*future internet*

MDPI

*Article*

# The Role of Multi-Agent Systems in Realizing Asset Administration Shell Type 3

**Lucas Sakurada [1,*], Fernando De la Prieta [2] and Paulo Leitao [1,*]**

[1] Research Centre in Digitalization and Intelligent Robotics (CeDRI), Laboratório Associado para a Sustentabilidade e Tecnologia em Regiões de Montanha (SusTEC), Instituto Politécnico de Bragança, 5300-253 Bragança, Portugal

[2] BISITE Digital Innovation Hub, Edificio Multiusos I+D+i, University of Salamanca, 37007 Salamanca, Spain; fer@usal.es

* Correspondence: lsakurada@ipb.pt (L.S.); pleitao@ipb.pt (P.L.)

**Abstract**

In the context of Industry 4.0 (I4.0), the Asset Administration Shell (AAS) has been gaining significant attention in recent years. The AAS serves as a standardized digital representation of an asset, encapsulating all relevant information about the asset throughout its lifecycle. Since its introduction in 2015, the past decade has seen considerable progress in developing traditional AAS solutions, namely AAS Type 1 and Type 2. As this initial phase reaches maturity, it becomes essential to shift focus toward AAS Type 3 (proactive), a specific category that extends traditional AAS functionalities by incorporating higher levels of autonomy, intelligence, and collaborative capabilities. However, AAS Type 3 is still in its early stages, lacking formal specifications and comprehensive implementation guidelines. In this context, Multi-Agent Systems (MAS) have been investigated as a means to enhance traditional AAS solutions toward the realization of AAS Type 3, particularly by embedding autonomous, intelligent, and collaborative behaviors. Building on this perspective, this paper explores the role of MAS in realizing AAS Type 3 through a comprehensive analysis of existing agent-based AAS approaches in the literature. Furthermore, this paper proposes a reference model based on common patterns found in the literature to support the development of AAS Type 3 solutions, contributing to the discussion on the formalization of specifications and providing greater clarity on this emerging topic. Finally, to better demonstrate key aspects of the model, some illustrative examples are presented to guide its application and facilitate understanding.

**Keywords:** Asset Administration Shells; Industry 4.0; Multi-Agent Systems



Academic Editor: Gianluigi Ferrari

Received: 23 May 2025
Revised: 12 June 2025
Accepted: 17 June 2025
Published: 20 June 2025

**Citation:** Sakurada, L.; De la Prieta, F.; Leitao, P. The Role of Multi-Agent Systems in Realizing Asset Administration Shell Type 3. *Future Internet* **2025**, *17*, 270. https://doi.org/10.3390/fi17070270



## 1. Introduction

Modern factories are becoming increasingly complex and demanding intelligent, connected solutions to support flexible and adaptive operations. However, this evolution introduces significant challenges related to the integration and interoperability of industrial assets. These assets, encompassing a wide range of physical and logical components such as machines, sensors, software, and products, often originate from different vendors and exhibit considerable variation in their communication and functional capabilities. This heterogeneity complicates seamless integration and effective collaboration within increasingly complex industrial ecosystems.

To overcome these challenges, the Asset Administration Shell (AAS) [1,2] has emerged as a fundamental concept in I4.0, providing a standardized digital representation of assets.

The AAS encapsulates all relevant information about an asset throughout its entire lifecycle, enabling consistent identification, access, and management through standardized I4.0-compliant interfaces. By providing a standardized approach for asset description and interaction, the AAS acts as a cornerstone for interoperability, facilitating the integration, communication, and data exchange among heterogeneous devices, machines, and systems in industrial environments.

Since the introduction of AAS in 2015 [1], the past decade has seen considerable progress in developing traditional AAS solutions, namely AAS Type 1 (passive) and Type 2 (reactive) [3]. While these types offer considerable advantages in standardizing asset representation and enabling interoperability, they lack the active behaviors required to support fully autonomous interactions and intelligent decision-making demanded by dynamic I4.0 environments. In this context, AAS Type 3 (proactive) [3] is a distinct category that extends traditional AAS functionalities by incorporating higher levels of autonomy, intelligence, and collaborative capabilities. These features enable AAS Type 3 to autonomously respond to changing production conditions, negotiate with other assets, and optimize operations in real-time, addressing the flexibility, robustness, and reconfigurability demanded by modern industrial systems. Nevertheless, AAS Type 3 remains an underexplored topic in the literature, lacking formal specifications and comprehensive implementation guidelines.

The lack of a detailed specification for AAS Type 3 introduces uncertainty about its structure and implementation. A key question is whether AAS Type 3 requires a uniform, uniquely defined architecture, which may necessitate custom software solutions for its development and deployment, or if it allows for more flexible interpretations. For instance, the literature has increasingly recognized Multi-Agent Systems (MAS) [4] as a promising and suitable approach to support the implementation of AAS Type 3, given their conceptual alignment and inherent support for autonomy, intelligence, and collaborative capabilities [5–7]. Additionally, MAS have been developed and applied for decades in various domains, demonstrating their benefits and effectiveness in enabling decentralized control, adaptive behavior, and dynamic coordination among distributed entities [8,9], which are key requirements for realizing AAS Type 3 solutions.

Given the incipient nature of AAS Type 3, the lack of formal specifications, and the emerging trend of leveraging MAS to support its implementation, this paper explores the role of MAS in realizing AAS Type 3 through a comprehensive analysis of existing agent-based AAS approaches in the literature. Based on the insights gained from this analysis, this paper proposes an agent-based reference model for AAS Type 3 that offers a modular and standardized foundation to support the development of such solutions. This model consolidates and formalizes recurring architectural patterns and design guidelines identified, contributing to the discussion on the formalization of specifications and providing greater clarity on this emerging topic. Moreover, it extends the existing AAS structure [10], facilitating the seamless transition from established Type 1 and Type 2 implementations toward more autonomous and intelligent Type 3 solutions. Finally, to better demonstrate key aspects of the model, some illustrative examples are presented to guide its application and facilitate understanding.

The rest of the paper is organized as follows: Section 2 provides an overview of AAS, AAS Type 3, and MAS concepts. Section 3 explores the role of MAS in realizing AAS Type 3 through a comprehensive analysis of existing agent-based AAS approaches found in the literature. Section 4 proposes an agent-based reference model for realizing AAS Type 3. Finally, Section 5 concludes this paper and presents the potential future work.

## 2. Background

Several reference architectures have been established to facilitate the implementation of I4.0 [11]. Among them, the Reference Architecture Model Industrie 4.0 (RAMI4.0) [2] and the Industrial Internet Reference Architecture (IIRA) [12] stand out as the two most prominent. Although both offer valuable guidance for the development of I4.0-compliant solutions, and the IIRA is recognized as a complementary perspective to RAMI4.0 [13], this work is exclusively positioned within the RAMI4.0 framework, which formally defines and promotes the AAS as a core enabler of digitization and interoperability in industrial environments. In this context, this section introduces the foundational concepts of the AAS and MAS. These two concepts have gained increasing attention in the literature as complementary approaches for enhancing the intelligence, autonomy, collaboration, and interoperability of I4.0 solutions.

### *2.1. Asset Administration Shell*

The so-called I4.0 component, introduced by the RAMI4.0, encompasses an asset and its corresponding AAS. The AAS is defined as a standardized digital representation of an asset, encapsulating all relevant asset information throughout its lifecycle [10]. As illustrated in Figure 1 (right), the AAS has a well-defined structure comprising a collection of submodels, where each submodel can be viewed as a modular building block that encapsulates specific information about the asset, such as identification, capabilities, technical data, and operational data.

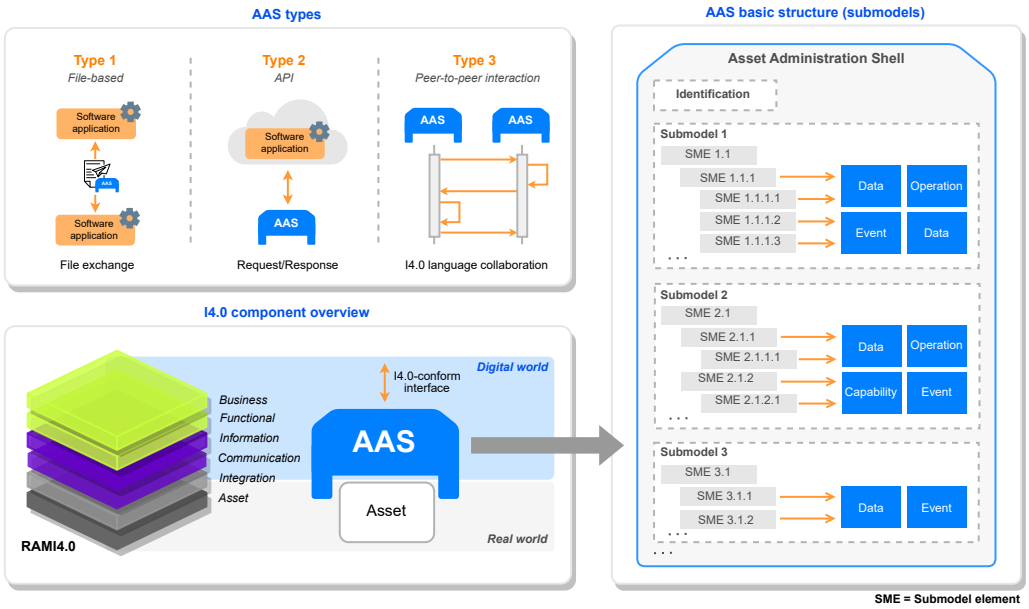


**Figure 1.** AAS overview (**bottom**), AAS types (**top**), and AAS basic structure (**right**).

Today, the Industrial Digital Twin Association (IDTA) is responsible for standardizing the AAS, providing a set of specifications and submodel templates to support AAS implementations. For instance, *Part 1: Metamodel* specification [10] describes a technology-neutral AAS information metamodel, presenting the main classes and their relationships to develop AASs. On the other hand, *Part 2: Application Programming Interfaces* specification [14] defines APIs for enabling access to the information provided by an AAS. Additionally, submodel templates guide the creation of submodels based on predefined structures. For instance, the *Asset Interfaces Description* [15] submodel template specifies an information model and a common representation for describing the interfaces of an asset service or asset-related service.

Additionally, as shown in Figure 1 (top) and described in Table 1, AASs can be classified according to their interaction pattern to exchange information and degree of autonomy to make decisions [3].

**Table 1.** Description of AAS types.

| AAS Type | Description | Interaction Pattern |
| --- | --- | --- |
| Type 1 (passive) | Represents the simplest type of AAS (file-based AAS) that stores the asset information throughout its lifecycle and can be exchanged as a whole in the form of a machine- and human-readable file, e.g., XML, JSON, and AASX | File exchange (manual) |
| Type 2 (reactive) | Acts as an API that reacts to external requests but lacks the capability to take initiatives and decisions. Such API enables the online access to asset information and can be specified in a technology-neutral way | Request/response (semi-autonomous) |
| Type 3 (proactive) | Represents a more advanced and extended form of AAS, acting as a decision-making entity that autonomously interacts with other AASs to exchange information and collaborate (e.g., through negotiation) | Peer-to-peer interaction (fully autonomous) |

The central idea behind the concept is that each asset in the industry is represented by its own AAS, transforming it into an I4.0 component. In this context, since all AASs adhere to a common standardized structure and expose I4.0-compliant interfaces, they can seamlessly provide and exchange information throughout the asset's entire lifecycle. This standardized communication enables interoperability and real-time data sharing among heterogeneous assets, regardless of the manufacturer or domain. Consequently, through their AASs, assets are expected to collaborate autonomously, coordinate processes, and support decentralized decision-making within industrial environments. However, this vision of autonomous, intelligent, and collaborative behavior is more closely aligned with AAS Type 3, which remains relatively underexplored compared to the other types, namely AAS Type 1 and Type 2, and currently lacks dedicated specifications and comprehensive implementation guidelines.

