Towards a Digital Twin Platform for Industrie 4.0

Magnus Redeker, Jan Nicolas Weskamp, Bastian Rössl, Florian Pethig
Fraunhofer IOSB, IOSB-INA Lemgo, Fraunhofer Institute of Optronics, System Technologies and Image Exploitation,
{magnus.redeker, jan.nicolas.weskamp, bastian.roessl, florian.pethig}@iosb-ina.fraunhofer.de

***Abstract*—In an Industrie 4.0 (I4.0), rigid structures and architectures applied in manufacturing and industrial information technologies today, should be replaced by highly dynamic and self-organizing networks. Today's proprietary technical systems lead to strictly defined engineering processes and value chains. Interacting Digital Twins (DT) are considered an enabling technology that could help to increase flexibility based on semantically enriched information. Nevertheless, for interacting digital twins to become a reality, their implementation should be based on existing standards like the Asset Administration Shell (AAS). Additionally, DT Platforms could accelerate development, deployment and ensure resilient operation of DT. This paper presents such a platform based on a microservices architecture and offering solutions for continuous deployment, data infrastructure and I4.0 business services. The platform is evaluated in the use case scenarios platform-based manufacturing and collaborative condition monitoring. As a result, implemented AAS-based microservices organize manufacturing, and submodels of the AAS enable cross-company data sharing for collaborative condition monitoring. Future work should focus on fault-management and service recovery, as well as integration of the AAS into lower platform layers, e.g. for improving data usage control.**

***Index Terms*—Industrie 4.0, Digital Twin, Asset Administration Shell, Semantic Interoperability, Big Data Platform, Microservices Architecture, Manufacturing, Collaborative Condition Monitoring, GAIA-X**

I. Introduction

In this day and age, increasing manufacturing productivity and sustainability faces challenges regarding interoperability and scalability of technical systems [1], [2]. The rigid Operational Technology (OT) and Information Technology (IT) architectures widely used today, lead to the establishment of equally inflexible value chains and prevent the implementation of highly dynamic and globally connected value networks that could enable new forms of collaboration and business models [3]. It is the vision for Industrie 4.0 (I4.0) to enable this implementation through the establishment of standards, that should be implemented based on highly scalable and sovereign data infrastructure as an environment of trust [4], [5]. Major standards evolving in the German initiative "Plattform Industrie 4.0" currently focus on the implementation of one main technological concept that should foster semantic interoperability and autonomy of technical systems: Digital Twins (DT) for I4.0 [6], [7]. Benefits expected from DT for I4.0 include efficient re-configuration of production lines and machines for mass customization in application scenarios like "Plug-and-Produce" or "Order-controlled Production" [8]–[11], as well as cross-company collaboration across value chains and life cycle stages in order to optimize products and production processes [12].

The Reference Architecture Model Industrie 4.0 (RAMI 4.0) includes the concept of an I4.0 component, consisting of Asset and Asset Administration Shell (AAS) [13]. The AAS concept "helps implement digital twins for I4.0 and create interoperability across the solutions of different suppliers" [14]. The AAS specifies a meta information model for I4.0 that is based on properties standardized according to IEC 61360 [8], [15]. Property specifications in dictionaries like the IEC Common Data Dictionary include formalized semantic descriptions and are an important step towards unambiguous and automatic interpretation of knowledge by I4.0 components [16]. The AAS meta information model is being standardized as IEC 63278 ED1 and first mappings of the meta model to specific technologies, e.g. to the OPC Unified Architecture (OPC UA), exist [17], [18]. In addition to the information model, first infrastructure components for AAS are being developed as well, e.g. a Registry for AAS and runtime environments based on different programming frameworks [19], [20]. AAS services executed in such an environment exchange information via a specified Application Programming Interface (API) or they interact autonomously via a specified language for I4.0 [21], [22]. Additional I4.0 software components and services will be developed within the recently founded Industrial Digital Twin Association (IDTA) [23].

In order to create the desired highly dynamic value networks, services should be "built around business capabilities and independently deployable by fully automated deployment machinery" [24]. Services should be loosely coupled in a distributed architecture and developed by small teams to accelerate testing and enable continuous deployment. This emerging architectural style is called a microservices architecture [25]. At the moment, software components are rapidly being developed in parallel I4.0 projects. For this reason, it seems promising to combine them on a joint Platform for DT in I4.0 according to established microservices patterns.

This paper focuses on the implementation of the two I4.0 Use Cases "Platform based Manufacturing" and "Collaborative Condition Monitoring" based on a DT Platform for I4.0. The paper is organized as follows: Section II introduces development and operation (DevOps) of a DT Platform based on a microservices architecture. Section III and Section IV introduce Use Cases that have been implemented based on the introduced Platform. Section V concludes this paper, identifies challenges and gives an outlook on future work like the evaluation of different technologies for data sovereignty and usage control.

