself the operations already performed for its production;
- Agents Machine (Ag Mac): This type of agent provides data of interest on a particular machine, such as availability, queued products, production delays and so forth. It can still be used to send alerts of interest about the machine on which it is allocated, warning about breakages or possible defects, for example;
- Agents Supervisor (Ag Sup): This type of agent supervises a certain number of machines with specific characteristics, collecting information individually from each machine about availability, delays, performance and so forth. It can still have a dispatching rule to determine the best processing sequence of its supervised machines according to an objective function;
- Agent Coordinator: This agent receives the processing sequences created by each Agent Supervisor (based on these agents particular local information) and generates a new schedule based on this information considering the best global performance. Agent Coordinator can also be used for coordinating other agents exclusively using dispatching rules (and therefore does not

creating a schedule per se) and for providing periodically information needed by the ERP, such as order status, machine breaks, new scheduling and so forth;

- AI Agent: This agent has an A.I. algorithm that is applied in a simulation model, which uses the schedule provided by Agent Coordinator as input, to create an optimized schedule. This agent can use optimization or A.I. approaches for exploring the search space in an effective and efficient way—as pointed out in Reference [69]—or to build and learn an understanding of possible results' panorama while going in the direction of the solution space—hence reducing computational times.

For avoiding communication overhead, the Blackboard concept can be used through a wireless network for data exchange between all agents, as well as between the two domains considered. In this way, an Internet of Things environment is created for the agents.

In this environment, the information flow starts with the ERP system, which periodically sends a demand to Agent Order Fulfillment and is updated with data from Agent Coordinator. Agent Order Fulfillment splits the demand from ERP into several Product Agents, which in turn send their individual information to Agents Supervisor, which also receive local information from its supervised Agents Machine. Agents Supervisor exchange information with Agent Coordinator, which centralizes all Agents Supervisor data (and therefore, the current status of all factory's products and machines) to create a global schedule. Once this schedule is generated for a time "t", Agent Coordinator sends it to Agents Supervisor, which pass it to Agents Machine for being executed during a certain period of time "t". At the same time, Agent Coordinator provides its schedule as input for a simulation model, which is used by AI Agent to generate a partially new improved schedule to be executed from time period "t + 1". Once the schedule is improved by AI Agent, it is sent back to Agent Coordinator, which passes it again to Agents Supervisor, to be passed to Agents Machine for execution from time period "t + 1".

Figure 4 presents a UML representation of the message exchange between agents, exemplifying the proposed time periods "t" and "t + 1" for illustrating its concept. These parameters represent the time interval for executing the schedule defined by dispatching rules on Agent Coordinator and the time interval for improving this schedule through the AI Agent, respectively.

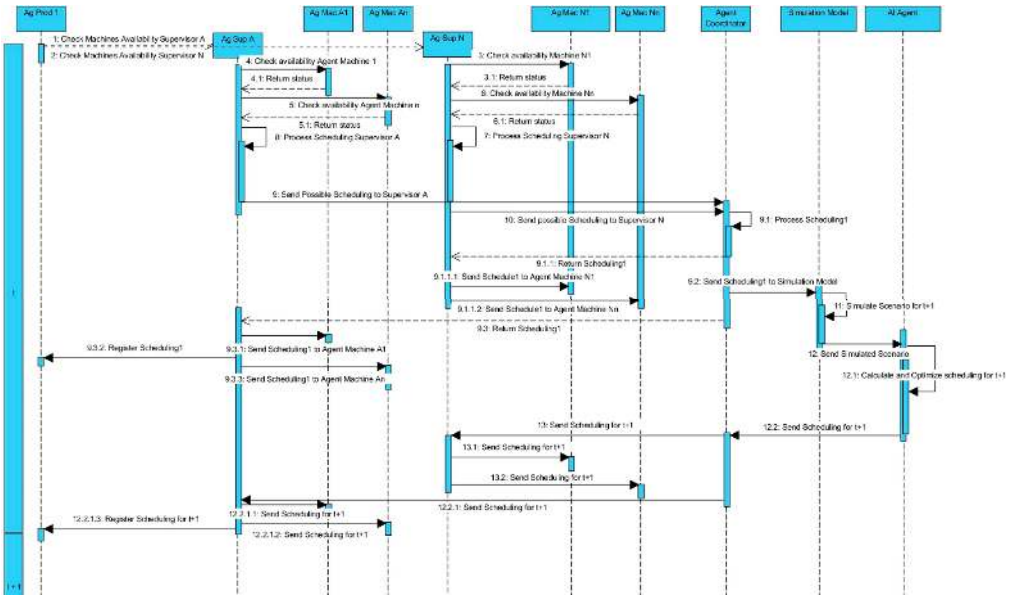

Figure 4. UML sequence diagram of the data exchange between agents.

The main variable to be considered when implementing the framework is the time "t" to create the schedule. In factories with high production and complexity, the AI Agent might take a long time to generate a new schedule. In this case, selecting an algorithm that reaches a suboptimal solution faster might be more appropriate or even using only dispatching rules (thus eliminating AI Agent). On the

other hand, factories with low production and larger processing times can obtain great benefits with the scheduling improvement made possible by AI Agent. Another important variable to consider is the periodicity of information exchanged between MOM and ERP. A simulation model is recommended for choosing the most appropriate parameters for each case in the framework implementation.

In this way, the proposed framework can be classified as a dynamic approach with the use of dispatching rules in an Agent-based decision mode, which can be extended to use simulation and artificial intelligence (through the use of the AI Agent). Thus, it is possible to choose between heuristics, optimization and artificial intelligence approaches, being the first recommended for lower level agents (Agents Order Fulfillment, Agent Product and Agent Machine) and the second and third for upper level agents (Agent Supervisor, Agent Coordinator and A.I. Agent). In particular, regarding the taxonomy proposed by [39], the use of AI Agent with a simulation model is classified as Optimization based on Simulation-based Iterations (OSI). The framework can still be implemented with any of the three rescheduling range policies (right-shift, complete and partial) highlighted in Reference [70], as well as any of the three updating options presented in References [15,31,71] (periodic, event-based or hybrid), depending on the company's interest. In particular, for using right-shift policy it is mandatory the use of an AI Agent for complementing Agent Coordinator' functionality, based on dispatching rules.

The proposed framework also has the autonomy characteristics presented in Reference [43]: considering the scheduling as a separated system inside the factory, the framework implementation acts as an independent decision-making module able to perform actions by itself (e.g., generate and implement schedules). At the same time, the characteristics of autonomous control are achieved by combining the framework' heterarchical structure with its decentralized decision-making possibilities, seen on the agents' functionality. Thus, the advantages related to autonomous systems might be achieved through the framework implementation, for example greater agility and flexibility to deal with stochastic events and less need for data distribution capacity. Similarly, the implementation provides a simplified and modular view of the scheduling system, thus facilitating analysis, improvements and integration with other autonomous systems of the factory. Once built for the job-shop, the system can be extended for integrating intelligently the remaining areas of the company in a modular way by adding new functions to the high level agents (or by creating new types of agents).

4.2. Theoretical Implementation

As previously mentioned, the development of a simulation model (or at least a computational/knowledge model, as cited in Reference [7]) is a step of great importance to be considered previously to the framework implementation in the real job-shop scenario. The simulation model makes it possible to (i) analyze the data collection and processing parameters to define the best suited to the company's objectives; (ii) verify the possible improvements of performance coming from the framework implementation; (iii) find the balance between desired performance and investments required in sensors and data processing devices; (iv) facilitate the transition of the company from its current state to the state of effective implementation, anticipating future necessary adjustments and (v) testing and choosing the dispatching rules that better fit the company' needs. In addition, the AI Agent application requires a virtual environment to emulate the real one on which the agents will make decisions, which can be provided by the simulation model developed. Figure 5 presents a concept for virtualization of the job-shop through a simulation model.

