Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 59-30 (2025) 671–676

aas-middleware: Enabling Interoperable, Real-Time Integration for Smart Manufacturing

Sebastian Behrendt * **Finn Bail** * **Martin Benfer** * **Gisela Lanza** *

* *wbk Institute of Production Science, Karlsruhe Institute of Technology (KIT), Kaiserstraße 12, 76131 Karlsruhe, Germany (e-mail: sebastian.behrendt@kit.edu)*

Abstract: Interoperability and data integration are significant challenges in production planning and control (PPC) due to heterogeneous data formats and fragmented software systems. This paper introduces aas-middleware, an open-source software system leveraging Asset Administration Shells (AAS) to enable interoperable, real-time integration for smart manufacturing. By realizing service-orientation, aas-middleware addresses data heterogeneity and system integration challenges, facilitating automation and information orchestration in PPC. This research details the middleware's architecture and showcases its application and evaluation in a modular assembly station, demonstrating its ability to bridge IT and OT systems effectively in real-time and to orchestrate complex planning tasks automatically.

Copyright © 2025 The Authors. This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)

Keywords: Asset Administration Shell (AAS), Middleware, Interoperability, Data Integration, Smart Manufacturing, Production Planning and Control (PPC), Service-Oriented Architecture (SOA)

1. INTRODUCTION

Volatile market conditions and changing customer requirements demand that production systems adapt efficiently to changes without significant planning efforts or downtime (Mack et al., 2016). Although digitization shows the potential to enable rapid adaptations of a production system, it comes with the cost of managing the increased complexity of software systems due to a multitude of services, interfaces, and data sources (Monostori, 2014). To maximize the benefits of digitization in manufacturing and avoid additional IT and planning costs, a scalable approach is needed that addresses the complexity of digitalized production systems, makes them manageable, and enables automation in production planning and control (PPC).

Smart production planning and control (SPPC) is a research field focused on improving the efficiency and effectiveness of PPC by considering methods and technologies of Industry 4.0. SPPC's main drivers, as identified by Bueno et al. (2020) in their systematic review, are digitization, integration of software systems, and automation in PPC. Despite significant advances in digitization to overcome previous technological limitations with solutions such as OPC UA or the Asset Administration Shell (AAS), PPC processes still face significant challenges in interoperability and automation of information flow (Napoleone et al., 2020; Qu et al., 2019). Key issues include data heterogeneity, as various systems (ERP, MES, shop-floor control) employ diverse formats (e.g., JSON, XML, CSV) and custom schemas, complicating integration. Additionally, many legacy systems and PPC software tools are not integrated and remain isolated due to proprietary data formats and communication protocols. This fragmentation contrasts with the vision of Industry 4.0: an interoperable information network without technical barriers between assets and software (Trunzer et al., 2019). Lastly, PPC software tools are often not configurable via interfaces other than a user interface. Thus, substantial manual input is required to use PPC software tools, limiting the potential for automation, integration, and reusability across different production setups (Bueno et al., 2020).

A suitable software architecture enabling data integration and interoperability is foundational for SPPC. Lee et al. (2015) propose a layered software architecture concept that enables interoperability by structuring integration in multiple layers, each concentrating on specific aspects such as data exchange and transformation. A frequent method to apply such an architecture for SPPC is by integrating services with a unified domain data model (Liu et al., 2020). Similarly, simplifying and scaling integration of enterprise applications has been explored by disassembling them into distinct services (Wang et al., 2020). Alternatively, dedicated parsers have been used to integrate information into other systems (Biesinger et al., 2019).

However, current approaches often fail to fully address these challenges. For instance, while established SCADA platforms or MES systems provide extensive connectivity, they are typically proprietary, costly, and lack native support for the AAS, hindering semantic interoperability. Other solutions often rely on extensive, custom-

⋆ Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1574 – 471687386

2405-8963 Copyright © 2025 The Authors. This is an open access article under the CC BY-NC-ND license.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2025.12.315

coded integrations, which are brittle, difficult to scale, and require significant domain expertise for each new component. A unified framework for interoperability and information exchange across all PPC services with real-time synchronization and handling of data heterogeneity, especially when involving different data formats, schemas, and communication protocols, is missing.