Finally, a recurring question in the community concerns the relationship between the AAS and the Digital Twin (DT) since both are defined as digital representations of assets. Given the growing popularity of both concepts, several studies [16–18] have investigated their interrelation and distinctions. These studies reveal that the relationship between AAS and DT is subject to varying interpretations. Some sources treat the AAS as a synonym for or an implementation approach to DTs, while others regard it as a foundational information model that enables the creation and interoperability of DTs. Moreover, the suitability of AAS as a representation of a DT is seen to increase along the AAS types, from Type 1 to Type 3. In particular, AAS Type 3 is considered to have the strongest alignment with the DT concept, potentially serving as a near-complete implementation technology for DT, as discussed in [18].

It is important to emphasize that this work does not aim to engage in broader conceptual debates. Therefore, it specifically focuses on the AAS, and in particular AAS Type 3, as a standardized framework for advancing digitization and interoperability in I4.0 environments, aligned with the principles initially proposed by Plattform Industrie 4.0 and IDTA.

### *2.2. Asset Administration Shell Type 3 Definition and Requirements*

As previously discussed, AAS Type 3 is still in its early stages. In this context, this section aims to examine and clarify the concept of AAS Type 3 by analyzing how it is defined in official documents, e.g., those provided by the Plattform Industrie 4.0 and IDTA, along with some definitions from selected academic research. The objective is to offer a

clearer understanding of its key characteristics and intended functionalities. Based on this analysis, a general definition and the requirements for AAS Type 3 are established.

#### 2.2.1. Definition

As summarized in Table 2, the definitions of AAS Type 3 presented in official documents, namely [3,19–21], as well as in selected research works, are brief, generic, and remain limited in scope, evidencing the early stage of conceptual development. However, they consistently emphasize several common and core aspects: the ability to interact and collaborate through the I4.0 language (VDI/VDE 2193) and the integration of autonomy and intelligence to support decision-making processes. Additional requirements, as well as concrete guidelines for the design and implementation of such components, are still in the early stages of development, particularly in official documents, which only highlight the importance of adopting the I4.0 language for communication.

**Table 2.** Summary of AAS Type 3 definitions obtained from official documents and research works.

| Ref. | AAS Type 3 (Proactive AAS) Definition |
|---|---|
| [3] * | "(...) can participate in protocol-based interactions, as defined, for example, in the VDI/VDE 2193 standard for the bidding process (...) The goal is to design decentralized processes that rely on a certain degree of autonomy or decision-making capability of the AASs" |
| [19] * | "Interactions which need peer-to-peer interactions between AAS of I4.0 components. The exchange of messages may be structured according to a grammar of an Industrie 4.0 language or may be implemented by using standardized APIs. Type 3 AAS interactions enable I4.0 components to become pro-active components which can support system integration and system interaction in a peer-to-peer environment between I4.0 components, e.g., to enable AAS assisted plug & play scenarios between I4.0 components" |
| [20] * | "An AAS Server Application with a corresponding interface for the I4.0 language is at the same time a client and a server and is a so called proactive AAS" |
| [21] * | "Type 3 is the so called "Industrie 4.0 Language" with active AAS, which is similar to agents" |
| [22] | "AASs are able to interact autonomously with each other via a standardized interface with a common syntax and semantics base, thus realizing peer-to-peer AAS interaction. (...) AASs can understand and cooperate with each other. (...) An AAS with such autonomy is called a proactive one" |
| [23] | "(...) proactive AAS can implement intelligence. The proactive AAS, that is not fully specified yet, can realize semi- or full-autonomous interactions which enables order driven production" |
| [24] | "AASs with the ability to interact as equals among themselves, offering and requesting services when necessary to meet their goals" |
| [25] | "(...) can take decisions and participate in protocol-based interactions" |
| [26] | "(...) proactive AAS also provides CRUD (create, read, update, delete) capabilities but can additionally communicate and bid with other AAS using the I4.0 language" |

* Plattform Industrie 4.0 or IDTA official document.

Based on these definitions, this work proposes the following definition for AAS Type 3: *AAS Type 3 represents the next evolutionary step of the AAS concept, extending AAS Type 1 and Type 2 solutions. AAS Type 3 not only serves as a key enabler of interoperability in I4.0 environments but also introduces a higher degree of autonomy, intelligence, and collaborative capabilities. It acts as a decision-making entity capable of autonomously interacting with other AAS Type 3 instances to exchange information and collaborate, e.g., through negotiation. Furthermore, it possesses the intelligence to self-manage its associated asset, leveraging AI techniques to support the decision-making process and adaptive behavior throughout the asset's lifecycle.*

#### 2.2.2. Requirements

From the available definitions, it is possible to identify key requirements/characteristics for AAS Type 3, namely autonomy, intelligence, and collaboration. Although this set of requirements is still limited, these elements provide a useful basis for recognizing and

classifying an AAS solution as Type 3, serving as a first step toward the development of more comprehensive design guidelines and implementation strategies. In this context, and to provide a clearer understanding of these requirements, this work defines them as follows:

- Autonomy: "the ability to make your own decisions without being controlled by anyone else" (Cambridge Dictionary). Autonomy does not mean absolute freedom to decide arbitrarily but rather the capability to make decisions respecting predefined criteria and aligned with business purposes.
- Intelligence: "the ability to learn and understand things quickly and easily" (Cambridge Dictionary). Intelligence can have different levels, e.g., from basic rule-based mechanisms to advanced AI techniques.
- Collaboration: "involving two or more people or organizations working together for a particular purpose" (Cambridge Dictionary). In this context, collaboration refers to AASs working together for a particular purpose. Unlike simple communication, collaboration goes beyond the exchange of information and comprises a deeper, interdependent process where all participants play crucial roles in achieving a common goal.

### 2.3. Multi-Agent Systems

As previously mentioned, the definitions and requirements associated with AAS Type 3 closely align with the fundamental principles of MAS, particularly regarding autonomy, intelligence, and collaborative behavior. In this context, this section introduces the concepts of agents, industrial agents, and MAS, highlighting the characteristics and functionalities that make MAS a suitable and promising approach for implementing AAS Type 3 solutions.

The development of agent-based applications has gained significant attention due to the distributed and autonomous nature of these software entities. Although no universally accepted definition exists, and the term remains a subject of ongoing debate and controversy, a classic definition of agent is as follows: "An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives" [4]. Moreover, the agents possess several characteristics:

- Encapsulation: Agents represent physical or logical objects within an environment.
- Autonomy: Agents operate without direct human or external intervention and have control over their actions and internal state.
- Learning ability: Agents can learn from their experiences.
- Reactivity: Agents perceive their environment and respond in a timely manner to changes in order to fulfill their design objectives.
- Proactiveness: Agents exhibit goal-directed behavior by taking the initiative to achieve their design objectives.
- Social ability: Agents interact with other agents (and possibly humans) to achieve their design objectives.

Additionally, agents can be specifically designed to meet the I4.0 requirements, known as industrial agents, a specialization of traditional software agents, which inherit their characteristics and are enhanced with additional capabilities to comply with several industrial requirements, namely hardware integration, reliability, fault-tolerance, scalability, industrial standard compliance, quality assurance, resilience, manageability, and maintainability [9].

Generally, industrial agents follow a two-layered approach (see Figure 2) comprising high-level control (HLC) and low-level control (LLC) [9,27]. HLC is responsible for providing intelligence and adaptation features to support decision-making processes, e.g., through monitoring, diagnosis, prediction, and optimization functions. However, it is incapable of

delivering real-time performance. On the other hand, LLC is responsible for executing low-level control tasks that guarantee real-time constraints, often through PLCs or industrial PCs running control programs compliant with IEC 61131, IEC 61499, or similar standards. This separation allows HLC to handle complex, high-level decision-making while LLC maintains stable, real-time operations.

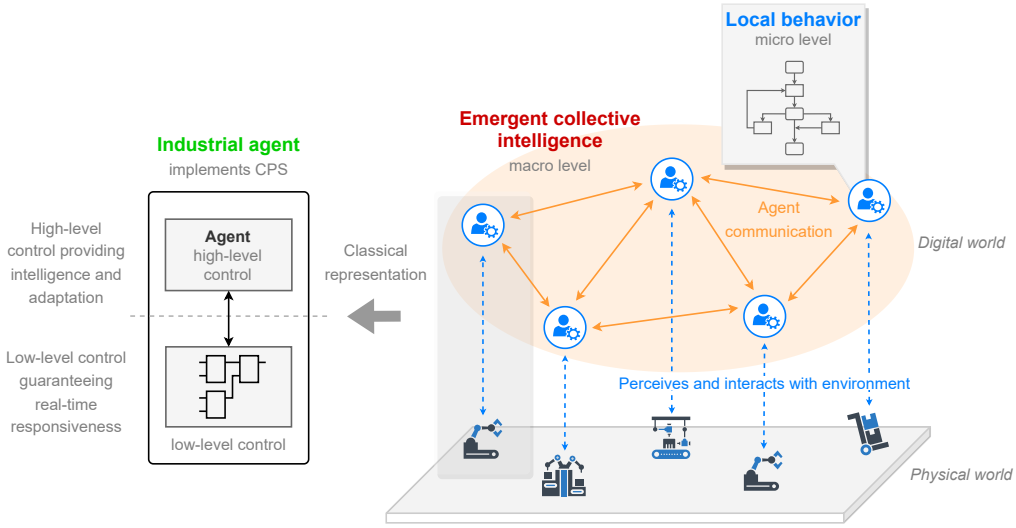


**Figure 2.** Multi-Agent Systems overview.

Addressing complex real-world problems often requires the combined and cooperative efforts of multiple agents, as a single agent has limited capabilities. As illustrated in Figure 2, a MAS can be described as a society of intelligent, autonomous, and cooperative agents representing physical or logical objects of a system. In contrast to traditional centralized control strategies, MAS offers a decentralized approach to designing large-scale, complex systems by decentralizing the system's control and distributing it to autonomous and cooperative agents rather than being concentrated in a single authority. The overall functionality of the system emerges through interactions among these individual agents, e.g., through collaboration, negotiation, and delegation of responsibilities, where each agent contributes with unique knowledge and skills toward achieving both individual and shared objectives. Furthermore, MAS enables the decomposition of complex tasks into manageable subtasks, each assigned to a capable agent, ensuring that even intricate challenges can be effectively addressed in a decentralized manner [4,8].

Beyond their theoretical foundations, several surveys on MAS (e.g., [8,9,28], to name a few) have highlighted the practical benefits of applying MAS in industrial environments, particularly offering alternatives to overcome the typical problems presented by the traditional centralized control approaches, which are not able to address the flexibility, robustness, and reconfigurability imposed on the current industrial systems.

