

Seitz, Matthias; Gehlhoff, Felix; Cruz Salazar, Luis Alberto; Fay, Alexander; Vogel-Heuser, Birgit

Article — Published Version

Automation platform independent multi-agent system for robust networks of production resources in industry 4.0

Journal of Intelligent Manufacturing

Provided in Cooperation with:

Springer Nature

Suggested Citation: Seitz, Matthias; Gehlhoff, Felix; Cruz Salazar, Luis Alberto; Fay, Alexander; Vogel-Heuser, Birgit (2021) : Automation platform independent multi-agent system for robust networks of production resources in industry 4.0, Journal of Intelligent Manufacturing, ISSN 1572-8145, Springer US, New York, NY, Vol. 32, Iss. 7, pp. 2023-2041, <https://doi.org/10.1007/s10845-021-01759-2>

This Version is available at:

<https://hdl.handle.net/10419/286990>

Standard-Nutzungsbedingungen:

Die Dokumente auf EconStor dürfen zu eigenen wissenschaftlichen Zwecken und zum Privatgebrauch gespeichert und kopiert werden.

Sie dürfen die Dokumente nicht für öffentliche oder kommerzielle Zwecke vervielfältigen, öffentlich ausstellen, öffentlich zugänglich machen, vertreiben oder anderweitig nutzen.

Sofern die Verfasser die Dokumente unter Open-Content-Lizenzen (insbesondere CC-Lizenzen) zur Verfügung gestellt haben sollten, gelten abweichend von diesen Nutzungsbedingungen die in der dort genannten Lizenz gewährten Nutzungsrechte.

Terms of use:

Documents in EconStor may be saved and copied for your personal and scholarly purposes.

You are not to copy documents for public or commercial purposes, to exhibit the documents publicly, to make them publicly available on the internet, or to distribute or otherwise use the documents in public.

If the documents have been made available under an Open Content Licence (especially Creative Commons Licences), you may exercise further usage rights as specified in the indicated licence.



<https://creativecommons.org/licenses/by/4.0/>



Automation platform independent multi-agent system for robust networks of production resources in industry 4.0

Matthias Seitz¹ · Felix Gehlhoff² · Luis Alberto Cruz Salazar^{1,3} · Alexander Fay² · Birgit Vogel-Heuser¹

Received: 2 April 2019 / Accepted: 11 March 2021 / Published online: 27 April 2021
 © The Author(s) 2021

Abstract

The Cyber-Physical Production System (CPPS) is a concept derived from software (cyber) and hardware (physical) applications and is based on global information exchange between such systems. The CPPS is known as a trend of Industry 4.0 (I4.0) focusing on flexibility regarding new products and adaptability to new requirements. This paper focuses on two I4.0 scenarios described by the Platform Industrie 4.0 that describe challenges for the industry towards its digital future. First, it looks at the Order Controlled Production (OCP) scenario that deals with flexible and self-configuring production networks. It describes the dynamic organization of production resources required to execute a production order. Second, the Adaptable Factory (AF) application scenario is discussed, which focuses on the configuration of production resources and describes the adaptability of an individual facility through (physical) modification. This paper first provides a detailed analysis of the requirements from these scenarios. Furthermore, it analyses the current Multi-Agent System (MAS) architectures and agent-based planning and decision support systems requirements. MAS can be used to create application-independent I4.0 systems with arbitrary hardware automation platforms. To create a scalable communication network that also supports application independence and enables the semantically machine-readable description of the exchanged data, the OPC UA standard was adopted. As a result of the study, the concept shows how different and independent automation platforms can be seamlessly connected via OPC UA. The proposed MAS concept has been evaluated in different use cases, namely OCP and AF.

Keywords Multi-agent systems · Adaptable factory · Industry 4.0 · Order controlled production

Abbreviations

AMS	Agent Management System
AF	Adaptable Factory
CP	Coordination Process
CPS	Cyber Physical System
CPPS	Cyber Physical Production System
DF	Directory Facilitator
HSU	Helmut Schmidt University
I4.0	Industry 4.0
JPP	Joghurt-Production-Protocol
KB	Knowledge Base
MAS	Multi-Agent System
OEE	Overall Equipment Efficiency
OCP	Order Controlled Production
PPR	Product Process Resource
RA	Resource Agent
RAMI4.0	Reference Architecture Model Industrie 4.0
SoS	Systems of Systems
TUM	Technical University of Munich
xPPU	Extended Pick and Place Unit-Demonstrator

All author are members of the Technical Committee FA 5.15 “Agent systems” of the Society Measurement and Automatic Control (GMA) within the Society of German Engineers (VDI), German Electrical Engineers (VDE), Association of Automation Technology in Process Industries (NAMUR), IFAC’s National Member Organizations (NMO); and most are members of the IEEE-IES Technical Committee on Industrial Agents (TC-IA).
 *Mr. Seitz was with the Technical University of Munich when the paper’s submitted.

✉ Matthias Seitz
matthias.seitz@tum.de

✉ Luis Alberto Cruz Salazar
luis.cruz@tum.de

¹ Institute of Automation and Information Systems, Technical University of Munich, Boltzmannstr. 15, 85748 Garching near Munich, Germany

² Institute of Automation Technology, Helmut Schmidt University, Holstenhofweg 85, 22043 Hamburg, Germany

³ Universidad Antonio Nariño, Cartagena, Colombia

Introduction and motivation for multi-agent systems for industry 4.0

The Cyber-Physical Production System (CPPS) is a concept derived from software (cyber) and hardware (physical) applications in manufacturing control systems (Ribeiro & Hochwallner, 2018; Ueda et al., 2009). CPPSs are known as one trend of the 4th industrial revolution called *Industry 4.0* (I4.0). It includes mechatronic systems that should provide flexibility regarding innovative products, adaptability to new industrial requirements or to cover the failure of components (Karnouskos et al., 2019). The initiative I4.0 focuses on the entire lifecycle of production systems (Platform Industrie 4.0, 2016).

A particularly strong development trend for the control of software entities is the agent-based approach. This approach is not yet widely established in industrial environments that require high robustness, such as manufacturing and process control. Several types of distributed agents have been implemented in hard and soft real-time devices, so that multi-agent systems (MAS) are becoming increasingly suitable for CPPS (Leitão & Vrba, 2011).

The agent-based CPPS architecture is a specialization of the concept of *Cyber-Physical System* or CPS, which is often defined as "integrations of computation with physical processes" (Lee, 2008). Within a CPPS network, agents control physical units or even systems, e.g. a measurement device, an actuator, a machine or even a plant as a whole. Using agents, these are connected to other related entities within a production environment to establish a system of systems on different levels and domain application (Leitão et al., 2016). To cope with the problem of linking several remote production or manufacturing facilities, this paper deepens the concept of the underlying logical MAS architectural design that is necessary for creating such agent-based solutions. The MAS approach addresses requirements—which resulted from previous research in (Cruz et al., 2018)—for the implementation of a CPPS and the key I4.0 demands, which have been derived from the scenarios described below.

As the main contribution of this paper, we deliver a novel method to couple heterogeneous aPS to perform two use cases like AF and OCP defined by the platform Industry 4.0. For this purpose, we developed a MAS-based method for establishing networks of production systems by combining a variety of specialized MASs. Thus, we are providing a concept of how I4.0 scenarios can be implemented using an agent-based automation platform to connect different heterogeneous production systems and their components, e.g., small sensors. We demonstrated and evaluated the benefit of the approach by integrating different lab size plants from two different research groups at two universities.

The presented approach uses model-based methods for the information and knowledge representation as well as the agents' interactions: For a uniform representation of the plants' characteristics, the properties of the respective plants are modelled according to the Product Process Resource (PPR) description ("[Agent-based planning and decision systems](#)" section). This information model serves as the agents' knowledge base that can be exchanged using OPC UA. To model the different interactions of the agents, sequence diagrams are used ("[MAS communication infrastructure using OPC UA](#)" section).

The most important requirements are listed in Table 1. We have also elaborated the valuable insights we have gained in the process of this work at the end of this article.

Table 1 outlines requirements (2.X) that stem from the Reference Architecture Model Industrie 4.0 (RAMI4.0) (DIN SPEC, 2016). The general requirements (1.X) in Table 1 summarize basic prerequisites for an agent-based CPPS that were introduced in (Leitão & Vrba, 2011) and further specified regarding RAMI in preliminary work in (Cruz et al., 2018). Besides these general requirements for CPPS within I4.0 environments, scenario-specific requirements for the application scenarios OCP and AF (Platform Industrie 4.0, 2016) need to be met. These requirements will be derived first.

In addition, the use case of the "Smart Product Development for Smart Production" was implemented using agents to connect various systems for data analysis and thus constantly optimize the overall equipment efficiency (OEE) of the entire system by means of condition monitoring. For example, in (Yazdi et al., 2018), empirical research shows the relationship between OEE and manufacturing sustainability in I4.0. However, this is not the focus of this paper, for further information please refer to the different domain of industrial agents in (Leitão et al., 2016).

I4.0 Scenarios and requirements

Order controlled production (OCP)

Production companies are confronted with a two-fold challenge. On the one hand, the product variety and complexity, i.e. the number of different products sold and components needed for these products, has increased significantly while on the other hand product lifecycle duration is in steady decline (Roland Berger Strategy Consultants, 2012). This poses a threat to traditional, capital intense yet inflexible production systems, as companies might not be able to sell the necessary number of units to amortize their investment.

Therefore, the need arises to increase a company's manufacturing flexibility, i.e. the capability to offer different products without having to provide all the necessary machinery

Table 1 Minimal systems requirements for I4.0 scenarios [adapted from (Platform Industrie 4.0, 2017a)]

Requirement	Description
Application and level independence (R1.1)	Basic MAS architecture, protocols and messages should be independent of a specific application, and all levels of automation for ISA 95 should be integrated
Platform-independent implementation (R1.2)	Components must be simple to be integrated with independent implementations (open architecture)
Robustness against errors (R1.3)	MAS must be robust against unforeseen faults in the CPPS network and react dynamically to unexpected situations
Interlinked engineering data/Standardized semantics (R2.1)	Workpieces shall be able to specify the production procedure applied to them
Interoperability/Standardized communication infrastructure (R2.2)	Composed of interoperable components that are interoperable (e.g. device and communication models of OPC UA)
Plug and produce availability (R2.3)	Each newly connected field device shall receive network connectivity without manual intervention
Automated coordination mechanism (R2.4)	Possibility of moving software component for process control between decentralized control units, while adhering to constraints such as production output and availability
Reusability (R2.5)	Support component-based engineering, where libraries of reusable components are used (e.g. IEC 62,714 “AutomationML” or NAMUR MTP)

at a single location. This can be achieved by leveraging flexible production networks of multiple manufacturing facilities. These networks enable companies to combine their capabilities and automatically configure required value chains on demand. This approach requires a standardized (self-) description of production capabilities as well as standardized process steps to configure required production processes. Companies can use this network to extend their capabilities as well as to offer capacities to increase the utilization of their machinery (Platform Industrie 4.0, 2016).