To address these research gaps and build on our previous conceptual work presented in Behrendt et al. (2023), we introduce *aas-middleware* (Behrendt, 2025) [1], an open-source software system designed to support interoperability and integration of planning software through service-oriented information flow orchestration for PPC. It leverages AAS while maintaining compatibility with other commonly used IIoT and IT standards. The rest of this paper details the design and architecture of aas-middleware (Section 2), demonstrates and evaluates its application on a modular assembly station to realize SPPC (Section 3), and concludes with a summary and outlook (Section 4).

2. ARCHITECTURE AND DESIGN

The design of *aas-middleware* aims to provide a foundation for SPPC by enabling automated and continuous information flow between software systems and manufacturing equipment. To realize this goal, the following requirements were considered:

- Enable interoperability between services and manufacturing equipment.
- Resolve data heterogeneity by supporting systematic data integration.
- Allow for complex real-time orchestration of information flows.
- Utilize principles of service-oriented architecture (SOA) for modularity, scalability, and flexibility.
- Manage information flow for PPC with minimal effort and required expert knowledge.

In general, a middleware is a software that connects and facilitates communication between different applications or systems, enabling them to work together and share data effectively. In the past, the integration capabilities of middlewares were mostly limited to technical integration without consideration of semantic interoperability. This limitations is resolved by aas-middleware for PPC in industrial environments, enabling continuous information flow between software systems and IoT components within an industrial information network, as envisioned in Industry 4.0. Consequently, aas-middleware realizes seamless technical, syntactical and semantic integration of the industrial communication standards OPC UA and AAS, aligning with proposals by standardization organizations (Drath et al., 2023). Furthermore, it supports communication standards commonly used in IT, such as JSON Schema, OpenAPI, REST, and GraphQL, demonstrating its potential in IT/OT integration scenarios.

The software architecture of aas-middleware, visualized in Figure 1, is based on a modular design that allows defined, yet extensible and flexible interaction with data sources

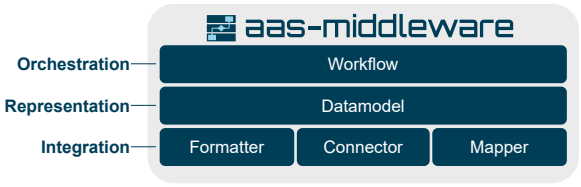

Fig. 1. Software architecture of aas-middleware based on a modular approach consisting of integration, representation and orchestration of information.

and sinks, following the concepts of Integration, Representation, and Orchestration (Behrendt et al., 2023; Lee et al., 2015). The architecture consists of five core building blocks: Datamodel, Connector, Formatter, Mapper, and Workflow.

The **Datamodel** forms the basis for data representation, allowing the specification of data schemas (e.g., using JSON Schema, AAS Templates) and serving as a container for instances of these schemas. By providing functionality for efficient data management, including querying (e.g., via JSONPath) and validation, the Datamodel provides a defined foundation for data integration tasks.

The integration framework used in aas-middleware to create interoperability and resolve data integration problems relies on a generic 3-step approach based on Connectors, Formatters, and Mappers. This integration approach provides a structured framework that guides data integration projects while offering the flexibility needed for complex industrial integrations.

A **Connector** handles technical data integration by enabling communication with data sources and sinks via interfaces to various communication protocols, such as OPC UA, MQTT, HTTP (REST), WebSocket, and direct interfaces like Robot Teach Pendant Ethernet (RTDE). For real-time capabilities, aas-middleware can be deployed in real-time operating systems, utilizing protocols like RTDE and OPC UA PubSub to enable real-time IoT control.

Formatters are responsible for changing the notation of data retrieved or sent via Connectors, thus covering notational data integration. Most notably, aas-middleware supports the transformation between object-oriented schemas/instances (JSON Schema, OpenAPI, GraphQL) and representations conforming to the AAS Meta Model (Industrial Digital Twin Association (IDTA), 2023), facilitating the use of AAS as a canonical data model.

Mappers address conceptual data integration by allowing the flexible definition and use of data transformation procedures to integrate different schemas. Mappers translate data structures between different contexts (e.g., from an OPC UA structure to an AAS submodel structure, or to the input format of a planning service).

