OPEN INDUSTRY 4.0 ALLIANCE

SOFTWARE DEFINED AUTOMATION

RETHINKING HOW AUTOMATION IS BUILT

SHORT CUT

WHAT

Software Defined Automation means you stop building automation around devices and start building it around software. Usually, machines run with applications that are tightly installed on specific hardware, meaning the software is directly tied to the device it runs on. Now try to connect these machines or use their data across systems, and this tight coupling quickly becomes a limitation.

WHY

To avoid having to change and adapt individual machines every time something evolves, which slows down updates, limits scalability, and makes connecting systems unnecessarily complex.

WHO

For companies, engineers, and decision-makers who are reaching the limits of their current automation systems, especially when changes take too long or systems are difficult to scale and connect.

SDA TEAM @OPEN INDUSTRY 4.0 ALLIANCE

WITH THIS APPROACH, WE REDUCED TIME TO MARKET FROM MONTHS TO JUST ONE TO TWO WEEKS.

DOES IT TAKE MONTHS TO INTRODUCE NEW FUNCTIONALITY?

IS IT DIFFICULT TO CONNECT SYSTEMS AND USE DATA ACROSS THEM?

DOES SCALING YOUR SETUP REQUIRE SIGNIFICANT MANUAL EFFORT?

IF YOU ANSWER ONE OF THE Q'S WITH „YES" - YOU SHOULD CONTINUE READING.

CONTENT

Authors

Laurids Beckhoff | Process Industry Mgmt. | Beckhoff Automation GmbH & Co. KG

Samuel Greising | Co-Founder & Managing Director | FLECS Technologies GmbH

Tobias Schneck | Principal Software Architect | Kubermatic GmbH

Editor

Lucas Wolf | Technical Content Manager | Open Industry 4.0 Alliance

Publisher
Open Industry 4.0 Alliance
Christoph Merian-Ring 12, 4153 Reinach, Switzerland
https://openindustry4.com
info@openindustry4.com

OPEN INDUSTRY 4.0 ALLIANCE

01 | WHAT IS SOFTWARE DEFINED AUTOMATION?

FROM DEVICE-BOUND SYSTEMS TO SOFTWARE-DRIVEN AUTOMATION

Software Defined Automation, or SDA, describes a shift in how automation systems are designed, deployed, and operated. Today, machines are often designed with digital concepts in mind, but implemented in a fragmented way. Each component supplier brings its own digital solution, resulting in a large number of isolated devices and applications within a single system. Instead of a connected environment, this leads to a landscape of disconnected solutions. SDA addresses this by introducing a consistent approach based on three core principles.

1 First, software and hardware are separated. This reduces dependency on specific devices and allows applications to run independently of the underlying infrastructure.

2 Second, software is built and deployed using open and standardized approaches, inspired by concepts established in IT, such as cloud native development. This enables flexible supplier selection and avoids vendor lock-in.

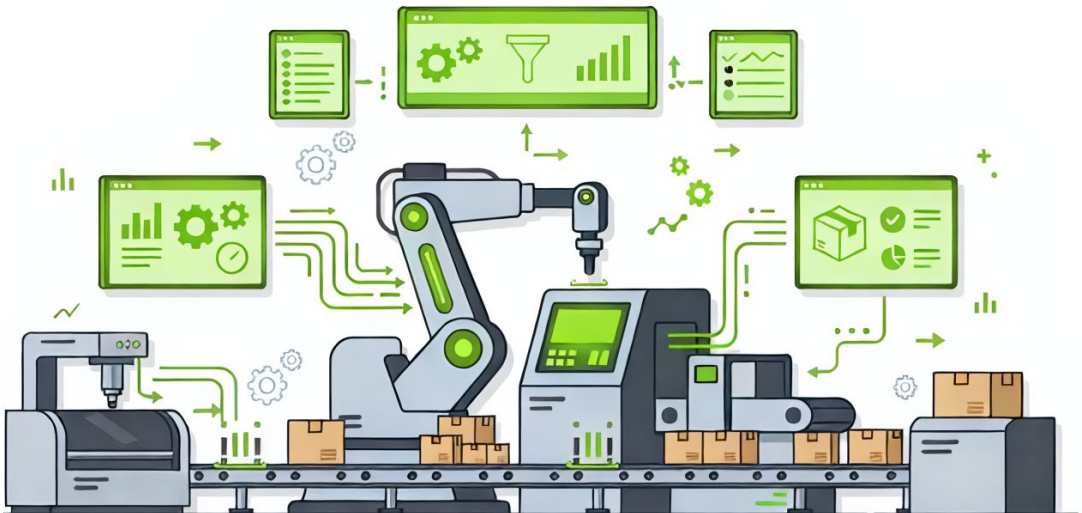
3 Third, software development becomes a central capability. Instead of adapting systems around hardware, value is created through software that can be continuously developed, deployed, and scaled.