The OCP imposes high demands on intelligent connectivity of all entities involved in the manufacturing process in order to utilize the existing manufacturing facilities in the most effective and cost-efficient way. This is particularly evident as the fulfilment of a production order can involve multiple facilities that can be located in separate locations or even belonging to different companies, such as suppliers. Complex orders require a significant coordination effort, where plants need to communicate remotely. Thereby, the communication can include time-critical notifications or requests as well as long-term diagnostics. This requires a suitable and flexible architecture for software agents as well as resource-friendly and accepted communication protocols and messages.

Adaptable factory (AF)

While the application scenario OCP focuses on the flexible operation of existing production facilities by intelligent networks, the application scenario “Adaptable Factory” (AF) describes the versatility of a single factory through a physical transformation. This can be a fast and possibly largely

automated conversion of a production plant with regard to both changed production capacities and changed production possibilities.

In order to provide a highly scalable system, the AF scenario embraces Plug & Produce capabilities by relying on modular production resources with a component-based approach. For a uniform modular approach, the control technology must also meet these requirements. Those modular, order-oriented and adaptable manufacturing configurations become more and more attractive to enhance flexibility and maximize utilization. However, cooperation between these modular components results in an increased need for communication. In addition, distributing manufacturing processes also implies the logical distribution of control intelligence.

A central concept for achieving these objectives is a modular and therefore highly flexible production infrastructure within a factory. Intelligent and interoperable modules, which largely adapt to a changed configuration, and standardized interfaces between these modules enable a simple and fast changeover. Thus offering the potential to adapt to changing market and customer requirements (Platform Industrie 4.0, 2016).

Requirements from OCP and AF

In addition to the basic requirements in Table 1, the following requirements for MASs are derived from the I4.0 scenarios OCP and AF:

(R3.1) Standardized communication infrastructure (protocol and components): A standardized communication protocol, which defines syntax, semantics and possible

interaction sequences, is required that enables communication across company borders. This protocol should be accepted by the industry, secure, fast and flexible as well as simple regarding the structuring of interactions. When selecting a communication infrastructure, standardized multiplatform support must be assured. Due to the highly reactive environment, asynchronous message exchange is necessary to handle tasks while waiting for a response and also to give priority to a task with higher importance. The authenticity of the messages must also be guaranteed by encryption.

Additionally, the use of a central communication platform as part of the communication infrastructure within the production network is advisable. This approach facilitates one-to-many communications as every participant can monitor or subscribe topics of interest and thereby receive all desired information (Hoffmann et al., 2016). Fault tolerance can be integrated by creating backup instances of central components. As an order-controlled production must provide an entry point for orders, which usually is a website or a similar application, at least this component is usually centralized. Further developing this component to provide the necessary functionalities for communication does not significantly decrease the overall fault-tolerance of the system.

(R3.2) Standardized semantics: Building on the joint communication infrastructure, the semantics of the communication must be defined and known to all participants within the production network. This applies to the description of products as well as production capabilities (Platform Industrie 4.0, 2017a).

(R3.3) Coordination mechanism: To realize an automated configuration of value chains also across companies that react dynamically to incoming orders, i.e. changing demand, the mechanism that coordinates which facility is producing which component must be agreed upon mutually.

(R3.4) Simple configuration of the production network: The registration process to the production network must be as simple as possible. Prospective participants should not have to implement proprietary protocols or learn about complex, unstandardized interaction sequences.

(R3.5) Dynamic/automatic/flexible reconfigurability: A production is rebuilt or modified because a new product variant is to be manufactured. The control/software relevant changes are to be detected and propagated automatically to all connected facilities involved.

(R3.6) Plug and Produce for field devices: The adaptable factory requires swift solutions to react to changes in the plant layout of a CPPS. A new device is automatically detected and integrated into the existing network. In an adaptable factory, the production components should contain a machine-readable, semantically unique self-description of their properties and capabilities. This enables (semi-) automatic integration of the production components to achieve a

production goal and can reduce the manual effort of today's machine operators and service personnel.

(R3.7) Modularization of software and hardware: To quickly change the plant setup with a low effort of changing the corresponding control software, both the hardware and the software components must follow a modular design approach.

Related work: manufacturing system approaches regarding key requirements for I40 OCP and AF

To persist in today's global markets, information must flow between all layers of a company and even between collaborating companies. This requires new approaches to communication and production. A common way to access the factory floor is using gateways. In (Sauter & Lobashov, 2011) an overview of suitable high-level protocols to access data from automation systems using special gateways is presented. In (Sauter & Lobashov, 2011) an overview of suitable high-level protocols to access automation data via gateways is presented. One possible application of a reconfigurable sensor interface and gateway is presented in (Tao et al., 2014). This work also presents a new design method, but only focuses on the lowest layer of the IoT architecture.

In (Girbea et al., 2014) the design process for a SOA capable of real-time operation is focused. This is achieved using a priori algorithms and is thus not suitable for dynamically changing environments. Other approaches use SOA for diagnosis (Calvo et al., 2012) and the concept presents a possible architecture and diagnosis algorithms without implementation. Another implementation language for SOA is IEC 61499, e.g., proposed by (Barata et al., 2008) in combination with a message broker. A runtime based on a formal mapping between SOA and IEC 61499 is proposed by (Delamer & Lastra, 2006). In contrast to other programming languages, IEC 61499 only runs on special hardware or with dedicated runtimes. Additionally, reconfiguration and behavioral intelligence are accumulated inside a central node running on powerful hardware. Other research on new approaches by using IEC 61499 as an emerging standard for industrial automation is presented in (Vyatkin, 2011).

MAS has also been used and implemented in production and logistics environments. For example, Leitao et al. (2019) develop a CPS that consists of conveyor modules controlled by JADE agents, which reside on Raspberry Pis. These agents determine their position within the conveyor system themselves. Kovalenko et al. (2019) develop a MAS that incorporates planning and exploration of production sequences, i.e., the search for appropriate resources and the execution of planned operations. In this approach, product and resource agents communicate in a flow-like

architecture to determine each production operation's most suitable resources. Karnouskos et al. (2020) present how the industrial agent is key to realizing industrial CPS. This paper presents the author's view of recent trends and significance, with the fundamental facilitating features of agent-based CPPS. They also assess how the agent definitions, development, and solutions have improved over the years and where existing efforts may focus. Colombo et al. by the EU PERFoRM project, investigated MAS deployment opportunities (Colombo et al., 2019), particularly to apply in the Reference Architectural Model Industrie 4.0 (RAMI4.0). The industrial agent has played a central role in PERFoRM scenarios, enabling flexibility, adaptability, and distributed control functionalities.

Other work targets the use of MASs in the field of intelligent energy systems or smart grids (Vrba et al., 2014). Apart from the manufacturing industry, MASs are also used in the process industry, e.g., to control critical processes (Metzger & Polakow, 2011). In Barata et al. (2008) another concept and implementation for reconfiguration focused on challenges of the shop floor, especially addition and removal of manufacturing components during runtime is presented. Communication is based on JADE, using JAVA and FIPA ACL Messages. Thus, automation platform independence is limited. The IDEAS project (Onori et al., 2012) uses specifically designed and produced boards to bring agent technology to lower automation levels. These boards are designed to support JAVA. Thus, to deploy this agent system, specifically designed hardware must be used. Small microcontrollers or other existing automation hardware are not supported.

State of the art of agent-based cpps architecture for the industry 4.0

The agent-based approach of this contribution is structured based on CPPS requirements and the requirements for I4.0.

MAS logical architecture

An agent-based CPPS requires three basic organizational entities, e.g., for discovery purposes: an agent management system (AMS), a message transport system (MTS) and a directory facilitator (DF) (FIPA, 2002).

The DF is a service that stores agents' abilities, i.e., the production capabilities of a system. All agents register themselves with these organizational entities.

Derived from the already existing standard of the Foundation for Physical Agents (FIPA) for MASs, the architecture shown in Fig. 1 was developed. All agents of the CPPS are connected via a common network, e.g., the global internet. To support many different use cases (cp. R1.1), an agent of

the CPPS can have one of two characteristics: each agent that appears in the network of the manufacturing facilities represents either the physical system itself or an organizational entity. Such an agent provides diagnosis services (diagnosis agent) or introduces production requests into the system (product agent) across the CPPS network.

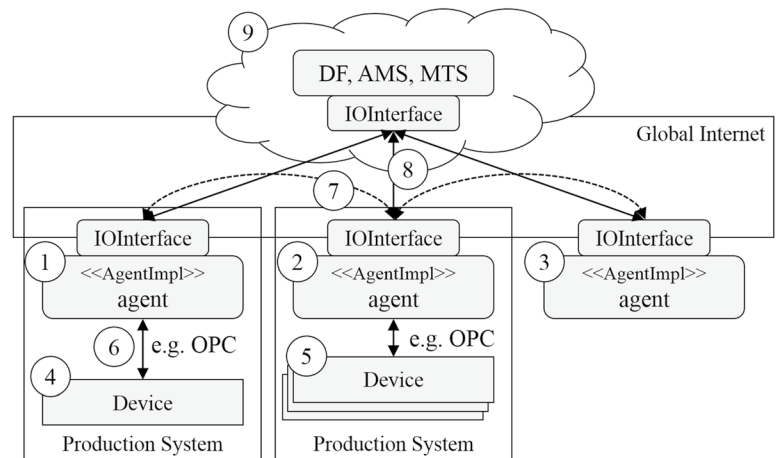
In the first case, the (physical) agent provides access for the production system to the virtual world and the CPPS; therefore, there are various authors working on different MAS-based CPPS architectures. For example, the pattern of the *Resource Agent (RA)* architecture presented by Wannagat (Cruz et al., 2019), offers an agent-based interface for equipment in the field control level (see Fig. 2).

