

A MODULAR BEHAVIOR TREE ENGINE FOR SMART MANUFACTURING: AN AAS-DRIVEN FRAMEWORK FOR AUTONOMOUS SKILL EXECUTION

 **Mahdi Rezapour***

Innovative Factory Systems (IFS)
German Research Center for Artificial Intelligence (DFKI)
Trippstadter Str. 122, 67663 Kaiserslautern
mahdi.rezapour@dfki.de

Abdallah Darwish

Department of Electrical and Computer Engineering
RPTU University Kaiserslautern-Landau
Gottlieb-Daimler-Straße, 67663 Kaiserslautern
abdallah.darwish@edu.rptu.de

 **Achim Wagner**

Innovative Factory Systems (IFS)
German Research Center for Artificial Intelligence (DFKI)
Trippstadter Str. 122, 67663 Kaiserslautern
achim.wagner@dfki.de

 **Daniel Görges**

Department of Electrical and Computer Engineering
RPTU University Kaiserslautern-Landau
Gottlieb-Daimler-Straße, 67663 Kaiserslautern
goerges@eit.uni-kl.de

 **Martin Ruskowski**

Innovative Factory Systems (IFS), German Research Center for Artificial Intelligence (DFKI)
Trippstadter Str. 122, 67663 Kaiserslautern
Machine Tools and Control Systems (WSKL), RPTU University Kaiserslautern-Landau
Gottlieb-Daimler-Straße, 67663 Kaiserslautern
martin.ruskowski@dfki.de

ABSTRACT

The shift toward adaptive production ecosystems has increased the demand for automation frameworks capable of coordinating complex skills across heterogeneous manufacturing assets. Behavior Trees (BTs) offer strong potential for orchestrating robotic and cyber-physical systems. A unified notion of behavior would further enhance their industrial applicability. In parallel, the Asset Administration Shell (AAS) is emerging as the core digital-twin representation of Industry 4.0, providing structured and interoperable asset descriptions. Yet executable logic is seldom integrated into AAS structures, creating a gap between semantic descriptions and operational skill execution.

This work investigates how behaviors can be represented through BTs and deployed via the AAS to enable decentralized and autonomous skill execution. To address this, we introduce a Modular BT Engine that retrieves BT models stored in AAS submodels and autonomously executes skills through an Information Technology/ Operational Technology (IT/OT) interface. The framework is implemented and evaluated on the SmartFactory Kaiserslautern (SFKL) testbed, demonstrating flexible orchestration and adaptive skill execution in modular manufacturing environments.

Keywords Behavior Trees · Smart Manufacturing · Cyber-Physical Systems · Asset Administration Shell · Distributed and Modular Control

1 Introduction

Modern manufacturing systems are transitioning toward highly adaptive production environments where digitalization, cyber-physical connectivity, and autonomous decision processes form the basis of operational capability (1). In

*Corresponding author. Tel.: +49 631 20575 7611

these ecosystems, production tasks necessitate representations that enable flexibility, composability, and transparent execution across various equipment. Behavior Trees (BTs) provide a structured framework for operational logic and are progressively applied to coordinate complex tasks with precision and modularity. Their hierarchical composition makes them well suited for industrial settings that must adapt quickly to changing conditions (2).

Despite these advantages, the industrial adoption of behavior-based automation remains limited, as task logic is often isolated, restricting reuse and coordinated execution across devices (3; 4). Furthermore, mechanisms for linking digital models to executable task logic are still fragmented. Digital twin technologies, including the AAS (5), provide structured descriptions of capabilities and configurations, yet explicit integration with executable behavioral models is still underdeveloped (6). This gap limits system synchronization as operational requirements change.

Motivated by this challenge, the objective of this work is to develop a coherent foundation for representing and executing industrial behaviors by combining the expressive structure of BTs with AAS. The central question concerns how behaviors can be defined in a manner that is precise enough for automation, yet general enough to support distributed execution across diverse production assets. Establishing such a foundation requires a mechanism through which behavior models can be stored, discovered, and deployed in a consistent way, while maintaining compatibility with existing Industry 4.0 infrastructures.

