2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA) | 978-1-6654-9996-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/ETFA52439.2022.9921705

# AI Asset Management: a Case Study with the Asset Administration Shell (AAS)

Lukas Rauh*, Mike Reichardt† and Hans D. Schotten†‡
*Fraunhofer Institute for Manufacturing Engineering and Automation (IPA), Stuttgart, Germany
†German Research Center for Artificial Intelligence GmbH (DFKI), Kaiserslautern, Germany
‡Department of Electrical and Computer Engineering, Technische Universität Kaiserslautern, Kaiserslautern, Germany
Email: lukas.rauh@ipa.fraunhofer.de, {mike.reichardt, hans_dieter.schotten }@dfki.de

***Abstract*—Driven by the goal of maintaining competitiveness within the shift toward personalized production, manufacturing companies are increasingly adopting Artificial Intelligence (AI) in their manufacturing facilities, empowered by the digital transformation, Big Data, and Cyber-Physical Systems (CPS). However, the complexity caused by the continuously growing variety of AI implementation frameworks, the initial high investment in resources, and the lack of industry-wide AI standards are hindering the rapid adoption of industrial AI. To improve asset interoperability for AI assets and thereby ensure sustainable reliability by reducing investment risks, previous work has proposed an information model to wrap AI assets into the Industry 4.0 framework. Therefore the approach utilizes the Asset Administration Shell (AAS) as a realization standard of the Digital Twin (DT) in the manufacturing Industry 4.0 context. This paper provides additional practical experience from implementing the AI AAS concept in a real use case as a basis for initiating a standardization process of the resulting AI AAS information model.**

***Index Terms*—AI Asset, Digital Twin, Asset Administration Shell**

## I. Introduction

### A. Context

With the digital transformation, Cyber-Physical Systems (CPSs) evolve the system landscape in manufacturing environments. As a unification of the physical process with added digital capabilities, a CPS enables the control, monitoring, and interconnection of physical infrastructure [1]. Applied in the manufacturing context, where CPSs are typically referred to as Cyber-Physical Production Systems (CPPSs), state-of-the-art Information Technology (IT) from the information and communications sector is disseminating into the Operation Technology (OT) world, merging the two worlds into a single heterogeneous infrastructure platform [2], [3].

As an enabler for the digital transformation, CPSs provide the ability to gather a new dimension of data variety and volume, which is made available to other connected devices and services. To fully operationalize the accumulating data, Artificial Intelligence (AI) and in particular Machine Learning (ML) methods are widely acknowledged as a promising key technology to handle the emerging data volume efficiently and extract knowledge in order to act intelligently or optimize data-producing processes [4]. The so-called field of industrial AI forms the conjunction of AI methods applied to CPSs with deep integration into the manufacturing process by directly accessing production data and providing additional value [5]. Therefore, AI models with their directly dependent software stack have to be integrated into the software stack and environment of the CPS.

### B. Motivation

In contrast to the prominent dissemination of successful applications and business models in the Information and Communications Technology (ICT) sector that leverage AI, the potential of AI in manufacturing is being realized more cautiously [6]. Even after years of digital innovation, including the introduction of CPS and Industry 4.0, high-performance cloud and edge technology, and an increasing amount of software in manufacturing environments, the number of AI applications is increasing slower in manufacturing than in the ICT sector. AI applications are often restricted to isolated development environments, typically under the supervision of a research department, including testing environments, labs, and centres. On the one hand side, this is crucial to evaluate and learn how AI will collaborate with humans to shape future work environments and identify future research directions. On the other hand side, according to Gartner, only 53% of AI solution prototypes in development environments make it to production or into the market, even when the performance results of an AI solution are promising [7].

The challenges hindering the adoption of AI are multi-faceted. There are general challenges, that require a large share of the project effort, such as the availability of a large volume of high-quality data. In manufacturing, in addition to general challenges, the main concern of hesitant companies is the effort in terms of human and financial resources to transfer AI solutions from the development and test environment to production [8]. Expert knowledge and various stakeholders are required to be involved in the development process for a successfully operating AI solution while considering industry-specific requirements such as real-time demands or documentation and audit responsibilities.

An AI development team is inter-disciplinary and involves exemplary: domain experts to provide their process knowledge, data engineers to handle the data wrangling, ML experts to build the models, and software engineers to provide software structure as a basis for continuous operating software solutions [9] [10]. To involve all stakeholders in the development process, enable faster deployment to production, and improve the maintenance of AI solutions, the term ML Operations (MLOps) appeared in

the AI community during the last decade [11]. MLOps is seen as a set of principles to continuously operate and improve AI solutions with a focus on the entire lifecycle of the AI model [12]. Tools that implement MLOps principles seek to provide an entry point for all stakeholders to interact with AI solutions, from domain experts with minor AI knowledge to AI experts.

With the increasing number of models emerging from the development phase, the need for MLOps continues to grow. The topic has gained additional momentum after the expectation of growing technical debt in production-ready AI in 2015 encouraged the development of AI reproducibility practices. According to a Cognilytica estimate, the global MLOps market will be worth 4 billion US dollars by 2025, currently attracting startups aiming to secure their market share [13]. Consequently, startups dominate the current market, and competition is driving rapid development. However, Gartner recently reported that the current tool landscape lacks mature tools [7]. In addition, due to the rapid development and competition, standards are rare. A positive example in the AI context is Open Neural Network Exchange (ONNX), a file exchange format to transfer model binary files between AI frameworks [14]. Due to tools or information models with proprietary design, compatibility and interoperability of AI tools, artifacts, and processes are often a problem. This added complexity creates the risk of dependency on the vendor of the AI tools used in a company and makes it difficult to perform a future-oriented selection of the appropriate AI toolchain. This further slows down the spread of AI.