The objective is a consistent deployment model across environments. Applications should behave the same way regardless of where they run. This includes development environments, data centers, and industrial devices in production.

This shift changes how automation is approached in practice. Instead of building systems around fixed setups, companies can structure their automation in a way that is easier to adapt, extend, and connect across different environments. Applications are no longer confined to a single machine or system, but can be used consistently wherever they are needed. This creates a foundation where changes can be implemented more directly, systems can be combined more easily, and new requirements can be integrated without having to redesign the entire setup.

TODAY: SOFTWARE IS TIED TO SPECIFIC MACHINES AND HARD TO MOVE.

WITH SDA: SOFTWARE RUNS ACROSS SYSTEMS AND IS USED WHERE NEEDED.



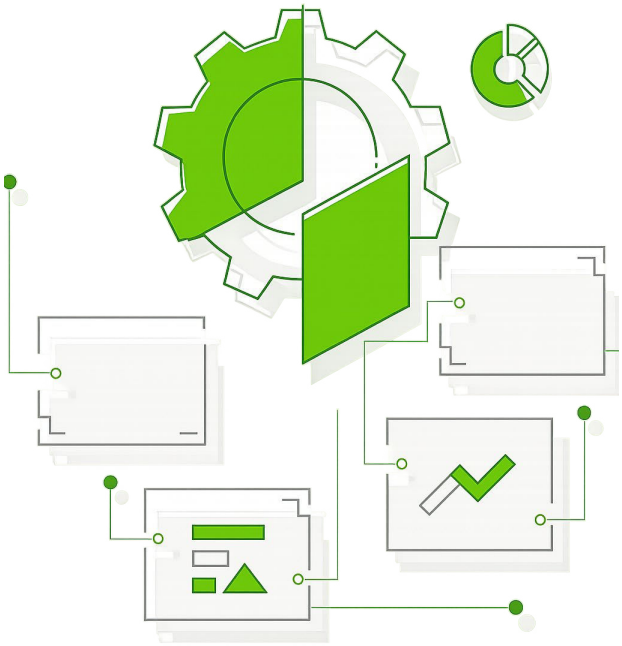
02 | WHY IS SOFTWARE DEFINED AUTOMATION IMPORTANT?

UNDERSTANDING THE LIMITS OF TODAY'S AUTOMATION SETUPS

In current automation environments, complexity doesn't arise from individual components, but from their accumulation and interaction over time. Each additional system introduces its own interfaces, lifecycle, and operational requirements. As these systems grow, integration becomes increasingly difficult, and the effort required to operate and maintain them rises accordingly. What initially appears as incremental improvement leads to a fragmented system landscape that limits flexibility and slows down further development.

This fragmentation has consequences for cost, scalability, and of course, speed. Infrastructure must be maintained for multiple parallel solutions, each requiring updates, monitoring, and specialized expertise. Expanding a system often means replicating existing structures along with their complexity, which slows down rollout across sites and increases operational overhead. At the same time, introducing new functionality requires coordination across multiple systems and environments, resulting in long implementation cycles and delayed availability of new capabilities.

Security and compliance add another layer of complexity. Distributed systems with different technologies and update cycles require continuous coordination to ensure a consistent security level. Managing this across a fragmented landscape increases effort and risk, especially when systems evolve independently. SDA introduces a consistent approach to handle these challenges. By standardizing how applications are being developed, deployed, and operated, it reduces the need for system-specific processes and enables a more predictable and scalable environment. Applications can be handled in a uniform way across their lifecycle, which shortens implementation cycles, reduces operational effort, and improves the ability to scale across different environments. In addition, the integration of IT-based approaches allows companies to extend their available expertise and apply established development and deployment practices within automation.



03 | HOW IS SOFTWARE DEFINED AUTOMATION ACHIEVED?