To address this need, we introduce a modular BT Engine that interprets BT models stored in dedicated AAS submodels and executes them across IT and OT layers. The engine links digital representations to physical actions and enables coordinated execution among assets. Embedding BT models into the AAS creates a unified mechanism for managing behavioral knowledge and supports runtime adaptability and interoperability. This integration establishes a scalable automation foundation in which behaviors can evolve with production requirements while maintaining transparency and modularity.

2 State of the Art

Modern industrial automation increasingly relies on skill-based and service-oriented paradigms to address high variability and rapid innovation cycles (7; 8; 9; 10). SFKL¹ provides reference architectures for modular and reconfigurable production via standardized interfaces, and further operationalize RAMI 4.0 (11) by using AAS submodels to represent capabilities, skills, and interaction protocols for Cyber-Physical Production Modules (CPPMs) (1; 12). This approach is further advanced by the Production Level 4 demonstrator (13). Within this ecosystem, Multi-Agent Systems (MAS) support distributed decision-making. The MAS4AI framework presented in (14) models resources, products, and services as agents that rely on the AAS as an information backbone for coordination in agile, human-centered manufacturing.

In parallel to these architectural developments, BTs have emerged as a powerful formalism for modular and hierarchical control. Starting from work by Ögren et. al., BTs have been shown to generalize and unify classical hybrid control schemes such as finite-state machines, subsumption architectures, and sequential behavior compositions, while preserving desirable properties like reactivity and analyzability (15). The monograph BTs in Robotics and AI provides a rigorous treatment of BT semantics, modularity, and design patterns, and has become a reference point for applying BTs to robotics and autonomous systems (3). Recent surveys further document the breadth of BT applications in robotics, games, and AI, and highlight their suitability for combining deliberative, reactive, and safety-relevant behaviors within a single hierarchical structure (16).

In manufacturing research, BTs are shown to serve as a supervisory layer for composing skills into modular behaviors in reconfigurable cyber-physical production modules (RCPPMs) (8). Further work explores the integration of BTs into industrial controllers, showing how behavior logic can bridge between high-level planning and PLC-level control execution (2). Within smart factories and multi-agent ecosystems, BTs are now employed alongside AAS-based representations to support autonomous manufacturing processes. Sidorenko et al. illustrate how agents can use AAS information to trigger BT-based decisions in adaptive and human-centered production scenarios (14). A recent advancement extends BT usage beyond machine control toward product lifecycle intelligence. Rezapour et al. present a method for embedding executable BTs into Digital Product Passports (DPP) through the AAS, thereby formalizing Circular Economy R-strategies in a machine-interpretable and transferable format (17).

Overall, the literature demonstrates a converging trend in which Digital Twins (DT), skill-based architectures, and BTs provide complementary structures for integrating Robot handling, transport and assembly tasks into modular and autonomous industrial control. This convergence creates fertile ground for a unified behavior execution engine driven by AAS representations and BT-based behavior formalization.

¹<https://www.smartfactory.de>

3 Methodology

The methodology defines how behaviors are conceptualized, parameterized, and executed through a BT engine integrated with the AAS. Building on the definition of skills as modular, parameterizable production functions for reconfigurable CPPMs (8), we define *behavior* as the executable control logic that enables these skills to be performed. Behaviors describe goal-directed actions and reactions that can be composed and parameterized, and they are realized through BTs, which provide a transparent and structured mechanism for sequencing, selection, and coordination. In the remainder of this paper, we adopt this behavior definition as the foundation for our execution framework. The detailed formulation of that has been planned as future work.

A central step in the methodology is retrieving configuration parameters from the AAS. The AAS (which serves as the DT for each asset) provides access to its operational context through dedicated submodels. Using a BaSyx-based² AAS server together with a lightweight client interface, the engine reads the required parameters, including target poses, process settings, and skill-specific options. Through this mechanism, behaviors are parameterized at runtime and remain consistent with the current production conditions. After the parameters are collected, the engine loads the BT model stored within the corresponding AAS submodel. During initialization, node attributes are linked to the values, execution options are applied, and communication interfaces are prepared. This step transforms the BT from a static structural description into an executable behavior specification that reflects the system context. The BT engine diagram is shown in Figure 1.