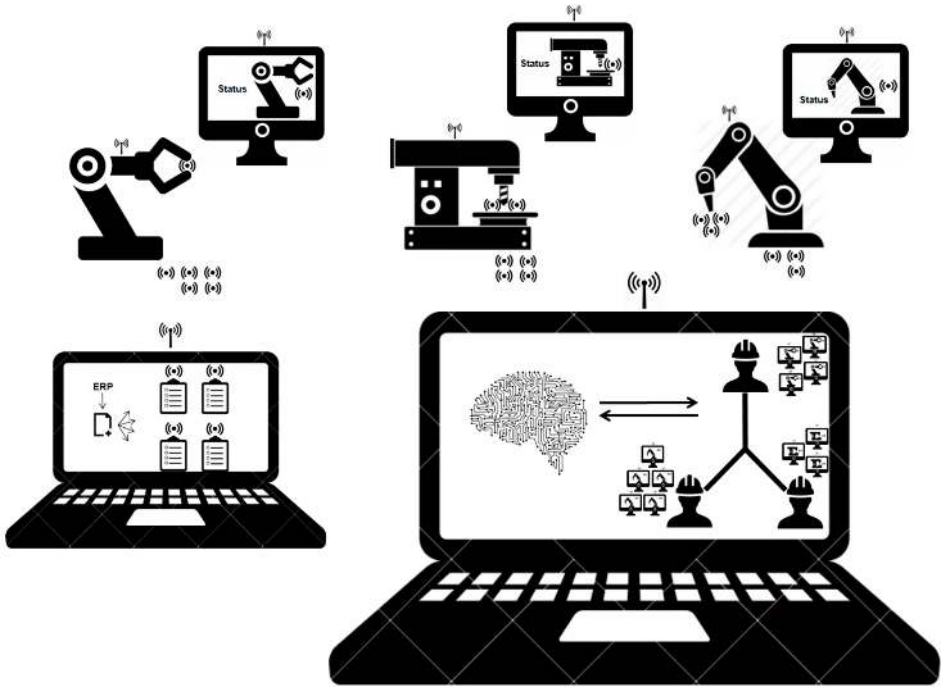

Figure 5. Job Shop Virtualization.

Two processing cores are shown in Figure 5, visualized in the form of computers. The first one, on the left, executes the function of Agent Order Fulfillment, receiving ERP' production orders and separating them into orders to be made. A sensor is allocated for each newly created order, represented individually in the virtual environment by an Agent Product. The sensors follow physically the order manufacturing through the factory, sending individual information about Agents Product in production (e.g., in processing, finished, in the queue, etc.). These Agents Product, in turn, have their production sequence defined by the second processing core, which centralizes all the information collected from the job-shop, including data from products' and machines' sensors, for scheduling (through Agents Supervisor, Agent Coordinator and AI Agent).

The physical machines are represented individually in the virtual environment by Agents Machine, exchanging information with Agent Product' sensors to change their state in real-time according to their own actions on it (e.g., put it in the queue, process it, make it unusable, etc.). Each Agent Machine is configurable using its own monitored machine-specific parameters (e.g., processing times, required setup time, failure rate, etc.). The machines have bidirectional data exchange with the second processing core, sending information about their own status and performance, as well as data about changes performed on Agents Product. Through a GUI interface, Agents Machine can make available in the job-shop real-time data about production, next products to be processed or indications of its own individual machine performance.

Regarding standardization, special attention should be given to the XML file exchanged between ERP and MOM domains, which should follow the nomenclature proposed by ISA 95 standard in its B2MML communication interface. Also, in relation to ISA-95, it is worth mentioning that each type of agent can incorporate the following production functions indicated by the standard:

- Agent Order Fulfillment: Production Resource Management and Product Definition Management;
- Agents Product: Product Definition Management, Production Tracking and Production Data Collection;
- Agents Supervisor: Production Resource Management, Production Dispatching, Production Execution, Production Tracking, Production Data Collection and Production Performance Analysis;
- Agents Machine: Production Resource Management, Production Dispatching, Production Execution, Production Tracking, Production Data Collection and Production Performance Analysis;

- Agent Coordinator: Detailed Production Scheduling, Production Dispatching, Production Data Collection and Production Performance Analysis.
- AI Agent: Detailed Production Scheduling.

In relation to data collection technologies, it is highlighted the greater suitability of Bluetooth Low Energy (BLE) devices, such as the Raspberry Pi presented in Reference [54], for being used in Agents Product. This technology is useful due to its potential to send information even at larger distances of receiver devices, as well as performing computing activities. At the same time, RFID readers are more suitable for Agents Machine, because of the proximity between machines and products at the time of processing and possible savings made by using RFID tags into products. In this case, the Agent Product functions, which no longer could be computed due to the RFID tags' lack of computational capability, can be divided between the Order Fulfillment and Machine agents. Finally, another option is using agents installed on computers with GUI interfaces for manual updating and communication through TCP/IP protocols using an Ethernet network. The choice of where to use Graphical User Interfaces (GUI), or even the range of functions adopted by the agents should also be adequate according to the implementation objectives. Other solutions previously mentioned may also be good options for dealing with time bottlenecks (e.g., cloud computing, as presented in Reference [12], or parallel computing, as presented in Reference [53]) or for integrating MOM with other plants (e.g., myJoghurt [3], SDWs [53] or IT systems [56]).

To better understand the framework implementation in a real job-shop, as well as its potential performance and adaptation capacity to attend factory' interests, an initial application developed considering a real case is presented in the next section.

5. Framework Implementation in a Real Case

5.1. Case Study

The case study chosen for the framework evaluation is a company that works primarily with customers in the automotive industry. This particular company produces about 430 types of products, manufactured on its about 800 manufacturing machines. The company produces for 495 h per month on average and delivers to customers four times per month, thus every 123.75 h of production.

Currently, the production scheduling of the industrial plant considered is created by a software named Preactor. Preactor is a finite capacity planning system, with a highly configurable and modular structure [72]. Reference [6] present a survey about production scheduling systems currently used in the industrial sector, offering an analysis of 16 systems including Preactor. The authors point out that, compared to other software, Preactor lacks the functionality to display information received from the sensors in the manufacturing system. Preactor is especially useful for generating scheduling from resource information, demands and constraint information (through its integration with an ERP system or through its bill of materials module, according to [72]).

In the case study analyzed, Preactor operates on 3 months demand forecasting, integrating scheduling functions with the factory' ERP. The content of this forecast used by Preactor includes information on the quantity of each product to be manufactured and its due dates. Based on this forecasting, the software creates a schedule for one month of production. Preactor takes 8 h on average to calculate this production schedule for the whole factory. Besides using Preactor for monthly scheduling, the company still uses it for activities such as: optimize the purchase of materials and productive resources, plan the production with preventive maintenance and control the outsourced operations.

Currently, the plant does not have an automated production data collection system to monitor the execution (such as an MES) of the generated schedule, being the production data updated manually from time to time during the month. This occasional update of production data creates difficulties in tracking the factory's production status, resulting in divergences between the available information

and the current production status. To minimize this problem, there is a monthly effort to update all production data before creating a new monthly schedule.