2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS) | 978-1-7281-6207-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICPS49255.2021.9468204

978-1-7281-6207-2/21/$31.00 ©2021 IEEE

II. Platform Architecture

The DT Platform for I4.0 presented in this section facilitates efficient development, deployment and operation of I4.0 business services. To account for nonfunctional requirements like scalability, robustness, performance, maintainability and security, I4.0 business services are implemented based on a set of data infrastructure microservices, e.g. for discovery, connectivity, synchronicity, and storage. A continuous development and integration pipeline facilitates the release of robust microservices that are deployed to and orchestrated by a Kubernetes cluster as backbone of the DT Platform for I4.0 [26]. Main features of the cluster include load balancing and fault management, e.g. by scaling and restarting service instances automatically. Moreover, the operational phase of the platform is supported by monitoring and logdata collection services. Here, fluentd collects logdata and stores it in an elasticsearch database [27], [28]. Please find more architecture specific orchestration details in [29] and [30].

Additionally, microservices could be restricted to perform only a limited number of tasks simultaneously. If its limit is reached, a microservice would have to delete its service descriptor from the submodel-registry, so that it is not discoverable, and, vice versa, could add the descriptor again when convenient. Such a restriction mechanism would assist in preventing monopoly situations and maintaining the quality standards of the platform.

Figure 1 depicts the architecture of the DT Platform for I4.0. It is composed of the following layers:

1) Business services layer: As value-added services, I4.0 services are located on the business services layer of the platform, utilizing the data infrastructure microservices on the layer below. I4.0 services are accessible and interact via an API.
2) Data infrastructure layer: The data infrastructure layer contains the microservices supporting the business services layer. They are accessible via an API and some of them are interacting. These services are implemented as generic as possible.
3) Continuous deployment layer: The continuous deployment (CD) layer holds the base software for a CD of microservices into the data infrastructure and business services layers.

A. The Continuous Deployment Layer

The CD layer contains a pipeline based on Gitlab, a Container-Registry and Kubernetes [31], [32]. Through this pipeline new versions of microservices are developed, tested and deployed to either the data infrastructure or business services layers of the platform. For each microservice a separate Gitlab project exists. Triggered by changes in these Gitlab projects, the CD pipeline performs the following tasks [31]:

- Build: Following an update to a project's master branch, a container image is built. Since updates to the master branch trigger the CD pipeline, project owners should decide when a new release is due, merge content from other branches and update the master accordingly. In particular, the master branch contains the corresponding container configuration file providing all necessary settings. Each new image is tagged with a version number and registered in the Gitlab container registry.
- Deployment: As soon as an updated container image is registered in the registry, the corresponding containerized microservice is updated, too. The deployment service shuts down the existing container and restarts it with the same options that were used when it was deployed initially.

B. The Data Infrastructure Layer

The data infrastructure layer contains the following microservices, that support the I4.0 business services. The Knowledge Base is a document based database storing all relevant metadata of known AAS- and OPC UA-servers as JSON in a predefined mapping. Here, elasticsearch is used as database [27]. Furthermore, the knowledge base contains semantic descriptions of the data contained in the data lake. A free text search can be used to find information relevant for the use case at hand.

Regarding data from AAS, in particular submodel templates, as the usual semantic description of submodel instances, as well as those descriptions of semanticIDs of submodel elements that had to be defined manually since no public repository contained proper descriptions, are included in the knowledge base. Parts of this database are accessible from the internet, so that the authorized partners of the manufacturer can find for example semantic descriptions for the submodels and elements contained in the AASs the manufacturer shares with them.

To ensure that the available data is always in sync, the OPC UA discovery service automatically discovers each available OPC UA-server in the Edge-Cloud ecosystem. Some assets hosting a local OPC UA-server, e.g. on a Programmable Logic Controller (PLC), posses additional AAS in AAS-servers in the cloud that provide corresponding meta data. In such a case, the knowledge synchronizer service is responsible for linking and collecting AAS and OPC UA-servers or rather meta data and data sources and storing them in the knowledge base.

AAS servers like [19], [20] with an http/REST- or an OPC UA-CRUD (create, read, update, delete) API host AASs in the manufacturer's intranet. AAS- and submodel registries contain descriptors of AASs and submodels [21]. These descriptors include at least identification and endpoint of either AAS or submodel.

On request, the data flow orchestrator establishes or cuts off data streams from OPC UA- and AAS-servers into the data lake, to the business microservices, or to the cross-company connectors. The data lake contains databases for any kind of (raw) data: structured like rows and columns in a relational database, semi-structured like JSON or unstructured like files. Due to the knowledge base, the data can be searched quickly in the lake. Via the connectors to cross-company data-sharing systems the manufacturer can exchange and collaboratively

