I4.0-compliant integration of assets utilizing the Asset Administration Shell

Jonathan Fuchs
Institute for Factory Automation and Production Systems
Erlangen, Germany
jonathan.fuchs@faps.fau.de

Jan Schmidt
Institute for Factory Automation and Production Systems
Erlangen, Germany
jan.schmidt@fau.de

Jörg Franke
Institute for Factory Automation and Production Systems
Erlangen, Germany
joerg.franke@faps.fau.de

Kasim Rehman
SAP
Walldorf, Germany
kasim.rehman@sap.com

Manuel Sauer
SAP
Walldorf, Germany
manuel.sauer02@sap.com

Stamatis Karnouskos
SAP
Walldorf, Germany
stamatis.karnouskos@sap.com

***Abstract*— Global trends such as mass customization and lot size one, demand flexibility, autonomy and adaptability in production. Interoperability of devices is a key challenge as production systems evolve into Cyber-Physical Production Systems (CPPS). As part of the German strategic project Industrie 4.0 (I4.0), concepts and solutions for continuous digitization of production are being developed, in order to meet these numerous challenges. The concept of the Asset Administration Shell (AAS) was introduced in order to provide data and information in a standardized and semantically described manner, thus enabling interoperability and easy interaction. In this paper, we show how users can translate a semantic description of plants, machines or individual components that are based on a standardized Open Platform Communication Unified Architecture (OPC UA) information model, into the AAS information model.**

Keywords— Asset Administration Shell, OPC UA, Information Model, Plug & Produce, Semantic Interaction

I. INTRODUCTION

The significant progress made in the field of information and communication technologies (ICT) in recent years in terms of performance and energy efficiency at an even smaller IT-component size has made it possible to equip automation components with dedicated computation and communication capabilities, even at field level. As a result, the integration of software artefacts in several industrial components has given rise to the CPPS. Although the integration of computers has already led to a very high degree of automation in various processes since the 1970s, ubiquitous communication without a supervisory control unit between all components of automation networks is not yet possible. Communication capability is one of the most important factors in the context of flexible, adaptive and therefore efficient production. Consistent vertical and horizontal integration and semantic descriptions are preconditions to use context-related data in different application scenarios globally. This ranges from user-oriented software tools for asset management to complex applications of machine learning algorithms and methods in the context of predictive maintenance or autonomous and automated interaction for order processing between *smart components* or even *smart factories*. Communication-enabled components in the shop floor are the most important sources of operational, condition and process data. However, real added value can only be generated if the components map and offer their data in accordance to standards. This requires common industrial communication protocols as well as standards for semantic information modelling and frameworks for application development. In addition, the digital description of different components is the basis for automated interaction between entities of an automation network. The virtual representation of physical objects with their properties and skills can be realized and integrated into digital platform environments.

This paper is structured as follows: Section 2 gives an overview of current developments and concepts regarding I4.0-compliant communication and interaction. Concepts such as the reference architecture model Industrie 4.0 (RAMI4.0), the I4.0 component as well as the AAS and relevant parts of the industrial communication protocol OPC UA are briefly explained. Section 3 explains the mapping procedure for creating submodels for the AAS based on the OPC UA Part 100: Devices standard. Section 4 shows how the submodels of an AAS can be serialized in a neutral data format and thus made available for further use in applications. Section 5 briefly summarizes the results of this paper and gives an outlook on further developments and practice-oriented applications of the presented concept.

II. I4.0-COMPLIANT COMMUNICATION AND INTERACTION IN AUTOMATION NETWORKS

The German initiative Plattform I4.0 aims to advance the digital transformation of production. To this end, various groups are working to create a common understanding of the I4.0 implications, and to develop concepts, architectures, and technologies for its implementation. The main elements to date are the RAMI4.0, the I4.0 component and the AAS. [1]

A. RAMI4.0 and the I4.0 component

RAMI4.0 is the reference architecture proposed by the Plattform I4.0 initiative. An essential task of RAMI 4.0 is to map and link different aspects of industrial production in a common model. For this purpose, a three-dimensional solution space is set up in which interdependencies and cross-links can be represented. The three dimensions are represented through the axes (i) *Life Cycle and Value Stream*, (ii) *Hierarchy Levels* and (iii) *Layers*. In addition, the early integration of the product itself as type or instance is possible with this model. Using the RAMI4.0, different views can easily be adopted which combine different aspects from the areas of information technology (IT), engineering and manufacturing, as well as the whole product life cycle. In the context of this paper, automation components are treated exclusively as instances and placed in the models *Layers* as

well as in the axis *Hierarchy Levels*. The aspects of engineering are not further considered. [3]