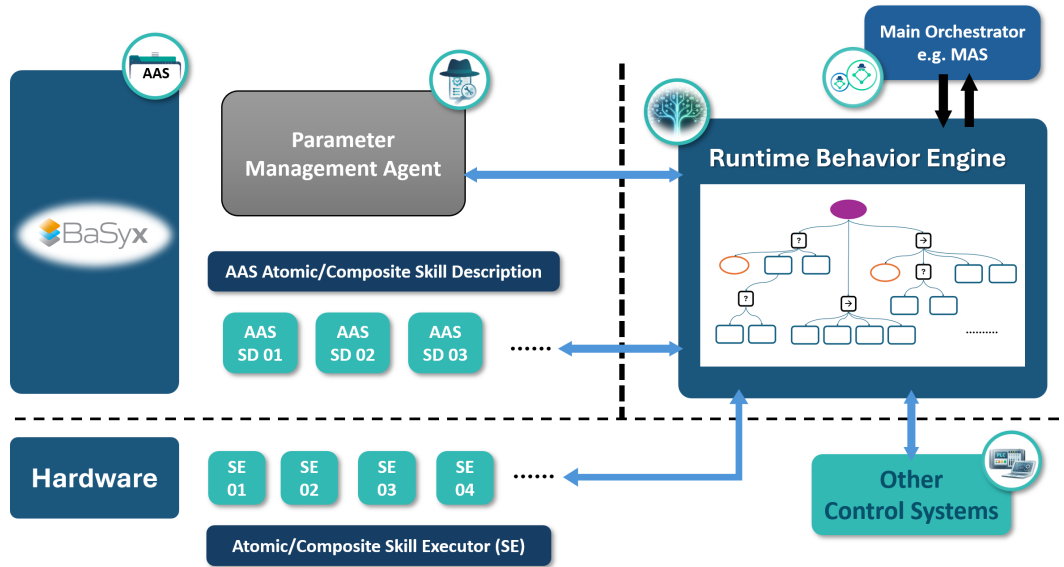


Figure 1: BT engine diagram

The execution layer integrates two complementary control pathways. For robotic assets, the engine provides atomic behaviors implemented through the RTDE protocol. These behaviors correspond to minimal action units, for example joint/linear motion, or gripping operations. Higher-level composite behaviors, including pick-and-place, are created as reusable BT subtrees that combine several atomic behaviors into complete executable skills. For non-robotic or pre-developed skills that follow the SFKL OPC UA information model, the engine uses a general OPC UA client. Through this client, the engine can call, coordinate, and monitor skills offered by external production modules while keeping them within the same BT execution framework. In this way, both RTDE-based robot behaviors and OPC UA-based skills are managed under a unified behavior engine, which ensures consistent orchestration across different assets. Figure 2 illustrates this architectural integration.

During execution, the engine continuously updates the status of the active BT within the AAS. This feedback loop ensures traceability, enables online monitoring, and provides higher-level systems such as multi-agent platforms or MES components with timely information about execution progress. The proposed BT Engine is integrated into the SFKL reference architecture at the IT/OT layer. More precisely, the engine extends the Smart Machine Logic Controller App, which traditionally encapsulates atomic and composite skills. The BT Engine communicates with the MES, the Smart Machine Interface, and the OT layer directly.