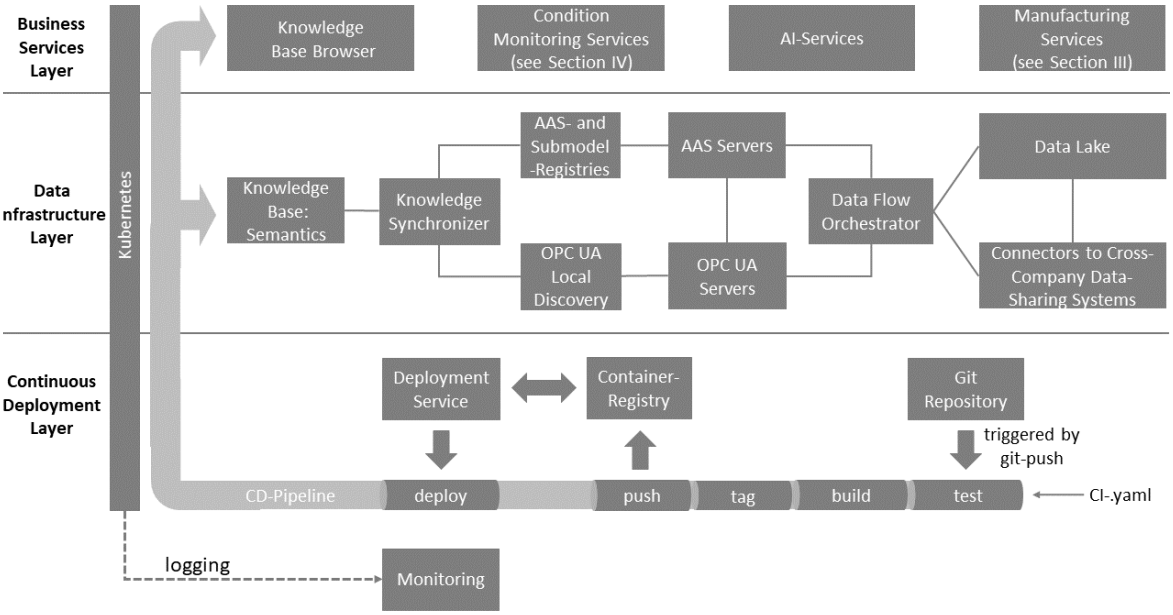

Fig. 1: The DT Platform for I4.0 consisting of a continuous deployment, a data infrastructure and a business services layer. Please find details of the interoperation between business and data infrastructure microservices in Sections III and IV.

enrich data with e.g. its collaborators in value-adding chains. Please find use case-specific details in Section III and Section IV.

An MQTT-broker is used for the execution of the bidding procedure [22], [33]. This procedure standardizes the interactions between I4.0 components and brings together providers and requesters of manufacturing services, e.g. for drilling. Since multiple MQTT-brokers operated by collaborating companies could be used, in order to interact across company borders, it is not explicitly contained in the data infrastructure layer in Figure 1.

C. The Business Services Layer

The business services layer finally contains all I4.0 business microservices. For example, the knowledge base browser enables a user to search – freely or using tags – for relevant data and its sources in the platform. The browser provides suitable results and optionally sets up a data pipeline from a selected source into the data lake. Please find a detailed description in [29].

The manufacturing services as well as the condition monitoring service are presented in detail in the Sections III and IV. These following sections also provide a deeper insight into the interoperation of the microservices in the business services and data infrastructure layers.

III. Platform-based Manufacturing

The DT Platform for I4.0 described in Section II facilitates the implementation of main application scenarios for I4.0: Mass customization and the resulting need for increased autonomy of manufacturing execution services. Using standardized I4.0 services enables cross-company collaboration of partners in a value chain consisting of product developers, manufacturers, vendors, as well as their customers. For example, vendors can request manufacturers' services to manufacture product instances of certain types, e.g. to cope with high order volume or to handle product orders that require specific manufacturing services or machinery.

Additionally, the DT Platform for I4.0 facilitates product and process optimization by utilizing a peer-to-peer sharing system to collaboratively enrich AASs of product types and instances with data from specific life cycle stages, e.g. product type development, customization of product instances, documentation of manufacturing processes or product usage data. This system combines a blockchain for booking who changed an AAS and when, a peer-to-peer distributed file system for encrypting and storing each version of an AAS, and a version control system for tracking changes to an AAS in detail. In combination, these technologies ensure the sovereignty of the collaborators' data, i.e. integrity, confidentiality and availability. Please find details in [12].

A. Manufacturing Execution based on I4.0 Standards

In this section, a concrete use case is considered in which manufacturing execution relies on platform-based microservices and existing standards for I4.0 like the AAS [34]. Manufacturing execution, i.e. the manufacturing of a product instance, is based on interacting business microservices that accept an order and afterwards schedule, execute and document the manufacturing process. A manufacturer's factory houses working stations and warehouses each with specific capabilities. These physical assets process their individual work orders successively according to the order in a queue. They, as well as the company's business microservices, all

comprise an AAS as standardized interface for I4.0 via which they can interact, offer their capabilities and trigger each others services. AASs are either implemented based on OPC UA or AAS-Server technologies (see Section II), [18]–[20]. Based on the following business microservices, manufacturing can be executed:

- Order taking service (OTS): taking orders of product instances from authorized vendors; starting point of each manufacturing process run.
- Manufacturing driving service (MDS): driving forward the manufacturing of product instances.
- Bidding request service (BRS): requesting and accepting services from the physical assets in the manufacturing process executing the requester side in the bidding procedure, [22], [33].
- Recording and notification service (RNS): recording the current status of product instances in the peer-to-peer sharing system and transmitting notifications to the corresponding vendors.

In contrast to the assets of the physical world, the assets of the information world can execute multiple instances of their services in parallel. Also, these I4.0 business microservices each serve their corresponding AAS on their own enabling authorized consumers http/REST-read access and the triggering of their specific service. For example, the order taking service serves its AAS via the internet. It can be accessed by collaborating vendors in order to trigger the manufacturing of product instances.