The factory currently experiences large fluctuations in the number of work-in-progress (WIP) over a month, validated by employees who explain that there is a greater allocation of production resources at the beginning of the month. In addition to generating a high volume of inventory, these oscillations also casually demand the use of extra hours at the beginning of the month and result in idle periods at the end of it. It is also observed that most part of the effort in scheduling is intended for minimizing the number of setup activities performed. Such planning is due to the lengthy time of setup activities, which can take up to 4 h, while the processing time of products takes less than 20 s on average.

5.2. Framework Application

The factory evaluates the possibility of implementing the proposed framework to perform a dynamic scheduling of the production, integrated to the existing ERP system. Thus, manual update of production data would be automated and the scheduling via Preactor is eliminated (although not its use for other of its mentioned functions).

The framework application was tested in a simulation model developed in Anylogic for an automotive transmission forks production line of this factory. The model developed in Anylogic sought to mimic as close as possible the current situation of the analyzed production line, using real data of the monthly volumes of production and sequence of operations needed for manufacturing each type of product considered, the quantity and type of machines available and their respective production capacities and average times used in processing and setup activities. Then another scenario was developed, implementing the agent-based framework for carrying out the scheduling activities, thus replacing the schedule generated by Preactor. Finally, simulations made using these two computational models allowed evaluating the potential performance of the framework implementation, which was compared with the current performance obtained in the factory through Preactor adoption.

The line considered has 9 types of machines with distinct functions involving the operations of turning, broaching, heat treatment, straightening, cracking, drilling, grinding, assembly and inspection. There are two parallel machines for broaching and heat treatment and one machine for each other operation so that the total number of machines in the analyzed production line is 11. Except for the heat treatment machines, which are shared for manufacturing also other products than those analyzed in this paper (Other Products), all machines are exclusively dedicated to five specific product types, denominated as products 414, 415, 416, 419 and 422.

For this initial implementation, AI Agent is not used and Agent Coordinator is used only for coordinating other agents exclusively through using dispatching rules. Therefore, there is neither creation nor optimization of a schedule. Agent Order Fulfillment, on the other hand, is used for creating Agents Products, communicating exclusively with the ERP system, which provides demand data. Figure 6 shows a schematic overview of the communication between agents for this framework application.

The simulation model starts with the reading of forecasted data, as it currently is by Preactor. This information includes the quantity of each product to be made and the start and due dates of production. Agent Order Fulfillment reads this data and creates new agents (Agent Products) for each order to be manufactured, individually recording in these agents a priority number, which will be used later to choose its individual priority in each machine queue (by Agent Machines). At each completed production step, Agents Products individually request to Agent Coordinator information on which of the machines available should be chosen for its next processing operation. To make the choice of this machine, Agent Coordinator collects real-time information about the status of all job-shop agents, provided by Agents Supervisors.

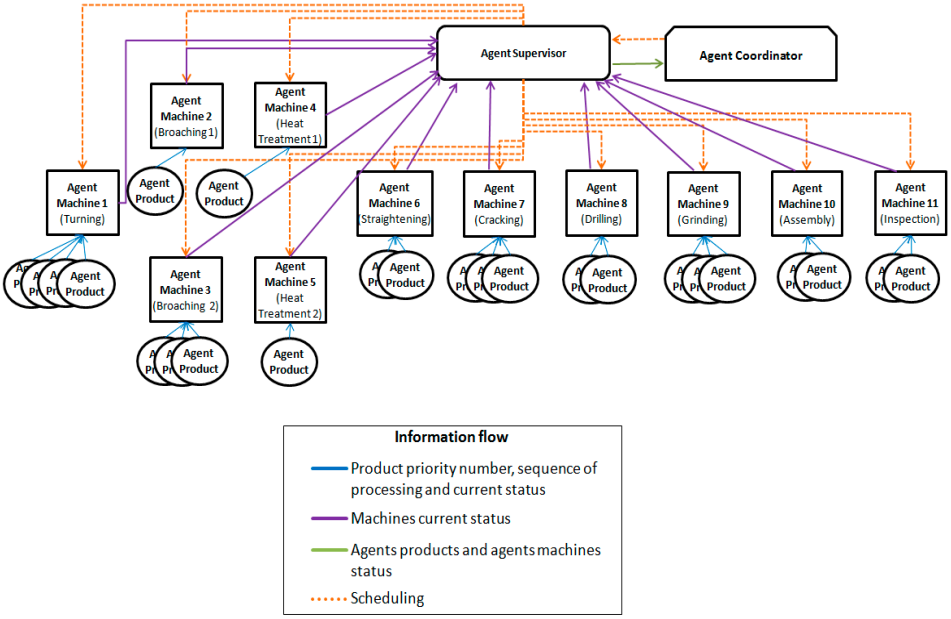

Figure 6. Implementation of the framework for the Case Study analyzed, presented preliminarily in Reference [23].

Two steps were chosen to compose the dispatching rule applied by Agent Coordinator for coordinating the agents in the considered case. In the first step, Agent Coordinator verifies the current machines setup type (each type of product has its own setup type). For this, this agent verifies which is the next production operation to be performed on each product and which machines are available to carry out this step, checking between the available machines if one is already configured to produce the considered product. If so, the product is sent directly to this specific machine. This rule is used to minimize the number of setup operations. The second step is applied only when the first step results negative (that is, there are no preconfigured setups in any of the machines available for the analyzed product). In this case, agents Products are sent to the machines with the lowest number of products in the queue. This rule is applied to balance machines utilization.

Finally, Agents Machine modify the production order of products waiting in their own queues, passing products with higher priority forward and sending products with lower priority to the end of the queue. Therefore, Agents Machine consider only the priority of the orders in their respective queues and not the setup for which they are configured. Only one Agent Supervisor is required in the case study chosen since only one production line is considered. This Agent Supervisor collects data about the availability, queue size and current setup of all available machines, making this information available to Agent Coordinator. Agent Supervisor also creates and removes Agents Machine to adapt the model to changes in the number of machines available on the production line (as would happen in the case of a machine breakdown for example).

In all, 4 different scenarios were simulated for the framework application, all of them involving changes in the logic used by Agent Order Fulfillment to create new Agents Products. In the first scenario, Agent Order Fulfillment creates Agents Products following exactly the same overproduction volume as defined by Preactor. In this way, the performance difference between Preactor's scheduling and this Scenario 1 is given by the priorities defined for each product type (which impacts the choice of Agents Machine among those products waiting in their individual queue) and by the dispatching rule applied by the Agent Coordinator to select a machine for each Agent Product's production. The priority set values for each product types are 6, 5, 4, 3, 2 and 1 for products of types 414, 415, 416, 419, 422 and other products, respectively.

In the remaining three scenarios, Agent Order Fulfillment uses the monthly forecasted demand to define the start the production of Agents Products at different periods of the month. An interval of

1 h has been defined between the creation of different types of products. In Scenario 2, Agent Order Fulfillment established the creation of the full weekly demand at the beginning of the week. In Scenario 3, Agent Order Fulfillment divides the weekly demand into two, creating Agents Products twice a week, every week. Finally, in Scenario 4, Agent Order Fulfillment divides the weekly demand into three, creating Agents Products three times a week, every week. Thus, the difference between these three scenarios is the volume and the moment when the products are placed in production during the month.

5.3. Framework Performance

Figure 7 shows the current situation of the analyzed production line compared to 4 new scenarios resulting from the framework applied variations for a typical month of production. For comparison purposes with the framework new scenarios, the real industrial case is called Test Case.