## 3. AAS Type 3 Approaches Based on Multi-Agent Systems

As previously discussed, given the advantages provided by MAS, along with their shared characteristics and conceptual alignment with AAS Type 3, particularly regarding autonomy, intelligence, and collaborative capabilities, the current literature has increasingly considered MAS as a suitable approach for enhancing AAS solutions, often referred to as agent-based AAS solutions. Building on this perspective, this section examines the role of MAS in existing agent-based AAS approaches and explores their potential to support the realization of AAS Type 3. Moreover, it is important to highlight that a few other studies are also investigating AAS Type 3, namely [23,25]. However, due to the current trend in

the literature of using MAS to enhance AAS capabilities, this work focuses exclusively on agent-based AAS approaches.

To identify existing agent-based AAS approaches in the literature, a bibliographic search was conducted in the Scopus database using the keywords "Asset Administration Shell" and "Multi-Agent Systems", along with variations of these terms, combined using the AND operator. The search was restricted to publications from 2015 to 2025 (first semester), written in English, and classified as journal articles, conference papers, or book chapters. Additionally, only documents in which the search terms appeared in the title, abstract, or keyword fields were considered.

From the initial set of 35 documents retrieved, a screening process was carried out to identify the most relevant studies for in-depth analysis. Several factors may account for the relatively low number of documents: (i) the AAS concept is still recent, having been formally introduced only in 2015; (ii) much of the research over the past decade has focused on consolidating foundational concepts, particularly related to AAS Types 1 and 2; (iii) overall, the current literature on AAS remains limited, with only 503 documents identified through a broader Scopus search using the keyword "Asset Administration Shell".

Following the screening process, Table 3 summarizes the reviewed research works, offering valuable insights into how MAS have been integrated with AAS to enhance their capabilities. These studies cover a wide range of application domains where agent-based AASs interact to support autonomous, collaborative, and intelligent processes in industrial environments. The identified applications include smart manufacturing, dynamic production replanning, shared production, control, scheduling, human collaboration, diagnosis, and fault detection. This diversity illustrates the versatility of agent-based AASs, showing how they can be effectively adapted to meet the specific demands and operational contexts of various industrial scenarios.

**Table 3.** Summary of research works that combine MAS and AAS.

| Ref. | Application Domain | Solution Description |
| --- | --- | --- |
| [26] | Smart manufacturing (prototypical implementation in a demonstrator) | - Discusses that the asset information described in the AAS may be used as a standardized agent's knowledge representation, i.e., the agents get all the relevant information to make decisions from AAS submodels<br>- Assets: demonstrator (a stack, a crane, a stamp, and a conveyor with ramps and pushers) |
| [29,30] | Smart manufacturing (prototypical implementation in a demonstrator) | - Proposes a design pattern for implementing AASs based on industrial agents. The agents are responsible for decision-making capabilities and integration with physical assets<br>- Assets: product and industrial robot |
| [31] | Smart manufacturing (simulated experiments) | - Presents a method for developing AASs using the MAS as a key enabler to distribute intelligence and offer collaborative functionalities<br>- Assets: assets simulated in a virtual environment |
| [32] | Dynamic replanning in production environments (prototypical implementation in a demonstrator) | - Concentrates on modeling AAS submodels for resource and product agents to use in dynamic replanning applications<br>- Assets: demonstrator (a module for transportation, assembly, flashing of information on the product, and two quality control modules) |
| [33] | Human–robot shared assembly task (prototypical implementation in a demonstrator) | - Proposes a framework that combines MAS with AASs to enable the human-centered production paradigm. The focus is on the active involvement of operators in the industrial collaboration process with other assets<br>- Assets: human, robot, and camera |
| [34] | Plan and control in industrial automation systems (prototypical implementation in a demonstrator) | - Uses LLM agents to interpret complex information, generate insights, and support decision-making in industrial automation systems. The information is primarily sourced from AASs, which provide descriptive details about a specific asset<br>- Assets: transport robots and painting stations |

**Table 3.** *Cont.*

| Ref. | Application Domain | Solution Description |
| --- | --- | --- |
| [35] | Smart manufacturing (prototypical implementation in a demonstrator) | - Presents a methodology positioned with respect to the RAMI4.0 layers, which provides guidelines for integrating AASs and MAS<br>- Assets: small-scale production system (robot, punching machines, and indexed line machines) |
| [36] | Shared production in manufacturing (prototypical implementation in a demonstrator) | - Introduces a MAS architecture designed to enable shared production. The proposed architecture is grounded in holonic principles, and the AASs serve as the self-description mechanism for these agents/holons, providing the necessary information for their operation, interaction, and integration<br>- Assets: robot module, assembly module, 3D printer module, connector module, conveyor module, and quality module |
| [37] | Production scheduling (deployed in an industrial pilot coming from the bicycle production industry) | - Presents an approach in which agents operate in a decentralized manner, interacting with each other to solve scheduling problems, while the AAS provides production- and agent-related information (e.g., agent identity, description, skills, and configurations)<br>- Assets: production lines (e.g., painting and assembly lines) |
| [38] | Diagnosis and fault detection in production environments (prototypical implementation in a demonstrator) | - Integrates MAS and AASs to perform fault detection and diagnosis. All asset- and fault-related information is modeled within AAS submodels, while communication between system components is enabled through agent-based interaction using the I4.0 language<br>- Assets: production modules |

Overall, there is a general consensus that submodels serve as standardized knowledge representations or information sources for agents. These submodels capture all asset-related information in both human- and machine-readable formats, which agents can leverage to support decision-making processes and to construct or enrich their internal knowledge bases. In this context, several standardized submodels are available for this purpose, e.g., *Control Component Instance* [39], *Asset Interfaces Description* [15], and *Time-Series Data* [40], which can be used to describe the skills provided by an asset (e.g., drilling, assembly, and transport), the methods for interfacing with it (e.g., host address, port, connection state, communication protocol, endpoints, data formats, and input/output parameters), and the mechanisms for managing, accessing, and storing operational data. Additionally, other submodels can be defined to address specific application needs and to represent agent-specific information, such as descriptions of agent behaviors, decision-making strategies, or interaction protocols, providing greater transparency and understanding of agent capabilities.

The standardized structure of submodels enables agents to interpret and process them using suitable programming logic, promoting reusability, scalability, and consistency across heterogeneous assets and application domains. Moreover, agents that are equipped with Large Language Models (LLMs) could enhance their learning and interpretation of submodels. This is possible due to LLMs' ability to process both structured and unstructured data, generalize across different formats, and adapt to varied contexts, ultimately improving the system's flexibility and responsiveness.

The review also reveals that communication capabilities stand out as one of the most emphasized and valuable features of agents. This natural and inherent characteristic of agents positions them as a powerful enabler for achieving the decentralized and collaborative interactions (particularly through negotiation strategies) envisioned in AAS Type 3 implementations.

As shown in Figure 3, in a negotiation-based production scenario, information about capabilities (capability: an abstract description of a function or operation that an asset can perform, which results in tangible real-world effects, such as drilling. This description includes specific properties and constraints that define the scope and limitations of the

operation [41]) and skills (skill: the concrete implementation of a function provided by an asset, as defined by a capability. A skill enables the practical execution of a capability and typically involves parameters, including inputs and outputs, that allow for the control and monitoring of the skill's execution. Each skill must be accessible through at least one interface, which provides interaction with the skill's state machine, as well as its parameters [41]) is organized into submodels, which the agents utilize to manage their respective assets. As a result, agents representing resources possess information of the capabilities their assets can provide and the skills to execute these capabilities. On the other hand, agents representing products possess information of their required capabilities structured in a submodel that outlines the steps and requirements for production based on the required capabilities. This information allows products and resources to collaborate seamlessly to complete the production process, where resources may act as service providers by offering their capabilities as services, while products act as service requesters, seeking services aligned with their production requirements.

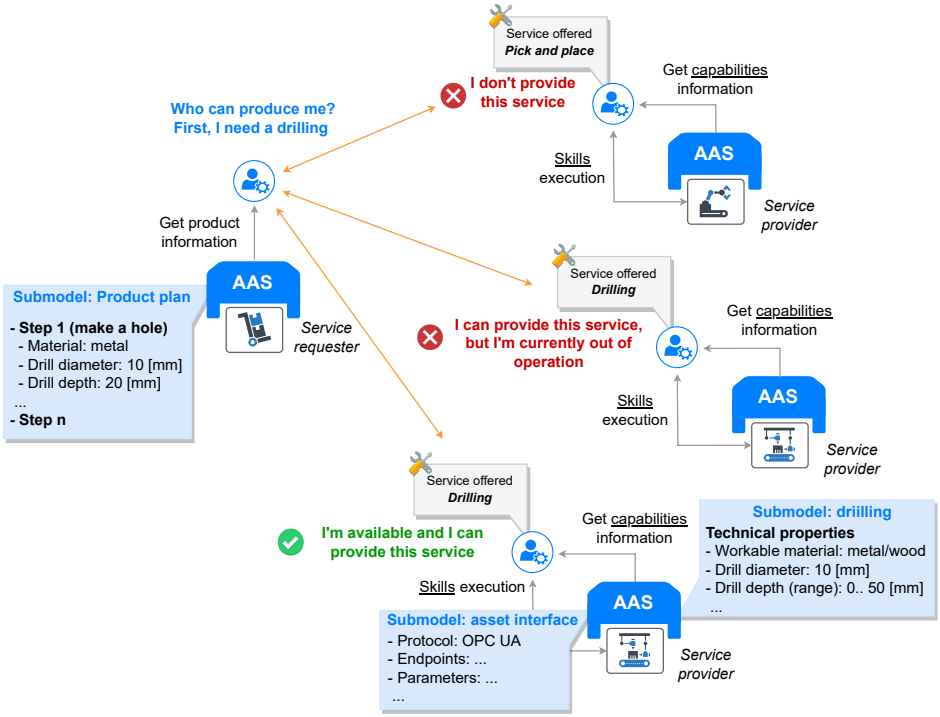


**Figure 3.** Conceptual view of a negotiation-based production scenario using agent-based AAS.

Moreover, such communication capabilities enabled by agents could open up new possibilities by transcending traditional organizational boundaries. This advancement allows the creation of shared production environments where multiple companies can collaborate in real-time. Through secure and standardized communication protocols, agents representing assets from different organizations can negotiate, allocate resources, and coordinate production tasks seamlessly.

Another important aspect concerns the interface with physical assets for data collection or control adaptation. While the AAS provides I4.0-compliant interfaces to enable seamless data exchange at the information level, it is not inherently designed to interface directly with physical assets. Instead, this connection requires customized implementation tailored to the specific characteristics, communication protocols, and constraints of each asset. In this sense, agents (or well-known industrial agents) are suitable for this purpose, as they are primarily designed to interact with both the physical and digital layers. They can serve as intelligent intermediaries that interpret and process submodel data and translate high-

level commands into asset-specific control actions, thus enabling flexible and autonomous monitoring and control of physical assets.

Although existing agent-based AAS approaches vary from simple rule-based logic to more advanced AI-based behaviors, most current implementations tend to rely primarily on the former, with limited exploration of sophisticated AI techniques. However, the broader agent literature highlights the potential for agents to act as "model consumers" of AI techniques, transitioning toward the so-called AI agents. This includes the integration of advanced learning algorithms, reasoning capabilities, and adaptive behaviors, which could significantly enhance the intelligence and autonomy of AAS-based solutions.

Additionally, some research efforts also explore the use of agents for service orchestration and the adoption of recursive architecture inspired by holonic systems (see Figure 4). In this context, each agent-based AAS encapsulates the capabilities of its associated assets and exposes them as services, enabling the dynamic composition and orchestration of services to realize higher-level services aligned with production goals.