Each of the business services' AASs contains a Submodel describing their administered asset's specific service. It contains a SubmodelElement of type Operation, [34], which in turn contains SubmodelElements of type OperationVariables that define the necessary input parameters and the service's output.

In the sense of I4.0, product types and produced product instances are themselves considered as assets as well. Here, their corresponding AASs are provided via AAS-servers with CRUD APIs. These AASs provide submodels containing data necessary for the manufacturing of the product instances and submodels for the documentation. In addition to the collaboration use case described before, an instance's AAS facilitates product optimization based on manufacturing data and enables vendors and customers to track manufacturing progresses.

What the business services of the information world have in common with the working stations and warehouses of the physical world is that they only consider the data contained in their own AAS, as well as the data contained in the AASs of the product instance and its type they are currently working on. This distributed knowledge enables fast decision making and flexible manufacturing management.

B. Interaction of Business Microservices

How do the business microservices discover and trigger each other? This key question is answered by the use of the registries for AASs and submodels with CRUD APIs, [21], in which each microservice with an AAS can register a descriptor of, in particular, the submodel describing its specific service.

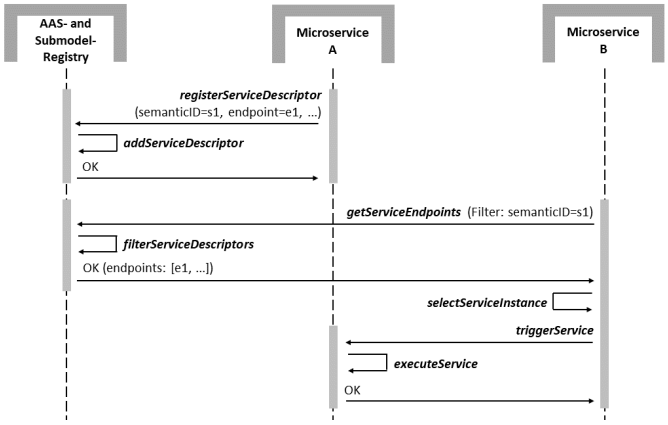

Fig. 2: The mutual discovery and triggering of business microservices. Microservice A adds a service descriptor, containing the service's semanticID and its trigger-endpoint, to a registry. B filters the registry for this semanticID and triggers A's endpoint.

Such a descriptor contains the identification and as key attributes the semanticID and the endpoint of the submodel. Their specific service can then be discovered by filtering the registry according to the service's semanticID and subsequently be triggered by calling the operations' endpoint with the necessary input parameters attached. Figure 2 illustrates the interaction of microservices and registries.

Schemata and descriptions of the input and output parameters of a microservice are described in the knowledge base under the operation's semanticID. To avoid false calls, descriptors in the registries must be confirmed every ten minutes and will be removed otherwise.

C. The Initialization of the Business Microservices

Kubernetes deploys the containerized business microservices OTS (order taking), MDS (manufacturing driving), BRS (bidding request) and RNS (recording and notification) and sets for each of them the initial configuration. What they all require is a free port for serving their AASs including their trigger. Also, they all need to know the endpoints of the AAS- and submodel-registries in order to, first, add their service-descriptors; second, to filter endpoints of product types' and instances' AASs and submodels from identifications they receive when triggered; and third, to filter endpoints of other business microservices.

Additionally, OTS and RNS require an endpoint of the peer-to-peer sharing system in order to either receive or record AASs. What OTS furthermore needs is an endpoint of an AAS server with a CRUD API for making AASs of product types and instances accessible and editable. Finally, for executing the bidding procedure, BRS requires the endpoints of the MQTT brokers, that the assets of the physical world use for offering their services.

Those microservices that trigger other I4.0-services must know their semanticIDs. These, however, are hard-coded, since input and output of the services must already be taken into account during implementation.

D. The Execution of Manufacturing Processes

Authorized vendors can trigger the execution of a manufacturing process run. For each product instance to be manufactured the vendors create an AAS containing the product type's identification, customization details as well as an endpoint via which they want to be notified of manufacturing progresses. They share their AASs with the manufacturer via the peer-to-peer sharing system.

When triggered, OTS – the starting point of each manufacturing process run – puts the vendor's AAS of a product instance into an AAS-server, extends it with submodels providing information required for the execution and submodels for the documentation of the manufacturing process and adds descriptors of AAS and submodels to the registries. OTS completes with triggering RNS and MDS providing each with the identification of the AAS and, in addition, RNS with "Manufacturing accepted".

RNS, when triggered, records the respective AAS's current state in the peer-to-peer sharing system, notifies the corresponding vendor transmitting the status message and terminates.

MDS drives the manufacturing process of a product instance. It is triggered either by OTS for the first manufacturing step or by a working station or warehouse that completed a step. It determines, from the process documentation in a product instance's AAS, the step to be executed next, triggers BRS providing as parameter the identification of the submodel describing this step and terminates.

BRS publishes a call for proposal, using the interaction element from the specified submodel, [22], [33], in the designated MQTT-brokers. It selects the received proposal suiting the manufacturer's guidelines – top priority instances: as soon as possible; lowest priority instances: as inexpensive as possible – best, confirms it to the proposer, documents it in the respective submodel, triggers RNS with the message "Manufacturing step <step-number> scheduled" and terminates.