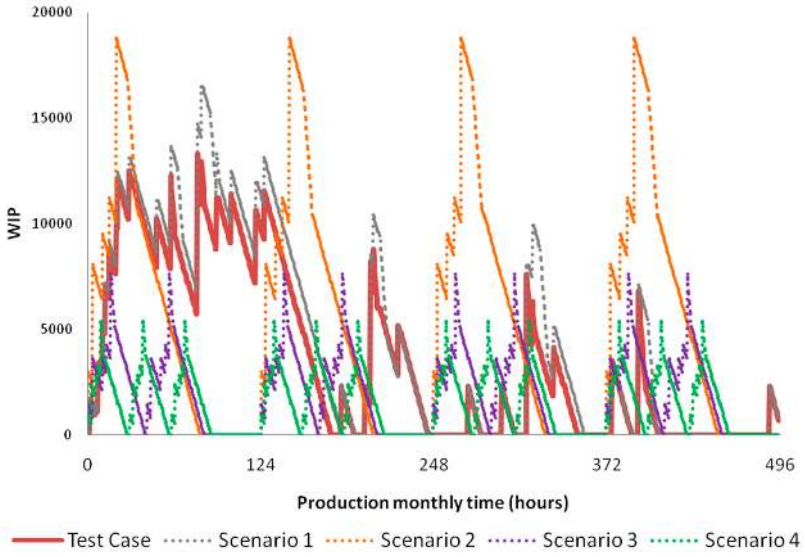

Figure 7. Scenarios performance regarding WIP, as presented preliminarily in Reference [23].

The maximum WIP values for the 4 new scenarios analyzed were 16.469, 18.776, 7.589, 5.364 for Scenarios 1, 2, 3 and 4 respectively, compared to a maximum WIP of 13.260 for the Test Case. In addition, the number of hours in which the WIP value was 0 (that is, the whole line was idle) is 183, 170, 176, 168 and 163 for the Test case, Scenario 1, Scenario 2, Scenario 3 and Scenario 4, respectively. It can be seen from the Figure 7 that Scenario 1 had little difference in WIP in relation to the Test case, following a very similar behavior during the month. This was the only new scenario that ended the analyzed period with a not null WIP value, with a total of 686 products of type 419 still being processed, compared to a total of 682 products of the same type for the Test Case.

In relation to the number of setup activities performed during the analyzed period, Figure 8 presents the performance of each new scenario in relation to the Test case.

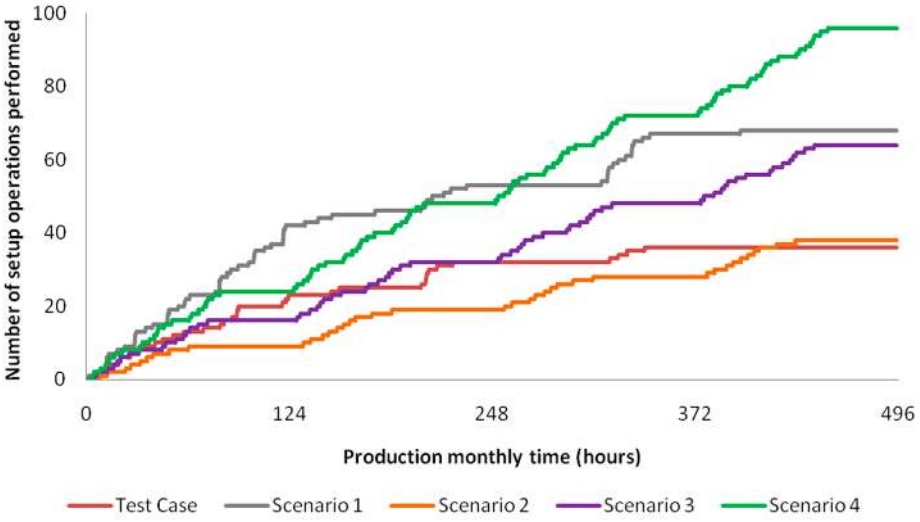

Figure 8. Performance of the scenarios in relation to setup operations, as presented preliminarily in Reference [23].

In all, 68, 38, 64 and 96 setup operations were performed for Scenarios 1, 2, 3 and 4 respectively, compared to 36 setup operations performed in the Test Case.

It was also analyzed the attendance to customers' demand, in comparison with the number of products made each week, shown in Figure 9.

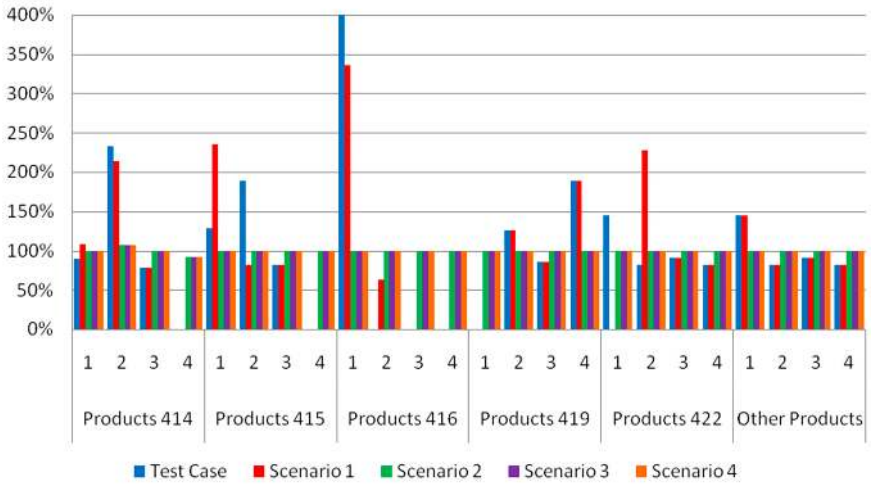

Figure 9. Scenarios performance in relation to weekly demand, presented preliminarily in Reference [23].

It is worth mentioning here that all the analyzed scenarios were able to meet the total monthly customers demand during the considered period, with a minor exception for the Test Case and Scenario 1 which had some products of type 419 pending on the end of the month (less than 0.05% of the monthly demand were not delivered in these scenarios). The production of each product type also varied considerably mainly in these two scenarios, while Scenarios 2, 3 and 4 performed identically. Finally, Figure 10 presents the average utilization indices of the 11 total machines available in the considered job-shop. It is worth mentioning here that setup operations were considered a no-utilization of the machine for calculating this average.

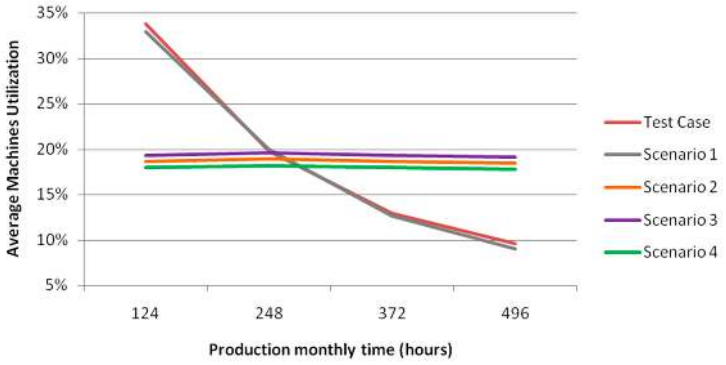

Figure 10. Scenarios performance in relation to machines utilization, as presented preliminarily in Reference [23].

Again the Test Case and Scenario 1 had similar performances, while the remaining three scenarios also had similar performances when compared to each other.