To address the compatibility and interoperability challenges in general, a current approach in the European manufacturing sector is the establishment of the asset administration shell (AAS). As an abstract standard, the AAS was initially developed by the German Platform Industry 4.0 (I4.0) to harmonize the digital representation of industrial assets [15]. Since the AAS allows to represent physical and digital assets within its standardized information meta-model, it enables interoperability between them and reduces integration efforts to maintain competitiveness. To this end, in previous work, a so-called AI AAS was introduced to represent AI models with their corresponding dataset and configuration in this interoperable AAS format [16]. This paper extends the AI AAS concept through supplementary practical experience gained through an implementation example.

## II. Related Work

### A. AI Assets and their Lifecycle

Outside of the AI context, AI and ML are often used interchangeably as synonyms. This results from the fact that currently popular AI applications are extensively based on ML methods, as their basic requirements include the availability of large amounts of data and powerful computer hardware that have become widely accessible in the last several decades. Despite the name distinction, both AI and ML share the same term to define the central element in any of their application, the model. The model is a mathematical algorithm that is responsible for interpreting inputs and delivering prediction outputs. In addition to the model, any application utilizing AI or ML methods requires other dependencies. Multiple software tools and artifacts are required to get the mathematical algorithm operational. As a consequence, the following high-level grouped artifacts are generally considered in an AI application and its lifecycle [17][18]:

- **Model artifacts:** The model as the central element is trained as a combination of the mathematical algorithm or architecture, a set of parameters, and data(set) artifacts.
- **Data(set) artifacts:** as the fundamental resource for data-driven technology, AI and ML require a sufficiently large amount of high-quality data, during training and operation, to maintain the prediction performance.
- **Code artifacts:** for varying purposes, from the actual model code in the appropriate modeling framework to evaluation and monitoring code as well as infrastructure and serving code.
- **Configuration artifacts:** includes configuration of the code and parameters for the training or the deployment.
- **Evaluation artifacts:** used to compare and decide based on the performance of the model for the specific instance of the model used in production.

Since each of the artifacts plays an essential role in the successful development, training, evaluation, and operation of an AI model, all of the listed artifacts are important to a comprehensive lifecycle view of the AI model and therefore must be tracked throughout their respective lifecycles. With the definition of ISO/IEC 19770-1, this qualifies the listed artifacts as assets, since they have potential or actual value to an organization [19].

A similar view on the qualification as an asset can be derived from an analogy to physical assets, such as a drilling machine. The AI model as an asset is designed, produced, used, and destroyed similar to the physical machine. Further are multiple stakeholders involved during the lifecycle of the AI model asset, similar to the engineer, manufacturer, operator, and maintenance expert for a drilling machine. Finally, the AI model asset as an instance emerges from the training procedure, where material parts (data) and design (algorithm or architecture) are combined together [17]. This hierarchical analogy is sketched in Fig. 1.

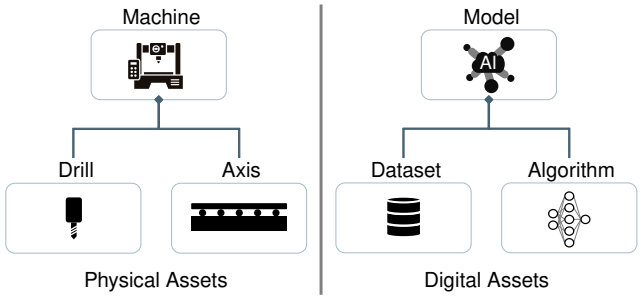


Figure 1. Hierarchical asset analogy for physical (drilling machine) and digital (AI model) assets.

### B. AI Asset Lifecycle Management

With the ongoing shift from data mining to AI and ML methods for modeling data-based systems, there has also been an

evolution in the standard process models for their development and operation lifecycle. In the last few years, multiple novel process models were proposed to substitute the long-lasting CRoss-Industry Standard Process for Data-Mining (CRISP-DM) process model, to overcome its lack of capturing the broader scoped lifecycle of AI and ML models [20]. Especially in the operational phase after modeling, there is a lack of monitoring and maintenance practices to ensure long-term high-quality predictive performance. To this end, as an example, the CRISP-ML(Q) process model aims to overcome the shortcoming in the operational phase (see the illustration in Fig. 2) and additionally adds a Quality Assurance (QA) methodology in order to integrate it directly into the development process [21]. In this way, the CRISP-ML(Q) process model aims to create a comprehensive, end-to-end process model.

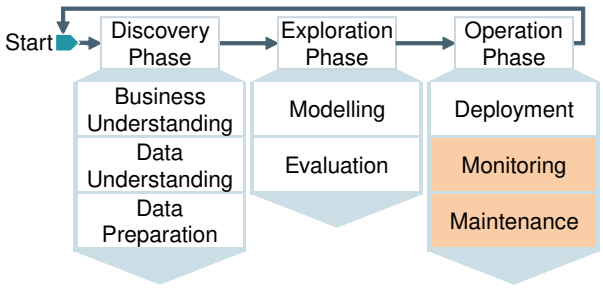


Figure 2. CRISP-ML(Q) additional stages (orange) extending the CRISP-DM stages (white) for an end-to-end AI lifecycle process model.