Workflows orchestrate the information flow by connecting Datamodels, Connectors, Formatters, and Mappers in defined sequences. This enables automation workflows that integrate different data sources and sinks with heterogeneous communication protocols, data formats, and data structures. Workflows provide a flexible foundation for automation as they support various execution poli-

[1] The open-source project is available at: `https://github.com/sdm4fzi/aas_middleware`

cies (manual trigger, event-based, periodic execution), can contain error-handling logic, and allow for reusability and composition into more complex orchestration logic.

The components of aas-middleware can be defined and configured in two ways: programmatically using a Python Software Development Kit (SDK) or via a declarative REST API. The SDK offers full flexibility for defining components with custom code, while the REST API allows dynamic configuration and execution. SDK-defined components can be containerized and exposed via automatically generated REST APIs for deployment. The service-orientation and container-based architecture of aas-middleware inherently supports scalability in practice.

aas-middleware allows specifying a persistence connector for a Datamodel, acting as the ground truth. Other connectors linked to this Datamodel can be automatically synchronized with the persistence, ensuring data consistency across the network. Upon deployment, aas-middleware automatically creates CRUD (Create, Read, Update, Delete) APIs (REST and GraphQL) for interaction with its managed Datamodels, alongside endpoints for executing Connectors, Formatters, Mappers, and Workflows.

3. DEMONSTRATION AND EVALUATION

This section evaluates how aas-middleware manages to tackle the challenges of interoperability and real-time synchronization by application in a use case.

Use Case Setup

The use case for evaluation features a modular assembly station, whose architecture is conceptually shown in Figure 2. The physical realization of this station, used for our evaluation, is depicted in the photograph within the same figure. The use case demonstrates the capabilities of aas-middleware in complex industrial integration and automated workflow orchestration scenarios relevant to SPPC.

The station comprises of a handling robot and various process modules equipped with universal hardware and electrical interfaces that facilitate their flexible relocation. To detect the current configuration, RFID sensors are positioned at potential module locations, identifying each module along with its precise position and orientation. Building on this adaptable hardware setup, the use case objective is to enable flexible operation by dynamically generating and executing assembly sequences. These sequences should be derived based on a given high-level process list and the station's current, RFID-detected layout.

A key enabler for the automatic adaptation is the AAS provided by each station module, which details its specific capabilities, processes, handling points, and communication interfaces (e.g., OPC UA or RTDE).

The approach for flexible operation involves automatically planning the required robot movements for material handling with a Robot Planning Service (RPS) and the orchestration of commands for process modules, gripper, and robot. Crucially, the AAS self-descriptions allow the system, orchestrated by aas-middleware, to autonomously deduce these necessary handling steps and interaction details without manual intervention, directly realizing the goal of adaptable automation.

Middleware Implementation

In this use case, aas-middleware serves as the central integration and orchestration hub by implementing the following core components.

The **Datamodel** maintains a real-time digital representation of the station's state, including module configurations and operational statuses. It utilizes a standardized data model – the SDM reference model (Ellwein et al., 2023) – represented as AAS instances stored persistently on a BaSyx server. This standardized representation is key to achieving semantic interoperability.

The **Integration** components bridges the gap between diverse systems:

- **Connectors** handle technical communication with OT components (Robot via RTDE, Process Modules and RFID sensors via OPC UA) and IT services (UI, RPS, 3D Visualization and AAS Server via REST). This directly addresses protocol heterogeneity.
- **Formatters** manage notational transformations, converting between JSON (common in IT/UI), OPC UA variables and the AAS format used for persistence and module descriptions, tackling data format heterogeneity.
- **Mappers** resolves conceptual heterogeneity by translating data structures between the context of AAS self-descriptions, the internal state model, and the specific input/output requirements of external services like the RPS.

Workflows automate the information flow and process logic of the assembly station:

- **Update station configuration:** Uses RFID Connectors to identify modules and their positions or orientations, updating the central Datamodel (AAS instances).
- **Plan process sequence:** Receives a high-level target process sequence (e.g., from the UI). It queries the Datamodel (containing AAS information) to identify suitable process modules and their locations. It determines necessary intermediate handling steps (robot movements) based on module interface specifications (from AAS). For each handling step, it invokes the external RPS (via REST Connector, using Mappers for data translation) providing the current station configuration and task details and the RPS returns a suitable robot trajectory. The workflow combines these trajectories with commands for process module activation (via OPC UA Connector) and robot gripper activation (via RTDE) into a complete, executable sequence stored within the middleware. This exemplifies automated, context-aware planning.
- **Execute process sequence:** Executes the process sequence, sending commands in the correct order to the Robot (via RTDE Connector), Robot gripper (via RTDE Connector) and Process Modules (via OPC UA Connector). It uses synchronization mechanisms (e.g., OPC UA boolean flags) to ensure proper handshakes and timing between robot actions and module operations, crucial for reliable execution.