The I4.0 component depicted in Figure 1, serves as a model that can describe properties, like configuration parameters, and skills of cyber-physical systems (CPS) and can be exposed to interaction partners. CPS represent real objects, e.g. a sensor, which are linked to virtual objects or processes. An I4.0 component consists of two unique parts: A physical or logical object combined with its virtual representation. This digital mapping of assets is achieved by the AAS, whereby these assets can have various manifestations. An asset can be a single component, an assembly or even a complex plant. It holds its AAS until the end of the product life cycle, during which time the amount of information available grows continuously. Through the digital mapping of components and their ability to communicate, autonomously acting holistic value creation networks will be developed in future. [4]

B. Asset Administration Shell

To meet the requirements of I4.0-compliant production environments and seamlessly integrate devices into production networks, AAS was developed as an integral part of the I4.0 component. AAS enables ubiquitous communication and common understanding between hardware and software components as well as humans through their virtual representations. As a virtual representation, the AAS transfers properties and capabilities of assets to the digital world and makes them available for active and passive interaction of all kind [5]. It bundles characteristics of assets in so-called submodels. Each of these models consists of structured information, an exemplary structure is displayed in Figure 4 on the right-hand side. Ideally, submodels map existing information models such as *OPC UA Companion Specifications*. Basically, a distinction must be made between mapping an AAS to OPC UA and creating explicit submodels based on existing specifications. Methodically, this mapping is performed in two steps. First, the implemented OPC UA server that contains a Devices model is converted to an AAS with the submodel OPC UA Devices. This model in turn is integrated into an OPC UA server that embodies an AAS. The second step is currently being discussed in an UAG of AG1. From our point of view, however, the second step is not necessary. Information is stored in form of semantically described properties holding real values in a machine-readable manner. Semantic descriptions, submodels, assets and AAS are uniquely identifiable and can therefore be addressed and referenced in I4.0 systems. Submodels are summaries of interdependent feature sets and, for example, represent skills [6]. The purpose of the AAS is the application-specific sufficient description of properties and skills of assets through those expandable submodels, which must be implemented in a consistent way using established standards for each aspect. Each of these submodels is dedicated to a different aspect of the respective AAS, e.g. safety features, process flows or specific skills such as drilling or assembly functions. The functions and information embedded in the AAS can also be used for autonomous order negotiations between s*mart factories* and their components in future [7–9]. The AAS concept offers several options for the location of an asset's AAS which are illustrated in Figure 2. The AAS can be stored on an I4.0-component as well as externally in e. g. cloud-based solutions, even hybrid distribution of AAS storage is possible. [10]

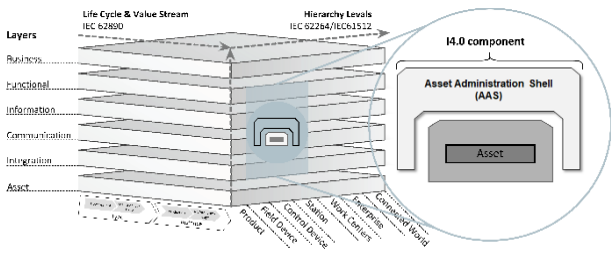

Fig. 1. RAMI4.0 according to [2] and the I4.0-component

In addition to the variable location of the AAS, there are different options regarding the implementation: (i) the passive AAS in file format, (ii) the passive AAS with an Application Programming Interface (API) and (iii) the active AAS. At the beginning of the product development process, it must be decided which AAS type is to be implemented, as the AAS remains with the asset for its entire life time. A passive role is taken if an AAS should provide all available information but is not capable of invoking actions. The passive AAS is also used as data format for the exchange of relevant data between partners. A second, more special type of a passive AAS is the representation of a server, which can be accessed through an API. In contrast, the active role is applied if the AAS communicates using I4.0-compliant languages [9]. Its communication pattern is equivalent to a peer-to-peer interaction, but not restricted to this communication pattern only. These AAS even initiate interactions themselves and carry out cooperative tasks without the need for a supervisory control unit. The three concepts are presented in the following and graphically described in Figure 2.