The idea of ML Operations (MLOps) has a similar goal. The MLOps framework gathers scientific principles, tools, and techniques of machine learning and traditional software engineering to design and build end-to-end ML development solutions. Thereby the focus of the framework is continuity [12]. Since the world changes over time, data-based systems will always be challenged to adapt to new data gathered from the changing world. As a result, continuous monitoring and maintenance are fundamental to data-based systems. For ML-based models, this includes continuous monitoring of the data and model performance with a prepared strategy to react in the form of re-training, testing, and re-deployment [22].

In the MLOps framework, the tracking and versioning of assets in the ML lifecycle is a fundamental aspect to achieve reproducibility in order to leverage transparency and a high degree of automation to reduce the effort in repetitive tasks [23]. With the gained momentum through the rise of AI and the estimated market worth (4 billion US dollars by 2025), the MLOps topic attracts software vendors to provide their MLOps tools to secure their market share [13]. There are different approaches to this among companies. Partly, companies attempt to cover a broad part of the AI lifecycle. Others rather focus on specific assets and attempt to offer the best possible solution for that asset and its dependencies. For instance, this includes specific solutions in the following areas:

- **Data Registry:** Involves the versioning and storage of data(sets) after it has been gathered from data sources.
- **Model Registry:** Involves the versioning and storage of models after they have been trained in an experiment.
- **Code Tracking:** Involves the versioning and storage of code regarding the data preparation, training, evaluation, model serving, and pipelines.
- **Experiment Tracking:** Involves the tracking, management, and comparison of training meta-data, including a reproducible combination of the involved training code, (hyper-)parameters, learning algorithm, dataset, training environment, and evaluation metrics.
- **Pipeline Execution:** Involves the continuous execution environment for data gathering, data storage, data preprocessing, training

Consequently, for the providers of these specialized solutions, integration with other tools to form a comprehensive end-to-end lifecycle toolchain is critical and at the same time their biggest challenge, considering how fast the market is evolving and the large number of new tools appearing on the market in a short period of time. Additionally, with integration between tools, the orchestration and management across the toolchain become an additional challenge [17]. Typically, the solution providers will have their own data structure and information model. Even when it is based on the JSON Notation as a quasi-standard for object information exchange, interoperability cannot be provided. As a result, customers are vendor-locked to the solution provider or at least will require a certain amount of effort to map information models if they want or have to switch the provider.

To this end, a homogeneous way of AI asset management starts with an open and standardized information model for all asset types in the AI lifecycle. A step toward interoperability would encourage the adoption of the individual tools and overall AI [17]. With the realization of the Digital Twin (DT) in the CPS ecosphere, similar challenges have been addressed with the introduction of the AAS

### *C. Digital Twin and the AAS Information Model*

With the adoption of CPPS technology and the increasing integration of digital added value into production systems, the term ”Digital Twin (DT)” became popular as a digital representation of physical assets. However, there is no clear definition of the term, which led to several interpretations. Besides the understanding as a 3D model representation for visualization or factory layout planning, the Reference Architecture Model for Industry 4.0 (RAMI4.0) defines the DT as a key element for Industry 4.0 (I4.0) [24]. To fulfill the requirements of I4.0 applications, especially interoperability between value chain stakeholders, the German Plattform I4.0 committee developed the AAS as a realization of the DT for the I4.0 domain. The AAS is a technical approach to form a digital and active representation of an asset in an I4.0 system, either physical or digital. Therefore, the AAS gathers all lifecycle-relevant hardware and software information to form the virtual representation in form of a self-description with a declarative shell structure. As a result, together AAS and asset form the so-called I4.0 component. Besides the unifying description of all types of assets in a common information meta-model, this enables interoperable

communication between the components, with standardized communication patterns and the dedicated I4.0 language [25]. This facilitates retrofitting old and proprietary assets seamlessly and enables horizontal as well as vertical communication patterns. In the future, this will enable the autonomous capabilities of I4.0 components in I4.0 systems.

The value proposition of the AAS is fundamentally based on the internal AAS information meta-model, as it is defined in [26]. The full standardization of the AAS is currently in development as an IEC standard (project IEC 63278-1 ED1). The information model is designed to meet the requirements of the RAMI4.0 architecture, is implementation language-independent, and consists of standardized entities. At its core, the AAS of an asset consists of a component that uniquely assigns the AAS to its asset. In addition, depending on the asset, there are the so-called submodels, each of which refers to a well-defined domain or subject matter. Each submodel consists of so-called submodel elements, which are described as IEC 61360 data elements and represent attributes or properties of the asset in form of a self-description for the specific submodel view in which they are contained. To achieve widely-applicable submodels to specific subjects, submodels can become standardized and thus become so-called submodel template specifications including the semantic description of contained properties. A high-level overview of the AAS entity with two exemplary submodels including two properties each is sketched in Fig. 3.

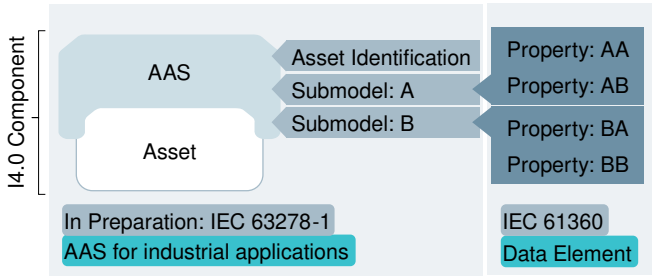


Figure 3. The AAS meta model simplified.