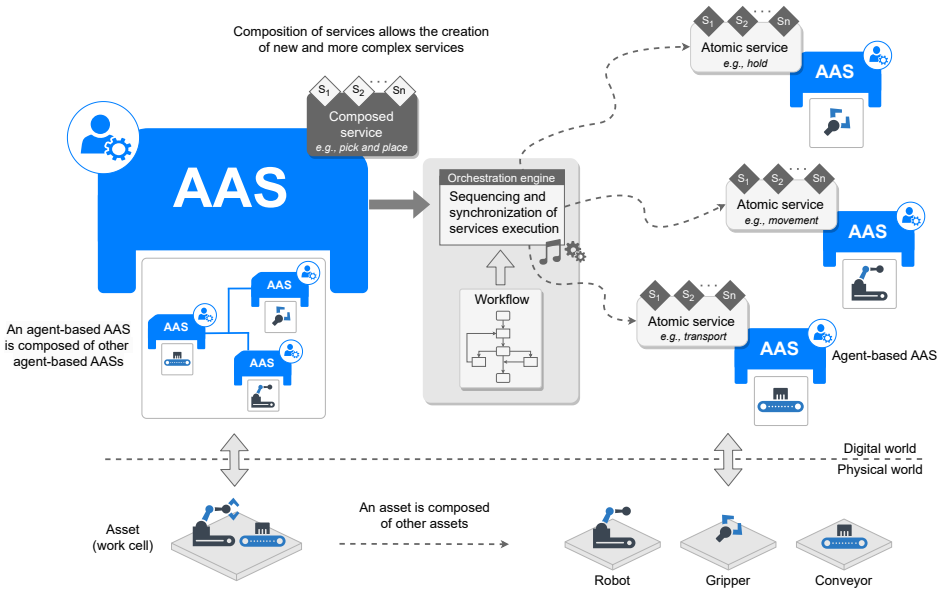


**Figure 4.** Service orchestration of holonic agent-based AAS.

For example, consider an illustrative example of a work cell comprising a robot, a gripper, and a conveyor. Individually, their respective agent-based AAS expose atomic services such as movement, hold, and transport and are responsible for managing their own services. In this context, the agent-based AAS associated with the work cell contains an orchestration engine to coordinate the execution of these services, relying on effective communication and negotiation between agents to ensure proper sequencing and synchronization. Through this engine, the work cell agent can compose atomic services into a higher-level service, such as pick and place. Extending this concept further, a station agent can orchestrate multiple work cell agents, which in turn coordinate their assets, enabling the composition of complex production tasks to be executed efficiently and flexibly.

Finally, the agent-based AAS approach is not limited to typical physical assets such as machines and production units but is also being applied to represent human operators. In this context, agent-based AASs can be specifically designed to encapsulate operator-related information, such as skills, preferences, fatigue levels, and availability. By leveraging these data, agents can dynamically allocate tasks in a manner that aligns with the operator's current capabilities and conditions, thereby enhancing task performance, ensuring quality of work, and supporting overall human well-being.

In summary, the integration of MAS with AASs represents a promising pathway toward the realization of AAS Type 3, enabling autonomous, intelligent, and collaborative behaviors in industrial environments. The reviewed literature highlights a range of innovative concepts, applications, and use cases, including decentralized negotiation, dynamic service orchestration, and human-centric approaches. Although real-world deployments are currently lacking, simulation and prototype results demonstrate significant potential for enhancing interoperability, flexibility, scalability, responsiveness, and intelligence in industrial systems. Furthermore, most studies are focused on addressing a particular use case and do not provide clear guidance for generalizing the approach or developing comprehensive, reusable solutions. Finally, given that this is a relatively new and evolving field, further research and development are essential to consolidate its theoretical and conceptual foundations before transitioning from the state of the art to practical implementation in industrial environments.

## 4. Agent-Based Reference Model for Realizing AAS Type 3

Based on the findings of the previous section, there is both theoretical and practical evidence supporting the investigation of MAS as a foundation for realizing AAS Type 3. In this context, and inspired by these insights, a reference model is proposed.

The proposed reference model aims to consolidate and formalize recurring architectural patterns and design guidelines identified by analyzing agent-based AAS approaches from the literature. Most of the approaches reviewed converge around similar solutions and principles. However, they are often presented in isolated or case-specific contexts. This convergence indicates a shared underlying structure that, while implicitly recognized, lacks an explicit and unified representation. The proposed model captures these commonalities, providing a modular and reusable foundation that supports both conceptual clarity and the practical implementation of AAS Type 3 solutions. Moreover, the model serves not only as a general guideline for developing AAS Type 3 but also as a template that offers a high-level structure from which more detailed and application-specific architectures and specifications can be derived.

### *4.1. Overview*

To develop the proposed reference model, a three-step methodology was adopted. First, a literature review was conducted to identify existing approaches integrating AAS and MAS, as discussed in the previous section. Second, these approaches were analyzed to extract common architectural patterns and design guidelines. Third, based on the most frequently observed principles and structures, a generic and reusable reference model was synthesized to guide the development of agent-based AAS solutions, with particular emphasis on supporting AAS Type 3 capabilities.

Additionally, the proposed model (see Figure 5) is designed as a modular extension of the traditional AAS structure [10]. Rather than replacing existing AAS Type 1 or Type 2 implementations, the model builds upon them by introducing an intelligent layer that enables autonomy, collaboration, and adaptive behavior. This modularity allows existing AAS-based solutions to be incrementally extended toward AAS Type 3 capabilities, enhancing their functionality without requiring a complete architectural redesign.

As shown in Figure 5, the defined reference model comprises two complementary parts: the information part and the intelligent part. The information part consists of a structured collection of submodels, which encapsulate detailed asset-specific information. These submodels provide a standardized knowledge base and serve as a comprehensive information source that supports agents in their decision-making processes and effective collaboration within industrial environments. On the other hand, the intelligent part is

conceptualized as an agent that embodies the intelligent capabilities required for AAS Type 3. This agent is responsible for interpreting and processing the information stored in the submodels, making decisions, and executing appropriate actions. Moreover, it facilitates collaboration by communicating and negotiating with other agents, enabling decentralized coordination and adaptive behavior. In summary, the intelligent part goes beyond generic AAS requirements by integrating autonomous decision-making and agent-based collaboration capabilities, which are not formalized in existing AAS specifications but are essential for realizing the full vision of AAS Type 3.

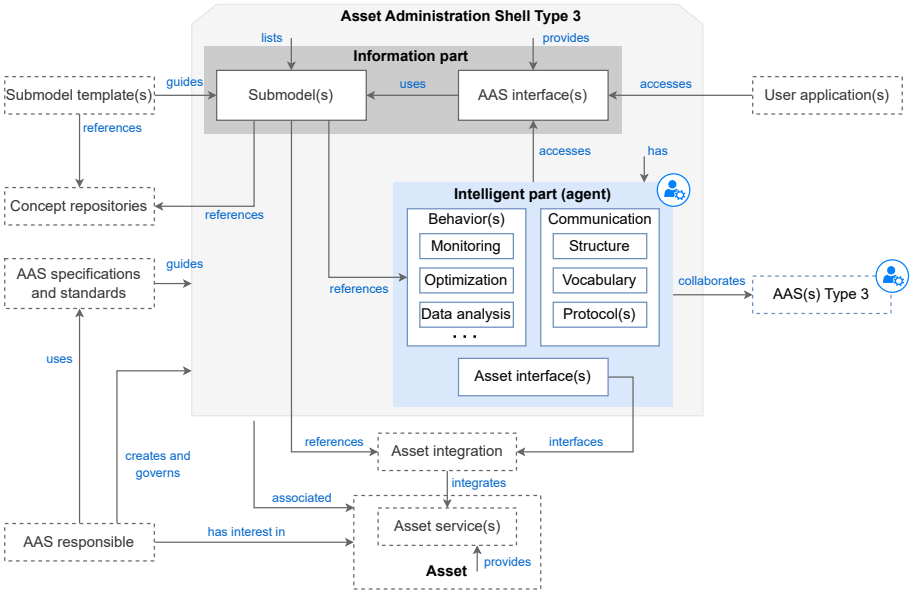


**Figure 5.** Proposed reference model for AAS Type 3 using Multi-Agent Systems.

As part of the model, a set of design guidelines have been defined to characterize the development of AAS Type 3. These guidelines serve as design principles and requirements, ensuring that any developed solution aligns with the conceptual and technical expectations of AAS Type 3. These guidelines are as follows:

- AAS Type 3 has an associated asset (individual or composite).
- AAS Type 3 provides one or more I4.0-conform interface(s).
- AAS Type 3 lists one or more submodel(s).
- AAS Type 3 uses information from submodels to support decision-making processes, enabling autonomous behavior and intelligent responses.
- AAS Type 3 collaborates with other AAS(s) Type 3 (e.g., through negotiation)
- AAS Type 3 has one or more intelligent behavior(s), e.g., based on AI techniques.
- AAS Type 3 interfaces with the asset, e.g., to collect operational data or adapt control.
- The information and intelligent parts can be deployed together or separately, depending on the system architecture and deployment requirements.
- Submodels reference the asset services provided by an asset via an asset integration and other related services (e.g., based on AI techniques).
- The user application(s) accesses the information of the AAS Type 3 via IT interfaces.
- The submodel template guides the creation of a standardized submodel.
- The submodel template may reference concept dictionaries and ontologies.
- Concept dictionaries and ontologies define the common vocabulary.
- Existing AAS Type 1 and Type 2 specifications and standards guide the development of the information part of AAS Type 3.
- The AAS responsible creates and governs the AAS Type 3.

Although the holonic concept identified in some existing works (see Figure 4) is not part of the proposed reference model, the intelligent part of the model is designed to be flexible enough to support such holonic orchestration mechanisms when needed. However, implementing such approaches would require additional specifications, such as service orchestration and composition mechanisms, which are not fully defined in the current version and are considered directions for future work.

### 4.2. Alignment with RAMI4.0

While RAMI4.0 provides a valuable framework for developing I4.0-compliant solutions, its high level of abstraction and generality may pose challenges when it comes to implementing autonomous and intelligent behaviors within AAS instances. To address these challenges, the reference model proposed in this work incorporates principles from MAS, enabling a structured approach to designing AAS Type 3. By introducing a separation between the information and intelligent parts, the model provides clearer guidance for integrating autonomy and proactive behavior in a way that remains compatible and aligned with the RAMI4.0.

As previously mentioned, the AAS reflects the asset in the digital world and is conceptually aligned with the upper five layers of RAMI4.0, namely *integration*, *communication*, *information*, *functional*, and *business*. However, the main AAS specifications (Part 1 [10] and Part 2 [14]) primarily address the *communication* and *information* layers. While these existing specifications provide a solid foundation for organizing and exchanging asset information, the *integration*, *functional*, and *business* layers are not explicitly defined.

The *integration*, *functional*, and *business* layers of RAMI4.0 can theoretically be implemented using various approaches/technologies. However, it is crucial to select one that also offers the intelligence, autonomy, and collaboration capabilities envisioned for AAS Type 3. In this context, agents are particularly well-suited to address these requirements due to their unique characteristics, as previously discussed. Agents bring the essential capabilities of autonomy, decision-making, and collaboration, which are critical for enabling AAS to act as a self-managing, intelligent component.