The RA is composed of four main modules as part of common MAS patterns functionalities (Cruz et al., 2019): The Coordination Process, the Communication interface, and the Knowledge Base. The *Control Module* connects the plant hardware with the I/Os (sensors and actuators signals) in order to send the data of the control variables and to acquire the measurements from the sensors. A *Diagnosis Module* allows the detection of failures and relates these by the RA *Status*, which communicates the existing situation by *Agent Interaction* interfaces. Incoming sensor data is processed to detect sensor failures. In this case, the *Diagnosis Module* connects to the *Knowledge Base Module* that specifies the system model to the corresponding equipment. Finally, the *Planning Module* covers local goals and negotiates time schedules for message exchange with other types of agents.

As results, a CPPS consists of a varying number of these agents, (e.g. Wannagat's RA networks). In order to support the easy migration of existing arbitrary systems to network-enabled systems (cp. R1.1, R1.2), an agent may be a dedicated part of the automation software as well as separate software on separate hardware. It may represent a single device as well as several devices. Application, level and automation platform independence (cp. R1.1, R1.2), as well as migration, are further increased by not setting a default for the communication with attached physical systems. Possible implementations are e.g., field bus protocols, OPC or proprietary protocols. To support as many different hardware devices as possible, the code that manages communication with the physical systems is released in separate communication modules. This also allows the future implementation of further use cases (cp. R1.1).

Agent systems are deployed in a variety of environments, interacting with human workers. In order for the operator to have trust in the autonomous system, agents must understandably communicate their decisions, e.g. why a specific adaption in the production process was made by the agent or ask the worker beforehand which exact adaptation should be made. In addition, agents can provide passive support by informing the operator of current and potential future risks

Fig. 1 Logical architecture of a automation platform-independent MAS (Cruz et al., 2018)



as soon as they are detected. Usually, each agent has to refer to many inputs and states to make its decision. Therefore, the amount of information multiplies by the number of agents and their interactions. To visualize the current state of a MAS to the operator in an expressive way, three-dimensional graphics proved to be suitable (Bockel et al., 2007).

In Fischer et al. (2019) visualization for the layout of the agent controlled myYoghurt demonstrator is introduced. During operation, the agents send live data gathered from the demonstrator to the visualization. This live data can include the conveying direction of the currently controlled actuators as well as information calculated by the MAS on how the identified transport routes are then displayed for the plant operator or maintenance employee. This allows the operator to distinguish the normal behavior of the MAS from possible malfunctions that require operator intervention.

To support a broad variety of hardware automation platforms, a MAS was designed as a lightweight implementation in the programming language ANSI C in Cruz et al. (2018). However, to connect different existing MASs, this implementation lacks flexibility. To create a more scalable communication network that also supports application independence and enables the semantically machine-readable description of the exchanged data, the OPC UA standard was adopted.

MAS communication infrastructure using OPC UA

In comparison to special self-developed protocol solutions, free software projects are well-maintained and arising problems are quickly and actively resolved by participating developers. Many self-developed solutions also originated from a time in which those open-source implementations did not yet exist or were lacking the necessary maturity. With regards to cross-platform support, OPC UA also supports encryption and authentication capabilities out of the box, implemented and tested by experts from the community. Due

to the open and language-independent standard, open-source libraries are available for every common programming language (R3.1).

Open Source libraries are used by a wide range of developers. Therefore, there is usually a high-level interface available which reduces the amount of code required to perform frequent functions to a minimum (R3.1). This leads to less error-proneness when used by the developer and also shortens the time a developer needs to set up a new application.

OPC UA has also been chosen because it provides communication protocols, i.e. semantics and syntax for information exchanges, as well as the overall communication infrastructure, i.e. transmission of messages and a unified information model that is accessible by all participants in the network or production facility (Weyrich et al., 2014) (R3.2). In addition, OPC UA does not specify which language (e.g. JAVA) an application has to use. This allows different implementations with different (programming) languages to be integrated on the same OPC UA server (R1.2). A major benefit of OPC UA is that the OPC UA standard is known beyond specific implementations and separates information modelling from the communication infrastructure (Hoffmann et al., 2016) (2.2, 3.2). This enables OPC UA clients to search for information without knowing the specific implementation of the information model (Mahnke et al., 2009). As OPC UA is already widely adopted in the industrial context, modern PLCs/ Industrial PCs often already provide an embedded OPC UA server (R3.6, R3.7). Some manufacturer also allows for editing the UA namespaces of their controllers to match specific semantics. Besides, OPC UA makes it very easy to control and restrict access to specific data or determine certain rights for users (R3.1). As the communication between participants in the production network for the OCP and AF scenarios evolves around order and production management activities, the communication speed does not have to fulfil hard real-time restrictions, which might be hampered by the OPC UA protocol (Weyrich et al., 2014).

Agent-based planning and decision systems

As shown in “[MAS Logical Architecture](#)” section, a possible collaboration of patterns agents (e.g. between AMSs and RAs) is provided by internal components (e.g., Diagnosis module) that are independent of other modules (e.g., for order planning, Decision Support Systems, etc.). However, agents for the realization of I4.0 can be parallel to the existing agents and can also be integrated into them (VDI/VDE 2019). Patterns agents that are part of other agents are called “sub-agents”, as given in (Cruz et al., 2019). Therefore, because of the independent and cooperative nature of MAS, there are different categories of sub-agents able to plan and schedule system tasks individually.

For example, planning systems can be integrated from a higher level of decomposed sub-agents with separated functionalities (job scheduling, prognosis, diagnosis agents, etc.) into a lower level (shop floor agents) (Salvador Palau et al., 2019). As given in (Weiming et al., 2006), those sub-agents can solve common-planning problems using cooperation and negotiation. In the specific case of Decision Support Systems (Hess et al., 2008), the benefits of agents can be the collection and analyses of data outside of the organization, e.g., to project future variances in materials, labor, prices, business, overhead, etc., (Sharda et al., 2019). In this case, the agents are equipped with specific components to generate decision-making alternatives allowing external operators to focus on relevant variances. Agents can provide an automated and cost-effective means making projections and creating alternative courses of action (e.g., usually called “bots”), as given in (Sharda et al., 2019). Table 2 shows multiples examples where the use of agents is enabled to support planning and scheduling systems based on (Salvador Palau et al., 2019), and Decision Support Systems based on (Hess et al., 2008). Both works referenced in Table 2 evidenced that agents can be equipped with planning and decision-making elements based on specific task capabilities.

Case of study OCP and AF with the I4.0 demonstrators

This section presents the application of the OCP and AF scenarios in the I4.0 community demonstrator myYoghurt in combination with I4.0 demonstrators at HSU and TUM.

Another contribution from the authors in (Vogel-Heuser et al., 2020), compares how agent-based systems can meet emerging challenges in those I4.0 scenarios. Software agents are an option for the realization of such OCP and AF. Due to their features, MASs are mostly well appropriate for representing their I4.0 components and enabling I4.0 interactions. Agents in different I4.0 demonstrators can recognize and integrate not only the necessary I4.0 scenario languages, but also the essential methods for self-organization and self-optimization in value creation (Vogel-Heuser et al., 2020).

The myYoghurt demonstrator is a joined academic initiative from automation research that implements I4.0 scenarios across multiple universities (Mayer et al., 2013). Within the myYoghurt scenario, an agent registers with the coordinator for a specific capability or production process, e.g. yoghurt or lid production (Mayer et al., 2013). Subsequently, the suppliers of a production step are responsible for the fulfilment of an order. Due to this separation of concerns within the limited scenario of myYoghurt, a detailed analysis if a certain product can be manufactured by a production partner is not necessary, as the registration at the DF is sufficient. The coordinator asks for and selects proposals from participants within the production network. In contrast to (Platform Industrie 4.0, 2017a), where the selection process is carried out by the regular customer, in the myYoghurt scenario the coordinator is not only responsible for obtaining suitable offers from production partners, but also for selecting the most appropriate offers for inclusion in the offer available to the customer. To reduce complexity during order creation, the customer, which is in this case, the end-user, is not supposed to decide between different

Table 2 Examples of agent’s task capabilities utilized in planning and decision systems

System	Application	Example of the agent’s task capability
Planning and scheduling (Salvador Palau et al., 2019)	Job scheduling	Shop floor agents enable dynamic job scheduling and other agents negotiate and evaluate its optimal cost
	Prognosis, diagnosis, and prognostics	An agent can analyze data, e.g. for prognostics of shipboard power systems
	Maintenance planning	Maintainer agents collaborate to improve maintenance schedule
	Task sequencing	Machine agents able to optimize tasks, e.g. sequencing operation
Decision support systems (Hess et al., 2008)	Data-monitoring	An agent can report if any price change crosses given threshold values
	Data-gathering	Agent identify a supply of manufactured parts at a rational price
	Modeling	Agents report significant modeling trends, e.g. dollar changes consequences
	Preference-learning	Agent able to learn a user’s preferences based on its historical data, e.g. bots

manufacturers or production processes, but the CPPS itself by negotiation in between all participating plants. However, that requires every participant's agreement to the algorithm that determines how offers are rated within the network. In this implementation, the lowest price is the essential criterion for the selection. Possible cost functions can include factors like material selection, set-up costs or location of production facilities (logistics) as well as trust. Additionally, if the participants of the coordination algorithm stem from different companies and use real prices (in contrast to transfer prices) the figure has to include the company's profit margin (Gehlhoff et al., 2019). The coordinator also implements scheduling functions that for example take sequence restrictions into account (Weyrich et al., 2014). These have a major impact on possible lead times of a product because some processes can occur in parallel (yoghurt and lid production) while others can be carried out only sequentially (yoghurt production and refinement). Communication is handled by a client-socket-based protocol, the *Joghurt-Production-Protocol 2* (JPP2) (Mayer et al., 2013), which is a lightweight but inflexible XML-based communication protocol.