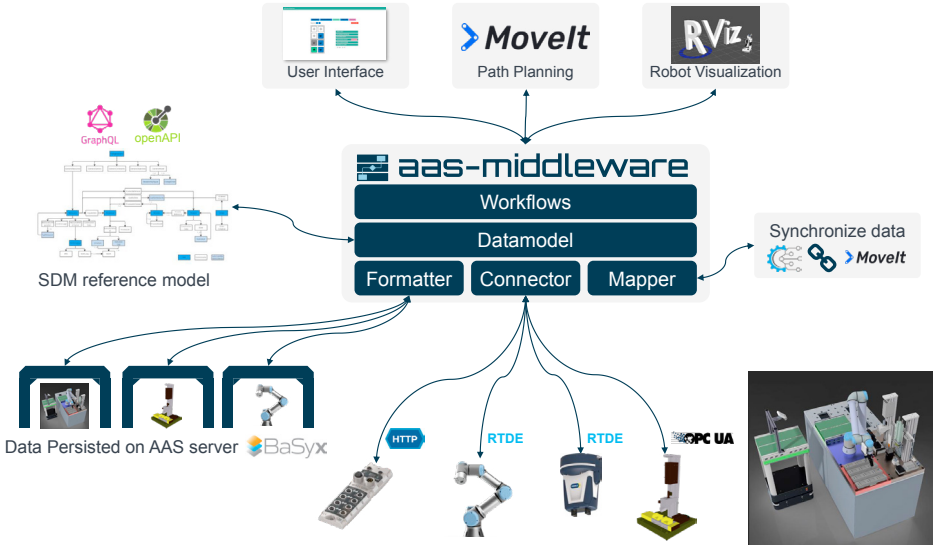

Fig. 2. Conceptual overview of the modular automated disassembly station showing connections managed by aas-middleware.

The reliance on AAS self-descriptions is fundamental, enabling the middleware, in tandem with the planning service, to automate the understanding of module capabilities and interfaces, thus facilitating dynamic planning and execution without manual control logic reconfiguration.

Evaluation

The performance and behavior of aas-middleware were assessed using runtime data from two operational scenarios. The primary objectives were to: (1) demonstrate effective orchestration involving multiple, heterogeneous IT and OT components, showcasing real-time interaction and synchronization; and (2) quantify communication characteristics (frequency, latency, message size) to evaluate its ability to bridge the IT/OT gap and handle demanding planning tasks.

The event timelines provide insight into the dynamic orchestration capabilities. Figure 3 illustrates a scenario involving replanning between two process sequences. It shows, how aas-middleware starts at first with updating the station configuration, visible by the BaSyx and OPC UA interactions at the beginning. Afterwards aas-middleware initiates the RPS multiple times, receiving the robot trajectories. Afterwards aas-middleware receives the execution trigger three times from the UI. In each execution, aas-middleware coordinates the Robot, Gripper, and OPC UA modules for execution. After the first process sequence, a second one is provided from the UI which is planned and also executed three times. However, in the execution, we see two OPC UA interactions now. The second OPC UA interaction shows the synchronization mechanism using OPC UA Boolean variables for handshaking, ensuring precise timing between robot movements and module actions. This highlights the system's adaptability and the middleware's central role in managing dynamic changes and orchestrating interaction of various heterogeneous system components.

Figure 4 details planning and execution of a three-step process sequence, emphasizing the possibility for more complex process sequences. Both timelines validate the middleware's capacity to execute complex, dynamically generated plans across heterogeneous resources and protocols, leveraging AAS self-descriptions for automated interaction and fulfilling the need for robust real-time orchestration of production resources.

Analysis of interaction frequency (Figure 5) reveals a high volume of exchanges managed by the middleware, with the Robot dominating. This reflects the robot's pivotal role in physical execution and underscores the middleware's continuous real-time coordination with station components, including triggering process modules via OPC UA signals.