a) The passive AAS in file format provides a standardized and serializable data format that exposes all contained information to users and user groups with appropriate access rights. Serialization can be implemented through common formats like *JavaScript Object Notation* (JSON) or *eXtensible Markup Language* (XML) with available schemas [10]. This concept ensures a continuous, standardized flow of directly accessible information throughout all phases of an asset's life cycle

b) The structure of the AAS utilizing an API is very similar to the passive AAS and contains the same information. The distinction is made in its information provision and access: The structure can only be explored through a defined interface and therefore adds security. The interface structure is technology dependent, a general specification for basic *Create*, *Read*, *Update*, and *Delete* (CRUD) actions can be utilized.

c) In addition to CRUD functionalities, active AAS can also invoke and participate in interactions defined by protocols. Early examples are given in the proposed specification [9], containing both CRUD accesses and interaction patterns for illustrative use cases. The goal is to establish autonomously organized processes. [11]

C. Modelling production systems using OPC UA

OPC UA is a platform-independent industrial communication protocol and offers standards for data exchange in e.g. automation networks. The major added value lies in the ability to describe data machine-readable, i.e. semantically. OPC UA uses standardized information models

for this purpose, for which numerous extensions for particular applications have already been specified. [12]

To cope with the growing number of heterogeneous devices in the shop floor and to integrate them into diverse IT systems, OPC Foundation has developed the *OPC UA Part 100: Devices* specification (IEC 62541-100), which is also used in the context of this paper. This norm standardizes the mapping of and access to individual components in a generic structure. OPC UA information models are not ordered hierarchically and implement a type class system, a concept known from object-oriented programming (OOP). They resemble full-mesh node networks interconnected by references. Each node belongs to a defined *NodeClass*, which characterizes the purpose and the attributes of each particular node instance. OPC Foundation has already defined numerous convenient *NodeClasses* for further usage. Like all object types, the OPC UA Part 100 core element *TopologyElementType* and its subtypes are derived from the OPC UA *BaseObjectType*. The *BaseObjectType* is defined in Part 5 of the OPC UA specification. OPC UA Part 100 not only defines the general structure of variables and objects, but also particular properties, data elements and their data types, such as *SerialNumber*, *Model* etc. [13, 14]

III. Mapping OPC UA and the AAS

This section introduces the method for the mapping of OPC UA information models, in particular OPC UA Part 100, into submodels of the AAS. Through the recent publication of the specification "Details of the Administration Shell" new concepts, based on the AAS, can be developed. Previous work has to be validated in regard to that specification. The OPC UA model used is integrated as a stand-alone submodel, which represents a specific domain-oriented functionality. In the context of this work, this is the connection to semantically modelled data from devices in the field level. The resulting structure helps to classify, and address functionalities use case-, domain- or device-specifically. At the same time, it supports the virtual representation of the physical asset. It is intended to standardize valid submodels of the AAS and thus ensures uniformity in data representation. To achieve a valid mapping, the AAS provides different elements in a class hierarchy: The abstract class *SubmodelElement* serves as a superclass for the abstract *DataElement* class. The classes *Blob*, *ReferenceElement* and *Property* are derived from this. The *SubmodelElement-Collection* class allows you to group several *SubmodelElements* or subtypes. Only the *Blob*, *ReferenceElement*, *Property*, and *SubmodelElement-Collection* elements can be instantiated directly. Within the context of this paper we use the following two elements for

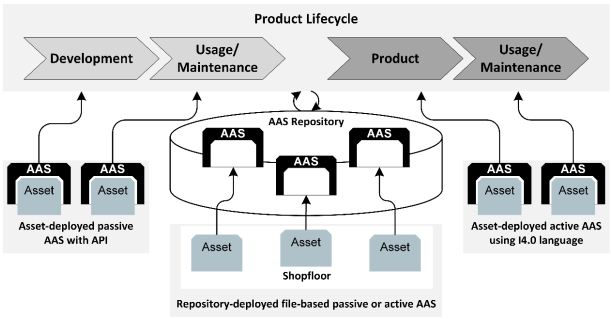

Fig. 2. Possible locations and types of ASS

the information mapping:

- The *Property* class is used to store specific values of various data types. Descriptive aspects of an asset are located here, including the device properties defined in the OPC UA specification, e.g. *SerialNumber*, *RevisionCounter*, *Manufacturer*, etc.
- The *SubmodelElementCollection* represents a flat or nested set of *SubmodelElements*. This allows grouping and hierarchical structuring within submodels. This modeling component is suitable for the mapping of OPC UA specification elements such as *DeviceType*, *ParameterSet*, *MethodSet*, *FunctionalGroup*. *ImageSet* or *Documentation*

The mapping schema is depicted on the right-hand side in Figure 3, while the class hierarchy is shown in the left-hand side. In the first step, only mandatory attributes for the OPC UA Part 100 information model are mapped. For reasons of space, the depicted table shows an excerpt of available attributes of the DeviceType only. After mapping the data model, the mandatory attributes of the OPC UA nodes such as *BrowseName*, *Description* or *Value* can be transferred. The introduced mapping method allows transformation of further existing OPC UA information models into AAS compliant submodels. These can then be standardized and therefore read and interpreted globally by applications or other I4.0 components.

