Towards the generic integration of agent-based AASs and Physical Assets: a four-layered architecture approach

Alejandro López
Systems Engineering and Automatic Control Department
University of the Basque Country (UPV/EHU)
Bilbao, Spain
alejandro.lopez@ehu.eus

Oskar Casquero
Systems Engineering and Automatic Control Department
University of the Basque Country (UPV/EHU)
Bilbao, Spain
oskar.casquero@ehu.eus

Elisabet Estévez
Electronics and Automation Engineering Department
University of Jaén
Jaén, Spain
eestevez@ujaen.es

Paulo Leitão, *Senior Member, IEEE*
Research Centre in Digitalitzaion and Intelligent Robotics
Instituto Politécnico de Bragança
Bragança, Portugal
pleitao@ipb.pt

Marga Marcos, *Senior Member, IEEE*
Systems Engineering and Automatic Control Department
University of the Basque Country (UPV/EHU)
Bilbao, Spain
marga.marcos@ehu.eus

***Abstract*—The I4.0 Component is a key concept of the Platform Industrie 4.0 initiative. The I4.0 Component is made up of an asset performing services and an Asset Administration Shell (AAS) representing the asset in the system. I4.0 Components must meet the requirements of interoperability, identification, representation, information management, integration, and asset management. Industrial agents are a suitable alternative to implement I4.0 Components since they meet these requirements and provide additional features such as distributed decision-making capabilities. Nevertheless, the development of generic approaches for the integration of AASs and assets is not straightforward due to the diversity of physical assets in the factories. This paper proposes a four-layered architecture for the implementation of I4.0 Components based on industrial agents, two in the agent-based AAS and the other two in the physical asset. The division into layers enhances the separation of concerns, encapsulating the different integration aspects at different abstraction levels. This approach is complementary to the standard IEEE 2660.1-2020 on recommended practices for industrial agents, which focuses on the deployment. As a proof of concept, the proposed architecture will be applied for the integration of a software agent and a robot from an assembly cell.**

Keywords— AAS, Industrial Agent, 2660.1-2020

I. INTRODUCTION

The current context of economic uncertainty pushes the industry to adopt the latest trends and paradigms to endure in a hostile and narrowed market [1]. Thus, companies rush to embrace the precepts of the so-called Industry 4.0 to adapt their production, minimizing manufacturing costs and increasing their manufacturing flexibility to fit the demands of a fickle market. The Industry 4.0 paradigm proposes the use of technologies such as machine learning, big data, simulation, Cyber-Physical Systems or the Industrial Internet of Things in manufacturing to enable intelligent and decentralized manufacturing [2].

Government and industrial institutions all around the world have considered this paradigm as a key factor in their research strategies at mid-long term. This interest is evident with the emergence of initiatives such as Industrial Internet Consortium, Industrial Value Chain Initiative, or China Manufacturing 2025, being the German Platform Industrie 4.0 the reference in Europe [3]–[5]. The approach proposed by this initiative relies in the concept of I4.0 Component. The glossary provided by the Platform Industrie 4.0 describes this concept as a participant of an I4.0 System comprising an Asset Administration Shell (AAS) and an asset. [6]. The same glossary defines the AAS as a virtual representation of the I4.0 Component in the system, containing sub-models which can be accessed through a component manager.

I4.0 Components must meet certain requirements, namely: provide communication capabilities to ensure interoperability between the I4.0 Components within the system (R1); uniquely identify both the AAS and the asset (R2); represent the state and the services (functions and data) of the asset in sub-models (R3); manage the access by other components to the sub-models of the I4.0 Component (R4); enable integration between AASs and assets (R5); and, finally, implement the automation and execution of the services performed by the asset (R6) [7].

