# Cloud-based automation architecture for competitive manufacturing systems

C. Faller[1], L. Friedrichsen[1], P. Stenkamp[1]

[1]Department for Automation Engineering,
Campus Velbert-Heiligenhaus, Hochschule Bochum, Germany

Abstract

The competitiveness of future manufacturing systems depends on their flexibility and availability. The development of the last years within the topic of Industry 4.0 have shown that the convergence of the information technology and operational technology (IT-OT-Convergence) is key to ensure this aim. Cyber-Physical-Systems (CPS) and Industrial Internet-of-Things (IIoT) are core technologies to realize the IT-OT-Convergence. This paper shows a future-oriented, decentralized architecture model for CPS based on the synergy between elements of classical automation technology and modern IT paradigms to leverage service-orientation down to the edge-level.

Keywords

Industrie 4.0, Cyber-Physical-Systems, Industrial Internet of Things, Cloud Services, System Theory

## 1 INTRODUCTION

Looking at the developments within automation technology over the last few years, it becomes obvious that the trend is increasingly moving towards individual production (‘batch size one’, cf. e.g. [1]).

Along with the currently ongoing shift from an industrial to post-industrial market in the consumer industry, manufacturing is, in direct consequence, undergoing drastic changes as well.

Whereas previously in a quasi-stable market situation with a more or less guaranteed acceptance of goods the focus in production was on maximizing output volumes while simultaneously minimizing costs, today flexibility and versatility are becoming increasingly important. Manufacturers need to be able to adapt to customer wishes as quickly as possible in order to remain competitive in an oversaturated environment [2].

This shift is, however, accompanied by the need to rethink existing automation technology paradigms such as the traditional automation pyramid in order to meet the requirements imposed by the recent market developments.

In this context, blueprints based on the further evolution of the existing approaches can be found, for example, in the work of the ‘Industrie 4.0 Platform’ in the form of the ‘Reference Architecture Model for Industry 4.0 (RAMI4.0)’ [3] or in the research agenda CPS of the VDI/VDE society of measurement and automation technology [4].

In this paper, however, a more far-reaching approach is presented, which, instead of merely focusing on the further refinement of existing models, synergistically integrates principles of modern software development.

## 2 CONVENTIONAL DESIGN OF AUTOMATION SYSTEMS

Historically, conventional production systems were initially designed with the goal to optimize repetitive, always identical processes. As aforementioned, since the acceptance of goods was guaranteed within the scope of the manufacturer's acquisition potential, priority was given to maximizing the output quantities while at the same time minimizing the cost. Due to the latter, all elements not essential for the production process were streamlined or eliminated.

This resulted in a lean, highly specialized architecture with real-time capabilities as the main focus, as the individual manufacturing steps were highly synchronized to maximize output. This involved the monolithic centralization of the process logic on the field layer to further reduce the reaction times as well as the system’s complexity as the core principle.

This pattern is also reflected in the traditional automation pyramid. Even though today, layers 3 and above are gradually dissolving and becoming increasingly decentralized (as described in [4]), the control layer and below remain largely unchanged. This remains true as well for the improved model as defined in the RAMI4.0 guidelines [3], which adds additional life-cycle and infrastructure perspectives to the automation pyramid (making it a three-dimensional cube) and proposes changes concerning the hierarchical structure, e.g. logically merging the field and the control level, but does not break up the centralized process logic.

The problem with this is, that for a complex and highly dynamic market situation where flexibility and versatility are key to stay competitive [5], such a monolithic architecture is far too rigid. Today, the strong specialisation in the Taylorist industrial model is proving counterproductive in terms of adaptability since the original framework conditions no longer apply. New, alternative approaches are needed.