Finally, one machine was subtracted from the simulation model for analyzing the adaptability of the agents to a machine breakdown situation. Scenario 2 was chosen for this hypothetical case due to its higher average WIP. It was simulated the breaking of a heat treatment machine in the tenth hour of the analyzed period, with results shown in Figure 11.

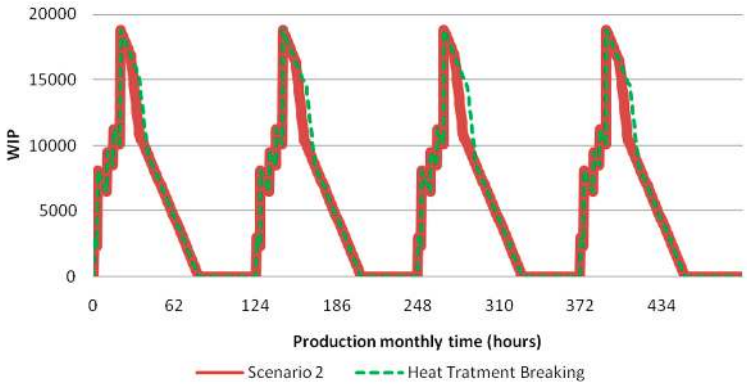

Figure 11. Hypothetical machine breaking situation, as presented preliminarily in Reference [23].

A small change in performance in relation to the no-machine-break scenario is seen in Figure 11, which affects production in a similar way during the 4 weeks analyzed. It is worth noting here that the perceived difference in performance between the two scenarios analyzed in this experiment is solely due to the reduction of production capacity caused by the individual machine breaking. This finding is verified analyzing that the same behavior of decrease is seen in the 4 weeks presented in Figure 11, which means that the breaking effect occurred at 10th hour did not affect the agents' performance to adapt and choose the next available machine to perform the same operation.

5.4. Discussion and Implementation

5.4.1. Technical Details of Implementation

Considering the analyzed industrial case, a TCP/IP network based on a three layers structure will be used for implementing the framework, namely: (i) physical devices layer; (ii) middleware layer and (iii) upper layer.

The physical devices layer covers all physical devices built in with data exchange technology. In this particular case, we opted for RFID technology based on readers and tags, given its high reliability, low application costs and relative simplicity. It is worth highlighting here the six classes used for dividing RFID tags, which increase in the order of complexity of functions and price [73,74]. Class 0

tags are read-only, while Class 1 are writeable once. Class 2 devices can be writing as many times as it is needed, while in Class 3 there is the addition of sensors for recording parameters of interest, such as temperature. Finally, there is the addition of transmitters in Classes 4 and 5, which allow sending signals independently to the readers (in class 4) and communication with other devices or other same class tags (in Class 5).

To perform the functions chosen for the Agents Products, which include the Production Data Collection function in the case analyzed, it is necessary to use tags of at least Class 2. The low amount of information needed for performing the Production Data Collection enables using tags with low storage capacity (for cost reduction), such as 65 kbs. Here the authors still analyze the possibility of using these tags in batches of products, rather than individually per product, for further minimizing implementation costs. This possibility still will be evaluated in detail during the framework implementation.

For the middleware layer, there are several commercial or open source tools available for developing MAS applications. These tools enable agent classes with specific attributes, as well as creating a virtual environment where these agents can send and receive information. Attendance to FIPA (Foundation for Intelligent Physical Agents), a reference organization responsible for establishing and developing software standards for agents and systems based on agents, is a relevant indicator for choosing the software to be used [40]. Among the available options, reference [75] cite as examples of software developed according to FIPA standards: FIPA-OS, Java Agent Development Environment (JADE), Manufacturing Agent Simulation Tool, ZEUS, MAST and GrassHopper.

For the case analyzed, we intend to use the JADE open source platform, which has an extensive application in the literature [1,55,76,77] and is referred as a simple, compact, graphically friendly [48,78] and easy to use [79] platform. The adoption of this software in each desktop computer used in the workstations will allow a GUI interface to the workers, who will be able to follow in real-time the current performance of system' agents and possible scheduling changes.

Finally, the Upper layer connects the local MAS implementation in JADE to the remaining organization areas, receiving ERP demand information and providing real-time production data in an ISA-95 based XML file (namely the B2MML application already mentioned). Regarding the A.I. Agent implementation in particular (not yet considered in this phase of the case analyzed), this layer may also contain the functionality performed by this type of agent, using a simulation model for optimizing an existing schedule. In this case, the software Anylogic, used in this work, also supports the exchange of data in XML format, which would facilitate the implementation and use of this software to perform the optimization based on simulation.

Finally, the choice made for the 65 kb Class 2 tag also allows estimating the maximum data flow exchanged at the shop floor level for the analyzed production line. Considering the highest WIP of all considered scenarios (about 19,000 products in Scenario 2) and a network with a capacity of 10 MB/s, the value of this exchange would reach approximately 1206 MB of data (65 kb $\times$ 19,000/(1024 kb/MB)), which would require about 2.01 minutes (1206 MB/s (10 MB/s) $\times$ (60 s)) to load all factory data in the event of a total system abrupt restart/collapse. Once the system is fully operational, the data exchange flow is much less intensive, since only changes in agents states (functioning/not functioning or manufactured/to be manufactured, for example) are incorporated into the system' data stream.

5.4.2. General Discussion

The application of a simulation model to an industrial case to test the framework performance allows the perception of some interesting insights. Through simulation, the WIP oscillation in the current industrial scenario is verified, highlighting the current use of larger stocks levels. It was seen that, although the option to work with higher stock levels favors reducing the number of setup operations performed, it results in less flexibility in adapting to changes in customer demands. With the framework application, this flexibility situation is changed. Even Scenario 1, which followed the same volume of production as defined by Preactor (but using dispatching rules for controlling the

production), could satisfy a change in customer demand by inserting this information into Agent Order Fulfillment. Once the framework uses dispatching rules for the production, including a new demand could cause these products started to be produced immediately (by increasing the priority value thereof), or even to pass these to the end of the current production week queue (by using a priority value lower than the one of the products being made). This example shows the importance of the autonomy provided to agents by a priority rule. In the scenarios tested a static value was determined but the framework application enables the inclusion of dynamic values for prioritizing each product type. This could involve such a rule that the priority value of a product corresponds to its own due time to delivery (which would probably lead to a greater number of setup operations, as seen in Scenario 1 performance in relation to this variable).

In the 3 remaining scenarios, the model flexibility can be better visualized. In Scenario 2, the option to produce the weekly demand at once allowed a better performance in relation to machine setups operations, which were very close to the best performance seen in the Test case. This option, however, generates a high number of WIP, although it generates a lower inventory value of finished products in relation to the Test Case and Scenario 1. In Scenarios 3 and 4, on the other hand, the value of WIP is reduced dramatically at the cost of an increase in setup operations. However, it should be highlighted that very simplistic dispatching rules were used in this initial implementation so that the full potential of the framework might increase considerably with more advanced and appropriate dispatching rules, as well as with the use of an AI Agent. Nevertheless, Scenario 4 WIP value deserves to be mentioned, which had its peak 67.4% lower than that verified in the Test case. Updating the demand every 1/3 week with the company's ERP also enables a responsiveness up to 12 times superior in relation to current performance to variations in customer orders, when demand is only considered to create a new scheduling once a month.

