A Methodological Approach for Integrating Physical Assets in Industry 4.0

Alejandro López, Elisabet Estévez, Oskar Casquero, and Marga Marcos, *Senior Member, IEEE*

***Abstract*—The paradigm of the fourth industrial revolution poses the combined use of technologies that provide companies with adaptability to face uncertain market scenarios. Different institutions have presented their reference architectures to apply principles of the fourth industrial revolution in a standardized way, being the German proposal Industry 4.0 (I4.0) the most widespread. Nowadays most companies are still in the computerization and digitalization phases prior to being able to connect to I4.0. This situation highlights the need for solutions for the integration of physical assets in I4.0. However, most of the work in this area focuses on ad-hoc solutions, or at least technology-dependent ones. This article presents a multilayer approach for integrating physical assets in I4.0. The goal is to separate the different concerns related to the integration of physical assets into abstract layers. This approach allows addressing these concerns in a generic way, regardless of the type of asset and/or the technology used, as well as efficient, since it addresses each concern separately. This approach is complemented by a methodology that provides guidelines for integrating physical assets according to the multilayer approach. The applicability of the multilayer approach and the integration methodology is put to the test by integrating a roboticarm.**

***Index Terms*—Asset administration shell (AAS), asset integration, Industry 4.0 (I4.0), Reference Architectural Model for Industrie 4.0 (RAMI 4.0).**

I. Introduction

THE paradigm of the fourth industrial revolution is based on the acquisition, processing, and use of information generated at all levels of an organization to modify or optimize its industrial processes [1]. To do so, it relies on the combined use of technologies such as cyber-physical systems (CPS), Internet of Things (IoT), artificial intelligence, augmented reality, or big data [2], [3]. The use of these technologies provides companies with greater adaptability, making them more competitive and able to overcome severe market disruptions, such as the one caused by the COVID-19 pandemic [4].

However, the fourth industrial revolution aims to achieve total connectivity not only within organizations, but globally. This necessarily implies a standardized use of the aforementioned technologies, rather than ad-hoc implementations. To this end, different public and private institutions all over the world have been working for years to set a contextual framework for the fourth industrial revolution. In this regard, the Reference Architectural Model for Industrie 4.0 (RAMI 4.0), proposed by the German initiative Plattform Industrie 4.0 (also referred as Industry 4.0 or simply I4.0), stands out by being the oldest reference architecture (2015), serving as a baseline for other reference architectures [5], [6]. RAMI 4.0 contextualizes I4.0 by means of a three-dimensional (3-D) map, which ensures a common understanding between its participants, called I4.0 Components, which are assets integrated in I4.0 through an asset administration shell (AAS) [7].

In this regard, the German National Academy of Sciences and Engineering (Acatech) proposed the I4.0 Maturity Index [8]. This document is intended to ease the transition for companies toward I4.0. To that end, Acatech has defined a roadmap consisting of six stages: computerization, connectivity, visibility, transparency, predictive capacity, and adaptability. The first two steps of this roadmap conform a preliminary digitalization phase that must enable the basic conditions for I4.0: the computerization stage is related to the automation of manufacturing services, while the connectivity stage focuses on communication with and between assets to provide accessibility to these services. This becomes relevant when analyzing the results from the survey published by Acatech regarding the current situation of companies with respect to the maturity index [9]: In April 2020, only 4% of the participants had achieved the first stage of I4.0 (visibility). Therefore, a 96% of the surveyed companies were still dealing with the computerization (16%) and connectivity (80%) stages (see Section II-A of [9] for further detail on these figures).

These results are striking to say the least and lead us to wonder why so many companies have not yet passed the connectivity stage. The answer to this question could be that the integration of physical assets is a complex task due to the diversity of assets that are usually present in factories, with large differences in their functional and communicative capabilities between them. This explains why, despite significant research efforts in this

Manuscript received 23 September 2022; revised 24 November 2022; accepted 13 December 2022. Date of publication 20 December 2022; date of current version 24 July 2023. This work was supported by the MCIU/AEI/FEDER, UE under Grant RTI2018-096116-B-I00. Paper no. TII-22-4032. *(Corresponding author: Alejandro López.)*

Alejandro López, Oskar Casquero, and Marga Marcos are with the Systems Engineering and Automatic Control Department, University of the Basque Country, 48013 Bilbao, Spain (e-mail: alejandro.lopez@ehu.eus; oskar.casquero@ehu.eus; marga.marcos@ehu.eus).

Elisabet Estévez is with the Electronics and Automation Engineering Department, University of Jaén, 23071 Jaén, Spain (e-mail: eestevez@ujaen.es).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2022.3230714.

Digital Object Identifier 10.1109/TII.2022.3230714

area, solutions are usually ad-hoc, targeted to specific kinds of assets or integration technologies [10], [11], [12]. Given this situation, following two research questions (RQs) arise:

1) RQ1: Are there currently methodologies to guide in the integration of physical assets in I4.0?
2) RQ2: If so, do they allow seamless integration (i.e., regardless the type of asset or the technologies required)?

Aiming to answer these questions, this article presents a multilayer approach for asset integration within I4.0 Components. This approach focuses on physical assets, since their integration is not as straightforward as it is for logical assets (e.g., intangible assets, such as databases with process traceability information or company know-how), making it more difficult to implement generic solutions. Therefore, in the following, by asset, we mean physical asset, unless noted otherwise. The objective of this approach is to ease the integration of assets and provide I4.0 Components with access to asset services (i.e., services that offer asset related functionalities). To that end, the concerns related to asset integration have been abstracted into layers, so they can be addressed both generically (i.e., regardless of the type of asset to be integrated and the technologies required) and independently (as they are decoupled from each other). Along with the multilayer approach, an integration methodology is proposed that provides guidelines to implement each layer, resulting on a solution tailored to the users' needs.

The remainder of the article is organized as follows. Section II analyzes the solutions present in the literature to integrate physical and virtual counterparts from the perspective of I4.0 Components and CPS. Section III presents the approach proposed in this article following a top-down approach to define the layers, while Section IV follows a bottom-up approach to describe the integration methodology. This methodology is applied in Section V to implement an integration for a robotic arm controlled by a programmable logic controller (PLC). Section VI rounds up the article with the conclusions and the future article.