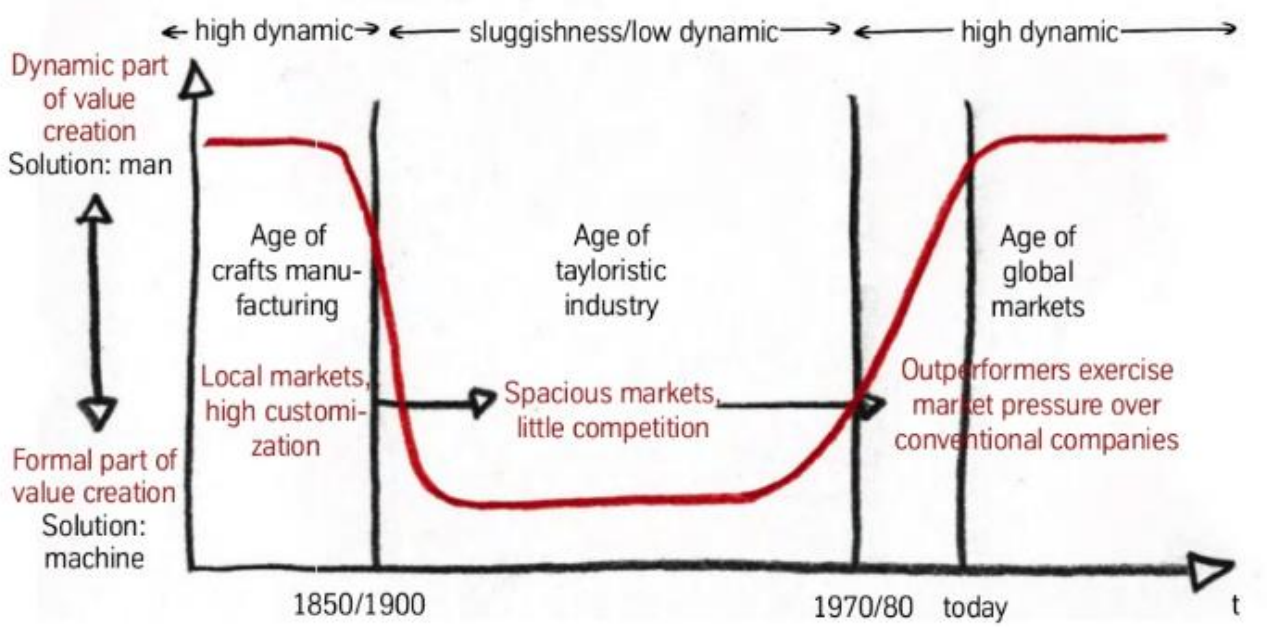



*Figure 2-1 The historical course of market dynamics and the recent rise of complex and dynamic markets [2]*

## 3 DECENTRALIZATION AND MODULARIZATION

One such alternative and sensible approach with regard to system flexibility requirements consists in the modularization of process logic.

This methodology is in line with the procedures established in the IT industry for the development of complex, highly dynamic software systems (cf. e.g. 'Microservices Architecture Pattern', [6]).

Contrary to the widespread assumption in automation technology, decentralization is not necessarily synonymous with a decline in system safety. Although the inherent complexity increases, this is only a consequence of the fact that a system cannot be depicted losslessly by a model of a lower complexity level [5]. This measure does thus not increase the general susceptibility to errors any more than the retention of the existing monolithic design with respect to the changing environment would do. Merely the type of errors changes; meta-level deficiencies are replaced by potential technical deficiencies.

While the former eventually proves to be fatal from a business perspective, the latter is preventable. As modularization is a common procedure in software development, there are already many guidelines that describe how to create fault tolerant distributed systems (cf. e.g. [7]).

Consequently, it comes down to the decision whether to stick to the centralized approach in favor of a lower system complexity and thus to hazard shortcomings on the strategic side, or whether to adapt to the shifting market situation and accept the inherent, yet avoidable technical pitfalls associated with decentralization of process logic.

If one decides in favor of modularization, it is furthermore intuitive to detach oneself from the coventional models coupling hierarchical layers to physical equivalents as they reach their limits with regard to service-oriented architectures (as illustrated in [4]). Consequently, this means to move away from the traditional automation pyramid to a perspective that focusses mainly on organizational dependencies.

In the following, one approach to achieve decentralization of process logic is shown using the development of a testbed for innovative architectures in automation technology as an example.