In summary, the AAS with the information meta-model including the domain-specific submodel template specifications and the communication language specification at its core defines an open-source standard for the implementation of the digital twin for Industry 4.0.

### D. The AI AAS

In the current standardization process, the Industrial Digital Twin Association (IDTA), as a founded user organization for the establishment of the DT for Industry 4.0 based on the AAS, is currently developing several standardized submodel template specifications [27]. Since the goal of the IDTA standardization work is the public release of the submodel template specifications, the development process is publicly available. In this way, the IDTA intends to encourage other companies to participate in the process in order to create a widely accepted and thus rapidly spreading common standard specification [28]. For this purpose, companies with their own draft ideas for additional submodels can additionally participate in the Interopera research project and contribute their submodel drafts to the standardization process in a guided process [29].

As the first standardized submodel template specification, the digital nameplate was introduced at the end of 2020 [30]. Similar to a physical nameplate attached to a machine, it describes the most basic characteristics of a physical asset for identification in the form of a standardized self-description. A closely related submodel to gather basic characteristics of software as an asset in a uniform representation, the ”software nameplate” is currently under development. The motivation for this is mainly based on the increased importance of software in general in today’s production systems and the complexity of software management including use cases like updates, patch management, license management, audits, etc.

However, the software nameplate submodel template specification will not cover the full lifecycle of AI-based applications, based on the fundamental differences between traditional software and AI software. The main difference is due to the experimental nature of AI. The development process of traditional software follows other standard processes that are far less experimental than the AI development process. The broader context of AI software includes additionally the lifecycles of the underlying dataset as well as the learning algorithm used. Consequently, meta-data must be captured at each lifecycle stage of the AI model (before training, during training, and during operational deployment).

To this end, recently the AI AAS was proposed to fill the remaining gap [16]. The concept allows relevant meta-data of AI assets to be gathered during the lifecycle phase of the AI model with its corresponding dataset and learning algorithm. Therefore, the AI AAS gathers data in the unifying AAS information meta-model. As an advantage, the AAS information meta-model allows referencing other AASs with a reference element. In this way, the data source, for instance, the real CPS can be referred to. Hence the real communication interface description can be made machine-readable to simplify the data gathering for the AI model development process or the integration of an operating AI application into the CPS. The unifying vision of the AAS for assets of any type is illustrated in Fig. 4.

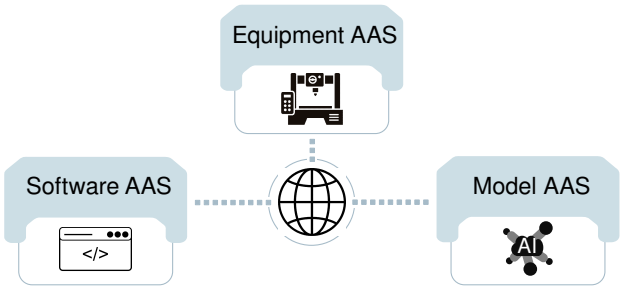


Figure 4. Vision: the AAS as a unifying self-description of any asset.

In summary, the AI AAS completes the picture of the AAS as a unifying description of all relevant assets in an Industry 4.0 environment, where CPSs are fully connected and distribute

data continuously to power AI applications. A provided self-description of any asset forms a uniform I4.0 landscape and provides the digital basis for autonomous production systems.

## III. Case Study: The AI Service Demonstrator

### A. Context and Hardware Setup

In order to gain more practical experience with the definition and implementation of a self-description for AI assets, the AI Service Demonstrator was intended as a case study for the AI AAS. According to Design Science Research evaluation, a case study applies the designed artifact to a real-world situation to evaluate its effect [31]. Thus, the AI Service Demonstrator was intended to evaluate the approach to integrate the AI AAS in an industrial application. The considered application scenario aimed to prevent potentially dangerous situations for humans in a factory environment. An AI solution is used to detect if humans and automated-driven vehicles are at a small distance from each other to prevent a collision.

For the demonstrator, the hardware setup consists of a Turtlebot (waffle form factor), an industrial camera, a router, a mini PC, and a tower PC, as outlined in Fig. 5. The industry camera is directly connected to the Windows PC with two GPUs to power the training and inference of the AI application. The PC is connected to a router via a network cable. The router is also wirelessly connected to two other devices. The first device is the mini PC with Ubuntu 20.04 LTS as an Operating System (OS). This mini PC runs a Redis server used as a message broker. The second device is the Turtlebot3 waffle. The Turtlebot is controlled via the keyboard unless the AI application detects a dangerous situation in the camera image stream.

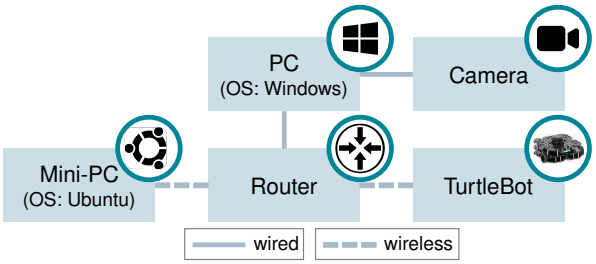


Figure 5. Hardware setup of the AI service demonstrator.