²<https://basyx.org/>

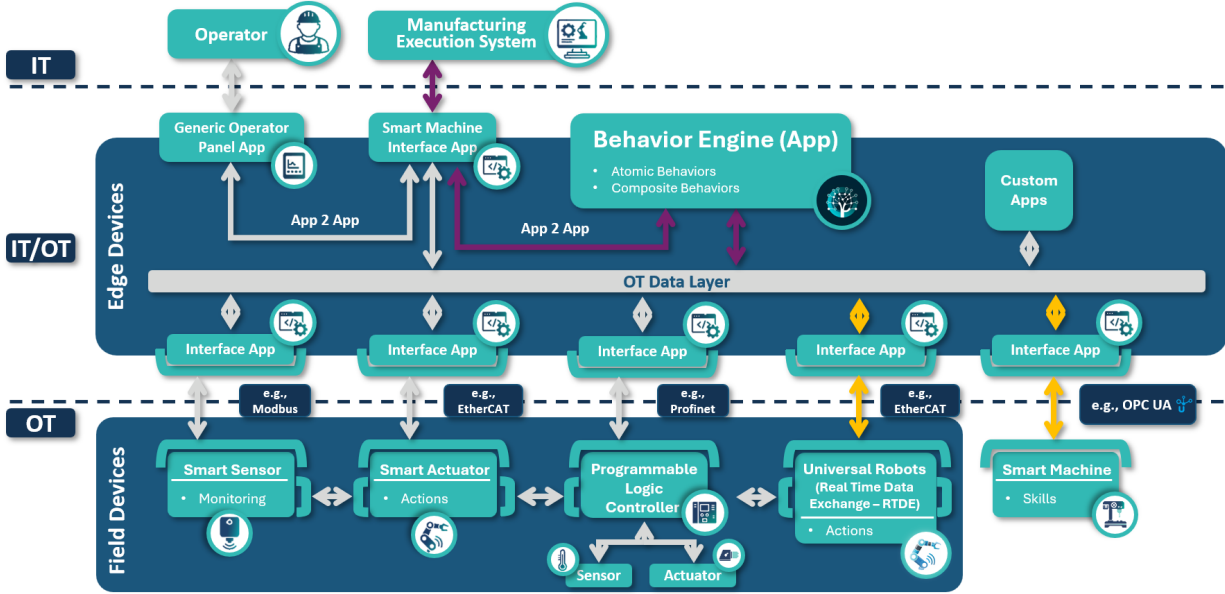


Figure 2: SmartFactory architecture with BT engine

4 Implementation and Results

Our implementation uses the AAS as a source of configuration information, similar to its use in the automated robot initialization approach of Richard et al. (18). In their work, system configuration and deployment steps are encoded in AAS submodels and executed by a Resource Initializer Agent (RIA). Our approach focuses on BT execution, applies AAS submodels only for configuration and behavior loading. It generates robot interfaces via RTDE, while ROS 2 integration is reserved for future work.

The implemented real scenario in SFKL involves the production of a 3D-printed planetary gearbox, using a Prusa MK4 3D printer. A UR5e robot then handles the part and places it onto a Montratec transport system for quality inspection and subsequent storage. The workflow initiates with the BT invoking the `print_product` skill, which activates the 3D printer via its established OPC UA interface. The tree remains idle until the printer indicates that the post-condition of print completion has been met, thereby ensuring that the subsequent step is initiated only when the product is prepared. Once printing is complete, the UR5e takes over. The BT activates a `PickProduct` and `PlaceProduct` subtree executed through RTDE, using atomic behaviors for motion and gripping that follow standard BT semantics. Before the placement step can proceed, the engine verifies the required pre-condition (an empty transport shuttle) by querying the transport system via its AAS (or OPC UA data). Only when this condition is met does the robot place the part into the shuttle.

Subsequently, the BT activates the shuttle's `TransportTo` skill, resulting in the product being delivered to the quality inspection station. An image is captured, and the evaluation result is recorded in the AAS. Upon product approval, a final skill transfers it to storage. In this setup, the robot performs fundamental behaviors, while all other modules are accessed via pre-established skills, demonstrating how BTs offer an integrated and condition-responsive orchestration framework across IT/OT assets. Figure 3 shows the BT used to coordinate this scenario. This BT structure can also be represented as a property in AAS submodel, as demonstrated by Rezapour et al. in (17).

The BehaviorTree.CPP³ library and Groot2 environment were used to model the BT structure in this implementation. Groot2 makes process planning easier by giving you a graphical drag-and-drop interface. This makes it easy to set up, change, and save BT structures. But to still be able to define a correct and useful BT, you need to know a lot about the field, especially when it comes to coordinating and sequencing processes across all the assets involved.