## 4 CASE STUDY: CONCEPT OF A CLOUD-BASED ARCHITECTURE FOR MANUFACTURING SYSTEMS WITH CASCADED PROCESS LOGIC

### 4.1 Background

The aforementioned testbed was developed as part of the research into future-oriented architectures for competitive manufacturing in the department for automation engineering at the Bochum University of Applied Sciences, Campus Velbert-Heiligenhaus. It is intended to serve as a basis for investigating the effects of service-oriented architecture in automation technology under realistic conditions.

### 4.2 Architectural design

Following the previous argumentation, a modular, decentralized and logically cascaded approach was chosen as the basis for the implementation.

As illustrated in figure 4-1, the organizational layout is based on the infrastructure layers known from software development at its core. Contrary to the

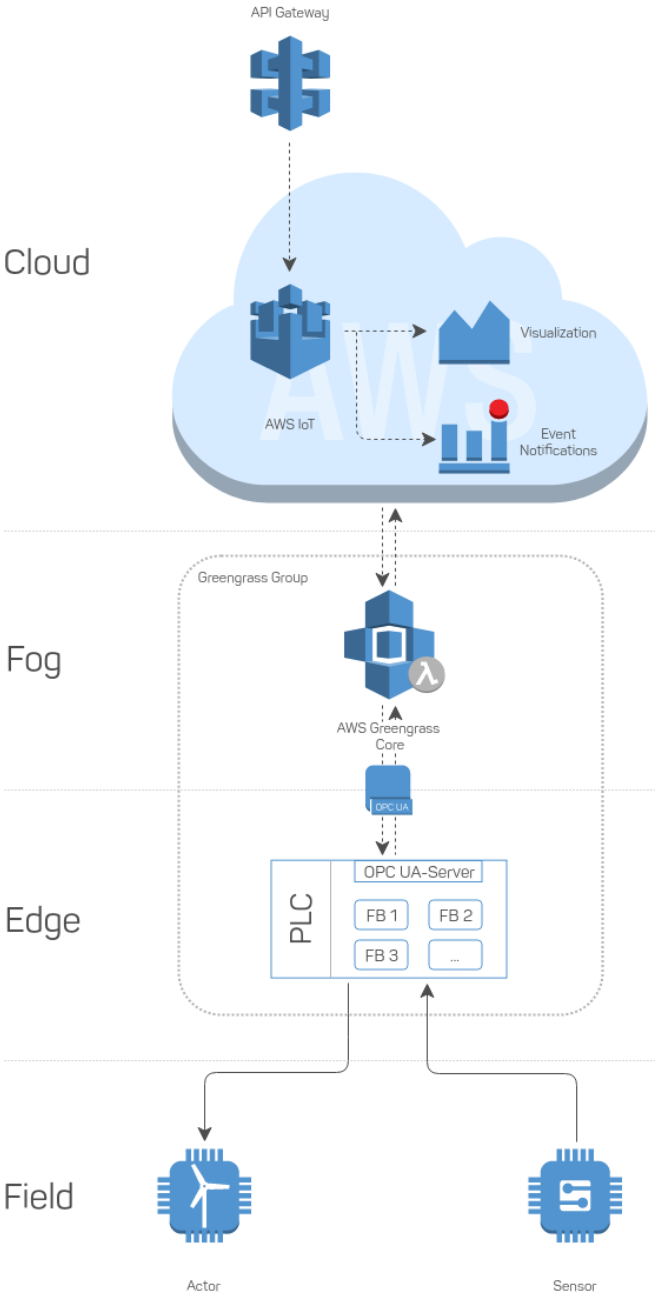



*Figure 4-1 Conceptual architecture with layers*

traditional automation pyramid, this model does not imply immanent hierarchical links between the individual levels.

In order to achieve the greatest possible flexibility with regard to the extensibility of the testbed and to facilitate the subsequent integration of external services, a cloud solution was chosen for the back end. In this this case, the selection process favored **Amazon Web Service (AWS)**, since its IoT-stack supports the chosen decentralized approach with **AWS IoT Core** and **AWS Greengrass** well across all levels of infrastructure.

The latter serves to leverage part of the service landscape from the cloud to the fog layer, thus not only providing offline capabilities and drastically reducing latencies – which is crucial in terms of systems safety – but also enabling a service-oriented programming paradigm in a local environment. Furthermore, it offers an integrated deployment mechanism for software updates. This is an important feature as it allows for the application of subsequent security patches as well as for fast validation of effects from measures.