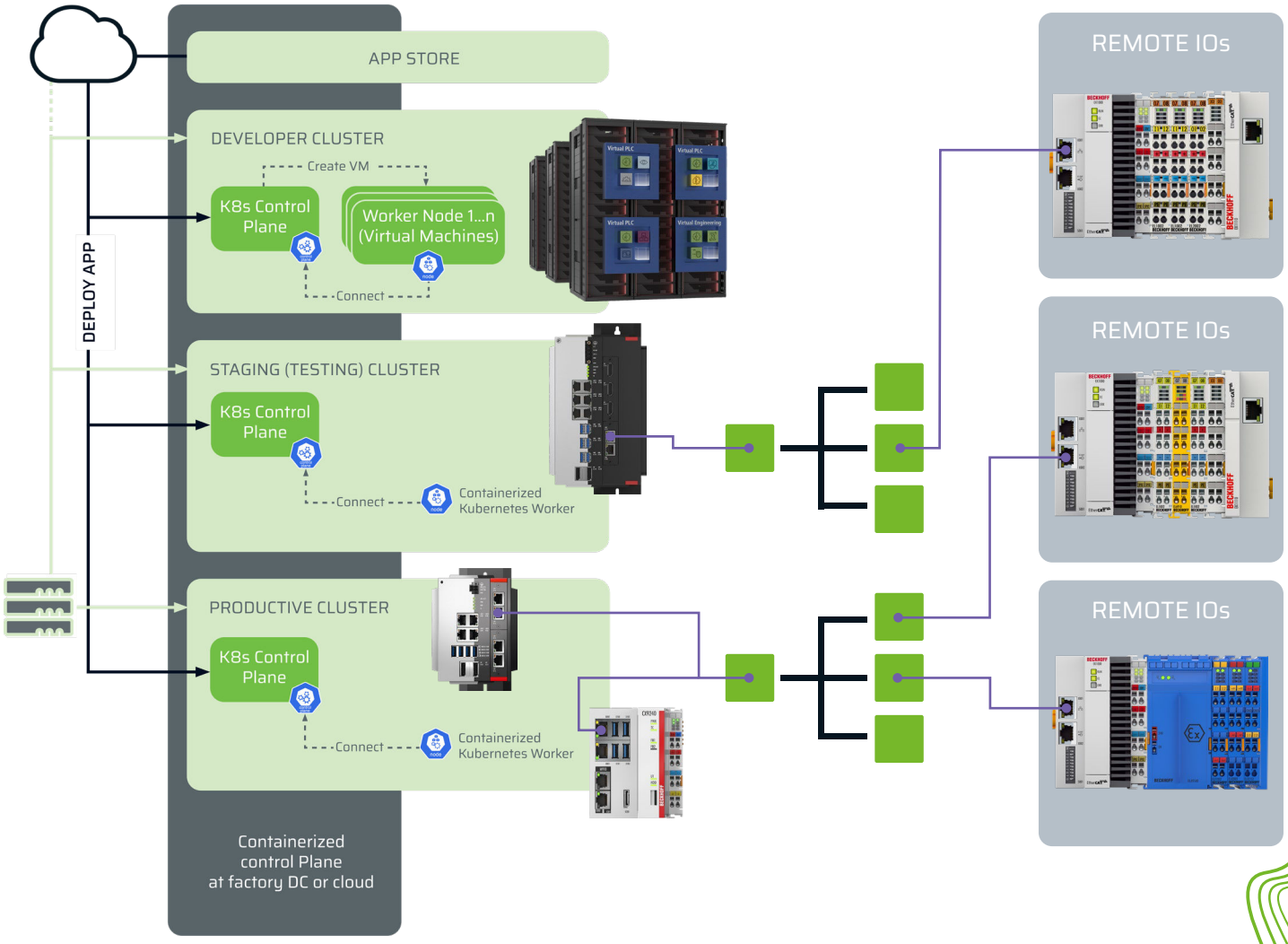
SEPARATING SOFTWARE FROM HARDWARE & STANDARDIZING DEPLOYMENT

SDA is implemented through a variety of combinations of the architectural principles, standardized technologies, & consistent deployment processes. Central element is the use of container-based applications, which allows software to be packaged independently of the used infrastructure. This enables applications to be deployed in a uniform way across different environments without requiring adaptation to specific hardware.

A standardized orchestration layer f.e. manages how these applications are deployed, updated, and operated. Instead of manually installing software on individual systems, applications are distributed through defined processes that can be repeated across environments. This creates a consistent deployment logic that applies from development to production. At the same time, the architecture separates management from execution. A central management layer defines which applications are deployed and how they are configured, while execution takes place on distributed systems such as virtual machines or industrial devices.