³<https://www.behaviortree.dev>

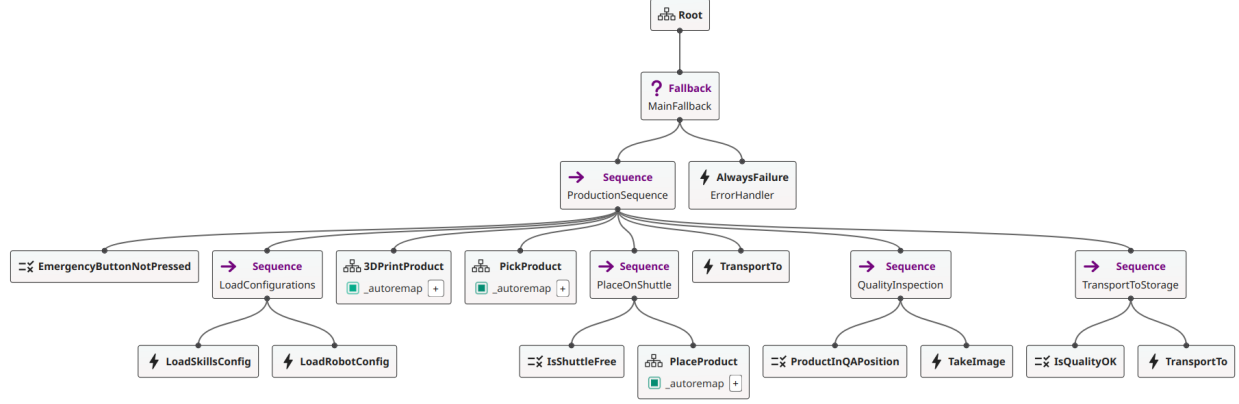


Figure 3: BT model for coordinating the process

5 Discussion and Conclusion

This work focuses on the design and evaluation of a modular BT Engine for adaptive execution of behaviors. The main contribution lies in the execution architecture (Figure 1 and 2), where behaviors are defined as modular BT structures and executed by a dedicated engine that coordinates skills through standardized interfaces, particularly OPC UA. By separating behavior from skill implementation, the BT Engine acts as a flexible control layer responsible for sequencing and decision-making, while concrete actions are handled by interoperable skills. This design supports reuse of behavior models across different production setups, reduces dependency on hardware-specific implementations, and simplifies behavior evolution as production requirements change. Experimental results indicate that modular behaviors can be composed and executed with reduced engineering effort.

At the same time, the approach contains some limitations. Embedding executable structures in the AAS does not, by itself, guarantee correct or efficient execution; it requires expert evaluation of the BT structure and careful integration of it. This implies additional engineering effort to ensure consistency between the BT model, the available skill interfaces. Within this context, the AAS serves as a representation and exchange mechanism rather than an execution environment. Storing BT structure as XML file in AAS submodels enables behavioral logic not only to be documented and transferred but also instantiated by the BT Engine (while execution responsibility remains fully within the engine). This separation keeps the BT Engine as the main authority for execution and makes the AAS a structured way to carry behavior definitions that can be used for deployment.

6 Future Works

An important direction for future efforts is the enhancement of the BT Engine to enable online reconfigurable execution, allowing for the modification of BT structures during runtime without interrupting system functionality. The integration of artificial intelligence techniques for the automated generation and alteration of BT frameworks represents a promising avenue for future investigation. In addition, extending the BT Engine to improve compatibility with existing execution frameworks would further support broader adoption.

7 Acknowledgements

This paper is funded by Resilient and Adaptive Supply Chains for Capability-based Manufacturing as a Service Networks (RAASCEMAN) project which is a EU-RIA Program under the grant agreement No 101138782.

References

- [1] S. Jungbluth, B. Blumhofer, P. Rübel, S. Bergweiler, C. Harms, M. Ruskowski, Smartfactory reference architecture: Standardised connection of IT and OT (Nov. 2025). doi:10.5281/zenodo.17651696.
- [2] A. Sidorenko, M. Rezapour, A. Wagner, M. Ruskowski, Towards using behavior trees in industrial automation controllers., *Procedia CIRP* 130 (2024) 1234–1243.