14.0 demonstrators

For the case study of the scenarios, the demonstrators serve as real plants connected by an MAS to form an agent-based network of production systems. The institutes HSU and TUM are equipped with a variety of demonstrators for research and education and are close to industrial applications. The demonstrators address the automation of different domains like logistics or process industry. To present both scenarios using an MAS, each institute integrates existing demonstrators with similar capabilities. Overall, the production of yoghurt bottles should be considered for the two scenarios. For this purpose, each institute integrates a demonstrator that produces the actual yoghurt as well as a demonstrator that produces the necessary lids for the bottles. In addition, each institute provides an intralogistics demonstrator that links the intralogistics processes with each other. Most of these demonstrators are already equipped with agent systems from previous research projects. However, as the Self-x logistic demonstrator is not yet operated by a running agent system, the integration of a new manufacturing plant into the agent automation platform is being investigated. The used range of demonstrators include the following:

- *myYoghurt* (TUM) A modular material flow and process demonstrator for implementing and evaluating different control concepts in the field of intralogistics. The process technology section of the demonstrator consists of a (process) workstation and two filling stations for dispensing yoghurt. The demonstrator serves as a test-

bed for research in the field of intralogistics and process technology intending to improve the interchangeability of components, both software and hardware.

- *Extended Pick and Place Unit-Demonstrator* (*xPPU*, TUM) A production demonstrator, which consists of storage, manufacturing and logistics. Because of its versatility, the PPU supports various production scenarios of the manufacturing domain, such as hardware customization.
- *Self-x* (TUM) A modular Material Flow demonstrator for implementing and evaluating different control concepts in the field of intralogistics.
- *MPS500* (HSU) A flexible manufacturing system that produces customized air cylinders and thermometers.
- *PL* (*Production and Logistics demonstrator*) An MAS that controls production and intralogistics processes.
- *MODVA* (HSU) A modular process plant that can be controlled by remotely calling different recipes.

These demonstrators differ in many aspects like accessibility through different protocols, modularity or automation platform requirements. Table 3 shows the result of an analysis of these demonstrators and relevant criteria.

The demonstrators were implemented using different platforms (e.g. JADE or C#) and accessibility protocols. This drives the need for an automation platform-independent integration approach that is implemented with OPC UA.

In addition, to integrate multiple production facilities within the same production network, a unified communication platform and front-end (human interaction) are needed, which currently does not exist. Security measures are also an important factor for robustness that is currently not sufficiently addressed. OPC UA provides these functionalities and enables a unified approach to integrate the demonstrators within the same network and also provides comprehensive security measures and common standards (Vargas et al., 2017).

Summarizing, the case studies are highly related to demonstrators and standard automation platforms from market-leading automation vendors available in HSU and TUM. Still, the MAS of this work becomes an automation platform-independent by the interoperability of OPC UA, as an independent service-oriented architecture.

Order controlled production

The OCP scenario was implemented within the myYoghurt research project. It enables the automated control of the production facilities, initiated by a customer via a web front-end, at the HSU and the TUM. The MPS500, the MODVA and the PL serve in this scenario as exemplary production facilities that can produce lids (MPS500) and yoghurt (MODVA) respectively or organize intralogistics activities (PL).

Table 3 Overview of I4.0 research demonstrators at the research institutes regarding implemented MASs

	TUM			HSU		
	My-Joghurt	xPPU	Self-x	MPS 500	PL	MOD-VA
MAS/SOA Platform	C#, C89, JADE	–	–	JADE	JADE	SOA
Accessibility (protocols)	JPP, FIPA-ACL	HTTP, OPC UA	–	JPP, FIPA-ACL	FIPA-ACL	OPC UA
Modular HW setup	–	–	+	+	+	+
Human Inter-action	Yoghurt web shop	Web interface	–	Yoghurt web shop	–	HMI
Security	–	+	–	–	–	–

The implementation is building on a legacy agent-system, in which the agents “speak” the already mentioned JPP2, including the DF-AMS, coordinator and system agents. The DF-AMS combines, in this case, the yellow-page (who provides which service) and white-page (addresses of participants) services. The coordinator implements the already mentioned order management functionalities. Keeping these agents and their ability to speak JPP2 makes it possible to connect agents that do not possess OPC UA capabilities as well. Within this agent-system, the DF-AMS agent is equipped with OPC UA capabilities to communicate with the OPC UA server and thereby serves as a gateway agent. In Fig. 4 the gateway interactions OPC UA—JPP2 are marked green and JPP2—OPC UA blue.

The higher-level system agents start the actual production by triggering a Manufacturing Execution System (MES) that controls the MPS500 and a process control system that controls the MODVA. It has been decided to keep the exiting low-level control of the production equipment to preserve real-time responsiveness (Leitão et al., 2015) and to enable the agents to start production processes by triggering the existing control components. The P&L demonstrator can provide intralogistics functions. However, these require additional interactions that are not elaborated here. In (Spindler et al., 2016) this subject is covered in greater detail.

Order management using OPC UA

One of the central components of the OPC UA information model for the OCP scenario is the “CreateGetOfferMessageObject”-method. The website’s OPC UA client uses this method to create nodes for every order that the customer has posted via the web front-end. The method input is the desired yoghurt configuration. An order node then contains every information the agent system needs to make an offer. The interactions required for the order management are depicted in Fig. 4. The agents within the legacy agent-system are requested via JPP2 to make offers for those parts of the order that match their specific capability. OPC UA enables the integration of further demonstrators without using JPP2 or adjusting the implementation of the coordinator or other legacy agents. Agents

that do not speak JPP2 subscribe the getOfferMessage-Type directly and call the createOffer-method to post an offer. This offer is routed to the coordinator by using post and subscribe functions and the DF-AMS gateway agent, which translates the message to JPP2.

The agents in this demonstrator use details of the order (e.g. cost of materials and order size) to calculate their costs. Afterwards, they calculate prices based on the margin that they aim for and additional optimization that includes the conditions on the market place and possible synergies (Gehlhoff et al., 2019). The coordinator chooses suitable offers to create the total offer and determines the total price and delivery date for the order. This price, as well as the expected delivery date, are returned to the customer by setting specific variables within a separate folder for the specific offer. The OPC UA server informs the website’s OPC UA client about changes in that folder (realized with an OPC UA subscription); the client reads the new values and manages the information of the customer. Upon receiving this feedback, which can also be a refusal, e.g. if a required capability is currently not available in the production network, the customer can decide whether the order is to be produced with adjustments, postponed or cancelled. This dynamic pricing model reflects the flexible production network configuration, i.e. the nature and number of participants, which can result in very different production conditions. For example, a large number of yoghurt producers represented on the network will lead to lower prices as these suppliers will participate in price competitions. Another solution would be to use pre-determined prices or price ranges displayed to the customer, for example, suggested in (Platform Industrie 4.0, 2017b). However, this is not part of the original myYoghurt getOffer configuration, which is used in this approach, and it is also less responsive regarding the resulting price that is displayed to the customer.

If the customer accepts the offer, the OPC UA client that resides at the website will call the “AcceptOfferMethod”. This method instantiates a setOrder-message that is read by the DF-AMS agent. The DF-AMS again takes care of implementing the JPP2 protocol and sends a “setOrder” message to the coordinator. The coordinator then splits the message and sends setOrder messages to every agent involved in the