The camera provides a continuous image stream with approximately *20 fps*, where the AI application, in particular a Neural Network (NN), detects automated driven vehicles and humans. When a vehicle and a human are detected on the same image, the Turtlebot stops automatically until the human cannot be detected for a time period of *3s*. The AI application is designed as an AI pipeline that allows to replace the NN as the central element in the pipeline during operation. This is required since the surrounding environment of the Turtlebot, e.g. the lighting conditions, can negatively influence the accuracy of the NN in operation. Therefore, the current surrounding situation is continuously analyzed and the best suited NN is selected accordingly. The selection is facilitated with the AI AAS as it provides the self-description of the NNs including all the meta-data gathered during training and forms a digital representation of the NN as an asset.

### B. Component Architecture and Workflow

In Fig. 6 the component architecture and workflow of the implemented AI Service Demonstrator is depicted. The AAS-related components were implemented in the Eclipse BaSyx AAS framework, as an open-source implementation of the AAS [32]. Since the focus of the demonstrator was on the implementation of the AI AAS and its handling in live operation, the live AI application including the NN was not the focus and was therefore summarized as a black box.

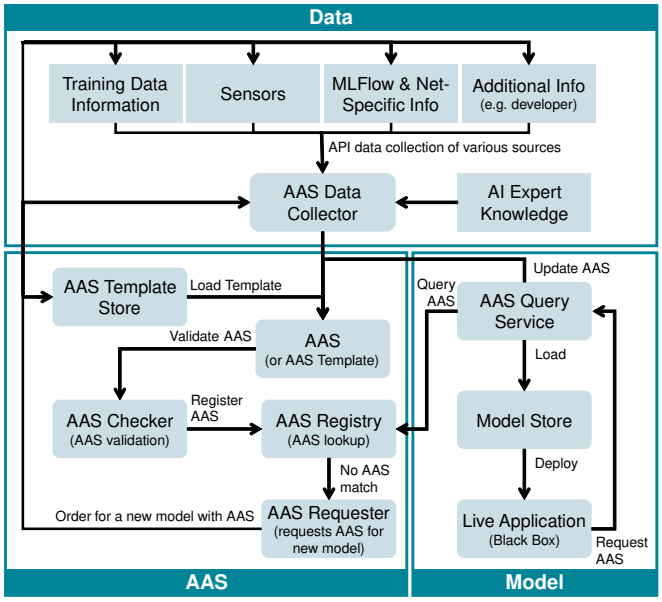


Figure 6. General component architecture of the AI service demonstrator.

If the AI live application detects a change in performance, e.g. through a drop of confidence, a new NN is requested. An AAS Query Service then uses the information of environment and net information and invokes a lookup in the AAS registry for a suitable NN, that was trained in a similar environment. For this purpose, the AAS Query Service relies on the meta-data of the AI models stored in the AI AAS during its training. If a model exists with the appropriate environment, it is loaded into the live application from the model store where each trained model is stored. If the AAS Query Service is unable to find an existing model for the present conditions, a new model training is requested. Hereby, two processes are initiated. The first one is the well-known data collection for gaining training data. For this purpose, the data is gathered from different data sources, e.g. sensors, net-specific information, and expert knowledge. This information is used for the model training. The second process is the instantiation of an AI AAS as a digital representation for the new model. The AAS Template Store and the AAS Data Collector are the key components during this process. All data sources for the training data are collected with the AAS

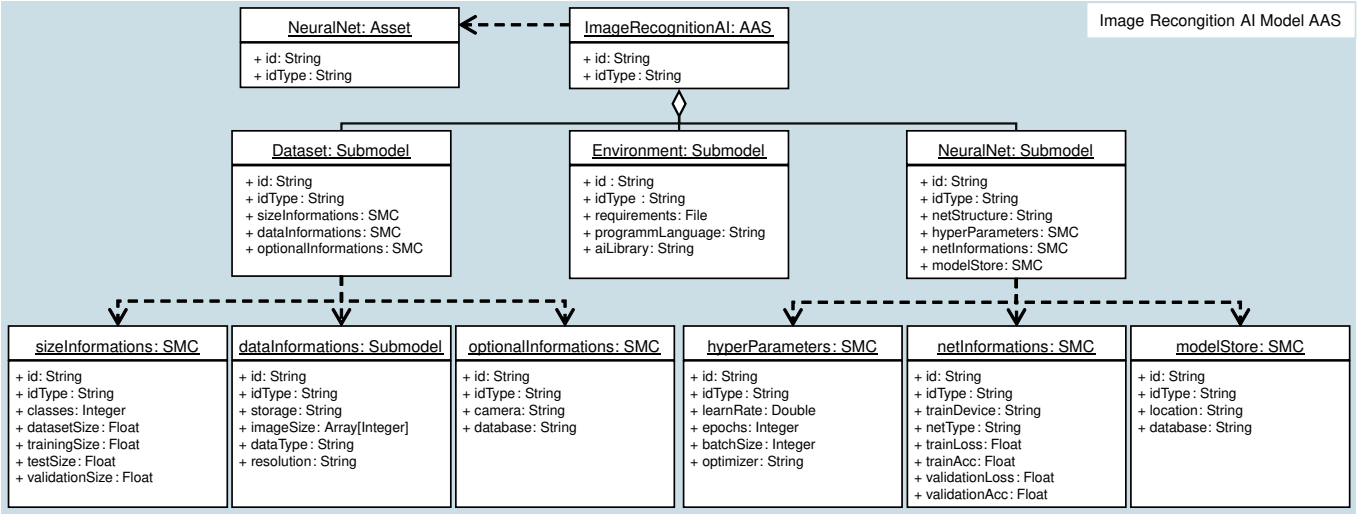


Figure 7. AAS template for the AI AAS of an image recognition CNN.