Applications move through a structured lifecycle. Development begins in a virtual environment, where functionality can be created and tested without hardware dependencies. In the next stage, applications are connected to real hardware to validate their behavior under realistic conditions.

Finally, they are deployed into production systems, where they run on infrastructure that meets the specific requirements of the use case. This staged approach ensures that applications are validated step by step while maintaining a consistent deployment mechanism.



The infrastructure supports different requirements across environments. Virtual environments enable rapid development and testing, while production systems can provide the necessary performance and real-time capabilities. Communication between systems is handled through controlled interfaces, allowing secure deployment and operation without exposing internal structures unnecessarily. The result is a continuous and automated workflow that connects development, testing, and operation while maintaining flexibility in system design.

04 | IMPLEMENTATION WITHIN THE OPEN INDUSTRY 4.0 ALLIANCE

APPLYING SDA PRINCIPLES IN A REAL INDUSTRIAL ENVIRONMENT

Within the Open Industry 4.0 Alliance, SDA has been implemented through collaborative projects involving multiple companies. Each participant contributed specific components from their portfolio to build a shared architecture that follows the defined principles. The focus was on demonstrating that a consistent deployment model can be applied across different environments and that interoperability between vendors can be achieved in practice.

Boehringer Ingelheim · KUBERMATIC · conplement

BECKHOFF · FLECS · SALZ AUTOMATION

COPADATA · margo

PROJECT INFORMATION? CLICK HERE.

SDA Project 1 · SDA Project 2

The implementation used a container-based approach to package and distribute applications. These applications were deployed across different environments, starting from virtual development systems, moving through testing stages with real hardware, and finally reaching production systems. The same deployment process was applied throughout all stages, which ensured consistency and reduced manual effort.

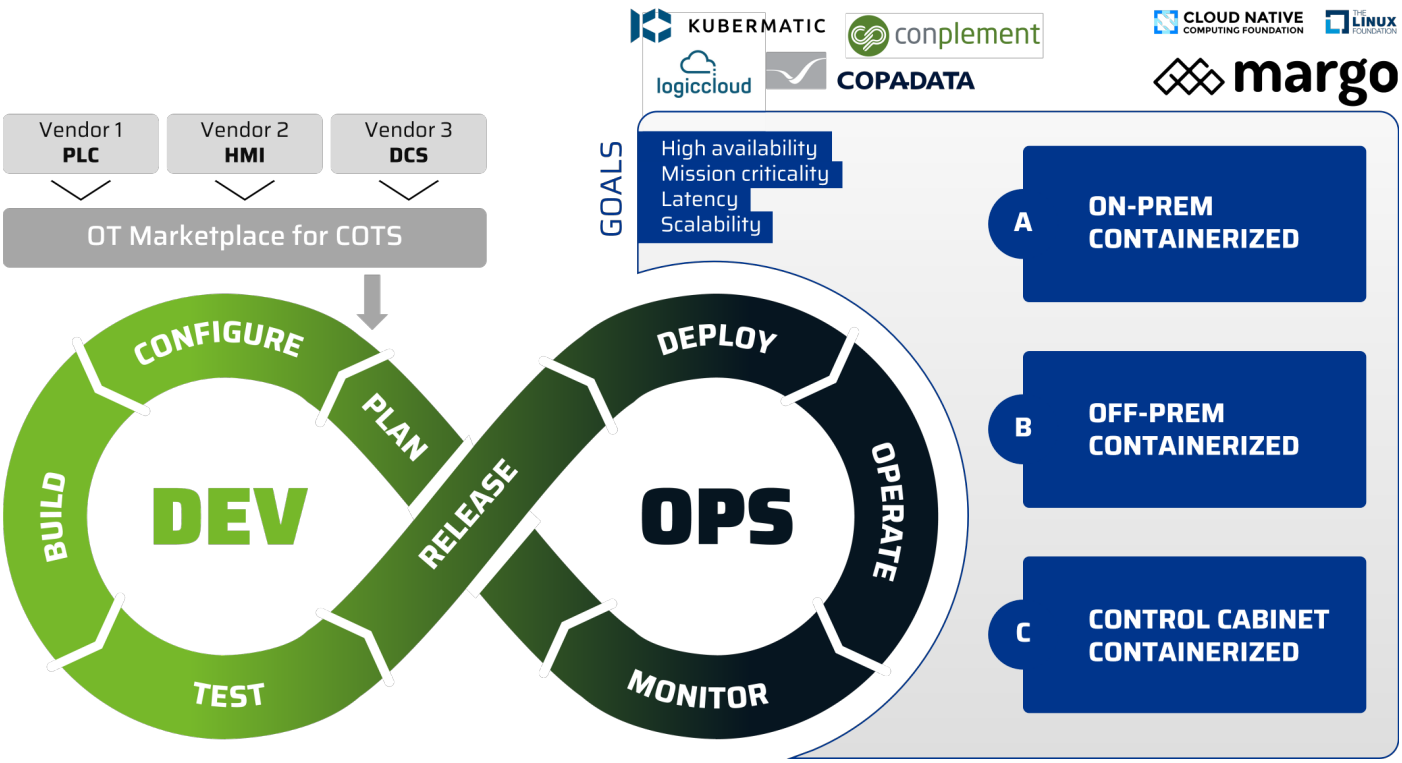
The architecture allowed the integration of hardware and software from different vendors. Control logic, communication components, and even the higher-level applications were combined within a shared environment, while maintaining access to physical input and output systems. At the same time, the setup addressed typical industrial requirements such as system stability and hardware dependencies by placing applications on appropriate systems based on their needs.