The winning working station or warehouse of a bidding procedure is responsible for the completion of the respective manufacturing step. Its responsibility also includes the picking up of the product instance and the documentation of the step in the corresponding submodel. When completed, it triggers RNS with "Manufacturing step <step-number> completed" and, again, MDS.

When MDS determines that the manufacturing of a product instance is completed, it adds to the corresponding AAS a submodel describing the instance's nameplate, [35], triggers RNS with "Manufacturing completed" and terminates – the endpoint of each manufacturing process run.

IV. Platform-based Collaborative Condition Monitoring

A networked data infrastructure forms the basis for new and innovative business models and implies changes regarding the cooperation between all participants of a value chain. In contrast to Section III, in this section the manufacturer under consideration is the final element in a value chain consisting of component suppliers, integrators and finally the manufacturer, who operates the integrators' machines.

The use case collaborative condition monitoring (CCM), drafted originally by the German initiative *Plattform Industrie 4.0*, describes a shift from a bilateral data exchange to a multilateral relationship. The leading idea is that by sharing and using data in a collaborative way, insights can be gained and thus processes and products can be optimized as described in Section III, [36].

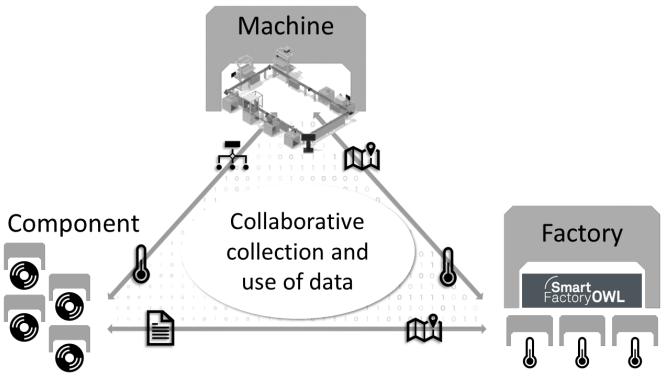

Fig. 3: The three point fractal sketches the minimum of a multilateral data exchange.

In Figure 3 the three stakeholders of this use case are depicted. The collaborative use of data between the component supplier and the factory operator is particularly noteworthy since that business relationship is not as close as the others.

This section presents an implementation approach for sharing temperature data, where the manufacturer uses the DT Platform for I4.0 developed in Section II, and points out the major challenges the collaborators must face.

A. Collaborative Condition Monitoring using Temperature Data

A component supplier is generally unaware of the environments in which his products are actually used, since this knowledge depends on the cooperation of integrator and machine operator. The partners can close this gap by providing data access – not only to the component itself, but also to the higher-level machine and factory or even to other integrated components.

In an ongoing project, the manufacturer provides the collaborative exchange of temperature data. Various built-in sensors are capable to measure environmental or operating temperature meaning that data is already generated by the equipment parts running in production. At factory level additional thermometers are installed to track the environmental temperature.

Drives inside a machine monitor their internal temperature and technical data sheets describe their operating window. However, a meaningful combination of these information is only possible when the measuring points are brought into spacial correlation. Therefore (relative) localization data of machines, components and sensors is documented and shared as well.

Please note, that components or machines without internal temperature sensors are not automatically excluded from the use case. A promising retrofit approach is to interpolate the element's temperature from temperature sensors nearby and thus enable condition monitoring.

B. Challenges

The collaborators must face the following major challenges to successfully implement the use case.

1) Interoperability: For maximum interoperability, live process data like temperature data can be accessed via OPC UA, where submodels in the assets' AASs provide semantic descriptions. At first sight, the use of the AAS companion specification [18] for OPC UA seems obvious. But at a closer look, the integrative nature dictates a few constraints. The AAS of a thermometer or an integrated device is likely to be co-managed in a host environment. The integrator can provide such as an OPC UA server running on a PLC or Human Machine Interface (HMI), but it is not given.

During the engineering phase the physical assets need to be plugged together as well as their AASs. In the end, each AAS, co-managed or self-managed, must be registered in the platform's registries for later discovery. To streamline that process – to create an integrated, co-managed component running within the platform – dedicated submodels with a common understanding across all involved partners in the chain must be developed.

2) Self-Describing and Managed Data Flow: Due to the nature of data collection, CCM mainly occurs during the usage and maintenance life cycle phase of an asset. Nevertheless, some important preparations are needed in earlier phases to enable a quick and easy integration. In the end, it should be clear which data can be used collaboratively and which collaborations are actually accepted by the partners.

The component supplier knows about type specific data points relevant for condition monitoring and might show interest in environmental data captured outside the instance. This information is stored in a dedicated submodel. Furthermore, an AAS specialized for collaborative data collection is deployed in the cloud. The integrator performs same actions for the machine.

A cross-company connector on the data infrastructure layer (see Figure 1) queries and evaluates such submodels to preconfigure the connections to cloud-based AASs. The operator must decide which data flows to allow and which not.

In contrast to Section III, the shared data is not the AAS in a serialized form, but rather a data synchronization mechanism between the partners' AASs administering the same asset.