Software agents (hereafter simply agents for short) are autonomous software entities capable of competing or collaborating with other software agents in the system to achieve their goals [8]. Agents are a suitable alternative for the implementation of the AAS, since they fulfil interoperability (R1), identification (R2), representation (R3) and sub-model management (R4) requirements, and provide additional features, such as distributed decision-making capabilities [9]–[11]. However, the agent technology does not contemplate either the integration with or management of assets and consequently, the paradigm of industrial agents arises. An industrial agent is an agent which integrates a physical asset [12], and therefore fulfils the integration (R5) and management (R6) of the asset. However, the scalability and reusability of specific integration solutions is still challenging. The reason is that factories usually present a wide diversity of physical assets, which are sometimes supported by systems

This work was financed by MCIU/AEI/FEDER, UE (grant number RTI2018-096116-B-I00) and by GV/EJ (grant number IT1324-19).

difficult to interact with (such as legacy systems or proprietary solutions).

This work proposes a four-layered architecture for the implementation of the different aspects of the integration between agent-based AASs and assets. The division into layers enhances the separation of concerns, encapsulating the different integration aspects at different abstraction levels: an upper layer will implement communication and interoperability capabilities, in addition to the intrinsic capabilities of agents, such us negotiation and decision-making; an intermediate layer will transform the data received from the asset into useful information for the I4.0 Components in the system; and the lower layers will manage the services and the exchange of data resulting from the execution of services or changes in the state of the asset. Besides, this approach makes the proposed architecture complementary to the standard IEEE 2660.1-2020 on recommended practices for industrial agents. This standard focuses on deployment alternatives for the integration between agents and physical assets, identifying four different approaches [13].

The rest of the paper is organized as follows: Section 2 reviews the related work on the integration between agent-based solutions and physical assets. Section 3 presents the architecture proposed by the authors and the required steps for its integration. This methodology is applied in Section 4 to integrate an agent and a robotic arm from an assembly cell following one of the integration practices identified in the standard IEEE 2660.1-2020. Finally, the conclusions and future work are depicted in Section 5.

II. Related Work

The objective of this section is to review research dealing with the integration of agent-based solutions and physical assets. The analysis focuses on the different mechanisms proposed to achieve the communication between the virtual and physical counterparts (agents and physical assets, respectively), rather than on specific details of the implementation of the MAS.

Some works have focused on the application or deployment of specific integration solutions.[14] proposes a multiagent approach for implementing the control of an automated distribution center. This work details the integration between agents and PLCs, consisting of a shared data structure following a specific format. This structure is ad-hoc, considering assets which provides transport services exclusively, as it defines the product id, its origin, and its destination. [15] presents an interface for the communication of agent-based systems with the controllers managing the execution of physical assets in the factory. This interface mentions three main aspects to be considered independently pursuing the separation of concerns: the channel, which can be implemented network-based or using shared memory (just in case both the agent and the control of the asset reside on the same controller); the message, provided in an asynchronous-based way; and the message content, which must be equally understood and interpreted by all the participants. Despite these aspects could be generically implemented, in this approach the message content for data exchange is based on IEC 61499 datatypes. This limits the applicability of this approach to controllers (mainly PLCs) which comply with the standard IEC 61499, excluding other devices. This concept of integration between systems based on agents and PLCs programmed following the standard IEC 61499 can be seen in other works in the literature [16], [17]. [18] presents an agent-based solution applied to control an assembly cell. This work proposes Manufacturing Resource Agents (MRA), which are abstract enough to adapt different assets that provide equivalent services (e.g., two manipulating robots from different developers), relying on the ontology of system. MRAs can integrate different physical assets regardless of their implementation specifications thanks to an Agent-Machine Interface (AMI). The AMI links the MRA and the physical asset, managing the execution of services by the asset in response to service requests received by the MRA. This is an interesting approach, but it does not provide guidance about how to design or implement it. [19] proposes a four-layered architecture approach for the integration of Java Agent Development Framework (JADE) agents and transport vehicles handled with Robot Operating System (ROS). This architecture leans on the negotiation capabilities of JADE agents so they can compete to offer transport services. However, this architecture is ad-hoc, as it is only focused on the integration technologies and the type of services offered by the asset from the use cases. The analysis of these works shows that integration of agents and assets and physical assets is receiving great attention. However, it is also clear that it is not possible to impose a specific integration approach due to the heterogeneity of the assets in the factories.