The framework implementation also highlighted agents' autonomy to adapt to a broken machine situation, by instantly queuing the related products to the next available machine. This is due to the advantages provided by real-time data exchange and the agility made possible by dispatching rules. Furthermore, it is worth noting that monthly demand was practically satisfied in all scenarios, with a high number of hours of production in which the entire line analyzed was idle and a very low average value for machines utilization. As a consequence, it is evident that the production line analyzed can reduce the number of hours allocated to production or share its machines for manufacturing other products, as already happens with the heat treatment machines. In the case of sharing, it is clearly preferable to perform a constant performance as seen in Scenarios 2, 3 and 4 to an unstable one, as currently verified.

Some insights related to industrial applications can still be pointed out. First, it turns out that solving the JSP involving only reducing the number of setup activities performed should not be taken as a golden rule for all industrial cases. The analysis showed that the effort to reduce the number of setup activities performed in the analyzed production line results in higher WIP volumes in the factory, leading to the production of final product inventories up to 400% higher than the weekly delivery demand. In addition to resulting in the use of larger areas to accommodate these products, which in turn also generate capital opportunity costs, these large inventories minimize the line flexibility to meet changes in customer demand.

In this sense, it was visualized that it is possible to attend all the weekly demand of the line through the balance of production, even with the increase in the number of setup operations performed. In this case, real-time data collection and the use of dispatching rules made it possible to balance the use of the machines and the adaptation to unforeseen disturbances, such as the breaking of a machine.

Thus, the emergence of new perspectives provided by technological development brings light to the exploration of other possibilities. As seen in this case, better resource utilization and adaptability both to job-shop variations and customer demand may be more beneficial to industrial performance than the reduction in the number of setup activities performed. The improvement of these indicators

of performance is benefited by data-driven technologies, which allow real-time production monitoring and decision making.

It is still evident the utility of a simulation model for strategic decision making, facilitated by information digitization. Through the simulations, it was verified that, despite high setup times and large production oscillations, the biggest line problem is not directly related to the number of machines nor the capacity available. In this sense, data-driven decision making still has much of its potential benefits highly dependent on identifying what is more important to each particular case. Even with the possibility of meeting customer demand changes involving a prioritization policy, for example, there are trade-offs that must be considered (such as performance reductions seen in the increase of setup operations, as in the case analyzed). Therefore, as big as the potential benefits in CPS era are the factory's management choices among several trade-offs available.

6. Conclusions

This paper proposed a novel conceptual framework for dynamically dealing with the Job-shop Scheduling Problem through a Multi-Agent System architecture. The proposed framework is highly flexible to several industrial contexts, having characteristics of dynamic adaptation and autonomous control, which in turn can provide greater agility and flexibility to deal with stochastic events in job-shops. It also considers the use of current industry standards, in order to facilitate its implementation and subsequent modification in real industrial cases. The application of the framework was simulated for a real industrial case, which enabled the verification of its potential in relation to an existing application. In the analyzed case the simulations show that the use of the framework brought more flexibility and agility to deal with stochastic effects in the JSP, maintaining a good performance compared to the current scheduling system even with the use of simple dispatching rules. For future work, we intend to continue with the framework implementation in the analyzed line, which will make it possible to deepen the analysis regarding costs and viability, data flow, system reactivity to different types of events, possibility of using economical lots of tags for reducing implementation costs and evaluate the need of using the A.I. Agent for optimizing the scheduling on this production line.

We still identify two distinct paths in relation to future research on the combination of MAS and data-driven technologies in the CPS era. The first relates to the use of these technologies to bridge the boundaries of digitization and decision-making beyond the job-shop, including the company's remaining operations as well as its integration with other organizations. Due to the agents' autonomy characteristics, its use can be disseminated to fulfill the most distinct operations related to the organization and its supply chain. Another possibility is exploring the use of data collected in the job-shop to improve the performance of agents' decisions. The use of techniques related to the use of collected data, such as those involving artificial intelligence and big data, for example, can help agents "learn" to make better decisions based on past results. Potentially, the combination of data-driven technologies and MAS makes it possible achieving all expected typical advantages of the CPS era, ranging from taking the most diverse types of decisions to its implementation and subsequent control and monitoring.

Author Contributions: Investigation, M.E.L.; Conceptualization, E.M.F. and M.U.M.; Funding Acquisition, E.M.F. and M.F.; Visualization, M.E.L.; Writing-Review & Editing, M.E.L., E.M.F., M.U.M., M.K. and M.F.; Supervision, E.M.F., M.U.M. and M.F.; Writing-Original Draft Preparation, M.E.L. and M.F.; Project Administration, E.M.F., M.K. and M.F.

Funding: This work was funded by CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Brazil) under reference number 99999.006033/2015-06 and by the German Research Foundation (DFG) under reference number FR 3658/1-1, in the scope of the BRAGECRIM program.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Leitao, P.; Rodrigues, N.; Barbosa, J.; Turrin, C.; Pagani, A. Intelligent products: The grace experience. *Control Eng. Pract.* **2015**, *42*, 95–105. [CrossRef]
2. Leitao, P.; Colombo, A.W.; Karnouskos, S. Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges. *Comput. Ind.* **2016**, *81*, 11–25. [CrossRef]
3. Vogel-Heuser, B.; Lee, J.; Leitao, P. Agents enabling cyber-physical production systems. *AT-Autom.* **2015**, *63*, 777–789. [CrossRef]
4. Lee, E.A.; Seshia, S.A. *Introduction to Embedded Systems: A Cyber-Physical Systems Approach*; MIT Press: Cambridge, MA, USA, 2016.
5. Boulekrouche, B.; Jabeur, N.; Alimazighi, Z. Toward integrating grid and cloud-based concepts for an enhanced deployment of spatial data warehouses in cyber-physical system applications. *J. Ambient Intell. Humaniz. Comput.* **2016**, *7*, 475–487. [CrossRef]
6. Toader, F.A. Production scheduling in flexible manufacturing systems: A state of the art survey. *J. Electr. Eng. Electron. Control Comput. Sci.* **2017**, *1*, 1–6.
7. Xiang, W.; Lee, H.P. Ant colony intelligence in multi-agent dynamic manufacturing scheduling. *Eng. Appl. Artif. Intell.* **2008**, *21*, 73–85. [CrossRef]
8. Hall, N.G.; Potts, C.N. Rescheduling for new orders. *Oper. Res.* **2004**, *52*, 440–453. [CrossRef]
9. Asadzadeh, L. A local search genetic algorithm for the job shop scheduling problem with intelligent agents. *Comput. Ind. Eng.* **2015**, *85*, 376–383. [CrossRef]
10. Vinod, V.; Sridharan, R. Scheduling a dynamic job shop production system with sequence-dependent setups: An experimental study. *Robot Comput. Integr. Manuf.* **2008**, *24*, 435–449. [CrossRef]
11. Kundakcı, N.; Kulak, O. Hybrid genetic algorithms for minimizing makespan in dynamic job shop scheduling problem. *Comput. Ind. Eng.* **2016**, *96*, 31–51. [CrossRef]
12. Wang, S.Y.; Wan, J.F.; Zhang, D.Q.; Li, D.; Zhang, C.H. Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. *Comput. Netw.* **2016**, *101*, 158–168. [CrossRef]
13. Leitão, P. Agent-based distributed manufacturing control: A state-of-the-art survey. *Eng. Appl. Artif. Intell.* **2009**, *22*, 979–991. [CrossRef]
14. Kouider, A.; Bouzouia, B. Multi-agent job shop scheduling system based on co-operative approach of idle time minimisation. *Int. J. Prod. Res.* **2012**, *50*, 409–424. [CrossRef]
15. Merdan, M.; Moser, T.; Sunindyo, W.; Biffl, S.; Vrba, P. Workflow scheduling using multi-agent systems in a dynamically changing environment. *J. Simul.* **2013**, *7*, 144–158. [CrossRef]
16. Shaw, M.J. Dynamic scheduling in cellular manufacturing systems: A framework for networked decision making. *J. Manuf. Syst.* **1988**, *7*, 83–94. [CrossRef]
17. Shen, W.; Hao, Q.; Yoon, H.J.; Norrie, D.H. Applications of agent-based systems in intelligent manufacturing: An updated review. *Adv. Eng. Inform.* **2006**, *20*, 415–431. [CrossRef]
18. Lou, P.; Ong, S.K.; Nee, A.Y.C. Agent-based distributed scheduling for virtual job shops. *Int. J. Prod. Res.* **2010**, *48*, 3889–3910. [CrossRef]
19. Shen, W. Distributed manufacturing scheduling using intelligent agents. *IEEE Intell. Syst.* **2002**, *17*, 88–94. [CrossRef]
20. Leitão, P.; Mařík, V.; Vrba, P. Past, Present and future of industrial agent applications. *IEEE Trans. Ind. Inform.* **2013**, *9*, 2360–2372. [CrossRef]
21. Choi, S.S.; Jung, K.; Kulvatunyou, B.; Morris, K.C. An analysis of technologies and standards for designing smart manufacturing systems. *J. Res. Natl. Inst. Stand. Technol.* **2016**, *121*, 422–433. [CrossRef]
22. Frey, D.; Nimis, J.; Worn, H.; Lockemann, P. Benchmarking and robust multi-agent-based production planning and control. *Eng. Appl. Artif. Intell.* **2003**, *16*, 307–320. [CrossRef]
23. Leusin, M.E.; Kück, M.; Frazzon, E.M.; Maldonado, M.U.; Freitag, M. Potential of a multi-agent system approach for production control in smart factories. In Proceedings of the 16th IFAC Symposium on Information Control Problems in Manufacturing (INCOM), Bergamo, Italy, 11–13 June 2018; pp. 1459–1464.
24. Borshchev, A. *The Big Book of Simulation Modeling: Multimethod Modeling with Anylogic 6*; Any Logic North America: Chicago, IL, USA, 2013.