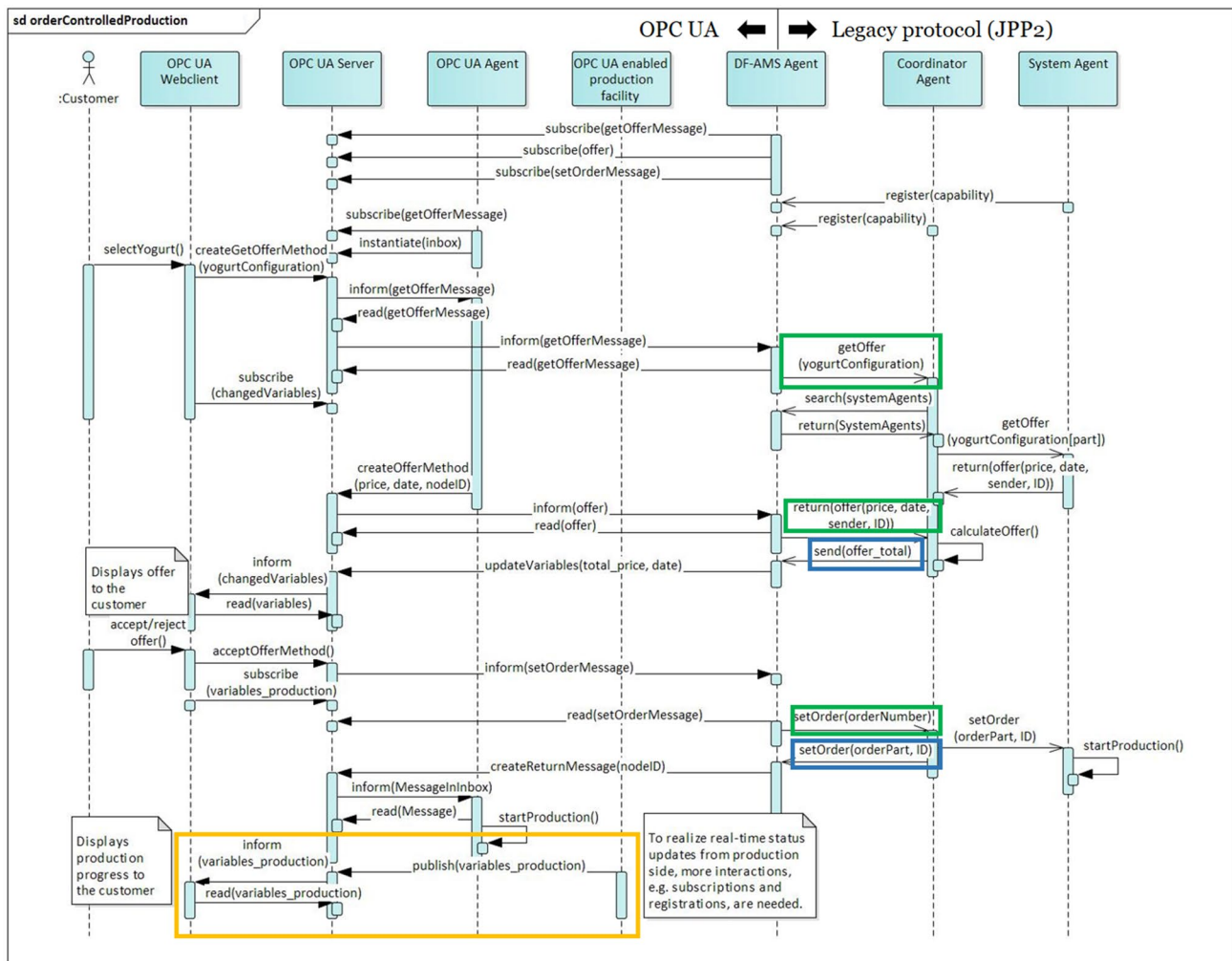
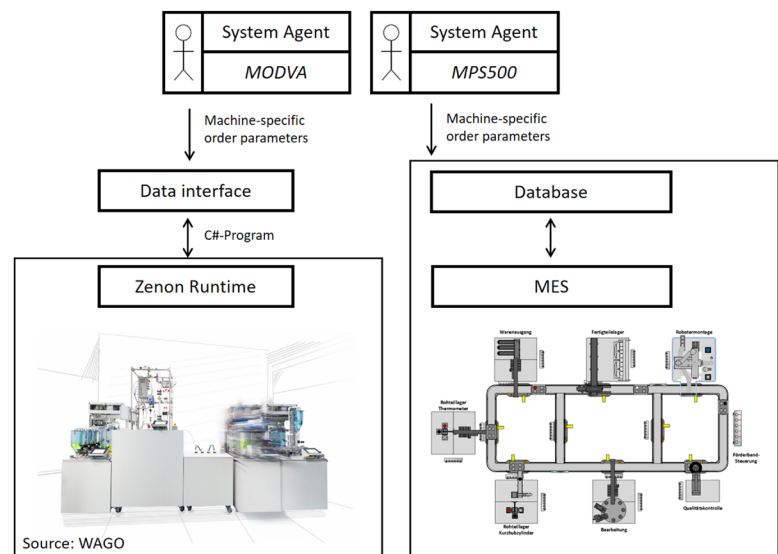
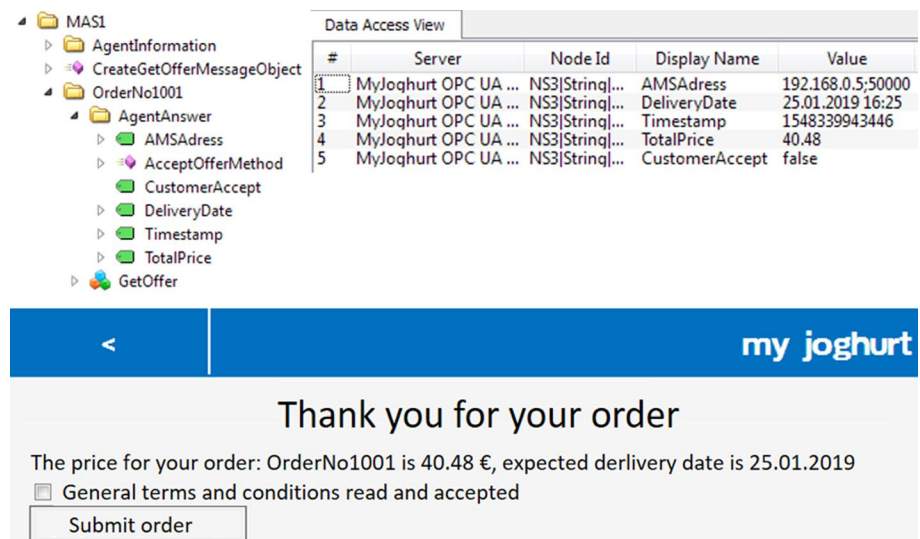
Fig. 3 Control flow for the production facilities at the HSU**Fig. 4** UML Sequence diagram of interactions within the OCP scenario

Fig. 5 Part of customer interaction and OPC UA namespace



The figure shows a screenshot of a customer interaction interface for 'my joghurt'. The interface includes a navigation menu on the left with items like 'MAS1', 'AgentInformation', 'CreateGetOfferMessageObject', 'OrderNo1001', 'AgentAnswer', 'AMSAAddress', 'AcceptOfferMethod', 'CustomerAccept', 'DeliveryDate', 'Timestamp', 'TotalPrice', and 'GetOffer'. The main content area displays a 'Thank you for your order' message with the price for OrderNo1001 as 40.48 € and an expected delivery date of 25.01.2019. Below the message is a checkbox for 'General terms and conditions read and accepted' and a 'Submit order' button. A 'Data Access View' table is also shown, listing data points from the OPC UA namespace.

#	Server	Node Id	Display Name	Value
1	MyJoghurt OPC UA ...	NS3 String ...	AMSAAddress	192.168.0.5;50000
2	MyJoghurt OPC UA ...	NS3 String ...	DeliveryDate	25.01.2019 16:25
3	MyJoghurt OPC UA ...	NS3 String ...	Timestamp	1548339943446
4	MyJoghurt OPC UA ...	NS3 String ...	TotalPrice	40.48
5	MyJoghurt OPC UA ...	NS3 String ...	CustomerAccept	false

Information Modeling

One major aspect of I4.0 is communication across all levels. To meet the challenges of ambiguity in information exchange, a common understanding of the messages exchanged in the agent network is required. By also detailing the interactions between the agents, a rich communication workflow is achieved.

This section covers how messages between agents are exchanged using the OPC UA protocol and the message vocabulary used to make messages interpretable for agents. The formalized process description VDI 3683 is used as a means of describing the production capabilities. The allocation of a technical resource to a process operator takes place based on various characteristics (e.g. geometry of the input and output product, process-related quality characteristics or technical data of the resource). By modelling the product, process, and resource as I4.0 components, they can realize a dynamic assignment at runtime through interaction.

Due to the concept of the Digital Factory and the associated necessity to electronically describe all planning data involved in the production process, a three-way division into resource, process and product data has proven itself in practice (Drath, 2010). All three views are linked with each other. With the product, process and resource model (PPR model) and the data model of the VDI/VDE 3682 guideline for formalized process description, it is possible to describe the capabilities of resources in the sense of machines and plants as well as the requirements on the production process by products. An important feature of the three different elements is that they are closely related: A product is processed by a resource within a process (Fig. 7). This three-view concept is a prime example of networked engineering data (Schleipen & Drath, 2009). For agents, it serves as a recipe for uniformly describing their abilities.

Agents responsible for executing a manufacturing process need detailed knowledge of the configuration of the production unit they control. This requires the existence of an appropriate knowledge base when commissioning the agent. Agents with primarily coordinating and monitoring activities need knowledge about other agents and their condition.

The purpose of the description model is to provide all necessary information for the classification of the production systems in a semantically uniform way. Ultimately, it should enable the agents to derive the corresponding production plan or to derive the basic suitability for the fulfilment of the inquiry, starting from an inquiry for the manufacture of a product.

Adaptions by agents

In adaptable manufacturing systems, several products can be manufactured using the same resources. If possible, the factory setting is reused for the different products and only the action sequences are updated accordingly. In some cases, the factory setting may need to be changed. Changes include adding or removing resources or changing the location of a resource within the factory. It is also possible to use different modules of a resource for different products. In order to generate action sequences for different products automatically, two aspects must be considered during generation: the product description and factory setting. By decoupling the software control from the machines, the network of agents forms an adaptable architecture that can be flexibly scaled regarding the production system's needs. Coordinating high amounts of individual and distributed modules with specific interfaces is a challenging task that agents can tackle by providing fast communication channels to a multitude of automation platforms and establishing networks of interconnected facilities.

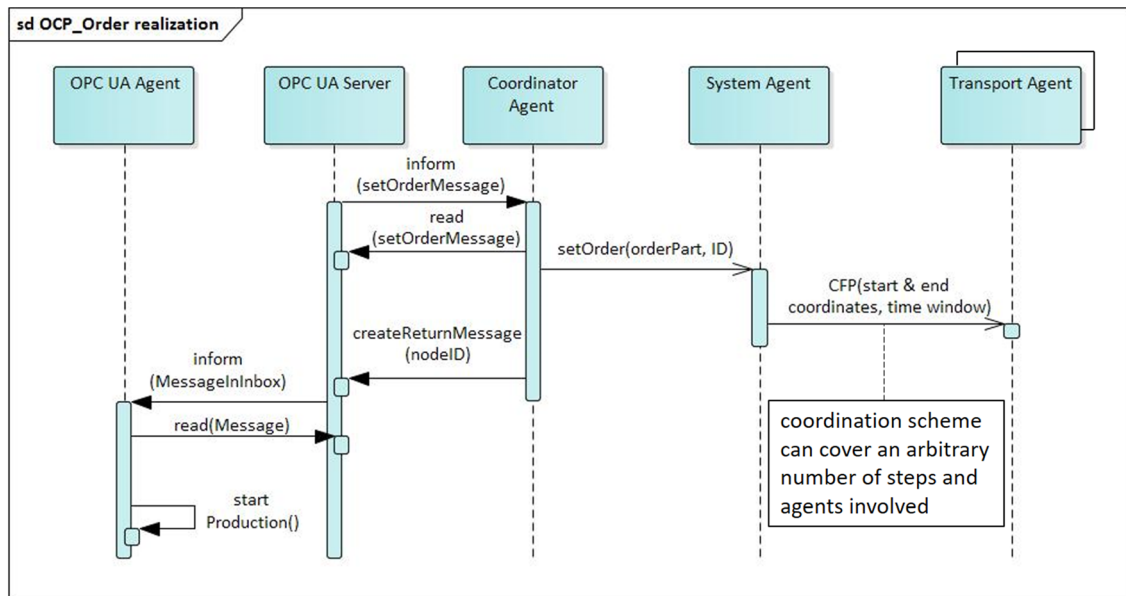


Fig. 6 UML Sequence diagram of interactions within the OCP scenario—coordinator as gateway agent

Modularization of the software in the form of agents with the capability to distribute functionalities makes it possible to scale the network of production facilities. To meet the flexibility requirements of hardware modules, the software architecture in an adaptable factory must also be modular. By linking agents together, messages or updates can also be automatically sent to all using the shared communication network. In order to be able to interact with other systems that may have a different implementation, a communication channel via the OPC UA protocol is selected to unify the exchange of information. In addition to the communication channel, the type of messages used is also displayed uniformly. The capabilities of the individual production sites are described by a uniform vocabulary so that all different MASs can understand the messages and express themselves by using the same vocabulary. The Configuration is updated once the agent detects a change in the setup.