Other authors have faced the integration between agents and physical assets from a generic perspective. [20] analyzes the requirements for the implementation of agent-based solutions in industrial applications considering three dimensions: integration with physical equipment, real-time constraints, and communication infrastructure. Regarding the first issue, this work discusses the suitability of two alternatives: to run the agent directly in the controller of the asset or run it in an external platform and build an interface. It is concluded that the first option is problematic since these controllers usually do not have the computational resources to host agents. Instead, the second approach is more feasible, but usually requires developing asset-dependent interfaces, and thus, the research towards transparent integration mechanisms is encouraged. [21] proposes an interface to integrate agents with physical assets. This approach is derived from the Manufacturing Message Specification (MMS) protocol and it is based upon two concepts: on the one hand, the virtual resource, which recalls the AAS information sub-models later proposed by RAMI 4.0; on the other hand, a client-server relationship which allows the agent accessing to the services provided by the physical asset, invoking their representations in the virtual resource. These services are defined in compliance with MMS specifications and clustered in libraries). This allows to abstract the agent from the implementation of these services, but limits the available services to those considered in the MMS specifications. [22] puts in practice this approach in a case study comprising heterogeneous physical assets that were integrated with their agents using different technologies (e.g., OPC-UA, Modbus, proprietary serial communication protocols,…). [23] proposes an implementation of the AAS concept, named NOVAAS. This work also points towards the need of an internal interface that connects the physical asset with the agent and proposes a Façade software design pattern for its implementation. Nevertheless, this pattern for the integration with different assets is not discussed in detail and cannot be applied with the information provided. These works present interesting concepts for interfacing agent-based architectures with

physical assets in a generic way (such as the AMI or the virtual resource). However, these approaches have not considered the abstraction of all the integration aspects of the I4.0 Component because are focused on the agent-asset integration [20], are based on previous specifications [21], [22], or only include aspects related to the AAS [23].

III. Four-Layered Architecture Proposal

This section presents the architecture proposed by the authors for the generic integration of agent-based AASs and physical assets. Starting from the idea depicted in [19], the authors rethink the four-layered concept in a generic way within the context of the Industry 4.0 to support the integration with different assets. Subsection A describes the layers comprising the architecture and the interfaces between them, while subsection B details the integration methodology with the steps to implement the architecture.

A. Architecture Overview

The proposed architecture comprises four layers. As Fig. 1 illustrates, two are responsibility of the agent-based AAS and the other two are responsibility of the physical asset. The first two layers implement the AAS of the I4.0 Component extended with communication and distributed decision-making capabilities. These upper layers are implemented in an agent. The other two layers are oriented to manage the asset. These lower layers are deployed at edge level to meet real-time requirements. Each layer is described as follows:

- Intelligence layer: it implements the service-oriented interface of the I4.0 Component, allowing the interaction with other I4.0 Components in the system (R1). This layer also identifies the AAS and the asset so they can be recognized in the communications (R2). Besides, it also implements the decision-making capabilities, based on the information received from outside and the information residing in the management layer.
- Management layer: it corresponds to the component manager of the AAS. It contains the sub-models representing the state and the services of the asset and interacts with the operative layer to keep them updated (R3). It also manages the requests from the intelligence layer to access to the data and functions in the sub-models (R4). Therefore, at this level, the execution of services is abstracted to the transmission and update of information in the sub-models.
- Operative layer: it implements the logic behind the execution of services (R6). This allows the operative layer to interpret the update requests received from the management layer and deliver a response. This layer interacts with the functional layer in case it is required the execution of a service to answer to an update request.
- Functional layer: it performs the services offered by the I4.0 Component (R6). It receives information from the operative layer through the input parameters when invoking the service. Similarly, the results from the execution of the service are returned by means of output parameters.