25. Ennigrou, M.; Ghedira, K. New local diversification techniques for flexible job shop scheduling problem with a multi-agent approach. *Auton. Agents Multi-Agent Syst.* **2008**, *17*, 270–287. [CrossRef]
26. Jain, A.S.; Meeran, S. Deterministic job-shop scheduling: Past, present and future. *Eur. J. Oper. Res.* **1999**, *113*, 390–434. [CrossRef]
27. Ghedira, K.; Ennigrou, M. How to schedule a job shop problem through agent cooperation. In *Artificial Intelligence: Methodology, Systems, Applications, Proceedings*; Cerri, S.A., Dochev, D., Eds.; Springer: Berlin, Germany, 2000; Volume 1904, pp. 132–141.
28. Kück, M.; Ehm, J.; Hildebrandt, T.; Freitag, M.; Frazzon, E.M. Potential of data-driven simulation-based optimization for adaptive scheduling and control of dynamic manufacturing systems. In Proceedings of the 2016 Winter Simulation Conference, Washington, DC, USA, 11–14 December 2016; pp. 2820–2831.
29. Sellers, D.W. A survey of approaches to the job shop scheduling problem. In Proceedings of the Southeastern Symposium on System Theory, Baton Rouge, LA, USA, 31 March–2 April 1996; p. 396.
30. Durasević, M.; Jakobović, D. A survey of dispatching rules for the dynamic unrelated machines environment. *Expert Syst. Appl.* **2018**, *113*, 555–569. [CrossRef]
31. Ouelhadj, D.; Petrovic, S. A survey of dynamic scheduling in manufacturing systems. *J. Sched.* **2009**, *12*, 417. [CrossRef]
32. Sabuncuoglu, I.; Bayız, M. Analysis of reactive scheduling problems in a job shop environment. *Eur. J. Oper. Res.* **2000**, *126*, 567–586. [CrossRef]
33. Rajabinasab, A.; Mansour, S. Dynamic flexible job shop scheduling with alternative process plans: An agent-based approach. *Int. J. Adv. Manuf. Technol.* **2011**, *54*, 1091–1107. [CrossRef]
34. Zhang, G.; Ye, D. A note on on-line scheduling with partial information. *Comput. Math. Appl.* **2002**, *44*, 539–543. [CrossRef]
35. Li, R.-K.; Shyu, Y.-T.; Adiga, S. A heuristic rescheduling algorithm for computer-based production scheduling systems. *Int. J. Prod. Res.* **1993**, *31*, 1815–1826. [CrossRef]
36. Renna, P. Job shop scheduling by pheromone approach in a dynamic environment. *Int. J. Comput. Integr. Manuf.* **2010**, *23*, 412–424. [CrossRef]
37. Mehta, S.V. Predictable scheduling of a single machine subject to breakdowns. *Int. J. Comput. Integr. Manuf.* **1999**, *12*, 15–38. [CrossRef]
38. Vieira, G.E.; Herrmann, J.W.; Lin, E. Analytical models to predict the performance of a single-machine system under periodic and event–driven rescheduling strategies. *Int. J. Prod. Res.* **2000**, *38*, 1899–1915. [CrossRef]
39. Figueira, G.; Almada-Lobo, B. Hybrid simulation-optimization methods: A taxonomy and discussion. *Simul. Model. Pract. Theor.* **2014**, *46*, 118–134. [CrossRef]
40. Owliya, M.; Saadat, M.; Jules, G.G.; Goharian, M.; Anane, R. Agent-based interaction protocols and topologies for manufacturing task allocation. *IEEE Trans. Syst. Man Cybern. Syst.* **2013**, *43*, 38–52. [CrossRef]
41. Shen, W.; Wang, L.; Hao, Q. Agent-based distributed manufacturing process planning and scheduling: A state-of-the-art survey. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2006**, *36*, 563–577. [CrossRef]
42. Barber, K.S.; Liu, T.H.; Goel, A.; Ramaswamy, S. Flexible reasoning using sensible agent-based systems: A case study in job flow scheduling. *Prod. Plan. Control* **1999**, *10*, 606–615. [CrossRef]
43. Scholz-Reiter, B.; Freitag, M. Autonomous processes in assembly systems. *CIRP Ann. Manuf. Technol.* **2007**, *56*, 712–729. [CrossRef]
44. Scholz-Reiter, B.; Kolditz, J.; Hildebrandt, T. Engineering autonomously controlled logistic systems. *Int. J. Prod. Res.* **2009**, *47*, 1449–1468. [CrossRef]
45. Li, D.N.; Wang, Y.; Xiao, G.X.; Tang, J.F. Dynamic parts scheduling in multiple job shop cells considering intercell moves and flexible routes. *Comput. Oper. Res.* **2013**, *40*, 1207–1223. [CrossRef]
46. Cicirello, V.A.; Smith, S.F. Wasp-like agents for distributed factory coordination. *Auton. Agents Multi-Agent Syst.* **2004**, *8*, 237–266. [CrossRef]
47. Caridi, M.; Cavalieri, S. Multi-agent systems in production planning and control: An overview. *Prod. Plan. Control* **2004**, *15*, 106–118. [CrossRef]
48. Wong, T.N.; Leung, C.W.; Mak, K.L.; Fung, R.Y.K. Dynamic shopfloor scheduling in multi-agent manufacturing systems. *Expert Syst. Appl.* **2006**, *31*, 486–494. [CrossRef]
49. Wong, T.N.; Zhang, S.C.; Wang, G.; Zhang, L.P. Integrated process planning and scheduling—multi-agent system with two-stage ant colony optimisation algorithm. *Int. J. Prod. Res.* **2012**, *50*, 6188–6201. [CrossRef]