An integration agent is responsible for managing the network of production facilities. By linking different facilities, it creates a distributed control system consisting of one distributed facility. These systems, in particular logistics systems, consist of various numbers of modules and can be flexibly combined to produce a new type of product or offer new functionalities. Within a network of different agents, individual configurations can be distributed very quickly. Agents can also aggregate different functions due to their modular character. In this way, functions can also be dynamically shifted between agent systems and thus react quickly to changes in the plant layout. By using agents, the network can distribute software modules for controlling the manufacturing processes within the MAS.

Coupled by agents that can all exchange information via a uniform communication channel, the demonstrators in Table 3 form a network that enables adaptation across different granularity tiers. Each demonstrator has a counterpart at the other institute, such as for yoghurt or lid manufacturing. Using OPC UA, the agents can communicate even if their local system primarily uses a different protocol (e.g. JPP2). Within this network, agents can react flexibly to any changes in their plant and thus relocate functionalities to other facilities by negotiation with other agents or request services from them. Using an OPC UA interface, the different MASs can also exchange information with each other, even if their implementation differs. Agents can also react locally to changes in their plant. If a filling station fails at the myYoghurt system, the agent system adjusts the routing of the workpieces so that the failed unit does not result in the system shutting down.

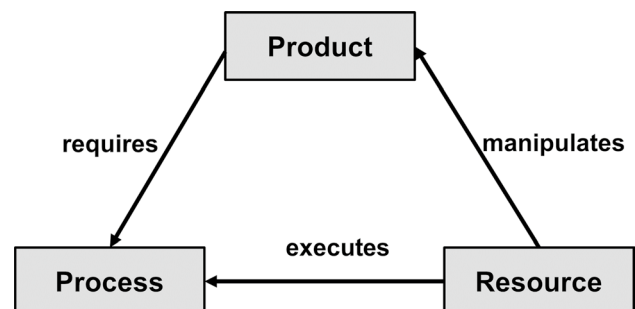
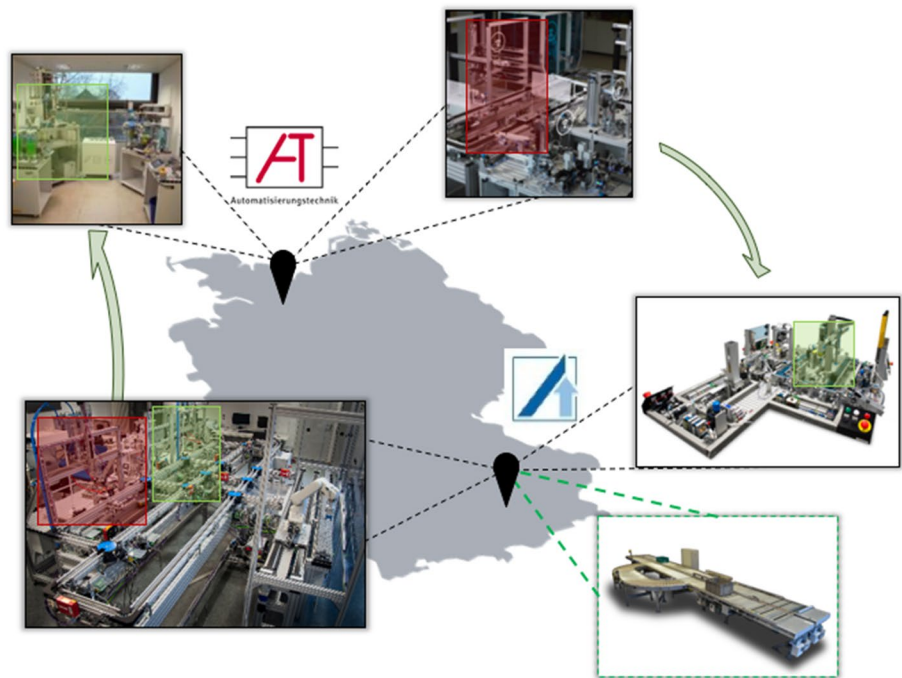


Fig. 7 Elements of the PPR concept (Schleipen & Drath, 2009)

Fig. 8 Adaptations of the manufacturing process by the agent system



As described in "MAS communication infrastructure using OPC UA" section each institute has three facilities involved in the production of yoghurt bottles (cp. Table 3). Except for the Self-x demonstrator, each system is operated by an individual agent system. In order to integrate Self-x with its logistics services into the network, an independent MAS that can manage the control processes of the plant was initially set up. For the integration OPC UA was used, through which a dedicated agent in the Self-x MAS contacts the network and registers himself with his capabilities. After registration, the system with all its functions is at the disposal of the network. Therefore, this process of establishing contact and capability registration can be considered as an agent-based plug & produce process.

In addition to adopting new plants, the agent network can also be used to shift assignments of facility configurations when a change occurs (e.g. as a result of a failure of a component). The agent network can be understood as a collective unit of which each demonstrator is a part of. Figure 8 shows an example of the physical distribution of the plants and how the production configuration can be shifted between the institutes as a result of a change of a component's state. The Fig. 9 illustrates the adaption of configurations of the agents. For example, the transfer of the ability to produce yoghurt from the myYoghurt plant, which is necessary due to the failure of one of the two filling stations, can be either directly adopted locally to its remaining filling station or even relocated to the MODVA plant.

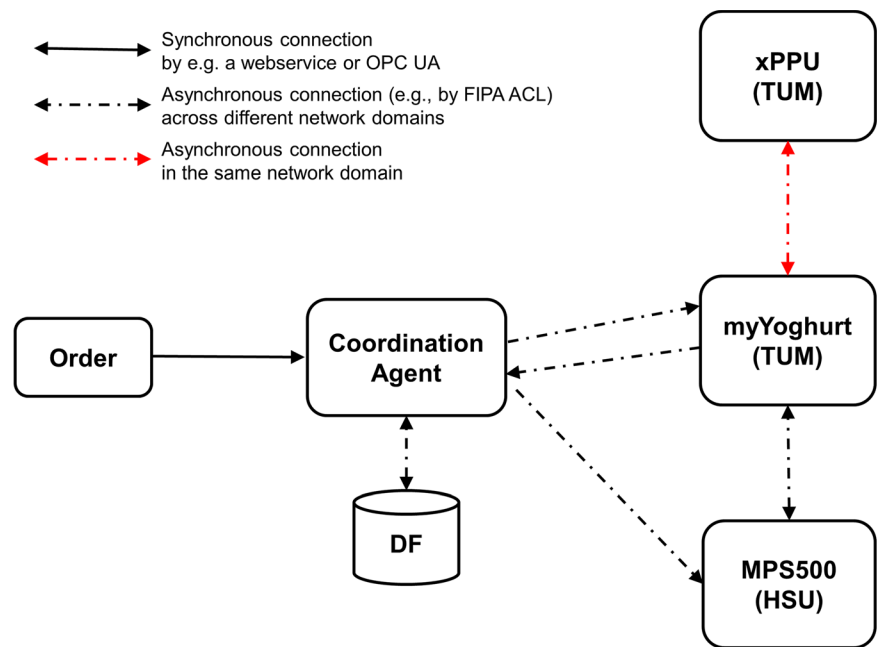
Likewise, the ability to manufacture lids can be shifted from the MPS500 agent to the xPPU agent in the event of

an adaptation. In Fig. 9, the adaption process is illustrated. The myYoghurt, xPPU and MPS500 are each controlled via the agent architecture presented. Both systems are registered with their respective capabilities (PPR) in a central directory, the DF. A Coordination Agent accepts new orders and coordinates them by delegating the production of individual sub-products to the appropriate production facilities by comparing the order with the capabilities registered in the DF. When an order for a yoghurt arrives, a new production order is created and forwarded to the agent system. Based on the product characteristics and the capabilities stored in the DF, the myYoghurt system is selected as a suitable production system and receives the production order.

Within the myYoghurt system, the order is assigned to a new product agent who coordinates the production, as introduced by (Kovalenko et al., 2019). Whenever new lids are required to seal the filled yoghurt bottles, a production order for lids is sent to the xPPU. If the xPPU reports that no lids are currently available, it reports an error back to the myYoghurt system. This sends a response to the Coordination Agent. The Coordination Agent in return compares the stored production properties in the DF with the requirements for lid production and selects the MPS500 as a suitable alternative. The order is now forwarded to the MPS500 with the contact information of the myYoghurt system. The agents of the MPS500 then start producing lids and report to the agents in the MyYoghurt system.

To always be able to provide information on the current status of their hardware components, the agents need to constantly follow the modifications to the hardware

Fig. 9 Adaption procedure for lid production of the demonstrators



they represent and reconfigure themselves according to the changes. The management agent takes over the device management of the network by keeping track of the respective systems and their configuration. This makes it possible to automatically reconfigure the configuration after the conversion and to initiate the redistribution of production services.

The software architecture of the agent system responsible for handling the logistics operations within the myYoghurt plant is depicted in Fig. 10. The Software on each of the Programmable Logic Controllers (PLC) contains two

parts: a classic control program for controlling the hardware (controlling the actuators, reading the sensors) and an agent system (transport agent and framework). The active framework functions as the central intelligence (coordinator) of the entire plant's agent system, which simultaneously serves as the uniform interface to higher-level systems (e.g. Warehouse Management System, ERP, Databases, etc.).