II. Related Work

The integration of assets with their virtual counterparts has captioned the interest of researchers for a long time, in the context of holonic manufacturing systems [13], and more recently, in the context of digital twins [14], CPS [15], and I4.0 Components, which propose the AAS as the mean to integrate assets in the system [16]. In this section, we focus on reviewing existing asset integration approaches in the literature that are aligned with I4.0, to analyze whether they follow any integration methodology (RQ1), and if so, if it is seamless or it is focused on specific assets or technologies (RQ2).

There are a large number of articles devoted to this problem. However, most of them are targeted to solve specific integration problems. Some works describe ad-hoc implementations for specific assets as case studies [17], [18]. Others, although not ad-hoc, focus on a specific type of asset. For instance, several articles focus on integrating PLCs. Most of them are oriented to standards such as IEC 61131-3 [11], [19], [20] and IEC 61499 [21]. These articles also present different alternatives regarding where to deploy the AAS: inside the PLC as a function block [19], or in an external server, outside the PLC [20]. The latter defend that the interactions of the AAS may compromise the real-time constraints of PLC programs.

Another type of asset receiving great attention is robots. Robot operating system (ROS) is commonly used at different levels for integrating robots [12], [22]: some researchers use ROS as a means of communication between AASs and assets [12]; others, instead, propose to use ROS for interactions between AAS, choosing the most suitable communication protocol to interact with each asset [22]. OPC unified architecture (OPC UA) is also used as an alternative to integrate robots, given its suitability to transfer different types of data structures easily [23]. In the opinion of the authors, these solutions are suitable for solving specific problems, as they propose the technology best suited to each type of asset. However, this also limits them in terms of reusability and scalability.

In other articles, the contribution does not focus on the type of asset to be integrated, but on which technologies to use for this purpose. In this regard, several researchers propose solutions based on the technologies proposed by Plattform I4.0. For instance, Contreras et al. [24] proposed using standards, such as OPC UA (IEC 62541) for the communication between the asset and the AAS, and a combination of Field Device Integration (IEC 62769), AutomationML (IEC 62714), and OPC UA for the communication between AASs. Fuchs et al. [25] proposed an AAS submodel to represent devices at the field level based on the OPC UA Part 100: Devices specification standard.

Another option is using industrial agents for the integration of assets, following international standards such as the IEEE 2660.1-2020 on recommended practices for industrial agents [26]. Industrial agents meet the requirements of I4.0 thanks to the natural predisposition of software agents (i.e., the virtual part of industrial agents) to collaborate and compete to achieve their goals [10]. Moreover, they naturally address the use of ontologies, which are a key feature to achieve interoperability in I4.0 systems [27]. For example, Sakurada et al. [28] proposes an agent-based AAS concept whose functionalities are coded as microservices, which must include the agent-based AAS interface (to interact with other agent-based AASs), the asset interface (to interact with the asset), and the asset information (to store the static information of the asset); Leitão and Barbosa [29] proposed an agent-based solution to stablish standard interfaces, regardless the concrete communication protocols, presenting an approach based on the use of the ISO 9506 Manufacturing Message Specification international standard; Bakliwal et al. [30] proposed a three level approach, in which the first layer which receives the information from the asset and provides it a standardized format, so upper data processing layers can be generic; similarly, López et al. [31] proposed a four-layered integration approach complementary to the standard IEEE 2660.1-2020, in which the lower layers are implemented in the asset, and the upper ones in the AAS. These solutions can be applied to different assets, but they still depend on the characteristics of the technology used to implement the integration solution. This can be a limitation in the event of having to integrate an asset that is not compatible with the technologies used in the solution.

Beyond papers oriented to specific kinds of assets of based on specific technologies, there is another group of works which follow a more generic approach, sharing one common

TABLE I
SUMMARY OF COMPLIANCE WITH INTEGRATION CRITERIA

Integration proposal	R1	R2	R3
[11], [17], [20], [23]	No	No	No
[10], [24], [25], [28], [29], [30]	Yes	No	No
[21]	No	Yes	No
[12], [22]	No	No	Yes
[32], [33], [34]	Yes	Yes	No
[26], [31]	Yes	No	Yes

concern: heterogeneity in AAS development, which includes asset integration. In this sense, Ashtari Talkhestani et al. [32] proposed a software module consisting of an individual interface, particularized to the capabilities of the asset to which it will give access, and a generic interface on top to facilitate interoperability. In [33], a methodology for the implementation of AASs is proposed. Within this methodology, the third step corresponds to asset integration. This step comprises the creation of databases, implementation of a communication protocol, and IoT integration. Finally, Ye et al. [34] presented a methodology to implement an AAS enabling the concept of plug and produce. This methodology comprises five steps (physical setup, preintegration, digital representation, AAS communication, and system assessment), from which the first three are related to the integration of assets and AASs. All these articles propose methodologies for implementing AASs, including the integration of their assets. However, none of them provide guidelines to facilitate this integration; at most, they propose the use of some technologies for this purpose.

After analyzing these works and taking into account the RQs raised in this article, the requirements that an approach for asset integration in I4.0 should meet have been identified: first, it should be based on generic concepts that are applicable to any type of asset (R1, asset awareness); second, it should be, as far as possible, independent of any specific technology (R2, technology independence); and third, it should include guidance for carrying out the integration process (R3, asset integration methodology). A solution that meets these requirements would help companies move beyond the stages of computerization and connectivity and integrate their assets into I4.0.

Table I summarizes the compliance of the articles reviewed in this section with respect to these requirements. As it can be seen, some of these articles address two out of three criteria, but none of them meet all three criteria together. On the one hand, some articles are too specific, addressing integration by focusing only on one type of asset or using a particular technology. On the other, articles that presenting a generic approach do not provide details on asset integration. But, as far as the authors know, there is no integration proposal that meets the three requirements together. This article addresses this problem by presenting a multilayer approach to abstract the concerns involved in the integration of assets regardless the technology to use and the type of asset to integrate.