On the field layer, it was decided to use the **Modular Production System** by Festo Didactic as it is well suited for a testbed application due to its modularity and extensibility.

To ensure maximum flexibility with regard to the later adaptability and modifiability of the application, no conventional PLC running proprietary software was used. Instead, the setup is controlled by a Linux based programmable logic controller (PLC), specifically a **RevPi Core 3** by KUNBUS, providing full, unhindered access to all underlying processes.

As shown in figure 4-1, an OPC UA server based on the open source project **open62541** was set up on the PLC as an interface between the edge and the fog layer. OPC UA is well suited for this task since it already supports asynchronous and event-based communication by default, which is a core requirement for a service-oriented architecture.

Special to this implementation is, that it was carried out right from the start according to the 'API first' principle by making use of the 'UA Methods' defined in the OPC UA standard. The result is a so-called 'API as a product' [8], an abstraction layer against which can be developed from the outside and which, in contrast to a pure mirroring of the PLC's input and output states on the server, allows for elements to be facaded towards the outside. Thus, for example, all real-time critical program sections can run locally encapsulated with guaranteed safe execution, while the actual process flow is defined via the API at the fog or cloud layer. This enables effective cascading of the logic across the various infrastructure levels. From the perspective of a manufacturing system, such a design is highly advantageous, as it significantly increases the immanent flexibility and safety in a service-oriented architecture.

In the following, the OPC UA information model as well as the cloud-based services and the communication interface between the fog and the edge layer shall be discussed in more detail.

## 4.3 Implementation

### *4.3.1 OPC UA information model*

Building on the aforementioned ‘API first’ principle, the key requirements for the creation of the OPC UA information model were as follows: First, all IOs of the PLC shall be accessible through open62541 while also being easily configurable without changing the code of the server or editing configuration files on the PLC. Secondly, all IOs as well as the states of core modules like indicators, sensors and actuators have to be provisioned with getter- and setter-methods to provide a basic level of API.