Examining communication characteristics – latency (Figure 6) and message size (Figure 7) – reveals distinct patterns confirming effective IT/OT integration. OT communication (Robot via RTDE, Process Modules via OPC UA) exhibits very low latency, essential for real-time control and synchronization, with small message payloads (status updates, triggers). It should be noted here, that the interaction latencies of Gripper and Robot cover not only the trigger but also the execution, hence the values in the range of seconds. However, the latencies of OPC UA interactions with a mean below 9 milliseconds shows the feasibility for fast and real-time synchronization between IT and OT. In contrast, IT service interactions (RPS, BaSyx) show higher latencies, acceptable for less time-critical tasks, and often involve larger payloads (configuration data, complex plans). This clear differentiation demonstrates aas-middleware's proficiency as a bridge, effectively managing the disparate communication requirements inherent in smart manufacturing environments.

In summary, the demonstration and evaluation validate that aas-middleware effectively addresses key challenges outlined earlier. Data heterogeneity (formats, protocols, schemas) is managed through the modular Integration

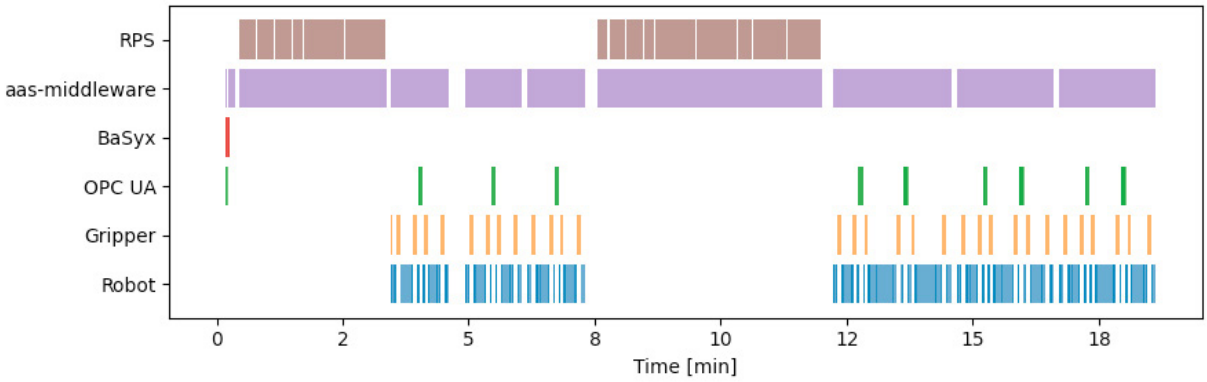

Fig. 3. Event timeline for an planning and execution sequence involving replanning between process steps.

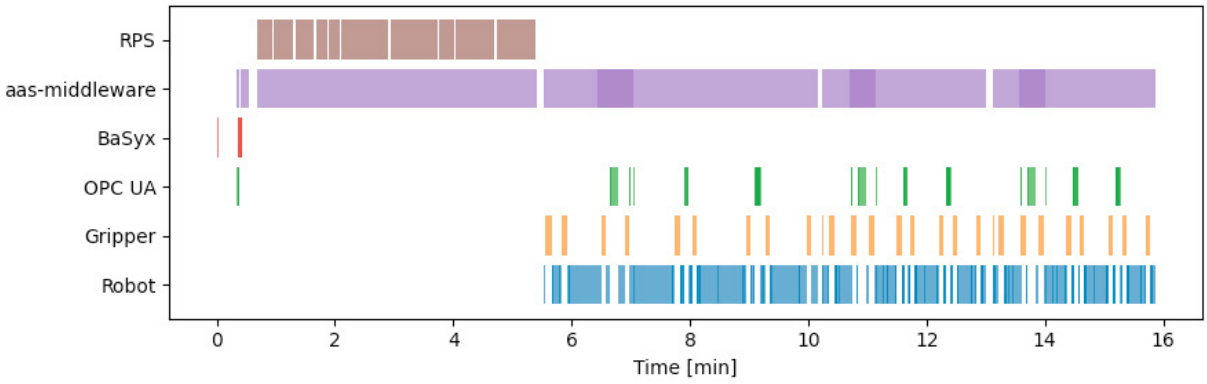

Fig. 4. Event timeline for the planning and execution of a complex three-step process sequence.