IV. Serialization of the AAS Using JSON

Serialization describes the conversion of structured data to a sequential form of representation. XML and JSON formats are proposed in the specification *Details of the Administration Shell* published by the Plattform I4.0 for the serialization of AAS contents. In the context of this paper a JSON-based serialization was chosen. [10]

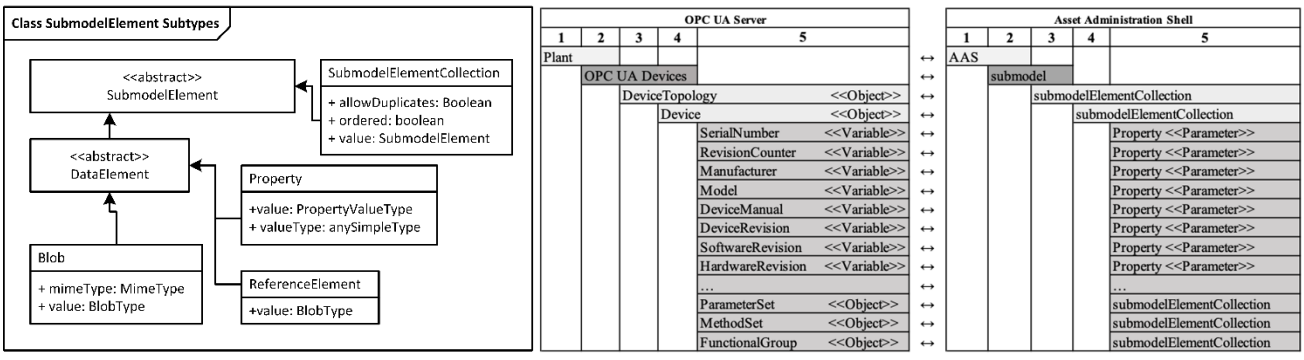

Fig. 3. Class hierarchy of the SubmodelElement and its subtypes (left) and the OPC UA AAS mapping (right)

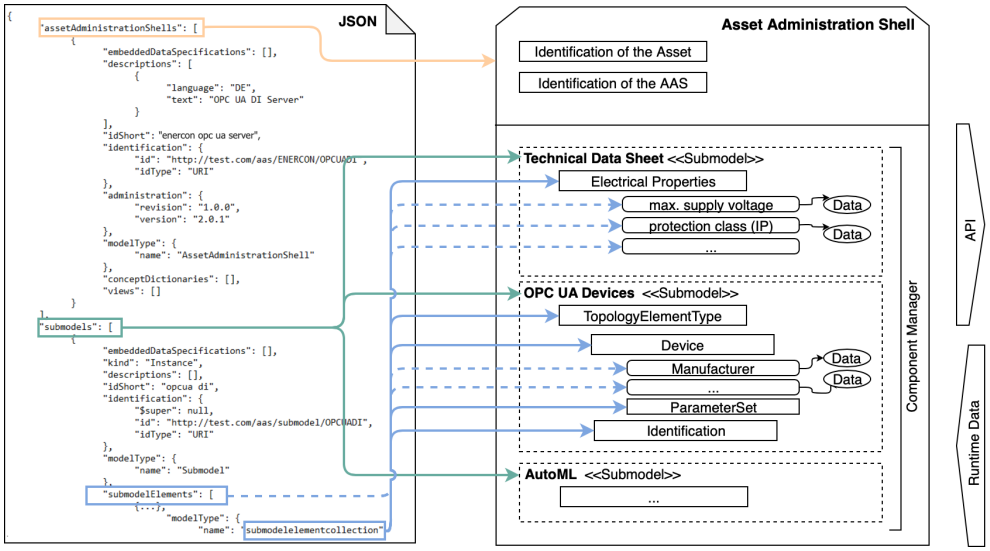

Fig. 4. Overview over the serialization of AAS submodels using JSON

The submodels of the AAS enable the standardized exchange of information between different components in automation and IT networks. On the one hand, the serialization of the AAS models in JSON allows information to be stored, enriched and exchanged over the entire life cycle of an asset. On the other hand, the connection to and linkage with OPC UA enables ubiquitous access to all static technical information as well as to variable status and process data of plants. The OPC UA connection also ensures access to higher-level systems such as the Manufacturing Execution Systems (MES) or the Enterprise Resource Planning (ERP) system if these systems implement appropriate interfaces.