To that end, this approach follows concepts from some of the reviewed works, such as the use of internal interfaces to enable AAS-asset communication [29], which are generic and reusable for assets of the same type [34], but takes them one step further: by defining the layers, the interfaces between them are also defined, with specific needs and requirements. In addition, an integration methodology is proposed to facilitate decision-making, regarding where to deploy each layer, which technology to use, and how to implement them.

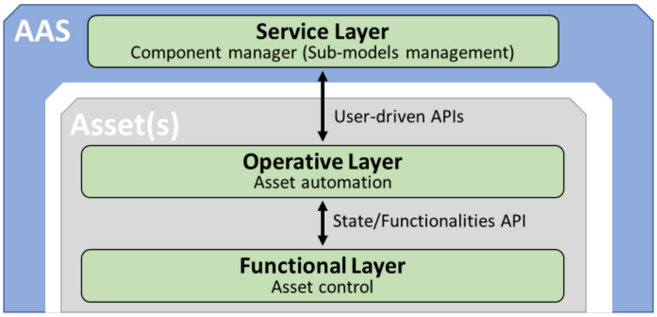


Fig. 1. Multilayer approach overview.

III. MULTILAYER APPROACH

The integration of physical assets in I4.0 is a complex task involving several concerns, such as asset-related information management, asset automation, or service implementation. However, if these concerns are properly identified and decoupled, they can be addressed in a generic way, which can be applied to assets of any type (R1). Furthermore, if the concerns are defined in an abstract manner, they can be addressed regardless of the technology required for each use case (R2).

This section presents a multilayered approach resulting from the subdivision of these concerns into abstract layers. This approach, depicted in Fig. 1, is structured in three layers. An upper layer (service layer) handles service requests in a standardized way [35], interacting with the physical assets when the requests involve asset services. An intermediate layer (operative layer) supervises the execution of the functionalities comprising asset services, and provides the service layer with methods organized in application program interfaces (APIs) as appropriate in each case. Finally, a lower layer (functional layer) implements the functionalities that develop asset services and manage the state of the asset. These are offered to the operative layer through another API. The following subsections will describe these layers in detail.

A. Service Layer

As its name suggests, the service layer offers standardized services at the I4.0 Component level, which may be of interest to other I4.0 Components. At this level, the execution of any service is abstracted to information in the submodels of the AAS. This layer must manage these submodels to handle and respond in an appropriate manner to the service requests it receives, so it must implement the Component Manager.

In this regard, it is fundamental to determine a format for the service requests. As shown in Fig. 2, they may be divided into two categories: 1) meta information and 2) core information. Meta information consists of those data, which does not have an influence in the execution of a service, but they are required to

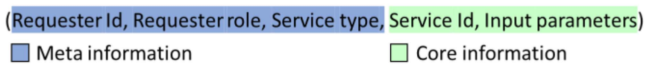

(Requester Id, Requester role, Service type, Service Id, Input parameters)
Meta information
Core information

Fig. 2. Information required to attend a service request at the service layer.

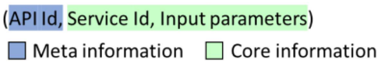


Fig. 3. Information required to transmit a service request to the operative layer.

manage a service request. It must include information regarding the service requester and its role in the system (to determine if the requester has access to the service), and the type of service (to determine how to handle it). Core information consists of those data that are actually necessary to perform the service. It must allow to identify and parameterize the service requested.

The type of service becomes relevant at this layer, since it has an impact on how service requests are handled. Plattform I4.0 differentiates between application relevant services, and systemic relevant services (infrastructure services), which are out of the scope of this article [35]. With respect to application relevant services, two different types of services are identified as follows:

1) Submodel services, which can be performed by accessing all or part of the information within a submodel. These services are fulfilled by the service layer without interacting with the asset. Services of this type are queries (e.g., concerning the list of available services, the serial number of the asset, etc.).
2) Asset services, which require interacting with the asset to access to updated status information, or to the results of the execution of some specific functionality. In addition, these services allow the management of the asset (start-up, controlled shutdown, etc.).

If a service request corresponds to an asset service, it is necessary to transmit the core information of the service to the operative layer through the service-operative interface. This interface is organized into APIs to efficiently manage access to services, so it is necessary to check whether the requester has access to the API offering the service at issue. To do so, the list of API Ids associated with the user's role is checked (the number and type of roles depends on the domain, although some common examples might be asset operator or production manager). Thus, as illustrated in Fig. 3, if the requester has access, the core information is sent to the operative layer, and, if not, it is reported that the requested service is not accessible.

B. Operative Layer

This layer is conceived to support the service layer: when it receives requests concerning asset services, the service layer cannot carry out the services by itself, and it leans on the operative layer. To do this, the operative layer automates the asset. This implies, on the one hand, monitoring its status and acting on it or reporting events as appropriate in each case; on the other hand, it implies managing the asset service requests, i.e., it has to infer the asset operations (smaller, more manageable functionality fragments) corresponding to a service id, and trigger them in an organized way. These operation(s) are invoked through the operative-functional interface. At this level, each asset operation corresponds to a direct call to a functional unit at the functional layer (e.g., a robot program). The input parameters corresponding to each asset operation are sent through this interface, receiving in response the results from each operation. These partial results can be reported to the service layer directly or at the end of the service, as required.

C. Functional Layer

This layer is in charge of controlling all the asset operations required to carry out the asset services. These operations can basically be divided into two groups: either they are used to ask the asset to do things (i.e., to execute operations that fulfill a specific functionality taking into account the input parameters, if any) or they are used to ask the asset for information (i.e., to respond to queries about the asset status). Finally, note that the execution of these operations must always involve a response to the operative layer, even if it is simply to confirm that the operation has been completed.

IV. Integration Methodology

This section proposes a four-step integration methodology, which complements the multilayer approach (integration criteria R3, see Table I). The objective of this methodology is to provide indications and recommended practices for implementing the multilayer approach. The steps comprising the methodology are detailed in the following subsections.