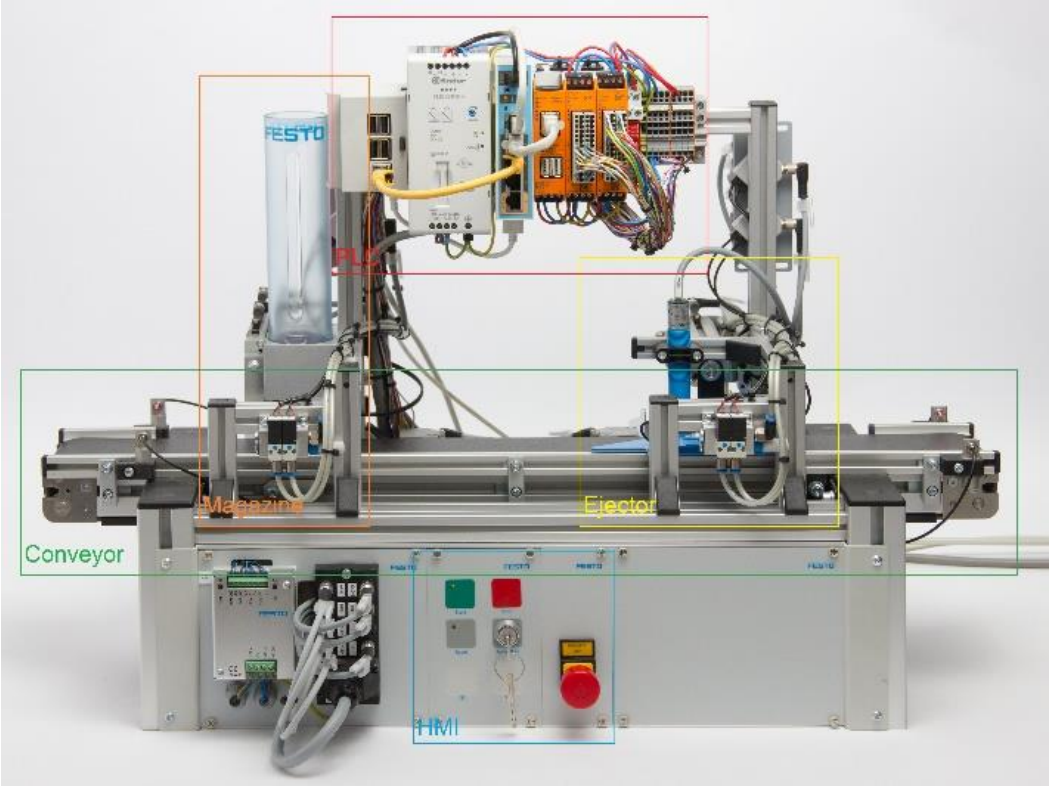



*Figure 4-3 The testbed, build on the Festo Modular Production System and a RevolutionPi 3 with additional IO*

Individual IOs shall not be writable directly by a client, but rather be linked with the core modules they belong to. These core modules can perform basic functions which they provide in form of methods to clients via OPC UA. Picture a pneumatic cylinder which is controlled by two valves and which position can be monitored via two sensors. Not only does the information model have to represent the position of the cylinder; it also has to provide methods to command a change of position. Therefore, the PLC needs to know what the current position is, which inputs the sensors are connected to and which outputs must be set to a given state to move the cylinder to the target position.

This information shall be provided by the model itself to allow for easy adaptation and modification of the system by application engineers while sparing them the time and effort changing and testing the program itself would require. This is achieved by utilization of OPC properties containing all the necessary information. These properties are evaluated at runtime by the server. If necessary and permitted, these can be edited at runtime via the same tools used to monitor the system (e.g. a regular OPC UA client), allowing for a reconfiguration of individual modules without shutting down the overlying system.

At the current state of implementation, the information model only provides methods for the IO and core modules as well as for safety critical elements. Higher level functions, for example the process logic required to transform a pneumatic cylinder and a number of sensors into a system module like a part magazine, are implemented at the fog and cloud layer.

### *4.3.2 Cloud-based services and communication architecture*

As mentioned before, the core cloud services used in the testbed application are AWS IoT Core and AWS Greengrass.

The former provides the classic features of an IoT platform like secure communication, data processing and routing, and device management. In the implementation process, it was used for the creation of a so called ‘Digital Twin’ (cf. e.g. [3]) of the testbed, meaning a virtual representation of the physical object, as well as an interface to other cloud services.

For this purpose, the submodules of the MPS were recreated in the device management section as virtual devices. Through the constant synchronization with Greengrass and under usage of the AWS Shadow functionality, they were then configured to be updated automatically when a state change on the physical testbed is registered via the OPC UA server. As a result, the state of the digital twin is always consistent with the ‘real’ MPS. Other cloud services can access the virtual devices and thus obtain the live data from the manufacturing process.

The other way around, it is also possible to define a so called ‘desired’ state of the digital twin through the Shadow protocol. This will cause a gateway application – an AWS Lambda function hosted locally on the Greengrass device (cf. figure 4-1) – to be notified which then again resolves the specified condition into a function call and invokes the respective method on the OPC UA server. Thus a fully bidirectional synchronization between the virtual representation and the physical testbed is achieved.

Since the whole communication process is event based, this harmonizes well with a service-oriented architecture.

For demonstration purposes, the application was additionally extended by a service-based automatic mode for the MPS as well as by a voice control option based on Amazon Alexa. The former is hosted locally on the Greengrass device alongside the OPC UA facing gateway in order to ensure fail-safe operation even in the event of a network failure while the latter is located in the cloud and serves as an example for the integration of further services into the described architectural design.

### 4.4 Conclusions

Already during the implementation process, several positive aspects of the chosen architectural layout became apparent: The high degree of modularization allowed for a parallel and agile development of core software components.

Due to the 'API first' design approach and the resulting facading of functional entities like the core modules of the MPS, changes in the process logic of the PLC could be made without requiring an adaptation in the information model or the gateway functions and vice versa. One disadvantage of this approach however is the redundant curation of the information model on the edge and the fog layer. These are currently modelled in different formats, although this could possibly be resolved by the use of an additional abstraction layer or common format in the likes of AutomationML (cf. [9]) in the future.