3) Authorization and Access Control: It is assumed that the operator will not accept access to its own IT resources and will prefer to forward the data to a passive instance. With reference to a classic client-server architecture, the data connector acts as a client. As a consequence, the cloud-based AAS can be accessed across applications and, above all, across companies, so the question of access control arises.

For the AAS, the Security Part is explicitly intended for this purpose. As stated in [34], the configuration modeling of an attribute-based access control (ABAC) shall be an integral part of the AAS. In [37] ABAC has been stated as a more suitable solution than a simpler (e.g. role-based) approach. For now, an access control specification is pending and suitable concepts should be evaluated.

4) Authentication and Usage Control: At runtime a cross-asset data consumption for condition monitoring within a protected space seems fairly manageable. As soon as collaboratively shared information leave that space and cross-company data access is deployed, the demand for some kind of "terms and conditions" for data usage is rising. E.g. International Data Spaces Association (IDSA) deals with these questions and also provides a solution for the upstream authentication process [38]. The IDS reference architecture could also become a major enabler for the European cloud initiative GAIA-X [5].

5) Time Series Data: Any condition monitoring application based on machine learning algorithms relies on a large volume of high-quality historical data. To a certain degree it will be feasible to put time series within the AAS model but there is doubt that Big Data is manageable in such a structure. For that reason, a Internationalized or Uniform Resource Identifier (IRI/URI) establishes a semantic connection between an AAS and a database, placed in the data lake as mentioned in Section II or in the cloud, depending on the deployment model. If technically required, native database queries can be executed to bypass the AAS interface restrictions.

In summary, the presented major challenges emphasize the complexity of the use case. Interoperability is key for the engineering phase as well as for the operation phase. The toolbox of the AAS meta model provides solution approaches but the security aspect requires a more intensive examination.

V. Conclusion and Outlook

This paper presented a Digital Twin (DT) Platform for Industrie 4.0 (I4.0) facilitating efficient development, deployment and operation of I4.0 business microservices, each of which comprises a software asset and an Asset Administration Shell (AAS, standardized DT for I4.0).

The platform consists of three layers: i) Continuous deployment, ii) data infrastructure and iii) business services layer. The services in the continuous deployment layer, the backbone of the platform, ensure that the services in the data infrastructure and business services layer are deployed continuously and highly available. Data infrastructure services introduced in this paper, integrate and manage data from the shop-floor in order to provide semantically enriched information to I4.0 business

services that use these information to perform their specific services.

In a manufacturer's actual application of the DT Platform for I4.0, four business microservices accept manufacturing orders from trusted vendors, control manufacturing processes, commission shop-floor assets and record the manufacturing of product instances. Each I4.0 business microservice serves its AAS on its own, enabling read access for authorized consumers via HTTP. They discover and trigger one another by connecting to a proper endpoint found in a registry and calling specific services with the necessary input parameters attached.

Originally, the controlling of a product's manufacturing was to become a service of the product's AAS. For the execution of the manufacturing process the manufacturer makes these AASs accessible and editable by putting them to an AAS server with a CRUD API (create, read, update and delete). Using one AAS server per product instance, as for the business microservices, seems unfeasible in terms of scalability. For the incorporation of a manufacturing controlling service, however, the products' AASs would have to be enabled to execute operations, which in turn would have to be implemented in the AAS server. Since no exclusive relationship between server and product instance's AAS can exist, the controlling would ultimately be a sole service of the server, though, with one endpoint per AAS. Therefore, it proved significantly more practical to create a separate business microservice with only one endpoint for triggering.

For each manufactured product instance, needed steps are each assigned to a capable working station by a dedicated business microservice. After execution and documentation of an assigned step, the working station in charge hands back the responsibility to the business microservices. Step-by-step these services take decisions, where only data from their own AAS and from the AASs of the product instance, they are currently working on, and its product type are taken into account. This fast and decentralized decision-making process effects that the overall manufacturing system is highly adaptable to e.g. customer requests and schedule deviations.

The manufacturer, in its role as a machine operator, shares machine operation data in a collaborative way with the respective machine integrator and the component suppliers. Data connectors in the data infrastructure layer of the DT Platform for I4.0 use the AAS registry as a starting point to query submodels dedicated for cross-company collaboration. They enable an automatic configuration to cloud connected data streams and ensure interoperability. Acquired data will either reside on site or in the cloud, as desired by the manufacturer. Their networked data infrastructure forms the basis for new and innovative business models like collaborative condition monitoring for gaining insights and thus optimizing processes and products of the manufacturer as well as of the component suppliers and machine integrators.

The presented platform comprises some similarities to the Cloud manufacturing platform (CMfg) as both represent manufacturing resources and tasks by digital representations in platforms in order to organize manufacturing [39], [40]. Whereas a central service allocates and schedules in CMfg, this paper's platform, in the spirit of Industrie 4.0 and based on the conviction that the determination of global optima is too complex to cope with rapidly changing manufacturing environments, enables resources, tasks and their digital twins to autonomously build highly-dynamic networks and flexibly negotiate the scheduling among themselves.

Future research and development of the DT Platform for I4.0 focuses on evaluating and optimizing the effectiveness and performance of the platform.