In this context, the *integration* layer can be enhanced through industrial agents, which are typically designed to interface with assets to collect data or to adapt control. The *functional* layer can be improved by incorporating agents that utilize AI techniques for tasks such as monitoring, diagnosis, optimization, and predictive analysis. Additionally, the *business* layer can benefit from agents' ability to make autonomous decisions, optimizing processes like resource allocation, predictive maintenance scheduling, and fault management in alignment with business strategies. Additionally, agents can enhance the *communication* layer by facilitating autonomous, decentralized communication through strategies like collaboration and negotiation. In summary, the map of the proposed model to the RAMI4.0 layers is described in Table 4.

In this context, the benefits of the proposed model are reflected across multiple layers of RAMI4.0, particularly *integration*, *communication*, *functional*, and *business*. It enables the seamless integration of assets into I4.0, supports decentralized communication through strategies such as collaboration and negotiation, empowers assets to perform autonomous functions based on AI techniques, and improves decision-making processes aligned with business objectives.

Since the information part is already well supported by formal specifications and standards provided by the IDTA (e.g., through the metamodel [10] and standardized API interfaces [14], which agents can access via standardized protocols), the following sections discuss how the agent (intelligent part) can be designed in compliance with existing standards and best practices to support broader industrial adoption and enable AAS

Type 3 capabilities. In this sense, the following sections address key features highlighted in Figure 5 (blue box) and Table 4, including interfacing with the physical asset for data collection and control adaptation, communication between AAS Type 3 instances, and strategies for developing AI-based behaviors.

**Table 4.** Relationship of the proposed model to the layers of RAMI4.0.

| RAMI4.0 Layer | Relation to Proposed Model |
| --- | --- |
| Asset | Represents the asset, i.e., every physical or logical object that has value for a company |
| Integration (*intelligent part*) | Industrial agents interface with the asset to collect data or adapt control processes (supported by the IEEE 2660.1 standard [27]) |
| Communication (*information and intelligent part*) | Data exchange follows a client–server schema (supported by AAS Part 2 specification [14]). Additionally, agents enable decentralized communication through interaction strategies such as collaboration and negotiation (supported by FIPA specifications and VDI/VDE 2193 standards [42,43]) |
| Information (*information part*) | Data collected from the asset, including sensor readings and other relevant information, are stored in a standardized and structured manner using AAS submodels (supported by AAS Part 1 specification [10]) |
| Functional (*intelligent part*) | Agents can autonomously execute adaptive functions (e.g., based on AI techniques) for tasks such as monitoring, diagnosis, optimization, and predictive analysis |
| Business (*intelligent part*) | Agents facilitate self-optimizing, decentralized decision-making aligned with organizational business objectives to enhance operational efficiency |

### *4.3. Recommended Practices for Interfacing with Assets Supported by IEEE 2660.1 Standard*

The integration with the asset is a critical aspect of AAS solutions. While the AAS may describe how to interface with an asset through a suitable submodel (e.g., *Asset Interfaces Description* [15]), the actual connection between the AAS and the physical asset remains unspecified and lacks formal guidance. Consequently, this interface is often developed in a customized, case-by-case manner, depending on the specific characteristics and communication protocols of the asset. Furthermore, even when such connections are realized, data exchange is typically unidirectional since AAS Types 1 and 2 do not include the mechanisms required for direct, active, or bidirectional interaction with the physical asset.

In this context, agents are particularly well-suited for interfacing with automation and control devices. Unlike passive data collection mechanisms, agents are intelligent components that not only gather data but also process them to make decisions and provide feedback to the asset. This enables bidirectional communication and real-time adaptability, which are essential for implementing the autonomy and intelligence envisioned in AAS Type 3 solutions. For example, consider an agent associated with a drilling machine. The agent can continuously gather real-time operational data to monitor the drilling process. If it detects an anomaly or suboptimal condition, it can autonomously respond, e.g., by adjusting a control parameter on the machine to correct the issue or by issuing a warning to a supervisory system or human operator.

In order to achieve interoperability and facilitate this integration, it is fundamental to have an easy, transparent, and reusable way of interconnecting agents with different assets. In this context, the IEEE 2660.1 standard [27] presents generic interface practices based on interaction mode and location levels. These interface practices can be used in the context of the defined reference model, particularly to show the different possibilities and guide the selection of the best interface practice to integrate agents with assets, depending on the use case and application requirements.

As illustrated in Figure 6, the IEEE 2660.1 standard defines and compares four integration patterns based on two key factors: the interaction mode between the agent and the asset and the agent's deployment location.

The interaction mode refers to the way the agent interacts with the asset:

- Tightly coupled: This interaction mode is characterized by a direct, permanent, and non-mediated connection between the agent and asset. The participants are connected via direct network communication or shared memory and follow a traditional request–response schema. This schema typically involves the agent sending a request to the asset, which processes the request and sends back a response.
- Loosely coupled: This interaction mode is characterized by an indirect connection between the agent and asset, usually mediated by one or more message brokers, following a publish–subscribe schema. Messages are sent asynchronously, meaning that the agent does not have to wait for an immediate response from the asset.

Regarding the deployment location, two possibilities are identified:

- Hybrid: The agent operates on a separate computational platform from the asset controller.
- On-device: The agent operates on the same computational platform as the asset controller.

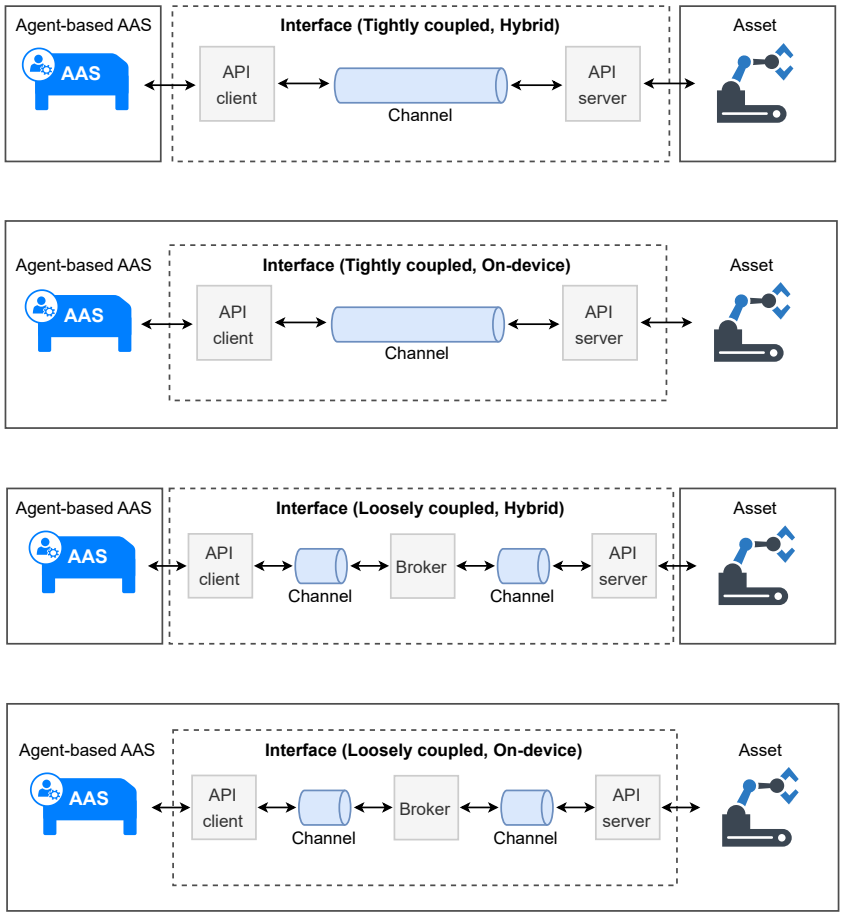


**Figure 6.** IEEE 2660.1-2020 generic integration patterns (adapted from [27]).

In industrial settings, any proposed solution must be integrated with existing deployed systems. In this context, agent integration can be quite restricted depending on the specifics of the asset, but it can also be flexible, allowing multiple integration patterns and technologies to be applied for the same use case. In such circumstances, it is important to take into account the advantages and disadvantages of each integration pattern for different applications. In this context, the IEEE 2660.1 also defines a recommendation method

that recommends the best interface practices to interconnect agents and physical assets, particularly low-level automation devices, based on user-selected criteria. These criteria include functions (e.g., control, monitoring, or simulation), application domains (e.g., factory automation, building automation, or energy and power systems), technological constraints (e.g., the ability to host agents in the asset controller), as well as the relevant characteristics for the application scenario (e.g., response time, scalability, and reusability). Based on these criteria defined by the user, the method uses a recommendation engine that evaluates existing interface practices stored in a repository, which is continuously updated with interface practices collected from several industrial agent experts who have implemented and used different interface practices. As a result, the recommendation engine produces a ranked list of recommended interface practices, each assigned a score ranging from 0 to 5.

To illustrate the application of the IEEE 2660.1 recommendation method, consider the digitization of an inspection gauge placed in a production line performing inspection tasks to detect deviations in product production. In this context, the application domain is "factory automation", the asset cannot host agents locally, and the functionality provided by the agent is the "control" of the asset, e.g., to adjust its measurement parameters that need to be changed according to the specifications of the product. In such a configuration, "response time" and "reusability" are the most important characteristics to be considered (weighted as response time 80% and reusability 20%). The results provided by the IEEE 2660.1 recommendation method are illustrated in Figure 7.

**Results**

**Recomended interface practice**

| Id practice | Location | Interaction mode | API client | Channel | Broker | Score |
|---|---|---|---|---|---|---|
| HT-7 | Hybrid | Tightly coupled | Java | OPC UA | | 3.20 |

**Details**

| Id practice | Location | Interaction mode | API client | Channel | Broker | Score |
|---|---|---|---|---|---|---|
| HT-7 | Hybrid | Tightly coupled | Java | OPC UA | | 3.20 |
| HT-5 | Hybrid | Tightly coupled | Apache Milo | OPC-UA | | 3.20 |
| HT-6 | Hybrid | Tightly coupled | Java | Ethernet/IP | | 2.56 |
| HT-3 | Hybrid | Tightly coupled | Java | HTTP | | 2.56 |
| HT-2 | Hybrid | Tightly coupled | Java | Sockets | | 1.82 |
| HT-4 | Hybrid | Tightly coupled | C/C++/SQL | Sockets | | 1.63 |
| HL-1 | Hybrid | Loosely coupled | Apache Paho | MQTT | Eclipse Mosquitto | 1.63 |
| HT-1 | Hybrid | Tightly coupled | Java | Modbus | | 1.56 |
| HL-2 | Hybrid | Loosely coupled | Java | OPC-UA | | 1.45 |
| HL-3 | Hybrid | Loosely coupled | C/C++/SQL | Sockets | DBMS | 1.15 |
| HL-4 | Hybrid | Loosely coupled | Apache Paho | MQTT | | 0.41 |

**Figure 7.** Results of the recommended interface practices for the inspection gauge.

The recommended practice ("HT-7") involves implementing a tightly coupled interface utilizing the OPC UA protocol. Due to hardware constraints preventing the hosting of agents on the asset, the suggested strategy relies on a hybrid location setup. This implies that the client operates remotely, potentially leading to response time being influenced by the communication infrastructure. If the asset is not capable of supporting the OPC UA protocol, other alternative interface practices can be considered, e.g., "HT-6" and "HT-3", which consider Ethernet/IP and HTTP protocols, respectively. Based on these considerations, the agent can be designed and implemented in alignment with the characteristics of the recommended interface to meet the desired requirements.