Another big advantage of this highly modular design is the minimal effort required to integrate additional services on all layers from edge to cloud into the application. The deployment of these services on the AWS infrastructure as well as the local Greengrass instance has proven itself as very versatile and intuitive.

Overall, the immanent synergies between automation technology and IT became apparent and proved to be very advantageous for the development process. For example, the utilities provided by the latter greatly facilitated the state synchronization between the 'real' testbed and the digital twin on the technical level.

Concluding, the chosen approach has proven very promising with respect to the initially formulated requirements. The decentralisation of process logic significantly enhanced the flexibility of the system. In general, the adaptability increased drastically due to the service-oriented architecture. Choosing a cloud-stack as the backend for the application augmented this aspect even further since the extensive service landscape available makes it possible to deploy and evaluate changes in a fast and simple way.

Even though, while this work provides first findings regarding the feasibility of such a design approach in the domain of industrial automation, it does this only in the scale of a small and independent system. While this is sufficient in the context of an initial case study, a quantitative analysis of the scalability as well as potential risks of this approach in a production environment require further research.

Nevertheless, one thing that this case study shows for sure is that in the future the job profile of the automation technician will change drastically. With eventually increasing system complexity, the number of elements from conventional software development in this area will rise as well. In the long run, at least a partly merging of both domains, also referred to as IT-OT-Convergence, is inevitable. On the other hand, some traditional components of automation technology like the automation pyramid will most likely be abandoned or evolved as their initial framework conditions no longer apply.

From a corporate perspective, this means that employees have to be actively educated and trained to enable them adapt to the shifting circumstances. However, as assimilating such disruptive changes is a very time-consuming process, it is advisable to start as soon as possible.

## 5 REFERENCES


[1] NN: Implementation Strategy Industrie 4.0 – Report on the results of the Industrie 4.0 platform, Pages 26-28, Bitkom, VDMA, ZVEI, 2015

[2] Pfläging, Nils, Herrmann, Silke, Vollmer, Lars, Carvalho, Valérya: Organize for Complexity, BetaCodex Network White Paper No. 12, BetaCodex Network, 2012

[3] NN: Implementation Strategy Industrie 4.0 – Report on the results of the Industrie 4.0 platform, Pages 40-70, Bitkom, VDMA, ZVEI, 2015

[4] NN: Cyber-Physical Systems: Chancen und Nutzen aus Sicht der Automation, VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik, 2013

[5] Snowden, Davd J., Boone, Mary E.: A Leader's Framework for Decision Making, Harvard Business Review, 2007

[6] Wolff, Eberhard: Microservices – Grundlagen flexibler Softwarearchitektur, dpunkt.verlag, Heidelberg, Baden-Wuerttemberg, 2016

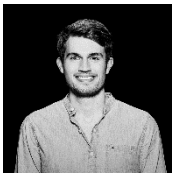
[7] Hanmer, Robert S.: Patterns for fault tolerant software, John Wiley & Sons Ltd, England, 2007

[8] NN: Technology Radar Nov '16, Page 5, ThoughtWorks Inc., 2016

[9] Faller, Clemens., Höftmann, Max.: Service-oriented communication model for Cyber-Physical-Production-Systems, 11th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 2017


## 6 BIOGRAPHY

Prof. Dr.-Ing. Clemens Faller studied Mechanical Engineering at the Ruhr Universität Bochum and researched for his doctorate in the field of industrial communication standards. In 2007 Mr. Faller changed to the French company Schneider Electric. Since April 2013 he is professor for automation technology and researches in the field of digitization of production, industrial IT and industrial communication technologies.



Lukas Friedrichsen obtained his bachelor's degree in Engineering (Mechatronics and Information Technology) from the Bochum University of Applied Sciences as part of a cooperative dual-study programme with integrated apprenticeship. He is currently working as a research assistant in the Department of Automation Technology while simultaneously completing his master's degree.

Philipp Stenkamp holds a bachelor's degree in Engineering (Mechatronics and Information Technology) from the Bochum University of Applied Sciences. He is currently working as a research assistant in the Department of Automation Technology while pursuing his master's degree.