Furthermore it seems appropriate to encapsulate the data infrastructure microservices – like the business microservices – each with an AAS composing I4.0-components. A promising retrofit approach for such a microservice is to deploy an additional application functioning as a proxy that serves an AAS describing the corresponding microservice. An endpoint in a submodel could act as a trigger, where input and output of the microservice are translated to SubmodelElements of type OperationVariables. AAS- and submodel-registries would be equipped with proxy-AASs as well, enabling descriptor registrations in other AAS- and submodel-registries. The development of a bootstrapping mechanism for registries would then be a logical consequence.

These targeted enhancements would decentralize the DT Platform for I4.0 even further and increase its adaptiveness significantly, since then deployed microservices would only require a free port for serving their own AAS and one endpoint of a submodel-registry via which all other data infrastructure and business microservices are discoverable.

What the manufacturer also needs to develop, is a recovery mechanism for process states in crashed business microservices. Taking into account the distributed knowledge in the decentralized platform, how can a product instance's manufacturing be resumed loss-free in the right process step?

For collaborative condition monitoring, missing or incomplete AAS security aspects need to be checked in more depth. Either the high complexity or reasoning errors in the security meta model could be the cause for missing examples, implementations and non-standardized submodels. As soon as these open points become more concrete, higher-level questions related to data sovereignty can rely on the implementation of an underlying access control. Future work will also include a complete integration into the GAIA-X architecture and a deeper investigation regarding data usage control technologies.

To conclude this paper, the authors would like to encourage the I4.0 community to more precisely refer to active AASs as active composite I4.0-components. The AAS is an I4.0 information model, accessible via file or API. The capability to actively take decisions and interoperate with other components in an I4.0 network is added by executable functions, that are specifiable by the operation meta model and invocable via the AAS-API. Therefore, the authors think of active composite I4.0-components composed of (self-managed) original assets and (self- or co-managed) software assets contributing the I4.0 activity capability.

ACKNOWLEDGMENT

The research and development projects "Technical Infrastructure for Digital Twins" (TeDZ) and "Industrial Automation Platform" (IAP) are funded by the Ministry of Economic Affairs, Innovation, Digitalisation and Energy (MWIDE) of the State of North Rhine-Westphalia within the Leading-Edge Cluster "Intelligent Technical Systems OstWestfalenLippe (it's OWL)" and managed by the Project Management Agency Jülich (PTJ). The research and development project "KI-Reallabor für die Automation und Produktion" initiated by the German initiative "Platform Industrie 4.0" is funded by the German Federal Ministry for Economic Affairs and Energy (BMWi) and managed by the VDI Technologiezentrum (VDI TZ). The authors are responsible for the content of this publication.

REFERENCES

[1] H. Panetto, B. Iung, D. Ivanov, G. Weichhart, and X. Wang, "Challenges for the cyber-physical manufacturing enterprises of the future," *Annual Reviews in Control*, vol. 47, pp. 200 – 213, 2019.

[2] K. Al-Gumaei, A. Müller, J. N. Weskamp, C. S. Longo, F. Pethig, and S. Windmann, "Scalable analytics platform for machine learning in smart production systems," in *24th IEEE ETFA*, 2019, pp. 1155–1162. [Online]. Available: https://dx.doi.org/10.1109/ETFA.2019.8869075

[3] Asset Administration Shell Reading Guide (11/2020). [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/Asset_Administration_Shell_Reading_Guide.html

[4] 2030 Vision for Industrie 4.0 - Shaping Digital Ecosystems Globally. [Online]. Available: https://www.plattform-i40.de/PI40/Navigation/EN/Industrie40/Vision/vision.html

[5] GAIA-X - A Federated Data Infrastructure for Europe. [Online]. Available: https://www.data-infrastructure.eu

[6] Bitkom e.V. and Fraunhofer-Institut für Arbeitswirtschaft und Organisation IAO, "Industrie 4.0 – Volkswirtschaftliches Potenzial für Deutschland," *Studie*, 2014.

[7] A. Deuter and F. Pethig, "The Digital Twin Theory - Eine neue Sicht auf ein Modewort," *Industrie 4.0 Management 35*, 2019.

[8] F. Pethig, O. Niggemann, and A. Walter, "Towards Industrie 4.0 compliant configuration of condition monitoring services," in *15th IEEE INDIN*, 2017, pp. 271–276.

[9] D. Lang, M. Friesen, M. Ehrlich, L. Wisniewski, and J. Jasperneite, "Pursuing the Vision of Industrie 4.0: Secure Plug-and-Produce by Means of the Asset Administration Shell and Blockchain Technology," in *16th IEEE INDIN*, 2018, pp. 1092–1097.

[10] S. Heymann, L. Stojanovci, K. Watson, S. Nam, B. Song, H. Gschossmann, S. Schriegel, and J. Jasperneite, "Cloud-based Plug and Work architecture of the IIC Testbed Smart Factory Web," in *23rd IEEE ETFA*, 2018, pp. 187–194.

[11] J. Jasperneite, S. Hinrichsen, and O. Niggemann, "Plug-and-Produce für Fertigungssysteme," *Informatik-Spektrum*, vol. 38, no. 3, pp. 183–190, 2015.