Layering allows dealing with the requirements separately. However, it does not mean that layers are independent of each other, since all are necessary to fulfill the functionality.

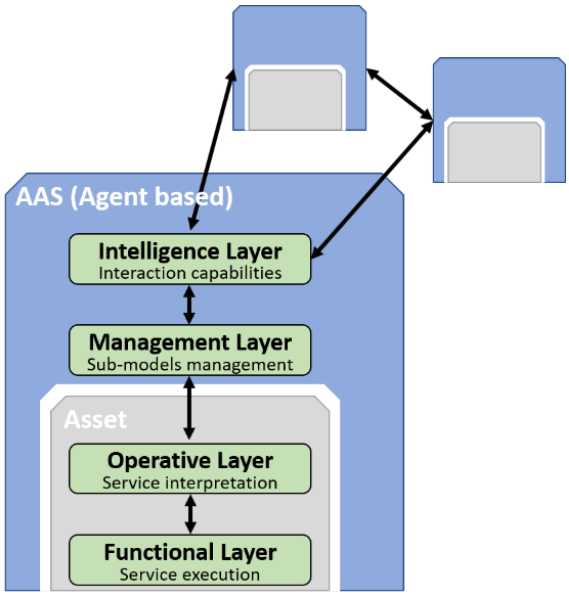


Fig. 1 Correlation between the four-layered architecture and an agent-based I4.0 Component.

Hence, intralayer interfaces are fundamental to implement the proposed approach: the intelligence-management interface must allow data to be sent and collected from the sub-models based on the messages exchanged with other I4.0 components; the management-operational interface handles the integration between the AAS and the physical asset and must translate from sub-models to data structures understandable for the device controlling the asset (R5); and the operative-functional interface must transmit the parameters to initialize a service as well as the results from its execution.

B. Integration methodology

The implementation of this architecture is not straightforward, because it depends to a great extent on the physical asset to be integrated and in the agent. Hence, the authors propose a methodology that consists of the following steps:

- First, the final purpose behind the implementation of an I4.0 Component is providing an asset the capability to connect to other I4.0 Components in the system. Therefore, before considering specific aspects for the integration of each asset, it is necessary to define a common ontology for the system that ensures interoperability (i.e., allowing an offer and request for services understandable and interpreted equally for all participants). This ontology defines the semantics of the messages exchanged by the intelligence layer.
- Once interoperability is guaranteed, the next step is to define the sub-models in the management layer according to the services that its physical asset can perform. Sub-models must be based upon concepts from the ontology to express a service through parameters that the device controlling the asset can interpret. These concepts are common to this first two layers, and therefore, they must drive the implementation of the intelligence-management interface.
- Next step is the implementation of the services that have been represented in the sub-models, which involves implementing the operational and functional