If the myYoghurt's conveyor system is modified or converted because a new product is required. The system and software-related changes must be identified and

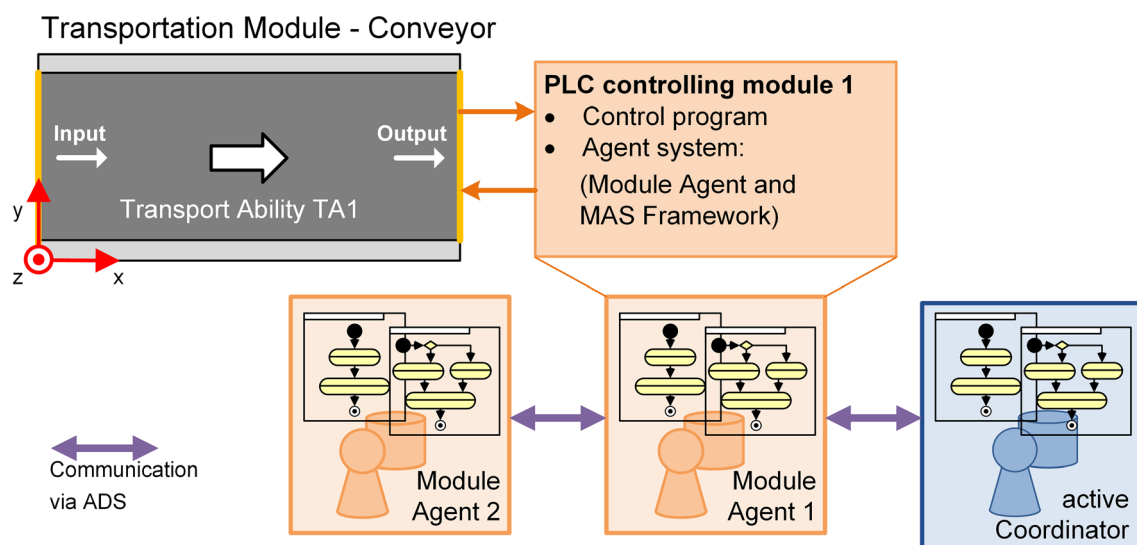


Fig. 10 Design of the agent-based control architecture for the transportation conveyor of the myYoghurt plant

automatically transferred to all participating agents. When the current layout is changed, two cases are distinguished: Adding and removing a conveyor belt. When adding an additional conveyor belt to the system, it must be registered in the MAS and its functionality must be published. The registration process is completed in a few minutes and is performed by the active coordinator. In order to determine the current plant layout, the coordinator needs to require the description of the newly added conveyor, which it requests from the designated transport agent.

Another advantage of MASs in this context is that additional services can be integrated with ease into an existing MAS. Furthermore, the abilities of agents to make decisions autonomously and to learn from past events lead to faster reactions and optimized control. The joint research demonstrator myYoghurt embodies such a distributed production facility that is powered by agent-based control applications. Each production facility resides in their very own location and communicates with other remote systems by the means of a cloud-based agent management system that coordinates all registered resources within the network.

Due to its application-independent nature, the developed MAS is open for extensions. Additional arbitrary systems can be integrated with ease. In addition, agents can introduce new services or products and advertise them within the network by the means of their underlying information model.

Integrated I4.0 scenarios

The integration of different MASs within a unified architecture enables the System of Systems (SoS) concept (Colombo et al., 2013), by joining its previously disjunctive components. MAS technologies take advantage of agent-inherent characteristics to develop large and complex SoS that exhibit modularity, adaptation, reconfiguration, and responsiveness to condition changes (Karnouskos et al., 2019; Leitão et al., 2016).

Such a use case is, for example, the exploration of possible paths through the system that a product can utilize. The result of this kind of search is commonly known as the process plan. To determine possible operation or process sequences, a coordinator agent (or any other entity responsible for this task) contacts each RA with a request that contains the description of the required in and outputs to produce a specific product or provide a particular aspect of its production. This description, for example formulated in the formalized process description (Gehlhoff et al., 2020), can be matched against the RA's capabilities. In case the RA can provide the necessary capability, it answers accordingly.

Otherwise, it can relate the message to further RA's that can, in turn, check if they can contribute to the requested task. Note that it is advisable to formulate the requests as 'feature-neutral' as possible, i.e. by stating that a whole is required instead of specifying that a drilling process must be provided. This enlarges the search space and can improve the solution. A product agent (or the coordinator agent) that is equipped with the resulting flexible process plan can utilize this plan during the scheduling process (Gehlhoff & Fay, 2020). It can evaluate different possibilities to be manufactured, i.e., calculate alternative paths through a graph. Besides, it can use the latter during the execution process if the preferred route is blocked.

There is also the possibility to provide further functionalities during the checking of producibility. If the capability model of the RA's is extended by, for example, an algorithm that provides the means to check adaptation options (Hoang et al., 2019). This enables the MAS to not only check the producibility with regards to the current and alternative configurations of the system, i.e. the selection of a specific tool in a multi-tool machine. It also provides a means to integrate possible adaptations, like for example the enlargement of a gripper to handle larger workpieces, into the solution space.

Conclusions and outlook

In this paper, the feasibility to apply MASs for the I4.0 scenarios OCP and AF was demonstrated. The integration of a facility that had not previously been controlled by agents required the development of a new MAS in addition to the integration and involved extensive implementation effort. Instead of using the domain-specific protocol solutions, remote communication channels based on the OPC UA standard are implemented. The integration of existing agent systems into a common automation platform was made possible by uniform OPC UA interfaces. Once defined, these interfaces could be quickly implemented for each MAS of the individual demonstrators.

By applying OPC UA in combination with an information model, the exchanged messages of the agents were structured in a semantically interpretable way. An enhanced MAS implementation on the myYoghurt automation platform served as an application example to demonstrate practical applications that fulfil the OCP and AF I4.0 scenarios. To provide OCP, it handles the execution of flexible production processes of customer-specific products with minimum lot sizes across the network. For the AF scenario, the MAS allows for online reconfigurations, such as adding or removing parts of a material flow system and adapts the production control accordingly to meet the requirements of the necessary resource modification.

When developing a standalone agent system, it has to be kept in mind that the system should be easily accessible from the outside or connected to an existing application. Furthermore, for retrofitting old MASs the use of standardized protocols instead of proprietary developments facilitates integration in general. When communicating between distributed systems, a stable connection is particularly important to ensure that messages are exchanged successfully. In the presented work, the implementation of redundant communication channels as well as a request-response schematic for message exchange increased the overall stability and drastically reduced the susceptibility to errors.

Consequently, during the implementation of the agent-based solutions, the following insights emerged, which will also be taken into account for future work:

A concise description realized as an interchangeable model of the functionality of systems enables easier coupling of applications from different production domains.

A model-based description designed as sequence diagrams of the communication and the nature of the messages the agents exchange shortens the development time for integration into other external interfaces.

Finally, future work also should cover the provision of the automation platform for further facilities. In addition to simplifying the adaptation of existing agent-controlled systems to the automation platform, model-based approaches will also be investigated to ease the implementation of MASs from the ground up. To further ensure the interpretability of exchanged messages between the agents with an increasing number of systems, methods are currently being investigated to model the messages using a uniform standard, such as eC@ss or the IEC Common Data Dictionary (IEC, 2017).

Acknowledgements The authors especially thank the DFG for partially funding this work in the frame of the SPP 1593 and SFB 768 T5.

Funding Open Access funding enabled and organized by Projekt DEAL.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Barata, J., Cândido, G., & Feijao, F. (2008). A multiagent based control system applied to an educational shop floor. *Robotics and Computer-Integrated Manufacturing*, 24(5), 597–605.
- Bockel, B., Wenzel, S., Wannagat, A., Pantförder, D., & Vogel-Heuser, B. (2007). Development and Application of Visualisation Techniques for Logistics Systems.
- Calvo, I., Portillo, E., García, O., Armentia, A., Marcos, M., Estévez, E., et al. (2012). *Ubiquitous computing and ambient intelligence* (pp. 282–289). Springer.
- Colombo, A. W., Gepp, M., Barata, J., Leitão, P., Barbosa, J., & Wermann, J. (Eds.). (2019). *Digitalized and Harmonized Industrial Production Systems. Digitalized and Harmonized Industrial Production Systems* (1st ed.). Boca Raton, FL: CRC Press/Taylor and Francis. doi:<https://doi.org/10.1201/9780429263316>
- Colombo, A. W., Karnouskos, S., & Bangemann, T. (2013). A system of systems view on collaborative industrial automation. In 2013 IEEE International Conference on Industrial Technology (ICIT) (pp. 1968–1975). IEEE. doi:<https://doi.org/10.1109/ICIT.2013.6505980>
- Cruz, S., Mayer, F., Schütz, D., & Vogel-Heuser, B. (2018). Platform Independent Multi-Agent System for Robust Networks of Production Systems. *IFAC-Papers OnLine*, 51(11), 1261–1268. <https://doi.org/10.1016/j.ifacol.2018.08.359>
- Cruz, S., Ryashentseva, D., Lüder, A., & Vogel-Heuser, B. (2019). Cyber-physical production systems architecture based on multi-agent's design pattern—comparison of selected approaches mapping four agent patterns. *International Journal of Advanced Manufacturing Technology*, 105(9), 4005–4034. <https://doi.org/10.1007/s00170-019-03800-4>
- Delamer, I. M., & Lastra, J. L. M. (2006). Service-Oriented Architecture for Distributed Publish/Subscribe Middleware in Electronics Production. *IEEE Transactions in Industrial Information*, 2(4), 281–294.
- DIN SPEC. (2016). Reference Architecture Model Industrie 4.0 (RAMI4.0).
- Drath, R. (2010). *Datenaustausch in der Anlagenplanung mit AutomationML*. (R. Drath, Ed.). Springer. doi:<https://doi.org/10.1007/978-3-642-04674-2>
- FIPA. (2002). *FIPA Abstract Architecture Specification*.
- Fischer, J., Lieberoth-Leden, C., Fottner, J., & Vogel-Heuser, B. (2019). Erhöhung der Rekonfigurierbarkeit automatisierter Materialflusssysteme durch einen agentenbasierten Steuerungsansatz. In *Deutscher Materialfluss-Kongress (MFK)* (pp. 91–97).
- Gehlhoff, F., & Fay, A. (2020). On agent-based decentralized and integrated scheduling for small-scale manufacturing. *Automatisierungstechnik*, 68(1), 15–31. <https://doi.org/10.1515/auto-2019-0105>
- Gehlhoff, F., Fay, A., & Köster, B. (2020). Dezentrales Scheduling für dynamische Systeme: Ein agentenbasierter Ansatz zur integrierten Feinplanung. *atp magazin*, pp. 54–61. <https://atpinfo.de/produkt/dezentrales-scheduling-fuer-dynamische-systeme/>
- Gehlhoff, F., Nabizada, H., & Fay, A. (2019). Optimization of multi-agent auctioning processes in flexible production networks. In *Proceedings, 2019 IEEE 17th International Conference of Industrial Informatics (INDIN)*. Helsinki, Finland.
- Girbea, A., Suciu, C., Nechifor, S., & Sisak, F. (2014). Design and implementation of a service-oriented architecture for the optimization of industrial applications. *IEEE Transactions in Industrial Information*, 10(1), 185–196.
- Hess, T. J., Rees, L. P., & Rakes, T. R. (2008). Using Autonomous Software Agents in Decision Support Systems. In *Handbook on Decision Support Systems 1* (pp. 529–555). Springer. doi:https://doi.org/10.1007/978-3-540-48713-5_25