A. Physical Arrangement: Analysis of Available Assets

The first aspect to consider is the physical arrangement, i.e., the auxiliary assets related to the asset we want to integrate (hereinafter, main asset). In most cases, the integration of an asset in I4.0 involves further assets. A common case is when the main asset (e.g., a robot) is managed by another asset (e.g., a PLC).

It is also usual that the main asset requires of other assets to perform its services (e.g., a gripper, a buffer of consumable materials, etc.). Finally, depending on the communication capabilities of the main asset, additional assets may be required to connect it to the I4.0 system (e.g., a single-board computer, such as Raspberry Pi).

It is important to identify all these auxiliary assets, since they may be necessary in the integration process (either for the execution of services or for the deployment of the layers).

B. Service Definition: Specification of Interlayer Interfaces

The next step consists of specifying the services to be offered by the main asset. As explained before, these services must be specified following an agreed format for the whole plant, containing meta information to determine how to process the service, and core information to identify and parameterize it. In case there is not yet an established format to characterize the services, it is recommended to define it considering all the assets of the plant to ensure the format is generic.

Note that when the services defined are asset services, every layer is involved. Therefore, it is necessary to define the methods to be offered by one layer to the upper one through the interfaces. As a rule, the operative layer must offer the service layer one method for each asset service, through one or several APIs (depending on how many roles are authorized to request that service). Regarding the functional layer, it must offer the operative layer one or more methods for each asset service (depending on the number of operations or queries needed to perform the service).

C. Deployment: Choosing Device and Technology

The third step is to determine where to deploy each layer and which technology to use for it, considering, on the one hand, the functional and communicative capabilities of the available assets (i.e., the main asset plus the auxiliary assets) and, on the other hand, the requirements of each layer. For example, in most cases, the operative and functional layers will require real-time constraints, while the service layer will require good communication capabilities and the capacity to store the submodels of the AAS.

It should also be noted that it is possible to deploy more than one layer in the same asset, if applicable (e.g., a PLC would implement the operative and functional layers of an actuator controlled and managed by itself). However, if this decision is made, it is highly recommended to decouple the layers to preserve the reusability and adaptability of the solution.

D. Development: Bottom-Up Implementation

Finally, the last step is the development of both the layers and the interfaces between them. It is proposed to follow a bottom-up approach, since each layer builds on the layer below it. Hence, the functional layer must first be implemented to provide the operations upon which the asset services are built.

Next, the operative layer must be implemented. At this point, it is important to ensure connectivity between the operative and the functional layers. In addition, the asset hosting the operative layer must manage a directory with the list of operations required to perform each service.

Finally, the service layer must be implemented. In this sense, although [26] is a standard oriented exclusively to industrial agents, it offers some considerations that can be applied generically. After selecting an approach, it is necessary to develop the submodels that the I4.0 Component will need.

V. Proof of Concept

In this section, the methodology proposed in Section IV is used to integrate the demonstrator presented in Fig. 4(a). This demonstrator recreates an industrial case in which several assets in the same cell are involved in the integration, automation, and control of the asset that provides asset services, in this case, a robot. Given its representativeness, it has been chosen for this asset integration problem proof of concept. This robot is placed on a worktable from which it has access to different 3-D printed parts emulating the parts of a stepper motor. The robot can combine these parts to perform a total or partial assembly of the stepper motor, offering 10 different assembly services. To perform these services, the robot moves the pallet from the input position to its working position. These services are applied to batches of different sizes that are transported on pallets with a maximum capacity of six items. For this purpose, the robot places the pallet in a working area and performs the assemblies directly on the positions of each item on the pallet. The following subsections correspond to the application of each step of the integration methodology to this example.

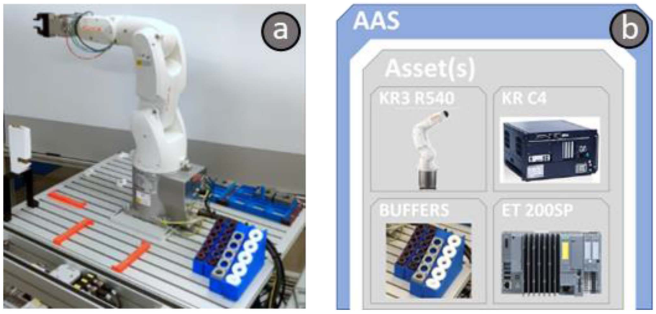


Fig. 4. Detail of (a) demonstrator and (b) I4.0 Component.

A. Physical Arrangement: Analysis of Available Assets

The first step is to identify which assets are available. The main asset is a KUKA KR3 R540 robot. This robot performs assembly operations by picking parts from the buffers it has on its working table. It is controlled by a KR C4 compact controller. In turn, the robot controller is managed by a Siemens ET 200SP open controller. This equipment has a virtualized software controller (i.e., a PLC), and an embedded Windows 10 operating system, with higher processing capabilities. Both environments are isolated from each other but can exchange data through shared memory or virtualized network interfaces. These assets are involved in the resulting I4.0 Component, shown in Fig. 4(b).

B. Service Definition: Specification of Interlayer Interfaces

In the second step, we have defined the services offered by the main asset, and the methods offered by the interfaces. As explained at the beginning of this section, this robot can perform 10 assembly services, which we will call assembly_X(nOfItems), where “X” represents the assembly number (from assembly_1 to assembly_10). These services imply the execution of an assembly operation in order to produce a variable number of items.