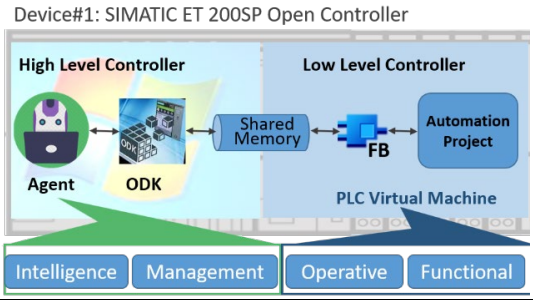
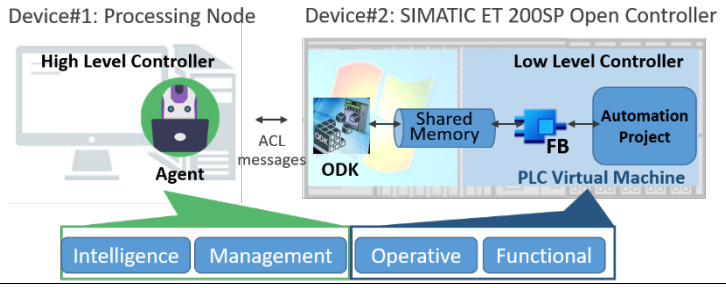
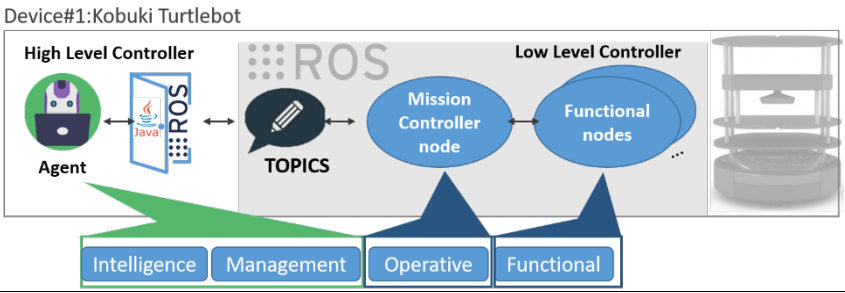
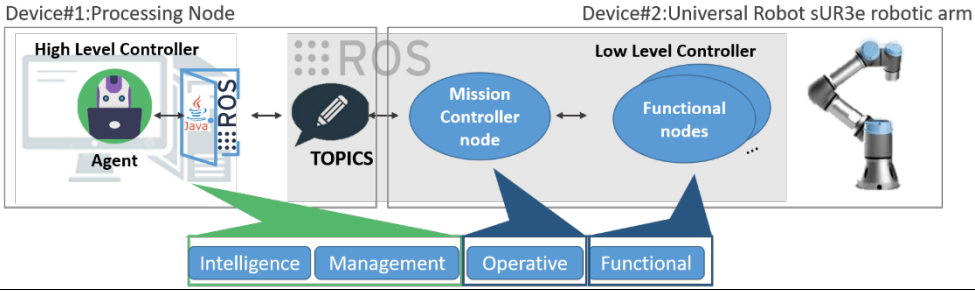
layers. Depending on the scenario, these two layers may be deployed on separated devices or the same, although they should not be mixed in any case. In this way, the operational-functional interface will be the set of input and output parameters for the call of each service.

- The last step should be the implementation of the management-operational interface between the agent implementing the AAS and the device controlling the physical asset. First, this interface must make the remaining formatting adjustments to ensure consistency in the exchange of information in both directions. Once this issue has been solved, they can now be connected.

At this point, the standard IEEE 2660.1-2020 and the proposed architecture are complementary since the standard focuses specifically on interfacing agents and assets. The standard presents four integration practices, attending to the interaction between the agent and the asset, as well as to the location of the agent in terms of deployment. The interaction mode can be either tightly coupled (where the connection is permanent and direct) and loosely coupled (where the connection is mediated via a data-driven communication). Regarding the location, the standard distinguishes between on device (the agent is deployed in the control device of the physical asset) and hybrid (the agent is deployed externally). This way, the IEEE 2660.1-2020 focuses on the integration of the two parts comprising an industrial agent, which, by extension, can implement an I4.0 Component.

On the other hand, the four-layered architecture proposed by the authors is oriented to encapsulate the different aspects required for implementing an I4.0 Component, regardless the integration kind between the counterparts. Thus, the proposed architecture should be applicable to the four integration practices identified in the standard. Table I summarizes the applicability of the architecture in the four scenarios by means of short examples.

TABLE I. APPLICATION EXAMPLES OF THE FOUR-LAYERED ARCHITECTURE TO THE INTEGRATION PRACTICES FROM IEEE 2660.1-2020

Integration practice	Interaction mode
Tightly Coupled, On Device	
Tightly Coupled, Hybrid	
Loosely Coupled, On Device	
Loosely Coupled, Hybrid	

IV. Application of the Methodology

This section presents a case study where the proposed architecture has been applied following the steps described in section 3. More specifically, the case study consists of the integration of an assembly cell and an agent implemented following the design pattern submitted by the authors to ICPS 2021 [24]. This cell, depicted in Fig. 2, consists of a KUKA KR 3 R540 robotic arm that assembles different shaft models for stepper motors. This robot is managed by a KUKA KR C4 Compact Controller and a Siemens SIMATIC ET 200SP Open Controller. This device features a software controller, which allows the execution of IEC 61131-3 compliant automation programs, and a Windows environment, which allows the execution of software agents.