- Hoang, X.-L., Caesar, B., & Fay, A. (2019). Adaptation of Manufacturing Machines by the Use of Multiple-Domain-Matrices and Variability Models. *IFAC-PapersOnLine*, 52(13), 1361–1366. <https://doi.org/10.1016/j.ifacol.2019.11.388>
- Hoffmann, M., Thomas, P., Schütz, D., Vogel-Heuser, B., Meisen, T., & Jeschke, S. (2016). Semantic integration of multi-agent systems using an OPC UA information modeling approach. *IEEE INDIN*. <https://doi.org/10.1109/INDIN.2016.7819258>
- IEC. (2017). *ISO 13584-42 / IEC 61360: Standard data element types with associated classification scheme (4th ed.)*.
- Karnouskos, S., Leitao, P., Ribeiro, L., & Colombo, A. W. (2020). Industrial agents as a key enabler for realizing industrial cyber-physical systems: Multiagent systems entering industry 4.0. *IEEE Industrial Electronics Magazine*, 14(3), 18–32. <https://doi.org/10.1109/MIE.2019.2962225>
- Karnouskos, S., Ribeiro, L., Leitao, P., Luder, A., & Vogel-Heuser, B. (2019). Key Directions for Industrial Agent Based Cyber-Physical Production Systems. In *IEEE International Conference on Industrial Cyber Physical Systems (ICPS)* (pp. 17–22). doi:<https://doi.org/10.1109/icphys.2019.8780360>
- Kovalenko, I., Ryashentseva, D., Vogel-Heuser, B., Tilbury, D., & Barton, K. (2019). Dynamic resource task negotiation to enable product agent exploration in multi-agent manufacturing systems. *IEEE Robotics and Automation Letters*, 4(3), 2854–2861. <https://doi.org/10.1109/LRA.2019.2921947>
- Lee, E. A. (2008). Cyber Physical Systems: Design Challenges. In 2008 11th IEEE international symposium on object and component-oriented real-time distributed computing (ISORC), pp. 363–369. doi:<https://doi.org/10.1109/ISORC.2008.25>
- Leitão, P., & Barbosa, J. (2019). Modular and Self-organized Conveyor System Using Multi-agent Systems. In D. la P. F. Demazeau Y., Matson E., Corchado J. (Ed.), *Advances in Practical Applications of Survivable Agents and Multi-Agent Systems: The PAAMS Collection. Lecture Notes in Computer Science*, vol 11523. (pp. 259–263). Springer. doi:https://doi.org/10.1007/978-3-030-24209-1_26
- Leitão, P., Karnouskos, S., Ribeiro, L., Lee, J., Strasser, T., & Colombo, A. W. (2016). Smart Agents in Industrial Cyber-Physical Systems. *Proceedings of the IEEE*, 104(5), 1086–1101. <https://doi.org/10.1109/JPROC.2016.2521931>
- Leitão, P., Rodrigues, N., Turrin, C., & Pagani, A. (2015). Multiagent system integrating process and quality control in a factory producing laundry washing machines. *IEEE Transactions on Industrial Informatics*, 11(4), 879–886. <https://doi.org/10.1109/tii.2015.2431232>
- Leitão, P., & Vrba, P. (2011). Recent Developments and Future Trends of Industrial Agents. In V. Malvrik, P. Vrba, & P. Leitão (Eds.), *Holonic and Multi-Agent Systems for Manufacturing* (pp. 15–28). Springer.
- Mahnke, W., Stefan-Helmut, L., & Matthias, D. (2009). *OPC unified architecture*. Springer.
- Mayer, F., Pantförder, D., Diedrich, C., & Vogel-Heuser, B. (2013). *Deutschlandweiter I4.0-Demonstrator*.
- Metzger, M., & Polakow, G. (2011). A survey on applications of agent technology in industrial process control. *IEEE Transmission in Industrial Information*, 7(4), 570–581.
- Onori, M., Lohse, N., Barata, J., & Hanisch, C. (2012). The IDEAS project: plug & produce at shop-floor level. *Assembly Automation*, 32(2), 124–134.
- Platform Industrie 4.0. (2016). *Aspects of the Research Roadmap in Application Scenarios I4.0. Platform Industrie 4.0*. Berlin, Germany. https://www.plattform-i40.de/I40/Redaktion/EN/Downloads/Publikation/aspects-of-the-research-roadmap.pdf?__blob=publicationFile&v=10
- Platform Industrie 4.0. (2017a). *Industrie 4.0 Plug-and-Produce for Adaptable Factories: Example Use Case Definition, Models, and Implementation*. ZVEI - Zentralverband Elektrotechnik- und Elektronikindustrie e. V. https://www.plattform-i40.de/I40/Redaktion/EN/Downloads/Publikation/Industrie-40-Plug-and-Produce.pdf?__blob=publicationFile&v=7%0Ahttp://www.plattform-i40.de/I40/Redaktion/DE/Downloads/Publikation/Industrie-40-Plug-and-Produce.pdf?__blob=publicationFile
- Platform Industrie 4.0. (2017b). *Application scenario in practice: order-controlled production of a customised bicycle handlebar*.
- Ribeiro, L., & Hochwallner, M. (2018). On the design complexity of cyberphysical production systems. *Complexity*, 2018, 1–13. <https://doi.org/10.1155/2018/4632195>
- Roland Berger Strategy Consultants. (2012). Roland berger: Mastering product Complexity.
- Salvador Palau, A., Dhada, M. H., & Parlikad, A. K. (2019). Multi-agent system architectures for collaborative prognostics. *Journal of Intelligent Manufacturing*, 30(8), 2999–3013. <https://doi.org/10.1007/s10845-019-01478-9>
- Sauter, T., & Lobashov, M. (2011). How to access factory floor information using internet technologies and gateways. *IEEE Trans. Ind. Inf.*, 7(4), 699–712.
- Schleipen, M., & Drath, R. (2009). Three-view-concept for modeling process or manufacturing plants with AutomationML. *IEEE Conference on Emerging Technologies and Factory Automation*. <https://doi.org/10.1109/ETFA.2009.5347260>
- Sharda, R., Delen, D., & Turban, E. (2019). *Analytics, Data Science, & Artificial Intelligence: Systems for Decision Support* (11 edition.). Pearson.
- Spindler, M., Aicher, T., Vogel-Heuser, B., & Günthner, W. A. (2016). Efficient control software design for automated material handling systems based on a two-layer architecture. In *Proceedings of the 5th IEEE international conference on advanced logistics and transport (ICALT)*.
- Sundermeier, J., Gehlhoff, F., & Fay, A. (2019). Development of a simulation model to analyze the performance of decentral rescheduling algorithms in production systems. Tagungsband ASIM 2018, 24. Symposium Simulationstechnik, ARGESIM Report Nr. 56.
- Tao, F., Cheng, Y., Xu, L. D., Zhang, L., & Li, B. H. (2014). CCIoT-CMfg: Cloud computing and internet of things-based cloud manufacturing service system. *IEEE Transmission in Industrial Information*, 10(2), 1422–1435.
- Ueda, K., Takenaka, T., Váncza, J., & Monostori, L. (2009). Value creation and decision-making in sustainable society. *CIRP Annals*, 58(2), 681–700. <https://doi.org/10.1016/j.cirp.2009.09.010>
- Vargas M., C., Langfinger, M., & Vogel-Heuser, B. (2017). A tiered security analysis of Industrial Control System Devices. In *15th IEEE International Conference on Industrial Informatics (INDIN)* (pp. 399–404). Emden, Germany. doi:<https://doi.org/10.1109/INDIN.2017.8104805>
- VDI/VDE. (2019). *Agents for the realisation of Industrie 4.0*. Düsseldorf, Germany. <https://www.vdi.de/ueber-uns/presse/publikationen/details/agents-for-the-realisation-of-industrie-40>
- Vogel-Heuser, B., Seitz, M., Cruz, S., Gehlhoff, F., Dogan, A., & Fay, A. (2020). Multi-agent systems to enable Industry 4.0. *Automatisierungstechnik*, 68(6), 445–458. <https://doi.org/10.1515/auto-2020-0004>
- Vrba, P., Marik, V., Siano, P., Zhabelova, G., Vyatkin, V., & Strasser, T. (2014). A review of agent and service-oriented concepts applied to intelligent energy systems. *IEEE Transmission in Industrial Information*, 10(3), 1890–1903.
- Vyatkin, V. (2011). IEC 61499 as enabler of distributed and intelligent automation: State-of-the-art review. *IEEE Transmission in Industrial Information*, 7(4), 768–781.
- Weiming, S., Lihui, W., & Qi, H. (2006). Agent-based distributed manufacturing process planning and scheduling: a state-of-the-art survey. *IEEE Transactions on Systems, Man and Cybernetics*,

- Part C (Applications and Reviews)*, 36(4), 563–577. <https://doi.org/10.1109/TSMCC.2006.874022>.
- Weyrich, M., Diedrich, C., Fay, A., Wollschlaeger, M., Kowalewski, S., Göhner, P., & Vogel-Heuser, B. (2014). Industrie 4.0 am Beispiel einer Verbundanlage. *Automatisierungstechnische Praxis*, 56(07–08), 52. <https://doi.org/10.17560/atp.v56i07-08.301>
- Yazdi, P. G., Azizi, A., & Hashemipour, M. (2018). An empirical investigation of the relationship between overall equipment efficiency (OEE) and manufacturing sustainability in industry 4.0 with time study approach. *Sustainability (Switzerland)*. <https://doi.org/10.3390/su10093031>
- Zhang, Y., Qian, C., Lv, J., & Liu, Y. (2017). Agent and Cyber-Physical System Based. *IEEE Transactions on Industrial Informatics*, 13(2), 737–747. <https://doi.org/10.1109/TII.2016.2618892>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.