Hence, an input parameter is required in order to specify the number of items to be manufactured. With this in mind, it has been decided that in this example the core information required

TABLE II
LIST OF VARIABLES EXCHANGED BETWEEN THE OPERATIVE AND THE FUNCTIONAL LAYERS

Manufacturing-related signals			
Name	**Type**	**PLC address**	**Description**
numberofItems	Int	%QB2247	Input parameter. Indicates the number of items to be assembled.
executeOperation	Bool	%Q2248.0	Auxiliar variable used to trigger an operation after validating the input parameters.
itemStarted	Bool	%I2246.0	Output parameter. Indicates that the robot has started the assembly of a new item.
itemCompleted	Bool	%I2246.1	Output parameter. Indicates that the robot has finished the assembly of a new item.
operationStarted	Bool	%I2247.0	Output parameter. Indicates that the robot has started the assembly operation.
operationCompleted	Bool	%I2247.1	Output parameter. Indicates that the robot has finished the assembly operation.
itemcompletedNumber	Int	%IB2248	Output parameter. Indicates the number of items of the batch already completed.
Asset- management-related signals			
Name	**Type**	**PLC address**	**Description**
$EXT_START	Bool	%Q2001.2	External start. Allows to start a robot program externally
$MOVE_ENABLE	Bool	%Q2001.3	Move enable. Allows the robot to run externally
$CONF_MESS	Bool	%Q2001.4	Confirmation message prompt.
$DRIVES_OFF	Bool	%Q2001.5	Switch on the drives.
$DRIVES_ON	Bool	%Q2001.6	Switch off the drives.

in the service requests consists of the name of the service and the value of its parameter [e.g., assembly_4(6)].

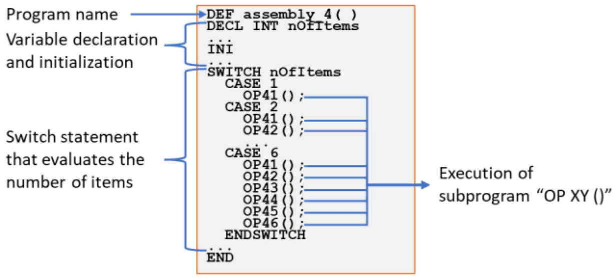
These are asset services, so interaction between layers is necessary to perform them. Thus, the operative layer must provide methods to attend these services, by which they receive the input parameter and return as output timestamps corresponding to the start and completion times of both the service as a whole and of each of the items separately. Note that the item timestamps must be reported as each one is completed. In addition, it is required to include an identifier of the machine performing the service. Similarly, the functional layer must provide methods to the operative layer in which the input parameter is received, and a signal is returned to true at the beginning and at the end of the production of each item.

C. Deployment: Choosing Device and Technology

The third step consists of choosing the asset and technology to deploy each layer according to its needs. Following a bottom-up approach, we started by deciding the location of the functional layer. This layer must have a direct access to the main asset to control its operations. Thus, it has been deployed on the KR C4 compact controller as a robot program.

Regarding the operative layer, its tasks involve filtering the service requests, interpreting them and invoking the operations necessary to perform the service. All these functions require intensive data exchange and good communication capabilities both southbound (to interact with the functional layer) and northbound (to interact the service layer). Having this in mind, the operative layer has been deployed in the PLC part of the ET 200SP as an automation project. Regarding the access to the functional layer, the PLC is connected to the KR C4 compact controller via Profinet. Through this communication protocol, the PLC will have access to the assembly operations implemented in the functional layer.

Finally, with respect to the service layer, it must implement the Component Manager in order to address the service requests. Depending on the type of service, this implies retrieving information from the submodels of the AAS or accessing to the asset (i.e., the robot) via the operative layer. It has been decided to deploy this layer in the Windows environment of the ET 200SP. On the one hand, this ensures a high computational capacity without compromising the real-time constraints of the PLC part of the ET 200SP. This has allowed to implement the service layer as a software agent, which is appropriate to implement the AAS, given its autonomy and cooperation capabilities [10]. On the other hand, having the service and the operative layers deployed in the same asset, the communication has been stablished by using shared memory. This ensures low latencies, enhancing the efficiency and reliability of communications.



```
DEF assembly_4( )
DECL INT nOfItems
...
INI
...
SWITCH nOfItems
   CASE 1
      OP41() ;
   CASE 2
      OP41() ;
      OP42() ;
      ...
   CASE 6
      OP41() ;
      OP42() ;
      OP43() ;
      OP44() ;
      OP45() ;
      OP46() ;
   ENDSWITCH
...
END
```

Program name
Variable declaration and initialization
Switch statement that evaluates the number of items
Execution of subprogram “OP XY ()”

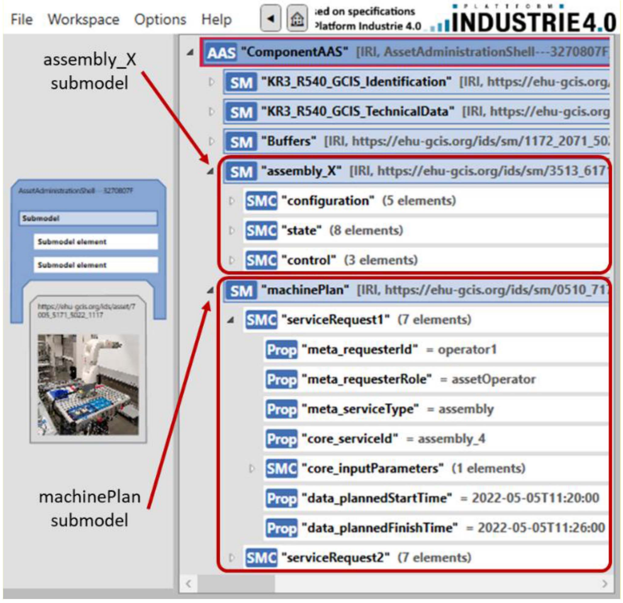
Fig. 5. Pseudocode representing the functional layer.