[12] M. Redeker, S. Volgmann, F. Pethig, and J. Kalhoff, "Towards Data Sovereignty of Asset Administration Shells across Value Added Chains," in *25th IEEE ETFA*, 2020, pp. 1151–1154. [Online]. Available: https://dx.doi.org/10.1109/ETFA46521.2020.9211955

[13] *DIN SPEC 91345: Reference Architecture Model Industrie 4.0 (RAMI4.0)*, DIN Std. DIN SPEC 91 345, 04 2016. [Online]. Available: https://dx.doi.org/10.31030/2436156

[14] The Asset Administration Shell: Implementing Digital Twins for use in Industrie 4.0 - A starter kit for developers. [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/VWSiD V2.0.html

[15] *IEC 61360 -1 to 4- Standard data element types with associated classification scheme for electric components*, IEC Std., 2002.

[16] International Electrotechnical Commission (IEC). (2016) IEC 61360 - Common Data Dictionary (CDD - V2.0014.0014). [Online]. Available: http://cdd.iec.ch/cdd/iec61360/iec61360.nsf

[17] IEC 63278-1 ED1 Asset administration shell for industrial applications – Part 1: Administration shell structure. [Online]. Available: https://www.iec.ch/dyn/www/f?p=103:38:25763300038578::::FSP_ORG_ID,FSP_APEX_PAGE,FSP_PROJECT_ID:1250,23,103536

[18] I4AAS - An OPC UA Information Model for the Industrie 4.0 Asset Administration Shell. [Online]. Available: https://opcfoundation.org/markets-collaboration/i4aas/

[19] AASX Server. [Online]. Available: https://github.com/admin-shell-io/aasx-server

[20] Eclipse BaSyx. [Online]. Available: https://wiki.eclipse.org/BaSyx

[21] Details of the Asset Administration Shell, Part 2, Version: 1.0. [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/Details_of_the_Asset_Administration_Shell_Part_2_V1.html

[22] VDI/VDE 2193 Blatt 1:2020-04: Language for I4.0 Components - Structure of messages. [Online]. Available: https://www.beuth.de/de/technische-regel/vdi-vde-2193-blatt-1/318387425

[23] User organization "Industrial Digital Twin Association" founded. [Online]. Available: https://www.vdma.org/en/v2viewer/-/v2article/render/52443301

[24] Microservices - a definition of this new architectural term. [Online]. Available: https://martinfowler.com/articles/microservices.html

[25] C. Richardson, *Microservices Patterns: With examples in Java*. Manning Publications, 2018. [Online]. Available: https://books.google.de/books?id=UeK1swEACAAJ

[26] Kubernetes. [Online]. Available: https://kubernetes.io

[27] Elasticsearch. [Online]. Available: https://www.elastic.co/elasticsearch

[28] Fluentd. [Online]. Available: https://www.fluentd.org

[29] J. Weskamp, A. Chowdhury, F. Pethig, and L. Wisniewski, "Architecture for Knowledge Exploration of Industrial Data for Integration into Digital Services," in *3rd IEEE ICPS*, 2020. [Online]. Available: https://dx.doi.org/10.1109/ICPS48405.2020.9274700

[30] A. Ghosh Chowdhury, M. Illian, L. Wisniewski, and J. Jasperneite, "An Approach for Data Pipeline with Distributed Query Engine for Industrial Applications," in *25th IEEE ETFA*, 2020.

[31] GitLab. [Online]. Available: https://about.gitlab.com

[32] L. Chen, "Continuous delivery: Huge benefits, but challenges too," *IEEE Software*, vol. 32, no. 2, pp. 50–54, 2015.

[33] VDI/VDE 2193 Blatt 2:2020-01: Language for I4.0 components - Interaction protocol for bidding procedures. [Online]. Available: https://www.beuth.de/de/technische-regel/vdi-vde-2193-blatt-2/314114399

[34] Details of the Asset Administration Shell, Part 1, Version: 3.0. [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/Details_of_the_Asset_Administration_Shell_Part1_V3.html

[35] Submodel Templates of the Asset Administration Shell - ZVEI Digital Nameplate for industrial equipment (Version 1.0). [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/Submodel_Templates-Asset_Administration_Shell-digital_nameplate.html

[36] Collaborative data-driven business models: Collaborative Condition Monitoring – How cross-company collaboration can generate added value. [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/collaborative-data-driven-business-models.html

[37] Access control for Industrie 4.0 components for application by manufacturers, operators and integrators. [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/Access control for Industrie 4.0 components.html

[38] IDS Reference Architecture Model. [Online]. Available: https://www.internationaldataspaces.org/wp-content/uploads/2019/03/IDS-Reference-Architecture-Model-3.0.pdf

[39] L. Zhang, Y. Luo, F. Tao, B. H. Li, L. Ren, X. Zhang, H. Guo, Y. Cheng, A. Hu, and Y. Liu, "Cloud manufacturing: a new manufacturing paradigm," *Enterprise Information Systems*, vol. 8, no. 2, pp. 167–187, 2014.

[40] F. Tao, L. Zhang, V. C. Venkatesh, Y. Luo, and Y. Cheng, "Cloud manufacturing: a computing and service-oriented manufacturing model," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 225, no. 10, pp. 1969–1976, 2011.