This cell performs manufacturing services consisting of assembling a batch of a variable number of items. The first step to apply the proposed architecture is to ensure the interoperability of the new I4.0 Component with the rest of the I4.0 Components in the system. To that end, the intelligence layer will receive messages with the information required to perform a manufacturing service which must meet a concrete ontology. It has been considered that these messages must provide at least information about the type of service requested and the characterization of the batch receiving the service (including its identity, and the type and number of items it contains). The following concepts emerge from this analysis: service_type, batch_id, product_type and item_id. These concepts and their values constitute the content of the messages received by the agent implementing the intelligence layer and for that reason they must be known by it.

The second step is to implement the sub-models defining the asset and the services it provides. In this case, the sub-model to characterize the services comprises three data structures:

- Configuration: this structure defines the information that must be received to request the service. It includes the identification of the machine offering the service (machine_id) and the information for the characterization of the service request: service_type, batch_id, product_type and n_items. The number of items is obtained from the sum of items that share a common batch_id.
- State: this structure defines the information that must be returned per item manufactured to fulfil the service. It includes the required information to trace the item (i.e., machine_id, service_type batch_id, product_type, and item_id), and the manufacturing times (initial_time_stamp and final_time_stamp).
- Control: this structure defines parameters that are used to manage the interactions between the agent and the asset. This data structure is not accessible from the intelligence layer, but rather allows the management-operational interface to manage the exchange of information. It includes a flag that allows to inform the controller that a new service is required, and two others that allow the controller to notify when an item or the entire service is completed.

Next, the implementation of the services in the physical asset. The operational layer has been implemented in the software controller of the SIMATIC ET 200SP.

Fig. 2 Detail of the assembly cell: a) KUKA KR 3 R540 b) Siemens SIMATIC ET 200SP Open Controller.

The automation project parameterizes the call to the functional layer based on the type of service and the number of items when the new service flag is true. The functional layer has been implemented as a program in the KR C4 Compact Controller. The code is structured modularly, so that it filters first which service must perform and, second, how many items require it. After each item is completed, the operational layer receives the results, sets the item completed flag to true and deliver the update to the management layer. When the last item is manufactured, the service completed flag is set to true.

Last step is the implementation of the management-operational interface. The integration practice selected to perform this application example is *Tightly coupled, On Device*. This approach applies in cases with direct communication and the agent deployed in the device controlling the asset. This option has been chosen because it avoids possible network latencies, improving the performance. It only applies if the device controlling the asset has enough computational capabilities to support the agent, which is the current case. In this case, the integration is achieved using the Siemens ODK library. This library allows the automation project to use function blocks executing functionalities in the Windows part of the device. It also allows to exchange information between the software controller environment and the Windows environment. This way, a gateway is deployed to interact with the agent implementing the management layer.

This example is illustrative of the benefits of the layering proposed in this architecture. If it is compared to the scenario *Tightly coupled, On-Device* from Table I, the difference is that in this case the low-level controller consists of two devices (in this case both the software controller of the SIMATIC ET 200SP and the KR C4 Compact Controller are involved). With respect to the standard, both scenarios represent the same case, but the implementation of the proposed architecture differs on the deployment of lower layers: in the scenario described in this section, the operative layer is implemented in the software controller of the SIMATIC ET 200SP and the functional layer is implemented in the KR C4 Compact Controller; in the scenario depicted in Table I, both layers are deployed in the software controller of the SIMATIC ET 200SP.