D. Development: Bottom-Up Implementation

In the last step, the methods defined for each layer have been implemented. Starting with the functional layer, it has been developed in the form of robot programs in the KR C4 compact controller. Fig. 5 exemplifies the program structure for the case of assembly_4(nOfItems) service. It presents an initial part for declaration and initialization of variables, followed by a switch statements that evaluates the number of items requested (SWITCH nOfItems in Fig. 5). As a result, one or more subprograms are executed. These subprograms are identified as “OP XY (),”where “X” corresponds to the service number (in this case, 4), and “Y” corresponds to the position of the item in

TABLE III
DATA STRUCTURES EXCHANGED BETWEEN THE SERVICE AND THE OPERATIVE LAYERS

CONFIGURATION		
Name	**Type**	**Description**
meta_requesterId	string	Id of the requester.
meta_requesterRole	string	Role of the requester.
meta_serviceType	string	Type of service
core_serviceId	string	Service reference.
core_noofItems	Int	Number of items requested.
State		
Name	**Type**	**Description**
meta_requesterId	string	Id of the requester.
meta_requesterRole	string	Role of the requester.
meta_serviceType	string	Type of service
meta_machineReference	int	Machine reference.
meta_itemNumber	int	Number of items finished.
data_initialTimeStamp	long	Item start TimeStamp.
data_finalTimeStamp	long	Item end TimeStamp.
data_serviceTimeStamp	long	Service start TimeStamp.
Control		
Name	**Type**	**Description**
flag_NewService	bool	New service request.
flag_ItemCompleted	bool	Completed item indicator.
flag_ServiceCompleted	bool	Completed service indicator.



AAS "ComponentAAS" [IRI, AssetAdministrationShell---3270807F
SM "KR3_R540_GCIS_Identification" [IRI, https://ehu-gcis.org,
SM "KR3_R540_GCIS_TechnicalData" [IRI, https://ehu-gcis.org
SM "Buffers" [IRI, https://ehu-gcis.org/ids/sm/1172_2071_50
SM "assembly_X" [IRI, https://ehu-gcis.org/ids/sm/3513_617
SMC "configuration" (5 elements)
SMC "state" (8 elements)
SMC "control" (3 elements)
SM "machinePlan" [IRI, https://ehu-gcis.org/ids/sm/0510_71
SMC "serviceRequest1" (7 elements)
Prop "meta_requesterId" = operator1
Prop "meta_requesterRole" = assetOperator
Prop "meta_serviceType" = assembly
Prop "core_serviceId" = assembly_4
SMC "core_inputParameters" (1 elements)
Prop "data_plannedStartTime" = 2022-05-05T11:20:00
Prop "data_plannedFinishTime" = 2022-05-05T11:26:00
SMC "serviceRequest2" (7 elements)

Fig. 6. AAS overview (snapshot generated with the “AASX Package Explorer” tool).

the pallet. For example, if the number of items requested is six (CASE 6 in Fig. 5), this will result in the consecutive execution of six subprograms, from OP41() to OP46(). Each of them will make the robot perform the assembly in each of the six positions available in the pallet. In return, at the beginning and at the end of each subprogram, different signals are used to indicate that the assembly has started or finished. In addition, an integer is also returned to indicate the number of items completed.

Following with the operative layer, we have specified the information to be exchanged with the functional layer in order to invoke its methods. Table II lists the parameters exchanged between the PLC part of the ET 200SP (operative layer) and the robot controller (functional layer). Next, a function has been developed to process the results received from the functional layer. This function generates the timestamps for the production of the items, using the itemStarted and itemCompleted signals (see Table II), and stores them in the data structures used to interact with the service layer. Besides, the itemCompleted signal is used to load the machine reference and the item number into the state data structure.

Concluding with the service layer, we have started once again by specifying the data exchange between this layer and the operative layer. As shown in Table III, information has been divided into three data structures: configuration, which includes the information required to request the service; state, which defines the information to be returned as a result; and control, which defines the internal flags that are used to regulate the interactions between service and operative layers.

After ensuring integration between the service and operative layers, the last step was to define the submodels of the AAS. With respect to this proof of concept, there is a submodel representing the services described in this section (assembly_X submodel), comprising the data structures presented in Table III. There are also submodels representing the assets involved in the integration process (the robot itself, the buffers, etc.). Finally, a machinePlan submodel has been included to register the requests concerning asset services. Thus, this submodel fulfills two functions: on the one hand, it allows the service layer to manage the service requests it receives in an orderly fashion; on the other hand, it allows third parties to query the workload of the I4.0 Component. Fig. 6 overviews the resulting AAS.

VI. CONCLUSION

The aim of this article is to help to deal with integration of physical assets into I4.0. On the one hand, the proposed multilayer approach allows to divide the services into more manageable tasks. On the other hand, the integration methodology serves a double purpose: first, to separate the design of the services from the implementation through the layers; and second, to define when decisions must be made that link the implementation to specific technologies.

In the opinion of the authors, this methodology offers a new perspective: instead of addressing physical asset integration as another part of AAS implementation, this methodology focuses exclusively on this problem, offering a generic approach that can be applied regardless the type of asset and the technology. For this reason, it is considered that this article can be complementary to other proposals that cover partial aspects of our methodology, such as [32] or [34].

As a proof of concept, this methodology has been used to integrate a robotic arm into an I4.0 Component. Through this example, it is illustrated how, in addition to the main asset, other assets can be part of the integration process (performing process control or automation, for example). It also shows how the available assets condition the deployment and implementation of the layers. Finally, it is also shown how to implement the layers following a bottom-up approach.

To sum up, it has been proven that the proposed approach and methodology together provide a tool that can facilitate the realization of the phases prior to I4.0 (computerization and connectivity), by addressing the integration between AASs and assets to attend service requests. In future article, the authors will work on interoperability between AASs, allowing them to interact on the basis of autonomous decision making for service supply and demand. In addition, they will also study how to adapt this approach and methodology to ease the integration of humans into the systems, in line with upcoming trends such as Industry 5.0 and Society 5.0 [36], [37].

References

[1] R. Dintén, P. L. Martínez, and M. Zorrilla, "Reference architecture for the design and development of applications for industry 4.0," *Revista Iberoamericana de Automática e Informática Ind.*, vol. 18, no. 3, Jul. 2021, Art. no. 300, doi: 10.4995/riai.2021.14532.

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

[3] D. Mourtzis and N. Panopoulos, "Digital transformation process towards resilient production systems and networks," in *Supply Network Dynamics and Control*, A. Dolgui, D. Ivanov, and B. Sokolov, Eds. Berlin, Germany: Springer, 2022, pp. 11–42, doi: 10.1007/978-3-031-09179-7_2.