- [3] M. Colledanchise, P. Ögren, Behavior trees in robotics and AI: An introduction, CRC Press, 2018.
- [4] D. Brandl, Design patterns for flexible manufacturing, ISA, 2006.
- [5] S. Bader, E. Barnstedt, H. Bedenbender, et al., Details of the asset administration shell – part 1: The exchange of information between partners in the value chain of industrie 4.0, Plattform Industrie 4.0, accessed: 2025-01-13 (2022).
URL https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/Details_of_the_Asset_Administration_Shell_Part1_V3.html
- [6] A. Köcher, A. Belyaev, J. Hermann, J. Bock, K. Meixner, M. Volkmann, M. Winter, P. Zimmermann, S. Grimm, C. Diedrich, A reference model for common understanding of capabilities and skills in manufacturing, at-Automatisierungstechnik 71 (2) (2023) 94–104.
- [7] A. Wagner, N. Gafur, A. Sidorenko, P. Pahlevannejad, K. Abuibaid, M. Ruskowski, Skill-based multi-agent control for safe and effective human-robot collaboration, at-Automatisierungstechnik 73 (9) (2025) 679–697.
- [8] A. Sidorenko, A. Wagner, M. Ruskowski, Skills composition framework for reconfigurable cyber-physical production modules, in: 2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, 2024, pp. 1–8.
- [9] K. Dorofeev, S. Bergemann, T. Terzimehić, J. Grothoff, M. Thies, A. Zoitl, Generation of the orchestrator code for skill-based automation systems, in: 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, 2021, pp. 1–8.
- [10] OASIS, Reference model for service oriented architecture, OASIS, accessed: 2025-12-05 (2006).
URL <http://docs.oasis-open.org/soarm/v1.0/soa-rm.pdf>
- [11] DIN SPEC 91345: Reference Architecture Model Industrie 4.0 (RAMI 4.0), German Institute for Standardization (DIN), DIN SPEC 91345 (2016).
URL <https://www.din.de/en>
- [12] J. Hermann, P. Rübel, M. Birtel, F. Mohr, A. Wagner, M. Ruskowski, Self-description of cyber-physical production modules for a product-driven manufacturing system, Procedia Manufacturing 38 (2019) 291–298.
- [13] M. Ruskowski, A. Herget, J. Hermann, W. Motsch, P. Pahlevannejad, A. Sidoreko, S. Bergweiler, A. David, C. Plociennik, J. Popper, K. Sivalingam, A. Wagner, Production bots für production level 4: Skill-basierte Systeme für die Produktion der Zukunft, atp magazin 62 (9) (2020) 62–71. doi:10.17560/atp.v62i9.2505.
- [14] A. Sidorenko, W. Motsch, M. van Bekkum, N. Nikolakis, K. Alexopoulos, A. Wagner, The MAS4AI framework for human-centered agile and smart manufacturing, Frontiers in Artificial Intelligence 6 (2023) 1241522.
- [15] M. Colledanchise, P. Ögren, How behavior trees modularize hybrid control systems and generalize sequential behavior compositions, the subsumption architecture, and decision trees, IEEE Transactions on Robotics 33 (2) (2016) 372–389.
- [16] M. Iovino, E. Scukins, J. Styrod, P. Ögren, C. Smith, A survey of behavior trees in robotics and AI, Robotics and Autonomous Systems 154 (2022) 104096.
- [17] M. Rezapour, C. Plociennik, A. Farrukh, M. Ruskowski, Representing executable circular economy r-strategies using behavior trees embedded in digital product passports, in: Proceedings of the 7th International Conference on Industry of the Future and Smart Manufacturing, 2025.
- [18] P. Richard, A. T. Bernhard, A. Luxenburger, H. Gösling, A. Wagner, Agent-based initialization of autonomous mobile robots: Leveraging the asset administration shell for automated commissioning, in: 2025 IEEE 30th International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, 2025, pp. 1–8.