Collector and prepared so that they can be written into the AI AAS. To provide an identical structured AAS for any new model the AAS Template Store is used. This Template Store is an additional registry, which is independent of the first AAS Registry, in which the AAS of the trained models are stored. In the AAS Template Store AAS templates, which provide the possible to instantiate AAS in a given structure, are stored. The AAS Requester selects the corresponding AI AAS template out of the Template Store and the collected data from the Collector is written into the template to create the AI AAS instance for the new model. Before the new AI AAS can be registered in the AAS Registry in order to be available for the live application, the AAS undergoes an additional verification process in the AAS Checker component. Common validation issues are for example missing values or wrong data types. If the verification succeeds, the AAS is automatically registered and the model can be deployed in the live application.

### C. AI AAS Template Design

The design of the AI AAS template for its intended use targeting the self-description of the Image Recognition AI models has been a key challenge for the AAS implementation. Fig. 7 shows the resulting AAS template, that is used for the demonstrator as a modification of the original AI AAS design. The AAS consists of three different submodels (SMs) representing the involved dataset (SM dataset), the learning algorithm (SM neuralNet), and an additional submodel related to the involved AI framework environment (SM environment). The dataset and neuralNet submodels each are represented by three submodel element collections (SMCs). The SM dataset has the SMCs sizeInformation, dataInformation, and optionalInformation. Hereby describes sizeInformation general meta-information about the dataset, like the number of classes, the dataset size, and the split of the dataset for training, validation, and testing. DataInformation describes information about the single image samples, like the resolution, the datatype (e.g. PNG), and where the samples are stored. On top of that, the third SMC adds supplementary information, for example, the used sensors (the camera) or an additionally connected database.

The SM neuralNet consists of the three SMC: hyperParameter, netInfos and modelStore. SMC hyperParameter contains the hyper parameter of the NN, e.g. learning rate and optimizer. In contrast, netInfos focuses more on training results, e.g. training accuracy and training equipment like used GPU. The main focus of the third SMC modelStore is the location of the trained model. This includes the database (DB), it is stored in. The third SM (SM environment) does not have any additional SMC. It contains basic information about the framework environment, the software requirements, and in which language the model was trained.

## IV. Evaluation

The presented implementation utilizes the AI AAS concept in the AI Service Demonstrator as a practical use case. Following the Design Science Research evaluation, we intended the implementation to be a case study to evaluate the impact of the AI AAS artifact on a real-world situation [31]. The involved service components (compare Fig. 6) have been implemented specifically for the presented scenario. Consequently, a transfer of pre-existing models or those from other use cases into the developed components was beyond the scope of this work. However, the service components can be adapted to other use case scenarios since the AI AAS concept builds on a general, simple, and hierarchical understanding of AI models [16]. Other common use cases in manufacturing today include voice recognition, quality control, predictive maintenance, and more.

In general, the presented case study scenario focuses on the main characteristic of continuous AI serving and AI asset management. Accordingly, in the following, we will evaluate the AI AAS as a concept and the AAS itself in the presented scenario. The AI AAS concept was initially introduced to address the following challenges:

- **C1:** Forming a generic and hierarchical base structure of AI assets and hence separate different views reproducible on accumulating meta-data of the AI assets in their lifecycle
- **C2:** Providing a standardized AAS submodel template to improve the degree of automation
- **C3:** Enabling "plug and play" compatibility with the Industry 4.0 ecosphere and CPSs/CPPSs
- **C4:** Improving reusability of AI assets, to reduce effort and cost through fine-grained documentation of tacit development knowledge

With the presented use case in this paper, all four challenges were addressed to a certain degree. The first challenge **(C1)** was addressed through the design of the exemplary submodel template specification provided. The template accumulates the required meta-data in the AI AAS specific hierarchical structure of dataset-, learning algorithm-, and model-related properties and was based on previous experience for relevant meta-data to keep. However, since the selection of attributes and properties has been subjective and primarily focused on the intended use case of image recognition, the selection is not generally valid for other use cases. We see our approach as a starting point for the standardization process of the submodel template specification. More research is necessary to target a generic submodel template for a variety of AI use cases.

With the implementation of the provided submodel template, we addressed the second challenge **(C2)**. As a result, gathered meta-data from the involved dataset, the training, and the execution environment can be instantiated in an AAS instance directly. Accordingly, this allows generating the representative AI AAS automatically as a digital representation. Consequently, the effort for the model development process is reduced. However, due to the lack of direct integration of AAS technology into AI frameworks, some effort for the initial implementation is still required to achieve the level of automation.

In this use case scenario, the third challenge **(C3)** was not addressed entirely. The Turtlebot, on the one-hand side, is a valid CPS and has a valid use case for in-line logistics in a manufacturing environment. However, the outcome of the case study may be limited by the use case described since CPPSs have a much broader scope of use cases and types of systems to be considered. Nevertheless, the use of AAS as a future standard in manufacturing and its relevance in current technical specifications (RAMI4.0) allows for more use cases to be covered in the future and seamless integration into CPS and CPPS infrastructure.