V. Conclusions and Future Work

The four-layered architecture presented by the authors proposes the use of industrial agents for the integration of AASs and physical assets. Layering allows the abstraction of different aspects of the I4.0 Component, dividing the efforts to deal with the implementation requirements. This paper also proposes an integration methodology detailing the steps to be followed to apply the architecture. This methodology is illustrated with a case study where a robotic arm is integrated with an agent-based AAS. The proposed architecture is complementary to the standard IEEE 2660.1-2020, since it can be applied to the four integration practices considered in the standard.

In this approach, the proposed AAS can perform pre-processing tasks in the management layer. Thus, an idea for future work is to develop a holistic concept of AAS complementing the current AAS with a part in charge of data processing, analysis of historical data, simulation of hypothetical scenarios, etc., deploying those functionalities above the workshop, that is, in the fog/edge or in the cloud. Furthermore, this paper focuses only on agents for managing physical assets of a factory, but manufacturing systems can comprise further types of agents. For this reason, the authors will work in the future in the development of an architecture for manufacturing systems in the context of Industry 4.0 following the same generic approach than this work.

References

[1] "pwc-economic-slowdown.pdf." Accessed: Feb. 08, 2021. [Online]. Available: https://www.pwc.com/us/en/library/assets/pwc-economic-slowdown.pdf.

[2] C. Santos, A. Mehrsai, A. C. Barros, M. Araújo, and E. Ares, "Towards Industry 4.0: an overview of European strategic roadmaps," *Procedia Manufacturing*, vol. 13, pp. 972–979, 2017, doi: 10.1016/j.promfg.2017.09.093.

[3] Q. Li *et al.*, "Smart manufacturing standardization: Architectures, reference models and standards framework," *Computers in Industry*, vol. 101, pp. 91–106, Oct. 2018, doi: 10.1016/j.compind.2018.06.005.

[4] M. Moghaddam, M. N. Cadavid, C. R. Kenley, and A. V. Deshmukh, "Reference architectures for smart manufacturing: A critical review," *Journal of Manufacturing Systems*, vol. 49, pp. 215–225, Oct. 2018, doi: 10.1016/j.jmsy.2018.10.006.

[5] Fraile, Sanchis, Poler, and Ortiz, "Reference Models for Digital Manufacturing Platforms," *Applied Sciences*, vol. 9, no. 20, p. 4433, Oct. 2019, doi: 10.3390/app9204433.

[6] "Glossary." https://www.plattform-i40.de/PI40/Navigation/EN/Industrie40/Glossary/glossary.html (accessed Feb. 13, 2021).

[7] X. Ye and S. H. Hong, "Toward Industry 4.0 Components: Insights Into and Implementation of Asset Administration Shells," *EEE Ind. Electron. Mag.*, vol. 13, no. 1, pp. 13–25, Mar. 2019, doi: 10.1109/MIE.2019.2893397.

[8] P. Leitão, "Agent-based distributed manufacturing control: A state-of-the-art survey," *Engineering Applications of Artificial Intelligence*, vol. 22, no. 7, pp. 979–991, Oct. 2009, doi: 10.1016/j.engappai.2008.09.005.

[9] W. Shen, Q. Hao, H. J. Yoon, and D. H. Norrie, "Applications of agent-based systems in intelligent manufacturing: An updated review," *Advanced Engineering Informatics*, vol. 20, no. 4, pp. 415–431, Oct. 2006, doi: 10.1016/j.aei.2006.05.004.

[10] R. S. Peres, A. Dionisio Rocha, P. Leitao, and J. Barata, "IDARTS – Towards intelligent data analysis and real-time supervision for industry 4.0," *Computers in Industry*, vol. 101, pp. 138–146, Oct. 2018, doi: 10.1016/j.compind.2018.07.004.

[11] L. A. Cruz Salazar, D. Ryashentseva, A. Lüder, and B. Vogel-Heuser, "Cyber-physical production systems architecture based on multi-agent's design pattern—comparison of selected approaches mapping four agent patterns," *Int J Adv Manuf Technol*, Jul. 2019, doi: 10.1007/s00170-019-03800-4.