50. Wong, T.N.; Leungy, C.W.; Mak, K.L.; Fung, R.Y.K. Integrated process planning and scheduling/rescheduling—An agent-based approach. *Int. J. Prod. Res.* **2006**, *44*, 3627–3655. [CrossRef]
51. Scholz-Reiter, B.; De Beer, C.; Freitag, M.; Jagalski, T. Bio-inspired and pheromone-based shop-floor control. *Int. J. Comput. Integr. Manuf.* **2008**, *21*, 201–205. [CrossRef]
52. Duffie, N.A. Synthesis of heterarchical manufacturing systems. *Comput. Ind.* **1990**, *14*, 167–174. [CrossRef]
53. Giannakis, M.; Louis, M. A multi-agent based system with big data processing for enhanced supply chain agility. *J. Enterp. Inf. Manag.* **2016**, *29*, 706–727. [CrossRef]
54. Semwal, T.; Nair, S.B. AgPi: Agents on Raspberry Pi. *Electronics* **2016**, *5*, 72. [CrossRef]
55. Reaidy, P.J.; Gunasekaran, A.; Spalanzani, A. Bottom-up approach based on Internet of Things for order fulfillment in a collaborative warehousing environment. *Int. J. Prod. Econ.* **2015**, *159*, 29–40. [CrossRef]
56. Cupek, R.; Ziebinski, A.; Huczala, L.; Erdogan, H. Agent-based manufacturing execution systems for short-series production scheduling. *Comput. Ind.* **2016**, *82*, 245–258. [CrossRef]
57. Hellingrath, B.; Pereira, C.E.; Espíndola, D.; Frazzon, E.M.; Cordes, A.-K.; Saalmann, P.; Zuccolotto, M. On the integration of intelligent maintenance and spare parts supply chain management. *IFAC-PapersOnLine* **2015**, *48*, 983–988. [CrossRef]
58. Mansour Fallah, S. Multi agent based control architectures. In Proceedings of the 26th DAAAM International Symposium, Vienna, Austria, 23–30 November 2016; pp. 1166–1170. [CrossRef]
59. Hwang, G.; Lee, J.; Park, J.; Chang, T.W. Developing performance measurement system for Internet of Things and smart factory environment. *Int. J. Prod. Res.* **2017**, *55*, 2590–2602. [CrossRef]
60. Cottyn, J.; Van Landeghem, H.; Stockman, K.; Derammelaere, S. A method to align a manufacturing execution system with Lean objectives. *Int. J. Prod. Res.* **2011**, *49*, 4397–4413. [CrossRef]
61. Saenz de Ugarte, B.; Artiba, A.; Pellerin, R. Manufacturing execution system—A literature review. *Prod. Plan. Control* **2009**, *20*, 525–539. [CrossRef]
62. Shaojun, W.; Gang, W.; Min, L.; Guoan, G. Enterprise resource planning implementation decision & optimization models. *J. Syst. Eng. Electron.* **2008**, *19*, 513–521.
63. Vidoni, M.C.; Vecchietti, A.R. An intelligent agent for ERP's data structure analysis based on ANSI/ISA-95 standard. *Comput. Ind.* **2015**, *73*, 39–50. [CrossRef]
64. Lee, H.; Ryu, K.; Son, Y.J.; Cho, Y. Capturing Green Information and Mapping with IVIES Functions for Increasing Manufacturing Sustainability. *Int. J. Precis. Eng. Manuf.* **2014**, *15*, 1709–1716. [CrossRef]
65. Zhang, Y.F.; Qu, T.; Ho, O.; Huang, G.Q. Real-time work-in-progress management for smart object-enabled ubiquitous shop-floor environment. *Int. J. Comput. Integr. Manuf.* **2011**, *24*, 431–445. [CrossRef]
66. Nagorny, K.; Colombo, A.W.; Schmidtmann, U. A service- and multi-agent-oriented manufacturing automation architecture an IEC 62264 level 2 compliant implementation. *Comput. Ind.* **2012**, *63*, 813–823. [CrossRef]
67. Chu, Y.F.; You, F.Q. Integrated Scheduling and Dynamic Optimization by Stackelberg Game: Bilevel Model Formulation and Efficient Solution Algorithm. *Ind. Eng. Chem. Res.* **2014**, *53*, 5564–5581. [CrossRef]
68. Popovics, G.; Pfeiffer, A.; Monostori, L. Generic data structure and validation methodology for simulation of manufacturing systems. *Int. J. Comput. Integr. Manuf.* **2016**, *29*, 1272–1286. [CrossRef]
69. Blum, C.; Roli, A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv. CSUR* **2003**, *35*, 268–308. [CrossRef]
70. Pfeiffer, A.; Kádár, B.; Monostori, L. Stability-oriented evaluation of rescheduling strategies, by using simulation. *Comput. Ind.* **2007**, *58*, 630–643. [CrossRef]
71. Vieira, G.E.; Herrmann, J.W.; Lin, E. Rescheduling manufacturing systems: A framework of strategies, policies, and methods. *J. Sched.* **2003**, *6*, 39–62. [CrossRef]
72. Barnett, L.; Rahimifard, S.; Newman, S. Distributed scheduling to support mass customization in the shoe industry. *Int. J. Comput. Integr. Manuf.* **2004**, *17*, 623–632. [CrossRef]
73. Zuravlyov, V.; Matrosov, A. Multi-agent system built using RFID technology. In Proceedings of the 6th International Conference on Electrical and Control Technologies, Kaunas, Lithuania, 5–6 May 2011; pp. 16–21.
74. Baudin, M.; Rao, A. RFID Applications in Manufacturing. 2000. Available online: https://www.researchgate.net/profile/Michel_Baudin/publication/267218891_RFID_applications_in_manufacturing/links/54b86f380cf2c27adc48b475.pdf (accessed on 4 November 2018).
75. Zhao, F.; Wang, J.; Wang, J.; Jonrinaldi, J. A dynamic rescheduling model with multi-agent system and its solution method. *Strojniski Vestn. J. Mech. Eng.* **2012**, *58*, 81–92. [CrossRef]

76. Zattar, I.C.; Ferreira, J.C.E.; Rodrigues, J.G.G.; De Sousa, C.H.B. A multi-agent system for the integration of process planning and scheduling using operation-based time-extended negotiation protocols. *Int. J. Comput. Integr. Manuf.* **2010**, *23*, 441–452. [CrossRef]
77. Asadzadeh, L.; Zamanifar, K. An agent-based parallel approach for the job shop scheduling problem with genetic algorithms. *Math. Comput. Model.* **2010**, *52*, 1957–1965. [CrossRef]
78. Guo, L.; Wang, S.; Kang, L.; Cao, Y. Agent-based manufacturing service discovery method for cloud manufacturing. *Int. J. Adv. Manuf. Technol.* **2015**, *81*, 2167–2181. [CrossRef]
79. Yang, L.; Sun, X.; Peng, L.; Yao, X.; Chi, T. An agent–based artificial bee colony (ABC) algorithm for hyperspectral image endmember extraction in parallel. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 4657–4664. [CrossRef]