[4] C. Duarte, P. Gaynor, M. Kande, C. Moldenhauer, S. Pillsbury, and R. Walker, "Fourth industrial revolution: A recovery plan for today's economic storm," 2020. Accessed: Feb. 02, 2022. [Online]. Available: https://www.pwc.com/us/en/library/assets/pwc-economic-slowdown.pdf

[5] S.-W. Lin et al., "Architecture alignment and interoperability," Dec. 2017. [Online]. Available: https://www.iiconsortium.org/iic-i40-joint-work.htm

[6] "Alignment report for reference architectural model for Industrie 4.0/Intelligent manufacturing system architecture," Apr. 2018. [Online]. Available: https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/hm-2018-manufacturing.html

[7] X. Ye and S. H. Hong, "Toward Industry 4.0 components: Insights into and implementation of asset administration shells," *IEEE Ind. Electron. Mag.*, vol. 13, no. 1, pp. 13–25, Mar. 2019, doi: 10.1109/MIE.2019.2893397.

[8] G. Schuh, R. Anderl, R. Dumitrescu, and A. Krüger, "Industrie 4.0 maturity index," Apr. 22, 2020. [Online]. Available: https://en.acatech.de/publication/industrie-4-0-maturity-index-update-2020/

[9] G. Schuh, R. Anderl, R. Dumitrescu, and A. Krüger, "Using the Industrie 4.0 maturity index in industry," Apr. 22, 2020. [Online]. Available: https://en.acatech.de/publication/using-the-industrie-4-0-maturity-index-in-industry-case-studies/

[10] S. Karnouskos and P. Leitão, "Key contributing factors to the acceptance of agents in industrial environments," *IEEE Trans. Ind. Inform.*, vol. 13, no. 2, pp. 696–703, Apr. 2017, doi: 10.1109/TII.2016.2607148.

[11] 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. Inform.*, vol. 13, no. 5, pp. 2417–2427, Oct. 2017, doi: 10.1109/TII.2017.2720696.

[12] 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, Jan. 2019, Art. no. 69, doi: 10.3390/s19010069.

[13] P. Leitao and F. Restivo, "ADACOR: A holonic architecture for agile and adaptive manufacturing control," *Comput. Ind.*, vol. 57, pp. 121–130, 2006, doi: 10.1016/j.compind.2005.05.005.

[14] D. Mourtzis, "Simulation in the design and operation of manufacturing systems: State of the art and new trends," *Int. J. Prod. Res.*, vol. 58, no. 7, pp. 1927–1949, Apr. 2020, doi: 10.1080/00207543.2019.1636321.

[15] 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 Ind. Electron. Mag.*, vol. 14, no. 3, pp. 18–32, Sep. 2020, doi: 10.1109/MIE.2019.2962225.

[16] "Interoperability — our vision for Industrie 4.0: Interoperable communication between machines within networked digital ecosystems," Nov. 2019. [Online]. Available: https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/position-paper-interoperability.pdf?__blob=publicationFile&v=4

[17] A. D. Neal et al., "The potential of Industry 4.0 cyber physical system to improve quality assurance: An automotive case study for wash monitoring of returnable transit items," *CIRP J. Manuf. Sci. Technol.*, vol. 32, pp. 461–475, Jan. 2021, doi: 10.1016/j.cirpj.2020.07.002.

[18] P. Marcon et al., "The asset administration shell of operator in the platform of Industry 4.0," in *Proc. 18th Int. Conf. Mechatron.*, 2018, pp. 1–5.

[19] F. Pethig, O. Niggemann, and A. Walter, "Towards Industrie 4.0 compliant configuration of condition monitoring services," in *Proc. IEEE 15th Int. Conf. Ind. Inform.*, 2017, pp. 271–276, doi: 10.1109/INDIN.2017.8104783.

[20] S. Cavalieri and M. G. Salafia, "Asset administration shell for PLC representation based on IEC 61131–3," *IEEE Access*, vol. 8, pp. 142606–142621, 2020, doi: 10.1109/ACCESS.2020.3013890.

[21] 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 Trans. Ind. Inform.*, vol. 13, no. 2, pp. 725–736, Apr. 2017, doi: 10.1109/TII.2016.2595401.

[22] C. Schou and O. Madsen, "A plug and produce framework for industrial collaborative robots," *Int. J. Adv. Robot. Syst.*, vol. 14, no. 4, Jul. 2017, Art. no. 1729881417717472, doi: 10.1177/1729881417717472.

[23] X. Ye, S. H. Hong, W. S. Song, Y. C. Kim, and X. Zhang, "An Industry 4.0 asset administration shell-enabled digital solution for robot-based manufacturing systems," *IEEE Access*, vol. 9, pp. 154448–154459, 2021, doi: 10.1109/ACCESS.2021.3128580.

[24] J. D. Contreras, J. I. Garcia, and J. D. Diaz, "Developing of industry 4.0 applications," *Int. J. Online Eng.*, vol. 13, no. 10, Nov. 2017, Art. no. 30, doi: 10.3991/ijoe.v13i10.7331.

[25] J. Fuchs, J. Schmidt, J. Franke, K. Rehman, M. Sauer, and S. Karnouskos, "I4.0-compliant integration of assets utilizing the asset administration shell," in *Proc. 24th IEEE Int. Conf. Emerg. Technol. Factory Automat.*, 2019, pp. 1243–1247, doi: 10.1109/ETFA.2019.8869255.

[26] *IEEE Recommended Practice for Industrial Agents: Integration of Software Agents and Low-Level Automation Functions*, IEEE Std 26601-2020, Jan. 2021, pp. 1–43, doi: 10.1109/IEEESTD.2021.9340089.

[27] T. Burns, J. Cosgrove, and F. Doyle, "A review of interoperability standards for Industry 4.0," *Procedia Manuf.*, vol. 38, pp. 646–653, 2019, doi: 10.1016/j.promfg.2020.01.083.