This implementation showed that the separation of software and hardware, combined with standardized deployment processes, can be realized using existing technologies. It also demonstrated that collaboration across companies can result in a shared approach that remains flexible enough to accommodate different requirements and solutions.

Building on this implementation, the second phase focused on validating the approach under real conditions and extending it into a more structured setup. The involved companies continued to work on the shared architecture, refining the interaction between components and ensuring that the deployment model holds across different environments and use cases. The objective was no longer just to demonstrate feasibility, but to confirm that the approach can be applied consistently when moving from isolated setups to more integrated scenarios.

This included further developing the workflow from development to production, ensuring that applications could be moved reliably across stages while maintaining consistent behavior. At the same time, the integration of hardware and software from multiple vendors was expanded, showing that interoperability can be maintained even as the system grows in complexity. Particular attention was given to how applications interact with real industrial environments, including hardware constraints and operational requirements.

As a result, the second phase confirms that the SDA approach can be extended beyond initial implementations and applied in a more systematic way. It shows that a shared architecture, combined with consistent deployment processes, can support the transition from isolated solutions to a more connected and scalable automation landscape.



05 | RESULTS AND IMPACT

REDUCING TIME TO MARKET AND IMPROVING SCALABILITY IN PRACTICE

The application of SDA leads to measurable improvements in how automation systems are developed and operated. One of the most visible effects is the reduction in time required to introduce new functionality. Deployment cycles that previously required several months can be shortened to one to two weeks, enabling faster iteration and quicker response to changing requirements.

This acceleration is accompanied by improvements in cost efficiency. Both capital expenditure and operational expenditure can be reduced through standardized deployment, lower system complexity, and more efficient use of infrastructure. Systems no longer need to be built and maintained as isolated units, which decreases the overall effort required for operation and maintenance.

Scalability is also significantly improved. Applications can be deployed across multiple environments without redesign, allowing companies to expand their systems more easily and operate them consistently across locations. This is particularly relevant in distributed production environments where consistency and repeatability are essential.

Operational processes become more efficient as well. Automated deployment reduces manual work, while standardized environments simplify updates and maintenance. In the event of hardware failure, systems can be restored quickly by replacing devices and redeploying applications, without requiring complex reconfiguration.

Overall, SDA changes how systems evolve over time. It creates a foundation where new functionality can be introduced continuously, systems can be adapted more easily, and operational complexity remains under control even as environments grow.

HELLO, WE ARE OPEN

The results also show that this approach requires coordination beyond a single company. Software Defined Automation touches multiple systems, roles, and responsibilities, which makes isolated implementation difficult to sustain. The Open Industry 4.0 Alliance provides a structured environment to address this. Companies work on shared topics, validate approaches in joint setups, and align on practical solutions that can be applied across different environments. For organizations that want to move in this direction, participation allows direct access to existing implementations, technical expertise, and ongoing developments. This supports a more efficient path from initial concepts to operational systems and reduces the need to build solutions independently from scratch.

START WITH OI4

OPEN INDUSTRY 4.0 ALLIANCE

VERSION HISTORY

VERSION	DATE	DESCRIPTION	EDITOR
1.0	June 22nd, 2026	Initial release	Lucas Wolf