[12] S. Karnouskos, P. Leitao, L. Ribeiro, and A. W. Colombo, "Industrial Agents as a Key Enabler for Realizing Industrial Cyber-Physical Systems: Multiagent Systems Entering Industry 4.0," *IEEE Industrial Electronics Magazine*, vol. 14, no. 3, pp. 18–32, Sep. 2020, doi: 10.1109/MIE.2019.2962225.

[13] "IEEE Recommended Practice for Industrial Agents: Integration of Software Agents and Low-Level Automation Functions," *IEEE Std 2660.1-2020*, pp. 1–43, Jan. 2021, doi: 10.1109/IEEESTD.2021.9340089.

[14] J. de las Morenas, A. Garcia-Higuera, and P. Garcia-Ansola, "Shop Floor Control: A Physical Agents Approach for PLC-Controlled Systems," *IEEE Trans. Ind. Inf.*, vol. 13, no. 5, pp. 2417–2427, Oct. 2017, doi: 10.1109/TII.2017.2720696.

[15] M. Merdan, W. Lepuschitz, I. Hegny, and G. Koppensteiner, "Application of a communication interface between agents and the low level control," in *2009 4th International Conference on Autonomous Robots and Agents*, Feb. 2009, pp. 628–633, doi: 10.1109/ICARA.2000.4804006.

[16] W. Lepuschitz, M. Vallée, A. Zoitl, and M. Merdan, "Online reconfiguration of the low level control for automation agents," in *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, Nov. 2010, pp. 1365–1370, doi: 10.1109/IECON.2010.5675483.

[17] W. Dai, V. N. Dubinin, J. H. Christensen, V. Vyatkin, and X. Guan, "Toward Self-Manageable and Adaptive Industrial Cyber-Physical Systems With Knowledge-Driven Autonomic Service Management," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 725–736, Apr. 2017, doi: 10.1109/TII.2016.2595401.

[18] G. Cândido and J. Barata, "A Multiagent Control System for Shop Floor Assembly," in *Holonic and Multi-Agent Systems for Manufacturing*, vol. 4659, V. Mařík, V. Vyatkin, and A. W. Colombo, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 293–302.

[19] J. Martin, O. Casquero, B. Fortes, and M. Marcos, "A Generic Multi-Layer Architecture Based on ROS-JADE Integration for Autonomous Transport Vehicles," *Sensors*, vol. 19, no. 1, p. 69, Jan. 2019, doi: 10.3390/s19010069.

[20] A. Pereira, N. Rodrigues, and P. Leitão, "Deployment of multi-agent systems for industrial applications," in *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies Factory Automation (ETFA 2012)*, Sep. 2012, pp. 1–8, doi: 10.1109/ETFA.2012.6489641.

[21] P. Leitão, R. Boissier, F. Casais, and F. Restivo, "Integration of Automation Resources in Holonic Manufacturing Applications," in *Holonic and Multi-Agent Systems for Manufacturing*, vol. 2744, V. Mařík, D. McFarlane, and P. Valckenaers, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 35–46.

[22] J. Dias, J. Barbosa, and P. Leitão, "Deployment of industrial agents in heterogeneous automation environments," in *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, Jul. 2015, pp. 1330–1335, doi: 10.1109/INDIN.2015.7281928.

[23] G. di Orio, P. Maló, and J. Barata, "NOVAAS: A Reference Implementation of Industrie4.0 Asset Administration Shell with best-of-breed practices from IT engineering," in *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, Oct. 2019, vol. 1, pp. 5505–5512, doi: 10.1109/IECON.2019.8927081.

[24] A. López, O. Casquero, and M. Marcos, "Design patterns for the implementation of Industrial Agent-based AASs," in *2021 IEEE Conference on Industrial Cyberphysical Systems (ICPS)*, May 2021, p. 6.