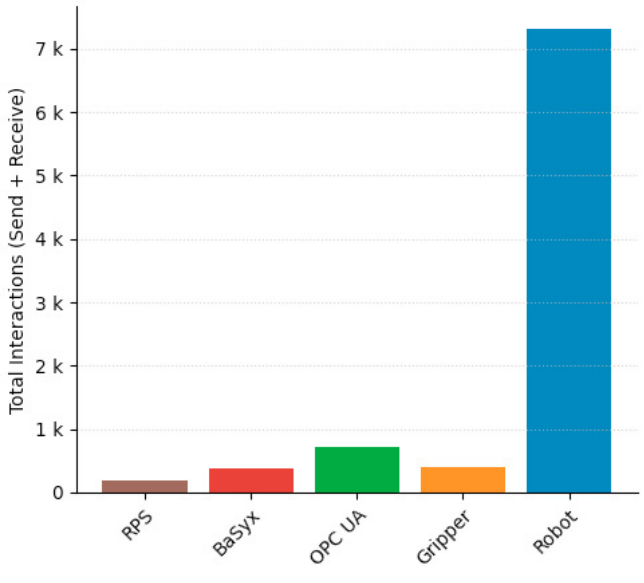

Fig. 5. Number of Interactions per Resource managed by aas-middleware during the evaluated period.

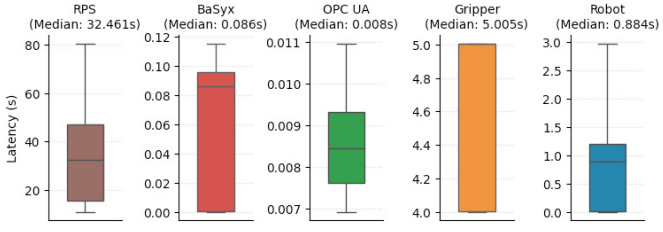

Fig. 6. Distribution of interaction latencies (send/receive round-trip) for different resources.

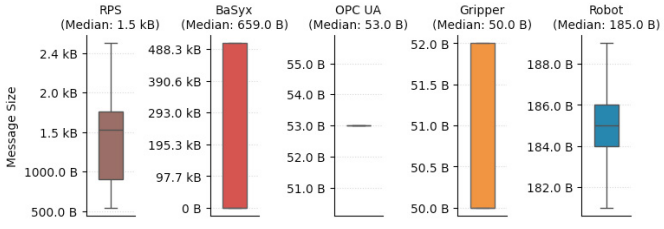

Fig. 7. Distribution of message sizes (sent data) for different resources.

layer (Connectors, Formatters, Mappers). System fragmentation is overcome by providing a unified orchestration platform. Lack of automation in planning and execution is addressed by leveraging AAS self-descriptions within automated Workflows. Real-time synchronization needs are met through low-latency OT communication and explicit coordination mechanisms. The ability to handle both rapid OT interactions and complex IT service calls confirms its role as an effective IT/OT integration bridge.

4. CONCLUSION AND OUTLOOK

This paper introduced aas-middleware, an open-source software system designed to address persistent challenges of interoperability, data integration, and automation within modern Production Planning and Control (PPC). Confronting issues like heterogeneous data formats, fragmented systems, and the need for real-time IT/OT integration, aas-middleware employs a service-oriented architecture (SOA) centered around the AAS. Its modular design, based on Representation (Datamodel), Integration (Connectors, Formatters, Mappers), and Orchestration (Workflows), provides a scalable and flexible solution for seamless communication across diverse industrial and IT protocols.

The practical applicability and effectiveness of aas-middleware were demonstrated in a modular assembly station use case. The evaluation confirmed its ability to successfully integrate disparate OT components (robot, PLCs via OPC UA) and IT services (planning, persistence), manage automatically dynamic reconfiguration using AAS self-descriptions and workflow orhcestrations, and effectively bridge the distinct communication demands of real-time control and higher-level service interactions. The results, evidenced by interaction patterns, latency distributions, and event timelines, underscore its capability to fulfill the critical requirements for building adaptable smart manufacturing systems. By leveraging AAS, it facilitates the automation required for Industry 4.0 scenarios, reducing manual configuration effort and enabling dynamic responses to changing production needs.