Finally, the fourth challenge **(C4)** was addressed as a result of the former (C1-C3). The evaluated implementation improves the reusability of the trained AI model instances by providing a representative shell including all relevant meta-data accumulating during the lifecycle. The implementation, including the submodel template, reduces the effort to generate the AI AAS instances and the documentation of the development process. Similar to C2, more integration with AI frameworks would further reduce the overall effort. However, the presented application already demonstrates that an AAS delivers additional value for digital assets similar to physical assets.

In addition to evaluating the capabilities of the AI AAS as a concept, we also gained additional experience with AAS implementation in general for the considered real-world use case scenario. Eclipse BaSyx used in the AAS implementation was capable of providing the required features for the use case. It allows to easily access a specific model with the help of filter operations in the MongoDB backend of the AAS registry implementation. However, there are challenges to face to improve the usability of an AI AAS with BaSyx. BaSyx and the concept of AAS are developed with physical assets in mind, e.g. industrial robots. Physical assets differ from virtual assets, e.g. they have fewer or no physical units. Therefore, the BaSyx framework has to do adjustments, for example adding the support for dynamic length arrays or a dynamic valueType for properties. Another issue is the missing compatibility between the different BaSyx SDKs, as the Java BaSyx SDK (e.g. with MongoDB) and the Python BaSyx SDK (with CouchDB) use different database backends for storing data. This can increase the initial implementation complexity and can restrict the use of a specific programming language for its advantages, like Python workloads related to data science. To this end, compatibility between different frameworks and storage backends should be improved in the future so that access to the same data is possible from different SDKs.

In summary, the presented real-world use-case scenario implementation provides promising results. The AI AAS facilitates closing the gap between the CPS/CPPS infrastructure and AI assets by providing the basis for lifecycle management of AI assets utilizing the AAS approach. Based on the AAS information meta-model, the AI AAS is prepared for future modifications and standardization through the flexibility and modularity of the AAS standard. By providing a machine-readable self-description of AI assets with potentially included semantic annotations, the AI AAS is ready to enable autonomous production systems to include their AI assets into their DT.

## V. Conclusion and Future Research Directives

The digital transformation of manufacturing environments toward connected cyber-physical production systems has brought promising conditions for industrial AI over the past decade. The transformation shifts the challenges of deploying AI applications at the edge from the availability of hardware resources to the management and configuration of those resources. The main concerns include the hardware dependability of training and serving tasks, the increasing diversity of AI frameworks, and the maintainability of an AI solution including re-training as a result of data or model drifts. In total, these concerns currently impede the process from development to operation and require a significant effort to meet the additional industrial requirements for AI applications, including security and privacy guarantees, real-time response time, and documentation regulations.

With the introduction of the AI AAS concept in previous work, the idea of a self-description for AI assets was developed to achieve a modular, reproducible, and self-documenting AI asset management solution based on the AAS. In this work, we applied the concept to a real-world use case in the form of a

case study to gain additional practical experience for further development of the AI AAS concept. Therefore, the AI AAS was applied to a scenario for autonomous transport vehicles in an industrial environment where humans need to be recognized as part of a safety mechanism. Specifically, in this scenario, retraining the involved AI model and selecting a suitable candidate model is essential. In summary, this case study indicates that the use of AI AAS is a promising approach to gather important lifecycle information of an NN in an industrial environment so that a model selection can be made automatically based on the metadata. However, there are challenges that need to be addressed to improve the usability of an AAS implementation, especially in BaSyx. As a primary concern, the compatibility between different frameworks and storage backends should be addressed so that access to the same data is possible from different SDKs. Besides considering a related potential fix with future BaSyx releases, to evaluate the generic character of the AI AAS, other practical industrial use cases need to be examined (e.g. with AI for production planning, voice recognition, machine parameter optimization, etc.), including an evaluation of cost-benefit aspects.

Additional future research should directly address the deployment requirements and related meta-data as part of the AI AAS to further facilitate continuous deployment mechanisms. One specific research directive is an automated or semi-automated deployment of new AI model revisions on suitable and available deployment resources. This would address the versatility and flexibility of the deployed AI services regarding the available computing resources at the edge. An AAS self-description of edge computing resources and an additional service requirements section in the AI AAS could enable an automatic mapping process between AI service requirements and available computing resources to facilitate automated deployments up to fully autonomous deployments.

To finally summarize, in this paper, we presented our latest practical experience to contribute to the research directive of the AI AAS and standardization process to form a standardized self-description of AI assets to facilitate their lifecycle management.

## ACKNOWLEDGMENT

We thank the German Federal Ministry for Economic Affairs and Climate Action (BMWK) and the German Federal Ministry of Education and Research (BMBF) for providing funding for this work through the project "FabOS" (01MK20010C) and "BaSys überProd" (01IS20094C) respectively.

## REFERENCES

[1] E. A. Lee, *Cyber-Physical Systems - Are Computing Foundations Adequate?* Position Paper for NSF Workshop On Cyber-Physical Systems: Research Motivation, Techniques and Roadmap, 2006.

[2] D. Stock, T. Bauernhansl, M. Weyrich, M. Feurer, and R. Wutzke, "System Architectures for Cyber-Physical Production Systems enabling Self-X and Autonomy," in *2020 25th IEEE ETFA Proceedings*, 2020, pp. 148–155. DOI: 10.1109/ETFA46521.2020.9212182.