A serialization represents at least one AAS with all available submodels. In addition, several AASs can be serialized simultaneously. The root element of the serialized JSON file has four aggregation levels, as indicated on the left in Figure 4: Asset Administration Shell, Assets, Submodels and Concept Descriptions. These components represent the minimum necessary structure of a serialized AAS. The rules to create an AAS-compliant JSON serialization are provided in [10].

V. Conclusion and Outlook

In the context of this paper, the basic concepts of AAS are explained and an insight to their extension is given. Also, a method for mapping OPC UA information models in submodels of the AAS was proposed. In addition, the serialization of an AAS in JSON was carried out using the example of the specification OPC UA Part 100. The implementation has shown that existing data models must be used and covered holistically in order to avoid loss of information. In addition, the manufacturers, i.e. the AAS suppliers, must pay meticulous attention to adhering to existing specifications and preventing data proliferation. In that way can the potential of AAS be exploited and automated system integration be successful. The aim is to propose the integration of the submodel developed in this paper, to the standardization committees of the Plattform I4.0 and thus create a generic model. This should be made available for the virtual representation of assets and thus ensure the continuity of information between OPC UA and the AAS. Currently, tools are being developed that use the AAS concept. One possible interface is the SAP Asset Intelligence Network (AIN), which enables AAS to be stored, displayed, supplemented and shared with cooperation partners.

Acknowledgement

The research presented in this paper was funded by the German Federal Ministry of Education and Research in the course of the project PRODISYS (FKZ 02K16C056).

References

[1] P. Marcon *et al.,* “The Asset Administration Shell of Operator in the Platform of Industry 4.0,” in *2018 18th International Conference on Mechatronics - Mechatronika (ME)*, 2018, pp. 1–5.

[2] *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*: IEEE, Jun. 2018 - Jun. 2018.

[3] P. Adolphs *et al.,* “Status report-reference architecture model industrie 4.0 (rami4. 0),” *VDI-Verein Deutscher Ingenieure eV and ZVEI-German Electrical and Electronic Manufacturers Association, Tech. Rep*, 2015.

[4] W. Dorst, *Umsetzungsstrategie Industrie 4.0: Ergebnisbericht der Plattform Industrie 4.0*: Bitkom Research GmbH, 2015.

[5] X. Ye and S. H. Hong, “Toward Industry 4.0 Components: Insights Into and Implementation of Asset Administration Shells,” *IEEE Industrial Electronics Magazine*, vol. 13, no. 1, pp. 13–25, 2019.

[6] H. Bedenbender, J. Bock, and B. Boss, “Verwaltungsschale in der Praxis,” Apr. 2019.

[7] Fachbereich Anwendungsfelder der Automation, “Sprache für I4.0-Komponenten - Interaktionsprotokoll für Ausschreibungsverfahren,” Jan. 2019.

[8] Fachbereich Anwendungsfelder der Automation, “Sprache für I4.0-Komponenten - Struktur von Nachrichten,” Jan. 2019.

[9] J. Vialkowitsch *et al.,* “I4.0-Sprache: Vokabular, Nachrichtenstruktur und semantische Interaktionsprotokolle der I4.0-Sprache,”

[10] Bundesministerium für Wirtschaft und Energie (BMWi), Ed., “Details of the Administration Shell. Part 1: The exchange of information between partners in the value chain of Industrie 4.0,” Plattform I4.0, Nov. 2018.

[11] Bundesministerium für Wirtschaft und Energie (BMWi), Ed., “Verwaltungsschale in der Praxis: Wie definiere ich Teilmodelle, beispielhafte Teilmodelle und Interaktion zwischen Verwaltungsschalen,” Plattform I4.0, Apr. 2019.

[12] W. Kim and M. Sung, “Standalone OPC UA Wrapper for Industrial Monitoring and Control Systems,” *IEEE Access*, vol. 6, pp. 36557–36570, 2018.

[13] *OPC Unified Architecture Part 5: Information Model*, 2017.

[14] *OPC Unified Architecture Part 100: Devices*, 2019.