While aas-middleware offers a powerful approach, its current practical deployment benefits most from environments with readily available digital interfaces and, ideally, AAS descriptions. Recognizing this, future work will prioritize enhancing its utility in brownfield scenarios where digitization may be limited, exploring methods for semi-automatic generation of interfaces or AAS models. Furthermore, research into integrating Large Language Models (LLMs) to assist in the creation of complex Mappers via schema matching holds potential for significantly reducing the manual effort often associated with data integration.

Overall, aas-middleware constitutes a significant contribution towards realizing truly interoperable and automated smart manufacturing. It provides a robust, flexible, and open source toolset for integrating and orchestrating the complex interplay of systems required for the next generation of intelligent and adaptive PPC solutions, paving the way for more efficient and resilient production environments.

ACKNOWLEDGEMENTS

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1574 – 471687386.

REFERENCES

Behrendt, S. (2025). aas_middleware. Zenodo. doi:10.5281/zenodo.12786362. Version 0.2.6.

Behrendt, S., Stamer, F., May, M.C., and Lanza, G. (2023). Towards a service-oriented architecture for production planning and control: A comprehensive review and novel approach. In D. Herberger (ed.), *Proceedings of the 5th Conference on Production Systems and Logistics (CPSL-2)*, 255–267. publish-Ing., Stellenbosch, South Africa. doi:10.15488/15271.

Biesinger, F., Meike, D., Kraß, B., and Weyrich, M. (2019). A digital twin for production planning based on cyber-physical systems: A case study for a cyber-physical system-based creation of a digital twin. *Procedia CIRP*, 79, 355–360.

Bueno, A., Filho, M.G., and Frank, A.G. (2020). Smart production planning and control in the industry 4.0 context: A systematic literature review. *Computers & Industrial Engineering*, 149, 106774.

Drath, R., Mosch, C., Stefan Hoppe, Andreas Faath, Barnstedt Erich, Bernd Fiebiger, and Schlögl Wolfgang (2023). Interoperabilit mit der verwaltungsschale, opc ua und automationml: Zielbild und handlungsempfehlungen f industrielle interoperabilit. URL `https://opcfoundation.org/wp-content/uploads/2023/04/Diskussionspapier-Zielbild-und-Handlungsempfehlungen-fur-industrielle-Interoperabilitat-5.3-protected.pdf`.

Ellwein, C., Neumann, R., and Verl, A. (2023). Software-defined manufacturing: Data representation. *Procedia CIRP*, 118, 360–365. doi:10.1016/j.procir.2023.06.062. 16th CIRP Conference on Intelligent Computation in Manufacturing Engineering.

Industrial Digital Twin Association (IDTA) (2023). Asset administration shell specification - part 1: Metamodel. URL `https://www.industrialdigitaltwin.org/`. Version 1.0RC03, IDTA Number: 01001-3-0.

Lee, J., Bagheri, B., and Kao, H.A. (2015). A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing Letters*, 3, 18–23.

Liu, C., Jiang, P., and Jiang, W. (2020). Web-based digital twin modeling and remote control of cyber-physical production systems. *Robotics and Computer-Integrated Manufacturing*, 64, 101956.

Mack, O., Khare, A., Krämer, A., and Burgartz, T. (eds.) (2016). *Managing in a VUCA world*. Springer, Cham, Heidelberg, New York, Dordrecht, London.

Monostori, L. (2014). Cyber-physical production systems: Roots, expectations and r&d challenges. *Procedia CIRP*, 17, 9–13.

Napoleone, A., Macchi, M., and Pozzetti, A. (2020). A review on the characteristics of cyber-physical systems for the future smart factories. *Journal of Manufacturing Systems*, 54, 305–335.

Qu, Y.J., Ming, X.G., Liu, Z.W., Zhang, X.Y., and Hou, Z.T. (2019). Smart manufacturing systems: state of the art and future trends. *The International Journal of Advanced Manufacturing Technology*, 103, 3751–3768.

Trunzer, E., Calà, A., Leitão, P., Gepp, M., Kinghorst, J., Lüder, A., Schauerte, H., Reifferscheid, M., and Vogel-Heuser, B. (2019). System architectures for industrie 4.0 applications: Derivation of a generic architecture proposal. *Production Engineering*, 13, 247–257.

Wang, J., Xu, C., Zhang, J., Bao, J., and Zhong, R. (2020). A collaborative architecture of the industrial internet platform for manufacturing systems. *Robotics and Computer-Integrated Manufacturing*, 61, 101854.