### *4.4. Communication Based on VDI/VDE 2193 and FIPA Standards*

The agent communication capability is another key feature for AAS Type 3 solutions and is essential for enabling seamless, real-time information exchange and collaboration among distributed assets. This capability allows AASs to negotiate, coordinate tasks in a distributed manner, and share knowledge/information, enhancing the flexibility,

reconfigurability, and responsiveness of I4.0 systems. In this context, the adoption of proper standards, such as the I4.0 language, as specified in the VDI/VDE 2193 standards, should be considered to enable interactions between AAS Type 3 solutions. These standards define both the vocabulary and message structures (VDI/VDE 2193-1) [42], as well as the interaction protocols (VDI/VDE 2193-2) [43], which can be used as the basis for agent communication design.

The importance of adopting the I4.0 language as the official communication standard rather than relying on more general agent-specific standards such as FIPA lies in two main reasons: (i) its direct alignment with the definitions and requirements of AAS Type 3, and (ii) although this work employs an agent-based approach to support AAS Type 3 functionalities, it must still be recognized and treated as an AAS Type 3 solution. Therefore, to ensure communication interoperability, it is essential that all solutions conform to common vocabulary, message structure, and interaction protocols, especially considering that other AAS Type 3 implementations may be based on different architectural or technological approaches. Nonetheless, by adopting the I4.0 language as a shared communication standard, these heterogeneous solutions can still interoperate and collaborate effectively, regardless of their internal implementation choices.

Although the I4.0 language is recommended for enabling I4.0-compliant interactions and has a broad scope, the current VDI/VDE 2193-2 standard specifies only the bidding protocol (based on the classic contract net protocol). This limitation restricts solutions that require a wider variety of interaction patterns to support diverse collaboration strategies among AAS Type 3 instances. In this context, several interaction protocols already defined in the FIPA agent communication language (ACL) standards could be adapted to align with the guidelines of the I4.0 language. This transition from FIPA-based interaction protocols to I4.0 language-based protocols is expected to be straightforward, given the strong similarity and compatibility between the two standards. In fact, the VDI/VDE 2193 standards are inspired by the FIPA specifications. For instance, Table 5 maps the parameters of the FIPA ACL message structure to the corresponding elements defined in VDI/VDE 2193-1, demonstrating their equivalence. Furthermore, the communicative acts defined in both standards exhibit only minimal differences, primarily in nomenclature. For a comparison, see [42,44].

In this context, future versions or extensions of the VDI/VDE 2193 standards could formally standardize the mapping between FIPA protocols and the I4.0 language, ensuring their official integration and enabling consistent, interoperable communication. Some examples of candidate FIPA interaction protocols are as follows: request interaction protocol [45], subscribe interaction protocol [46], propose interaction protocol [47], query interaction protocol [48], and request when interaction protocol [49]. While such formal mappings would facilitate development and promote interoperability, it is important to note that the proposed reference model remains implementable using the current FIPA and VDI/VDE 2193 standards independently. However, this may require manual integration efforts, which could limit interoperability and increase development complexity.

Another crucial aspect of communication is vocabulary, which relates to the semantic dimension of information exchange. In the I4.0 context, this information can be contained in individual submodels and should be exchangeable not only within an organization but also across company boundaries. Although these submodels follow a standardized structure defined by the AAS information metamodel, inconsistencies in their semantic interpretation may still arise. To address this, all communication partners must share a common vocabulary. This vocabulary should be based on property-based machine-readable descriptions, i.e., the attributes of an asset in the physical world that denote its characteristics. In this context, the use of standardized dictionaries such as ECLASS [50]

and IEC CDD [51] is recommended, as they enable the description of properties in an unambiguous, machine-readable, and industry-independent manner.

**Table 5.** Comparison between the message structure of VDI/VDE 2193-1 and FIPA ACL.

| VDI/VDE 2193-1 | | | | FIPA ACL |
|---|---|---|---|---|
| **Message Area** | **Message Element** | **Description** | **Use** | **Equivalent msg. Parameter** |
| `interaction-Elements` | `-` | Content of a message (submodels or another data elements) | O | `content` |
| | `type` | Purpose, i.e., intention of a message | M | `performative` |
| | `sender` | Sender of a message (AAS ID) | M | `sender` |
| | `receiver` | Receiver of a message (AAS ID) | O | `receiver` |
| | `conversationId` | ID of a conversation | O | `conversation-id` |
| `frame` | `messageId` | ID of a message | M | `-` |
| | `replyTo` | Expression referencing the original message | O | `reply-with` |
| | `replyBy` | Time by which the response must be received | O | `reply-by` |
| | `semanticProtocol` | Identification of the used semantic protocol | M | `protocol` |
| | `role` | Role of the sender of the message | O | `-` |

M: mandatory; O: optional.

Finally, to illustrate how agents enable the communication capabilities of an AAS Type 3 solution, Figure 8 presents an example in which agents communicate using the FIPA request interaction protocol, adapted to comply with the I4.0 language guidelines using proper message structures and a shared vocabulary.

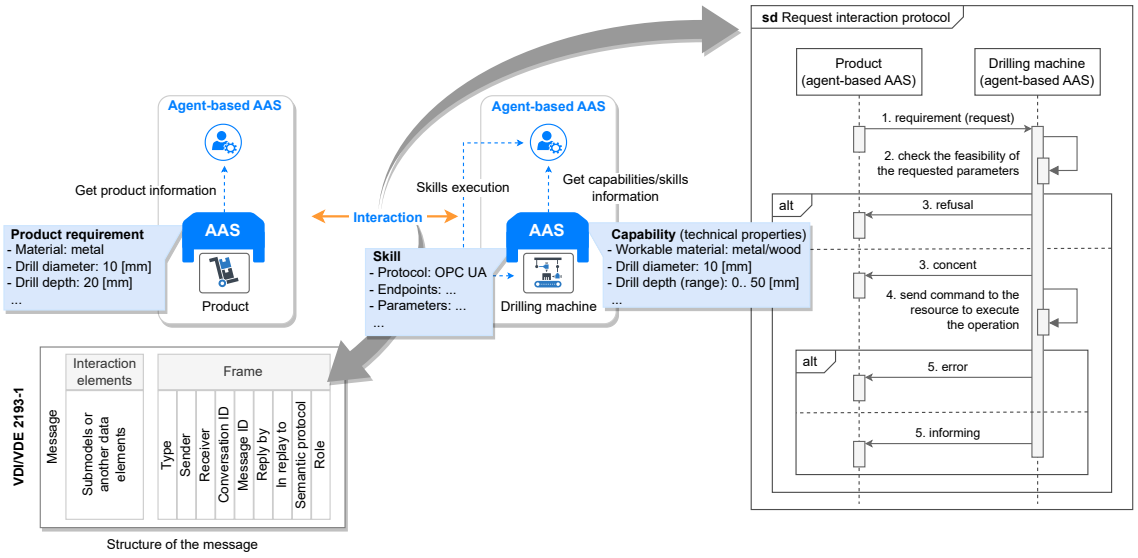


**Figure 8.** Agent communication capability to enable AAS Type 3 collaboration.

In this example, an agent-based AAS representing a product requests a drilling operation from another agent-based AAS representing a drilling machine. The drilling machine agent then evaluates the feasibility of the requested operation parameters and responds with its capability to perform the task. If the operation is feasible, the agent executes the drilling process and reports the results back to the product agent. Throughout this interaction, all necessary information, such as product requirements (e.g., drilling depth, diameter, and material), interfacing details (e.g., communication protocol, endpoints, and parameters) for executing the drilling operation, and other relevant production-related information, are stored within the respective submodels of each AAS. In this context, the content of the

exchanged messages (see Table 5) is derived from the information contained within the respective submodels (e.g., requirements), ensuring consistency and accuracy in the communication between agents. The messages are structured according to VDI/VDE 2193-1 message structure and can be exchanged in JSON format over appropriate communication protocol technologies.

### 4.5. Strategies for Developing AI-Based Behaviors

Decision-making is another key element identified in the definitions of AAS Type 3, with intelligence being a necessary requirement to support this process. In this context, agents are inherently intelligent entities capable of perceiving their environment, reasoning, and making autonomous decisions to achieve specific goals. However, implementing this process often requires AI-based strategies, ranging from rule-based mechanisms to more advanced AI techniques, e.g., based on Machine Learning (ML) or Deep Learning (DL).

The implementation of rule-based mechanisms can be simple and direct, typically using "if–then" statements that define clear decision-making logic. However, for more complex, dynamic environments, integrating advanced AI models based on ML/DL into agents significantly enhances their intelligence capabilities. In this sense, this section provides an approach to implementing and deploying such AI models into an agent-based AAS, particularly using standardized, machine-readable interchange formats, such as Predictive Model Markup Language (PMML), Portable Format for Analytics (PFA), and Open Neural Network Exchange (ONNX), which enables the description of AI models in a structured and interoperable manner.

It is important to note that the use of PMML, PFA, ONNX, or other standardized formats is highly recommended but not mandatory. These formats may bring several benefits, such as representing AI models in a standardized manner, enhancing interoperability, facilitating seamless integration across diverse systems, and reducing the time required to make a model production-ready. However, alternative approaches to representing and exchanging AI models may also be adopted, depending on specific requirements or constraints, such as unique application needs or legacy system compatibility.

As illustrated in Figure 9, the agent-based AAS should act as a model consumer, where AI models developed in a training/modeling environment are deployed to the deployment/operational environment for execution. The (re)design phase encompasses two key steps: (i) the *model producer* (e.g., experts, systems, or tools) is responsible for developing, training, and validating AI models, and (ii) the *modeling submodel* handles the conversion and exportation of these models into standardized, machine-readable interchange formats, such as PMML, PFA or ONNX, for inclusion in appropriate submodels (e.g., *AI Dataset* [52], *AI Model Nameplate* [53], and *AI Deployment* [54], which describe the AI lifecycle). On the other hand, the operational phase involves deploying the AI models. During this phase, the *model consumer*, represented by the agent-based AAS, retrieves the AI models described in PMML, PFA, ONNX, or another standardized format from the designated submodels and executes them seamlessly.

This decoupling, combined with the use of standardized, machine-readable interchange formats for AI models, facilitates AI development while enabling modularity, interoperability, and seamless sharing of models across various applications and systems. These standardized formats ensure that AI models, developed in specialized environments using appropriate programming languages and tools, can be deployed in environments that might be different from where the models were initially trained, facilitating their integration into diverse operational settings. Moreover, integrating these standardized AI models into AAS submodels offers an I4.0-compliant approach to describing and representing AI within the I4.0 context.