[28] L. Sakurada, P. Leitao, and F. D. la Prieta, "Agent-based asset administration shell approach for digitizing industrial assets," *IFAC Papercept*, vol. 55, no. 2, pp. 193–198, 2022, doi: 10.1016/j.ifacol.2022.04.192.

[29] P. Leitão and J. Barbosa, "Building a robotic cyber-physical production component," in *Service Orientation in Holonic and Multi-Agent Manufacturing*, vol. 640, T. Borangiu, D. Trentesaux, A. Thomas, and D. McFarlane, Eds. Berlin, Germany: Springer, 2016, pp. 295–305, doi: 10.1007/978-3-319-30337-6_27.

[30] K. Bakliwal, M. H. Dhada, A. S. Palau, A. K. Parlikad, and B. K. Lad, "A multi agent system architecture to implement collaborative learning for social industrial assets," *IFAC Papercept*, vol. 51, no. 11, pp. 1237–1242, 2018, doi: 10.1016/j.ifacol.2018.08.421.

[31] A. López, O. Casquero, E. Estévez, P. Leitão, and M. Marcos, "Towards the generic integration of agent-based AASs and physical assets: A four-layered architecture approach," in *Proc. IEEE 19th Int. Conf. Ind. Inform.*, 2021, pp. 1–6, doi: 10.1109/INDIN45523.2021.9557568.

[32] B. Ashtari Talkhestani et al., "An architecture of an intelligent digital twin in a cyber-physical production system," *Automatisierungstechnik*, vol. 67, no. 9, pp. 762–782, Sep. 2019, doi: 10.1515/auto-2019-0039.

[33] A. Seif, C. Toro, and H. Akhtar, "Implementing industry 4.0 asset administrative shells in mini factories," *Procedia Comput. Sci.*, vol. 159, pp. 495–504, 2019, doi: 10.1016/j.procs.2019.09.204.

[34] X. Ye, J. Jiang, C. Lee, N. Kim, M. Yu, and S. H. Hong, "Toward the plug-and-produce capability for Industry 4.0: An asset administration shell approach," *IEEE Ind. Electron. Mag.*, vol. 14, no. 4, pp. 146–157, Dec. 2020, doi: 10.1109/MIE.2020.3010492.

[35] T. Miny, G. Stephan, T. Usländer, and J. Vialkowitsch, "Functional view of the asset administration shell in an Industrie 4.0 system environment," Apr. 13, 2021. Accessed: Jan. 17, 2022. [Online]. Available: https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/Functional-View.html

[36] D. Mourtzis, J. Angelopoulos, and N. Panopoulos, “A literature review of the challenges and opportunities of the transition from Industry 4.0 to Society 5.0,” *Energies*, vol. 15, no. 17, Aug. 2022, Art. no. 6276, doi: 10.3390/en15176276.

[37] J. Leng et al., “Industry 5.0: Prospect and retrospect,” *J. Manuf. Syst.*, vol. 65, pp. 279–295, Oct. 2022, doi: 10.1016/j.jmsy.2022.09.017.

Alejandro López received the B.S. degree in industrial electronics and automation engineering from the University of the Basque Country (UPV/EHU), Leioa, Spain, in 2016, and the M.S. degree in control, automation and robotics engineering from the UPV/EHU, in 2019.

After almost 2 years as a Technician with the Automotive Smart Factory Competence Center in Advanced Manufacturing in AIC (Amorebieta), in November 2019, he started a Ph.D. program in Control, Automation and Robotics with the Faculty of Engineering in Bilbao. As a Researcher, he is a part of the Grupo de Control e Integración de Sistemas Research Group with the Department of Systems Engineering and Automatic Control, where he is committed to the study of smart and flexible automation production systems.

He has coauthored 2 indexed journal articles and 10 publications in national and international conference proceedings.

Elisabet Estévez received the B.S. degree in telecommunications engineering from the University of the Basque Country (UPV/EHU), Leioa, Spain, in 2002, and the Ph.D. degree in automatic control from the UPV/EHU, in 2007.

She has been a Full Professor with the Electronics and Automation Engineering Department, University of Jaén, Jaén, Spain, since 2019. She has authored and coauthored more than 150 technical papers in international journals and conference proceedings in the field of distributed industrial control systems. Her research interests include application of model-driven engineering to automation systems, the application of industrial agents in manufacturing, and digital twins.

Dr. Estévez has served and already serves as a Vice-Chair of the Technical Committee TC 3.1 Computers for Control (on social media) and Work in progress co-chair for IEEE ETFA 2010. She has been an Associate Editor for IEEE T-ASE journal, since 2021.

Oskar Casquero received the B.S. degree in telecommunication engineering from the University of the Basque Country (UPV/EHU), in 2003 and the Ph.D. degree in engineering from the UPV/EHU, in 2013.

From 2004 to 2007, he worked as an IT Architecture Analyst with the Virtual Campus, University of the Basque Country, Leioa, Spain. Since 2007, he has been working as an Assistant Professor with the Systems Engineering and Automatic Control Department, and currently with the Faculty of Engineering in Bilbao. He investigates on smart and flexible manufacturing systems using digital twins, model-driven engineering, multiagent systems, and cloud computing technologies.

Marga Marcos (Senior Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in control engineering from the Basque Country University, Leioa, Spain, in 1983, 1984, and 1988, respectively.

She is currently a Professor in control engineering with the Basque Country University, where she was a Vice-Dean with the Faculty of Engineering (1990–1993) and a Chairwoman with the Control Engineering Department (1995–2005). She has authored and coauthored more than 200 technical papers in international journals and conference proceedings. She has acted as the main researcher of more than 80 research projects funded by National and European R&D programs.

Dr. Marcos has served on several TCs of IFAC and IEEE. She was General Co-Chair for IEEE ETFA 2010. She has served as an Associate Editor for IEEE T-ASE journal (2018–2020). She is serving as a member of the IFAC Council for the 2020–2023 triennium. Her research interests include application of MDE and industrial agents in manufacturing.