[3] L. Monostori *et al.*, "Cyber-physical systems in manufacturing," *CIRP Annals*, vol. 65, no. 2, 2016. DOI: 10.1016/j.cirp.2016.06.005.

[4] Y. Duan, J. S. Edwards, and Y. K. Dwivedi, "Artificial intelligence for decision making in the era of Big Data – evolution, challenges and research agenda," *IJIM*, vol. 48, pp. 63–71, 2019.

[5] J. Lee, J. Singh, and M. Azamfar, *Industrial Artificial Intelligence*, 2019.

[6] Bitkom e.V., "Kuenstliche Intelligenz kommt in Unternehmen allmaehlich voran," *Bitkom e.V.*, 2021.

[7] Gartner, Inc., Ed., *Gartner Top Strategic Technology Trends for 2021*.

[8] A. Berg, *Kuenstliche Intelligenz: Wo steht die deutsche Wirtschaft?* 2021.

[9] S. Amershi *et al.*, "Software Engineering for Machine Learning: A Case Study," in *2019 IEEE/ACM 41st ICSE-SEIP Conference*, IEEE/ACM, Ed., 2019. DOI: 10.1109/ICSE-SEIP.2019.00042.

[10] L. Baier, F. Jöhren, and Stefan Seebacher, "Challenges in the Deployment and Operation of Machine Learning in Practice," in *ECIS 2019*, 2019.

[11] S. Mäkinen, H. Skogström, E. Laaksonen, and T. Mikkonen, *Who Needs MLOps: What Data Scientists Seek to Accomplish and How Can MLOps Help?* 2021.

[12] G. Symeonidis, E. Nerantzis, A. Kazakis, and G. A. Papakostas, *MLOps - Definitions, Tools and Challenges*, 2022.

[13] Cognilytica, Ed., *ML Model Management and Operations (MLOps) Report*, 2021.

[14] The Linux Foundation, *ONNX*. [Online]. Available: https://onnx.ai.

[15] Bitkom e.V., VDMA e.V., and ZVEI e.V., Eds., *Umsetzungsstrategie Industrie 4.0: Ergebnisbericht der Plattform Industrie 4.0*, 2015.

[16] L. Rauh, S. Gärtner, D. Brandt, M. Oberle, D. Stock, and T. Bauernhansl, "Towards AI Lifecycle Management in Manufacturing Using the Asset Administration Shell (AAS)," *Procedia CIRP*, vol. 107, pp. 576–581, 2022. DOI: 10.1016/j.procir.2022.05.028.

[17] S. Idowu, D. Strüber, and T. Berger, "Asset Management in Machine Learning: A Survey," in *2021 IEEE/ACM 43rd*, IEEE, 2021.

[18] R. Isdahl and O. E. Gundersen, "Out-of-the-Box Reproducibility: A Survey of Machine Learning Platforms," in *IEEE 15th eScience Conference*, Piscataway, NJ: IEEE, 2019, pp. 86–95. DOI: 10.1109/eScience.2019.00017.

[19] ISO/IEC, *19770: Information technology - IT asset management: Part 1: IT asset management systems - Requirements*, Vernier, 2017.

[20] F. Martinez-Plumed *et al.*, "CRISP-DM Twenty Years Later: From Data Mining Processes to Data Science Trajectories," *IEEE TKDE Conference*, vol. 33, no. 8, 2021.

[21] S. Studer *et al.*, "Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology," *Machine Learning and Knowledge Extraction*, vol. 3, no. 2, 2021.

[22] C. Pohlink, A. Klug, J. Besier, J. Niestroj, N. Ofenloch-Wendel, and M. Boerner, *Maschinelles Lernen 2022: Aktuelle Trends und deren Relevanz*. Berlin, 2022.

[23] S. Schelter, J.-H. Böse, J. Kirschnick, T. Klein, and S. Seufert, "Declarative Metadata Management: A Missing Piece in End-To-End Machine Learning," in *SysML Conference*, 2018.

[24] DIN e. V., *Referenzarchitekturmodell Industrie 4.0 (RAMI4.0)*, Berlin, 2016. DOI: 10.31030/2436156.

[25] S. Bader *et al.*, *Details of the Asset Administration Shell Part 2: Interoperability at Runtime: Exchanging Information via Application Programming Interfaces (V1.0RC01)*. 2020.

[26] M. Mendes *et al.*, *Details of the Asset Administration Shell - Part 1: The exchange of information between partners in the value chain of Industrie 4.0 (V3.0RC02)*. 2022.

[27] Industrial Digital Twin Association e.V., *Homepage IDTA*. [Online]. Available: https://industrialdigitaltwin.org/.

[28] M. Hoffmeister, A. Orzelski, and S. Adhikari, *Public Repository of Submodel Templates for the AAS*. [Online]. Available: https://github.com/admin-shell-io/submodel-templates.

[29] InterOpera Konsortium, "InterOpera Homepage," [Online]. Available: https://interopera.de/.

[30] H. Bedenbender *et al.*, *ZVEI Digital Nameplate for industrial equipment (V1.0): Submodel Templates of the Asset Administration Shell*. 2020.

[31] K. Peffers, M. Rothenberger, T. Tuunanen, and R. Vaezi, *Design Science Research Evaluation*. 2012.

[32] The Eclipse Foundation, *Eclipse BaSyx*. [Online]. Available: https://www.eclipse.org/basyx/.