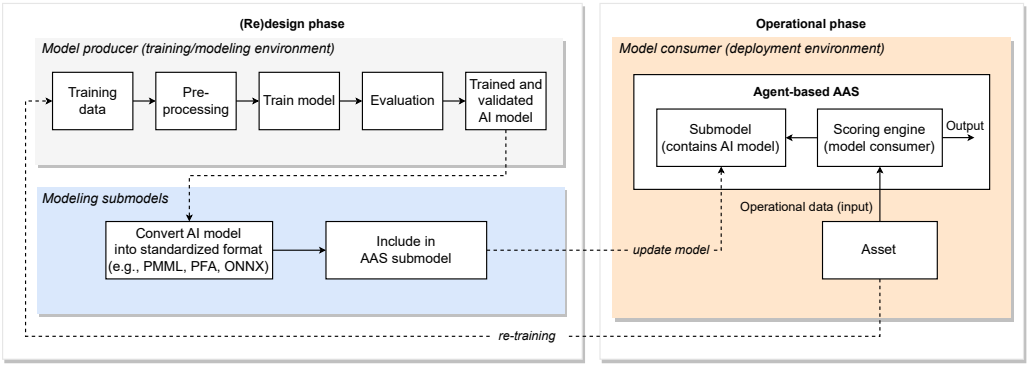


**Figure 9.** Deployment of AI models into the agent-based AAS.

Based on that, once an AI model is included in a specific submodel, the agent-based AAS needs to retrieve this model from the proper submodel and consume it using an appropriate scoring engine approach. A scoring engine is a software component that processes input data through an AI model to produce, e.g., predictions or inferences. It is able to interpret the specific structure of an AI model and applies the model to input data in real-time. Furthermore, as shown in Figure 9, as new data is generated, the AI model can be retrained and updated within the submodel, ensuring continuous improvement and adaptation to evolving operational conditions.

As an illustrative example, consider the classic Iris dataset translated into a manufacturing context, such as the monitoring of an asset. As shown in Figure 10, the asset dataset consists of various parameters collected from the asset, such as temperature, pressure, vibration, and humidity, recorded over a period of time. These parameters are typically labeled to indicate whether the asset is operating in a normal or faulty state. In a training/modeling environment, the dataset is used to train an AI model based on the random forest algorithm (e.g., utilizing tools or programming languages such as Python, R, or others). Once the model is trained and validated, it is converted into PMML format and included in the *AI Deployment* submodel. Additionally, information about the training process, dataset characteristics, requirements, and other relevant details are stored within the *AI Dataset* and *AI Model Nameplate* submodels.

In the operational phase, the agent-based AAS, which may be developed in a Java environment, retrieves the model from the *AI Deployment* submodel and consumes it using a scoring engine (e.g., through Java PMML API). By using this model, the agent-based AAS can predict asset states based on input data and trigger alerts when potential faults are detected. These alerts can inform supervisory systems, human operators, and other agents, enabling them to reorganize workflows and reallocate tasks on the fly. It is important to emphasize that the focus here is not on the AI algorithm itself (e.g., its performance or accuracy) but rather on demonstrating a standardized approach that enables agent-based AAS to consume diverse AI models tailored to specific applications in compliance with I4.0, aiming to implement intelligence in AAS Type 3 solutions.

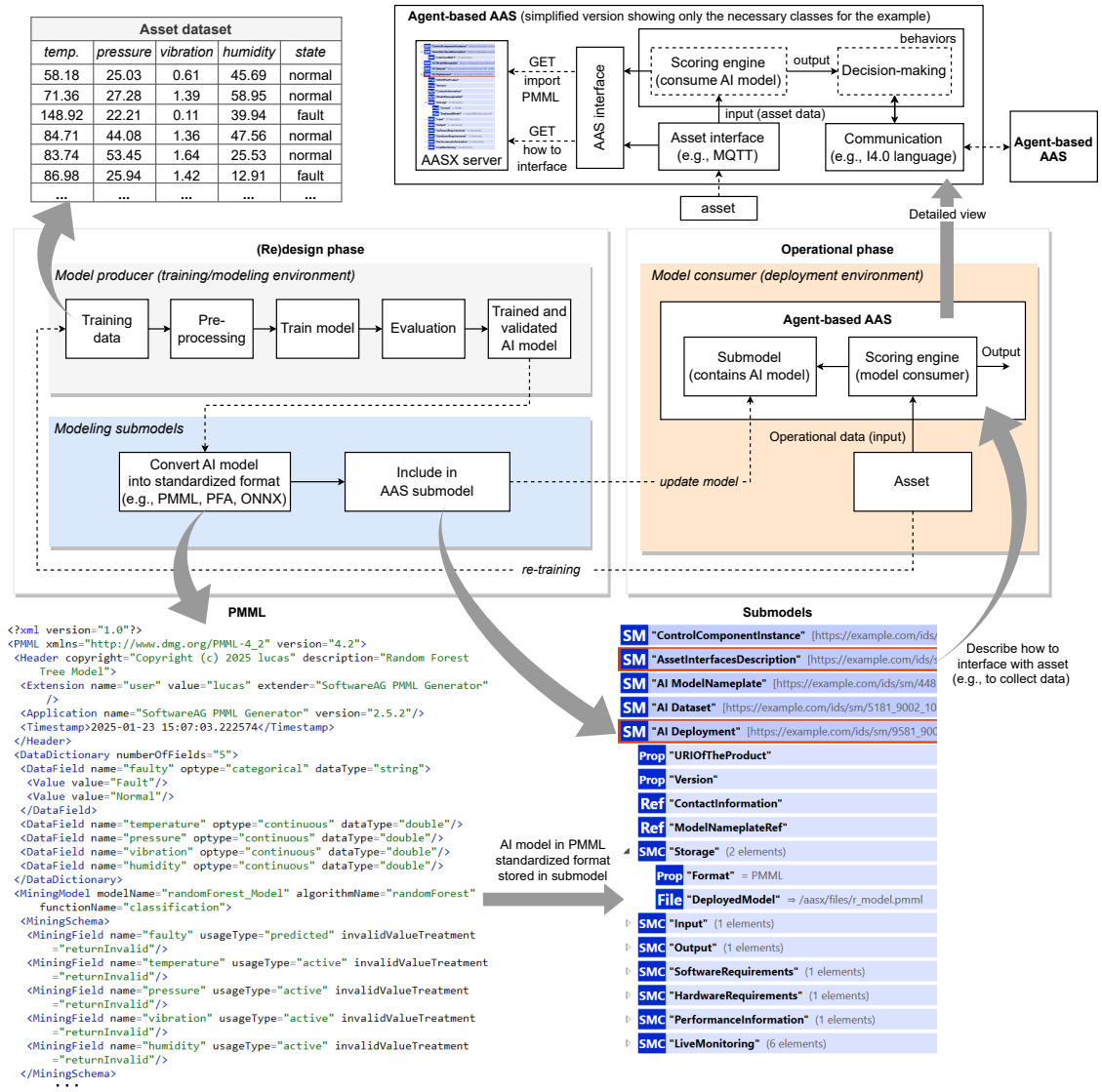


**Figure 10.** Example of AI model deployment for asset monitoring in agent-based AAS.

## 5. Conclusions and Future Work

AAS Type 3 plays a pivotal role in realizing the full vision of I4.0, characterized by fully interoperable, collaborative, intelligent, resilient, and adaptive industrial ecosystems. Despite its significance, AAS Type 3 remains in an early stage of development and still lacks comprehensive specifications and practical implementation guidelines. In this context, MAS have been investigated as a means to enhance traditional AAS solutions toward the realization of AAS Type 3, particularly by embedding autonomous, intelligent, and collaborative behaviors.

This paper explored the role of MAS in realizing AAS Type 3 through a comprehensive analysis of existing agent-based AAS approaches in the literature. In summary, MAS enable traditional AAS solutions to be extended with active and intelligent behaviors that support autonomous decision-making and collaboration in production processes. The reviewed literature highlights solutions across various application domains, namely smart manufacturing, dynamic production replanning, shared production, control, scheduling, human collaboration, diagnosis, and fault detection. However, none of the identified approaches have yet been deployed or tested in real-world scenarios, underscoring the early stage and limited practical validation of agent-based AAS solutions. On the other hand, the promising results from simulation and prototype experiments indicate significant potential for

these approaches to improve interoperability, flexibility, scalability, responsiveness, and intelligence in industrial systems. Furthermore, most studies are focused on addressing a particular use case and do not provide clear guidance for generalizing the approach or developing comprehensive, reusable solutions. Finally, given that this is a relatively new and evolving field, further research and development are essential to consolidate its theoretical and conceptual foundations before transitioning from the state of the art to practical implementation in industrial environments.

Based on the findings, a reference model was proposed that captures identified recurring patterns and best practices in a unified conceptual model, offering a standardized and modular foundation for understanding and guiding the development of AAS Type 3 solutions. The proposed model extends the existing AAS structure [10], facilitating the seamless transition from established Type 1 and Type 2 implementations toward more autonomous and intelligent Type 3 solutions. Moreover, as a reference model, it serves as a template that provides a high-level structure from which more detailed and concrete architectures and specifications can be derived and tailored to specific industrial applications and requirements.

Although the current version of the proposed reference model is conceptual in nature and has not yet been empirically validated, its structure and design guidelines are conceptually grounded in a formal analysis of existing agent-based AAS solutions reported in the literature. All these solutions have been validated through simulations, prototypes, or conceptual evaluations, as discussed in Section 3.

Future work will focus on developing a set of specifications for designing and implementing AAS Type 3 using an agent-based approach. These specifications will build upon the proposed reference model, providing more detailed technical guidance for both architectural design and practical implementation across various industrial scenarios.

Moreover, empirical validation will be pursued through complementary strategies. First, a prototype will be implemented based on a representative use case, such as negotiation among AASs in a production line, to demonstrate feasibility. Second, the model will be applied in a realistic case study, potentially in collaboration with an industrial partner, to assess its performance, scalability, and adaptability in practice. These efforts will serve to confirm the model's applicability and support its iterative refinement based on practical insights.

**Author Contributions:** Conceptualization, L.S.; methodology, L.S.; software, L.S.; investigation, L.S.; writing—original draft preparation, L.S.; writing—review and editing, L.S., F.D.l.P. and P.L.; supervision, F.D.l.P. and P.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by national funds: UID/05757—Research Centre in Digitalization and Intelligent Robotics (CeDRI), and SusTEC, LA/P/0007/2020 (DOI: 10.54499/LA/P/0007/2020). Lucas Sakurada thanks the FCT for the PhD Grant 2020.09234.BD (DOI: 10.54499/2020.09234.BD).

**Data Availability Statement:** Dataset available on request from the authors.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Plattform Industrie 4.0. *Structure of the Administration Shell—Continuation of the Development of the Reference Model for the Industrie 4.0 Component*; Plattform Industrie 4.0: Berlin, Germany, 2016.
2. *DIN SPEC 91345*; Reference Architecture Model Industrie 4.0 (RAMI4.0). DIN: Berlin, Germany, 2016. (In English)
3. Plattform Industrie 4.0. *Verwaltungsschale in der Praxis*; Plattform Industrie 4.0: Berlin, Germany, 2020.
4. Wooldridge, M. *An Introduction to MultiAgent Systems: Second Edition*; John Wiley & Sons: Hoboken, NJ, USA, 2009.

5. Karnouskos, S.; Ribeiro, L.; Leitão, P.; Lüder, A.; Vogel-Heuser, B. Key Directions for Industrial Agent Based Cyber-Physical Production Systems. In Proceedings of the 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS), Taipei, Taiwan, 6–9 May 2019; pp. 17–22.
6. Vogel-Heuser, B.; Seitz, M.; Salazar, L.A.C.; Gehlhoff, F.; Dogan, A.; Fay, A. Multi-agent systems to enable Industry 4.0. *at-Automatisierungstechnik* **2020**, *68*, 445–458. [CrossRef]
7. Sakurada, L.; Leitão, P. Multi-Agent Systems to Implement Industry 4.0 Components. In Proceedings of the 2020 IEEE Conference on Industrial Cyberphysical Systems (ICPS), Tampere, Finland, 10–12 June 2020; Volume 1, pp. 21–26.
8. Leitão, P.; Karnouskos, S.; Ribeiro, L.; Lee, J.; Strasser, T.; Colombo, A.W. Smart Agents in Industrial Cyber–Physical Systems. *Proc. IEEE* **2016**, *104*, 1086–1101. [CrossRef]
9. Karnouskos, S.; Leitão, P.; Ribeiro, L.; Colombo, A.W. Industrial Agents as a Key Enabler for Realizing Industrial Cyber-Physical Systems: Multiagent Systems Entering Industry 4.0. *IEEE Ind. Electron. Mag.* **2020**, *14*, 18–32. [CrossRef]
10. Industrial Digital Twin Association. *Specification of the Asset Administration Shell—Part 1: Metamodel*; Version 3.0.1; Industrial Digital Twin Association: Frankfurt am Main, Germany, 2024.
11. Nakagawa, E.Y.; Antonino, P.O.; Schnicke, F.; Capilla, R.; Kuhn, T.; Liggesmeyer, P. Industry 4.0 reference architectures: State of the art and future trends. *Comput. Ind. Eng.* **2021**, *156*, 107241. [CrossRef]
12. Industry IoT Consortium. *The Industrial Internet Reference Architecture*; Industry IoT Consortium: Boston, MA, USA, 2022.
13. Industry Internet Consortium. Plattform Industrie 4.0. *Architecture Alignment and Interoperability*; Industry Internet Consortium: Boston, MA, USA; Plattform Industrie 4.0: Berlin, Germany, 2017.
14. Industrial Digital Twin Association. *Specification of the Asset Administration Shell—Part 2: Application Programming Interfaces*; Version 3.0.3; Industrial Digital Twin Association: Frankfurt am Main, Germany, 2024.
15. Industrial Digital Twin Association. *IDTA 02017-1-0 Asset Interfaces Description*; Industrial Digital Twin Association: Frankfurt am Main, Germany, 2024.
16. Wagner, C.; Grothoff, J.; Epple, U.; Drath, R.; Malakuti, S.; Grüner, S.; Hoffmeister, M.; Zimermann, P. The role of the Industry 4.0 asset administration shell and the digital twin during the life cycle of a plant. In Proceedings of the 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Limassol, Cyprus, 12–15 September 2017.
17. Abdel-Aty, T.A.; Negri, E.; Galparoli, S. Asset Administration Shell in Manufacturing: Applications and Relationship with Digital Twin. *IFAC-PapersOnLine* **2022**, *55*, 2533–2538. [CrossRef]
18. Zhang, J.; Ellwein, C.; Heithoff, M.; Michael, J.; Wortmann, A. Digital twin and the asset administration shell. *Softw. Syst. Model.* **2025**, *24*, 771–793. [CrossRef]
19. Plattform Industrie 4.0. *Functional View of the Asset Administration Shell in an Industrie 4.0 System Environment*; Plattform Industrie 4.0: Berlin, Germany, 2021.
20. Plattform Industrie 4.0. *What Is the Asset Administration Shell from a Technical Perspective?* Plattform Industrie 4.0: Berlin, Germany, 2021.
21. Industrial Digital Twin Association. *Specification of the Asset Administration Shell—Part 4: Security*; Industrial Digital Twin Association: Frankfurt am Main, Germany, 2025.
22. Ye, X.; Song, W.S.; Hong, S.H.; Kim, Y.C.; Yoo, N.H. Toward Data Interoperability of Enterprise and Control Applications via the Industry 4.0 Asset Administration Shell. *IEEE Access* **2022**, *10*, 35795–35803. [CrossRef]
23. Stolze, M.; Belyaev, A.; Kosel, C.; Diedrich, C.; Barnard, A. Realizing Automated Production Planning via Proactive AAS and Business Process Models. In Proceedings of the 2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA), Padova, Italy, 10–13 September 2024; pp. 1–8.
24. López, A.; Casquero, O.; Estévez, E.; Armentia, A.; Orive, D.; Marcos, M. An industrial agent-based customizable platform for I4.0 manufacturing systems. *Comput. Ind.* **2023**, *146*, 103859. [CrossRef]
25. Grunau, S.; Redeker, M.; Göllner, D.; Wisniewski, L. The Implementation of Proactive Asset Administration Shells: Evaluation of Possibilities and Realization in an Order Driven Production. In Proceedings of the Kommunikation und Bildverarbeitung in der Automation, Lemgo, Germany, 28–29 October 2022; Jasperneite, J., Lohweg, V., Eds.; Springer: Berlin/Heidelberg, Germany, 2022; pp. 131–144.
26. Vogel-Heuser, B.; Ocker, F.; Scheuer, T. An approach for leveraging Digital Twins in agent-based production systems. *at-Automatisierungstechnik* **2021**, *69*, 1026–1039. [CrossRef]
27. *IEEE Std 2660.1-2020*; IEEE Recommended Practice for Industrial Agents: Integration of Software Agents and Low-Level Automation Functions. IEEE: New York, NY, USA 2021; pp. 1–43.
28. Monostori, L.; Váncza, J.; Kumara, S. Agent-Based Systems for Manufacturing. *CIRP Ann.* **2006**, *55*, 697–720. [CrossRef]
29. López, A.; Casquero, O.; Marcos, M. Design patterns for the implementation of Industrial Agent-based AASs. In Proceedings of the 2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS), Victoria, BC, Canada, 10–12 May 2021; pp. 213–218.

30. López, A.; Casquero, O.; Estévez, E.; Leitão, P.; Marcos, M. Towards the generic integration of agent-based AASs and Physical Assets: A four-layered architecture approach. In Proceedings of the 2021 IEEE 19th International Conference on Industrial Informatics (INDIN), Palma de Mallorca, Spain, 21–23 July 2021; pp. 1–6.
31. Sakurada, L.; Leitao, P.; De La Prieta, F. Engineering a Multi-agent Systems Approach for Realizing Collaborative Asset Administration Shells. In Proceedings of the 2022 IEEE International Conference on Industrial Technology (ICIT), Shanghai, China, 22–25 August 2022; pp. 1–6.
32. Jungbluth, S.; Hermann, J.; Motsch, W.; Pourjafarian, M.; Sidorenko, A.; Volkmann, M.; Zoltner, K.; Plociennik, C.; Ruskowski, M. Dynamic Replanning using Multi-Agent Systems and Asset Administration Shells. In Proceedings of the 2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA), Stuttgart, Germany, 6–9 September 2022; pp. 1–8.
33. Sidorenko, A.; Motsch, W.; van Bekkum, M.; Nikolakis, N.; Alexopoulos, K.; Wagner, A. The MAS4AI framework for human-centered agile and smart manufacturing. *Front. Artif. Intell.* **2023**, *6*, 1241522. [CrossRef] [PubMed]
34. Xia, Y.; Shenoy, M.; Jazdi, N.; Weyrich, M. Towards autonomous system: Flexible modular production system enhanced with large language model agents. In Proceedings of the 2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA), Sinaia, Romania, 12–15 September 2023; pp. 1–8.
35. Sakurada, L.; De La Prieta, F.; Leitao, P. A Methodology for Integrating Asset Administration Shells and Multi-agent Systems. In Proceedings of the 2023 IEEE 32nd International Symposium on Industrial Electronics (ISIE), Helsinki, Finland, 19–21 June 2023; pp. 1–6.
36. Bernhard, A.T.; Jungbluth, S.; Karnoub, A.; Sidorenko, A.; Motsch, W.; Wagner, A.; Ruskowski, M. *I4.0 Holonic Multi-Agent Testbed Enabling Shared Production*; Springer Nature: Cham, Switzerland, 2024; pp. 231–250.
37. Siatras, V.; Bakopoulos, E.; Mavrothalassitis, P.; Nikolakis, N.; Alexopoulos, K. Production Scheduling Based on a Multi-Agent System and Digital Twin: A Bicycle Industry Case. *Information* **2024**, *15*, 337. [CrossRef]
38. Rübel, P.; Motsch, W.; Bernhard, A.; Jungbluth, S.; Ruskowski, M. Agent-Based Communication for Fault Diagnosis in Skill-Based Production Environments Using Messages Based on I4.0 Language and Asset Administration Shells. In Proceedings of the Advances in Artificial Intelligence in Manufacturing II, Athens, Greece, 16 October 2025; pp. 157–170.
39. Industrial Digital Twin Association. *IDTA 02016-1-0—Control Component Instance*; Industrial Digital Twin Association: Frankfurt am Main, Germany, 2023.
40. Industrial Digital Twin Association. *IDTA 02008-1-1 Time Series Data*; Industrial Digital Twin Association: Frankfurt am Main, Germany, 2023.
41. Köcher, A.; Belyaev, A.; Hermann, J.; Bock, J.; Meixner, K.; Volkmann, M.; Winter, M.; Zimmermann, P.; Grimm, S.; Diedrich, C. A reference model for common understanding of capabilities and skills in manufacturing. *at-Automatisierungstechnik* **2023**, *71*, 94–104. [CrossRef]
42. *VDI/VDE 2193*; Part 1—Language for I4.0 Components - Structure of Messages. VDI/VDE: Düsseldorf, Germany, 2020.
43. *VDI/VDE 2193*; Part 2—Language for I4.0 components - Interaction protocol for bidding procedures. VDI/VDE: Düsseldorf, Germany, 2020.
44. Foundation for Intelligent Physical Agents. *FIPA Communicative Act Library Specification*; Foundation for Intelligent Physical Agents: Geneva, Switzerland, 2002.
45. Foundation for Intelligent Physical Agents. *FIPA Request Interaction Protocol Specification*; Foundation for Intelligent Physical Agents: Geneva, Switzerland, 2002.
46. Foundation for Intelligent Physical Agents. *FIPA Subscribe Interaction Protocol Specification*; Foundation for Intelligent Physical Agents: Geneva, Switzerland, 2002.
47. Foundation for Intelligent Physical Agents. *FIPA Propose Interaction Protocol Specification*; Foundation for Intelligent Physical Agents: Geneva, Switzerland, 2002.
48. Foundation for Intelligent Physical Agents. *FIPA Query Interaction Protocol Specification*; Foundation for Intelligent Physical Agents: Geneva, Switzerland, 2002.
49. Foundation for Intelligent Physical Agents. *FIPA Request When Interaction Protocol Specification*; Foundation for Intelligent Physical Agents: Geneva, Switzerland, 2002.
50. Plattform Industrie 4.0; ECLASS. *Modelling the Semantics of Data of an Asset Administration Shell with Elements of ECLASS*; Plattform Industrie 4.0: Berlin, Germany; ECLASS: Cologne, Germany, 2021.
51. *IEC 61360-4*; IEC/SC 3D—Common Data Dictionary (CDD—V2.0018.0001). International Electrotechnical Commission: London, UK, 2005.
52. Industrial Digital Twin Association. *IDTA 02058-1-0 Artificial Intelligence Dataset*; Industrial Digital Twin Association: Frankfurt am Main, Germany, 2025.

53. Industrial Digital Twin Association. *IDTA 02060-1-0 Artificial Intelligence Model Nameplate*; Industrial Digital Twin Association: Frankfurt am Main, Germany, 2025.
54. Industrial Digital Twin Association. *IDTA 02059-1-0 Artificial Intelligence Deployment*; Industrial Digital Twin Association: Frankfurt am Main, Germany, 2025.